Journal of Siberian Federal University. Mathematics & Physics 2015, 8(4), 478–486

УДК 004.056.5

Privacy-Preserving Building of Self-Organizing Maps

Alexey V. Vashkevich[*]
Vadim G. Zhukov[†]
Eugene S. Semenkin[‡]
Institute of Mathematics
University of Potsdam
Am Neuen Palais, 10, Potsdam, 14469
Germany

Received 03.06.2015, received in revised form 09.07.2015, accepted 24.08.2015

Various data mining techniques are designed for extracting significant and valuable patterns from huge databases. Today databases are often divided between several organizations for the reason of limitations like geographical remoteness, but the most important limit is preserving privacy, unwillingness of data disclosing. Every party involved in analysis wants to keep its own information private because of legal regulations and reasons of know-how. Secure multiparty computations are designed for data mining execution in a multiparty environment, where it is extremely important to maintain the privacy of the input (and possibly output) data. A self-organizing map is the data mining method by which analytics can display patterns on two-dimensional intuitive maps and recognize data clusters. This article presents protocols for preserving privacy in the process of building self-organizing maps. The protocols allow the implementation of a self-organizing map algorithm for two parties with horizontally partitioned data and for several parties with vertically partitioned data.

Keywords: secure multiparty computations, secure dot product, cluster analysis, self-organizing map.
DOI: 10.17516/1997-1397-2015-8-4-478-486

Introduction

Clustering is widely used in data analysis in many examples of applied and theoretical research [1]. In some cases of this research information may be split between several organizations or individual users. This information may be private, so traditional conducting of a cluster analysis will disclose private data to all (or almost all) other participants. The database partition may be horizontal, when each party owns a full dataset of some objects, or vertical, when each party owns some attributes of each object. In these situations it is necessary to use secure multiparty protocols for processing private input data.

Building of self-organizing maps is one of the most common and used methods of data analysis in which analytics may face the problem of keeping the privacy of input data. These maps are built by a neural network and solve the task of clustering. The main advantage of this data mining algorithm is the projection of multidimensional space on several two-dimensional maps, which show the evident final result. This clustering algorithm is particularly relevant at the stage

[*]alex23-5@yandex.ru
[†]vadimzhukov@mail.ru
[‡]eugenesemenkin@mail.ru

of "exploratory" analysis, when analytics have huge arrays of raw information, but they do not know anything about patterns in it. Nowadays there is one protocol of preserving the privacy of self-organizing maps in the vertical partition for two parties [2], but this algorithm cannot be extended to any other number of parties or to the horizontal partition.

The main contribution of this article to the research area of secure multiparty computations is:

1. A protocol developed for any number of parties with vertically partitioned data. The data of any party should not be disclosed even if all other parties collude with each other and combine their knowledge.

2. A protocol developed for two parties with horizontally partitioned data.

Both developed protocols should correspond to the following condition: preserving privacy should not bring inaccuracy into the result of analysis. In other words, the protocols should allow a correct building of self-organizing maps as it would be built for one huge database without preserving privacy.

This article is organized as follows: Section 1 contains a review of related work in privacy-preserving clustering. Section 2 focuses on the algorithm of self-organizing maps. Section 3 describes the use of a secure dot product as the basis for preserving data in the developed protocols. Sections 4 and 5 describe protocols for vertical and horizontal data partitioning respectively. This is followed by a conclusion and directions for further research in Section 6.

1. Related work

Andrew Yao first postulated the problem of secure multiparty computation in [3] and developed a mathematically proven solution. Yao's work describes a hypothetical situation, when two millionaires want to find who is richer, but both of them do not want to disclose the exact amount of their fortunes. The first work of privacy-preserving in data mining algorithm was given by [4] and describes the ID3 algorithm for decision tree building. Within a few years different authors published several more works of secure multiparty algorithms like the Bayes classifier [5] and association rules [6].

To the best of our knowledge, there are two works to date related to preserving the privacy of self-organizing maps (hereinafter SOM). In [2], the authors suggested a privacy-preserving protocol for only two parties with only the vertical data partition. Achieving data privacy, the authors use one of the protocols of secure dot product (hereinafter SDP) for finding the winner neuron. According to this SDP protocol [7] the parties should perform three interactions in each computation session.

In other work related to preserving the privacy of SOM [8], authors did not present any secure multiparty protocol, they only suggested the main idea: using principal component analysis for decreasing the count of data dimensions. According to the authors, principal component analysis "hides" private data, but it is not clear how parties should interact and how to disclose intermediate data (like the positions of neurons) during the algorithm. Also using principal component analysis does not allow fully corrected cluster analysis as it is provided without any previous data modification.

Due to the fact that there is no full analogue of our work, we should focus attention on other clustering privacy-preserving algorithms like K-means. This clustering algorithm has a lot of privacy-preserving protocols for any data partition and participant count [9].

Each work uses a different scheme of preserving privacy. Several solutions [10, 11] are based on a scheme with non-colluding parties that is usually unwanted in the real world. If some of these parties collude with each other then they can reveal the secret value of other parties.

Some of other works are designed for an arbitrary data partition like [12, 13], but this partition is rare in practice. Also these solutions are redundant when it is used with vertical or horizontal partition because it is required to protect all steps of the original algorithm. In contrast, protocols designed for just one partition should protect only part of the K-means steps, which decreases computational and communication complexity of secure protocols.

There are two popular ways of protecting private input in distributed computations: homomorphic encryption and random values. Homomorphic encryption has a huge computational cost compared with simple mathematical operations on small numbers; it is usually based on the mathematical difficulty of calculating inverse function, but theoretically an attacker may decompose it using large computing power. In contrast, random values hide private data by increasing the count of unknowns. It is impossible to definitely solve the undetermined system of linear equations when it has more unknowns than equations.

The most appropriate scheme for our work is presented in [14]. This solution is based on random values and it does not have any number of non-colluding parties while using the secure dot product [15]. Another advantage of the solution is that the dot product protocol requires only two interactions in each computational session unlike in [7]. The original solution was modified in [16] — there is an explanation of using three-dimensional vectors in SDP instead of two-dimensional and real numbers instead of finite fields.

2. Self-organizing maps

Self-organizing maps allows analytics to display clusters of data objects on several two-dimensional images [17]. The basic principle of the algorithm is taking into account the mutual location of neurons during the training of the neural network. Algorithm 1 shows the scheme of building SOM for a local database by only one participant of analysis.

Algorithm 1: Building SOM

```
  Data: dataset $X$, two-dimensional network of neurons $K$, function for change $h(t)$, threshold
        number of iterations $T$.
  Result: set of two-dimensional maps for every dimension; aggregate two-dimensional map for all
        dimensions
1 Initialization (random or by any rule) of parameters of all neurons $k_j$, $count = 0$;
  while $count < T$ do
2     Random selection of object $x_i$ from $X$;
3     Finding closest neuron $k_j$ to object $x_i$: $d_{ijmin} = |x_i - k_j|$;
4     Updating coordinates of closest neuron and its neighbours with respect to the object $x_i$ using
      function for change $h(t)$, where $t$ is iteration number: $k'_j = k_j + h(t) * |x_i - k_j|$;
5     $count = count + 1$
```

The stop condition may differ from described in the Algorithm 1, but typically it is reaching a certain number of iterations. There may be other stop conditions, but usually they are protected by the same tools that preserve privacy in described following steps.

During the building of SOM it is required to find the shortest distance between the object data and the neuron; it allows to find the nearest neuron to the object on each iteration. The cryptographic primitive that protects this step should provide a result of the comparison of two distances with each other. As a cryptographic primitive that is capable of this, it is proposed to use a modified secure dot product, based on [15]. This SDP protocol allows the dot product to be calculated without disclosing multiplied vectors (Equation 1):

$$\vec{x} \cdot \vec{y} = (x_1, x_2, ..., x_n) \cdot (y_1, y_2, ...y_n) = \sum_{i=1}^{n} x_i y_i \tag{1}$$

Using SDP for calculating the sum of several summands is listed in Section 3 and is almost the same as the scheme described in [14] except that it uses three-dimensional vectors instead of two-dimensional and real numbers instead of finite fields, as described in [16]. Section 3 has no new developments, but it is necessary for understanding the mechanisms of data protection listed in Sections 4 and 5.

3. Using SDP for finding multiparty sum

The sum of two private summands $x_1 + x_2$ can be represented as a product of two private multipliers $r_1 \cdot r_2$ according to Algorithm 2.

Algorithm 2: The sum of two summands

```
1 Party 1 generates random numbers $r_1 \neq 0$ and $z$, and then creates vector
      $X_1 = (\frac{x_1 - z}{r_1}; \frac{z}{r_1}; \frac{1}{r_1})$
2 Party 2 creates vector
      $X_2 = (1; 1; x_2)$
3 Parties 1 and 2 run SDP, and Party 2 gets the result
      $r_2 = \frac{x_1 + x_2}{r_1}$
```

The sum of three private summands $x_1 + x_2 + x_3$ can be represented as a product of three private multipliers $r_1 \cdot r_2 \cdot r_3$ according to Algorithm 3.

Similarly, as the sum of three summands, analytics can convert the sum of any number of summands into the product of the same number of multipliers (Equation 2):

$$\sum_{i=1}^{n} x_i = \prod_{i=1}^{n} r_i, \tag{2}$$

where x_i and r_i are known only by Party i.

4. Algorithm with vertical partition

With vertically partitioned data each party owns only some of the dimensions of each data object x_i, and the same dimensions of each neuron k_j. Thus, there is no need to protect step 4 of the Algorithm 1, because each party can locally (without data exchange with other parties) "move" the neurons in those dimensions that the party has. However, step 3 must be protected, because the distances between the objects and the neurons are calculated in all dimensions of all parties.

Algorithm 3: The sum of three summands

```
1 Party 1 splits its own data $x_1$ into two random parts $x_1 = x_{12} + x_{13}$ and generates random
  number $r_1 \neq 0$;
2 Parties 1 and 2 sum their summands $x_{12}$ and $x_2$ according to the Algorithm 2. Party 2 gets the
  result
      $s_2 = \frac{x_{12} + x_2}{r_1}$
3 Parties 1 and 3 sum their summands $x_{13}$ and $x_3$ according to the Algorithm 2. Party 3 gets the
  result
      $s_3 = \frac{x_{13} + x_3}{r_1}$
4 Party 2 generates random number $r_2 \neq 0$;
5 Parties 2 and 3 sum their summands $s_2$ and $s_3$ according to the Algorithm 2. Party 3 gets the
  result
      $r_3 = \frac{s_2 + s_3}{r_2}$
6 Thus, together parties convert sum of three summands into product of three multipliers:
      $x_1 + x_2 + x_3 = r_1 \cdot (s_2 + s_3) = r_1 \cdot r_2 \cdot r_3$
```

We denote the set of dimensions of the object x, belonging to Party i, as $\{x_{i1}, x_{i2}, ..., x_{im}\}$, where m is the number of dimensions stored by Party i. The set of dimensions of neuron k_j stored by Party i is denoted as $\{k_{ij1}, k_{ijm}, ..., k_{ijm}\}$. Firstly, each party calculates its own local distance between object x and neuron k_j in its own dimensions. For example, the local distance d_{ij} is calculated according to Equation 3:

$$d_{ij} = |x_{i1} - k_{ij1}|^A + |x_{i2} - k_{ij2}|^A + ... + |x_{im} - k_{ijm}|^A, \tag{3}$$

where A is the degree used in the metric (1 in Manhattan distance, 2 in Euclid distance etc.).

The local distance d_{ij} between the data object and the neuron is private. All parties should sum together their local distances which in result will give them the full distance between the object and the neuron. The distance between x and k_j is shown in Equation 4:

$$|x - k_j| = d_{1j} + d_{2j} + ... + d_{Pj}, \tag{4}$$

where P is the number of parties.

In the same way parties may get other distances between object x and other neurons. However, according to algorithm of building SOM there is no need to find these distances, because analytics should only compare the distances with each other for finding minimum. It is necessary to subtract one distance from another, as it is shown in Equation 5, for the determination of which neuron is closer to object x:

$$|x - k_j| - |x - k_s| = (d_{1j} - d_{1s}) + (d_{2j} - d_{2s}) + ... + (d_{Pj} - d_{Ps}) \tag{5}$$

The computation of difference $d_{ij} - d_{is}$ is performed by each party locally too and is private. That is why Equation 5 may be represented as the sum of several private summands; each party has only one of these summands. Using Equation 2 parties represent the difference between distances $|x - k_j| - |x - k_s|$ as the product of several private multipliers. Parties need to multiply only the signs of their private multipliers for the determination of which distance is greater than another (Equation 6):

$$|x - k_j| < |x - k_s| \equiv \prod_{i=1}^{P} sign(r_i) < 0 \tag{6}$$

It should be noted that in some metrics the sum of local distances (presented in Equation 4) is put under the root, but this mathematical operation does not affect the comparison of two distances, because (Equation 7):

$$a < b \Rightarrow \sqrt[n]{a} < \sqrt[n]{b} \tag{7}$$

if $n > 0$, $a > 0$, $b > 0$.

In this way analytics will find the closest neuron to the data object x. Using this algorithm parties cannot disclose the private data of another party or even distances between objects and neurons. Even if $P - 1$ parties collude against one victim party, they will find only a very long approximate interval which contains a victim's local difference $d_{ij} - d_{is}$ and nothing else.

Thus, step 3 of the Algorithm 1 is protected, therefore the developed protocol preserves information privacy during the process of building a SOM by any number of parties with the vertical partition.

5. Algorithm with horizontal partition for two parties

With the horizontal partition, each party has its own set of data objects, but neuron data is not belonged for any concrete party. During building of SOM parties at one time manipulate with only one data object belonged only one party. Other party can indirectly determine data of private object by neuron's relocation during the building of SOM — so neuron's data should be encrypted. It it suggested to split neuron's data into random shares (in this protocol — into two shares) between parties. Since the neuron data is split, the protocol should protect only the steps that use mathematical operations with neuron data. These are steps 3 and 4 of the Algorithm 1.

Parties may disclose to each other these random shares of the neurons only by mutual agreement and only after the SOM has finished being built. The disclosure of the locations of neurons will slightly help parties in the analysis of the results: for example, disclosure will show the coordinates of the so-called "dead" neurons — neurons that are not closest to any data object of each party. However, this disclosure may also help a curious party to better determine the approximate location of some data objects of another party, so such disclosure is not recommended for analytics even after building the SOM.

5.1. Finding closest neuron

Suppose that analytics at step 3 of the Algorithm 1 randomly select data object x kept by Party 1, (for objects kept by Party 2 the following scheme will be changed mirror-like). We will denote the set of dimensions of data object x kept by Party 1 as $\{x_1, x_2, ..., x_m\}$, where $m > 1$ is amount of dimensions. Neuron k_j is split into two random shares k_{j1} and k_{j2}, each share has a full set of dimensions; the sum of two private vectors $k_{j1} + k_{j2}$ will give original neuron k_j. For each dimension i of data object x parties calculate the distance between the object and neuron k_j according to Equation 8:

$$(x_i - k_{j1i}) - k_{j2i} = r_{j1i} \cdot r_{j2i} \tag{8}$$

where $x_i - k_{j1i}$ Party 1 calculates locally.

The final calculation of the difference between object x and neuron k_j is presented in Equation 9:

$$|x - k_j| = |r_{j11} \cdot r_{j21}| + |r_{j12} \cdot r_{j22}| + ... + |r_{j1m} \cdot r_{j2m}| \tag{9}$$

For comparing two distances like $|x - k_j| - |x - k_s|$ parties should calculate the result of the dot product (Equation 10) using SDP:

$$(|r_{j11}|, ..., |r_{j1m}|, |r_{s11}|, ..., |r_{s1m}|) \cdot (|r_{j21}|, ..., |r_{j2m}|, -|r_{s21}|, ..., -|r_{s2m}|) \tag{10}$$

SDP is conducted in this way that the result of comparing distances is attained by Party 1 who owns the object x. Despite the fact that Party 1 knows exactly how much closer (in the Manhattan metric) object x is to neuron k_j, than to neuron k_s (or vice versa), it is impossible to calculate the locations of neurons, because Party 1 still doesn't know the neuron data kept by Party 2.

Comparing pairwise distances from the object x to the different neurons, Party 1 calculates the nearest neuron to the object. At the further step 4 of the Algorithm 1 Party 2 anyway will know which neuron is the winner (at step 4 only the winner neuron's coordinates and its neighbours' coordinates will be updated), so it does not make sense for Party 1 to hide the results of pairwise comparisons of neurons from Party 2. The advantage of this situation is that it is enough to calculate only the $K - 1$ comparings of the distance to find the winner neuron; parties shouldn't compare all distances with each other, that will take $(K \cdot (K - 1))/2$ comparings, where K is the number of neurons.

During step 3 of the Algorithm 1 Party 2 has learned just which neurons are closest to the data object x. After that, if Party 2 during future iterations will have one or more objects that are close to the same neurons, then Party 2 will be able to say that "an object" of Party 1 is placed "relatively close" in space. But Party 2 cannot say this more definitely because neurons move in space during the execution of the protocol.

5.2. Updating neurons' coordinates

At the previous step 3 of the Algorithm 1 Party 1 finds the closest neuron k to the object x. There is a need to update the coordinates of neurons in proportion to the difference between x and k in each dimension. During the search for the winner neuron parties presented differences between neurons and objects for each dimension in the product of two multipliers, as shown in Equation 8.

Firstly, both parties add random numbers to their shares of the neuron's dimension so that $k'_{1i} = k_{1i} + z_1; k'_{2i} = k'_{2i} + z_2$. Then parties apply Equation 8 again, and in SDP Party 2 uses the same multiplier r_{2i}, as at step 3 of the Algorithm 1. However, Party 1 gets another multiplier r'_{1i}. If multiplier r'_{1i} is suddenly equal to the original multiplier r_{1i}, parties randomly select another z_1 and z_2 and the procedure repeats.

The multiplier r'_{1i} should be equal to $h(t) \cdot r_{1i}$. However, in reality, r'_{1i} and $h(t) \cdot r_{1i}$ are most likely not equal, so Party 1 calculates how many times the difference in dimension i between object and neuron $x_i - k_{j1i}$ has decreased or increased. That allows both parties to correct their own random numbers z_1 and z_2. The correction is shown in Equation 11:

$$z' = z \cdot \frac{r_{1i} \cdot h(t)}{r_{1i} - r'_{1i}} \tag{11}$$

During step 4 of the Algorithm 1 parties still do not know the locations of neurons or any exact information about the objects of other parties.

6. Conclusions

Building self-organizing maps is a popular method of data clustering; due to its clarity it makes data analysis easier, even if there is not enough knowledge about the possible nature of patterns. In many cases, preserving the privacy of such data is very important, which results in development and using secure protocols. This article presents a description of the developed protocols that provide the private building of a self-organizing map with the vertical partition (for any number of parties) and the horizontal partition (for two parties). These protocols are based on the secure dot product protocol, allowing the calculation of sums of several private summands without their disclosure, and to securely compare sums with each other.

In further research it is acceptable to improve these protocols, for example, for the acceleration of these protocols, and to create a protocol for the horizontal partition for any number of parties. Also there are many other different clustering algorithms, that haven't proven secure protocols. The protection of hierarchical clustering seems especially perspective. Most likely, the data protection of these clustering algorithms will be based on the same principles as in this article.

References

[1] A.K.Jain, Data clustering: 50 years beyond K-means, *Pattern Recognition Letters*, 2010, no. 31, 651–666.

[2] S.Han, W.K.Ng, Privacy-Preserving Self-Organizing Map, *Proceedings of the International Conference DaWaK*, **9**(2007), 428-437.

[3] A.C.Yao, Protocols for Secure Computations, *Proc. of the Symposium on FOCS*, **23**(1982), 160–164.

[4] Y.Lindell, B.Pinkas, Privacy preserving data mining, *Advances in Cryptology*, 2000, vol. 1880, *Lecture Notes in Computer Science*, 36–54.

[5] M. Kantarcioglu, J.Vaidya, Privacy preserving naive bayes classifier for horizontally partitioned data, *IEEE ICDM Workshop on Privacy Preserving Data Mining*, 2003, 3–9.

[6] M.Kantarcioglu, C.Clifton, Preserving data mining of association rules on horizontally partitioned data, *IEEE Transactions on Knowledge and Data Engineering*, **16**(2004), no. 9, 1026–1037.

[7] B.Goethals, S.Laur, H.Lipmaa, Mi T.elikainen, On private scalar product computation for privacy-preserving data mining, *Proceedings of the Annual International Conference ICISC*, **7**(2004), 104–120.

[8] R.Vidyabanu, N.Nagaveni, A Model Based Framework for Privacy Preserving Clustering Using SOM, *International Journal of Computer Applications*, **1**(2010), no. 13, 17–21.

[9] F.Meskine, S.N.Bahloul, Privacy preserving k-means clustering: a survey research, *International Arab Journal of Information Technology*, **9**(2012), no. 2, 194–200.

[10] J.Vaidya, C C.lifton, Privacy-preserving k-means clustering over vertically partitioned data, *Proc. of the ACM SIGKDD international conference on Knowledge discovery and data mining*, **9**(2003), 206–215.

[11] M.Doganay, P T.edersen, Y.Saygin, E.Savas, A.Levi, Distributed Privacy Preserving Clustering with Additive Secret Sharing, *Proceedings of the International Workshop on PAIS*, 2008, 6003–6011.

[12] M.Upmanyu, N A.M.amboodiri, K.Srinathan, C.V.Jawahar, Efficient privacy preserving k-means clustering, *Intelligence and Security Informatics*, **6122**(2010), 154–166.

[13] J N.B.inwala, G.B.Jethava, Privacy Preserving Using Distributed K-means Clustering for Arbitrarily Partitioned Data, *International Journal of Engineering Development and Research*, **2**(2014), no. 2, 2291–2295.

[14] S.Samet, A.Miri, L.Orozco-Barbosa, Privacy Preserving k-Means Clustering in Multi-Party En A.vironment, Proceedings of the International Conference SECRYPT, 2007, 381–385.

[15] B.Malek, A.Miri, Secure dot-product protocol using trace functions, Proceedings of the IEEE ISIT, 2006, 927–931.

[16] V.G.Zhukov, A.V.Vashkevich, Modified secure dot product in privacy-preserving clustering k-means with vertically partitioned data, *Vestnik SibGAU*, **50**(2013), no. 4, 18–22 (in Russian).

[17] Kohonen T. Self-Organized Formation of Topologically Correct Feature Maps, *Biological Cybernetics*, **43**(1982), no. 1, 59–69.

Построение самоорганизующихся карт с сохранением конфиденциальности

Алексей В. Вашкевич
Вадим Г. Жуков
Евгений С. Семенкин

Существует множество алгоритмов анализа данных, предназначенных для поиска значимых закономерностей в больших базах данных. Такие базы данных часто бывают разбиты между несколькими организациями по различным причинам например, из-за географической удаленности, но, как правило, самая важная причина — обеспечение конфиденциальности, нежелание раскрывать данные друг другу. Каждый участник анализа хочет сохранить свои данные конфиденциальными, чтобы выполнить требования нормативно-правовых актов или сохранить ноу-хау. Конфиденциальные многосторонние вычисления разработаны для проведения анализа данных несколькими участниками, когда крайне важно сохранить конфиденциальность входных (и иногда выходных) данных. Самоорганизующиеся карты — это метод анализа данных, с помощью которого аналитики могут отобразить закономерности на двумерных интуитивно понятных картах и визуально распознать кластеры данных. В статье представлено описание протоколов обеспечения конфиденциальности при построении самоорганизующихся карт. Эти протоколы позволяют строить самоорганизующиеся карты двум участникам при горизонтальном секционировании данных и нескольким участникам при вертикальном секционировании.

Ключевые слова: конфиденциальные многосторонние вычисления, безопасное скалярное произведение, кластерный анализ, самоорганизующиеся карты.