Research Article

Applying Deep Neural Networks over Homomorphic Encrypted Medical Data

Anamaria Vizitiu [iD],[1,2] **Cosmin Ioan Niţă,**[1,2] **Andrei Puiu,**[1,2] **Constantin Suciu,**[1,2] **and Lucian Mihai Itu** [iD][1,2]

[1]*Department of Automation and Information Technology, Transilvania University of Braşov, Braşov, Romania*
[2]*Corporate Technology, Siemens SRL, Braşov, Romania*

Correspondence should be addressed to Anamaria Vizitiu; anamaria.vizitiu91@gmail.com

In recent years, powered by state-of-the-art achievements in a broad range of areas, machine learning has received considerable attention from the healthcare sector. Despite their ability to provide solutions within personalized medicine, strict regulations on the confidentiality of patient health information have in many cases hindered the adoption of deep learning-based solutions in clinical workflows. To allow for the processing of sensitive health information without disclosing the underlying data, we propose a solution based on fully homomorphic encryption (FHE). The considered encryption scheme, MORE (Matrix Operation for Randomization or Encryption), enables the computations within a neural network model to be directly performed on floating point data with a relatively small computational overhead. We consider the well-known MNIST digit recognition problem to evaluate the feasibility of the proposed method and show that performance does not decrease when deep learning is applied on MORE homomorphic data. To further evaluate the suitability of the method for healthcare applications, we first train a model on encrypted data to estimate the outputs of a whole-body circulation (WBC) hemodynamic model and then provide a solution for classifying encrypted X-ray coronary angiography medical images. The findings highlight the potential of the proposed privacy-preserving deep learning methods to outperform existing approaches by providing, within a reasonable amount of time, results equivalent to those achieved by unencrypted models. Lastly, we discuss the security implications of the encryption scheme and show that while the considered cryptosystem promotes efficiency and utility at a lower security level, it is still applicable in certain practical use cases.

1. Introduction

Over the recent years, machine learning algorithms, with emphasis on deep neural networks, have delivered remarkable solutions for personalized medicine, enabling customized diagnosis, treatment, and prevention [1]. Since deep neural networks are entirely data-driven systems that can learn explicitly from past experiences, they are commonly used as a way to integrate the knowledge and experience of medical experts into solutions for computer-aided detection (CADe).

To deliver results that are sufficiently reliable to be considered in clinical routines, machine learning-based solutions have to heavily rely on available medical data records [2]. However, as patient health information has some of the highest privacy requirements among all data types associated with an individual, its usability is greatly hindered. Moreover, given the fact that machine learning-based solutions require access to such sensitive information, concerns have recently been raised regarding data privacy and security [3]. Furthermore, the confidentiality protection laws currently adopted for the manipulation of personal data (e.g., EU GDPR and US HIPAA) demand the use of more effective privacy protection strategies.

Usually, a proper anonymization must be done in order to export sensitive data without violating confidentiality.

Through data masking, some of the information properties are thus changed, resulting in a trade-off between confidentiality and utility. In certain cases, e.g., genomic data, the use of anonymized data limits the neural network's ability to gain valuable information and insight from the data. Herein, a method based on homomorphic encryption (HE) is employed as a way to address the limitations imposed by conventional methods, and to maintain confidentiality of biometric data. HE is a specific form of encryption which allows data to be encrypted while it is being manipulated. By preserving the mathematical structures that underline the data, HE represents a promising solution for guaranteeing privacy while still maintaining full utility. The chosen HE scheme (MORE) [4] allows for a limited set of operations to be conducted directly on encrypted data without exposing the underlying information or the encryption key. In the context of deep learning-based solutions, this property is especially useful as it ensures that both data and predictions are kept private while data are processed. Taking into account the practical difficulties arising from the use of deep networks on encrypted data and the inefficiency of current approaches, we propose a method that improves the effectiveness of encryption models in real-world applications by facilitating calculations over rational numbers, faster operations, and performance close to that achieved with an unencrypted model.

In order to evaluate the feasibility of the proposed privacy-preserving deep learning solution to produce reliable results, we consider a classic benchmarking application from the computer vision realm, i.e., digit recognition, and two personalized medicine applications. The herein proposed approach has been selected by the European Commission for the 2018 and 2019 Innovation Radar Prize Contest, whose goal is to identify Europe's top innovators and their innovations. In 2019, our solution has won the competition in the category "Industrial & Enabling Technologies" [5]. Parts of the work presented herein have been previously published in [6, 7]. They have been considerably extended herein to include a detailed description of privacy-preserving techniques for machine learning, homomorphic encryption, and deep learning. Furthermore, we have also included extensive details on the deep neural network models developed for the three applications that were considered, as well as detailed results in terms of model accuracy and runtime performance.

The remainder of the paper is structured as follows. Section 2 discusses the latest achievements in the domain. Section 3 includes a decryption of the considered homomorphic encryption scheme. Section 4 offers a brief overview of deep learning and neural networks. Sections 5 and 6 address the proposed privacy-preserving pipeline in three deep learning applications. Section 7 introduces the findings, with an emphasis on the correlation with the unencrypted counterpart. Finally, Section 8 draws the conclusions and outlines a set of issues that remain to be handled in future work.

2. Related Work

2.1. Privacy-Preserving Techniques for Machine Learning. In the past few years, great effort has been invested in the development of different privacy-preserving techniques with the potential of bridging the gap between data privacy and utility, demanded by the recent rise of privacy concerning scenarios. Among these methods, several privacy-preserving machine learning techniques, including homomorphic encryption (HE), secure multiparty computation (SMPC), and differential privacy (DP), have begun to advance rapidly. Such techniques ensure data privacy and at the same time allow for machine learning-based analysis to be performed. While these techniques have shown promising results, their adoption in modern machine learning applications remains bounded because they are highly dependent on the scenario. Moreover, there is always a trade-off involved between privacy and performance or between privacy and utility among these techniques, as each comes with specific strengths and vulnerabilities.

SMPC techniques provide a promising solution for data privacy by allowing analysis to be performed over sensitive data, distributed between different data providers, in a way that does not disclose the sensitive information beyond the analysis outcome. Although the idea of SMPC is not new, lately with the technological and hardware advances, more approaches using SMPC for data privacy guarantees in machine learning applications have emerged in the field [8–13]. The first attempt to train a neural network model in a SMPC setting has been made by Mohassel and Zhang [8], where the neural network-based analysis was performed inside a secure two-party computation for Boolean circuits via secret sharing, oblivious transfer, and garbled circuit. The greatest challenge in SMPC for machine learning is given by the computations of nonlinear functions as such operations introduce a high overhead in the training time. Moreover, the time needed for communications further limits their usability. Although great effort has been continuously invested in improving the technology, SMPC still implies a large communication overhead which makes it infeasible for machine learning, where a large amount of data is required. As the number of involved parties or the model complexity increases, the communication and computation costs are greatly affected.

An alternative solution is to use differential privacy which, as compared to SMP, implies a much lower computational complexity. Methods based on differential privacy provide good security and have been lately shown to achieve promising results when combined with machine learning techniques [14–22]. Such methods treat the privacy-preserving data analysis concerns by adding random noise into the algorithm at different stages. However, a proper noise injection mechanism is required to accomplish a reasonable trade-off between privacy and utility. Consequently, the added noise can greatly affect the precision of the machine learning-based analysis.

A few attempts have been made to address the challenge of data privacy-preserving in machine learning-based

analysis through HE. This special type of encryption allows data to be encrypted while it is being manipulated. Hence, it aims at keeping the data private by allowing a third party to process the data in the encrypted form without having to reveal the underlying information. By preserving the mathematical structures that underline the data, HE represents a promising solution for guaranteeing privacy while still maintaining full utility. As ciphertext data are centralized to one single entity, it does not imply any communication bottleneck, as compared to SMPC. Although early work on HE [23] involves highly intensive computations, making the method infeasible for the machine learning algorithm, the recent subsequent schemes gave rise to a series of privacy-preserving machine learning solutions [24–29]. The first notable work that combines HE with neural network was proposed by Orlandi et al. [28]. The method uses a HE cryptosystem known to handle only a few basic operations directly on the ciphertext data. Therefore, to perform computations that cannot be completed in the encrypted domain, an interaction between the data owner and the server was required. Those operations were carried out using garbled circuit protocol (e.g., functions expressed as circuits of logical gates). CryptoNets [25] completely eliminate the interaction between the involved parties by using low-degree nonlinear polynomial functions. The method is based on the idea of using an already trained neural network on encrypted data to retrieve encrypted results. The encryption scheme used in CryptoNets, YASHE [30], does not support floating-point numbers. For this reason, real numbers were converted to integers. The computational complexity alongside the performance limitation introduced when handling large networks limits their usability. To mitigate the problem introduced by the model complexity, CryptoDL [26] proposed to approximate all nonlinear functions within a model with low-degree polynomials. However, none of these schemes cover privacy-preserving training in deep neural network models. The main drawback of these privacy-preserving neural network solutions is the computational overhead: deeper networks require more computations which results in longer running time. Additionally, the attempt to address the nonlinearity property in neural network models through an approximation mechanism does not necessarily result in better performances. In fact, most of these HE solutions fail to maintain the highest prediction accuracy due to the polynomially approximated activation functions. Moreover, due to the HE cryptosystems, these approaches involve an encoding mechanism, i.e., scaling, that converts floating-point numbers with fixed precision to integers.

Alternatively, many researchers began to combine these techniques to improve the level of accuracy and privacy [31–33]. Chase et al. [32] combined SMPC with DP techniques to enable privacy-preserving collaborative neural network training. The proposed solution provides data security guaranties through DP and uses SMPC to allow machine learning-based analysis when data are distributed between multiple parties. The experimental results showed that the performance was affected when larger networks were used. Barni et al. [31] provided a solution to enable

computations within a neural network to be performed on homomorphically encrypted data but relied on the SMPC for the nonlinear functions. Gazelle [33] covers the privacy-preserving neural network inference phase by using HE for the linear operations and traditional SMPC (such as garbled circuits) for the activation function computation.

2.2. Homomorphic Encryption. With Gentry's first introduction of a fully homomorphic encryption (FHE) scheme [23], numerous variations of the original strategy were proposed in the literature [34]. Most of these schemes are known for their efficiency in terms of security, but they are computationally intensive and only a limited number of operations can be performed before decryption is no longer possible. This clearly restrains their usability in real-world applications. Aspects like computations being several orders of magnitude slower than the plaintext counterparts accumulated noise that limits the overall number of operations that can be performed, and all computations being implemented modulo N pose a great challenge for the synergy of deep learning and data analysis. While recent advances in HE led to many variants of encryption schemes, no currently available scheme can manipulate rational numbers.

Several open-source HE libraries have emerged in recent years, each one with different properties based on the employed encryption scheme [35]. Microsoft's Simple Encrypted Arithmetic Library (SEAL) [36] scheme, with support for the Brakerski/Fan-Vercauteren (BFV) [37] and the Cheon-Kim-Kim-Song (CKKS) scheme [38], and IBM's HELib [39] based on the Brakerski-Gentry-Vaikuntanathan (BGV) scheme [40] are two of the most widely used HE libraries. The first noticeable shortcoming of HELib is the lack of support for floating-point numbers. To allow for computations to be performed on rational numbers, SEAL takes advantage of a particular property of the CKKS scheme: rescaling can be performed without changing the encrypted value. Since a plaintext is represented as a polynomial with integer coefficients, floating-point parameters of the message are scaled by a parameter that affects the precision of the computations. Homomorphic operations performed with both HELib and SEAL introduce noise, thus limiting the number of operations that can be performed with ciphertexts. Noise-management techniques have been integrated to maintain the noise level below a certain threshold, such that the ciphertext does not become corrupted. While HELib uses the expensive procedure of bootstrapping to enable unlimited computations, SEAL uses a scale-invariant error reduction technique which requires an estimation of the number of operations that will be performed as an a priori information. Moreover, there are some limitations on the types of operations that can be performed on the ciphertext. The schemes used in HELib and SEAL are fully homomorphic with respect to addition and multiplication, and only polynomial functions can be easily performed. As a consequence, there is no implicit support for division, and nonlinear functions have to be approximated by low-degree polynomials.

While these schemes are known for their efficiency in terms of proven security, the above-mentioned restrictions, alongside their computational overhead, introduce noticeable constraints in the neural network topology, which in turn affect the performance of privacy-preserving neural networks [41].

Other proposed methodologies rely on employing partially homomorphic encryption (PHE) instead of FHE. Since FHE is currently practically impossible to be used in a real-world system, a viable approach is a system based on PHE that is specialized only for certain operations. Such an approach introduces a clear advantage in terms of running time and may be used in a practical application with reasonable overhead [42]. Various encryption schemes have homomorphic properties, out of which we mention the Paillier scheme [43], an additive homomorphic scheme where addition in the ciphertext space corresponds to multiplication in the plaintext space, and the ElGamal scheme [44], a multiplicative homomorphic scheme, which, through some modifications, can become additive. Other PHE with the potential to be used in a practical application are Goldwasser-Micaly [45] that allows computing the XOR operation on encrypted data, searchable encryption [46] with support for keyword search, order-preserving encryption [47] for sorting encrypted values, and deterministic encryption [43], that allows equality checks on encrypted values.

Another promising method is the algebra homomorphic encryption scheme (AHEE) [48], an encryption scheme that is homomorphic with respect to algebraic addition and multiplication, i.e., both addition and multiplications can be performed on encrypted data. A key advantage of this scheme is that it has a relatively small computational complexity, same as Paillier and ElGamal, while being homomorphic with both addition and multiplication. The main limitation of this scheme as well as for Paillier and ElGamal is that it only allows the encryption of relatively small integer numbers. More specifically, during the encryption process, an exponentiation operation needs to be evaluated, where the exponent is the message to be encrypted. Hence, even with a multiprecision arithmetic library, the operation can still cause an overflow. Using 1024 bit integers, we found that only numbers with up to about 10^3 can be encrypted. This limitation becomes even more important when performing computations on encrypted data, i.e., one cannot determine if an encrypted number is too large for performing a certain operation.

In order to facilitate the privacy-preserving deep learning-based analysis, the cryptosystem must allow the computations within the model to be performed on rational numbers. To address this requirement, an encryption mechanism is typically used to encode a given rational number as a sequence of integers [49]. As some of the basic operations are difficult, if not impossible, to apply on encoded data, such an approach has limited functionality when applied on real data. Furthermore, the encoding strategy not only explicitly limits the data utility but also directly affects the results of the computations.

Over the past few years, many HE schemes have been proven to meet the security requirements. Although sufficiently secure, most of these approaches offer poor performance as they suffer greatly from runtime bloat, i.e., several orders of magnitude slower than the plaintext computations. This clearly restrains their usability in real-world applications. Consequently, simplified encryption schemes based on linear transformations emerged in the field as more practical alternatives. Despite the criticism for weaker security [50, 51], this type of cryptosystems seems to be the currently only feasible method with the potential to enable privacy-preserving computations in real-world applications.

As a consequence, the herein employed methodology relies on a variant of the matrix-based homomorphic encryption scheme proposed in 2012 by Kipnis and Hibshoosh [4]. In contrast with the currently adopted schemes in privacy-preserving neural network-based solutions [25, 26, 29], the MORE (Matrix Operation for Randomization or Encryption) encryption scheme is noise free and nondeterministic (multiple encryptions of the same plaintext data, with the same key, result in different ciphertexts). An unlimited number of operations can therefore be performed on ciphertext data. Moreover, the MORE scheme enables all four basic arithmetic operations over encrypted data. Herein, MORE was redesigned to directly support floating-point arithmetic in order to address the floating-point precision constraint of privacy-preserving deep learning-based analysis on real-world data.

3. Matrix-Based Data Randomization

A variant of the MORE encryption scheme has been considered and adapted to directly operate on floating-point data. Following the MORE encryption strategy, a plaintext scalar is encrypted as a $n \times n$ ciphertext matrix, and matrix algebra is employed to enable computations on ciphertext data. All operations performed on ciphertext data are therefore defined as matrix operations, e.g., the multiplication of plaintext scalars is formulated as the matrix multiplication of ciphertext matrices. The order of the matrix used to encrypt a message represents an important factor that governs the trade-off between security and efficiency. For a 2×2 setup, the MORE cryptosystem is summarized in Table 1.

The MORE scheme allows for algebraic operations to be performed on ciphertext matrices, i.e., given two encrypted matrices $C_1 = SM_1S^{-1}$ and $C_2 = SM_2S^{-1}$, for addition

$$C_1 + C_2 = SM_1S^{-1} + SM_2S^{-1} = S(M_1 + M_2)S^{-1}, \quad (1)$$

which is the encryption of $M_1 + M_2$, and for multiplication

$$C_1C_2 = SM_1S^{-1}SM_2S^{-1} = SM_1M_2S^{-1}. \quad (2)$$

The same property applies to subtraction and division, as well as to plaintext scalar operations, making the scheme fully homomorphic for algebraic operations.

TABLE 1: MORE encryption scheme setup over rational numbers.

Message	Scalar value $m \in \mathbb{R}$
Secret key generation	Invertible matrix $S \in \mathbb{R}^{2\times2}$
Matrix construction	$M = \begin{pmatrix} m & 0 \\ 0 & r \end{pmatrix}$, where $r \in \mathbb{R}$ is a random parameter
Encryption operation	$\text{Encryption}(m) = C = SMS^{-1}$
Decryption operation	$\text{Decryption}(C) = K = (S^{-1}CS)$
Message recovery	$m = K_{(1,1)}$

3.1. Encryption of Rational Numbers. The original MORE scheme, like any FHE or PHE approaches, is only applicable to positive integer numbers modulo N, with all operations being performed modulo N. To be able to operate on rational numbers, these schemes rely heavily on an encoding mechanism. As a consequence, a real number is converted into an integer or a set of integer numbers, and only afterwards the scheme is used to homomorphically encrypt the encoded number. A typical approach to formulate the encoding is through continued fractions [49]. Whilst a precise representation can be obtained, even basic operations on numbers expressed in this form are difficult to perform. Alternatively, a simpler encoding can be infused by multiplying the rational numbers with a large scaling factor. Although, more permissive, it requires a mechanism that manages the scaling factor, which is difficult to achieve for certain operations, e.g., division, where this factor is reduced. In addition, noise is typically introduced in the cryptosystem by extending the methodologies to operate on rational numbers. Consequently, a noise-management strategy has to be employed to limit the noise level. Even though the handling of rational numbers seems to be a straightforward problem, there is currently no solution that allows them to be used in the context of HE.

One of the main benefits of the MORE encryption scheme is that it can be directly formulated for rational numbers. The drawback is that the method becomes vulnerable to known ciphertext attacks, as described in Section 7.3.

3.2. Performing Operations over Encrypted Data. The MORE encryption scheme has been shown to be fully homomorphic with respect to basic algebraic operations. In real-world applications, including deep learning-based approaches, nonlinear (e.g., exponential, logarithmic, square root, etc.) functions need to be handled. Most of the regular approaches adopted to conduct nonlinear operations are based on the idea of approximating the specified function with a finite polynomial series (e.g., truncated Taylor series). Following this approach, nonlinear function computation relies solely on algebraic operations, being fully compliant with the MORE encryption setup. However, within the MORE cryptosystem, a more convenient approach is possible.

Given the properties that govern the encryption scheme, and knowing that ciphertext-based operations rely on matrix algebra, nonlinear functions can be computed either (i) directly as matrix functions or (ii) through matrix decomposition. While the first method is straightforward, the second approach is based on the property according to which a message m, that is to be encrypted, will be always found among the eigenvalues of the ciphertext matrix C. For example, in a 2×2 setup, one of the eigenvalues corresponds to the random number r used during the matrix construction, while the other corresponds to the message m. To ensure that the message can only be identified through proper decryption and by possessing the secret key, the random number r is generally chosen to be statistically indiscernible from the message. Applying a function f on the ciphertext data C is therefore equivalent to applying f directly on the eigenvalues of C. Thus, matrix decomposition VLV^{-1} is first used to decompose the ciphertext matrix C into eigenvalues L and eigenvectors V. Thereafter, the nonlinear function that is to be evaluated is applied independently on each of the eigenvalues. Finally, the resultant ciphertext matrix is reconstructed as $C_f = Vf(L)V^{-1}$ using the original eigenvectors and the eigenvalues evaluated on the function f. As compared to the direct matrix function-based computations, this approach can even be used to perform comparisons between the ciphertext matrix C and a plain scalar s. Neither of these two methods support nonlinear binary operations between two ciphertext data. However, in deep learning, such operations can be completely avoided.

Starting from these strategies, Algorithm 1 shows how, given any ciphertext $C \in \mathbb{R}^{2\times2}$, the two proposed methods can be used to formulate the function $f(x) = (1/(1 + e^{-x}))$ defined on $x \in \mathbb{R}$, under the MORE assumptions. This function is known as the logistic sigmoid function and is widely used in neural networks for its nonlinear properties, as will be outlined in the next sections.

4. Deep Learning

Since their first appearance in 1943 [52], the functionality of neural networks has been constantly associated with the way people learn and process information. More specifically, they were designed to emulate the synaptic connections between brain neurons and later on became the foundation of deep learning.

Fueled by the recent hardware advances, the rise of big data and desire for exceeding human-level performances, deep neural networks are currently becoming a widespread technology in data mining. The most notable turn point was marked in 2012 [53], when a deep learning-based approach, AlexNet, obtained unprecedented results (11.5% error rate) in computer vision on the ImageNet Large-Scale Visual Recognition Challenge [54], surpassing the winning entry of 2011 by an improvement of $\approx 45\%$. Thereafter, the ImageNet

classification accuracy gradually improved; in 2017, the lowest reported error rate on ImageNet has dropped to 2.25% [55], being superior to the reported human error rate of approximately 5% [54]. Since then, deep neural networks experienced an extraordinary growth rate, leading to an explosion of research in many fields.

Medical data interpretation is highly subjective, prone to errors, and most often it depends on the experience of the medical expert. To automatically capture the high variability in anatomical structures and features, deeper neural networks are required. As compared to traditional machine learning models, deep networks can model the complex relationships and capture relevant information and patterns from different perspectives, representing a much better candidate for the development of CADe solutions [1]. In consequence, nowadays, deep learning is being widely used for tasks that were previously known to represent great challenges, achieving state-of-the-art performances in different healthcare domains and applications [1].

4.1. Neural Networks. On a high level, a neural network can be defined as a computational model that maps inputs to outputs through a composition of layers with interconnected processing blocks (transformations and activation functions). The architecture of a simple neural network is depicted in Figure 1. To allow for a complex arbitrary functional mapping, nonlinear activation functions are typically added at each processing block. They filter the information that passes through the network, determining what input signal is relevant to be forwarded to the following layer. Essentially, they decide whether a certain neuron should be activated or not and without them the neural network becomes a simple linear model. There are several functions used as activation functions, including but not limited to logistic sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU).

In supervised learning, the model automatically learns the mapping function (parameters of the model) based on labeled training examples in an iterative fashion by gradually making an adjustment with the effect of minimizing an error function, i.e., loss, between expected and achieved outputs. Algorithm 2 describes the typical operations involved in the neural network training phase. The term epoch refers to the complete processing of the entire training dataset. Due to computational reasoning, for large datasets, the processing, and hence the parameter adjustment, is done on subsets of data (batches). In the first iteration, the forward propagation step provides the predicted outputs for the input samples given a set of randomly initialized parameters. Thereafter, the error (loss) between the predicted outputs and the desired ones is computed and passed backwards through the network to determine the direction in which each parameter has to be adjusted to decrease the overall prediction error. Finally, the obtained gradients are used to update the network parameters following a numerical optimization method (e.g., gradient descent). This process is repeated over many epochs until the network error stops decreasing.

Upon training, the network should be able to provide results which are statistically similar to the expected ones even when presented with input data never encountered by the network during training. Consequently, neural networks can be used in predicting an output from certain input features, classifying data and even localizing patterns or objects in images.

4.2. Going Deeper: Deep Neural Networks. In essence, a deep neural network is nothing else than a neural network model composed of several layers of processing blocks and organized as an input layer, followed by multiple hidden layers, and an output layer. Over the years, it has been shown that such an architecture facilitates the modeling of highly complex functions, allowing for the learning of richer intermediate representations. Hence, the key difference between shallow and deep neural networks is given by the depth of the models; although not standardized, typically a network with depth higher than two falls into the deep learning category.

Over the past few years, we have seen a global trend for neural networks becoming deeper. Within 3 years, reported state-of-the-art models experienced a massive increase in depth, from the 8 layers of the AlexNet [53] to 100+ layers found in residual networks [56]. This contradicts the universal approximation theorem according to which a neural network with a single wide hidden layer is already enough to approximate any function [57]. While in theory, the same mapping function between input and output may be learned by shallow and deep networks, empirical work showed that it is harder to optimize a shallow network to approximate a complex function as accurate as deep networks, even when the same number of parameters is used in both models [58]. By using deep architectures, the learned function is expressed as a composition of several simple functions. It was previously shown [59] that the same degree of accuracy can be achieved by shallow and deep networks in approximating a function using the property of compositionality, but exponentially lower number of training parameters and sample complexity were required by the deeper models. When AlexNet won the ImageNet competition, it has been reported that removing any of the hidden layers led to a 2% loss in performance [53]. Moreover, as emphasized by Goodfellow at al. [57], a deeper model can reduce the generalization error. This statement is backed by the results obtained in practice in many fields and tasks where a greater depth was associated with better generalization [60–63]. These results suggest that deeper networks are more powerful at encoding the multilevel feature representation directly from the high dimensional input data.

However, the training of deep networks is challenging. As the network becomes deeper, problems such as vanishing or exploding gradients may appear, which hamper convergence and degrade the model performance [64]. These challenges have been largely addressed by the deep learning community leading to improvements in the architectures by introducing intermediate normalization layers [65] and shortcut connections [56]. Moreover, deeper networks
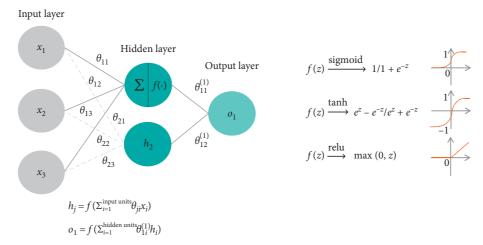
FIGURE 1: Architecture of a simple neural network described by the input, output, and a hidden layer in between. Information flows through all layers, starting from the input layer to the output layer. Herein, every neuron receives information from all neurons of the previous layer (in literature called a fully connected layer). The connections θ between the processing blocks are the parameters that have to be adjusted in accordance with data and the formulated problem. Each processing block performs a transformation (herein a weighted sum of the input parameters) and the result is passed to an activation function f that will be used to add nonlinear properties in the network. Activation functions are usually selected from a set of limited functions with certain mathematical properties. Nonlinearity is needed to allow for a complex arbitrary functional mapping between input and the output data.

require substantially more computing power, which sometimes leads to a trade-off between model complexity and performance. As in any data-driven model, the training dataset represents the fundamental resource behind the achieved performance, and deeper networks require a considerably larger amount of training examples to generalize well. While the performance of traditional machine learning approaches saturates once a certain training dataset size is reached, deep learning models have the capability to further improve their performance with more data.

4.3. Deep Learning Models. The deep learning model that won the ImageNet competition in 2012 was not a traditional deep neural network. Instead, a deep convolutional neural network (CNN) architecture was proposed to enable feature learning directly from the input images, completely mitigating the need for hand-designed features as in traditional learning-based models. In a CNN, the meaningful contents for a specific task, usually described as high-level features, are learned from the lower ones in a fully automatic manner incorporated in the backpropagation-based training procedure. Since its inception, CNN remains one of the most popular types of deep learning approaches used in data-driven medical imaging analysis [66].

Compared to a traditional deep neural network, where all layers are fully connected, a CNN relies extensively on convolution and pooling layers (downsampling units). The operation performed inside a convolutional layer is a dot product between the input of the layer and a small learnable filter (kernel). To cover the entire input, a sliding window strategy is adopted. During training, the parameters of the filters are adjusted to extract relevant information for the given task directly from the input data. Pooling layers have no learnable parameters and are used to reduce the spatial dimension of the data by reducing the information in a small

area to a single value (by performing an averaging or maximum operation). By using a combination of such layers, the network exploits local connectivity making the model invariant to scaling or shifting transformations. By increasing the number of layers, the network's receptive field is expanded, which in turn forces the model to progressively capture more complex patterns, from edges to shapes or objects. Moreover, the use of local receptive fields, sparse connectivity, and parameter sharing drastically reduces computational overhead and the number of parameters that have to be learned, as compared to traditional neural networks.

Over the past few years, based on the connection pattern between layers, many neural network architectures have been proposed for specific tasks: fully connected neural networks (FCNNs) are primarily used for classification or regression tasks where a global relationship between the input features is required and spatial information can be discarded, CNN specialized on data that pose a certain spatial relationship (e.g., images), and recurrent neural networks (RNNs) are used for sequence or time-series data processing.

Motivated by the input data types, the amount of available data, and the formulated problems, herein we focus on a FCNN for solving a regression task and on a CNN for image-based analysis. The general architectures of the deep neural networks underlying the studied methods are represented in Figures 2 and 3.

5. Deep Neural Networks over Encrypted Data

In this section, aspects of privacy-preserving deep neural networks are described. The proposed method relies on the MORE homomorphic encryption scheme and enables both the training and the exploitation of classical neural network models directly on homomorphically encrypted data.
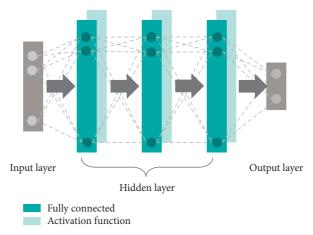
Figure 2: General structure of a deep fully connected or feed-forward neural network (FCN). The data are fed through a stack of fully connected layers, where each neuron is connected to all neurons of the following layer. The network is typically used on tabular data or on handcrafted features.
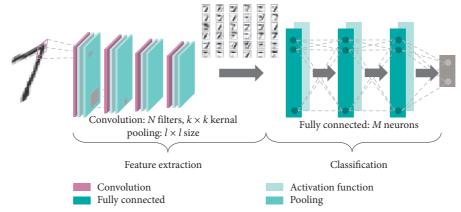

Figure 3: General structure of a deep convolutional neural network (CNN). The network involves two main parts: a feature extractor and a classifier. The feature extractor is composed of multiple alternating convolutions and pooling layers. Each convolution layer performs a full scan of its input by gradually analyzing small regions $(k \times k)$ and extracting a number N of distinctive features. The extracted information is further passed to the pooling layer where its dimensionality is reduced based on the pooling size $(l \times l)$. With each additional layer, the network learns to extract higher-level features while the dimensionality of each feature is continuously reduced. The classifier takes the final extracted features as input to a fully connected neural network that combines the information to produce the final output. Such a network is primarily used for processing images or videos.

5.1. Method. Over the past few years, we have seen remarkable performances being obtained by deep learning-based analysis in the medical field. The complex mathematical formulation of deep learning models ultimately breaks down to a series of repeating blocks of computations that rely on a limited set of simple operations over rational numbers. In fact, many of the deep learning-based state-of-the-art results have been achieved by deep neural network models that were using only limited types of operations (e.g., multiplication, addition, division, subtraction, exponential and logarithm). By leveraging the homomorphic property of the MORE scheme, the functionality of neural network models can be further formulated to account for operations on ciphertext data.

The proposed workflow, based on HE and deep learning, is outlined in Figure 4. Before being processed, training data are encrypted with a secret key that is never shared (Algorithm 3). Thereafter, the deep learning-based model will

have access only to the encrypted version of the data (ciphertext), while the actual data (plaintext) are detached from the processing unit and remain private on the side of the data provider. Finally, with the homomorphic property underlying the MORE encryption scheme, the direct support for floating-point arithmetic, and with all operations performed inside the network formulated to ensure applicability on ciphertext data, the network can be trained directly on ciphertext data following the classical pipeline described in Algorithm 2. This results in a model that provides encrypted predictions, which can only be decrypted by the owner of the secret key, following Algorithm 4. Once the training phase is finalized, the encrypted form of a model can be employed to predict new encrypted instances (inference phase), where input samples are encrypted with the same key as the one used during the training phase. The MORE cryptosystem relies on symmetric keys. Hence, a secret key, generated following the approach presented in Algorithm 5, is used for
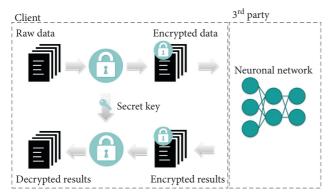
FIGURE 4: Workflow of the proposed privacy-preserving deep learning-based application relying on homomorphic encryption.

both the encryption of the plaintext data and the decryption of the ciphertext data.

The proposed deep learning-based ciphertext data analysis framework is presented in Algorithm 6. For the sake of comparison and validation, we also provide the pipeline used for plaintext data analysis (Algorithm 7). Note that in Algorithm 6, all operations performed during training and prediction are formulated in accordance with Sections 3.1 and 3.2.

Following this approach, data privacy is retained during both training and inference, as the external party operates explicitly on ciphertext data and delivers results as ciphertext data. Consequently, the secure processing of medical data is performed in such a way that the external party cannot derive knowledge from the patient data, and the user is unable to obtain information regarding the machine learning model [7].

6. Experiments

To validate the proposed method, we focused on solving three types of deep learning applications: regression, binary, and multiclass classification. We first addressed a well-known benchmarking application (digit classification) and then aimed at addressing the privacy issue in two healthcare related applications by training neural network models on encrypted data (i) to assess whole-body hemodynamics and (ii) to distinguish coronary artery angiographic views.

The aim of the conducted experiments was not to achieve deep learning-based state-of-the art results for the proposed problems but to investigate the possibility of maintaining data privacy while still allowing for computations within a neural network to be successfully performed over the encrypted version of the data.

This section describes the datasets and the experimental setup, including the proposed deep neural network models and the common hyperparameters used for each of the above formulated problems.

6.1. Problem Formulation

6.1.1. MNIST: A Typical Dataset for Neural Networks. A typical problem studied in the context of neural networks is that of classification. More specifically, the problem of image categorization is in accordance to the information depicted in the image. The MNIST (Modified National Institute of Standards and Technology) database [67] contains images representing handwritten digits and is typically employed as reference for benchmarking image classification algorithms.

The choice for the digit recognition task as a first experiment that addresses the challenges of privacy-preserving computations in neural network models was made with the intention of providing valuable insights about the strengths and vulnerabilities of the proposed method in a task that is already considered solved.

The approaches used in the literature to solve the digit recognition problem range from the classical linear classifier, to support vector machine (SVM), and more recently to deep neural network models. However, deep CNN models have been shown to perform significantly better than other types of classifiers on MNIST, leading to the lowest reported test error rates [68]. Moreover, when corresponding shallow networks were employed, the error rate has increased [68], reemphasizing the need for deeper models.

The digit recognition problem is framed as predicting the probability of an image belonging to each of the 10 classes (0–9 digits). Hence, target labels are typically represented as one-hot vectors with only the associated class having the value 1 and 0 for the rest. This example is a typical case of a multiclass classification problem ($C = 10$ classes) that can be solved by a neural network model trained to minimize a cross entropy error between the predicted ($\tilde{y}$) and expected (y) probability distributions:

$$CE(y, \tilde{y}) = - \sum_{c=1}^{C=10} y_i \log (\tilde{y}_i). \tag{3}$$

(1) Dataset. The MNIST dataset consists of 70,000 gray-scale images of relatively small dimension, 28×28, each image being labeled with the digit it depicts (Figure 5). The digits are size-normalized and centered in the images. The MNIST samples were partitioned into three datasets resulting in 50,000 cases used for training a neural network classifier, 10,000 for validating the trained model, and 10,000 for assessing the classifier's performance. The training samples were balanced over the ten classes to avoid class imbalance problems that commonly arise in classification.

In MNIST images, pixel values range from 0 to 255. In order to improve training convergence, pixel values were scaled to [0, 1] based on the minimum and maximum pixel intensity. To perform the neural network training, the MNIST labels represented by numerical values from 0 to 9 were encoded into categorical values as one-hot vectors. Therefore, each digit was represented by a vector of length equal to the number of classes, and where the digit position
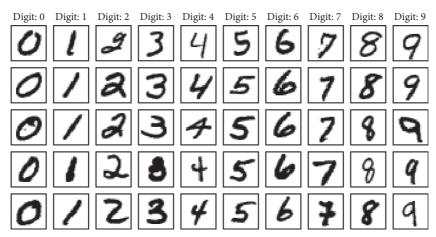
Figure 5: Example images from MNIST dataset.

was marked in the vector with a value of 1, all other values being set to 0.

6.1.2. Whole-Body Circulation Model.

In the following we showcase the use of the considered approach in a personalized medicine application, based on a whole-body circulation model (WBC). The cardiovascular system is a closed loop system and cannot be modeled using spatial hemodynamic models (3D models especially) due to the associated large computational complexity. Thus, the typical approach is to employ lumped parameter models, which rely on the analogy between electricity and hydraulics. The model considered for this study is depicted in Figure 6 and is composed of the left and right ventricles and atria, the arteries, capillaries, and veins of the systemic circulation, and the arteries, capillaries, and veins of the pulmonary circulation [69].

Each of the four heart chambers is represented by a time-varying elastance model:

$$P(t) = E(t) \cdot (V(t) - V_0) - R_S Q(t), \tag{4}$$

where the time-varying elastance is E, the chamber volume is V, the dead volume of the chamber is V_0, and an additional term accounts for the relationship between cavity pressure and flow, parameterized by R_s [70] $(R_s = K_s E(t)(V(t) - V_0), K_s$ – constant). The chamber volume is computed as

$$\frac{\mathrm{d}V}{\mathrm{d}t} = Q_{\mathrm{in}} - Q_{\mathrm{out}}. \tag{5}$$

The four heart valves (aortic, mitral, pulmonary, and tricuspid) are modeled using a diode, a resistance, and an inertance. The valve closure and opening is triggered by the upstream-downstream pressure difference. The following hemodynamic relationship is employed when the valve is open (flow is zero when the valve is closed):

$$P_{\mathrm{in}} - P_{\mathrm{out}} = R_{\mathrm{valve}} \cdot Q + L_{\mathrm{valve}} \cdot \frac{\mathrm{d}V}{\mathrm{d}t}, \tag{6}$$

where P_{in} is the pressure at the inlet of the valve and P_{out} is the pressure at the outlet of the valve. The valve is opened when P_{in} becomes larger than P_{out} and is closed when a negative flow rate is encountered. The systemic circulation is represented by a three-element Windkessel model:

$$\frac{\mathrm{d}P_{Ao}}{\mathrm{d}t} = R_{sys-p} \frac{\mathrm{d}Q_{Ao}}{\mathrm{d}t} - \frac{P_{Ao} - P_{ven}}{R_{sys-d} \cdot C_{sys}} + \frac{Q_{Ao}(R_{sys-p} - R_{sys-d})}{R_{sys-d} \cdot C_{sys}}, \tag{7}$$

where the distal and proximal resistances are R_{sys-p} and R_{sys-d}, respectively, the compliance is C_{sys}, and the venous pressure is P_{ven}. The systemic venous circulation is represented by a two-element Windkessel model:

$$\frac{\mathrm{d}P_{ven}}{\mathrm{d}t} = \frac{Q_{ven}}{C_{sysVen}} - \frac{dP_{ven} - P_{RA}}{R_{sysVen} \cdot C_{sysVen}}. \tag{8}$$

The pulmonary circulation is modeled analogously.

When run under personalized conditions, this hemodynamic model can determine different quantities which are clinically relevant, namely, the arterial compliance, the arterial resistance, the dead volume of the heart chambers, the stroke work, the arterial/ventricular/atrial elastance, the ventricular-arterial coupling, the PV loop, etc. To personalize the model, its parameters need to be calibrated, to ensure that patient-specific measurements are matched by the model outputs.

The personalization procedure considered herein has been previously introduced in [71]. It relies on two steps: (i) a number of parameters are computed from the input measurements and (ii) an automated calibration method, relying on an optimization based iterative workflow, tunes the remaining parameter values.

The input measurements are as follows:

(i) Pulmonary circulation: systolic pressure in the pulmonary artery, end-diastolic pressure in the pulmonary artery, end-diastolic and end-systolic volumes of the right ventricle, and ejection time of the right ventricle.
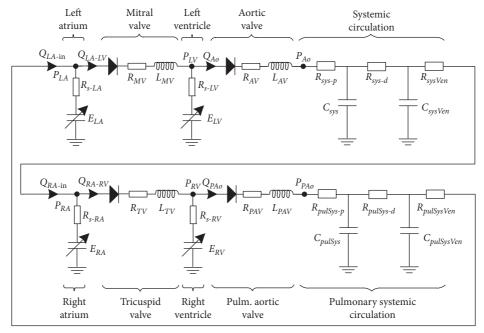
FIGURE 6: Whole-body circulation model.

(ii) Systemic circulation: systolic pressure in the aorta, end-diastolic pressure in the aorta, end-diastolic and end-systolic volumes of the left ventricle, and ejection time of the left ventricle.

The patient-specific outputs computed after finalizing the personalization are as follows:

(i) Pulmonary circulation: right ventricular dead volume, resistance and compliance of the pulmonary arterial circulation, and ratio between proximal and distal resistance in the pulmonary arterial circulation.

(ii) Systemic circulation: left ventricular dead volume, resistance and compliance of the systemic arterial circulation, and ratio between proximal and distal resistance in the systemic arterial circulation.

The automated personalization procedure described above is formalized as an optimization problem, having as goal the determination of a set of parameter values which minimize the difference between computed and reference objective values. The number of objectives is chosen to be equal to the number of parameters to be personalized; hence, we are solving a system of nonlinear equations. Concretely, the dogleg trust region method is employed for finding the root of the system of equations [71].

Although one WBC forward run is very efficient in terms of computation time, the hundreds of runs required for the calibration procedure lead to a runtime of up to one minute on a standard desktop hardware configuration. Hence, a solution capable of determining in real time the personalized hemodynamic measures of interest of the WBC model would be very valuable, even when considering only the plaintext version.

In the context of deep neural network, this problem is framed as predicting real-valued quantities from a set of input parameters. By modeling this regression task using neural network models, the parameters are adjusted during training to minimize the squared differences between the predicted ($\tilde{y}$) and expected target (y) values:

$$\text{MSE}(y, \tilde{y}) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2, \tag{9}$$

where n is the number of real-valued quantities.

(1) Dataset. A large training database is required for the deep learning based approach. Since such a database was not available, we have resorted to an approach that we have successfully employed in the past for diagnosing in real time coronary stenoses from computer tomography data [72]: synthetic data are generated, and the training of the deep neural network is based only on synthetic data. The calibration framework is run for each synthetic dataset and the hemodynamic model is employed to compute the measures of interest. In the prediction phase, actual patient data are used as input and the patient-specific measures of interest are determined by employing the learned model (Figure 7).

We generated a training dataset containing 10000 synthetic samples, covering a wide range of functional and anatomical variations that can be identified in patients and in a healthy population [73]. For each sample, the input data were represented by 9 parameters, and the WBC model computed 12 measures of interest as described above. The 10000 synthetic samples were divided as follows: 7000 for training, 1000 for validation, and 2000 were used for the final test.
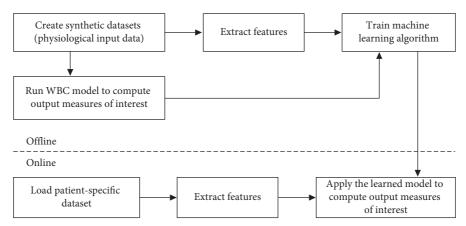
FIGURE 7: Proposed workflow of the deep learning-based model.

6.1.3. X-Ray Coronary Angiographies. The main imaging modality for the diagnosis of coronary artery disease (CAD) is invasive X-ray coronary angiography (ICA) [74]. It allows for a comprehensive assessment of both the function and the structure of the heart. During the invasive procedure, a dye with radio opaque characteristics is inserted in the coronary vessels and a set of images is recorded in succession by an X-ray scanner. This offers a comprehensive overview of the coronary trees, allowing for an evaluation of the coronary stenosis severity, which may be performed either quantitatively (QCA—quantitative coronary angiography) or qualitatively (visual assessment) [75].

However, a solely anatomical evaluation of the stenoses has limited accuracy, and hence a diagnostic index performing a functional assessment (FFR—fractional flow reserve) has been proposed as a superior approach in terms of long-term outcome [76]. Recently, methods for computing FFR directly from the angiographic images have been also introduced [72, 77, 78]. In all cases, angiographic images are acquired independently for the left and the right coronary artery (LCA and RCA) (Figure 8).

Thus, an active research topic in the coronary artery assessment is the automated processing of angiographic images [79], having as goals

  (i) Automated assessment of the coronary stenosis degree.

 (ii) Image-based computation of functional diagnostic indices [72, 78].

(iii) Automated composition of medical reports starting from the findings in the angiographic images.

These represent just a subset of the use cases based on coronary angiography, where an accurate LCA/RCA view classification is a crucial prerequisite for the subsequent processing steps.

The X-ray coronary angiography view recognition can be formulated as a binary classification problem, where a neural network model learns to predict the probability of an image belonging to the positive class (represented by the value 1). Hence, the two categories, LCA and RCA, can be encoded as 0 and 1, respectively. This translates to training a deep neural network in a supervised manner, to minimize a binary cross entropy error:

$$\text{BCE}(y, \tilde{y}) = -\sum_{c=1}^{C=2} y_i \log(\tilde{y}_i) = -(y \log(\tilde{y}) + (1 - y)\log(1 - \tilde{y}),$$

(10)

where y represents the expected class label and $\tilde{y}$ is the predicted probability (as output by the classifier) of the input image as being an RCA image. Hence, the output value of the classifier must reside between 0 and 1 (Figure 9).

(1) Dataset. To investigate the feasibility and effectiveness of deep learning-based analysis for X-ray coronary angiography view classification, an in-house database was used. The available database consisted of 3378 coronary angiographies with 512×512 pixel resolution. A manual annotation of the images was performed to determine the ground truth LCA/RCA view information. One frame, well contrasted by the injected dye, was considered for each angiographic image [9].

To conduct learning-based experiments, the database was split at patient level into training, validation, and test sets. Thus, the training set consisted of 1996 images, the validation set of 680 images, and the testing set, employed only for the end evaluation of the model, consisted of 702 images. Balancing was performed in all 3 sets (prevalence of 1 : 1 for the RCA and LCA images).

Due to privacy considerations, there is a well-known lack of publicly available databases in the medical field. Therefore, to increase the amount of training samples, we opted to synthetically infuse new images through data augmentation. Offline augmentation was employed, leading to a four-fold increase of the training dataset size. The considered strategies relied on transformations like rotation (± 10 degrees), zooming, and shifting. Moreover, the coronary angiography images were downsampled by a factor of 2, resulting in a 256×256 pixel resolution. The range of the input pixel values was rescaled to $[0, 1]$ based on the min-max normalization strategy. Hence, given a pixel value x and the minimum and maximum pixel values across the image ($\min_x$, $\max_x$), the normalized value was computed as
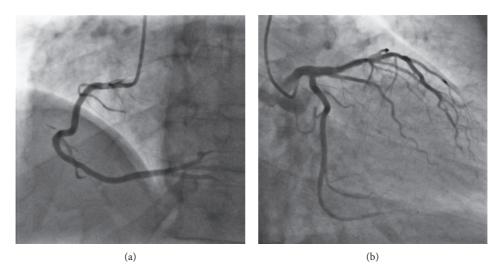
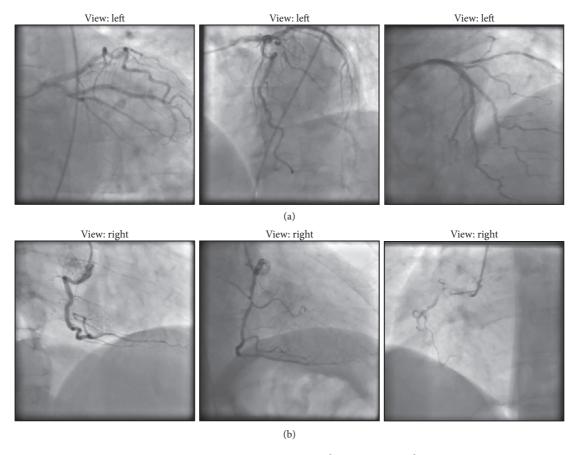FIGURE 8: (a) Right coronary artery. (b) Left coronary artery.

FIGURE 9: X-ray coronary angiography training samples.

$((x - \min_x)/(\max_x - \min_x))$. Based on annotations, the output, i.e., target, was assigned with one of the two values: 1 for RCA and 0 for LCA.

6.2. Ciphertext Database Preparation.

To evaluate the performance of the proposed privacy-preserving method, three databases have been used: images of handwritten digits (MNIST), X-ray coronary angiographies, and synthetically generated WBC samples. A brief overview of these databases is given in Table 2. More details can be found in Sections 6.1.1, 6.1.2, and 6.1.3.

To address the challenge of privacy-preserving computations and to evaluate the use of deep neural network

Input: Ciphertext $\mathbf{C} \in \mathbb{R}^{2 \times 2}$
Output: Ciphertext $\mathbf{R} \in \mathbb{R}^{2 \times 2}$
(1) **function** Sigmoid $(\mathbf{C})$ \\ Using direct matrix operations
(2) $\mathbf{R} \longleftarrow \mathbf{I}_2 \times (\mathbf{I}_2 + \text{MatrixExp}(-\mathbf{C}))^{-1}$ \\$\mathbf{I}_2$ represents the identity matrix
(3)   **return R**
(4) **end function**
(5) **function** Sigmoid $(\mathbf{C})$ \\ Using eigen decomposition
(6) $\mathbf{L}, \mathbf{V} \longleftarrow \text{EigenDecomposition}(-\mathbf{C})$
(7) $\mathbf{L}_f \longleftarrow \text{Diag}(\text{Exp}(\mathbf{L}))$
(8) $\mathbf{C}_{\exp} \longleftarrow \mathbf{V} \times \mathbf{L}_f \times \mathbf{V}^{-1}$
(9) $\mathbf{R} \longleftarrow \mathbf{I}_2 \times (\mathbf{I}_2 + \mathbf{C}_{\exp})^{-1}$
(10)   **return R**
(11) **end function**

ALGORITHM 1: Implementation of the sigmoid function under the MORE encryption scheme.

models over encrypted data, for each dataset, the input samples, i.e., image or feature vectors, were encrypted following the MORE encryption strategy, as described in Algorithm 3. Similarly, the target values, i.e., class labels or real-valued quantities, were also encrypted, except for the binary classification problem where the target was given as plaintext. We chose to encrypt only the input data, i.e., the coronary angiography images, and leave the target, i.e., binary label 0 or 1, as plaintext to show that training can as well be performed if labels are formulated as plaintext. For each experiment, a different secret key was generated following Algorithm 5. In the preprocessing step, the input and output features were normalized, i.e., using mean and standard deviation or minimum and maximum values, to achieve faster convergence.

6.3. Deep Neural Network Model Architecture. To assess the feasibility and effectiveness of deep neural networks to operate directly on homomorphically encrypted data, we conducted three experiments and trained (i) a CNN for digit recognition on encrypted handwritten images, (ii) a traditional FCNN for real-time hemodynamic analysis, where both the input feature vector and the ground truth outputs were encrypted, and (iii) a CNN for encrypted X-ray coronary angiography view classification. For a comparison of model performance and convergence, we also trained the counterpart models on plaintext data.

Although more efficient alternative deep neural network models (e.g., improved activation functions and greater depth) can be adopted to ensure better convergence and superior performance, herein the purpose of the experiments was to assess the correctness and effectiveness of different deep neural network models operating on ciphertext data, as compared to the counterpart models trained on plaintext data.

6.3.1. Deep Neural Network for Handwritten Digit Classification. Starting from the latest results obtained by CNN models on the MNIST digit recognition task, a CNN was employed on encrypted input-output value pairs. The

topology of the proposed privacy-preserving CNN is described in Table 3.

The input image represented as ciphertext data is passed through a stack of 2 convolution layers where 8 and 16 filters with small 3×3 receptive fields were used to extract hierarchical image features. Through averaging, the pooling layer downsamples the images by a factor of two. The last two fully connected layers cover 100 and 10 nodes, respectively, and all activation functions employed in the network, except for the ones in the last layer, are sigmoid functions. Although more efficient options exist to introduce nonlinearities in the network, herein we focused on sigmoid as it is one of the most frequently used activation function. To convert the two-dimensional matrix of features into a vector, a flat layer has been added between the last convolutional layer and the first fully connected layer. A softmax activation function was considered in the output layer to provide class probabilities. Given an input value y_i, the softmax function converts the value into a probability and ensures that output probabilities sum to 1: $S(y_i) = (e^{y_i} / \sum_{j=1}^{10} e^{y_j})$.

6.3.2. Deep Neural Network for Real-Time Hemodynamic Analysis. Given the nature of the input data, i.e., information represented as a feature vector, and driven by the need to model the decision of the network based on a global dependency between input features, a fully connected neural network with 3 hidden layers was employed. The topology of the proposed privacy-preserving FCNN is described in Table 4.

Although deeper and wider networks could have been considered, in the absence of a larger training dataset, this would have led to overfitting. Hence, the number of layers and units was chosen to facilitate the generalization capability of the network. Both the logistic sigmoid and the hyperbolic tangent (tanh) functions were chosen as nonlinearities, with all 3 hidden layers holding the same number of neurons [40]. Herein, a hyperbolic tangent activation function was used to facilitate training and decrease the risk of vanishing gradients. Moreover, we show that neural networks over ciphertext data are not bounded to only one

Input: neural network model $N_{\theta_{h+1}}$ with h hidden layers, training set $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$, number of training epochs M, batch size b, learning rate $\lambda \in \mathbb{R}$, and loss function L
Output: parameters θ_{h+1} of the trained model
(1) **procedure** Train $(\mathbf{x}, \mathbf{y})$
(2)   Initialize network parameters: θ_{h+1}
(3)   **for** epoch $\in M$ **do**
(4)   **for** batch $\in (N/b)$ **do**
(5)   Sample a batch of b training samples: $(\mathbf{x}_s, \mathbf{y}_s) \longleftarrow \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$
(6)   Compute predictions by feeding $\mathbf{x}_s$ forward through the network: $\bar{y}_s \longleftarrow N_{\theta_{h+1}}(\mathbf{x}_s)$
(7)   Compute loss: $L(\mathbf{y}_s, \bar{y}_s)$
(8)   Compute gradient of the loss w.r.t. the parameter in the output layer: $\nabla\theta_{h+1} \longleftarrow (\partial L(\mathbf{y}_s, \bar{y}_s)/\partial\theta_{h+1})$
(9)   **for** $l \in h$ **do**
(10)   Compute gradient of the loss w.r.t. current parameters θ_l of each hidden layer l via backpropagation, following the chain rule: $\nabla\theta_l \longleftarrow (\partial L(\mathbf{y}_s, \bar{y}_s)/\partial\theta_l)$
(11)   **end for**
(12)   **for** $l \in h + 1$ **do**
(13)   Update the model: $\theta_l = \theta_l - \lambda\nabla\theta_l$
(14)   **end for**
(15)   **end for**
(16)   **end for**
(17) **end procedure**

ALGORITHM 2: General training algorithm.

Input: Plaintext data $m \in \mathbb{R}$
Input: Secret key $\mathbf{S} \in \mathbb{R}^{2\times2}$
Output: Ciphertext $\mathbf{C} \in \mathbb{R}^{2\times2}$
(1) **function** Encryption $(m, \mathbf{S})$
(2) $\mathbf{M} \in \mathbb{R}^{2\times2} \longleftarrow$ zero matrix
(3) $\mathbf{M}(\mathbf{0}, \mathbf{0}) \longleftarrow m$
(4) $\mathbf{M}(\mathbf{1}, \mathbf{1}) \longleftarrow$ RandomUniform (min val, max val)
(5) $\mathbf{C} \longleftarrow \mathbf{S} \times \mathbf{M} \times \mathbf{S}^{-1}$
(6)   **return** $\mathbf{C}$
(7) **end function**

ALGORITHM 3: MORE encryption.

type of nonlinear functions, i.e., sigmoid, and that more functions can be adopted if the involved operations are supported by the MORE encryption scheme. Since the problem being solved is formulated as a regression task, no activation function was specified in the output layer, every output value being a linear combination of the outgoing values of the last hidden layer. This ensures that the outputs are real values ranging from $[-\infty, +\infty]$.

6.3.3. Deep Neural Network for View Classification in X-Ray Coronary Angiography.
Motivated by the latest results in data-driven image-based analysis, a deep CNN was adopted to solve the coronary angiography image recognition task. The topology of the proposed privacy-preserving CNN is described in Table 5.

To solve the classification problem, the network has to perceive the overall shape of the coronary artery and not just individual pixels. Therefore, to capture features of increasingly higher order, we opted for a deeper convolutional network that expands the model's receptive field and automatically learns relevant features. As the receptive field becomes larger, the layers can capture features with vast semantic meaning. Moreover, to model the global relationship between the extracted features, fully connected layers have been added. Hence, 4 convolutional layers were used to capture relevant features and 2 fully connected layers were used to capture feature dependencies. We gradually increased the number of features by a factor of 2 with the intention of capturing more complex semantic information. To limit overfitting, a dropout layer was employed for regularization, to randomly ignore 25% of the neurons during the training phase. A sigmoid activation function was used in the latest layer to squash the values between $[0, 1]$ and formulate the prediction of the model as a probability.

6.4. Setup of the Deep Neural Network Models.
To train the deep neural network models on ciphertext data, we used the approach described in Algorithm 7. The hyperparameters

```
Input: Ciphertext C ∈ ℝ^{2×2}
Input: Secret key S ∈ ℝ^{2×2}
Output: Plaintext data m ∈ ℝ
(1) function Decryption(C, S)
(2)     K⟵S^{−1} × C × S
(3)     m⟵K(0, 0)
(4)     return m
(5) end function
```

ALGORITHM 4: MORE decryption.

```
Output: secret key S ∈ ℝ^{2×2}
(1) function Keygeneration( )
(2)     while True do
(3)         S⟵RandomUniform(size = (2, 2), min val, max val)
(4)         if det(S) ≠ 0 then \\ Ensure matrix invertibility
(5)             break
(6)         end if
(7)     end while
(8)     return S
(9) end function
```

ALGORITHM 5: MORE secret key generation.

considered for each experiment are presented in Table 6. To avoid slow convergence and mitigate the chances of vanishing or exploding gradients, especially when using sigmoid or tanh as activation functions, we initialized all weights following Xavier's method [64] with random values chosen from a truncated uniform distribution within $[\sqrt{(6)}/\sqrt{(n_i + n_{i+1})}]$ where n_i and n_{i+1} represent the number of input and output units of the layer. We chose a variant of gradient descent to update the model's parameter across minibatches of training examples to avoid storing the entire dataset into memory and to speed up training in the context of large datasets.

During the data-driven model optimization, it is necessary to closely monitor the training phase to ensure an optimal performance. An inadequate optimization may lead to a network that can neither model the training data nor generalize on new data. Overfitting and underfitting are the two well-known learning-based problems that greatly affect the performance of the model on an unseen dataset. Therefore, these issues can be avoided by knowing when to stop the training. A commonly adopted strategy to prevent the degradation of the model's performance is to define early stopping criteria based on the error on a validation dataset. More specifically, if the error on a held-out dataset does not improve over time or the gap between the training and validation errors widens, the training can be interrupted. In both strategies, the stopping criteria are set upon the error analysis. Although easily adopted during the training phase on plaintext data, these strategies are becoming highly impractical when operating on ciphertext data. Due to the

adopted cryptosystem, and the fact that the error metric is in turn a ciphertext, it cannot be used within a conditional statement. To overcome this limitation, the privacy-preserving models are trained for a predefined number of iterations.

As the overall goal of the study is to assess the feasibility of the deep neural network to operate directly on ciphertext data, i.e., demonstrate that the performance does not drop compared to the plaintext setting, a proper stopping criterion can be predefined. Hence, for usability and simplicity, we have chosen an arbitrarily large number of epochs to conduct the experiments and report the performance.

For each of the tasks, we computed both the unencrypted and the counterpart encrypted version. While the first experiment implies regular training and inference operations, for the encrypted version, the neural network exclusively operates on ciphertext data, with all trainable parameters being completely encrypted. To enable a fair comparison, all networks (plaintext and ciphertext) were trained using the same training strategy, hyperparameters, and initialization method, as outlined in Table 6. Moreover, for consistency, the models trained on both ciphertext and plaintext data started from the same set of initial values.

To measure the performance of the neural network models trained on ciphertext data on the held-out testing set, all evaluation metrics are computed on the decrypted results, where decryption is performed as shown in Algorithm 4.

7. Results

To evaluate the performance of the proposed privacy-preserving deep neural network models, two criteria have been examined: reliability and applicability in medical scenarios.

Consequently, for each of the use cases, the performances, and hence the results, of the data-driven models were analyzed by applying the models on both encrypted (ciphertext) and unencrypted (plaintext) data. By comparing the outcomes of the two scenarios, we analyzed and measured the privacy-preserving models' ability to retain the performances.

Besides reliability, another factor that plays an important role in determining the viability of the privacy-preserving models to operate in clinical routines is the runtime. Therefore, a detailed analysis of the runtime was performed, and both the inference and the training times were reported.

The analyses showed that data security can be ensured on the basis of homomorphic encryption, and that, at the same time, deep learning-based data analysis can be efficiently performed. Furthermore, experiments have indicated that in both encrypted and unencrypted versions, the data-driven models are similarly optimized, as outlined in the following.

7.1. Performance. To showcase the ability of the network to learn from ciphertext data, the training loss for the regression task, as resulted after decryption, is depicted in Figure 10(a). Similarly, the evolution of the training and validation accuracy of the privacy-preserving CNN model fed with encrypted X-ray coronary angiographies, obtained after decryption, is depicted in Figure 10(b).

The training evolution demonstrates the capability of the proposed method to preserve the correctness of the computations. Moreover, after decryption, the parameters learned by the model when trained on ciphertext data were found to be identical up to machine precision to those learned by the unencrypted model. Therefore, the overall performance of the deep learning models on the held-out testing samples has been proven to be consistent regardless of whether the model was previously trained with or without encryption. Hence, all performance metrics reported hereafter are based on the results obtained by the encrypted deep neural network models.

7.1.1. MNIST Binary Classification. The default metric used to assess the performance of a classifier on the MNIST dataset is given by the absolute accuracy of the classification models, i.e., the percentage of correctly labeled digit images. The unencrypted network achieved a classification accuracy of 98.2% on the testing dataset, which was preserved by the encrypted network. In the MNIST database, the class distribution in the held-out testing set is well balanced across the 10 labels, and hence accuracy can be seen as a reliable metric for assessing the classification performance.

For a more detailed evaluation of the performance, we used three additional criteria defined as follows:

(i) Precision: $(TP/(TP + FP))$

(ii) Recall: $(TP/(TP + FN))$

(iii) $F1-score$: $(2 \cdot precision \cdot recall/ (precision + recall))$

where TP represents the number of images correctly predicted as being positive, TN is the number of images correctly predicted as being negative, FP shows the number of images incorrectly predicted as positive, and FN is the number of images incorrectly predicted as negative. Being a multiclass classification problem, the evaluation metrics were computed following the one-vs-rest strategy. More specifically, to compute the metrics, each label was individually considered positive while the others were set as being negative. The precision, recall, and $F1$-score for each digit class are reported in Table 7. To evaluate the digit recognition performance of the proposed CNN model, we show the confusion matrix in Figure 11.

Although 98.2% is an acceptable accuracy for the MNIST digit recognition task, the proposed model did not reach the reported state-of-the-art accuracy of 99.77%. This was to be expected as the classifier proposed to solve the problem of digit recognition was primarily adopted to assess the correctness of the privacy-preserving computations within a traditional convolutional neural network.

The performance of any neural network model can be generally improved by tuning the architecture of the network or by adopting more favorable activation functions and optimization algorithms. Additionally, training data augmentation techniques, e.g. elastic distortions, can be employed to further decrease the classification error rate, as shown in [80]. However, optimizing the model or the hyperparameters to achieve maximum performance on the MNIST digit recognition task was beyond the scope of this work.

7.1.2. Hemodynamic Analysis. The performance of the deep neural network model predicting the outputs of the WBC model was evaluated based on the Pearson correlation and the average absolute relative error: Table 8 shows the results. Figure 12 depicts scatter plots of the predicted vs. measured quantities, having the smallest (ratio of proximal to distal resistance in the systemic circulation) and the largest (systemic resistance) Pearson correlation.

7.1.3. X-Ray Coronary Angiography Classification. For the binary classification problem, i.e., coronary angiography view recognition, the predicted value is the probability of a given input image to represent a right coronary artery (RCA). Thus, the threshold can be chosen to favor a certain problem specific behavior. It introduces flexibility in interpretation, allowing for a trade-off between false negative and false positive findings, by varying a probability threshold. The ROC (receiver operator characteristic) curve is widely used as a way of interpreting the performance of a classifier by describing the trade-off between true and false positives as the threshold is varied.
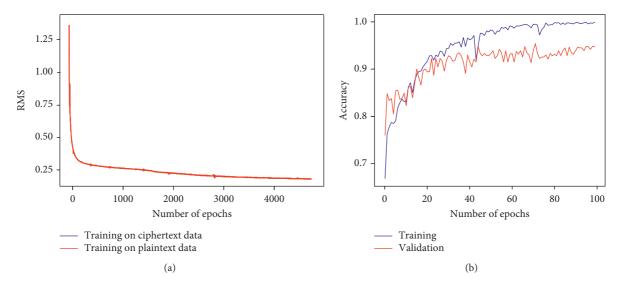
FIGURE 10: (a) Evolution of the training loss for encrypted and unencrypted networks: differences between the learning curves, caused by floating-point arithmetic, are unnoticeable. (b) Evolution of the accuracy when training on ciphertext data.

```
 (1) function TrainOnCiphertext( )
 (2)     X_train, Y_train ←LoadDataset
 (3)     X_train←Normalize (X_train)
 (4)     S←KeyGeneration
 (5)     X_train_enc←Encryption (X_train, S)
 (6)     Y_train_enc←Encryption (Y_train, S)
 (7)     BuildModel ()
 (8)     Train (X_train_enc, Y_train_enc)
 (9)     return model_enc
(10) end function
(11) function PredictOnCiphertext( )
(12)     X_test←LoadSamples
(13)     X_test←Normalize (X_test)
(14)     S←LoadKey
(15)     X_test_enc←Encryption (X_test, S)
(16)     LoadModel
(17)     Ỹ_test_enc←Predict (X_test_enc)
(18)     Ỹ_test←Decryption (Ỹ_test_enc, S)
(19)     return Ỹ_test
(20) end function
```

ALGORITHM 6: Deep learning-based analysis on ciphertext data.

To assess the accuracy of the coronary angiography view recognition model, the obtained ROC curve is shown in Figure 13. Table 9 lists the precision, recall, and $F1$-score for both the LCA and RCA labels. Figure 14 shows the confusion matrix, portraying measures of association between the true labels and the deep neural network predictions of LCA and RCA.

In the angiographic view classification use case, the CNN network trained on ciphertext data classified 96.2% of the samples correctly when evaluated on the held-out testing angiographies. When compared to the unencrypted model, accuracy was identical.

7.2. Execution Time.

All runtimes reported in the current section were measured on a machine equipped with an Intel(R) Xeon(R) CPU running at 2.10 GHz. The deep learning library which integrates the MORE encryption scheme was written in C++. The library is still under active development, with minimal multithreading support.

A detailed comparison of the runtime for each of the applications is given in Tables 10 and 11. All reported results were obtained by employing data parallelism (8 threads), both during training and inference. Although deep learning models run directly on MORE, homomorphically encrypted data are significantly slower (up to one order of magnitude)
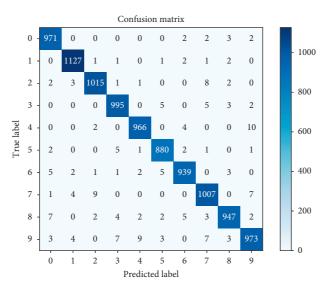
FIGURE 11: Confusion matrix of the MNIST digit classification task on the test set. The number on the diagonal indicates the number of correctly classified images, while the rest represent the misclassified ones.

```
(1)  function TrainOnPlaintext( )
(2)      X_train, Y_train ←LoadDataset
(3)      X_train ←Normalize(X_train)
(4)      BuildModel
(5)      Train(X_train, Y_train)
(6)      return model
(7)  end function
(8)  function PredictOnPlaintext( )
(9)      X_test ←LoadSamples
(10)     X_test ←Normalize(X_test)
(11)     LoadModel
(12)     Ỹ_test ←Predict(X_test)
(13)     return Ỹ_test
(14) end function
```

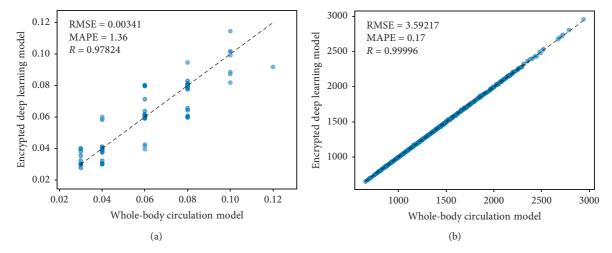ALGORITHM 7: Deep learning-based analysis on plaintext data.

FIGURE 12: Predicted vs. ground truth. (a) Ratio of proximal to distal resistance in the systemic circulation and (b) systemic resistance.
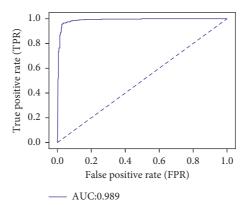
FIGURE 13: ROC curve of the view classification task in X-ray coronary angiography.

TABLE 2: Overview of databases used for experimental evaluation.

	MNIST	WBC	Angio
Normalization method	Min-max	Mean-std	Min-max
Input dimension	(28, 28)	(1, 9)	(256, 256)
Output dimension	(1, 10)	(1, 12)	(1, 1)
Data augmentation	—	—	Rotation/shifting/zooming
Number of training samples	50000	7000	1996 (7984*)
Number of validation samples	10000	1000	680
Number of testing samples	10000	2000	702

*After augmentation.
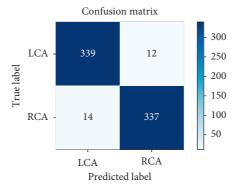
FIGURE 14: Confusion matrix of the X-ray coronary angiography view classifier.

TABLE 3: CNN-MNIST: the topology of the CNN for handwritten digit classification.

Layers	Parameters	Dimensions
Input	—	**(1,28,28)**
Convolution	(8,3,3)	(8,28,28)
Activation (sigmoid)	—	—
Average pooling	(2,2)	(8,14,14)
Convolution	(16,3,3)	(16,14,14)
Activation (sigmoid)	—	—
Average pooling	(2,2)	(16,7,7)
Flatten	—	(784)
Fully connected	100	(100)
Activation (sigmoid)	—	—
Fully connected	10	(10)
Activation (softmax)	—	—

TABLE 4: FCNN-WBC: the topology of the FCNN for hemodynamic analysis.

Layers	Parameters	Dimensions
Input	—	**(9)**
Fully connected	40	(40)
Activation (tanh)	—	—
Fully connected	40	(40)
Activation (tanh)	—	—
Fully connected	40	(40)
Activation (sigmoid)	—	—
Fully connected	12	(12)
Activation (linear)	—	—

TABLE 5: CNN-Angio: the topology of the CNN for view classification in X-ray coronary angiographies.

Layers	Parameters	Dimensions
Input	—	**(1,256,256)**
Convolution	(4,3,3)	(4,256,256)
Activation (sigmoid)	—	—
Average pooling	(2,2)	(4,128,128)
Convolution	(8,3,3)	(8,128,128)
Activation (tanh)	—	—
Average pooling	(2,2)	(8,64,64)
Convolution	(16,3,3)	(16,64,64)
Activation (tanh)	—	—
Average pooling	(2,2)	(16,32,32)
Convolution	(32,3,3)	(32,32,32)
Activation (tanh)	—	—
Average pooling	(2,2)	(32,16,16)
Flatten	—	(8192)
Fully connected	64	(64)
Activation (tanh)	—	—
Dropout	25%	—
Fully connected	—	(1)
Activation (sigmoid)	—	—

TABLE 6: Hyperparameters considered for the learning algorithms.

Hyperparameters	CNN-MNIST	CNN-Angio	FCNN-WBC
Objective function	CE	BCE	MSE
Weight initialization method	Xavier	Xavier	Xavier
Optimizer		Minibatch gradient descent	
Momentum	0.9	0.9	0.9
Learning rate	0.01	0.01	0.01
Batch size	32	16	32
Dropout rate	—	—	25%
Epochs	100	100	4500

TABLE 7: Precision, recall, and F1-score of the deep neural network based MNIST digit classification.

Digit	Precision (%)	Recall (%)	F1-score (%)
0	97.9	99.0	98.5
1	98.8	99.2	99.0
2	98.5	98.3	98.4
3	98.1	98.5	98.3
4	98.4	98.3	98.4
5	98.2	98.6	98.4
6	98.4	98.0	98.2
7	97.3	97.9	97.6
8	98.3	97.2	97.7
9	97.5	96.4	97.0
Average	*98.1*	*98.1*	*98.1*

TABLE 8: Results of the deep neural network for real-time hemodynamic analysis on the testing dataset.

Circulation	Parameters	MAPE (%)	Pearson correlation (%)
Systemic	Dead volume	7.03	0.9997
	Time at max. elastance	0.13	0.9995
	Resistance	0.17	0.9999
	Compliance	2.45	0.9867
Pulmonary	Dead volume	9.88	0.9991
	Time at max. elastance	0.10	0.9994
	Resistance	0.32	0.9998
	Compliance	0.67	0.9983

TABLE 9: Precision, recall, and F1-score of deep neural network for hemodynamic analysis.

Label	Precision (%)	Recall (%)	F1-score (%)
LCA	96.0	96.5	96.3
RCA	96.5	96.0	96.2
Average	*96.2*	*96.2*	*96.2*

TABLE 10: Runtime analysis: mean values and standard deviation of the encrypted and plaintext CNNs for MNIST digit recognition.

Operation	Runtime (s) on ciphertext data	Runtime (s) on plaintext data	Encrypted-unencrypted ratio
Data encryption and key generation	2.44 ± 0.016	—	—
Training (1 epoch)	444.59 ± 8.53	12.98 ± 1.17	34.25
Data encryption	0.39 ± 0.009	—	—
Inference (10 K images)	20.42 ± 0.32	0.54 ± 0.08	37.81
Data decryption	0.001 ± 0.0005	—	—

TABLE 11: Runtime analysis: mean values and standard deviation of the encrypted and plaintext networks for the two personalized medicine use cases.

Task	Operation	Runtime (s) on ciphertext data	Runtime (s) on plaintext data	Encrypted-unencrypted ratio
Angiographic view classification	Training (1 epoch)	1075.47 ± 45.54	34.48 ± 1.12	31.19
	Inference (702 images)	26.36 ± 1.98	0.8 ± 0.06	32.95
Whole-body circulation hemodynamic analysis	Training (1 epoch)	0.66 ± 0.09	0.021 ± 0.001	31.4
	Inference (2000 samples)	0.102 ± 0.01	0.006 ± 0.0009	17

during both training and inference, and the scheme is currently outstandingly faster compared to classic fully homomorphic encryption schemes where the difference is of around 6 to 7 orders of magnitude, even when performing very basic algebraic operations.

7.3. Security Concerns.
Despite the fact that the considered MORE encryption scheme has some advantages in terms of simplicity, clearness, and practicability, with properties tailored to privacy-preserving machine learning, it offers limited security compared to other HE schemes.

The most significant security concern is given by the linear nature of MORE [50, 51], whereas typical encryption schemes are based on strongly nonlinear functions and modular arithmetics over large numbers. This linearity may allow one to determine the secret key by having access to a large enough number of pairs of encrypted and unencrypted values. Otherwise stated, if a sufficiently large number of plaintext-ciphertext data pairs $\{C_i, m_i\}_{i=1}^{N}$ are available, the secret key S can be determined through an optimization problem. The key search attack can be formulated as finding the best fit of a matrix $S\prime$ such that $(S\prime^{-1} C_i S\prime)_{1,1} = m_i$ for each known pair s.

Although less secure than other homomorphic encryption schemes, the MORE scheme remains a viable solution for certain privacy-preserving applications. Consequently, it can be employed in scenarios where the key is never disclosed, e.g., ciphertext data are uploaded to an external computing service while the raw data remain private on the side of the data provider. For example, personal medical data can be uploaded to a dedicated service, i.e. patient-level encryption, that provides a personalized risk factor or other health-related indicators [81].

8. Discussion and Conclusions

The growing concern over recent years to preserve the privacy of sensitive patient health information, while promoting the development of personalized medicine, has increased the demand for cryptographic techniques suitable for addressing privacy-related issues in data-driven models.

The current work was focused on designing fully automatic data-driven personalization-based medicine solutions by protecting the integrity of patient health data. A variant of a noise-free matrix-based homomorphic encryption scheme (MORE) was proposed for privacy-preserving computations within deep learning models. Although a homomorphic cryptosystem is governed by a private key to encrypt and decrypt data, it greatly varies from other forms of encryption as it preserves the algebraic properties to allow a variety of operations to be performed directly on the encrypted data (ciphertext data) without requiring access to the decrypted information (plaintext data) or the encryption key.

We have showcased the applicability of incorporating the MORE encryption scheme into deep learning models by tackling three different problems: digit recognition, whole-body hemodynamic analysis, and coronary angiography view classification. The first application focused on a standard benchmarking application from the computer vision realm (MNIST digit recognition) to evaluate the feasibility of a network to operate directly on encrypted data, whereas the latter two models target clinically realistic problems. Therefore, one model was designed to estimate the outputs of a whole-body circulation (WBC) hemodynamic model. The second clinically realistic application was responsible for classifying encrypted X-ray coronary angiography medical images.

For each application, we have addressed both the training and the inference phase and showed that both can be performed on MORE homomorphically encrypted data. The reported results indicate that the proposed solution has great potential: (i) computational results are indistinguishable from those obtained with the unencrypted variants of the deep learning-based applications and (ii) runtimes increase only marginally. The encryption scheme incurs a reasonably small computational overhead and, importantly, allows for operations to be performed directly on floating-point numbers, which represents a critical property for artificial neural networks.

Although the MORE encryption scheme offers lower security compared to standard schemes, it is one of the few schemes with the potential to be used in practical applications. Consequently, it can be incorporated in two deep learning-based scenarios. The first scenario refers to the training of a model with encrypted input-target data and is suited for problems where the requirements are targeted on keeping both the data and the model private. This scenario implies that the unseen data on which the trained model will be applied will have to be encrypted with the same key as the data used for training. The second scenario covers the training of a model on plaintext input-target data, which can then be applied on encrypted data. This facilitates the use of existing deep learning models but also the idea of transferring the knowledge from one problem to another. Although the neural network will not be private, it will be applied on private data, resulting in private results, and more importantly will not be dependent on the key, e.g., if two data owners, with different keys, want to use a deep learning model trained on X-ray coronary angiographies, they can encrypt the data with the private key, feed the encrypted image to the network, and obtain encrypted results which will be interpreted only by the owner of the key.

In conclusion, we showed that a class of homomorphic methods based on linear transformations has a great potential towards facilitating data sharing and outsourcing to third parties for data analytics in regulated areas, but it comes at a cost of weaker security. The security compromise is caused by changing the original homomorphic encryption scheme to enable computations to be performed directly on rational numbers, a fundamental requirement for machine learning models. While the preliminary proposed solution is promising, for practical applications, further improvements are needed to strengthen the security of the scheme.

Data Availability

The MNIST database used to validate the methodology is available at http://yann.lecun.com/exdb/mnist/. The coronary angiography data used to support the findings of this study have not been made available because they were acquired in a research grant (heart.unitbv.ro) which does not allow the publication of the data.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

References

[1] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: review, opportunities and challenges," *Briefings in Bioinformatics*, vol. 19, no. 6, pp. 1236–1246, 2018.

[2] Z. Obermeyer and E. J. Emanuel, "Predicting the future—big data, machine learning, and clinical medicine," *New England Journal of Medicine*, vol. 375, no. 13, pp. 1216–1219, 2016.

[3] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, ser. CCS '15*, pp. 1310–1321, ACM, Denver, CO, USA, October 2015.

[4] A. Kipnis and E. Hibshoosh, "Efficient methods for practical fully homomorphic symmetric-key encrypton, randomization and verification," *IACR Cryptology ePrint Archive*, vol. 2012, p. 637, 2012.

[5] Innovation radar prize," 2019, https://ec.europa.eu/digital-single-market/en/news/winner-2019-innovation-radar-prize-announced.

[6] A. Vizitiu, C. I. Niţă, A. Puiu, C. Suciu, and L. M. Itu, "Privacy-preserving artificial intelligence: application to precision medicine," in *Proceedings of the 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 6498–6504, IEEE, Berlin, Germany, July 2019.

[7] A. Vizitiu, C. I. Niţă, A. Puiu, C. Suciu, and L. M. Itu, "Towards privacy-preserving deep learning based medical imaging applications," in *Proceedings of the 2019 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pp. 1–6, IEEE, Istanbul, Turkey, June 2019.

[8] P. Mohassel and Y. Zhang, "SecureML: a system for scalable privacy-preserving machine learning," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*, pp. 19–38, IEEE, San Jose, CA, USA, May 2017.

[9] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: scalable provably-secure deep learning," in *Proceedings of the 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, June 2018.

[10] M. S. Riazi, C. Weinert, O. Tkachenko et al., "A hybrid secure computation framework for machine learning applications," in *Proceedings of the ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, Abu Dhabi, UAE, 2017.

[11] P. Mohassel and P. Rindal, "Aby3: a mixed protocol framework for machine learning," in *Proceedings of the 2018 ACM Conference on Computer and Communications Security*, Toronto, Canada, October 2018.

[12] S. Wagh, D. Gupta, and N. Chandran, "SecureNN: efficient and private neural network training," *IACR Cryptology ePrint Archive*, vol. 2018, p. 442, 2018.

[13] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in *Proceedings of the 2017 ACM Conference on Computer and Communications Security*, Abu Dhabi, UAE, 2017.

[14] M. Abadi, A. Chu, I. J. Goodfellow et al., "Deep learning with differential privacy," in *Proceedings of the 2016 ACM Conference on Computer and Communications Security*, Vienna, Austria, October 2016.

[15] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: an application of human behavior prediction," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, 2016.

[16] N. Papernot, M. Abadi, Ú. Erlingsson, I. J. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," 2016, https://arxiv.org/abs/1610.05755.

[17] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson, "Scalable private learning with PATE," 2018, https://arxiv.org/abs/1802.08908.

[18] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *Proceedings of the 2017 International Conference on Learning Representations*, Toulon, France, April 2017.

[19] B. Ermis and A. T. Cemgil, "Differentially private variational dropout," 2017, https://arxiv.org/abs/1712.02629.

[20] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 909-910, Monticello, IL, USA, September 2015.

[21] E. Chou, J. Beal, D. Levy, S. Yeung, A. Haque, and L. Fei-Fei, "Faster cryptonets: leveraging sparsity for real-world encrypted inference," 2018, https://arxiv.org/abs/1811.09953.

[22] M. A. P. Chamikara, P. Bertók, I. Khalil, D. Liu, and S. Camtepe, "Local differential privacy for deep learning," 2019, https://arxiv.org/abs/1908.02997.

[23] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," in *Proceedings of the 2010 Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Monaco, France, May 2010.

[24] A. A. Badawi, J. Chao, J. Lin et al., "The alexnet moment for homomorphic encryption: HCNN, the first homomorphic CNN on encrypted data with GPUs," 2018, https://arxiv.org/abs/1811.00778.

[25] R. Gilad-Bachrach, N. Dowlin, K. Laine et al., "Cryptonets: applying neural networks to encrypted data with high throughput and accuracy," in *Proceedings of the 33rd International Conference on Machine Learning*, New York, NY, USA, 2016.

[26] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: deep neural networks over encrypted data," 2017, https://arxiv.org/abs/1711.05189.

[27] W. Liu, F. Pan, X. A. Wang, Y. Cao, and D. Tang, "Privacy-preserving all convolutional net based on homomorphic encryption," in *Proceedings of the 2018 International Conference on Network-Based Information Systems*, Bratislava, Slovakia, September 2018.

[28] C. Orlandi, A. Piva, and M. Barni, "Oblivious neural network computing via homomorphic encryption," *EURASIP Journal on Information Security*, vol. 2007, pp. 1–11, 2007.

[29] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *IACR Cryptology ePrint Archive*, vol. 2017, p. 35, 2017.

[30] J. W. Bos, K. E. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," *IACR Cryptology ePrint Archive*, vol. 2013, p. 75, 2013.

[31] M. Barni, C. Orlandi, and A. Piva, "A privacy-preserving protocol for neural-network-based computation," in *Proceedings of the 8th Workshop on Multimedia and Security*, Genenva, Switzerland, September 2006.

[32] M. Chase, R. Gilad-Bachrach, K. Laine, K. E. Lauter, and P. Rindal, "Private collaborative neural network learning," *IACR Cryptology ePrint Archive*, vol. 2017, p. 762, 2017.

[33] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: a low latency framework for secure neural network inference," in *Proceedings of the 2018 27th USENIX Security Symposium*, Baltimore, MD, USA, August 2018.

[34] A. El-Yahyaoui and M. D. Elkettani, "Fully homomorphic encryption: state of art and comparison," *International Journal of Computer Science and Information Security*, vol. 14, no. 4, p. 159, 2016.

[35] S. S. Sathya, P. Vepakomma, R. Raskar, R. Ramachandra, and S. Bhattacharya, "A review of homomorphic encryption libraries for secure computation," 2018, https://arxiv.org/abs/1812.02428.

[36] Microsoft SEAL (release 3.2)," 2019, https://github.com/Microsoft/SEAL.

[37] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *IACR Cryptology ePrint Archive*, vol. 2012, p. 144, 2012.

[38] J. H. Cheon, A. Kim, M. Kim, and Y. S. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proceedings of the 2017 International Conference on the Theory and Application of Cryptology and Information Security*, Hong Kong, China, December 2017.

[39] S. Halevi and V. Shoup, "Algorithms in helib," *IACR Cryptology ePrint Archive*, vol. 2014, p. 106, 2014.

[40] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, Cambridge, MA, USA, January 2012.

[41] J. Mancuso, "Privacy-preserving machine learning 2018: a year in review," 2018, https://medium.com/dropoutlabs/privacy-preserving-machine-learning-2018-a-year-in-review-b6345a95ae0f.

[42] R. A. Popa, "Building practical systems that compute on encrypted data," Doctoral thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2014.

[43] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the 1999 Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Prague, Czech Republic, May 1999.

[44] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[45] S. Goldwasser and S. Micali, "Probabilistic encryption and how to play mental poker keeping secret all partial information," in *Proceedings of the 1982 STOC*, San Francisco, CA, USA, May 1982.

[46] C. Bösch, P. H. Hartel, W. Jonker, and A. Peter, "A survey of provably secure searchable encryption," *ACM Computing Surveys*, vol. 47, no. 2, pp. 1–51, 2014.

[47] R. K. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order-preserving encryption for numeric data," in *Proceedings of the 2004 SIGMOD Conference*, Paris, France, June 2004.

[48] P. V. Parmar, S. B. Padhar, S. N. Patel, N. I. Bhatt, and R. H. Jhaveri, "Survey of various homomorphic encryption algorithms and schemes," *International Journal of Computer Applications*, vol. 91, no. 8, pp. 26–32, 2014.

[49] H. Chung and M. Kim, "Encoding rational numbers for fhe-based applications," *IACR Cryptology ePrint Archive*, vol. 2016, p. 344, 2016.

[50] D. Vizár and S. Vaudenay, "Cryptanalysis of chosen symmetric homomorphic schemes," in *Proceedings of the CRYPTO 2014*, Santa Barbara, CA, USA, August 2014.

[51] B. Tsaban and N. Lifshitz, "Cryptanalysis of the more symmetric key fully homomorphic encryption scheme," *Journal of Mathematical Cryptology*, vol. 9, no. 2, pp. 75–78, 2014.

[52] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biology*, vol. 52, no. 1-2, pp. 99–115, 1988.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[54] O. Russakovsky, J. Deng, H. Su et al., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[55] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Salt Lake City, UT, USA, June 2018.

[56] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[57] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, MA, USA, 2016, http://www.deeplearningbook.org.

[58] G. Urban, K. Geras, S. E. Kahou et al., "Do deep convolutional nets really need to be deep and convolutional?" in *Proceedings of the ICLR 2016*, San Juan, PR, USA, May 2016.

[59] T. A. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao, "Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review," *International Journal of Automation and Computing*, vol. 14, no. 5, pp. 503–519, 2017.

[60] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the 2015 IEEE Conference on computer vision and pattern recognition (CVPR)*, pp. 1–9, Boston, MA, USA, June 2015.

[61] F. Altaf, S. M. S. Islam, N. Akhtar, and N. K. Janjua, "Going deep in medical image analysis: Concepts, methods, challenges, and future directions," *IEEE Access*, vol. 7, pp. 99540–99572, 2019.

[62] D. Yu, F. Seide, and G. Li, "Conversational speech transcription using context-dependent deep neural networks," in *Proceedings of the ICML 2011*, Bellevue, WA, USA, 2011.

[63] B. Schoelkopf, J. C. Platt, and T. Hofmann, "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems*, Springer, Berlin, Germany, 2007.

[64] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy, 2010.

[65] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, https://arxiv.org/abs/1502.03167.

[66] G. Litjens, T. Kooi, B. E. Bejnordi et al., "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[67] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[68] L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141-142, 2012.

[69] V. Mihalef, L. Itu, T. Mansi, and P. Sharma, *Lumped Parameter Whole Body Circulation Modelling*, Springer International Publishing, Cham, Switzerland, 2017.

[70] S. G. Shroff, J. S. Janicki, and K. T. Weber, "Evidence and quantitation of left ventricular systolic resistance," *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 249, no. 2, pp. H358–H370, 1985.

[71] L. M. Itu, P. Sharma, B. Georgescu, A. Kamen, C. Suciu, and D. Comaniciu, "Model based non-invasive estimation of PV loop from echocardiography," in *Proceedings of the 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 6774–6777, Chicago, IL, USA, August 2014.

[72] L. Itu, S. Rapaka, T. Passerini et al., "A machine-learning approach for computation of fractional flow reserve from coronary computed tomography," *Journal of Applied Physiology*, vol. 121, no. 1, pp. 42–52, 2016.

[73] D. Mann, D. Zipes, P. Libby, and R. Bonow, *Braunwald's Heart Disease E-Book: A Textbook of Cardiovascular Medicine*, Elsevier Health Sciences, Amsterdam, Netherlands, 2014, https://books.google.de/books?id=1R44BAAAQBAJ.

[74] T. J. Ryan, "The coronary angiogram and its seminal contributions to cardiovascular medicine over five decades," *Circulation*, vol. 106, no. 6, pp. 752–756, 2002.

[75] V. G. Ng and A. J. Lansky, "Novel QCA methodologies and angiographic scores," *The International Journal of Cardiovascular Imaging*, vol. 27, no. 2, pp. 157–165, 2010.

[76] P. A. L. Tonino, B. De Bruyne, N. H. J. Pijls et al., "Fractional flow reserve versus angiography for guiding percutaneous coronary intervention," *New England Journal of Medicine*, vol. 360, no. 3, pp. 213–224, 2009.

[77] S. Tu, E. Barbato, Z. Köszegi et al., "Fractional flow reserve calculation from 3-dimensional quantitative coronary angiography and TIMI frame count: a fast computer model to quantify the functional significance of moderately obstructed coronary arteries," *JACC: Cardiovascular Interventions*, vol. 7, no. 7, pp. 768–777, 2014.

[78] M. Tröbs, S. Achenbach, J. Röther et al., "Comparison of fractional flow reserve based on computational fluid dynamics modeling using coronary angiographic vessel morphology versus invasively measured fractional flow reserve," *The American Journal of Cardiology*, vol. 117, no. 1, pp. 29–35, 2016.

[79] A. Arbab-Zadeh, "What will it take to retire invasive coronary angiography," *JACC: Cardiovascular Imaging*, vol. 9, no. 5, pp. 565–567, 2016.

[80] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proceedings of the 7th International Conference on Document Analysis and Recognition*, pp. 958–963, Edinburgh, UK, August 2003.

[81] J. Hippisley-Cox, C. Coupland, J. P. Robson, and P. Brindle, "Derivation, validation, and evaluation of a new qrisk model to estimate lifetime risk of cardiovascular disease: cohort study using qresearch database," *BMJ*, vol. 341, p. c6624, 2010.