Privacy-Preserving Linear Regression for Brain-Computer Interface Applications

Anisha Agarwal[1], Rafael Dowsley[2], Nicholas D. McKinney[1], Dongrui Wu[3],
Chin-Teng Lin[4], Martine De Cock[1,5], Anderson Nascimento[1]
[1]School of Engineering and Technology, University of Washington, Tacoma, USA
Email: {anisha3,mckinnnd,mdecock,andclay}@uw.edu
[2] Dept. of Computer Science, Aarhus University, Aarhus, Denmark, Email: rafael@cs.au.dk
[3] School of Automation, Huazhong University of Science and Technology, Wuhan, China, Email: drwu09@gmail.com
[4] School of Software, University of Technology, Sydney, Australia, Email: Chin-Teng.Lin@uts.edu.au
[5] Dept. of Applied Math., Comp. Sc. and Statistics, Ghent University, Email: martine.decock@ugent.be

I. Introduction

Many machine learning (ML) applications rely on large amounts of personal data for training and inference. Among the most intimate exploited data sources is electroencephalogram (EEG) data. The emergence of consumer-grade, low-cost brain-computer interfaces (BCIs) and corresponding software development kits[1] is bringing the use of BCI within reach of application developers. The access that BCI applications have to neural signals rightly raises privacy concerns. Application developers can easily gain knowledge beyond the professed scope from unprotected EEG signals, including passwords, ATM PINs, and other personal data [1]. The challenge we address is how to engage in meaningful ML with EEG data while protecting the privacy of users.

To this end, we use Secure Multiparty Computation (SMC) [2], a form of cryptology, to perform linear regression (LR) over EEG signals from many users in a fully privacy-preserving fashion, i.e. such that each individual's EEG signals are not revealed to anyone else. As a use case, we show that our secure framework can estimate the drowsiness of drivers from their EEG signals as would be possible in the unencrypted case, at a very reasonable computational cost. Our solution is the first application of commodity-based SMC to EEG data, as well as the largest documented experiment of secret-sharing based SMC in general, namely with 15 parties involved in all the computations.

II. Methods

The use of SMC for privacy-preserving training of ML models is gaining popularity (see, e.g., [3], [4] and references therein). Existing frameworks for SMC, including Sharemind [5], FairPlay [6], and Chameleon [7], are all written for 2 or 3 parties to jointly perform the computations. We developed a framework for SMC based LR in which any number of parties can participate. To the best of our knowledge, ours is the first documented application of secret-sharing based SMC for ML with computations done by more than 3 parties.

[1]E.g. https://www.emotiv.com/, http://neurosky.com/, https://myndplay.com/

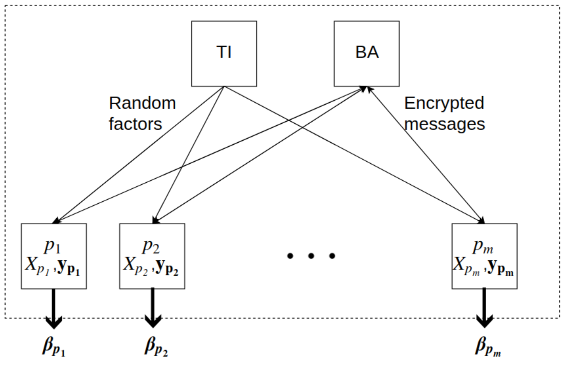
TI
BA
Random factors
Encrypted messages
p_1 $X_{p_1}, \mathbf{y}_{\mathbf{p_1}}$
p_2 $X_{p_2}, \mathbf{y}_{\mathbf{p_2}}$
p_m $X_{p_m}, \mathbf{y}_{\mathbf{p_m}}$
$\boldsymbol{\beta}_{p_1}$
$\boldsymbol{\beta}_{p_2}$
$\boldsymbol{\beta}_{p_m}$

Fig. 1. Training phase of privacy-preserving target-independent LR.

We work in the commodity based model [8], and set up a Trusted Initializer (TI) that pre-distributes correlated random numbers during an initialization phase(offline) to the parties participating in the protocol. To limit the number of communication channels (sockets) among all parties, we use a Broadcast Agent(BA). We perform secure computations using additively secret sharings over a finite field $\mathbb{F}_q$. A value $x \in \mathbb{F}_q$ is secret shared between parties $p_1, \ldots, p_m$ by picking $x_{p_1}, \ldots, x_{p_m} \in \mathbb{F}_q$ uniformly at random subject to the constraint that $x = x_{p_1} + \ldots + x_{p_m} \mod q$, and then revealing x_i to p_i.

We designed cyrptographic protocols for LR with data from many parties in two scenarios. In the 1st scenario, called **target-independent LR** (Figure 1), a set of m *source parties* $p_1, \ldots, p_m$ work together to train a LR model in a distributed fashion (many-party SMC). None of the parties can see the data from the other parties in an unencrypted way at any point. At the end of the protocol, all parties hold encrypted shares of the coefficient vector $\boldsymbol{\beta}$ of the trained model, and a *target party* obtains a prediction for its data by engaging in a protocol with all of the source parties (many-party SMC).

We assume that the training data can be thought of as a $n \times k$ matrix X and a vector $\mathbf{y}$ of length n, where each row of X corresponds to the input feature values of a particular training example, and the corresponding entry in $\mathbf{y}$ is the label of that example. We assume that, instead of residing in one place, X

and $\mathbf{y}$ are horizontally distributed across m source parties that each hold a non-overlapping subset of the training examples. Taking advantage of the fact that the data is horizontally partitioned in this way, allows us to propose a protocol that is more efficient in this situation than the more general protocol from De Cock et al. [4]. We use the protocol for covariance matrix inversion Π_{MatInv} that is based on a generalization of the Newton-Raphson division method to matrices [4], secure matrix multiplication π_{DMM} and a slightly modified version Π_{Trunc} of the truncation protocol of Catrina and Saxena [9] to compute the shares $\boldsymbol{\beta}_{p_i}$ of the estimated regression coefficient vector: $\boldsymbol{\beta} = \boldsymbol{\beta}_{p_1} + \boldsymbol{\beta}_{p_2} + \ldots \boldsymbol{\beta}_{p_m} \mod q$.

For a new prediction, the target party sends shares of its input to all m source parties, who engage in a SMC protocol among themselves, using the secure matrix multiplication π_{DMM} and secure truncation Π_{Trunc} protocols to compute shares of the predicted result $\hat{y}_{p_i}$. Finally, the source parties send the computed shares to the target, who combines them to learn the prediction: $\hat{y} = \hat{y}_{p_1} + \hat{y}_{p_2} + \ldots + \hat{y}_{p_m} \mod q$.

In the 2nd scenario, called **target-calibrated LR**, the target party has calibration data that can be leveraged to train a personalized model. The target party engages in a separate protocol with each of the source parties (2-party SMC) to train LR models, namely as many models as there are source parties. Each model is trained on data from one source party, as well as on some of the calibration data from the target party. As before, any individual's data is not disclosed to anyone else. Furthermore, at the end of the training protocol, no single party knows any of the trained models. Instead, knowledge about the models is split into encrypted shares that are "owned" in a distributed fashion by the parties (including the target). Finally, the target party obtains a prediction for its data, as an average of the predictions by all trained models, through engaging in a protocol with all the source parties (many-party SMC). The participation of the target party in the prediction process removes the need for the target party to disclose its own data to any of the other parties.

Source code for all protocols in the SMC framework Lynx is available online.[2]

III. Results

We evaluated our protocols using the same data and scenarios for detecting drowsiness based on EEG signals from 15 drivers as Wu et al. [10]. We use data from 15 subjects who participated in a 60-90 minutes sustained-attention driving experiment in a real vehicle mounted on a motion platform immersed in a 360-degree virtual-reality scene. For each of the 15 drivers we have a dataset consisting of 1200 rows, in chronological order, each consisting of 30 numerical input values and a response value (the level of drowsiness). Experiments were run on a machine with 36 vCPUs, 72.0 GiB memory.

The training process in the 1st scenario (Fig. 2) scales well with an increasing number of parties (drivers) in the

[2]https://bitbucket.org/uwtppml/lynx

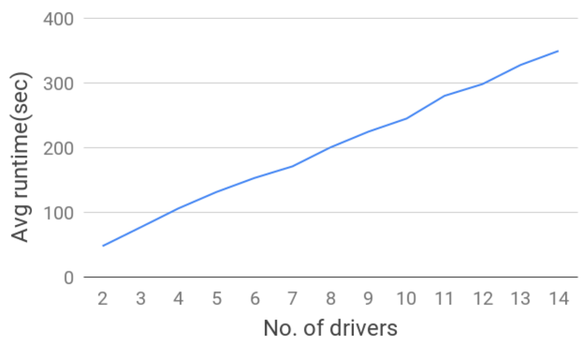

Fig. 2. Runtime for many-party SMC training of LR model

computation, taking between 48.51 sec ($m = 2$) up to 350.39 sec ($m = 14$). This is in contrast with 0.10 sec to 0.43 sec in the clear. The runtime grows with the number of drivers because there is more training data available for processing, and more parties that need to communicate. However, we see that the impact of each new party participating in the computation is moderate. Making a prediction for a target driver takes on average 2.82 sec ($m = 2$) and 10.80 sec ($m = 14$) using our framework.

In the 2nd scenario, the training comprises of 14 rounds of 2-party SMC. It takes 51.73 sec on average for the training, since all 14 models can be run in parallel. The prediction from the trained models involve all 14 parties as well as the target, taking 2.89 sec on average. This is in contrast to 0.06 sec (training) and 0.008 sec (inference) when done in the clear, i.e. without encryption.

There is no increase in RMSE whatsoever in the process, i.e. our framework produces the exact same results as in the clear, adding privacy for the dataset holders.

References

[1] I. Martinovic, D. Davies, M. Frank, D. Perito, T. Ros, and D. Song, "On the feasibility of side-channel attacks with brain-computer interfaces," in *Proc. of the 21st USENIX Security Symposium*, 2012.
[2] R. Cramer, I. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.
[3] C. C. Aggarwal and S. Y. Philip, "A general survey of privacy-preserving data mining models and algorithms," in *Privacy-Preserving Data Mining*. Springer, 2008, pp. 11–52.
[4] M. De Cock, R. Dowsley, A. C. A. Nascimento, and S. C. Newman, "Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data," in *AISec*, 2015, pp. 3–14.
[5] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *ESORICS*, 2008, pp. 192–206.
[6] A. Ben-David, N. Nisan, and B. Pinkas, "FairplayMP: A system for secure multi-party computation," in *CCS*, 2008, pp. 257–266.
[7] M. Sadegh Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A Hybrid Secure Computation Framework for Machine Learning Applications," *ArXiv e-prints*, 2018.
[8] D. Beaver, "One-time tables for two-party computation," in *Computing and Combinatorics*. Springer, 1998, pp. 361–370.
[9] O. Catrina and A. Saxena, "Secure computation with fixed-point numbers," in *International Conference on Financial Cryptography and Data Security*. Springer, 2010, pp. 35–50.
[10] D. Wu, V. J. Lawhern, S. Gordon, B. J. Lance, and C.-T. Lin, "Driver drowsiness estimation from EEG signals using online weighted adaptation regularization for regression (OwARR)," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1522–1535, 2017.