SUPPLEMENT for: PrivLogit: Fast Privacy-preserving Logistic Regression by Tailoring Numerical Optimization

Wei Xie
EECS, Vanderbilt University
Nashville, TN, USA
wei.xie@vanderbilt.edu

Yang Wang
SIE, University of Virginia
Charlottesville, Virginia, USA
yw3xs@virginia.edu

Steven M. Boker
University of Virginia
Charlottesville, Virginia, USA
boker@virginia.edu

Donald E. Brown
SIE, University of Virginia
Charlottesville, Virginia, USA
deb@virginia.edu

APPENDIX

A. DETAILED SECURITY ANALYSIS

A.1 What information is leaked?

Our protocols reveal: 1) the regression coefficients or model outputs (Steps 13 and 15 in Algorithm 1; Steps 14 and 16 in Algorithm 3), and 2) the total iteration numbers (Step 2 in Algorithm 1; Step 3 in Algorithm 3). Revealing model output to local Nodes (the former) is mainly for performance and accuracy considerations (similarly found in [4, 1]), because otherwise the $\boldsymbol{\beta}$ coefficients need to be encrypted and then local (secure) computation of Hessian and gradient becomes directly *infeasible* due to the non-linearity of logistic function (Equation 2). Some security literature has proposed approximations to the logistic function to make it securely computable, but their accuracy loss is noticeable and thus not widely adopted. Since function approximation is a separate topic and conflicts with our accuracy guarantees, we do not cover them in our work.

In addition, intermediate model outputs (revealed to local Nodes) share the same privacy properties as the final model output of the whole protocol, because they are all simply regression coefficients. By definition, cryptographic protocols do not protect final output, so this practice does not violate security. The only possible way to breach security in PrivLogit protocols is for local Nodes to accumulate all regression coefficients across all iterations and form a linear equation system to solve for individual record-level input values [7, 2]. However, since the number of iterations is quite small (at most linear in dimensionality p) and the (privacy-sensitive) data – the attack target – is huge in size ($n \times p$, where n is extremely large), the system is severely undetermined and such attacks are infeasible mathematically.

Regarding the second leak of iterations number, this is a common problem for any secure protocols safeguarding numerical optimization (the foundation of machine learning), such as for Newton method [5], gradient descent [3], and matrix inversion (decomposition) [6]. This is due to the iterative nature of numerical optimization and that the stopping/convergence criteria is adaptive and dependent on previous iteration result (thus at least comparison result needs to be revealed). An easy way to overcome the limitation is to enforce a maximum number of iterations beforehand (larger than is actually necessary) and to always iterate to the maximum iterations. However, this comes at a significant cost in unnecessary computations/iterations and has limited practical relevance. We thus do not address this issue and directly follow existing literature.

B. SECURE CHOLESKY DECOMPOSITION

Secure Cholesky decomposition (Algorithm 1; also used by [6]) is used in our protocol to help "invert" the (negated) Hessian, which is the main computation in PrivLogit (and the bottleneck in Newton method). We denote the input matrix to be $\mathbf{B} \in \mathbb{R}^{p \times p}$, and each elements of it as $L_{i,j}$, where i, j index the row and column positions, respectively.

C. YAO'S GARBLED CIRCUIT.

Yao's garbled circuit [9] is a popular secure two-party computation technique for securely supporting evaluation of joint function $f(x_1, x_2)$ from two sources of secret inputs, i.e., x_1 (held by Party P_1) and x_2 (held by Party P_2). In brief, function $f(x_1, x_2)$ is first represented in a binary circuit form. Then Party P_1 translates ("garbles") the function circuit into a secure version ("garbled" versions of circuit and computation table based on its private input x_1). The resulting garbled circuit and computation table are later sent to the opponent P_2, who initiates a 1-out-of-2 oblivious transfer (OT) protocol (assisted by P_1) to obliviously compute garbled values corresponding to his input x_2 and outputs the result to prescribed parties. The whole protocol reveals nothing to any parties other than the final result.

Algorithm 1 Secure Cholesky decomposition of matrix $\mathbf{B}$.

Input: Encryption $Enc(\mathbf{B})$, where $\mathbf{B} = (L_{i,j}) \in R^{p \times p}$
Output: Encrypted triangular matrix $Enc(\mathbf{L})$, such that $\mathbf{L}\mathbf{L}^T = \mathbf{B}$

[At Center] :

for j = 1 **to** p **do**
  for k = 1 **to** j-1 **do**
    for i = j **to** p **do**
      $Enc(L_{i,j}) = Enc(L_{i,j}) \ominus Enc(L_{i,k}\ L_{j,k})$
    end for
  end for
  $Enc(L_{i,j}) = E_{sqrt}(L_{j,j})$
  for k = j+1 **to** p **do**
    $Enc(L_{k,j}) = Enc(L_{k,j}) \oslash Enc(L_{j,j})$
  end for
end for
return $Enc(\mathbf{L}) = Enc(updated\ \mathbf{B})$

D. PAILLIER CRYPTOSYSTEM.

The Paillier cryptosystem [8] is a public-key cryptographic scheme that is additively homomorphic. In brief, a message m ("secret") can be encrypted by:

$$Enc(m, r) = g^m r^n \mod n^2 , \quad (1)$$

where $n = pq$ corresponds to an RSA modulus, g is a public parameter and r is a randomization. Here the public (i.e., encryption) key would be (n, g), and the private (i.e., decryption) key would be (p, q, λ) (where λ equals the least common multiple of $p - 1$ and $q - 1$). In addition to its strong security guarantees, Paillier cryptosystem also possesses a few useful partially homomorphic properties. For instance, secure addition can be computed as:

$$Enc(m_1 + m_2, r_1 r_2) = Enc(m_1, r_1) \times Enc(m_2, r_2) \mod n^2$$

and multiplication-by-constant follows (for constant k):

$$Enc(k * m) = Enc(m, r)^k \mod n^2$$

E. ADDITIONAL AUTHORS

F. REFERENCES

[1] C. C. Aggarwal and S. Y. Philip. *A general survey of privacy-preserving data mining models and algorithms.* Springer, 2008.

[2] K. El Emam, S. Samet, L. Arbuckle, R. Tamblyn, C. Earle, and M. Kantarcioglu. A secure distributed logistic regression protocol for the detection of rare adverse drug events. *Journal of the American Medical Informatics Association*, 20(3):453–461, 2013.

[3] S. Han, W. K. Ng, L. Wan, and V. C. Lee. Privacy-preserving gradient-descent methods. *IEEE Transactions on Knowledge and Data Engineering*, 22(6):884–899, 2010.

[4] W. Li, H. Liu, P. Yang, and W. Xie. Supporting regularized logistic regression privately and efficiently. *arXiv preprint arXiv:1510.00095*, 2015.

[5] W. Li, H. Liu, P. Yang, and W. Xie. Supporting regularized logistic regression privately and efficiently. *PloS one*, 11(6):e0156479, 2016.

[6] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 334–348, 2013.

[7] C. M. O'Keefe and J. O. Chipperfield. A summary of attack methods and confidentiality protection measures for fully automated remote analysis systems. *International Statistical Review*, 81(3):426–455, 2013.

[8] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology – EUROCRYPT'99*, pages 223–238. Springer, 1999.

[9] A. C. Yao. Protocols for secure computations. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 160–164. IEEE, 1982.