A Faster and More Accurate Secure Model Serving Framework on the Cloud

PI: Eric P. Xing, Professor, School of Computer Science, Carnegie Mellon University

Cash funding needed: 80,000 USD

AWS promotional credits needed: 18,987 USD

Abstract:

Serving large-scale machine learning (ML) models with secure guarantees usually incur prohibitive computation overhead. In this proposal, we propose to study three coherent thrusts to perform fast and accurate secure cloud serving for deep learning models leveraging the multi-party computation (MPC) framework. From the algorithmic perspective, it has been widely observed that certain operators in the ML models, *e.g.*, ReLU and Softmax, when deployed under the MPC framework, dominate the computation overhead. Thus, prior work proposes to approximate such operators using alternatives that are more computation friendly in MPC, e.g., replacing ReLU with its polynomial approximations. Enabling such approximations, however, inevitably introduces model accuracy drops. To mitigate the accuracy drop of existing secure inference methods, we propose to use knowledge distillation (KD) to improve the accuracy of the secure models when enabling the MPC-friendly approximations, we refer to the method as "*knowledge-distilled secure training*" (**Thrust 1**). Guided by an empirical observation that when enabling MPC-friendly approximations, adding additional residual connections in the neural network architectures helps to improve the network accuracy effectively. Therefore, from a model architectural perspective (**Thrust 2**), we propose a novel security-aware model architecture (SAMA) that is easier to encrypt after model development for secure model serving and enjoys high model accuracy. From the system perspective (**Thrust 3**), we propose an algorithm-system co-optimization to serve as a secure model for fast inference. The three thrusts can be combined to serve the model **accurately**, **fast**, and with **security guarantees**.

Keywords: Deep learning; data privacy; cloud computing; model serving

Introduction:

Machine Learning as a service (MLAAS) has become a paradigm in the deployment of ML algorithms [1]. Usually, the cloud provider conducts model development before making it available for users to perform inference. During the model serving phase, however, either sending users' data to the model provider, or revealing model weights to the user without any protection can bring severe privacy issues. For instance, the face recognition system at home collects users' personal data; model weights can reveal training set information as discussed in [2]. Addressing these concerns can vastly facilitate the deployment and serving of deep learning models over the cloud. In this project, we propose to address the privacy issues in a principal way by enabling the three most important characteristics of data privacy model inference in the cloud serving environment: **fast**, **accurate**, and **secure**, through *algorithm* level, *model* level, and *system*-level designs.

Currently, the ML community and security community address the problem discussed above independently with their own skill sets. However, neither the ML nor the security community has formulated this problem in a systematic way. Moreover, there is no existing solution that focuses on the entire software/application stack, *i.e.*, system, algorithm/model design, and applications. ML

community has been focusing on introducing privacy to the model parameters of the developed models for data-private model serving. For instance, differential privacy (DP) introduces private guarantees via adding an amount of Gaussian noise to the data [4]. However, enabling DP requires a good trade-off between the privacy guarantees and model accuracy, *i.e.*, pushing hard on a perfect DP guarantee will vastly sacrifice the model accuracy and vice versa. The security community, on the other hand, focuses on providing strong security/privacy guarantees. However, such strong guarantees are unavoidably attained at the cost of heavy computation costs. For instance, secure MPC frameworks use heavy cryptographic protocols, *e.g.*, fully homomorphic encryption (FHE), to let the model provider and users jointly compute the prediction while revealing no additional information to each other.

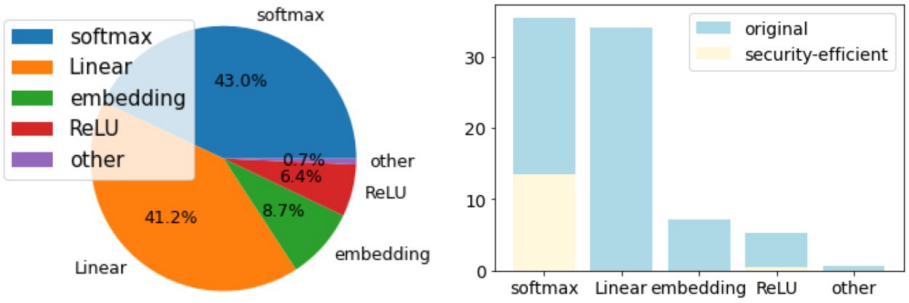

Figure 1: ***(Left)*** *Inference run-time decomposition of a large language model [3]. Fast ML operations (ReLU, softmax) are slow under the MPC framework. (**Right**) Run-time comparison with efficient approximations on a transformer model.*

In this proposal, *we seek to provide a solution for a fast, accurate model serving over the cloud with strong security/privacy guarantees via system-algorithm co-optimization*. Now we briefly introduce our technical roadmap: to achieve high-security guarantee, we follow frameworks in the security community, i.e., the MPC-based security methods for encrypting developed neural network models. Concretely, MPC approaches have been considered in ML tasks [5]. However, achieving accurate and fast inference in MPC frameworks is challenging. This is because: (i) accurate ML models include non-polynomial operations, which are fast in plain-text inference, but slow when converting to encrypted version under MPC frameworks (Figure 1); (ii) the scale of state-of-the-art ML models are typically massive, e.g., 175 billion for GPT-3 [13]. Even with fast and effective operations in the MPC framework, extensively evaluating such operations is still time-consuming. We propose three coherent thrusts to solve these two challenges. To tackle (i), we find that approximating non-polynomial operations with cryptography-friendly operations can significantly reduce the run-time, *e.g.*, replacing ReLU with its polynomial approximation (Figure 1). However, naively enabling these MPC-friendly approximations will give significant accuracy drops [6]. From an algorithmic aspect (**Thrust 1**), we propose a KD approach to transfer knowledge from non-polynomial operations to approximated operations and to enhance the model accuracy when enabling MPC-friendly approximations. From the model perspective (**Thrust 2**), we propose a novel security-aware model architecture (SAMA) that is easier to encrypt after model development for secure model serving and enjoys high model accuracy. This is based on our empirical observation that certain components, e.g., residual connections in the model architecture help the model accuracy to improve further when converted to MPC frameworks. To tackle (ii), we conduct algorithm-system co-optimization (**Thrust 3**) for a more efficient private model serving on the cloud. On the algorithmic side, we plan to enable sparsification and quantization methods introduced to faster inference to further reduce the computation complexity of our

proposed SAMA models. On the system side, we plan to add a layer to compile our sparsified and quantized SAMA models before handing them to the existing secure-inference engines, *e.g.*, Crypten to achieve further inference speedups [5, 12]. An overview of our proposed fast secure inference framework is shown in Figure 2.

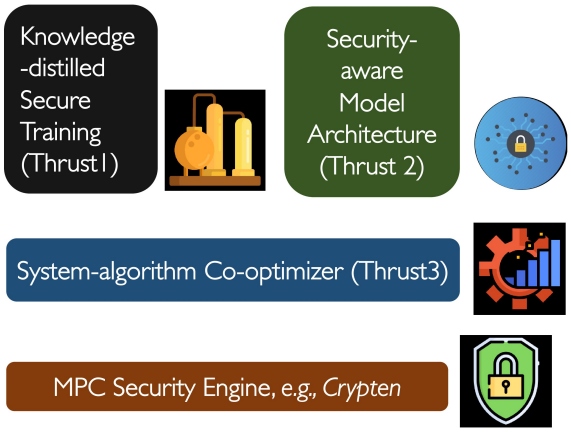

Figure 2. An overview of our proposed secure inference framework.

Methods:

- **Thrust 1 – Knowledge-distilled secure training for faster and more accurate secure inference**.

Formulation (informal): We define two types of networks, the original model $\boldsymbol{M}$(which enjoys the full accuracy of the developed model, but has also high computation overhead), and the approximation model $\acute{\boldsymbol{M}}$ (where MPC-friendly approximations are enabled, which mitigates the computation overhead significantly but also causes accuracy drop). We propose to develop a training methodology that can guide $\acute{\boldsymbol{M}}$ to be more accurate under the supervision of $\boldsymbol{M}$ In the end, we can use $\acute{\boldsymbol{M}}$ to perform inference with high accuracy (and fast). A typical construction of $\acute{\boldsymbol{M}}$ is to replace non-linear functions (which are necessary operations in neural networks) in $\boldsymbol{M}$with some approximations. For instance, [6] approximates a ReLU function with a quadratic function, which preserves the modeling capacity but is very fast to evaluate in a secure framework.

The proposed technical roadmap: One promising direction that can guide the learning of $\acute{\boldsymbol{M}}$ is the knowledge distillation framework [7], where $\boldsymbol{M}$ transfers knowledge to $\acute{\boldsymbol{M}}$ by providing its intermediate values as soft labels. We conduct an ablation study of enabling KD over the MPC-encrypted networks, i.e., $\acute{\boldsymbol{M}}$(the results are shown in Table 1). The performance of a model instance of $\acute{\boldsymbol{M}}$(Transformer based language model in the example). One can observe that training the $\acute{\boldsymbol{M}}$ architecture directly from scratch (i.e., randomly initialized) gives 69.82% accuracy. However, when incorporating the knowledge of $\boldsymbol{M}$(i.e., using the developed $\boldsymbol{M}$with 84.4% original accuracy as a teacher), the final accuracy of the student network $\acute{\boldsymbol{M}}$jumps directly to 79.3%. The 10% accuracy improvement introduced by KD demonstrates its effectiveness promising direction to preserve the accuracy from $\boldsymbol{M}$ Table 1 (bottom) shows the results of a similar experiment on convolutional neural networks (CNNs). One can also observe, from the results in Table 1, that the Transformer-based model enjoys more benefits because $\acute{\boldsymbol{M}}$ is constructed with more approximations (*i.e.*, a transformer has one softmax function per layer, while a CNN only has one softmax function per network).

Model candidate	w/o Distillation Accuracy (%)	w/ Distillation Accuracy (%)	Speedup
$\boldsymbol{M}$	84.4		1.0×
$\acute{\boldsymbol{M}}$	69.8	79.3	1.94×

Model candidate	w/o Distillation Accuracy (%)	w/ Distillation Accuracy (%)	Speedup
$\boldsymbol{M}$	90.5		1.0×
$\acute{\boldsymbol{M}}$	82.7	84.1	1.37×

*Table 1: (**Top**) Distillation experiments using TinyBert [8] on the MNLI dataset. (Bottom) using CNN on Cifar-10 dataset.* $\acute{\boldsymbol{M}}$ *is constructed using approximations to ReLU and Softmax [6], [9].*

- **Thrust 2- Security aware model architecture (SAMA)**

In this Thrust, we seek to fundamentally understand, from the model architecture perspective, why $\acute{\boldsymbol{M}}$ has lower accuracy than $\boldsymbol{M}$ Thus, we can train $\acute{\boldsymbol{M}}$ with high accuracy while remaining fast. Our preliminary studies have found residual connections to be one such factor (Table 2). When $\acute{\boldsymbol{M}}$ is trained with residual connection, the accuracy improves by 5.3%, where the gap to $\boldsymbol{M}$ with residual connection is 9.45%. One natural future direction is to include more structured residual connections (e.g., DenseNet style architecture [10]).

An orthogonal preliminary study along the architecture direction shows that Transformer-style architecture may be robust to approximated operations (Table 3). On three different tasks (MNLI, Cifar-10, and WMT-14), transformer-based models show only small accuracy drops when using the approximated version $\acute{\boldsymbol{M}}$ In conclusion, we plan to more systematically study factors that cause the accuracy gap between $\acute{\boldsymbol{M}}$ and $\boldsymbol{M}$, and how to bridge it.

	$\acute{\boldsymbol{M}}$ w/o residual	$\acute{\boldsymbol{M}}$ w/ residual	$\boldsymbol{M}$
Accuracy (%)	82.7	88.0	92.2

Table 2: residual connection experiments on CIFAR-10 dataset. $\boldsymbol{M}$ *is a 6-layer CNN network.*

	MNLI	Cifar-10	WMT-14
$\boldsymbol{M}$	64.0	82.0	27.77
$\acute{\boldsymbol{M}}$	62.4	80.4	27.81

Table 3: $\boldsymbol{M}$ and $\acute{\boldsymbol{M}}$ performance on various datasets(tasks). *Both architectures are transformers.*

- **Thrust 3 - System-algorithm co-optimization for faster inference**

The second challenge can be solved trivially by training smaller models; however, the final model accuracy will surely be worse. Existing techniques such as quantization and pruning have shown to be effective in plain-text ML models [12]. In Figure 2, we conduct experiments to verify the potential of quantization and pruning in the model serving setting by randomly adding (or dropping) weights to a pre-trained network.

In the MPC framework, the underlying protocols may also benefit from these techniques. For instance, [6] co-design the protocols to enable quantization and pruning in the evaluation of activation functions. In this direction, we plan to discover more opportunities in the underlying protocols such

that we can apply the model compression technique. The current secure inference engine, however, does not directly support fast inference for sparsified and quantized model inference. Thus, on the algorithmic side, we plan to enable sparsification and quantization methods introduced for faster inference to further reduce the computation complexity of our proposed SAMA models. On the system side, we plan to add a layer to compile our sparsified and quantized SAMA models before handing them to the existing secure-inference engines, e.g., Crypten to achieve further inference speedups [5, 12].

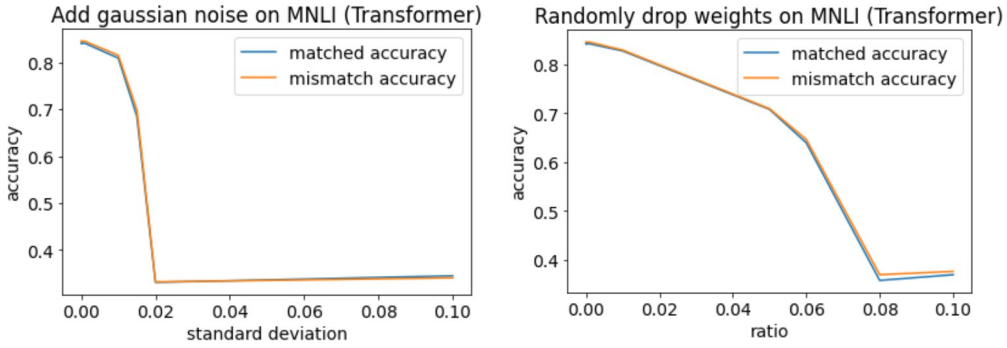

Figure 2: (Left) Adding Gaussian Noise with different standard deviation (Right) Randomly dropping weights with different ratios to a trained transformer network on MNLI.

Expected Results: The final product will combine all these three orthogonal directions to enable fast, secure, and accurate model inference. More concretely, we plan to submit each direction to conferences as our milestones (shown in Table 4).

Direction	Submission	Estimated date	Code release
Thrust 1	ICLR 2023	Oct 2022	Yes
Thrust 2	ICML 2023	Jan 2023	Yes
Thrust 3	NSDI 2024	Apr 2023	Yes
Final Product	Open-source platforms	July 2023	Yes

Table 4: project milestones and expected deliverables

Final product: At a high level, the final product will use better $\acute{M}$(model direction) together with guidance (algorithm direction) from M for fast and accurate model inference, evaluated on our new system (system direction) where they can be secure and additionally sped up.

Budget breakdown: We expect to cover the stipend for one graduate student during the period of the project (12 months anticipated). The monthly stipend for a graduate student at Machine Learning Department at CMU is 3,200 USD. Tuition and other fees are 49,592 (including required insurance), which leads to 87,992 USD in total. Thus, we request the maximum amount of funding of 80,000 per year. The rest amount outside the funding, e.g., the extra student stipend and the travel costs will be covered by the other funding resources from the PI.

The usage of AWS credits: We plan to use a *p3.2xlarge* EC2 instance for 5 hours per day for development, and a *p3.8xlarge* instance for 3 hours for evaluating proposed methods, which summarizes to 18,987 for one year of use.

Appendix A - Reference:

[1] Bae, Ho, et al. "Security and privacy issues in deep learning." *arXiv preprint arXiv:1807.11655* (2018).

[2] Fredrikson, Matt, Somesh Jha, and Thomas Ristenpart. "Model inversion attacks that exploit confidence information and basic countermeasures." *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015.

[3] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).

[4] Abadi, Martin, et al. "Deep learning with differential privacy." *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016.

[5] Knott, Brian, et al. "Crypten: Secure multi-party computation meets machine learning." *Advances in Neural Information Processing Systems* 34 (2021): 4961-4973.

[6] Chou, Edward, et al. "Faster cryptonets: Leveraging sparsity for real-world encrypted inference." *arXiv preprint arXiv:1811.09953* (2018).

[7] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." *arXiv preprint arXiv:1503.02531* 2.7 (2015).

[8] Jiao, Xiaoqi, et al. "Tinybert: Distilling bert for natural language understanding." *arXiv preprint arXiv:1909.10351* (2019).

[9] Mohassel, Payman, and Yupeng Zhang. "Secureml: A system for scalable privacy-preserving machine learning." *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017.

[10] Huang, Gao, et al. "Densely connected convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[11] Weihao Yu, et al. "Metaformer is actually what you need for vision." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

[12] Song Han, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." *ICLR* 2016.

[13] Tom B. Brown et al. "*Language Models are Few-Shot Learners*", NeurIPS 2021.

Appendix B – CV of PI

Name: Eric P. Xing

Position Title and institution: Professor, Carnegie Mellon University; President, Mohamed bin Zayed University of Artificial Intelligence

A.Professional Preparation

1993 Physics, B.Sc. Tsinghua University, Beijing, China

1999 Molecular Biology and Biochemistry, Ph.D. Rutgers University, New Brunswick, NJ

2004 Computer Science, Ph.D. University of California, Berkeley, Berkeley, CA

B.Appointments

2016 - present Associate Department Head of Research, Machine Learning Department, Carnegie Mellon University

2014 - present Professor, Machine Learning Department & Language Technologies Institute & Computer Science Department, School of Computer Science, Carnegie Mellon University

2020 - present Founder, Chief Scientist, Chairman of the Board, Petuum, Inc.

2011 - 2014 Associate Professor with Tenure, Machine Learning Department & Language Technologies Institute & Computer Science Department, School of Computer Science, Carnegie Mellon University

2004 - 2009 Assistant Professor, Machine Learning Department & Language Technologies Institute & Computer Science Department, School of Computer Science, Carnegie Mellon University

C. Publications

[1] Qiao, Aurick, et al. "Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning." 15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21). 2021.

[2] Ho, Qirong, et al. "More effective distributed ml via a stale synchronous parallel parameter server." Advances in neural information processing systems 26 (2013).

[3] Xie, Pengtao, et al. "Orpheus: Efficient distributed machine learning via system and algorithm co-design." Proceedings of the ACM Symposium on Cloud Computing. 2018.

[4] Wei, Jinliang, et al. "Managed communication and consistency for fast data-parallel iterative analytics." Proceedings of the Sixth ACM Symposium on Cloud Computing. 2015.

[5] Qiao, Aurick, et al. "Litz: Elastic Framework for {High-Performance} Distributed Machine Learning." 2018 USENIX Annual Technical Conference (USENIX ATC 18). 2018.

[6] Zhang, Hao, et al. "Poseidon: An efficient communication architecture for distributed deep learning on {GPU} clusters." 2017 USENIX Annual Technical Conference (USENIX ATC 17). 2017.