ESCAPED: Efficient Secure and Private Dot Product Framework for Kernel-based Machine Learning Algorithms with Applications in Healthcare

Ali Burak Ünal [1], Mete Akgün [1,2], Nico Pfeifer [1,3]

[1]Methods in Medical Informatics, Department of Computer Science, University of Tuebingen, Germany
[2]Translational Bioinformatics, University Hospital Tuebingen, Tuebingen, Germany
[3]Statistical Learning in Computational Biology, Max Planck Institute for Informatics, Saarbrücken, Germany
{uenal,akguen,pfeifer}@informatik.uni-tuebingen.de

Abstract

Training sophisticated machine learning models usually requires many training samples. Especially in healthcare settings these samples can be very expensive, meaning that one institution alone usually does not have enough. Merging privacy-sensitive data from different sources is usually restricted by data security and data protection measures. This can lead to approaches that reduce data quality by putting noise onto the variables (e.g., in ϵ-differential privacy) or omitting certain values (e.g., for k-anonymity). Other measures based on cryptographic methods can lead to very time-consuming computations, which is especially problematic for larger multi-omics data. We address this problem by introducing ESCAPED, which stands for Efficient SeCure And PrivatE Dot product framework. ESCAPED enables the computation of the dot product of vectors from multiple sources on a third-party, which later trains kernel-based machine learning algorithms, while neither sacrificing privacy nor adding noise. We have evaluated our framework on drug resistance prediction for HIV-infected people and multi-omics dimensionality reduction and clustering problems in precision medicine. In terms of execution time, our framework significantly outperforms the best-fitting existing approaches without sacrificing the performance of the algorithm. Even though we only present the benefit for kernel-based algorithms, our framework can open up new research opportunities for further machine learning models that require the dot product of vectors from multiple sources.

Introduction

In the era of data, the same kind of data is produced by multiple sources. Utilizing this variety of sources is one of the easiest ways to satisfy the hunger of machine learning algorithms for data. Often, one can train a machine learning model on the pooled data from different sources to get high accuracy on a particular prediction task. However, gathering data can compromise the sensitive information of the samples in the data. Ayday et al. (2015) showed that genomic data can be used to infer the physical and mental health condition of a patient with the support of information about the patient's lifestyle and environment. Furthermore, Kale, Ayday, and Tastan (2017) introduced a method to keep kinship private in an anonymously released genomic dataset, from which such information could otherwise be inferred. Several studies (Lunshof et al. 2008; Azencott 2018; Bonomi, Huang, and Ohno-Machado 2020) discussed various privacy issues that occurred in studies using medical data from different aspects.

One class of machine learning methods that usually requires gathering the whole data is kernel-based learning methods. To train such a model privately, one of the architectural models in the literature is the distributed model, where each party in the computation has its own data, and the desired kernel matrix contains the whole data that all parties have. Note that throughout this paper, we refer to a source having data or an entity performing a computation as "*party*". Vaidya, Yu, and Jiang (2008) proposed an algorithm that uses such a model to compute the gram matrix of the whole data belonging to the parties in the computation and train a support vector machine (SVM) privately afterwards. The disadvantage of the proposed algorithm is that it focuses only on binary vectors because it utilizes private set intersection to compute the dot product. In addition to the distributed model, there is also the outsourced model where the data is outsourced after encryption and then these encrypted data are used to train a kernel-based machine learning method. Liu, Ng, and Zhang (2015) proposed an approach to use an SVM on the encrypted outsourced data. Due to the nature of encryption, the proposed approach is very time consuming. Zhang et al. (2017) introduced a key-switching (Zhou and Wornell 2014) based secure dot product calculation method. The basic idea is to change the key of the dot product of the vectors, which is originally the combination of the keys utilized to encrypt these vectors, to the key of the server. Ünal, Akgün, and Pfeifer (2019) demonstrated the inefficiency of this method and proposed a randomized encoding based framework to compute the dot product of the vectors of two parties in a third-party, which later trains an SVM model. However, the framework is not extendable to more than two data sources, since this would compromise the data due to the nature of elementwise multiplication of the vectors, which they use to compute the dot product. Furthermore, for the same reason, their approach has a potential privacy leakage for binary encoded data, even for the case with two data sources. We will show that our approach outperforms their framework in such a scenario. Moreover, the randomized encoding itself (Applebaum, Ishai, and Kushilevitz 2006a) is independently applicable to our scenario. The authors claimed that any function expressed by a logarithmic depth arithmetic circuit can be encoded by randomized encoding. In this work, we implemented and applied the randomized encoding based approach and show that it is not as efficient as our framework in terms of the communication cost.

In this paper, we address the privacy problem of data gathering for dot product based algorithms such as kernel-based learning methods. We first implement and apply one of the fastest encodings in the literature, namely the randomized encoding, to our scenario. Due to the inefficiency of the randomized encoding based approach, we come up with a new encoding scheme that enables the secure and private computation of the dot product of vectors. Furthermore, we build a new framework, called efficient secure and private dot product (ESCAPED), which allows multiple data sources, called *input-parties*, to involve in the computation of the dot product. ESCAPED allows a third-party, called *function-party*, to privately obtain the dot product of input-parties' vectors of size larger than 1, while neither gathering the data in plaintext domain nor compromising the privacy of the data. Then, the function-party trains a kernel-based machine learning method. We utilized ESCAPED to predict personalized treatment recommendations for HIV-infected patients in a supervised learning experiment and to perform privacy preserving multi-omics dimensionality reduction and clustering in unsupervised learning experiments. To the best of our knowledge, this is the first study that enables the privacy preserving multi-omics dimensionality reduction and clustering.

Background

Radial Basis Function Kernel

Among the kernel functions, the radial basis function (RBF) kernel is one of the most effective and widely used kernels (Schölkopf, Smola et al. 2002; Kauppi et al. 2015; Pfeifer and Kohlbacher 2008; Zhang et al. 2004). The computation of the RBF kernel for samples $x, y \in \mathbb{R}^n$ can be expressed based on only the dot product of these samples. The formula is as follows:

$$K(x, y) = \exp\left(- \frac{\|\langle x, x\rangle - 2\langle x, y\rangle + \langle y, y\rangle\|^2}{2\sigma^2} \right) \quad (1)$$

where "$\langle \cdot, \cdot \rangle$" represents the dot product of vectors and σ is the parameter that adjusts the similarity level between the samples. Equation 1 indicates that the gram matrix is enough to compute the RBF kernel. We benefit from such computation to obtain the RBF kernel matrix in ESCAPED.

Randomized Encoding

Randomized encoding (RE) is designed to hide the input value s in the computation of a function $f(s)$ by encoding the function with a randomized function $\hat{f}(s; r)$, where r is a uniformly chosen random value (Applebaum, Ishai, and Kushilevitz 2006a,b). The decoding of the encoding reveals only the output of the function f but nothing else.

Applebaum (2017) introduced the perfect decomposable and affine randomized encoding (DARE) of some operations in their study. For a randomized encoding to be affine and decomposable, all components of the encoding should be affine functions over the set on which the function is defined and each of these components should depend on only a single input value and a varying number of random values. Here, we give only the encodings that we used, which are addition and multiplication-addition operations.

Definition 1 (**Perfect RE for Addition (Applebaum 2017)).** Let there be an addition function $t = f(s_1, s_2) = s_1 + s_2$ defined over some finite ring R. The following DARE can perfectly encode such a function:

$$\hat{f}(s_1, s_2; r) = (s_1 + r, s_2 - r)$$

where r is a uniformly chosen random value. The decoding can be done by summing up the components of the encoding, and the simulation of the function can be performed by sampling two random values whose sum is t.

Definition 2 (**Perfect RE for Multiplication-Addition (Applebaum 2017)).** Let there be a function $t = f(s_1, s_2, s_3) = s_1 \cdot s_2 + s_3$ defined over a ring R. The following DARE function $\hat{t} = \hat{f}(s_1, s_2, s_3; r_1, r_2, r_3, r_4)$ can perfectly encode the function f:

$$\hat{t} = (s_1 - r_1, r_2 s_1 - r_1 r_2 + r_3, s_2 - r_2, r_1 s_2 + r_4, s_3 - r_3 - r_4)$$

where r_1, r_2, r_3 and r_4 are uniformly chosen random values. Given the encoding $(c_1, c_2, c_3, c_4, c_5)$, the recovery of $f(s_1, s_2, s_3)$ is done by computing $c_1 \cdot c_3 + c_2 + c_4 + c_5$. In order to simulate $\hat{f}$, one can employ the simulator $\mathsf{Sim}(t; c_1, c_2, c_3, c_4) := (c_1, c_2, c_3, c_4, -c_1 c_3 + t - c_2 - c_4)$.

In addition to the given DAREs, the authors claim that any arithmetic circuit with logarithmic depth can be encoded by a perfect DARE (Applebaum 2017). An example of such an arithmetic circuit that computes the dot product of two vectors is given in the Supplement (Ünal, Akgün, and Pfeifer 2020). Taking this into account, we encode the dot product of the vectors by utilizing the aforementioned encodings. Since we only deal with the private computation of the dot product of the vectors, we optimize the generation of the encoding. Let us assume that we have vectors $x, y \in R^D$, where R is a finite ring and $D \in \mathbb{Z}^+$. In the dot product computation, we have D multiplication nodes in the circuit and the results of these multiplication nodes are summed up by using the addition nodes. To generate the encoding of the dot product of vectors x and y, we first find the largest 2's power smaller than D, which we represent here as P, where $2^q = P$ for $q \in \{\mathbb{Z}^+ \cup \{0\}\}$ and $P < D \leq 2 \cdot P$. We separate the summation of the first P of those multiplication nodes from the summation of the remaining $D - P$ multiplication nodes using Definition 1. We repeat the same procedure for these two parts recursively until we end up with a multiplication node. Once we reach the multiplication node i from an addition node, we utilize the DARE for multiplication-addition given in Definition 2, where $s_1 = x_i$, $s_2 = y_i$ and s_3 represents the resulting value of addition/subtraction of the random values separating summations up to that node. The pseudo code of the randomized encoding generation of the dot product of two vectors of size D and the encoding of the sample arithmetic circuit are given in the Supplement (Ünal, Akgün, and Pfeifer 2020).

Randomized encoding has two main applications, namely, secure computing and parallel cryptography. It is commonly used in multi-party computation (MPC) to minimize the round complexity of MPC protocols (Prabhakaran and Sahai 2013). Thus, more efficient MPC protocols can be designed using randomized encoding.

Methods

In this section, we first explain the scenario employed in the paper. Then, we introduce the randomized encoding based approach and our proposed framework ESCAPED. Later, we give the security definition as well as the security analysis of ESCAPED based on the given definition. Finally, we explain the data we used.

Scenario

We consider a scenario where we have multiple input-parties and a function-party, which computes the dot product of vectors of these input-parties and then trains a kernel-based machine learning algorithm. The real life correspondence of such a scenario would be a study in which a researcher wants to employ the same type of data from different patients collected by multiple hospitals, like cancer subtype discovery. In this scenario, one would like to group cancer patients according to similarities with respect to their omics data. For a new patient, the subtype could give first hints about how severe the cancer is and how well the prognosis is with regard to potential treatments and life expectancy. Due to patient privacy, such data cannot be shared without a permission process, which can significantly slow down the study. However, a framework, like ESCAPED, ensures the protection of the privacy of patients' data, hence enabling the researcher to speed up permission processes and enables studies that would otherwise not be approved due to privacy concerns. While describing the approaches, even though both the randomized encoding based approach and ESCAPED can have multiple input-parties, for simplicity, we use a scenario with three input-parties, namely, Alice, Bob and Charlie with ids 1, 2 and 3, respectively, and a function-party.

Randomized Encoding Based Approach

To address the aforementioned problem, we first implemented a randomized encoding based approach and applied it to our scenario. In this scenario, each of the input-parties has their own data $X \in R^{f \times n_a}$, $Y \in R^{f \times n_b}$ and $Z \in R^{f \times n_c}$, respectively, where f represents the number of features, n_x represents the number of samples in the corresponding input-party and R is a finite ring. Each pair of input-parties needs to communicate separately, i.e., there is communication between Alice and Bob, Alice and Charlie, and Bob and Charlie. For simplicity, we explain only the communication between Alice and Bob. To compute $X^T Y$, they first exchange the size of their own data. Afterwards, Alice generates the scheme of the encoding of the dot product by utilizing the randomized encoding generation algorithm given in the Supplement (Ünal, Akgün, and Pfeifer 2020). Using the resulting encoding scheme, she creates a new set of random values for each possible pair of samples, consisting of one sample from Alice and one sample from Bob. This is quite important in order to protect the relative difference of the features of the input-parties' samples from the function-party. For instance, using the component $s_1 - r_1$ in the encoding, the function-party could learn the relative differences of the input values in the case that the same random value r_1 is utilized for more than one pair of samples. Once Alice created all random values, she sends Bob the part of these random values that he will use to encode his own data. Afterwards, both Alice and Bob encode their data by employing the corresponding random values and send the resulting components to the function-party along with the gram matrix of their own samples. To compute the dot product of samples of Alice and Bob, the function-party combines these components according to the decoding described in Definitions 1 and 2. Such communication is done between all possible pairs of the input-parties, which means that if we have M input-parties, there will be $\binom{M}{2}$ communications in total (more detailed communication cost analysis in Table 1). Once the function-party has all partial gram matrices, it constructs the gram matrix by vertically concatenating the horizontally concatenated partial gram matrices $[X^T X, X^T Y, X^T Z]$, $[Y^T X, Y^T Y, Y^T Z]$ and $[Z^T X, Z^T Y, Z^T Z]$. Then it can compute the desired kernel matrix, which can be computed via the gram matrix, and train a kernel-based machine learning method. The overview of the dot product computation procedure via the randomized encoding based approach is given in the Supplement (Ünal, Akgün, and Pfeifer 2020). Note that in the supervised scenario the input-parties share the labels of the samples with the function-party in plaintext domain since they do not reveal any extra and sensitive information. However, this could easily be extended if more sensitive labels are supposed to be used in the learning process.

ESCAPED

Due to the high communication cost of the randomized encoding based approach to securely compute the dot product of vectors from multiple input-parties in the function-party, we propose a new, efficient and secure framework, called ESCAPED, which is based on a new encoding scheme for the dot product computation. In the computation, the input-parties do not learn anything about the data of the other input-parties or the result of any dot product computed by the function-party. Similarly, the function-party learns only the dot product of the data from the input-parties, but nothing else.

For simplicity, we explain only the computation of the dot product of the data from Alice and Bob in ESCAPED. Figure 1 depicts the overview of ESCAPED. First, Alice and Bob create matrices of random values $a \in R^{f \times n_a}$ and $b \in R^{f \times n_b}$, respectively, where R is a finite ring. Along with these random valued matrices, Alice also creates a random value $\alpha \in R \backslash \{0\}$. Afterwards, Alice computes $X - a$ and αa, and shares them with Bob. In the meantime, Bob computes $Y - b$ and sends it to Alice. Once Alice receives the masked data of Bob, she computes $A_1 = a^T (Y - b)$. Meanwhile, Bob computes $B_1 = (X - a)^T Y$ and $B_2 = \alpha a^T b$. Then, Alice sends A_1 and α, and Bob sends B_1 and B_2 along with the gram matrix of their own samples, which are $X^T X$

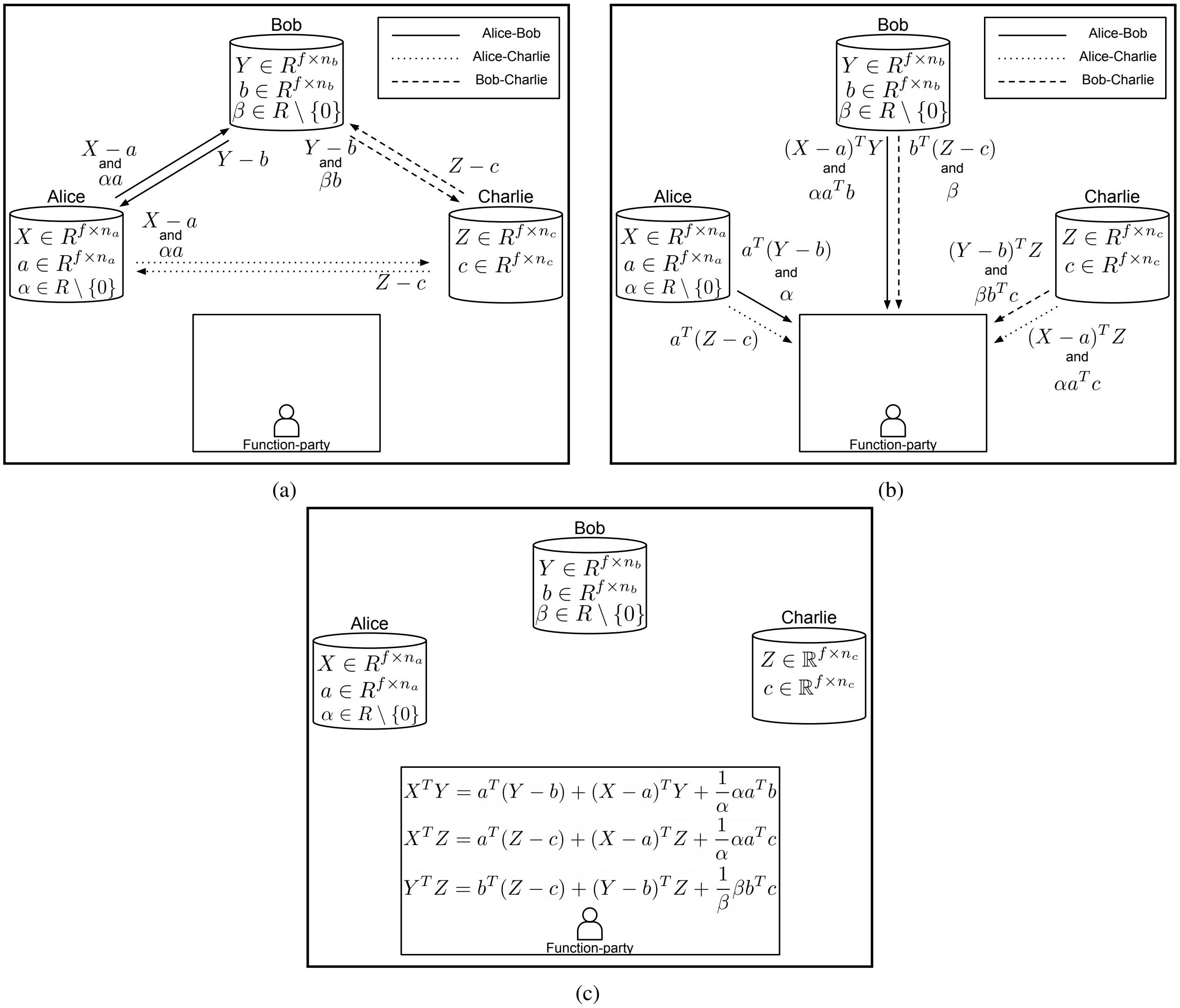

Figure 1: The overview of ESCAPED in our scenario. Each dash type corresponds to a specific part of the gram matrix computed by a pair of input-parties. (a) First, the input-parties exchange their masked input data (e.g. $X - a$) and masked masks (e.g. αa), if applicable. (b) Then, they compute the components of all dot products they are responsible for (e.g. $a^T(Y - b)$) and send them to the function-party along with the mask of the mask (e.g. α), if applicable. (c) The function-party computes the dot product based on the corresponding components of the input-parties.

and $Y^T Y$, respectively, to the function-party. At this point, the function-party computes $A_1 + B_1 + \frac{1}{\alpha} B_2$ to obtain the dot product of the data of Alice and Bob, which is $X^T Y$. Such communication is done similarly among all pairs of input-parties. In these communications, the input-party with a smaller id becomes *"Alice"* and the other becomes *"Bob"*. In the end, the function-party has the gram matrix of all samples. Afterwards, the function-party can compute the desired kernel matrix like the RBF kernel matrix, which can be calculated by using Equation 1, and train a kernel-based machine learning method using the computed kernel matrix to obtain a prediction model. Note that the input-parties share the labels with the function-party in the plaintext domain for of the same reason we mentioned earlier.

Table 1 summarizes the features and the communication cost analysis of ESCAPED, the randomized encoding based approach and the approach proposed by Ünal, Akgün, and Pfeifer (2019).

Security Definition

In our proof, we utilize two different adversarial models, which are the *semi-honest* adversary model, or *honest-but-curious*, and the *malicious* adversary model. A *semi-honest* adversary is a computationally bounded adversary that follows the protocol strictly but also tries to infer any valuable information from the messages seen during the protocol execution. On the other hand, in the *malicious* adversary model, a *malicious* adversary can arbitrarily deviate from the protocol specification. Although the semi-honest model has more restrictive assumptions than the malicious model, it makes the development of highly efficient privacy preserving protocols relatively easy.

Let there be M input-parties $(\mathcal{I}_1, ..., \mathcal{I}_M)$ and a function-party $\mathcal{F}$ in the proposed system. We assume that an adversary is either a semi-honest adversary corrupting a subset of input-parties or a malicious adversary corrupting the function-party. We restrict the collusion between the function-party and the input-parties so as not to allow the corruption of the function-party and at least one input-party at the same time. Otherwise, an adversary $\mathcal{A}$ who corrupts the function-party and at least one input-party obtains the inputs of all other input-parties. Even though we allow the collusion among input-parties, one might think that it is not so realistic because involved entities, such as medical institutions, lose their reputations if they misbehave in this setting.

We use the simulation paradigm (Lindell 2017) in our security proofs. In the simulation paradigm, the security is proven by showing that the simulator can simulate the input and the output of a party, given the actual input and output, such that the simulated input and output cannot be distinguished from the actual ones by an observer. Such an indistinguishability indicates that the parties cannot learn more than what can be learned from their inputs and outputs.

The function-party constructs the final output, i.e. the gram matrix, by using the partial outputs each of which is computed by a pair of input-parties. This enables us to consider these computations as a separate two-party computation. The following notations are used in the security definition:

- Let $f = (f_1, f_2)$ be a probabilistic polynomial-time functionality, where f_p is the input provided by the p-th party to f and let π be a two-party protocol for computing f.
- The view of the i-th party $(i \in 1, 2)$ during an execution of π over (x, y) is denoted by $v_i^{\pi}(x, y)$ and equals $(w, r^i, m_1^i, ..., m_t^i)$, where $w \in \{x, y\}$, r^i equals the contents of the i-th party's internal random tape and m_j^i represents the j-th message it received.
- The output of the i-th party during an execution of π over (x, y) is denoted by $o_i^{\pi}(x, y)$ and can be computed from its own view of the execution. We denote the joint output of both parties by $o^{\pi}(x, y) = (o_1^{\pi}(x, y), o_2^{\pi}(x, y))$.

Definition 3. Let $f = (f_1, f_2)$ be a functionality. We say that a protocol π is secure against semi-honest adversaries if there exist probabilistic polynomial time (PPT) simulators S_1 and S_2 such that:

$$(S_1(x, f_1(x, y)), f(x, y)) \overset{c}{\equiv} (v_1^{\pi}(x, y), o_1^{\pi}(x, y))$$

$$(S_2(y, f_2(x, y)), f(x, y)) \overset{c}{\equiv} (v_2^{\pi}(x, y), o_2^{\pi}(x, y))$$

where $\overset{c}{\equiv}$ denotes the computational indistinguishability. More details can be found in (Goldreich 2009).

Security Analysis

Theorem 1. ESCAPED is secure against a semi-honest adversary $\mathcal{A}$ that corrupts any subset of input-parties.

Proof. The proof is provided in the Supplement (Ünal, Akgün, and Pfeifer 2020). □

Theorem 2. Assume that the function-party is malicious and does not collude with any input-parties. Then, ESCAPED is secure against the malicious function-party $\mathcal{A}$ such that $\mathcal{A}$ cannot infer the data of input-parties from neither the components sent by the input-parties nor the resulting gram matrix.

Proof. The proof is provided in the Supplement (Ünal, Akgün, and Pfeifer 2020). □

Data

In this section, we briefly explain the datasets we employed in our supervised and unsupervised learning experiments, respectively.

***HIV V3 Loop Sequence Dataset*:** To predict the personalized treatmet of HIV-infected patients in the supervised learning experiments, we retrieved the HIV V3 loop dataset from Ünal, Akgün, and Pfeifer (2019). It consists of the protein sequence of the viruses as well as their coreceptor usage information. Due to the availability of drugs blocking the human CCR5 coreceptor, which is exclusively used by the most common variant of HIV to enter the cell, identifying the coreceptor usage is crucial for determining whether or not to use these drugs (Lengauer et al. 2007). The dataset consists of 642 samples for the class "CCR5 only" and 124 samples for the class "OTHER". The sequence data exists as a one-hot encoded data matrix with 766 rows and 924 columns.

***Head and Neck Squamous Cell Carcinoma Dataset*:** We aim to perform the privacy preserving multi-omics dimensionality reduction and clustering on the TCGA data for head and neck squamous cell carcinoma (HNSC) (Cancer Genome Atlas Network et al. 2015) to stratify patients into clinically meaningful subgroups. Therefore, we replicate a recent state-of-the-art study (Röder et al. 2019) in a privacy-preserving setting, obtaining the data from the authors. The data consists of 465 patients with their gene expression (IlluminaHiSeq), DNA methylation (Methylation450k), copy number variation (gistic2), and miRNA expression (IlluminaHiSeq) data. They have 19433, 57159, 23817 and 581 features, respectively. We also obtained the survival times of the patients.

Results

In order to simulate multiple input-parties, we created a process for each input-party and shared the data among them equally. We also created an additional process to simulate the function-party. All processes communicate with each other over TCP sockets and we assume that the communication is secure. We conducted the experiments on a server with has 512 GB memory, an Intel Xeon E5-2650 processor and a 64-bit operating system. We utilized Python to implement ESCAPED and the randomized encoding based approach.

Classification of HIV Coreceptor Usage

In these supervised learning experiments, we used an SVM with an RBF kernel matrix. We optimized the parameters of the SVM, which are the misclassification penalty $C \in$

Approach	Number of IPs		Communication cost		
	Two IPs	Three or more IPs	Among IPS	Between IPs and FP	Total
UAP	Yes	No	$3R^{f \times n^2}$ *	$4R^{f \times n^2}$ *	$7R^{f \times n^2}$ *
RE	Yes	Yes	$4\binom{M}{2}R^{f \times n^2}$	$5\binom{M}{2}R^{f \times n^2}$	$9\binom{M}{2}R^{f \times n^2}$
ESCAPED	Yes	Yes	$3\binom{M}{2}R^{f \times n}$	$3\binom{M}{2}R^{n^2}$	$3\binom{M}{2}(R^{f \times n} + R^{n^2})$

Table 1: The summary of the comparison of the methods utilized in this study from different aspects. The first part of the table presents the ability to handle a varying number of input-parties (IP) in the framework proposed by Ünal, Akgün, and Pfeifer (2019) (UAP), the randomized encoding based approach (RE) and ESCAPED. Moreover, n being the number of samples in each IP, M being the number of IPs and f being the number of features of samples, where $n, M, f \in \mathbb{Z}^+$ and $M \geq 2$, the second part of the table presents the communication cost analysis of RE and ESCAPED in terms of the communication cost among IPs, between IP and the function-party (FP) and the total communication cost. The communication cost analysis of UAP, however, is given without any dependency on M since it can only handle two input-parties scenario. Note that we omit the communication cost of sending the gram matrix of the samples belonging to the same IP since it is fixed for all approaches.

$\{2^{-5}, 2^{-4}, \cdots, 2^{10}\}$ and the weight $w_1 \in \{2^0, 2^1, \cdots, 2^5\}$ of the minority class, and the similarity adjustment parameter $\sigma \in \{2^{-5}, 2^{-4}, \cdots, 2^{10}\}$ of the RBF kernel via 5-fold cross-validation and F1-score. Note that we tuned the parameters outside of the approaches and used them in the experiments directly. However, one can employ both ESCAPED and the randomized encoding based approach for tuning. To have a fair evaluation, we repeated the optimization step 10 times with different random folds and conducted separate experiments by using each optimal parameter set. We evaluated the experiments via F1-score and area under receiver operating characteristic curve (AUROC).

We utilized our proposed framework, ESCAPED, to compute the dot product of samples of three different input-parties on a function-party. Once the function-party has the gram matrix, it computes the RBF kernel matrix based on the optimal σ using Equation 1. We separated 20% of the data of each input-party for testing. The function-party trains an SVM model on the rest of the data by employing the optimal parameters w_1 and C. Then, we tested the model on the test data. Finally, we evaluated the prediction of our model via F1-score and AUROC. We repeated this experiment for each optimal parameter set and obtained 0.843 (± 0.013) AUROC and 0.615 (± 0.016) F1-score on average. To demonstrate the scalability of ESCAPED in terms of the total dataset size, we conducted experiments in which we used a quarter, a half and the full dataset. The execution time of ESCAPED increases almost quadratically with respect to the size of the dataset. Figure 2a shows the trend of the increase in the execution time in parallel to the increment in the dataset size. Furthermore, we analyzed the performance of the framework for varying number of input-parties each of which has the same number of samples. Figure 2b displays the effect of the number of input-parties involved in the computation on the execution time of various parts. The total execution time and the total communication time between input-parties and the function-party (black and red, respectively) are almost linear. The total communication among input-parties (orange), however, displays a slightly different pattern. Since there is an idle party in each turn of the communication among input-parties when there is an odd number of input-parties, the execution time for the cases with an even number of input-parties is almost the same as the case where we have one less input-party.

We also applied the randomized encoding based approach to the same scenario. Similar to the experiments with ESCAPED, we repeated the whole experiment for each optimal parameter set. Since we obtained exactly the same F1-score and AUROC with ESCAPED for the same set of parameters, we demonstrate only the execution time of the randomized encoding based approach for the varying size of the dataset in Figure 2c. When we compared the execution time of the randomized encoding based approach to ESCAPED for full dataset experiments, the randomized encoding based approach took 1.3×10^4 $(\pm 1.4 \times 10^3)$ sec whereas ESCAPED took only 1.19×10^1 $(\pm 3.5 \times 10^{-2})$ sec. Since the randomized encoding based approach is quite inefficient compared to ESCAPED, we did not evaluate it in terms of the number of input-parties. Based on the results and the cost analysis shown in Table 1, it is fair to claim that ESCAPED is more efficient than the randomized encoding approach and other MPC protocols in the literature.

Even though Ünal, Akgün, and Pfeifer (2019) cannot handle three or more input-parties, we compared ESCAPED to this framework in case of a scenario with two input-parties. Since we obtained the same results for both methods, we only give the execution time comparison of them. Figure 2d shows that ESCAPED outperforms their framework. Based on this observation, we can state that ESCAPED is more efficient and comprehensive, especially considering its applicability to more than two input-parties.

*The communication cost analysis is given after an update on UAP to protect the privacy of relative differences between features of samples. Without any update, the communication costs would become $3R^f$, $4R^{f \times n}$ and $3R^f + 4R^{f \times n}$, respectively.

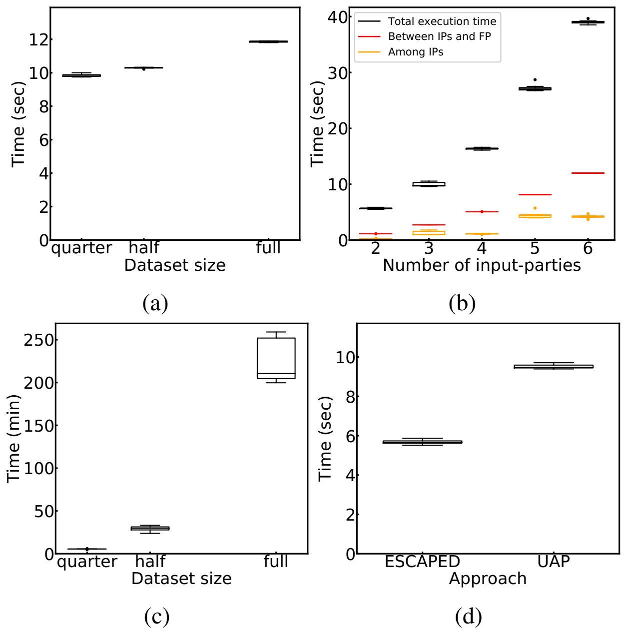

Figure 2: (a) The execution time of ESCAPED is shown for varying sizes of the dataset. (b) The analysis of ESCAPED for varying number of input-parties in terms of the total execution time, the communication time between the input-parties (IP) and the function-party (FP), and the communication time among IPs is shown. (c) The execution time of the randomized encoding based approach is depicted for different sizes of the dataset. The execution time increases quadratically in parallel to the increment in the dataset size. (d) The execution time comparison of ESCAPED and the UAP (Ünal, Akgün, and Pfeifer 2019) for two input-parties case is given.

Clustering of HNSC Cancer Patients

To demonstrate the applicability of ESCAPED on unsupervised learning problems on multi-view data, we employed it to determine biologically meaningful subgroups of cancer patients. Speicher and Pfeifer (2015) studied such a problem and suggested a regularized multiple kernel learning algorithm with dimensionality reduction (rMKL-DR). The method was recently evaluated as the best method in a large benchmark study that compared many different methods (Rappoport and Shamir 2018). Later, Röder et al. (2019) published the online version of the method called web-rMKL. In that study, one of their use cases is the identification of subgroups of HNSC. To stratify patients into biologically meaningful subgroups, they employed four different data types: gene expression, DNA methylation, miRNA expression and copy number variation. They computed one RBF kernel matrix, whose γ is chosen based on a rule of thumb, for each data type and input these kernel matrices to the web-rMKL to obtain the subgroups of patients. They pruned patients whose survival is longer than 5 years. Then they evaluated the results by survival analysis and obtained a $p = 0.0006$ in log-rank test. To show the applicability of ESCAPED, we replicated their study in a privacy preserving way. We utilized the same dataset and split it equally into three input-parties. We employed ESCAPED to compute the kernel matrix for each data type, based on the data belonging to different input-parties. It took 129.17 (± 3.81) sec to compute the required kernel matrices. We then input the resulting kernel matrices to web-rMKL with the same parameter choices to cluster patients. We applied the same filters and evaluated the results by survival analysis as they did. In the end, we obtained the same p-value, indicating that ESCAPED is capable of performing privacy preserving multi-omics dimensionality reduction and clustering. We were unable to conduct these experiments via the randomized encoding based approach due to the excessive memory usage stemming from the inefficiency of the randomized encoding on high dimensional data.

Conclusion

The tension between the unavoidable demand of machine learning algorithms for data and the importance of the privacy of the sensitive information in data urges researchers to come up with efficient and privacy preserving machine learning algorithms. To address this necessity, we introduced ESCAPED to enable the secure and private computation of the dot product in our scenario. In ESCAPED, we preserve the privacy of the data in the computation while neither sacrificing the performance of the model nor adding noise. We demonstrated the efficiency and applicability of ESCAPED on the personalized treatment prediction system of HIV-infected patients and the privacy preserving multi-omics dimensionality reduction and clustering of HNSC patients into biologically meaningful subgroups. Also, we implemented and applied the randomized encoding based approach to solve these problems securely. In the supervised learning problem, both approaches yielded the same result in terms of F1-score and AUROC, but ESCAPED outperformed the randomized encoding based approach in terms of execution time. In the unsupervised learning case, we replicated the state-of-the-art experiments conducted by Röder et al. (2019) in a privacy preserving way without sacrificing performance. This indicates that ESCAPED enables performing privacy preserving multi-omics dimensionality reduction and clustering whereas it was not possible to compute the required kernel matrices with the randomized encoding based approach, which is one of the fastest competitors, due to the excessive memory usage. Even though we applied ESCAPED to two machine learning methods, it is applicable to any method requiring the dot product of the vectors from multiple sources on a third-party, showing the promise of efficiently making other learning algorithms privacy preserving as well. As a future work, other commonly used operations in machine learning algorithms could be included in the framework to extend the scope of the framework. Furthermore, the interpretability of the resulting model could be improved to allow more sophisticated analyses via the model.

Acknowledgements

This study is supported by the DFG Cluster of Excellence “Machine Learning – New Perspectives for Science”, EXC

2064/1, project number 390727645 and by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A. Furthermore, NP and MA acknowledge funding from the BMBF within the 'Medical Informatics Initiative' (DIFUTURE, reference number 01ZZ1804D).

Ethics Statement

Thanks to the promising results of ESCAPED, our study could open up new collaboration opportunities among hospitals, universities, institutes, data centers and many other entities with faster permission processes by providing secure and private computation of dot product enabling, not only the kernel-based learning algorithms but also other methods requiring the dot product. This would help to speed up healthcare research that helps humanity and the world in general. We could not think of a negative ethical impact of our work.

References

Applebaum, B. 2017. Garbled circuits as randomized encodings of functions: a primer. In *Tutorials on the Foundations of Cryptography*, 1–44. Springer.

Applebaum, B.; Ishai, Y.; and Kushilevitz, E. 2006a. Computationally private randomizing polynomials and their applications. *Computational Complexity* 15(2): 115–162.

Applebaum, B.; Ishai, Y.; and Kushilevitz, E. 2006b. Cryptography in NCˆ0. *SIAM Journal on Computing* 36(4): 845–888.

Ayday, E.; De Cristofaro, E.; Hubaux, J.-P.; and Tsudik, G. 2015. Whole genome sequencing: Revolutionary medicine or privacy nightmare? *Computer* 48(2): 58–66.

Azencott, C.-A. 2018. Machine learning and genomics: precision medicine versus patient privacy. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 376(2128): 20170350.

Bonomi, L.; Huang, Y.; and Ohno-Machado, L. 2020. Privacy challenges and research opportunities for genomic data sharing. *Nature Genetics* 1–9.

Cancer Genome Atlas Network; et al. 2015. Comprehensive genomic characterization of head and neck squamous cell carcinomas. *Nature* 517(7536): 576–582.

Goldreich, O. 2009. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press.

Kale, G.; Ayday, E.; and Tastan, O. 2017. A utility maximizing and privacy preserving approach for protecting kinship in genomic databases. *Bioinformatics* 34(2): 181–189.

Kauppi, J.-P.; Kandemir, M.; Saarinen, V.-M.; Hirvenkari, L.; Parkkonen, L.; Klami, A.; Hari, R.; and Kaski, S. 2015. Towards brain-activity-controlled information retrieval: Decoding image relevance from MEG signals. *NeuroImage* 112: 288–298.

Lengauer, T.; Sander, O.; Sierra, S.; Thielen, A.; and Kaiser, R. 2007. Bioinformatics prediction of HIV coreceptor usage. *Nat Biotechnol* 25(12): 1407–1410. URL http://dx.doi.org/10.1038/nbt1371.

Lindell, Y. 2017. How to simulate it–a tutorial on the simulation proof technique. In *Tutorials on the Foundations of Cryptography*, 277–346. Springer.

Liu, F.; Ng, W. K.; and Zhang, W. 2015. Encrypted SVM for outsourced data mining. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, 1085–1092. IEEE.

Lunshof, J. E.; Chadwick, R.; Vorhaus, D. B.; and Church, G. M. 2008. From genetic privacy to open consent. *Nature Reviews Genetics* 9(5): 406.

Pfeifer, N.; and Kohlbacher, O. 2008. Multiple instance learning allows MHC class II epitope predictions across alleles. In *International Workshop on Algorithms in Bioinformatics*, 210–221. Springer.

Prabhakaran, M. M.; and Sahai, A. 2013. *Secure multi-party computation*, volume 10. IOS press.

Rappoport, N.; and Shamir, R. 2018. Multi-omic and multi-view clustering algorithms: review and cancer benchmark. *Nucleic acids research* 46(20): 10546–10562.

Röder, B.; Kersten, N.; Herr, M.; Speicher, N. K.; and Pfeifer, N. 2019. web-rMKL: a web server for dimensionality reduction and sample clustering of multi-view data based on unsupervised multiple kernel learning. *Nucleic acids research* .

Schölkopf, B.; Smola, A. J.; et al. 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Speicher, N. K.; and Pfeifer, N. 2015. Integrating different data types by regularized unsupervised multiple kernel learning with application to cancer subtype discovery. *Bioinformatics* 31(12): i268–i275.

Ünal, A. B.; Akgün, M.; and Pfeifer, N. 2019. A framework for a fast privacy preserving calculation of non-linear kernels for machine learning applications in precision medicine. In *Proceedings of the 18th International Conference on Cryptology and Network Security CANS*.

Ünal, A. B.; Akgün, M.; and Pfeifer, N. 2020. ESCAPED: Efficient Secure and Private Dot Product Framework for Kernel-based Machine Learning Algorithms with Applications in Healthcare. *arXiv preprint arXiv:2012.02688* .

Vaidya, J.; Yu, H.; and Jiang, X. 2008. Privacy-preserving SVM classification. *Knowledge and Information Systems* 14(2): 161–178.

Zhang, J.; Ma, K.-K.; Er, M.-H.; and Chong, V. 2004. Tumor segmentation from magnetic resonance imaging by learning via one-class support vector machine. In *International Workshop on Advanced Image Technology (IWAIT'04)*, 207–211.

Zhang, J.; Wang, X.; Yiu, S.-M.; Jiang, Z. L.; and Li, J. 2017. Secure Dot Product of Outsourced Encrypted Vectors and its Application to SVM. In *Proceedings of the Fifth ACM International Workshop on Security in Cloud Computing*, 75–82. ACM.

Zhou, H.; and Wornell, G. 2014. Efficient homomorphic encryption on integer vectors and its applications. In *2014*

Information Theory and Applications Workshop (ITA), 1–9. IEEE.