A Practical Data Privacy-preserving Machine Learning System for Cloud Computing Services

Mengxiao Lin, Jiemin Wu, Xiaorui Wang, Chuyuan Qu
Computer Science @ UC Davis

Abstract

Recently, the computation-expensive deep learning becomes more and more popular, which results in an emerge of training neural networks on cloud computing platforms. However, while the data plays a more and more important role in the model quality, how to protect data while maintaining the training speed also becomes more and more challenge. In this project, we propose a new system to protect user's data with minimal trust of the computation platforms, which makes it possible to protect data privacy from malicious cloud computing providers or attackers without sacrifice of the speed.

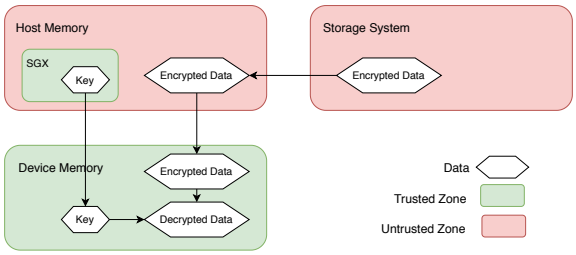

Figure 1: The figure of the trusted and untrusted zones in our system. Both the host memory and the storage system are not trusted by the proposed system. The arrows represent how the data is transferred in the system.

1 Introduction

Last decade witnesses the maturing of cloud computing and deep neural networks. The result is that more and more academic and industrial sectors start to train and deploy their neural networks and other machine learning models on public-available cloud computing services like AWS, Azure and Google Cloud. Machine Learning as a Service (MLaaS) is also proposed for this special use case, and some platforms even provide totally automatic machine learning service [1] based on AutoML technologies. However, in most cases mentioned above, the users of these services have to upload their data to a storage system provided by these cloud computing services, which means that they have to trust the cloud computing platforms.

The data privacy problem in machine learning has been noticed by many previous works. Researchers from different communities propose solutions from different levels to protect the data if the platform itself is untrusted [11, 9, 13]. While architecture research showed the possibility in building enclaves in virtual machines and GPUs [13], modern machine learning frameworks cannot work on these secure systems without the help from the manufacturers of computing acceleration devices. On the other hand, none of the research on machine learning algorithms provides protection against malicious platforms.

As it is found in [26], the most widely used computing acceleration device GPU itself is isolated from the general computing systems. This discovery guides us to a practical method in implementing data privacy-preserving machine learning systems: by trusting the isolated GPUs and open-source machine learning frameworks, we can transfer the encrypted data from storage systems into GPU memory and decrypt them with the key saved in a host memory enclave which is accessible to GPUs. The attackers cannot steal the data unless they bypass the GPU context protection mechanism. The idea is illustrated in Figure 1. The decrypted data will only appear in the device memory for computation. Even the attackers obtain the permission to access the storage system and host memory, our proposed system can keep the security of the data.

To sum up, we expect to build a really practical data privacy-preserving machine learning system that including both encryption tools and decryption components for deep learning frameworks. The proposed system should be able to work with at least one widely used deep learning framework (i.e. TensorFlow [2], PyTorch [22] and MXNet [5]). The performance restriction is that the difference in running speed should not be considerable and the obtained models should achieve similar accuracy on development and test datasets.

2 Background

In this section, we briefly introduce some background knowledge about the problem we are going to solve. Modern machine learning algorithms extremely rely on data,

which makes data more valuable. As machine learning matures, both the development frameworks and cloud computing platforms to run the training and inference codes become popular. They are the sources of the problem.

2.1 Deep Learning Frameworks

Since the training of the deep neural networks is a dense computing task, many frameworks[6, 3, 2, 22, 5] are proposed to simplify the developments of deep learning in order to fully utilize the computing power (especially GPUs) of computers. The most prominent pioneer among them is Theano [3]. Theano is based on the idea of symbolic computing and it compiles a computation graph into the binary code in order to speed up the computing. This idea is widely used by the followers in this area in order to achieve better performance in heterogeneous and distributed computing architectures [2, 5].

Our work should be built on these frameworks since using them is the best practice in developing deep learning applications. Modern frameworks [2, 22, 5] provide easy methods to extend, so we will embed our on-GPU decryption tools into these frameworks.

2.2 Cloud Computing and Security

The cloud computing model provides users with on-demand access to a shared pool of configurable computing resources [18]. Generally, the resources are split and distributed through virtual machines. Virtual machine monitors promise the separation of different virtual machines. However, the security problems raised in cloud computing are not simply eliminated by this separation. Previous works examine the security issues in storage system [27], networks [28] and other areas [30]. The threat models in most of these works trust the service providers and try to protect almost all workloads on clouds. Our work makes a more realistic assumption that the service provider does not worth the trust and focuses on the specific problem of protecting data in machine learning tasks.

3 Related Works

In this section, we briefly review general approaches for privacy-preserving machine learning, which are based on Trusted Execution Environments (TEE), data encryption and obfuscation separately.

3.1 Trusted Execution Environments

Trusted Execution Environments refers to isolated execution environments that guarantee data loaded inside to be protected with respect to confidentiality. Software Guard Extensions (SGX) [12] from Intel provide an enclave in the hardware level that protects the code and data from all other software on the platforms. However, SGX only supports Intel CPUs and host memory. [21] implemented a range of data-oblivious machine learning algorithms in trusted SGX-processors based on the elimination of data-dependent accesses. [15] Introduced a deep learning framework Myelin which transforms models into a privacy-preserving model graphs and train them on a remote server with enclave inside on sensitive data. [13] present a system Chiron which enables data holders to use ML-as-a-service without revealing their data to the service providers. They achieved this by performing the training process in Ryoan [14] sandbox, which provides an isolated environment for service providers to operate sensitive data. [8] presented a fully-automated system PRIVADO which can generate enclave-compatible code given an ONNX description of a model. [25] proposed a framework Slalom that securely delegates linear execution in a DNN from a TEE to a faster but untrusted processor to accelerate the training process. [10] proposed another framework MLCapsule which executes models on the user's side while offering the service provider sufficient control and security of its model. Since enclaves effectively secure private data, we will conduct our research on them. Graviton [26] implements an enclave on GPUs, which is most relevant to our work.

3.2 Data Encryption and Obfuscation

Data encryption and obfuscation are another efficient approach to prevent sensitive data from untrusted servers. Two of most common encryption techniques applied to privacy-preserving machine learning are homomorphic encryption (HE) [4, 7, 17, 20] and secure multi-party computation (SMPC) [24, 16, 23]. The goal of homomorphic encryption is to allow us to perform addition and multiplication operations on the encrypted data without changing the encryption structure. And secure multi-party computation (SMPC) is a type of protocol that allows multiple parties to jointly compute a function without learning each other's input. Compared with enclaves, however, both HE and SMPC bring in prohibitive computational and communication cost, especially when applied to deep neural networks [10, 15, 19].

In addition to data encryption, data obfuscation can also conceal information contained in the data to prevent its disclosure. [29] proposed a methodology to adds random noise to training data while maintaining high accuracy of models' output. [19] devised a system SHREDDER to eliminate the private information contain in communicated cloud data by learning the noise distributions of it. Nevertheless, the performance of these data-obfuscating approaches largely depends on the data itself and is less general than enclave-based approaches.

4 Research Plan

In this section, several details of the research plan including timeline, anticipated results, and evaluation methods are discussed.

4.1 Timeline

We have around 6 weeks for accomplishing this project. There are several subtasks in the project:

1. Implementation of data decryption on GPU. Data decryption on GPU should be implemented as an extension of the deep learning framework. We estimate it will cost two members 2 weeks.

2. Implementation of data encryption tools. Users should use these tools to encrypt the data before uploading them to the cloud storage systems. We estimate it will cost two members 2 weeks.

3. Key management. Keys for decrypting the data should be transferred to GPU through trusted SGX memory zones. It is based on subtask 1. We estimate it will cost two members another 2 weeks.

4. Testing code(models). We need to build some testing models for validating the evaluating our system. It will cost two members another 2 weeks.

We estimate the whole project will cost the whole team (4 members) 4 weeks. Another 2 weeks will be used for confirming the results and writing the report.

4.2 Anticipated Results and Evaluation

What we anticipate is that all testing models can be trained correctly and efficiently on our proposed system. The loss in training speed should be less than 5%, while the performance of testing models should maintain the same as when it runs on its original platform (with a difference less than 1% in performance evaluation metrics of the models).

References

[1] Cloud automl - custom machine learning models — google cloud.

[2] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016.

[3] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*, volume 4. Austin, TX, 2010.

[4] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser. Machine learning classification over encrypted data. In *NDSS*, volume 4324, page 4325, 2015.

[5] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015.

[6] R. Collobert, S. Bengio, and J. Mariéthoz. Torch: a modular machine learning software library. Technical report, Idiap, 2002.

[7] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*, pages 201–210, 2016.

[8] K. Grover, S. Tople, S. Shinde, R. Bhagwan, and R. Ramjee. Privado: Practical and secure dnn inference with enclaves. *arXiv preprint arXiv:1810.00602*, 2018.

[9] A. Hannun, B. Knott, S. Sengupta, and L. van der Maaten. Privacy-preserving contextual bandits. *arXiv preprint arXiv:1910.05299*, 2019.

[10] L. Hanzlik, Y. Zhang, K. Grosse, A. Salem, M. Augustin, M. Backes, and M. Fritz. Mlcapsule: Guarded offline deployment of machine learning as a service. *arXiv preprint arXiv:1808.00590*, 2018.

[11] E. Hesamifard, H. Takabi, M. Ghasemi, and C. Jones. Privacy-preserving machine learning in cloud. In *Proceedings of the 2017 on Cloud Computing Security Workshop*, pages 39–43, 2017.

[12] M. Hoekstra, R. Lal, P. Pappachan, V. Phegade, and J. Del Cuvillo. Using innovative instructions to create trustworthy software solutions. In *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '13, New York, NY, USA, 2013. Association for Computing Machinery.

[13] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel. Chiron: Privacy-preserving machine learning as a service. *arXiv preprint arXiv:1803.05961*, 2018.

[14] T. Hunt, Z. Zhu, Y. Xu, S. Peter, and E. Witchel. Ryoan: A distributed sandbox for untrusted computation on secret data. *ACM Transactions on Computer Systems (TOCS)*, 35(4):1–32, 2018.

[15] N. Hynes, R. Cheng, and D. Song. Efficient deep learning on multi-source private data. *arXiv preprint arXiv:1807.06689*, 2018.

[16] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan. {GAZELLE}: A low latency framework for secure neural network inference. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 1651–1669, 2018.

[17] J. Liu, M. Juuti, Y. Lu, and N. Asokan. Oblivious neural network predictions via minionn transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 619–631, 2017.

[18] P. Mell, T. Grance, et al. The nist definition of cloud computing. 2011.

[19] F. Mireshghallah, M. Taram, P. Ramrakhyani, D. Tullsen, and H. Esmaeilzadeh. Shredder: Learning noise to protect privacy with partial dnn inference on the edge. *arXiv preprint arXiv:1905.11814*, 2019.

[20] P. Mohassel and Y. Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 19–38. IEEE, 2017.

[21] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa. Oblivious multi-party machine learning on trusted processors. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 619–636, 2016.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[23] M. S. Riazi, M. Samragh, H. Chen, K. Laine, K. Lauter, and F. Koushanfar. {XONN}: Xnor-based oblivious deep neural network inference. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1501–1518, 2019.

[24] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar. Chameleon: A hybrid secure computation framework for machine learning applications. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 707–721, 2018.

[25] F. Tramer and D. Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. *arXiv preprint arXiv:1806.03287*, 2018.

[26] S. Volos, K. Vaswani, and R. Bruno. Graviton: Trusted execution environments on gpus. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 681–696, 2018.

[27] C. Wang, Q. Wang, K. Ren, and W. Lou. Privacy-preserving public auditing for data storage security in cloud computing. In *2010 proceedings ieee infocom*, pages 1–9. Ieee, 2010.

[28] H. Wu, Y. Ding, C. Winer, and L. Yao. Network security for virtual machine in cloud computing. In *5th International Conference on Computer Sciences and Convergence Information Technology*, pages 18–21. IEEE, 2010.

[29] T. Zhang, Z. He, and R. B. Lee. Privacy-preserving machine learning through data obfuscation. *arXiv preprint arXiv:1807.01860*, 2018.

[30] D. Zissis and D. Lekkas. Addressing cloud computing security issues. *Future Generation computer systems*, 28(3):583–592, 2012.