Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 232 (2024) 2201–2212

Procedia Computer Science

www.elsevier.com/locate/procedia

5th International Conference on Industry 4.0 and Smart Manufacturing

Deep Homeomorphic Data Encryption for Privacy Preserving Machine Learning

Vagan Terziyan [a,*], Bohdan Bilokon [b], Mariia Gavriushenko [a]

[a] *Faculty of Information Technology, University of Jyväskylä, 40014, Jyväskylä, Finland*
[b] *Department of Artificial Intelligence, Kharkiv National University of Radio Electronics, 61166, Kharkiv, Ukraine*

Abstract

Addressing privacy concerns is critical in smart manufacturing where sensitive data is used for machine learning. Data protection is essential to ensure model accuracy while upholding data privacy. Homeomorphic encryption, an algorithm for privacy-preserving machine learning, achieves this by transforming data using a neural network with secret key weights. This process conceals private data while retaining the potential to learn classification models from the anonymized data. This paper introduces a comprehensive quality metric to assess homeomorphic encryption across conflicting criteria: security (regarding private data), machine learning adaptability (tolerance), and efficiency (regarding needed extra resources). Through experiments on various datasets, the metric proves its effectiveness in guiding optimal encryption parameter selection. Our findings highlight homeomorphic encryption's strong overall quality, positioning it as a valuable Industry 4.0 solution. By offering a simpler alternative to fully homomorphic encryption, it effectively addresses privacy concerns and enhances data usability in the context of smart manufacturing.

Keywords: Smart manufacturing; data privacy; privacy-preserving machine learning; quality metric; homeomorphic encryption

1. Introduction

In his famous book [1], among the 21 challenges of our century, Harari stressed the power of data, which is becoming "the currency of the future". Should we care about huge and detailed information collected from and about us (physical or biological censors, cameras, texts, videos, user profiles, etc.)? Are the benefits from personalized technologies we are using every day worth of being "naked" in front of potential "digital dictatorships"? What would be the reasonable regulations regarding personal data collection and use so that, while obtaining useful value from

* Corresponding author. *E-mail address:* vagan.terziyan@jyu.fi

data, we will not lose our freedom for privacy (aka "Big Brother" scenarios)?

Security and privacy issues, as pointed out in [2], are of great importance to Industry 4.0 and smart manufacturing. Due to the full use of large-scale machine-to-machine communication, the internet of things, and machine learning to increase automation, improve communication, self-monitoring, diagnostics, and predictive maintenance (without human intervention but with massive amount of data), security and privacy issues have become emergent. Important decisions are made automatically based on such data [3]. Without strong cryptographic assurance, resource-limited devices could be easily compromised and hacked by adversaries. Therefore, assuring the safety and privacy of industrial data, data-driven products and decisions is a top concern for industry 4.0 nowadays.

The General Data Protection Regulation (GDPR), which came into force in the European Union in May 2018, has been an attempt to meet current challenges related to personal data protection [4]. While the basic GDPR principles were welcomed, two of them, namely the right to withdraw consent and the right to be forgotten, caused controversy within the academy, business, and manufacturing.

A special challenge for the GDPR is the domain of machine learning (ML). What would be a smart way to formulate and apply the regulations to ML with personal or sensitive industrial data? This challenge emerged as a new field, which is Privacy-Preserving Machine Learning (PPML), aiming to bridge the gap between ML and privacy communities [5]. The "right to be forgotten" assumes the ability to request deletion of personal data, including training records used to train ML models. However, keeping trained models instead of training data does not guarantee complete privacy protection. Even (deep) neural network (aka "black-boxes") models are vulnerable to various information leaking attacks (model inversion, membership inference, etc.), which may determine the presence and content of private examples in the training data [6]. A special threat of potential privacy violations and model stealing by third parties [7] is the case when ML is provided as a service. Hence, ML concerns encompass safeguarding ML models along with training data to prevent private data leakage, while ensuring these secured models maintain the necessary accuracy for their intelligent tasks like classification, regression, prediction, etc.

PPML is all about a trade-off between security of ML processes (training data and models' privacy and their secure use), efficiency and effectiveness of the secured models. In [8], the need is pointed out to ensure that ML models are trustworthy and, therefore, we must have some way (e.g. a metric) to evaluate these models based on their mutually competing features, such as security, privacy, fairness and accuracy, among others. Measuring the values of security and privacy threats as well as the values of fitness regarding security and privacy requirements for some developed systems are known to be the hard tasks [9]. Evaluating the relevance, performance and efficiency of the ML models is also a multidimensional and non-trivial task [10] Therefore, introducing some trade-off metrics between these concerns will be a far more challenging task [11].

In this paper, we suggest and justify a metric for quality assessment of PPML encryption instruments, which is based on a trade-off between the requirements: (a) to guarantee stronger security/privacy of encrypted data and models ("security" component of the quality metric); (b) to minimize the loss of performance of the models trained on encrypted data ("ML tolerance" component of the quality metric); (c) to minimize the extra time needed to train models on encrypted data ("efficiency" component of the quality metric). This quality metric will be introduced following an example related to homeomorphic encryption [12], [13] with experiments on particular datasets.

The rest of the paper is organized as follows: Section 2 describes PPML encryption methods (homomorphic vs. homeomorphic), each with its own specifics; Section 3 introduces the PPML quality metric with detailed analytics; Section 4 summarizes the experiments of applying the metric to the homeomorphic encryption of the popular datasets; and we conclude in Section 5.

2. Homomorphic vs. Homeomorphic Encryption of Private Data

PPML could be achieved by either getting rid of private data completely by: (A) generation of new (synthetic) data with a similar distribution to the original one [14]; or (B) by applying various privacy-preserving cryptography protocols [15], transforming the original data by secure multi-party computation [16], or homomorphic encryption [17]. In both cases, the required GDPR protection is supposed to be provided and the new (generated, transformed, anonymized, encrypted) data is supposed to be useful to train various classifiers. In this section, we will focus more on the transformation (anonymization, encryption) of private data for PPML purposes.

2.1. Homomorphic encryption as a homomorphic transformation

In mathematics, a *group* $(G,\circledast)$ is a numerical set G and an operation $\circledast$ that combines any two numeric instances of the set to produce a third numeric instance of the set; assuming that the operation is associative, has an identity element $\boldsymbol{e}$, and inverse g_i^{-1} exists for each g_i instance of the set, i.e.:

$$\forall g_i, g_j, g_k \in G, (g_i \circledast g_j) \circledast g_k = g_i \circledast (g_j \circledast g_k);$$

$$\forall g_i \in G, g_i \circledast e = e \circledast g_i = g_i;$$

$$\forall g_i \exists g_i^{-1}, g_i \circledast g_i^{-1} = g_i^{-1} \circledast g_i = e.$$

Given two groups, $(G,\circledast)$ and $(H,\odot)$; a *homomorphism* (or *homomorphic transformation*) from $(G,\circledast)$ to $(H,\odot)$ is a (transformation) function $F : G \to H$ such that:

$$\forall g_i, g_j \in G, F(g_i \circledast g_j) = F(g_i) \odot F(g_j).$$

Here one may say that group H (after the homomorphic transformation) has a similar algebraic structure as group G (before the transformation), i.e., the homomorphic transformation F preserves that similarity.

Therefore, homomorphism can be used as an algebraic term for a transformation function preserving some algebraic operations after the transformation, i.e.:

$$\forall g_i, g_j, g_k \in G, (g_i \circledast g_j = g_k) \to (F(g_i) \odot F(g_j) = F(g_k)).$$

A prevalent form of homomorphic transformation, where a secret key acts as the transformation function F, is termed *homomorphic encryption*. This method safeguards sensitive data's privacy. The core concept is that specific algebraic manipulations on encrypted data can occur for third parties without revealing original private data values. Furthermore, these manipulations' outcomes can be decrypted to yield meaningful results for the original data owner. Partially homomorphic cryptosystems facilitate homomorphic calculations for a single operation, such as addition or multiplication. A more robust cryptosystem that accommodates both addition and multiplication is referred to as fully homomorphic encryption. Homomorphic encryption's overarching goal is to ensure privacy for outsourced storage and computation within commercial cloud environments [18].

2.2. Homeomorphic encryption as a homeomorphic transformation

Morphism in mathematics is known to be a generic structure preserving (and reversible in the case of *isomorphism*) transformation of some abstract object. As we have seen above, homomorphism is such an isomorphism that preserves the algebraic structure of the object. However, there could be another (similar to isomorphism) concept called *homeomorphism*, which preserves the topological structure of the object. Topology is a mathematical field that studies homeomorphism, which is a continuous function (aka structure-preserving map) between multidimensional numeric data spaces (like topological spaces) that has a continuous inverse function. Topological space (which is rather geometrical than algebraic) is such a space where closeness is defined but not necessarily measured or computed. Such space is considered as a set of points (samples, elements) with the structure (neighborhoods, boundaries, etc.). Topological space is known to be an *n*-dimensional data manifold (according to popular ML terminology), which resembles Euclidean space in the neighborhood of each point. Homeomorphism (or topology preserving transformation) is a function between two manifolds, which sends points that are close to one another in the first manifold to points that are close to one another in the second manifold, and there exists an inverse function, which can send transferred points back to their original places. Therefore, we are talking about such continuous deformations (stretching and bending but not breaking or gluing) of the object into a new shape that leave its topological structure intact.

Let us define a Supervised ML (SML) group (D, L^k, M^D) as a numerical multidimensional data space (*n*-dimensional data manifold) D, each point (data sample defined by the vector of *n* numerical values) of which could get a unique label from the set of k labels L^k using the labelling operation (model) M^D, i.e., $\forall d_i \in D, \exists! \, l_j \in L^k, M^D(d_i) = l_j$. Given two SML groups, (D, L^k, M^D) and $(\tilde{D}, L^k, M^{\tilde{D}})$, which share the same labels' set L^k; a *homeomorphism* (or *homeomorphic transformation*) from and $(\tilde{D}, L^k, M^{\tilde{D}})$ is a function $F : D \to \tilde{D}$ such that:

$\forall d_i \in D, F(d_i) = \tilde{d}_i$; $\forall \tilde{d}_i \in \tilde{D}, F^{-1}(\tilde{d}_i) = d_i$; $\forall d_i \in D, \exists! \, l_j \in L^k, M^D(d_i) = l_j = M^{\tilde{D}}(\tilde{d}_i)$, which means that homeomorphic transformation of data (and, therefore, of the corresponding model) does not change the result of labelling for any of the data samples.

Such a transformation does not directly require preserving the algebraic operations with the data (like in the case of homomorphic transformation), but instead, it requires preserving the labelling topology (geometry of distribution and decision boundaries between k data manifolds) of the data space (as a topological space). Therefore, we will refer to such a transformation as a homeomorphic data space transformation.

The connection between the topology domain and a neural network (NN) concept has been noticed in [19] where the relationship between topological transformations and deep neural network learning has been described. The observation is as follows: each layer of a deep neural network with a continuous activation function "stretches" and "squishes" the space (defined in the very beginning by the input vector of the original data values), but it never "cuts", "breaks", or "folds" it. This means that topological properties of the original data space are preserved during data propagation through several hidden layers of the neural network. The author proved this observation as a theorem, which states that each neural network layer with the same number of inputs and outputs and continuous activation functions for the neurons is a homeomorphism, if the corresponding square weight matrix is invertible (i.e., non-singular).

In [12], we suggested *homeomorphic encryption* as a homeomorphic transformation of labelled data with the secret transformation parameters as encryption keys. The intended objective was to explore the possibility to use such encrypted data for PPML and particularly for safe supervised learning at third party servers (clouds). We performed a topology-preserving transformation of the original data as a kind of anonymization technique, which uses deep feedforward neural network (aka perceptron) to hide (by topology-preserving replacement) the coordinates (numeric attribute values) of sensitive or private data samples but still enables potential classification model to be learned on the basis of the anonymized data. Such homeomorphic encryption applied on top of an n-dimensional data manifold performs appropriate secret-key $n \times n$ non-singular matrices (neural network weights) multiplication and applies invertible activation function at each layer. After several layers of such transformation, according to [19], we will still keep the topologically equivalent (to the original one) data manifold, which hides private values of the data but preserves the labelling structure of the data needed to build the correct classification models.

Encryption, which involves homomorphic, homeomorphic or other techniques that fit for the data transformation purpose, is supposed to be done within secure private space before the data is send for further processing elsewhere. It is supposed that the original private data is not recoverable from the encrypted one without knowing the encryption keys. Encrypted data can be placed within the (potentially unsafe) public space (cloud), which is supposed to provide ML as a service (MLaaS) analytics. The encryption must be done so that the MLaaS would take it as some business-as-usual task input without even knowing what kind of data it processes and that it is encrypted. MLaaS will train the classification model (e.g., deep neural network) and test it using the encrypted data for both training and testing purposes. Because the data from which the model has been built was encrypted, the model itself can also be queried with encrypted queries only. Therefore, such a model (if stolen) would be useless without knowing the encryption key. The model can be safely kept within the public space (cloud) and queried remotely from the secure private space with the encrypted queries. If the original private data is meant only for the purpose of a particular classification model design task, then such data can be deleted after the model is built to avoid leakage or hacking (of data or the encryption keys) within the private space. This ensures security worries can be avoided and the original data is protected.

Therefore, homomorphic encryption is usually a complex procedure, which is applied to potentially different and unknown during the encryption phase purposes use cases. These use cases involve protected private data and include PPML just as one of possible options [20]. In contrast, the homeomorphic encryption (a quite simple and efficient procedure) can be used when we know that the main and sole purpose of the collected private data is a supervised ML (e.g., classification model building). However, in both cases, the concerns for the PPML are similar:

- *We need a high level of protection (security) provided by encryption.*
- *We want to have the as small as possible (if any) loss of classification accuracy of the model built from the encrypted data comparably to the one built from the original data.*
- *We want to have as small as possible complexity increase (and, therefore, appropriate processing time) of the model built from the encrypted data comparably to the one built from the original data.*

There is a definite need to create a quality metric capable of using these mutually conflicting criteria to assess the quality of encryption regarding the particular data and intended model. We are going to suggest one for homeomorphic encryption in the following section.

3. Quality Metric for Homeomorphic Encryption

In this section, we are going to present the metric to assess the quality of anonymization procedures (particularly homeomorphic encryption) by giving the same importance to the safety of private data after anonymization and to the preserved information of the anonymized data for potential ML on top of it. Safety (security) is provided by replacing the data samples as much as possible from their original places within the attributes' space. The original data should not be recoverable, ensuring that the models (potentially built on anonymized data by ML techniques) will be useless for anyone who steals them. Both of these conditions ("to be well replaced" and "to be well protected") will be assessed within the *security* component of the anonymization quality metric. The value for ML will depend, on the one hand, on the loss of accuracy of potential ML models after applying the anonymization to the original data, and, on the other hand, it will depend on additional resources (models' complexity) needed for ML to deal with the anonymized data. The first (major importance) of these two conditions will be assessed within the *ML tolerance* component of the anonymization quality metric, while the second (minor importance) of the two conditions will be assessed within the *ML efficiency* component of the anonymization quality metric. Let us consider the details of constructing the metric.

Assume we have a dataset D for supervised ML, particularly classification, containing k samples that have n attributes and labelled into m different class categories.

We suppose to introduce and experimentally test the intended quality metric based on real data. The generic schema of our experiments for this purpose is shown in Fig. 1.

Firstly, during our experiments, we configured and trained the neural network architecture, which will provide the best classification accuracy on top of the testing subset of the original data after learning. Let us assume that the architecture of the best preforming (on the original data) feedforward NN (perceptron) has the following total number of neurons: $n + m + H_{\text{before}}$, where: H_{before} is the number of hidden neurons within the best NN architecture used with the original data (before anonymization).

Secondly, during our experiments, we anonymized the original dataset (which had been named D before and became $\widetilde{D}$ after the anonymization) using different anonymization depths d (natural numbers from 1 to ∞). Therefore, the anonymizer here is another deep feedforward NN (aka perceptron), whose parameters (weights and biases are chosen as secret keys and frozen). After the frozen anonymization layers, another NN architecture will be configured and trained, which will provide the best classification accuracy on top of the testing subset of the anonymized data after learning. Let us assume that the architecture of the best preforming (on the anonymized data) feedforward NN (perceptron) has the following total number of neurons: $n + m + H_{\text{after}}$, where: H_{after} is the number of hidden neurons within the best NN architecture used with the anonymized data (i.e., after the original data has been anonymized).

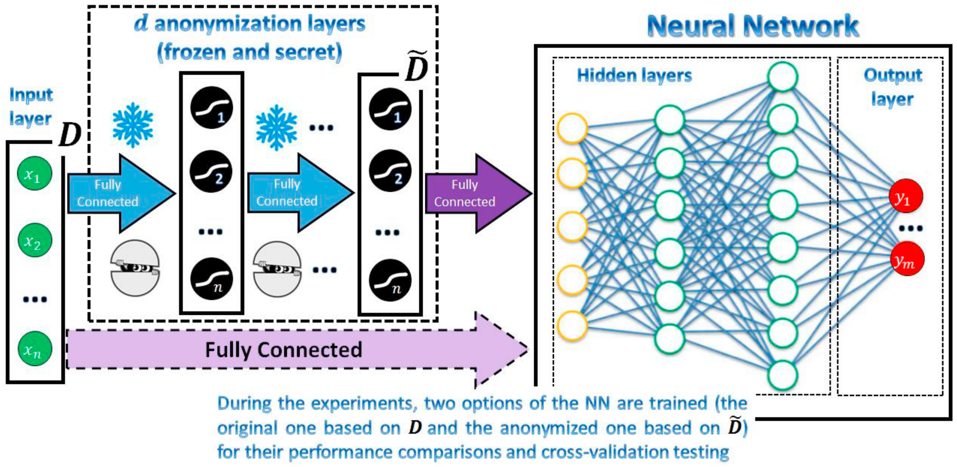
d anonymization layers (frozen and secret)
Neural Network
Input layer
D
D̃
Fully Connected
Fully Connected
Fully Connected
Hidden layers
Output layer
x1
x2
xn
y1
ym
Fully Connected
During the experiments, two options of the NN are trained (the original one based on D and the anonymized one based on D̃) for their performance comparisons and cross-validation testing

Fig. 1. Generic schema of the experiments with homeomorphic encryption. Experiments provide two sets of performance values to compare the NN trained on original data with another NN trained on anonymized data. Therefore, the first NN takes the input values for training from the original dataset, and the second NN takes the input from the anonymized values. The anonymization procedure here is propagation of the data through the anonymization layers (deep NN with frozen secret weights).

Our objective is to estimate (measure, compute) the quality of the anonymization algorithm for the particular dataset, taking into account several quality criteria (security, ML tolerance, and efficiency).

Assume that the training experiments with both NNs give us the following measurements as the outcomes:

$A_{D|D}$ – testing accuracy for the case when the classifier is both trained and tested on the original "clean" data (i.e., a certain subset of the original data is used for training and the rest for testing);

$A_{\widetilde{D}|\widetilde{D}}$ – testing accuracy for the case when the classifier is both trained and tested on the modified "anonymized" data (i.e., a certain subset of the anonymized data is used for training and the rest for testing);

$A_{D|\widetilde{D}}$ – testing accuracy for some kind of "cross-validation", or the case when the classifier is trained on the original "clean" data but tested on the corresponding modified "anonymized" data (i.e., all the original data is used for training and all the anonymized data is used for testing);

$A_{\widetilde{D}|D}$ – testing accuracy for another kind of "cross-validation", or the case when the classifier is trained on the modified "anonymized" data and tested on the corresponding original "clean" data (i.e., all the anonymized data is used for training and all the original "clean" data is used for testing).

Let us introduce the intermediate (anonymization) quality indicators: d (depth of anonymization), which value is supposed to be the more the better for the anonymization quality;

NOTE: The depth of anonymization (aka indicator for the data being "*protected*") contributes to the *security* component of the anonymization quality. Each additional layer of the anonymization network adds at least n^2 (more precisely $n \cdot (n+1)$ if secret biases are taken into account) secret keys (weights of the anonymization neural network), making potential hacking of the original data much more complicated;

$a = \left|A_{D|D} - A_{D|\widetilde{D}}\right|$ (the greater value the better for the anonymization quality); **(1)**

NOTE: This parameter is based on cross-validation and assesses how much replacement of the original data has been made by the anonymization procedure. Assume that we have built a model from the original data, test the model on the original data and achieve a certain accuracy $A_{D|D}$ during testing. Assume that now we will use the anonymized data $\widetilde{D}$ to test this model. If the anonymized data samples from $\widetilde{D}$ are placed differently enough from their original locations in D, which is good for data security, then the corresponding accuracy $A_{D|\widetilde{D}}$ during testing must be essentially different (lower) from $A_{D|D}$. Therefore, parameter a indicates the quality (its security component) of a particular anonymization;

$b = \left|A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}\right|$ (the greater value the better for the anonymization quality); **(2)**

NOTE: Similarly, to a, parameter b is based on cross-validation and also assesses how much replacement of the original data has been made by the anonymization procedure. Assume that we have built a model from the anonymized data, test the model on the anonymized data and achieve certain accuracy $A_{\widetilde{D}|\widetilde{D}}$ during testing. Assume that now we will use the original data D to test this model. If the original data samples from D are placed differently enough from their new locations in $\widetilde{D}$, which is good for data security, then the corresponding accuracy $A_{\widetilde{D}|D}$ during testing must be essentially different (lower) from $A_{\widetilde{D}|\widetilde{D}}$. Therefore, parameter b also indicates the quality (its security component) of a particular anonymization;

NOTE: Fig. 2 contains a simple example of a weak data replacement and, therefore, it demonstrates the importance of the cross-validation check mentioned above. Fig. 2(a) shows four original data samples of two classes (different colors) and an approximate decision boundary between the two classes, which clearly separates the classes according to some ML model. Assume that we apply three layers of homeomorphic encryption over the data, which means that we three times replace the data samples according to the weights (encryption keys) of the corresponding encryption neural network (Fig. 2(b)). However, we can see that, after such a complex replacement, the encrypted data samples accidentally appear close to their original places. Accordingly, the new learned decision boundary, which separates the encrypted data samples (Fig. 2(c)), will be close to the original decision boundary (Fig. 2(d)). This is bad for encryption security, and, therefore, this will be indicated by small values of the quality indicators a and b as the result of cross-validation.

$\gamma = \frac{\boldsymbol{2 \cdot a \cdot b}}{\boldsymbol{a + b}}$ – a harmonic mean (i.e., biased to the worst case scenario average) of a and b indicators; **(2*)**

NOTE: if $a = b = 0$, then $\gamma = 0$.

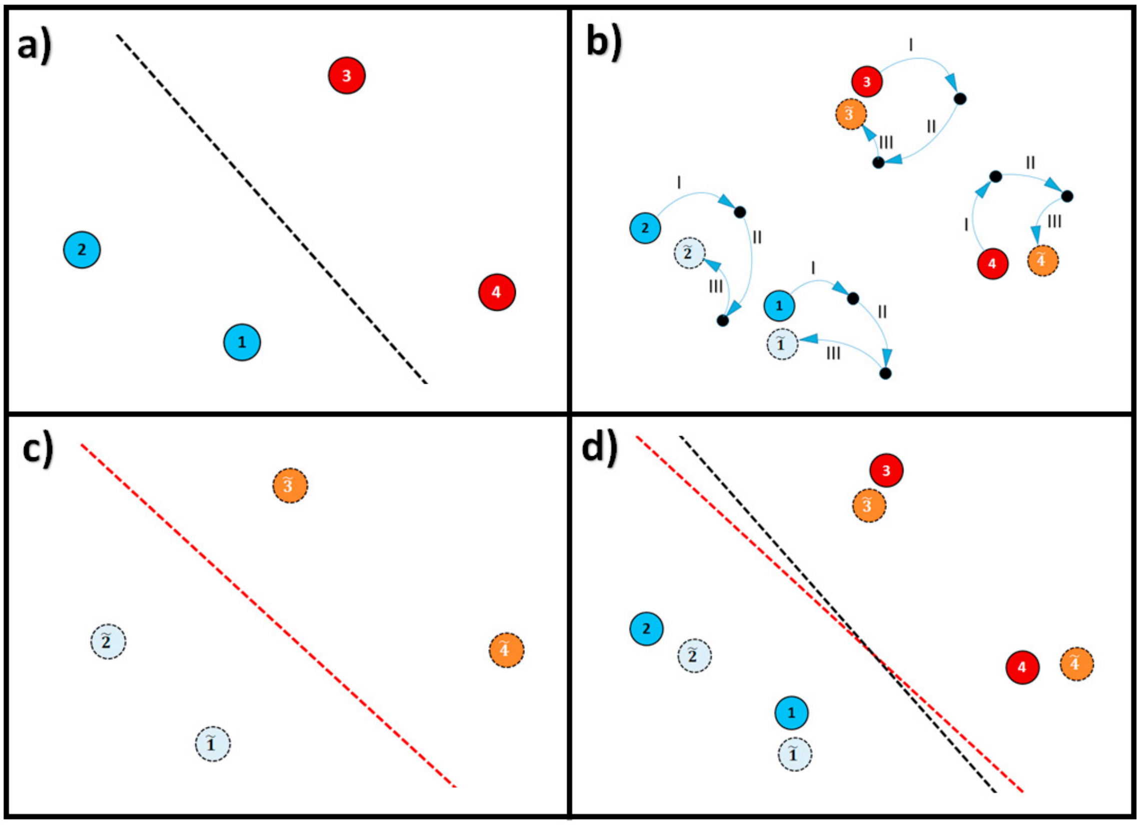

Fig. 2. The need for a cross-validation procedure to indicate weak homeomorphic encryption is visualized: (a) the original data samples and corresponding decision boundary between two classes; (b) replaced (with three layers of homeomorphic encryption) data samples occasionally appear close to their original places; (c) new decision boundary, which separates classes of the encrypted samples appears (d) to be close to the decision boundary for the original data, which indicates weak encryption.

NOTE: γ is a kind of indicator of how the data has been "*replaced*" (an important parameter of the *security* component of all the anonymization quality functions), which estimates how well the original data samples are replaced (hidden) from their original places;

$c = 1 - \left|A_{D|D} - A_{\widetilde{D}|\widetilde{D}}\right|$ (the greater value the better for the anonymization quality). **(3)**

$\delta = \frac{n+m+H_{\text{before}}}{n+m+H_{\text{after}}}$ (the greater value the better for the anonymization quality). **(4)**

The "SECURITY" (a major importance) component of the anonymization quality (which indicates how strongly the privacy of the original data is protected by the anonymization) benefits from both: the depth of anonymization d and the Harmonic Mean of both "cross-validation"-related quality indicators a and b as follows:

$$Q_{\text{Security}} = 1 - e^{-\gamma \cdot d} = 1 - e^{-\frac{2 \cdot a \cdot b \cdot d}{a+b}}\ . \quad \textbf{(5)}$$

NOTE: a more advanced option for Equation (5) would be the following one, which takes into account the number of instances k and the number of attributes n in the dataset:

$$Q^{*}_{\text{Security}} = 1 - e^{-\gamma \cdot d \cdot \frac{\ln n^2}{\ln k}} = 1 - e^{-\frac{2 \cdot a \cdot b}{a+b} \cdot d \cdot \frac{2 \cdot \ln n}{\ln k}}\ . \quad \textbf{(5*)}$$

The "SECURITY"* component of the anonymization quality (Equation (5*)) considers the importance of the anonymization depth d in the context (the second factor in the exponent) of the number of samples k and number of attributes n in the dataset. It gives an estimate of the likelihood of uncovering the remaining original data from D if an essential part of it has been leaked. Such a data hacking procedure would mean using the leaked data samples from D and their anonymized values $\widetilde{D}$ and solving the set of equations $\widetilde{D} = F(D, \vec{w})$, to get the vector of the anonymization neural network's weights $\vec{w}$ as the anonymization keys. One may see that, in addition to Equation (5), Equation (5*) adds the $\frac{\ln n^2}{\ln k}$ multiplier to the exponent as a kind of "defense vs. attack" opportunity ratio. "Attack opportunity" depends on the total number of samples k, which are being hunted for. "Defence opportunity" grows by adding anonymization layers, each of which brings at least n^2 additional keys to be uncovered. The logarithm function is added here to smooth the ratio and normalize the quality function to better fit the intended quality interval.

NOTE: another advanced option for Equation (5), with similar intuition as the one in Equation (5*), also takes into account the number of instances k and the number of attributes n in the dataset. It uses a negative power function instead of a logarithm to smooth the "attack opportunity" (driven by k and restricted by d). This advanced quality option is based on the following components:

$h = d \cdot n \cdot (n + 1)$ is the exact number of secret keys provided by d additional anonymization layer (i.e., an estimate for the "defence opportunity");

$K = k^{d^{-1}}$ is an estimate for the "attack opportunity";

$H = \frac{h}{h+K}$ is a special way to normalize the "*protected*" (regarding the data) indicator d to $[0,1]$ interval using k and n values as the characteristics of the dataset;

$$Q^{**}_{\text{Security}} = \sqrt{\gamma \cdot H} = \sqrt{\frac{2 \cdot a \cdot b}{a+b} \cdot \frac{d \cdot n \cdot (n+1)}{d \cdot n \cdot (n+1) + k^{d^{-1}}}}\,, \qquad \textbf{(5**)}$$

which is a geometric mean of the "replaced" (γ) and "protected" (H) indicators of security.

The "SECURITY"** component of the anonymization quality (Equation (5**)) is more sensitive to change of the anonymization depth d and the characteristics of the dataset (k and n), and, therefore, can be used as an alternative (when appropriate) to Equation (5) and Equation (5*).

Since Equation (5), Equation (5*), and Equation (5**) give different estimates for the security component of the anonymization quality, it will be useful to have also their average as follows:

$$\bar{Q}_{\text{Security}} = \frac{Q_{\text{Security}} + Q^{*}_{\text{Security}} + Q^{**}_{\text{Security}}}{3}\,. \qquad \textbf{(5}^{\dagger}\textbf{)}$$

The "ML TOLERANCE" (a moderate importance) component of the anonymization quality (which measures to what extent the accuracy is preserved after anonymization) is simply:

$$Q_{\text{ML Tolerance}} = c\,. \qquad \textbf{(6)}$$

The "EFFICIENCY" (a minor importance) component of the anonymization quality (which measures an additional resource (neurons) needed to deal with the anonymized data comparably to the original one) is as follows:

$$Q_{\text{Efficiency}} = \min(1, \delta). \qquad \textbf{(7)}$$

The final anonymization quality measure (which value belongs to $[0,1]$ interval, or [0-100%] interval) for the $D \to \widetilde{D}$ anonymization can be computed as a "Golden Ratio"-driven compromise between the three quality components as follows:

$$\mathrm{Q}(\mathrm{D} \to \widetilde{\mathrm{D}}) = \frac{1}{\varphi^2+\varphi+1} \cdot \left(\varphi^2 \cdot \mathrm{Q}_{\text{Security}} + \varphi \cdot \mathrm{Q}_{\text{ML Tolerance}} + \mathrm{Q}_{\text{Efficiency}}\right), \qquad \textbf{(8)}$$

where φ is a "Golden Ratio" constant ($\varphi = \frac{1+\sqrt{5}}{2} \approx 1.618$).

Taking into account the "Golden Ratio" properties, we have:

$Q(D \to \widetilde{D}) = \frac{1}{2 \cdot \varphi^2} \cdot \left(\varphi^2 \cdot Q_{Security} + \varphi \cdot Q_{ML\ Tolerance} + Q_{Efficiency}\right)$,

$Q(D \to \widetilde{D}) = 0.5 \cdot Q_{Security} + 0.5 \cdot (\varphi - 1) \cdot Q_{ML\ Tolerance} + (1 - 0.5 \cdot \varphi) \cdot Q_{Efficiency}$, and, finally:

$Q(D \to \widetilde{D}) \approx 0.5 \cdot Q_{Security} + 0.3 \cdot Q_{ML\ Tolerance} + 0.2 \cdot Q_{Efficiency}$, or in detail: **(9)**

$Q(D \to \widetilde{D}) \approx 0.5 \cdot \left(1 - e^{-\frac{2 \cdot a \cdot b \cdot d}{a+b}}\right) + 0.3 \cdot c + 0.2 \cdot min\,(1, \delta)$, and, therefore, in initial terms:

$$Q(D \to \widetilde{D}) \approx 0.5 \cdot \left(1 - e^{-\frac{2 \cdot \left|A_{D|D} - A_{D|\widetilde{D}}\right| \cdot \left|A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}\right| \cdot d}{\left|A_{D|D} - A_{D|\widetilde{D}}\right| + \left|A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}\right|}}\right) + 0.3 \cdot \left(1 - \left|A_{D|D} - A_{\widetilde{D}|\widetilde{D}}\right|\right) + 0.2 \cdot min\,\left(1, \frac{n+m+H_{before}}{n+m+H_{after}}\right). \qquad \textbf{(10)}$$

Three other options for Equation (10) would be the following ones where the security component of quality is computed based on Equation (5*), Equation (5**) or Equation (5†):

$$Q^{*}(D \to \widetilde{D}) \approx 0.5 \cdot \left(1 - e^{-\frac{2 \cdot |A_{D|D} - A_{D|\widetilde{D}}| \cdot |A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}|}{|A_{D|D} - A_{D|\widetilde{D}}| + |A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}|} \cdot \frac{2 \cdot d \cdot \ln n}{\ln k}}\right) + 0.3 \cdot (1 - |A_{D|D} - A_{\widetilde{D}|\widetilde{D}}|) + 0.2 \cdot min\,(1, \frac{n+m+H_{before}}{n+m+H_{after}}). \quad \mathbf{(10^{*})}$$

$$Q^{**}(D \to \widetilde{D}) \approx 0.5 \cdot \sqrt{\frac{2 \cdot |A_{D|D} - A_{D|\widetilde{D}}| \cdot |A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}|}{|A_{D|D} - A_{D|\widetilde{D}}| + |A_{\widetilde{D}|\widetilde{D}} - A_{\widetilde{D}|D}|} \cdot \frac{d \cdot n \cdot (n+1)}{d \cdot n \cdot (n+1) + k^{d^{-1}}}} + 0.3 \cdot (1 - |A_{D|D} - A_{\widetilde{D}|\widetilde{D}}|) + 0.2 \cdot min\,(1, \frac{n+m+H_{before}}{n+m+H_{after}}). \quad \mathbf{(10^{**})}$$

$$\bar{Q}(D \to \widetilde{D}) \approx 0.5 \cdot \bar{Q}_{Security} + 0.3 \cdot \left(1 - |A_{D|D} - A_{\widetilde{D}|\widetilde{D}}|\right) + 0.2 \cdot min\,\left(1, \frac{n+m+H_{before}}{n+m+H_{after}}\right). \quad \mathbf{(10^{\dagger})}$$

One may see from Equation (9) that half of the overall importance (0.5) is given to the *security* component of quality, which is responsible for assessing the safety of private data. The other half of the overall importance (0.5), which is related to the assessment of the value of the anonymized data for ML, is distributed between the *ML tolerance* (0.3), i.e., accuracy of models built on anonymized data, and the *efficiency* (0.2), i.e., the complexity of models built on anonymized data. Choice of a particular option of formula (10) for overall quality estimation depends on the available computational resources and the application specifics (i.e., how much in-depth analysis is needed).

Equation (10), Equation (10*), Equation (10**) and Equation ($10^{\dagger}$) are based on a weighted arithmetic mean of the quality components. However, other reasonable options would be based on a weighted geometric mean (i.e., no tolerance to a zero value of any component and have some small bias towards the weakest of them, which are the reasonable properties). Therefore, those who prefer to consider such a bias will use formula (11) or its modifications rather than modifications of formula (10). Here are the options:

$$\tilde{Q}(D \to \widetilde{D}) \approx \left(Q_{Security}\right)^{0.5} \cdot (Q_{ML\,Tolerance})^{0.3} \cdot \left(Q_{Efficiency}\right)^{0.2}. \quad \mathbf{(11)}$$

$$\tilde{Q}^{*}(D \to \widetilde{D}) \approx \left(Q^{*}_{\text{Security}}\right)^{0.5} \cdot (Q_{\text{ML Tolerance}})^{0.3} \cdot \left(Q_{\text{Efficiency}}\right)^{0.2}. \quad \mathbf{(11^{*})}$$

$$\tilde{Q}^{**}(D \to \widetilde{D}) \approx \left(Q^{**}_{\text{Security}}\right)^{0.5} \cdot (Q_{\text{ML Tolerance}})^{0.3} \cdot \left(Q_{\text{Efficiency}}\right)^{0.2}. \quad \mathbf{(11^{**})}$$

$$\bar{\tilde{Q}}(D \to \widetilde{D}) \approx \left(\bar{Q}_{\text{Security}}\right)^{0.5} \cdot (Q_{\text{ML Tolerance}})^{0.3} \cdot \left(Q_{\text{Efficiency}}\right)^{0.2}. \quad \mathbf{(11^{\dagger})}$$

4. Experiments on Applying the Metric

We tested the quality metric while applying the homeomorphic encryption on top of several popular datasets from the UCI ML Repository [21] according to the generic schema of experiments presented in Fig. 1. First, for each tested dataset (with basic parameters: $k; n; m$), we built three NN classifiers (with H_{before} neurons) based on the original data to get accuracy values $A_{D|D}$ (i.e., one classifier for each of three different divisions among the training and testing sets of the data: 70% – 30%; 50% – 50% and 30% – 70%). After that, original data was encrypted by one, two and three layers of homeomorphic encryption. For each of these three ($d = 1; 2; 3$) encrypted datasets, we build corresponding classifiers (with H_{after} neurons) and make all necessary measurements ($A_{\widetilde{D}|\widetilde{D}}$; $A_{D|\widetilde{D}}$; $A_{\widetilde{D}|D}$). Finally, we applied formulas (1-11) to compute all the quality components and overall quality of the encryption. This means that, in fact, we compute overall quality values for each of nine experiments (i.e. for three options of the anonymization depth multiplied by the three options of training-vs-testing set division). Therefore, we added a couple of average evaluations across all the nine experiments as follows:

$$\bar{\mathbb{Q}}(D \to \widetilde{D}) = \frac{1}{9} \cdot \sum_{i=1}^{9} \bar{Q}_i(D \to \widetilde{D}), \text{ where each } \bar{Q}_i(D \to \widetilde{D}) \text{ is defined by Equation } (10^{\dagger}); \quad \mathbf{(12)}$$

$$\bar{\tilde{\mathbb{Q}}}(D \to \widetilde{D}) = \frac{1}{9} \cdot \sum_{i=1}^{9} \bar{\tilde{Q}}_i(D \to \widetilde{D}), \text{ where each } \bar{\tilde{Q}}_i(D \to \widetilde{D}) \text{ is defined by Equation } (11^{\dagger}). \quad \mathbf{(13)}$$

For example, based on our experiments with the popular "Iris" dataset (https://archive.ics.uci.edu/ml/datasets/iris) ($k = 150; n = 4; m = 3$), we have got the following overall quality evaluations for the homeomorphic encryption: $\bar{\mathbb{Q}}(D \to \widetilde{D}) = 0.718$; $\bar{\tilde{\mathbb{Q}}}(D \to \widetilde{D}) = 0.69$. In "Breast Cancer" dataset (https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data) experiments ($k = 569; n = 30; m = 2$), we have got the following overall quality evaluations for the homeomorphic encryption: $\bar{\mathbb{Q}}(D \to \widetilde{D}) = 0.794$; $\bar{\tilde{\mathbb{Q}}}(D \to \widetilde{D}) = 0.776$. As one may see, both "Iris" and "Breast Cancer" contain a small number of data samples and small number of class labels. For a detailed analysis with more convincing estimates, let us consider the popular "Letter Recognition" dataset (https://doi.org/10.24432/C5ZP40; https://archive.ics.uci.edu/ml/datasets/letter+recognition) [21]; which has more samples ($k = 20000$), a reasonable number of numeric attributes ($n = 16$) and more class labels ($m = 26$). All the results of the quality evaluation experiments on the homeomorphic encryption of this dataset are listed in Table 1. Let us provide systematic comments regarding the experiment and the values from Table 1. Within each experiment, both the original and the encrypted (by three different depths $d = 1$, $d = 2$, and $d = 3$) datasets are being split into training / testing subsets (train/test ratio) by three different ways: 70% / 30%, 50% / 50%, and 30% / 70%. For each encryption case, the keys (i.e., the encrypting NN weights) were generated 20 times to get better-averaged estimations. Then, for each of these cases, the 10-fold cross-validation has been applied. All these have been performed to get trustful classification accuracy values for the "before" and "after" anonymization (encryption) comparisons. During the experiments, the parameters of the best performing (manually controlled) NN configuration are collected within Table 1. For example, one may see from Table 1 that, in the case $\text{train}/\text{test} = 70/30$, the classifier trained and tested on the original data has achieved the highest accuracy $A_{D|D} = 0.97$ on the configuration containing $H_{\text{before}} = 185$ neurons. Similar experiments have been made with the encrypted dataset. One can see that the classifier, trained and tested on the encrypted dataset with the ratio $\text{train}/\text{test} = 70/30$, achieves the best classification accuracy: (a) $A_{\tilde{D}|\tilde{D}} = 0.945$ on the configuration containing $H_{\text{after}} = 214$ neurons for the case of encryption depth $d = 1$; (b) $A_{\tilde{D}|\tilde{D}} = 0.92$ on the configuration containing $H_{\text{after}} = 245$ neurons for the case $d = 2$; and (c) $A_{\tilde{D}|\tilde{D}} = 0.895$ on the configuration containing $H_{\text{after}} = 275$ neurons for the case $d = 3$. Therefore, we can see that the natural payment for stronger encryption would be some loss of accuracy and growth of neuron resources in the best performing classifier configuration. In Fig. 3, one may see the plots for these experiments showing how the classification accuracy and neuron count are changing with the encryption depth. One may see that the optimal configuration has been chosen before the training process start to overfit.

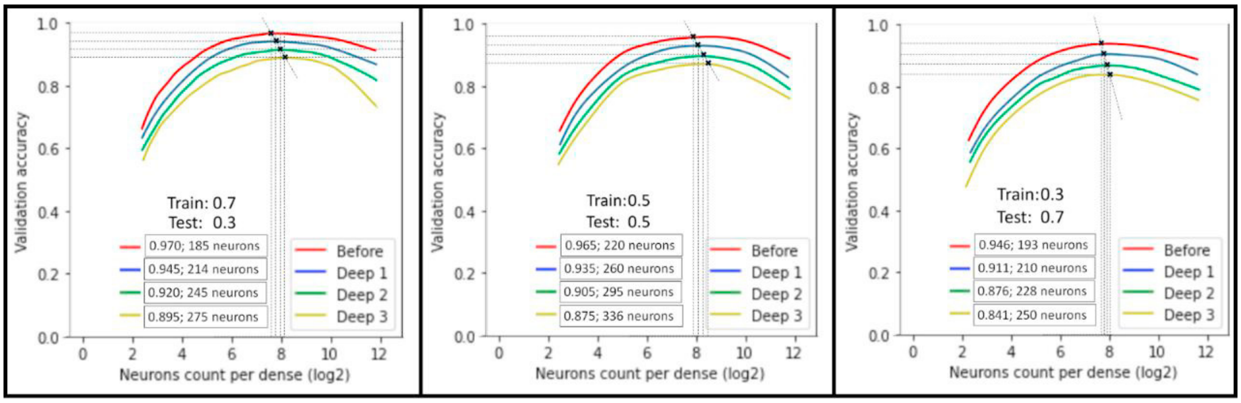

Fig. 3. Summary of experiments is demonstrated by finding the best performing NN configuration before (tagged as "Before") and after encryption separately for each depth of encryption ("Deep 1" for $d = 1$; "Deep 2" for $d = 2$; and "Deep 3" for $d = 3$). The experiments and plots are grouped for different ratios of training and test data used. An observation shows that encryption decreases slightly the classification accuracy over the encrypted data and demands more neurons for the best performing classifier configuration.

Important measurements (of how secure our original dataset would be after the encryption) have been performed with the cross-validation procedure. Here we train the classifier on the original data and test it on the encrypted data and vice-versa. One can see that the classifier, trained on the original dataset and tested on the encrypted one with the ratio $\text{train}/\text{test} = 70/30$, achieves the best classification accuracy: (a) $A_{D|\tilde{D}} = 0.05$ for the case of encryption depth $d = 1$; (b) $A_{D|\tilde{D}} = 0.04$ for the case $d = 2$; and (c) $A_{D|\tilde{D}} = 0.03$ for the case $d = 3$. We remind you that the small accuracy values here indicate good privacy protection of the original data. Also, one can see that the classifier, trained on the encrypted dataset and tested on the original one with the ratio $\text{train}/\text{test} = 70/30$, achieves the best classification accuracy: (a) $A_{\tilde{D}|D} = 0.06$ for the case of encryption depth $d = 1$; (b) $A_{\tilde{D}|D} = 0.04$ for the case $d = 2$; and (c) $A_{\tilde{D}|D} = 0.03$ for the case $d = 3$. Therefore, we have small accuracy values also here, which is good for data protection. After getting all these experimental measurements, we can compute all the necessary components for the encryption quality metric: Q_{Security} (all three options and the average of them); $Q_{\text{ML Tolerance}}$; and $Q_{\text{Efficiency}}$. All the computed values are presented in Table 1.

Table 1. Anonymization (Homeomorphic Encryption) quality evaluation sheet for the “Letter Recognition” dataset.

“Letter Recognition” Dataset (https://archive.ics.uci.edu/ml/datasets/letter+recognition)									
k	20000								
n	16								
m	26								
$train/test$	70/30			50/50			30/70		
d	1	2	3	1	2	3	1	2	3
H_{before}	185			220			193		
H_{after}	214	245	275	260	295	336	210	228	250
$A_{D\|D}$	0.970			0.965			0.946		
$A_{\tilde{D}\|\tilde{D}}$	0.945	0.920	0.895	0.935	0.905	0.875	0.911	0.876	0.841
$A_{D\|\tilde{D}}$	0.050	0.040	0.030	0.040	0.040	0.020	0.050	0.040	0.030
$A_{\tilde{D}\|D}$	0.060	0.040	0.030	0.060	0.050	0.020	0.050	0.030	0.030
a	0.920	0.930	0.940	0.925	0.925	0.945	0.896	0.906	0.916
b	0.885	0.880	0.865	0.875	0.855	0.855	0.861	0.846	0.811
c	0.975	0.950	0.925	0.970	0.940	0.910	0.965	0.930	0.895
δ	0.887	0.791	0.716	0.868	0.777	0.693	0.933	0.870	0.805
$Q_{Security}$	0.594	0.836	0.933	0.593	0.831	0.932	0.584	0.826	0.924
$Q^{*}_{Security}$	0.397	0.637	0.780	0.396	0.630	0.779	0.388	0.625	0.764
$Q^{**}_{Security}$	0.110	0.847	0.934	0.110	0.840	0.932	0.109	0.833	0.912
$\bar{Q}_{Security}$	0.367	0.773	0.882	0.366	0.767	0.881	0.360	0.761	0.867
$Q_{ML\ Tolerance}$	0.975	0.950	0.925	0.970	0.940	0.910	0.965	0.930	0.895
$Q_{Efficiency}$	0.887	0.791	0.716	0.868	0.777	0.693	0.933	0.870	0.805
$Q(D \to \tilde{D})$	0.767	0.861	0.887	0.761	0.853	0.878	0.768	0.866	0.892
$Q^{*}(D \to \tilde{D})$	0.668	0.762	0.811	0.663	0.752	0.801	0.670	0.766	0.812
$Q^{**}(D \to \tilde{D})$	0.525	0.867	0.888	0.520	0.857	0.878	0.531	0.870	0.886
$\bar{Q}(D \to \tilde{D})$	0.653	0.830	0.862	0.648	0.821	0.852	0.656	0.834	0.863
$\tilde{Q}(D \to \tilde{D})$	0.747	0.859	0.883	0.742	0.851	0.872	0.746	0.865	0.890
$\tilde{Q}^{*}(D \to \tilde{D})$	0.610	0.750	0.807	0.606	0.741	0.797	0.608	0.752	0.810
$\tilde{Q}^{**}(D \to \tilde{D})$	0.321	0.865	0.883	0.319	0.855	0.872	0.322	0.869	0.885
$\bar{\tilde{Q}}(D \to \tilde{D})$	0.587	0.826	0.858	0.583	0.817	0.848	0.585	0.830	0.862
$\bar{\bar{\mathbb{Q}}}(D \to \tilde{D})$	<u>0.780</u> { 0.652 (d = 1); 0.828 (d = 2); 0.859 (d = 3) }								
$\bar{\bar{\tilde{\mathbb{Q}}}}(D \to \tilde{D})$	<u>0.755</u> { 0.585 (d = 1); 0.824 (d = 2); 0.856 (d = 3) }								

The final quality evaluation is performed for each depth of encryption separately including overall estimation. The final quality estimations for the homeomorphic encryption are: (a) $\bar{\bar{\tilde{\mathbb{Q}}}}(D \to \tilde{D}) = 0.585$ for the case $d = 1$; (b) $\bar{\bar{\tilde{\mathbb{Q}}}}(D \to \tilde{D}) = 0.824$ for the case $d = 2$; (c) $\bar{\bar{\tilde{\mathbb{Q}}}}(D \to \tilde{D}) = 0.856$ for the case $d = 3$; and (d) $\bar{\bar{\tilde{\mathbb{Q}}}}(D \to \tilde{D}) = 0.755$ (75.5%) as a summary across all the experiments. One may see that further (deeper than $d = 3$) encryption will decrease the overall quality of encryption (i.e., security gain will not compensate for ML tolerance and efficiency losses). Therefore, for the “Letter Recognition” dataset, we can consider homeomorphic encryption with the depth $d = 3$ to be the best option to achieve the optimal quality (0.856) as a trade-off over the conflicting quality criteria.

5. Conclusions

The contradiction between the growing demands for personal and process data privacy in Industry 4.0 and the need to get maximal value out of the data through AI and ML inspires new solutions and algorithms towards PPML. Smart manufacturing and logistics environments are increasingly reliant on sensitive data for optimizing processes and decision-making. For instance, in manufacturing, ensuring the privacy of proprietary production techniques and quality control data is paramount. Additionally, supply chain and logistics operations involve sharing data among multiple parties, necessitating robust privacy protection to prevent data breaches. By safeguarding sensitive data while enabling accurate model training, our approach could easily find practical applications in scenarios such as proprietary process optimization, supply chain analytics, predictive maintenance, and quality assurance in the Industry 4.0 framework.

Homeomorphic encryption suggested in [12] and [13] pretend to be simple yet efficient tool for the PPML tasks compared to, e.g., more complex homomorphic encryption. In this paper, we studied the quality of the homeomorphic encryption technique against three mutually conflicting criteria (security, ML tolerance and efficiency). For that purpose, we have developed the quality evaluation metric, which can be used as a quality judge for homeomorphic encryption as well as for other similar encryption and anonymization techniques (algorithms). We have tested both the quality metric and the homeomorphic encryption (with different depths) with several different datasets. We noticed that such a metric is a good guide for choosing the optimal encryption parameters. Due to several experiments, we have also noticed that the homeomorphic encryption guarantees a good overall quality regarding the conflicting criteria and can be used as a relatively simple alternative to the traditional homomorphic (and related) encryption.

While recognizing the limitations of this study, which was confined to specific encryption parameters and public datasets, future research could delve into the integration of homeomorphic encryption with advanced machine learning techniques. Such a future exploration would target a harmonious balance between augmented privacy protection and elevated model performance within the dynamic context of smart manufacturing, encompassing its distinct datasets.

References

[1] Harari, Y. (2019). *21 Lessons for the 21st Century*. Vintage.

[2] Alazab, M., Gadekallu, T. R., and Su, C. (2022). ”Guest editorial: Security and privacy issues in Industry 4.0 applications”. *IEEE Transactions on Industrial Informatics*, **18(9)**: 6326-6329. https://doi.org/10.1109/TII.2022.3164741

[3] Elbasheer, M., Longo, F., Nicoletti, L., Padovano, A., Solina, V., and Vetrano, M. (2022). “Applications of ML/AI for decision-intensive tasks in production planning and control”. *Procedia Computer Science*, **200**: 1903-1912. https://doi.org/10.1016/j.procs.2022.01.391

[4] Tikkinen-Piri, C., Rohunen, A., and Markkula, J. (2018). “EU General Data Protection Regulation: Changes and implications for personal data collecting companies”. *Computer Law & Security Review*, **34(1)**: 134-153. https://doi.org/10.1016/j.clsr.2017.05.015

[5] Al-Rubaie, M., and Chang, J. M. (2019). “Privacy-Preserving Machine Learning: Threats and solutions”. *IEEE Security & Privacy*, **17(2)**: 49-58. https://doi.org/10.1109/MSEC.2018.2888775

[6] Graves, L., Nagisetty, V., and Ganesh, V. (2021). “Amnesiac machine learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 35, No. 13, pp. 11516-11524). https://doi.org/10.1609/aaai.v35i13.17371

[7] Kesarwani, M., Mukhoty, B., Arya, V., and Mehta, S. (2018). ”Model extraction warning in MLaaS paradigm”. In: *Proceedings of the 34th Annual Computer Security Applications Conference* (pp. 371-380). https://doi.org/10.1145/3274694.3274740

[8] Thuraisingham, B. (2022). “Trustworthy machine learning”. *IEEE Intelligent Systems*, **37(1)**: 21-24. https://doi.org/10.1109/MIS.2022.3152946

[9] Pfleeger, S., and Cunningham, R. (2010). “Why measuring security is hard”. *IEEE Security & Privacy*, **8(4)**: 46-54. https://doi.org/10.1109/MSP.2010.60

[10] Hossin, M., Sulaiman, M. N., Mustapha, A., Mustapha, N., and Rahmat, R. W. (2011). ”A hybrid evaluation metric for optimizing classifier”. In: *Proceedings of the 3rd Conference on Data Mining and Optimization* (pp. 165-170). IEEE. https://doi.org/10.1109/DMO.2011.5976522

[11] Zeng, X., Yang, C., and Dai, B. (2022). “Utility–privacy trade-off in distributed machine learning systems”. *Entropy*, **24(9)**: 1299. https://doi.org/10.3390/e24091299

[12] Girka, A., Terziyan, V., Gavriushenko, M., and Gontarenko, A. (2021). ”Anonymization as homeomorphic data space transformation for privacy-preserving deep learning”. *Procedia Computer Science*, **180**: 867-876. Elsevier. https://doi.org/10.1016/j.procs.2021.01.337

[13] Terziyan, V., Malyk, D., Golovianko, M., and Branytskyi, V. (2023). “Encryption and generation of images for privacy-preserving machine learning in smart manufacturing”. *Procedia Computer Science*, **217**: 91-101. Elsevier. https://doi.org/10.1016/j.procs.2022.12.205

[14] Yale, A., Dash, S., Dutta, R., Guyon, I., Pavao, A., and Bennett, K. P. (2020). “Generation and evaluation of privacy preserving synthetic health data”. *Neurocomputing*, **416**: 244-255. https://doi.org/10.1016/j.neucom.2019.12.136

[15] Rechberger, C., and Walch, R. (2022). “Privacy-preserving machine learning using cryptography”. In: Batina, L., Bäck, T., Buhan, I., Picek, S. (Eds.), *Security and Artificial Intelligence*. *Lecture Notes in Computer Science*, **13049**: 109-129. Springer, Cham. https://doi.org/10.1007/978-3-030-98795-4_6

[16] Tran, A. T., Luong, T. D., Karnjana, J., and Huynh, V. N. (2021). ”An efficient approach for privacy preserving decentralized deep learning models based on secure multi-party computation”. *Neurocomputing*, **422**: 245-262. https://doi.org/10.1016/j.neucom.2020.10.014

[17] Iezzi, M. (2020). “Practical privacy-preserving data science with homomorphic encryption: an overview”. In: *Proceedings of the IEEE International Conference on Big Data* (pp. 3979-3988). IEEE. https://doi.org/10.1109/BigData50022.2020.9377989

[18] Ogburn, M., Turner, C., and Dahal, P. (2013). “Homomorphic encryption”. *Procedia Computer Science*, **20**: 502-509. https://doi.org/10.1016/j.procs.2013.09.310

[19] Olah, C. (2014). “Neural networks, manifolds, and topology”. In: *Colah’s Blog*. https://colah.github.io/posts/2014-03-NN-ManifoldsTopology. Accessed 11.08.2023.

[20] Lee, J. W., Kang, H., Lee, Y., Choi, W., Eom, J., Deryabin, M., Lee, E., Lee, J., Yoo, D., Kim, Y., and No, J. S. (2022). “Privacy-preserving machine learning with fully homomorphic encryption for deep neural network”. *IEEE Access*, **10**: 30039-30054. https://doi.org/10.1109/ACCESS.2022.3159694

[21] Dua, D., and Graff, C. (2017). *UCI Machine Learning Repository*. Retrieved from http://archive.ics.uci.edu/ml. Accessed 11.08.2023. http://doi.org/10.17616/R3T91Q