MP2ML: A Mixed-Protocol Machine Learning Framework for Private Inference (Extended Abstract)*

Fabian Boemer
fabian.boemer@intel.com
Intel AI
San Diego, California, USA

Rosario Cammarota
rosario.cammarota@intel.com
Intel Labs
San Diego, California, USA

Daniel Demmler
demmler@informatik.uni-hamburg.de
University of Hamburg
Hamburg, Germany

Thomas Schneider
schneider@encrypto.cs.tu-darmstadt.de
TU Darmstadt
Darmstadt, Germany

Hossein Yalame
yalame@encrypto.cs.tu-darmstadt.de
TU Darmstadt
Darmstadt, Germany

ABSTRACT

We present an extended abstract of MP2ML, a machine learning framework which integrates Intel nGraph-HE, a homomorphic encryption (HE) framework, and the secure two-party computation framework ABY, to enable data scientists to perform private inference of deep learning (DL) models trained using popular frameworks such as TensorFlow at the push of a button. We benchmark MP2ML on the CryptoNets network with ReLU activations, on which it achieves a throughput of 33.3 images/s and an accuracy of 98.6%. This throughput matches the previous state-of-the-art frameworks.

KEYWORDS

private machine learning, homomorphic encryption, secure multi-party computation.

ACM Reference Format:
Fabian Boemer, Rosario Cammarota, Daniel Demmler, Thomas Schneider, and Hossein Yalame. 2020. MP2ML: A Mixed-Protocol Machine Learning Framework for Private Inference (Extended Abstract). In *2020 Workshop on Privacy-Preserving Machine Learning in Practice (PPMLP'20), November 9, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3411501.3419425

1 INTRODUCTION

Today, there is an inherent contradiction between ML utility and privacy: ML requires data to operate, while privacy necessitates keeping sensitive information private. Modern cryptographic techniques such as homomorphic encryption (HE) [18] and secure multi-party computation (MPC) [8] can help resolve this contradiction. Both techniques have their limitations, though. For this reason, recent research has demonstrated the ability to evaluate neural networks using a combination of HE and MPC [11, 14].

Historically, a major challenge for building privacy-preserving machine learning (PPML) systems has been the absence of software frameworks that support privacy-preserving primitives. To overcome this challenge, Intel introduced an HE-based private ML framework nGraph-HE [3, 4] and Microsoft introduced the MPC-based CRYPTFLOW framework [12] that are compatible with existing DL frameworks.

Our Contributions. In this work, we introduce MP2ML, a hybrid HE-MPC framework for privacy-preserving DL inference. MP2ML extends nGraph-HE [3, 4] using the ABY framework [6] to implement a 2PC version of the ReLU activation function. One of the primary advantages of MP2ML is the integration with DL frameworks, in particular TensorFlow. MP2ML provides the following core contributions:

– A privacy-preserving mixed-protocol DL framework based on a novel combination of nGraph-HE [3, 4] and ABY [6] that supports private inference on direct input from TensorFlow.

– Support for privacy-preserving evaluation of the non-linear ReLU activation function with high accuracy.

– The first DL application using additive secret sharing in combination with the CKKS homomorphic encryption scheme [5].

– An open-source implementation of our framework, available under the permissive Apache license at https://ngra.ph/he.

2 RELATED WORK

Previous work in private DL typically uses either exclusively HE or exclusively MPC. GAZELLE [11] and DELPHI [14] are notable exceptions, using both HE and MPC in a hybrid scheme. [2, Tab.9] shows a comprehensive comparison between MP2ML and previous work. MP2ML provides the *first* hybrid HE-MPC framework that integrates with a DL framework such as TensorFlow.

3 THE MP2ML FRAMEWORK

MP2ML is a hybrid HE-MPC framework integrating ABY [6] and nGraph-HE [3], and is compatible with DL frameworks such as TensorFlow. Our work focuses on the setting in which the client

*Please cite the full version of this paper published at ARES'20 [2].

PPMLP'20, November 9, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8088-1/20/11…$15.00
https://doi.org/10.1145/3411501.3419425

can privately perform inference without disclosing his or her input to the server as well as preserving the privacy of the server's DL model. MP2ML, for the *first* time in MPC-HE world, requires only a single line of code changes to existing TensorFlow code (cf. §5).
Adversary Model. In this work, we use the semi-honest adversary model in which the adversary follows the protocol honestly, but attempts to infer sensitive information [1, 11–16].

3.1 Private ML Workflow

We describe the steps in which a server performs private inference on a client's encrypted data. More details can be found in [2].
Client Input Encryption. First, the client encrypts its input using the CKKS HE scheme [5], as implemented by Microsoft SEAL [19], and sends it to the server. For increased throughput, multiple values are packed using batch-axis plaintext packing.
Linear Layers. The server evaluates linear layers using the HE-based nGraph-HE [3, 4] implementation. Using HE for the linear layers enables the server to hide the model structure/topology from the client, and results in no communication cost, avoiding the overhead of communication-heavy pure MPC frameworks [12, 16].
Non-Linear Layers. MP2ML evaluates non-linear functions using a secure two-party protocol between the client and the server, such that no sensitive information about intermediate values is leaked.
ReLU Evaluation. Our use of ABY enables secure evaluation of arbitrary activation functions. To do so, first, the ciphertext $[\![x]\!]$ is converted to an MPC value. Like Previous work [9, 11]: the server additively blinds $[\![x]\!]$ with a random mask r and sends the masked ciphertext $[\![x+r]\!]$ to the client, who decrypts. Then, both parties evaluate a subtraction circuit in MPC to remove the random mask. MP2ML extends this approach to fixed-point arithmetic. In our private ReLU protocol, the server and the client perform the following steps:

- *Conversion from HE to MPC*
- *MPC circuit evaluation*
- *Conversion from MPC to HE*

To evaluate a complete network, MP2ML computes linear layers using HE, and the above protocol for non-polynomial activations. The encrypted final classification result is sent to the client for decryption.

Our conversion protocol achieves two important tasks: First, it enables the secure computation of non-polynomial activation functions, i.e., without leaking values to the data owner. Second, our protocol refreshes the ciphertexts, resetting the noise and restoring the ciphertext level to the top level L. This refreshment is essential to enabling continued computation without increasing the encryption parameters. Rather than selecting encryption parameters large enough to support the entire network, they must now only be large enough to support the linear layers between non-linear activations.

4 EVALUATION

Evaluation Setup. For the evaluation we use two Intel Xeon® Platinum-8180 2.5 GHz systems with 112 cores and 376 GB RAM, running Ubuntu 18.04. The local area network (LAN) bandwidth is 9.6 Gbit/s, while the latency is 0.165 ms.

We evaluate a deep learning application, the CryptoNets [7] network, to show how our MP2ML framework can be leveraged. Table 1 shows the performance of MP2ML on CryptoNets in comparison with previous works. MP2ML uses encryption parameters $N = 8192, L = 5$, with coefficient moduli $(47, 24, 24, 24, 30)$ bits, scale $s = 2^{24}$, $\lambda = 128$-bit security, and Yao's garbled circuits for the non-linear layers.

Chameleon [17] and SecureNN [20] use a semi-honest third party, which is a different setting than our two-party model. XONN [16] binarizes the network, which results in high accuracy on the MNIST dataset, but will reduce accuracy on larger datasets and models. CryptoNets [7] and CryptoDL [10] use polynomial activations, which will also reduce accuracy on larger datasets and models. GAZELLE [11], whose method is most similar to our work, uses much smaller encryption parameters ($N = 2048, L = 1$), resulting in a significantly faster runtime (cf. [3, Tab.9]), albeit at a reduced 20-bit precision. nGraph-HE [4] uses a client-aided model to compute non-polynomial activations, which leaks intermediate values and potentially the model weights to the client.

5 DISCUSSION AND CONCLUSION

In this paper we presented MP2ML, the first user-friendly mixed-protocol framework for private DL inference. MP2ML is compatible with popular DL frameworks such as TensorFlow. Listing 1 and Listing 2 show the Python3 source code for the server and the client, respectively, to preform inference on a pre-trained TensorFlow model. Only minimal code changes are required compared to native TensorFlow, highlighting the practicality and usability of MP2ML.

```
import tensorflow as tf; import numpy as np
import ngraph_bridge
from mnist_util import *

# Load saved model
tf.import_graph_def(load_pb_file('./model/model.pb'))
# Get input / output tensors
graph = tf.compat.v1.get_default_graph()
x_input = graph.get_tensor_by_name("import/input:0")
y_output = graph.get_tensor_by_name("import/output:0")
# Create configuration to encrypt input
FLAGS, _ = server_argument_parser().parse_known_args()
config = server_config_from_flags(FLAGS, x_input.name)
with tf.compat.v1.Session(config=config) as sess:
    # Evaluate model (random input data is discarded)
    y_output.eval(feed_dict={x_input: np.random.rand
     (10000, 28, 28, 1)})
```

Listing 1: Python3 source code for a server to execute a pre-trained TensorFlow model in MP2ML. A server configuration specifies the encryption parameters and which tensors to obtain from the client. The server passes random dummy values as input. The client provides the encrypted input.

ACKNOWLEDGMENTS

We thank Casimir Wierzynski, Amir Khosrowshahi, and Naveen Rao for their unconditional support. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 850990 PSOTI). It was co-funded by the Deutsche

Table 1: MNIST inference performance comparisons. The network topologies are not identical across previous work, resulting in variations in accuracy.

Framework	Limitation	Accuracy (%)	Latency (s)	Throughput (images/s)
CryptoNets [7]	polynomial activation	98.95	250.00	16.4
Chameleon [17]	3-party	99.00	2.24	1.0
CryptoDL [10]	polynomial activation	99.52	320.00	45.5
GAZELLE [11]	hand-optimized	98.95	0.03	33.3
SecureNN [20]	3-party	99.00	0.08	49.2
XONN [16]	binarized network	98.64	0.16	6.2
nGraph-HE2 [3]	reveals intermediate values	98.62	0.69	2,959.0
CrypTFlow [12]	leaks model architecture	99.31	0.03	33.3
MP2ML (This work)	—	98.60	6.79	33.3

```python
import pyhe_client
from mnist_util import *

# Parse command-line arguments
FLAGS, _ = client_argument_parser().parse_known_args()
# Load data
(x_test, y_test) = load_mnist_test_data(
        FLAGS.start_batch, FLAGS.batch_size)
# Perform inference
client = pyhe_client.HESealClient(
    FLAGS.hostname, FLAGS.port, FLAGS.batch_size,
    { FLAGS.tensor_name:
        (FLAGS.encrypt_data_str, x_test.flatten('C')) })
results = client.get_results()
```

Listing 2: Python3 source code for a client to execute a pre-trained TensorFlow model in MP2ML.

Forschungsgemeinschaft (DFG) — SFB 1119 CROSSING/236615297 and GRK 2050 Privacy & Trust/251805230, and by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and the Arts within ATHENE.

AVAILABILITY

The open source code of MP2ML is freely available under the permissive Apache license at https://ngra.ph/he.

REFERENCES

[1] Nitin Agrawal, Ali Shahin Shamsabadi, Matt J Kusner, and Adrià Gascón. 2019. QUOTIENT: Two-Party Secure Neural Network Training and Prediction. In *CCS'19*.

[2] Fabian Boemer, Rosario Cammarota, Daniel Demmler, Thomas Schneider, and Hossein Yalame. 2020. MP2ML: A Mixed-Protocol Machine Learning Framework for Private Inference. In *ARES'20*.

[3] Fabian Boemer, Anamaria Costache, Rosario Cammarota, and Casimir Wierzynski. 2019. nGraph-HE2: A High-Throughput Framework for Neural Network Inference on Encrypted Data. In *WAHC'19*.

[4] Fabian Boemer, Yixing Lao, Rosario Cammarota, and Casimir Wierzynski. 2019. nGraph-HE: a graph compiler for deep learning on homomorphically encrypted data. In *ACM International Conference on Computing Frontiers*.

[5] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2017. Homomorphic Encryption for Arithmetic of Approximate Numbers. In *ASIACRYPT'17*.

[6] Daniel Demmler, Thomas Schneider, and Michael Zohner. 2015. ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In *NDSS'15*.

[7] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *ICML'16*.

[8] Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to play any mental game. In *STOC'87*.

[9] Wilko Henecka, Stefan Kögl, Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. 2010. TASTY: Tool for Automating Secure Two-party Computations. In *CCS'10*.

[10] Ehsan Hesamifard, Hassan Takabi, Mehdi Ghasemi, and Rebecca N. Wright. 2018. Privacy-preserving Machine Learning as a Service. *PETS'18*.

[11] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. 2018. *GAZELLE*: A Low Latency Framework for Secure Neural Network Inference. In *USENIX Security'18*.

[12] Nishant Kumar, Mayank Rathee, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. 2020. CrypTFlow: Secure TensorFlow Inference. In *S&P'20*.

[13] Jian Liu, Mika Juuti, Yao Lu, and Nadarajah Asokan. 2017. Oblivious neural network predictions via MiniONN transformations. In *CCS'17*.

[14] Pratyush Mishra, Ryan Lehmkuhl, Akshayaram Srinivasan, Wenting Zheng, and Raluca Ada Popa. 2020. DELPHI: A Cryptographic Inference Service for Neural Networks. In *USENIX Security*.

[15] Payman Mohassel and Yupeng Zhang. 2017. SecureML: A system for scalable privacy-preserving machine learning. In *S&P'17*.

[16] M Sadegh Riazi, Mohammad Samragh, Hao Chen, Kim Laine, Kristin E Lauter, and Farinaz Koushanfar. 2019. XONN: XNOR-based Oblivious Deep Neural Network Inference. In *USENIX Security'19*.

[17] M Sadegh Riazi, Christian Weinert, Oleksandr Tkachenko, Ebrahim M Songhori, Thomas Schneider, and Farinaz Koushanfar. 2018. Chameleon: A hybrid secure computation framework for machine learning applications. In *ASIACCS'18*.

[18] Ronald L. Rivest, Len Adleman, and Michael L. Dertouzos. 1978. On Data Banks and Privacy Homomorphisms. *Foundations of Secure Computation, Academia Press*.

[19] SEAL 2019. Microsoft SEAL (release 3.4). https://github.com/Microsoft/SEAL. Microsoft Research, Redmond, WA.

[20] Sameer Wagh, Divya Gupta, and Nishanth Chandran. 2019. SecureNN: 3-Party Secure Computation for Neural Network Training. *PETS'19*.