Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 180 (2021) 867–876

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry 4.0 and Smart Manufacturing

Anonymization as homeomorphic data space transformation for privacy-preserving deep learning

Anastasiia Girka [a], Vagan Terziyan [a], Mariia Gavriushenko [a], Andrii Gontarenko [b]

[a]Faculty of Information Technology, University of Jyväskylä, Jyväskylä, 40100, Finland
[b]Qvantel Finland Oy, Piippukatu 11, Jyväskylä 40100, Finland

Abstract

Industry 4.0 is largely data-driven nowadays. Owners of the data, on the one hand, want to get added value from the data by using remote artificial intelligence tools as services, on the other hand, they concern on privacy of their data within external premises. Ideal solution for this challenge would be such anonymization of the data, which makes the data safe in remote servers and, at the same time, leaves the opportunity for the machine learning algorithms to capture useful patterns from the data. In this paper, we take the problem of supervised machine learning with deep feedforward neural nets and provide an anonymization algorithm (based on the homeomorphic data space transformation), which guarantees privacy of the data and allows neural networks to learn successfully. We made several experiments to show how much the performance of the trained neural nets will suffer from the deepening of the anonymization power.

Keywords: Industry 4.0; Privacy; Neural Network; Deep Learning; Topology

1. Introduction

Major players in the market of cloud-based services offer various smart analytics including artificial intelligence (AI) and cognitive computing algorithms, tools and platforms for their individual and corporate customers. Such analytics is often based on machine learning (ML) as a tool to train predictive models on the basis of data. Existing ML-as-a-service (MLaaS) platforms require customers to disclose the data (needed for ML algorithms to learn) within the service provider cloud storage premises [1]. Modern ML algorithms (e.g., deep neural networks) when dealing with the potentially privacy-sensitive data can create potential security and privacy risks [2]. European General Data Protection Regulation (GDPR) set up strict rules regarding the storage and exchange of personally identifiable data. Good review on current and next-generation methods for federated, secure and privacy-preserving AI focused on

10.1016/j.procs.2021.01.337

medical imaging is presented in [3]. They noticed an unfortunate practice of the large-scale re-identification attacks and that the sale of re-identified medical records is becoming a business model for some companies.

Nowadays, there are different approaches to address privacy and security issues using blockchain technology. In their survey, Panarello et al. addressed these issues in relation with the Internet of Things (IoT) [4]. They pointed out specific security and privacy vulnerabilities in the IoT, which can be addressed with the blockchain technology. Longo et al. in [5] suggested special software connector, which can integrate the blockchain into the enterprise information system, and proved that the blockchain fosters collaboration (allows companies to safely share information with their partners with different levels of visibility) and reduces security and privacy (and generally trust) issues in a supply chain. Current blockchain-based deep learning solutions (to be able to preserve accuracy) require essential increase in end-to-end latency, mining time, computation and communication costs [6]. In their review, Lisin & Zapechnikov discuss two groups of privacy-preserving ML approaches: cryptographic (homomorphic encryption, garbled circuits or cryptographic protocols, order-preserving encryption and secure processors) and perturbation (adding special noise either to the input data, to the output data or to the model training) [7]. The choice of the appropriate method depends on the context of a particular problem and the level of requirements to privacy and to the available resources. There are also attempts to learn data distribution from the available dataset with sensitive data in a way that learned distribution preserves privacy and enables generating artificial data samples from the learned distribution for further ML without worrying on privacy. For example, in [8] authors suggest using the capabilities of the Generative Adversarial Network basic architecture and its variants to serve as a data generation model.

There is still a need for new privacy-preserving solutions on how to apply deep ML algorithms to encrypted data, train classifiers (predictors), make encrypted classifications (predictions), and, finally, return the results of classification (prediction) in encrypted form back to the original data owners. Special demand is how to preserve the quality of classification (prediction) made on top of encrypted data. In this paper, we focus on the problem of supervised machine learning with deep feedforward neural nets and provide an anonymization algorithm, which guarantees privacy of the data and allows neural networks to learn efficient classifiers.

The rest of the paper is organized as follows: in Section 2, we extend the summary of related work; in Section 3, we introduce the problem to be solved; in Section 4, we describe our approach and the algorithm; in Section 5, we report on experimental evaluation of the features of the proposed algorithm; and we conclude in Section 6.

2. Related work

The digital transformation trend with extensive exploitation of data enables new industrial models like the Industry 4.0 and underpins the emergence of a new data-driven economy. With the integration of modern IT into industrial world many new business opportunities can be envisioned. The technical challenge for these opportunities would be enabling correct and efficient analysis of the large amounts of data produced by factories and their large sensor networks. ML as a technology has progressed a lot during the last few years, however, the legal aspects of the collection and processing data (such as privacy concern regarding personal data) has been neglected. Nowadays the potential of the ML models is essentially limited by the strict privacy regulations. The actively discussed radical scenario, i.e., “the right to be forgotten” [9], is related to complete removing (aka “forgetting”) personal (sensitive) data and information from automated processing if requested by an individual. Potential negative impact of such scenario for ML is studied by [10], i.e., the deletion of existing data, would be the loss of model performance.

There are several popular instruments for privacy protection, such as k-anonymity [11], l-diversity [12], t-closeness [13] and differential privacy [14]. In spite of that fact that these techniques are known to improve the anonymity protection against attackers with different background knowledge, however, as argued in [15], they all have some common defects, e.g., questionable applicability in deep learning scenarios. However, industry still recognizes differential privacy as a practical standard for privacy protection. Four major applications of differential privacy in cyber-physical systems (together with open issues and remaining problems) has been reported in [16], which are energy systems, transportation systems, healthcare and medical systems, and industrial internet of things. Enhanced definition of differential privacy in conjunction with non-randomizing anonymizing strategies applied to the industrial medical internet of things has been reported in [17], which allows the formulation of privacy conditions over the evolving set of features. Hou et al. in [18] propose a differential privacy protection method (aiming a balance between privacy protection and utility). Differential privacy enhanced with semantic privacy has been reported in [19] in the

context of location privacy preservation for road networks. Zheng & Cai in [20] adopt the differential privacy concept for the problem of a privacy-preserved data sharing in industrial internet of things. Chamika et al. in [21] introduced a framework to address privacy and trustworthiness of the Industry 4.0 data by merging differential privacy, federated ML, blockchain, and smart contracts and they tested the feasibility of the framework using simulations. As noticed in [22], industrial data is often neither centralized nor shared, and the lack of massive public datasets leads to overfitting and low performance of the learned model. The emerging federated learning over distributed datasets is an emergent need. However, as noticed in [23], adversaries can still exploit the shared parameters and compromise such industrial applications as auto-driving navigation, wearable medical electronics, industrial robots, etc.

Deep learning is becoming popular due to its remarkable accuracy when trained with a massive amount of data, such as generated by IoT. However, as argued in [24], deep learning algorithms tend to leak privacy when trained on highly sensitive crowd-sourced data and they suggest adding a randomization layer before data leave the data owners' devices and reach a potentially untrusted machine learning service. For similar purposes, Alguliyev, Aliguliyev & Abdullayeva in [25] suggest architecture with modified sparse denoising autoencoder to perform transformation of data before giving it to a convolutional network for classification.

Summarizing, we admit that the vulnerabilities and new attacks on privacy evolve faster than corresponding protection does; therefore, new approaches for privacy protection are still needed. In this paper, we suggest addressing the privacy vulnerabilities of deep neural networks by using other deep learning networks (enabling anonymization as an invertible functional encryption) as a tool for safe, secret, fast and ML-tolerant data transformation.

3. Problem description

Assume that some company has collected data from its processes. The data also contains sensitive information about the personnel, customers (patients, etc.). This company wants to get some added value from the data but cannot afford expensive AI/ML resources within its own premises. Popular option would be to use external clouds (provided by, e.g., Microsoft, IBM, Google, Amazon, etc.), which provide solid MLaaS. Ordinary practice is that the data has to be placed within this remote cloud before the actual processing starts. ML tools of the cloud process the data, and, as a result, there will be trained model(s). Derived models may be stored and remain available in the cloud for the remote queries managed by the data owner. This traditional practice is not safe regarding the personal data privacy and derived knowledge ownership and does not fit current requirements, concerns and laws regarding personal data.

The data owner may apply some “naive” or “weak” anonymization technique while preparing sensitive data for the remote storage and processing. The “weakly” anonymized data is placed into the remote cloud. This would be much safer, but is it really safe? In some cases attackers may use publicly available data from elsewhere and also AI tools to uncover (re-identify) the sensitive information from “weakly anonymized” data or, even if the anonymized data itself is safe within the remote cloud, can one be sure that the knowledge (models trained by ML tools) will make any sense for potential use? Assume that we finally got the anonymized data and anonymized models derived from it. Now we need our (data owner) applications to be capable of sending anonymized queries to the models and get the anonymized responses, but such ones that (when “unlocked” in the safe place) will make sense for the applications as expected. Weak or "naive" anonymization is insufficient for both: safety of the private data and for the applicability of the derived knowledge models from such data. *Key objective*: Therefore, we intend a set of “Smart Anonymization” techniques (Smart Privacy Guardian) capable of not only protecting the privacy of the data but also enabling remote AI/ML tools (MLaaS) from untrusted clouds to capture applicable and protected models out of it.

4. Smart Privacy Guardian

4.1. Homeomorphic Data Space Transformation

Topology is a mathematical field that studies homeomorphism (topological isomorphism) as a continuous function between topological spaces that has a continuous inverse function. In his popular blog [26], Olah described the connection between topology and deep neural network learning. He argues that each fully-connected layer of a feedforward deep neural network stretches and squishes the space (defined by input vector of data), but it never cuts, breaks, or folds it, which means that it preserves topological properties of the data space. He proved a theorem that

states: 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).

We would like to use the topology-preserving transformation of the original data as a kind of anonymization technique, which uses multilayered neural network to hide (by replacement, which uses appropriate secret-key-matrices multiplication and activation function at each layer) the sensitive or private data samples but still enables potential classification model to be learned on the basis of the anonymized data. For the original *k*-dimensional data, the input vector (at each layer) will be multiplied with the randomly selected non-singular "secret key" *k* x *k* matrix and then activation function will be used. What is the role of activation function in anonymization? Assume that some of the original data samples with their attributes have been leaked (e.g., openly published by their owners). This would mean that potential hacker, partially knowing the original and the anonymized data, may try to uncover the anonymization key matrices. One may see that a simple matrix key (one layer without activation function) applied on top of *k*-dimensional data for anonymization, will be vulnerable given leak of at least *k* samples of data with their attributes. It would be worth trying yet another key ("hiding order" of dimensions) and now lock it again (homeomorphism) as before. Now the task is much more problematic for a hacker, even if there is a leak of data. Now (for *k*-dimensional data) the complexity of previously introduced problem for a hacker will grow *k*! times (because in the worst case scenario, he must try all possible permutations of *k* dimensions while uncovering the rest of secret information). Finally, we can also add non-linearity (some invertible activation function, e.g., *sigmoid* or *hyperbolic tangent*) as additional protection. This allows using many similar keys (additional security layers) because: $\sigma(A\times B)\neq\sigma(A)\times\sigma(B)$. Let us consider all these anonymization stages in detail in the next subsection.

4.2. Anonymization Algorithm

We describe the algorithm and synchronously follow and illustrate a simple example. The task in the example is to anonymize the original simple two-dimensional dataset with two numeric input attributes X1 and X2 and one output attribute Y (class label with two possible values Green or Red); the dataset has eight labelled data samples as shown on the left-hand side in Figure 1.

Stage 1: Data normalization

We begin with data normalization. Here we normalize the numeric attributes (vector X) of data by transforming their values to fit into the interval [-0.5, 0.5] (Figure 1). Notice that we save the MIN and MAX values for each attribute of the original data, and these values will serve as the first set of anonymization keys (keys may be used later to make the same transformation for the new testing data samples). [-0.5, 0.5] interval is chosen in order to make data suitable for potential application of such activation functions like sigmoid or hyperbolic tangent.

Stage 2: Hiding ("normalizing") the class labels

Here the class labels are hidden with a secret key (i.e., also "normalized" in a way): Red = 1, Green = 2(Figure. 2).

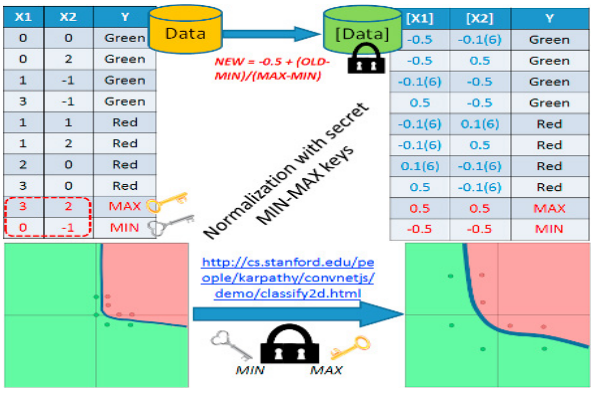


Fig. 1. Result of data normalization in the example.

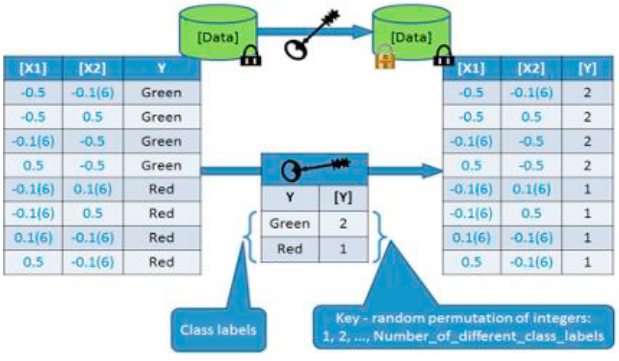


Fig. 2. Hiding (masking) the class labels.

Stage 3: Hiding the order of data samples

On this stage, we change (lock or hide) the original order of the data samples with yet another secret key as shown in Figure 3).

Stage 4: Hiding the order of dimensions (data attributes)

We change and hide the order of columns in the dataset according to the new secretly generated key (Figure 4).

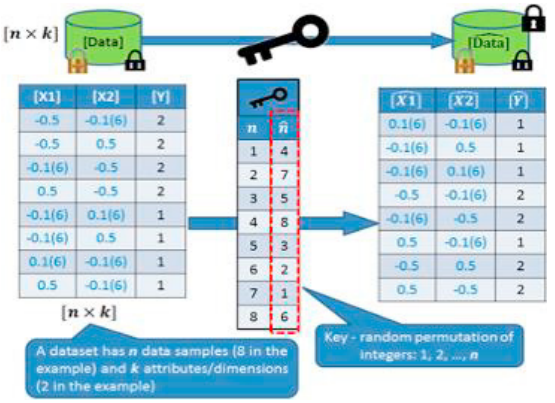


Fig. 3. The result of changing and locking the data samples order.

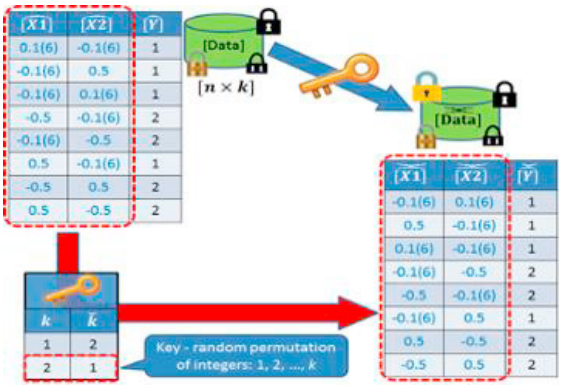


Fig. 4. Hiding and locking the order of attributes.

Stage 5: Homeomorphic data space transformation

This is a stage where attribute matrix is multiplied by a non-singular weight matrix, which is a new secretly generated key (Figure 5).

Stage 6: Adding secret bias

Randomly generated vector (as a secret key) is added to the attributes' values of each data sample (Figure 6).

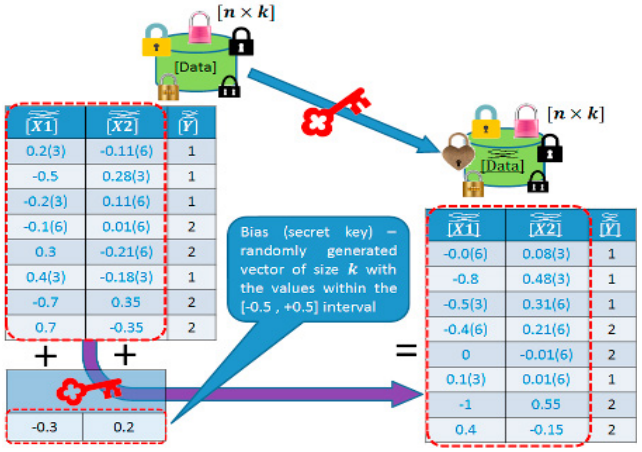


Fig. 5. Demonstration of the homeomorphic data space transformation.

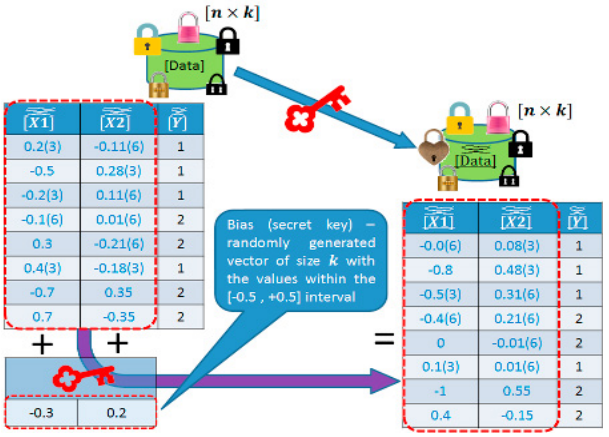


Fig. 6. Adding a bias vector (as a secret key) to the data.

Stage 7: Applying activation function

Now an invertible continuous activation function will be applied. In the example, the hyperbolic tangent is applied to the data attributes' values as shown in Figure 7.

We have completed one-layer anonymization (which could be enough in many cases). Comparison of the original dataset with the anonymized dataset is presented in Figure 8. In Figure 9, one may see how the query (i.e., test sample) is addressed in the original dataset and (after being anonymized) in the anonymized dataset, i.e., the class label (Green) is discovered correctly and kept in the encrypted form (as "2") until unlocked in a safe place.

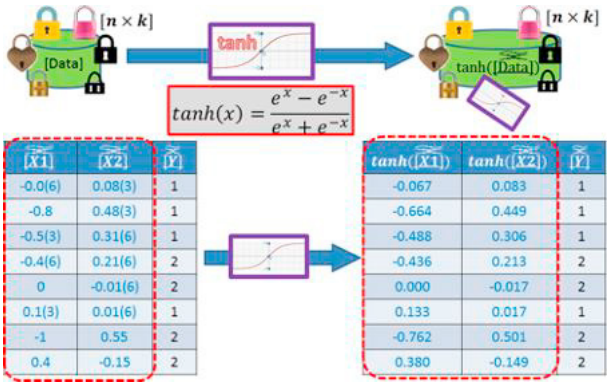


Fig. 7. Applying activation function (hyperbolic tangent in this case).

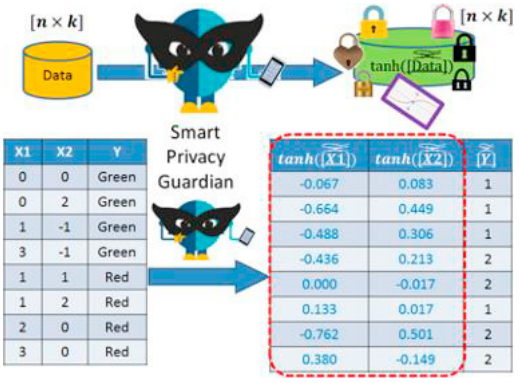


Fig. 8. Dataset before anonymization and after it.

Notice: stages 5, 6 and 7 (if necessary) can be recursively performed several times producing more keys and adding more protection. One may notice (Figure 10) that this kind of multi-layered (or deep) anonymization is equivalent to applying a deep neural network with, e.g., hyperbolic tangential layers and "frozen" weights and biases (aka keys), which make the whole process easy to implement using existing tools.

All the layers (starting from the vector of original k-dimensional data and ending with the vector of anonymized data) have the same size of k neurons.

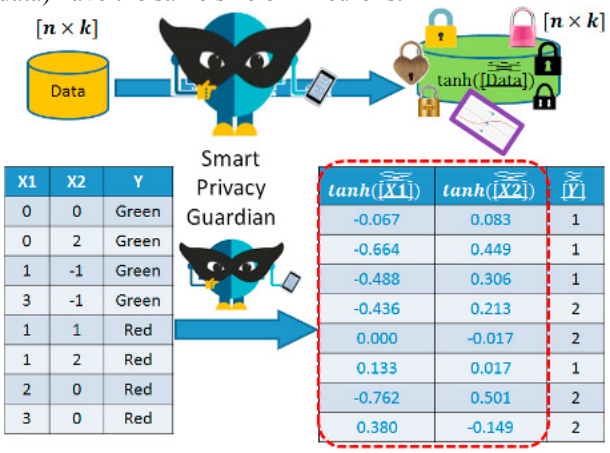


Fig. 9. From anonymized query to anonymized class label.

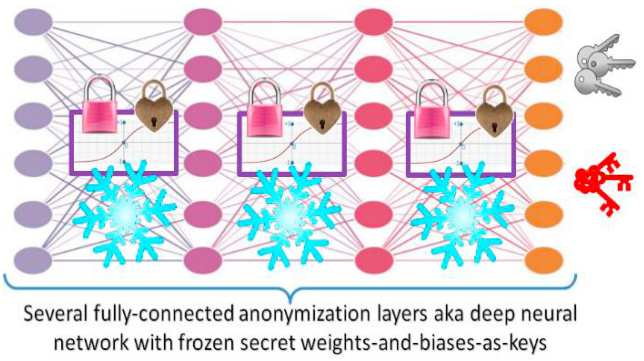


Fig.10 Smart Privacy Guardian solution as a deep neural network with frozen weights and biases (secret keys) and invertible activation.

Regarding the security of such anonymization solution, let us consider the unlikely worst-case scenario. Assume that due to a huge leak, a hacker knows almost everything about the original dataset and also, he got access to the anonymized data. Assume also (almost impossible) that he uncovered the correct order of samples in the dataset as well as the correct order of attributes. Now he wants to break the homeomorphic protection (i.e., uncover the transformation matrices and biases by solving the set of complex nonlinear equations). If he does this computationally huge task, then he can uncover the remaining part of the original data. Each additional layer of defense produces unknown variables (transformation matrix values and bias vector values) to the hacker's set of equations, however, if the number of data samples in the dataset, then the hacker will have enough equations to succeed. Therefore, for *complete protection* (i.e., even with all the data leaked (almost impossible scenario), the keys cannot be uncovered unambiguously), more layers are needed. Minimum number of layers for complete protection (assuming even 100% leakage) is as much as, for example, complete (theoretically unbreakable) protection of the 10-dimensional dataset with 10000 samples will require 91 protection layers.

5. Use-case description and experiments

In order to prove a concept, we demonstrate two different deep learning use-cases: classification problem and time-series prediction problem. All experiments were conducted with Python 3.7.4 and Keras 2.3.1 API for deep learning with TensorFlow 2.1.0 backend, on a GPU server with CentOS Linux 7 and with the following technical characteristics: Intel X86-64 architecture-based GPU-box with two GeForce GTX 1080Ti graphics cards, GCC v.7.3.0. Min-Max normalization was applied in both cases. The anonymization was performed by inserting additional non-trainable layers to the NN architecture, which hyperparameters were chosen with a grid search. Non-trainable layers were implemented with Keras Dense layer with default setting, which "trainable" attribute was set to False (let us call such a layer as "a frozen layer" onwards). Several activation functions were probed, as well as different number of frozen layers have been tested. Non-trainable Dense layer with certain activation function serves for multiplication of the matrix of input dataset by the matrix of weights. The weights of the Dense layer are initialized by default with 'glorot_uniform' kernel initializer, biases are set to zeros, an activation function provides a non-linearity to the transformation. Hyperbolic tangential, sigmoid, and exponential activation functions were probed for the frozen layers. ReLU activation function is not suitable for the frozen layers, because the transformation with ReLU is not invertible.

5.1. Classification Problem

Classification problem is demonstrated on UCI repository dataset (https://archive.ics.uci.edu/ml/ datasets/seeds), which contains 210 data samples each belongs to one of three different varieties of wheat, there are seven real-number attributes in the dataset. Model hyperparameters were defined as follows: activation functions for dense hidden layers (tanh, sigmoid, exp, ReLU); optimizers: (SGD, RMSprop, Adagrad, Adam); learning rates (0.1 and 0.01); number of neurons in hidden layers (from 8 to 13); number of additional hidden layers (from 1 to 3); batch size (1, 10, 21, and 70).

Loss function applied: categorical_crossentropy. During the grid search early stopping was applied with following parameters: monitor = 'val_loss', min_delta = 0.0001, patience = 80, mode = 'min'; and adapting learning rate technique ReduceLROnPlateau (monitor = 'val_loss', factor = 0.1, patience = 10, min_delta = 1E-7). Validation split was set as 0.2. Data were normalized with the min-max normalization. Accuracy was estimated with five-fold cross-validation (CV). As a result of the grid search the maximum accuracy value was found to be 94.76% +/- 2.33% for the following hyperparameters of the fully-connected layers and training (time to train five models for five CV folds: 17 sec): activation function for hidden layer: hyperbolic tangential; Optimiser: Adam; Learning rate: 0.1; Number of neurons in a hidden layer(s): 8; Number of hidden layers: 1; Batch size: 10.

5.2. Time-series prediction problem

Time-series prediction problem is demonstrated on machinery sales forecasting actual data. The dataset consisted of monthly sales records, 44 records altogether. Sales for three competitors served as predictors. Competitors were chosen based on correlation matrix and their sales (for two adjoint classes of machinery) served as predictors, as well as sales of target manufacturer adjoint classes (lower and higher) since there was a possibility of overlapping of the classes for some manufacturers. Competitors' sales values were summed up as follows: competitor_A sales for January for lower machinery class summed up with competitor_A sales for January for actual machinery class and summed up with competitor_A sales for January for higher machinery class. There were five predictors altogether: three for competitors and two for actual manufacturer (adjoint classes). Root mean squared error (RMSE) has been used as a model evaluation metric. Evaluation was performed with the CV technique, namely walk-forward CV aka time-series split, and one sample shift resulted with 18 splits. Training subset contained 21 sample, validation and testing subsets contained 12 samples each. The problem has been defined as a forecasting of sales for three months based on sales for the previous six months. The problem requires transformation of dataset so that a neural network receives six data samples at once, then a shift on one sample is done. Therefore, 21 samples turn into 12 and 12 samples turn into three. Due to the small amount of data available, the size of the neural network was kept minimal: two hidden LSTM layers, each with number of units equal to number of predictors (11). A grid search has been

performed in order to define an optimal activation function, optimizer and the learning rate. Probed values are: activation functions (sigmoid, tanh, ReLU); optimizers (SGD, RMSprop, Adagrad, Adam); learning rates (0.1 and 0.01).

5.3. Interpretation of experiments

The conducted experiments demonstrated the technical approach for anonymization: it can be performed by inserting additional layers to original NN. The results of the experiments are similar both for multilayer perceptron NN and for LSTM NN. Important results are collected and illustrated in Table 1.

The performance of the models (see Figure 11 and Figure 12 within Table 1) with additional frozen layers is slightly lower than the performance of the original model without frozen layers, however, mean values still lay in the range of standard deviation of the original model performance mean value.

Training of the models with additional frozen layers requires more epochs (about 10% more epochs) as can be seen in Figure 13 from Table 1, and it takes more time (from 1.5 to 2.5 times longer training) as illustrated in Figure 14 from Table 1.

It is notable, that there is no explicit dependence between the number of the frozen layers and performance, number of epochs, and time to train the models.

Exponential activation function appeared to be not suitable for the more than two frozen layers of LSTM NN: already with three frozen layers the RMSE value increases significantly and with more frozen layers predicted values go infinity as well as RMSE value. Increasing a learning rate from 0.01 to 0.1 does not solve the problem in general, however it remains valid only for some splits.

6. Conclusions and Future work

Privacy of sensitive data in industry in general and Industry 4.0 in particular remains in the focus of academic community. The importance of the challenge is increasing due to the Covid-19 crisis (companies start monitoring also the private health records of their workers to stop spreading the pandemics). ML applied on top of collected data creates important added value for the companies and in the same time raises the privacy concerns, especially if the sensitive data is processed within remote clouds.

Smart Privacy Guardian concept offers a reliable and accessible solution to save privacy of data processed by MLaaS platforms. In this paper, we take the problem of supervised machine learning with deep feedforward neural nets and provide an anonymization algorithm (based on the homeomorphic data space transformation), which guarantees privacy of the data and allows neural networks to learn successfully.

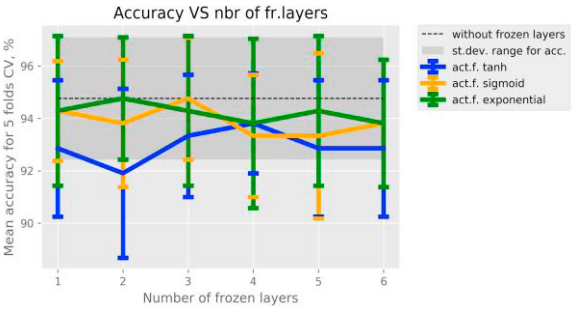
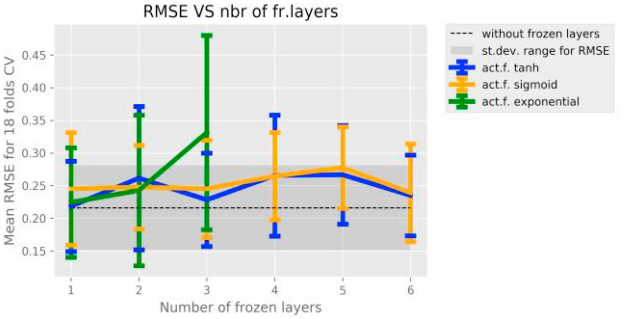
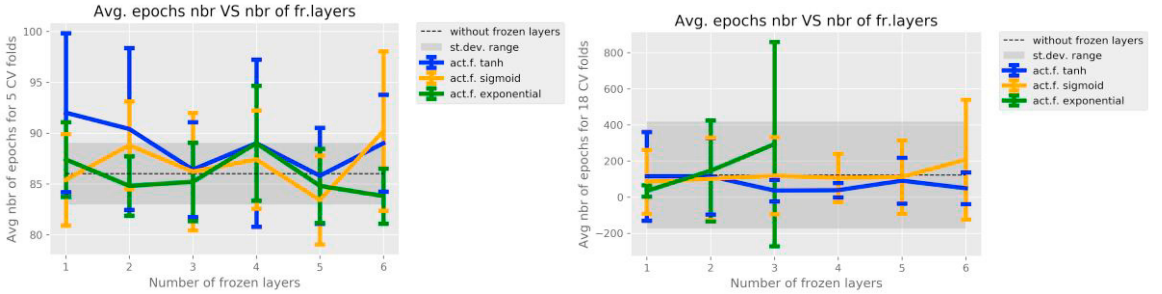
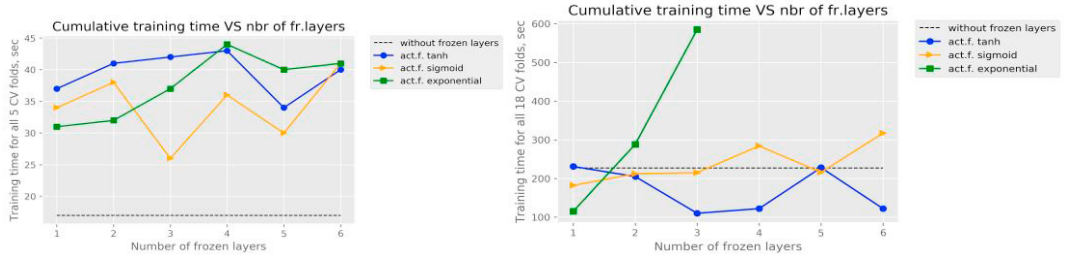
We made several experiments to show how much the performance of the trained neural nets will suffer from the deepening of the anonymization power. The limitations of the proposed approach are due to the basic settings of the Olah theorem [26], which requires the same amount of neurons at input layer and at the anonymization layers to keep the homeomorphic transformation while locking the private data. In addition, the anonymization layers must contain only invertible and non-linear activation functions (e.g., tanh or sigmoid) to guarantee homeomorphism. However, after the anonymization procedure, one can use wider networks, which will further learn (e.g., a classifier) based on the anonymized data. In addition, the approach promises quite high and controllable level of protection, which would be unfeasible to break.

Practical part of the paper demonstrated implementation of homeomorphic data space transformation for two types of NNs, namely Multilayer Perceptron and LSTM, with additional frozen layers. The conducted experiments revealed no decreasing performance with homeomorphic transformation introduced and no dependency between number of frozen layers and training time. This proves that presented approach of data anonymization with Smart Privacy Guardian can be integrated into the existing software and hardware solutions and, thus, can be widely adopted and bring benefit from ML everywhere it can make a value.

Future work includes experiments with frozen layers for all other activation functions provided by Keras library, as well as experiments with bias. In order to get more persuasive conclusion on an influence of frozen layers on training time it is worth to try them with a deep learning model that contains itself more layers and take relatively

longer time to train. In addition, other stages of anonymization algorithm (see Section 4.2) require an experimental investigation of their influence on model performance and training process.

Table 1. Results of the experiment.

Classification	Time series prediction
Performance	
 Figure 11: Mean accuracy against number of frozen layers for fthree activation functions compared with the mean accuracy for the model trained without frozen layers.	 Figure 12: Mean RMSE value for the models trained with different number of frozen layers and different activation functions compared with the mean RMSE value for a model trained without frozen layers.
Number of epochs	
 Figure 13: Average number of epochs to train models with different activation function of frozen layers against number of frozen layers and comparing with average number of epochs to train a model without frozen layers for five CV folds.	
Training time	
 Figure 14: Cumulative training time for models with different activation functions of the frozen layers for all five CV folds and different number of frozen layers against number of frozen layers, comparing to cumulative training time of the model without frozen layers	

References

[1] Hunt, Tyler, Song, Congzheng, Shokri, Reza, Smatikov, Vitaly, and Withchel, Emmett. (2018) "Chiron: Privacy-preserving machine learning as a service." arXiv preprint arXiv:1803.05961.

[2] Hesamifard, Ehsan, Takabi, Hassan, Ghasemi, Mehdi, and Rebecca N. Wright. (2018) “Privacy-preserving machine learning as a service.” Proceedings on Privacy Enhancing Technologies 2018 (3): 123-142.

[3] Georgios A. Kaissis, Marcus R. Makowski, Rückert, Daniel, and Rickmer F. Braren. (2020) "Secure, privacy-preserving and federated machine learning in medical imaging." Nature Machine Intelligence 2020: 1-7.

[4] Panarello, Alfonso, Tapas, Nachiket, Merlino, Giovanni, Longo, Francesco, and Puliafito, Antonio. (2018) “Blockchain and IoT Integration: A Systematic Survey.” Sensors 18 (8): 2575.

[5] Longo, Francesco, Nicoletti, Letizia, Padovano, Antonio, d’Atri, Gianfranco, and Forte, Marco. (2019) “Blockchain-enabled supply chain: An experimental study.” Computers & Industrial Engineering 136: 57-69.

[6] Tanwar, Sudeep, Parekh, Karan, and Evans, Richard. (2020) “Blockchain-based electronic healthcare record system for healthcare 4.0 applications.” Journal of Information Security and Applications 50: 102407.

[7] Lisin, Nikita, Zapechnikov, Sergey. (2020) “Methods and Approaches for Privacy-Preserving Machine Learning”, in: Misyurin S., Arakelian V., Avetisyan A. (eds) Advanced Technologies in Robotics and Intelligent Systems. Mechanisms and Machine Science 80: 141-148.

[8] Liu, Yi, Peng, Jialiang, James J. Q. Yu, and Wu, Yi. (2019) “PPGAN: Privacy-Preserving Generative Adversarial Network.” IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), IEEE, Tianjin, China: 985-989.

[9] Ausloos, Jef. (2012) “The ‘right to be forgotten’–worth remembering?.” Computer law & security review 28 (2): 143-152.Olah, Christopher. (2014) “Neural networks, manifolds, and topology.” Colah's blog, http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/ [Accessed 31 July 2020].

[10] Malle, Bernd, Kieseberg, Peter, Schrittwieser, Sebastian, and Holsinger, Andreas. (2016) “Privacy aware machine learning and the right to be forgotten.” ERCIM news 107 (10): 22-3.

[11] Sweeney, Latanya. (2002) “k-anonymity: A model for protecting privacy.” International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10 (05): 557-570.

[12] Machanavajjhala, Ashwin Kumar Venkatanaga, Kifer, Daniel, Gehrke, Johannes, and Venkitasubramaniam Muthuramakrishnan. (2007) “L-diversity: Privacy beyond k-anonymity.” ACM Transactions on Knowledge Discovery from Data (TKDD) 1 (1): 3-es.

[13] Li, Ninghui, Li, Tiancheng, Venkatasubramanian, Suresh. (2007) “t-closeness: Privacy beyond k-anonymity and l-diversity.” 2007 IEEE 23rd International Conference on Data Engineering, Istanbul, IEEE: 106-115.

[14] Dwork, Cynthia, McSherry, Frank, Nissim, Kobbi, and Smith, Adam. (2016) “Calibrating noise to sensitivity in private data analysis.” Journal of Privacy and Confidentiality 7 (3): 17-51.

[15] Zhao, Jingwen, Chen, Yunfang, and Zhang, Wei. (2019) “Differential privacy preservation in deep learning: Challenges, opportunities and solutions.” IEEE Access 7: 48901-48911.

[16] Hassan, Muneeb Ul, Rehmani, Mubashir Husain, and Chen, Jinjun. "Differential privacy techniques for cyber physical systems: a survey." IEEE Communications Surveys & Tutorials 22 (1): 746-789.

[17] Darwish, Salaheddin, Nouretdinov, Ilia, and Wolthusen, Stephen. (2019) “Modelling the Privacy Impact of External Knowledge for Sensor Data in the Industrial Internet of Things.” Security and Privacy Trends in the Industrial Internet of Things, Springer, Cham: 223-243.

[18] Hou, Jun, Li, Qianmu, Cui, Shicheng, Meng, Shunmei, Zhang, Sainan, Ni, Zhen, and Tian, Ye. (2020) “Low-cohesion differential privacy protection for industrial internet.” The Journal of Supercomputing: 1-23.

[19] Li, Yanhui, Cao, Xin, Yuan,Ye, and Wang, Guoren. (2019) “PrivSem: Protecting location privacy using semantic and differential privacy.” World Wide Web 22 (6): 2407-2436.

[20] Zheng, Xu, Cai, Zhipeng. (2020) “Privacy-preserved data sharing towards multiple parties in industrial IoTs.” IEEE Journal on Selected Areas in Communications 38 (5): 968-979.

[21] Chamikara, Mahawaga Arachchige Pathum, Bertok, Peter, Khalil, Ibrahim, Liu, Dongxi, Camtepe, Seyit, and Atiquzzaman, Mohammed. (2020). “A trustworthy privacy preserving framework for machine learning in industrial iot systems.” IEEE Transactions on Industrial Informatics 16 (9): 6092-6102.

[22] Zhang, Xiaoyu, Chen, Xiaofeng, Liu, Joseph K., and Xiang, Yang. (2019) “DeepPAR and DeepDPA: Privacy Preserving and Asynchronous Deep Learning for Industrial IoT.” IEEE Transactions on Industrial Informatics 16 (3): 2081-2090.

[23]Hao, Meng, Li, Hongwei, Luo, Xizhao, Xu, Guowen, Yang, Haomiao, and Liu, Sen. (2019). “Efficient and privacy-enhanced federated learning for industrial artificial intelligence.” IEEE Transactions on Industrial Informatics 16 (10): 6532-6542.

[24]Chamikara, Mahawaga Arachchige Pathum, Bertok, Peter, Khalil, Ibrahim, Liu, Dongxi, Camtepe, Seyit, and Atiquzzaman, Mohammed. (2020). “Local Differential Privacy for Deep Learning." IEEE Internet of Things Journal 7 (7): 5827-5842.

[25] Rasim M. Alguliyev, Ramiz M. Aliguliyev, and Fargana J. Abdullayeva. (2019) “Privacy-preserving deep learning algorithm for big personal data analysis.” Journal of Industrial Information Integration 15: 1-14.

[26] Olah, Christopher. (2014) “Neural networks, manifolds, and topology.” Colah's blog, http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/ [Accessed 31 July 2020].