Privacy preservation for Neural Network Training

Ms. Ch.Aparna,
Department of Computer Science and Engineering
RVR&JC College of Engineering,
Guntur, India
achaparala@gmail.com

Ms. K.Venkata Ramana,
Department of Computer Science and Engineering
RVR&JC College of Engineering,
Guntur, India
vinoliamanohar@gmail.com

Abstract— Data mining enables us to extract interesting patterns from huge volumes of data. This can be a security/privacy issue, if we make sensitive information available to unintended users as a result of mining. Preserving the privacy of sensitive data and individuals' information is a major challenge in many applications. Privacy concerns may prevent the parties from directly sharing the data and some types of information about the data. Back propagation (BP) is one of the algorithms used for training neural networks. This algorithm is suitable for both single-layer and multi-layer models. Another neural network training algorithm values for weights. Training the network is achieved by iteratively adjusting the weights based on a set of input parameters. During this process a training sample is presented to the network and is forwarded to determine the resulting output. The difference between the obtained output and expected output in each output unit represents error. Error is back propagated is the Extreme Learning Machine (ELM) algorithm. This algorithm works only for single-layer models. This restriction makes ELM more efficient than BP by reducing the communication between multiple parties during the learning phase. Most of the neural network learning systems is designed for single-layer and multi-layer models which can be applied to continuous data and differentiable activation functions. The present work proposes a protocol for neural network training systems that guarantee data confidentiality when data is partitioned among several parties. The proposed protocol preserves the privacy of both the input data and the constructed learning model.

Keywords— Data Confidentiality, Neural Networks, Back Propagation, Extreme Learning Machine, Paillier Encryption

I. INTRODUCTION

Neural Networks have been an active research area for decades. Trained neural networks can predict efficient outputs which might be difficult to obtain in the real world. When training neural network from distributed data, privacy is a major concern. With the invention of new technologies, whether it is data mining, in databases or in any networks, resolving privacy problems has become very important. Because all sorts of data are collected from many sources, the field of machine learning is equally growing and so are the concerns regarding the privacy.

Data mining, the discovery of new and interesting patterns in large datasets, is an exploding field. Data mining techniques are used in business and research and are becoming more and more popular with time [4]. Data mining can extract important knowledge from large data collections about individuals. Protecting private data is an important concern for society. Data mining is under attack from privacy advocates because of a misunderstanding about what it actually is and a valid concern about how it’s generally done. A key problem that arises in any mass collection of data is confidentiality. The generic solutions are not efficient, especially when large inputs and complex algorithms are involved. This rises a need for suitable data privacy techniques.

Privacy bothers many when the training dataset for the neural networks is distributed between two parties, which is quite common nowadays [18]. Existing algorithms provide neural network training by compromising privacy. In this paper we present an algorithm for preserving confidentiality in the neural network learning. We show that our algorithm is very secure and leaks no knowledge about other party’s data.

The back propagation technique is a gradient descent method that establishes the weights in a multi-layered, feed forward neural network [15,16]. This method initially, chooses random through the network in order to adjust the weights. The training process continues until the root mean squared error from the output signal falls below a pre specified threshold.

Following are the steps of back propagation algorithm.

1. Apply the inputs to the network, forward the inputs and work out the output.
2. Next work out the error for neuron B.

$$\text{ErrorB} = \text{OutputB}\ (1\text{-OutputB})(\text{TargetB} - \text{OutputB})$$

The "Output(1-Output)" term is necessary in the equation because of the Sigmoid Function – if only a threshold neuron is being used, then it would just be (Target – Output).

3. Change the weight. Let W+ AB be the new (trained) weight and WAB be the initial weight.

$$W+\ AB = WAB + (\text{ErrorB} \times \text{OutputA})$$

Update all the weights in the output layer in this way.

4. Calculate the Errors for the hidden layer neurons. Unlike the output layer these can't be calculated directly, so these are Back Propagated from the output layer. This is done by taking the Errors from the output neurons and running them back through the weights to get the hidden layer errors.
5. Having obtained the Error for the hidden layer neurons now proceed as in stage 3 to change the hidden layer weights.

Extreme Learning Machine (ELM) is a learning algorithm that is much faster than the conventional gradient-based neural network training algorithms. It is proposed by G. Huang et al. [9]. ELM considers generalized Single Layer Feed Forward Networks (SLFNs) whose hidden layer need not be tuned [9]–[12]. It was originally developed for the single-hidden-layer feed forward neural networks. It randomly selects nodes in the hidden layer, and analytically computes the output weights of the network, faster and with better average performance than the BP algorithm.

In the basic ELM, Hidden node parameters (ai, bi) remain fixed after randomly generated [10]. To train a single layer network is simply equivalent to finding a least-squares solution $\dot{\alpha}$ of the linear system $H\alpha = T$: $\|H\ \dot{\alpha} - T\| = \min\alpha \|H\ \alpha - T\|$

The steps of basic ELM training algorithm are as follows:

Inputs:

Training set $S = \{(xi,ti) \mid xi \in Pd, ti \in Pm, i = 1, \ldots, S\}$, an activation function, and L, the number of hidden nodes.

Hidden node output function F (ai, bi, x),

a) Randomly generate hidden node parameters (ai, bi), i =1, . . . , L., where L is hidden node number.
b) Calculate weights and thresholds for each hidden node.
c) Calculate the hidden layer output matrix H.
d) Calculate the output weight vector α : $\alpha = H\dagger T$

Training a neural network requires the training samples from data owners. But in many cases, data owners are not willing to share their private data for training the network at the expense of compromising on privacy. In some countries, there are acts like HIPAA, Health Insurance Portability and Accountability Act rule [13] to prohibit using or distributing individual's personal data. Even the insurance companies have to take authorization to reveal anyone's health-related data [5]. Often, the training data samples used can be partitioned in two ways: Horizontally or Vertically. With horizontal partitioning, each party holds a subset of their own samples. In this case of network training, it does not pose a significant privacy threat, since each data holder can train the network during his/her turn. Vertical partitioning distributes columns of dataset across multiple parties [17]. Some columns of the dataset are with one party and other parties hold the remaining columns. This section discusses privacy preserving extensions of two popular network training algorithms BP and ELM when data is either horizontally or vertically partitioned. There is a general purpose method in cryptography known as secure multi-party computation which can be applied during neural network training for privacy preservation [1]. This protocol can compute any probabilistic polynomial function privately.

However, this general solution is very expensive to be applied to many practical problems and so is infeasible.

II. RELATED WORK

Chris Clifton, Murat,Jaideep vaidya and Xiadong Lin[6] suggested a toolkit of components for privacy preserving in data mining applications.The toolkit secures data in single party computation.Their results shows that the toolkit does not solve multi party computations.

A. Evfimievski, J. E. Gehrke, and R. Srikant [7] define privacy breach as a property by which some private information of a client can be found by the server with high probability. They provide a new technique "Amplification" which ensures limitation on privacy breaches which occur using normal randomization technique

while protecting privacy. The authors claim to produce high quality limitation of privacy breach without the information regarding the distribution of the data.

H. Kargupta, S. Datta, Q. Wang,and K. Sivakumar [8] questions the usage of the randomization technique for privacy preserving in data mining techniques, they show that randomization does not help completely in preserving privacy. They show that original data could be retireved from the randomized dataset. They also provide explicitly the assumptions made in preserving privacy in exisiting systems and provide ways by which a general framework could be provided for preserving privacy better.

III. PROPOSED TECHNIQUES

In this section we are presenting new t data confidentiality technique, secure back propagation and privacy preserving ELM.

A. Data Confidentiality Technique

This technique is based on the problem to decide whether a number is an nth residue modulo n2. This problem is believed to be computationally hard in the cryptography community, and is related to the hardness to factorize n, if n is the product of two large primes.

The notation we use is the classic one, with n = p*q indicating the product of two large primes, Zn the set of the integer numbers modulo n, and Z*.n the set of invertible elements modulo n, that is, all the integer numbers that are relatively prime with n. As usual, the cardinality of the latter set is indicated by |Z* n| and it is equal to the Euler's totient function φ (n).

Key Generation:

a) Choose two large prime numbers p and q randomly and independent of each other such that
   a. gcd (p * q, (p-1) (q-1)) = 1.
b) Compute n=p*q and λ= lcm (p-1, q-1).
c) Select random integer g where $g \in \mathbb{Z}_{n^2}^*$
d) Compute $\mu = (L(g^{\lambda} \mod n^2))^{-1} \mod n$, where function L is defined as $L(u) = \frac{u-1}{n}$.
e) The public (encryption) key is (n,g).
f) The private (decryption) key is (λ , μ).

Encryption

Let $m < n$ be the plaintext and $r < n$ a random value. The encryption c of m is c = Encryption(m) = $g^m r^n \mod n^2$.

Decryption

Let $c < n2$ be the cipher text. The plaintext m hidden in c is m = Decryption(c) = $L(c^{\lambda} \mod n2)/L(g\lambda \mod n2) \mod n$, where $L(x) = (x - 1)/n$.

B. Secure Back propagation Algorithm

In this work, to make the proposed algorithm more secure, sum is to be computed instead of simply using a random number as in traditional crypto systems. For that purpose, homomorphic encryption techniques presented in [3, 19] were used. While applying homomorphic encryption, at the end of each round of training, all the parties hold the encrypted shares of the weights and not the exact weights; this guarantees more security and privacy against the intrusion by the other party.

In this training algorithm, the error function which is used to calculate whether the output is desired or not is given by:

$$e = \frac{1}{2} \sum_i (t_i - o_i)^2$$

where i varies from 1 to n (number of outputs). If the calculated error is above a pre-specified threshold, the training is to be continued by back propagating the error into the network and adjusting the weights accordingly.

Initially, during training, random weights are assigned. Assume training sample of party P1 holds attributes x11 ; x21 ; . . .; xn1 and party P2 holds attributes x12 ; x22 ; . . .; xn2. For each input node i of the neural network, party P1 holds xi1 and party P2 holds xi2 such that xi1 + xi2 = xi. The target value t(x) is known to both the parties. The aim of the algorithm is to train the network, so as to modify the weights.

While using this protocol, the product of the private inputs of the parties, xis, is converted to the summation of private outputs, yis:

$$\prod_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

Following steps explains training process for one sample. The procedure is same for all the remaining samples.

For each hidden layer node hj,

a) Parties P1 and P2 obtain intermediate outputs by using their private input values and weights.

b) Parties P1 and P2 jointly compute the sigmoid function for each hidden layer node hj, obtaining the random output shares.

For each output layer node oi,

a) Parties P1 and P2 securely compute the error and modified weights.

b) The computed weights are propagated backward through the network to adjust weights as in traditional back propagation.

C. Privacy preserving ELM

Assume that data records are ordered such that all the records owned by party Pi, come before the records owned by party Pi+1. Following are the steps of proposed ELM algorithm:

Compute each item of the hidden layer output matrix H.Apply singular value decomposition (SVD) [2] to compute H†.

Calculate the output layer weight vector α by multiplying H† and T

The output weight vector α is determined by using H† which is computed using SVD of hidden layer output matrix H. Secure multi-party SMM algorithms is used to extend the computation of the SVD to the multi-party ELM environment. Secure Scalar product is the main secure building block used to develop the privacy-preserving SVD protocol s which is applied to secure matrix multiplication. This could also be used in the multi-party case, where each party has a subset of rows from the whole matrix in the horizontally partitioned case, or each party has the values of a subset of columns in the vertically partitioned case. Secure multiplication could be used to extend the secure scalar product to more than two vectors.

IV. METHODS AND MATERIALS

The results of the present work are tested on benchmark data sets provided by UCI machine learning repository. Both of the proposed algorithms are implemented by using Matlab 7.6 and JDK 1.6 running on a dual core processor. Following figures show snapshots of plain and encrypted data used in our work.

	A	B	C	D	E	F	G	H	I	J
1	5	1	1	1	2	1	3	1	1	1
2	5	4	4	5	7	10	3	2	1	1
3	3	1	1	1	2	2	3	1	1	1
4	6	8	8	1	3	4	3	7	1	1
5	4	1	1	3	2	1	3	1	1	1
6	8	10	10	8	7	10	9	7	1	2
7	1	1	1	1	2	10	3	1	1	1
8	2	1	2	1	2	1	3	1	1	1
9	2	1	1	1	2	1	1	1	5	1
10	4	2	1	1	2	1	2	1	1	1
11	1	1	1	1	1	1	3	1	1	1
12	2	1	1	1	2	1	2	1	1	1
13	5	3	3	3	2	3	4	4	1	2
14	1	1	1	1	2	3	3	1	1	1
15	8	7	5	10	7	9	5	5	4	2
16	7	4	6	4	6	1	4	3	1	2
17	4	1	1	1	2	1	2	1	1	1
18	4	1	1	1	2	1	3	1	1	1

Fig. 1. Plain input data

Encrypted data:

```
 6407   14584   17543    4596   19622    7383   17632    1483   16834
17852    6522    7914   14893   18944    5301    8045    5849   20145
 1021   19854    4519   13626    7642    2555   18188   13443    2082
 5794   16165    6889   12735   18938   14992    2434    8990   11706
11547   14401    1511   11267    1427   16440   16509   14787   16433
 3763   12202   12915    1830   15789   16602    2368    2422   11140
10410   11588   11553    3655   11586   13892   12438     355    4038
 6735   19755   12357   20352    8446   14660   16487   19846   14821
 3566     590    6076   14986    2154   19537   14414    4725   17172
14401   17651    8410    3780    2271   17012   11989   10960    4022
10926   19220    5314   13133   11767    3000   18958    2304   13990
 5481   12395   12803   10676   11001   10299   13777    3684   19282
16376   17497   12981   11255   16191    8933     985   17766    8667
 2942   16395   10658    6957    7134    3942   15412    9436    8823
13826    3363     329   19518   17734    4274    5294     707    5741
15845   11788    8686   14692    7649    8035    5552   13397   12589
17707   18079   10418    4076   13272    4907   12750    5669    9511
 4561    7335   13707   17073   11523    5503   17255    2701    4325
13968   18409   11785    5062   15581   18367   13209   12175    7242
 6308    6771    4132    4231    9784   12159    5860    1230   16350
18762    5332   15383   14452   19468   11370   13908    1158   16503
  163   10903    3763    5987     853   19414   14081     862   17872
13733    1657   15942    5506   11356    2459   19460    2887   19493
 2902   18698    4938   12366   13419    4665   14325    7578   12124
 9155    5014   19380   15900   20395   17130    3539    6752   11290
20186   15150   15087    5530    6866    4701   19986   10006    1018

the encrypted data
```

Fig. 2. Encrypted output data

```
enter no of units in 1st hidden layer
5
Warning: Integer operands are required for colon operator when used as index
> In privacy at 39
the current epoch is 1
the current epoch is 2
the current epoch is 3
the current epoch is 4
the current epoch is 5
the current epoch is 6
the current epoch is 7
the current epoch is 8
the current epoch is 9
the current epoch is 10
TRAINING IS COMPLETED

testing_rate =

    0.7725
```

Fig. 3. Snapshot of the Sample Output

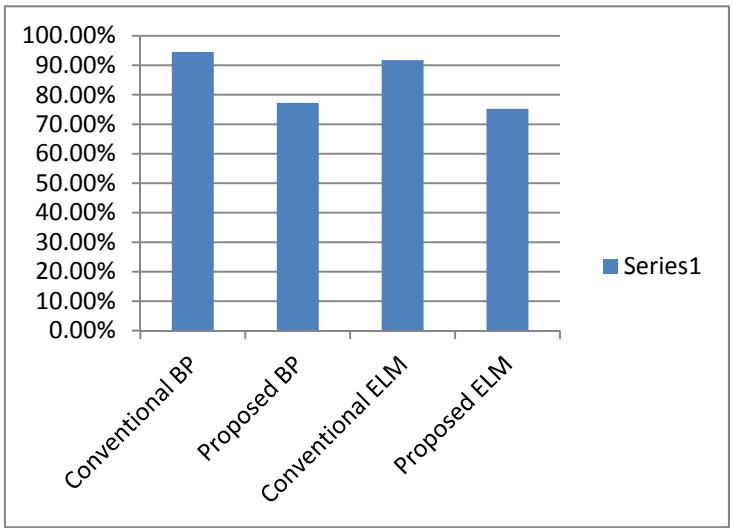

Fig. 4. Bar Chart of the Results

Following table shows the classifier accuracies obtained. Classifier accuracies are computed by using 10-fold Cross Validation method. Experimental results showed that the inclusion of cryptographic operations caused slight decrease in accuracy. Since the accuracy loss is within the acceptable range and confidentiality is of primary concern, we can conclude that proposed approaches are quite effective in learning real world datasets.

TABLE I. CLASSIFIER ACCURACIES

Neural Network Learning Method	Accuracy
Conventional BP	94.534%
Proposed BP	77.254%
Conventional ELM	91.729%
Proposed ELM	75.273%

V. CONCLUSION

In this paper we proposed a protocol for ensuring the confidentiality of the data among multiple parties during neural network training. The proposed protocol is implemented for two popular neural network training methods BP and ELM. The proposed method could be used successfully for neural network training using real world datasets. Results have shown that addition of cryptographic components has decreased classifier accuracy. The proposed methods are to be enhanced to address this issue. Other potential areas for future work include designing secure protocols for recurrent networks and learning models with activation functions other than the threshold and sigmoid functions.

REFERENCES

[1] Yao AC, “Protocols for secure computations,” In: Proceedings of the 23rd annual symposium on foundations of computer science, pp 160–164, 1982

[2] J. Ortega, Matrix Theory, Plenum Press, New York, London, 1987.

[3] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes”, In: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT), Prague, Czech Republic, 1999, pp. 223–238.

[4] R. Agrawal and R. Srikant, “Privacy-Preserving Data Mining,” Proc. ACM SIGMOD Conf. Management of Data, ACM Press, 2000, pp. 439–450; http://doi.acm.org/10.1145/342009.335438.

[5] Standard for privacy of individually identifiable health information. Fed Regist,2001,66(40). http://www.hhs.gov/ocr/privacy/hipaa/understanding/coveredentities/introdution.html

[6] C. Clifton et al., “Tools for Privacy Preserving Distributed Data Mining,” SIGKDD Explorations, vol. 4, no.2,2003,pp.28–34; www.acm.org/sigs/sigkdd/explorations/issue4/contents.htm.

[7] A. Evfimievski, J. E. Gehrke, and R. Srikant. Limiting Privacy Breaches in Privacy Preserving Data Mining. In Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2003). San Diego, CA. June 2003.

[8] H. Kargupta, S. Datta, Q. Wang,and K. Sivakumar. On the Privacy Preserving Properties of Random Data Perturbation Techniques. In Proceedings of the Third IEEE International Conference on Data Mining (ICDM'03). Melbourne, Florida, USA. November 2003. pp. 99-106.

[9] G.B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” in Proc. IJCNN, Budapest, Hungary, Jul. 25–29, 2004, vol. 2, pp. 985–990.

[10] G.B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” Neurocomputing, vol. 70, no. 1–3, pp. 489–501, Dec. 2006.

[11] G.B. Huang, L. Chen, and C.-K. Siew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” IEEE Trans. Neural Netw., vol. 17, no. 4, pp. 879–892, Jul. 2006.

[12] G.B. Huang and L. Chen, “Convex incremental extreme learning machine,” Neurocomputing, vol. 70, no. 16–18, pp. 3056–3062, Oct. 2007.

[13] G.B. Huang and L. Chen, “Enhanced random search based incremental extreme learning machine,” Neurocomputing, vol. 71, no. 16–18, pp. 3460–3468, Oct. 2008.

[14] Gahi Y., Guennoun M. and El-Khatib K., “A secure database system using homomorphic encryption schemes, The third international conference on advances in databases, knowledge and data applications”, 2011.

[15] Brakerski Z. and Vaikuntanathan V, “Fully homomorphic encryption from ring-LWE and security for key dependent messages”, CRYPTO 2011.

[16] Diana-Ștefania MAIMUȚ, Alecsandru PĂTRAȘCU, Emil SIMION “Homomorphic Encryption Schemes and Applications for a Secure Digital World”,Journal of Mobile, Embedded and Distributed Systems, vol. IV, no. 4, 2012 ISSN 2067 – 4074

[17] Chen T, Zhong S, “Privacy preserving back-propagation neural network learning,” IEEE Trans Neural Networks, Vol. 20, No. 10, pp.1554–1564

[18] HIPPA, National Standards to Protect the Privacy of Personal Health Information. http://www.hhs.gov/ocr/hipaa/finalreg.html

[19] S. Samet and A. Miri, “Privacy-Preserving Back-Propagation and Extreme Learning Machine Algorithms,” Journal of Data and Knowledge Engineering, vol. 79-80, pp. 40–61, 2012.