Performance Evaluation of a Differentially-private Neural Network for Cloud Computing

Nathaniel D. Hoefer, and Sergio A. Salinas Monroy, *IEEE Member*
Department of Electrical Engineering and Computer Science
Wichita State University
Wichita, KS

***Abstract*—Due to the large computational cost of data classification using deep learning, resource-limited devices, e.g., smart phones, PCs, etc., offload their classification tasks to a cloud server, which offers extensive hardware resources. Unfortunately, since the cloud is an untrusted third-party, users may be reluctant to share their private data with the cloud for data classification. Differential privacy has been proposed as a way of securely classifying data at the cloud using deep learning. In this approach, users conceal their data before uploading it to the cloud using a local obfuscation deep learning model, which is based on a data classification model hosted by the cloud. However, as the obfuscation model assumes that the pre-trained model at the cloud is static, it leads to significant performance degradation under realistic classification models that are constantly being updated. In this paper, we investigate the performance of differentially-private data classification under a dynamic pre-trained model, and a constant obfuscation model. We find that the classification performance decreases as the pre-trained model evolves. We then investigate the classification performance under an obfuscation model that is updated alongside the pre-trained model. We find that with a modest computational effort the obfuscation model can be updated to significantly improve the classification performance. under a dynamic pre-trained model.**

I. Introduction

Deep learning is a machine learning technique based on neural networks that has been successfully used to solve a myriad of problems that are difficult to formalize. For example, by employing easily accessible deep learning models such as MobileNet [1], developers have been able to perform facial recognition-based authentication, and speech recognition [2]. Due to the large computational cost of deep learning, resource-limited devices, e.g., smart phones, PCs, etc., offload their classification tasks to a cloud server, which offers extensive hardware resources [3] that lead to fast response times with high accuracy. In fact, some companies have started to train deep learning models, which is the most computationally intensive task in deep learning, at the cloud, and making them publicly available. For example, Google's Cloud Vision API [4] allows users to classify images with a pre-trained deep learning model that is hosted at the cloud.

However, since the cloud is an untrusted third-party, users may be reluctant to share their private data with the cloud to use the deep learning models. As an example, offloading medical images to a cloud-hosted deep learning model to diagnose diseases could compromise the patient privacy. Ultimately, once the private data is uploaded to the cloud there is no assurance that the cloud will solely use it for the user's intended purpose.

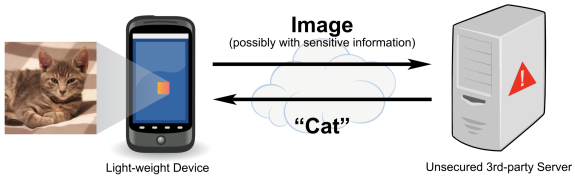

Fig. 1. Offloading classification scenario.

To address this issue, researchers have proposed several techniques including data partitioning, homomorphic encryption and differential privacy. Schlitter [5] proposes a privacy-preserving data partition scheme by utilizing secure matrix addition on horizontally partitioned data used in neural network learning from multiple parties. Each party trains their own network, then the parameters are shared and securely combined. Shokri et al. [6] build on [5] by having each party share only a small amount of their private network's parameters to improve overall accuracy and privacy.

Under homomorphic encryption, Rivest et al. [7] and Gilad-Bachrach et al. [8] encrypt users' data before offloading it to the cloud, and then use a special deep learning model at the cloud, which is also based on homomorphic cryptography, to perform classification over the encrypted data [9]. Yuan and Yu [10] utilize BGN homomorphic encryption for multi-party collaborative network learning over arbitrarily partitioned data. However, this scheme requires ciphertext to be sent back to their respective party multiple times. Zhang et al. [11] offer an improvement to the scheme by using BGV homomorphic encryption and approximate the Sigmoid function as a polynomial function. A common thread between these works is the very high computational expense of homomorphic encryption and the requirement for several rounds of communication between the user and the cloud in which the data is decrypted and re-encrypted to ensure a minimum level of accuracy.

Under differential privacy [12], the users' privacy is preserved by adding randomness to its uploaded data in such a way that values cannot be singled out. For example, Ren et al. [13] implements a privacy-preserving deep learning model that selectively obfuscates faces while retaining the ability to classify the images later with a special deep learning model. Leroux et al. [14] expounds the obfuscation deep

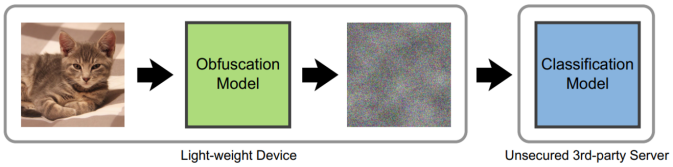

Fig. 2. High-level view of the system.

learning model in [13] to fully obfuscate the entire image while being able to use a generic deep learning model for image classification. The classification model in [14] assumes the generic classification model is static, which is not a reasonable assumption under current publicly available models which are constantly being trained.

Although differential privacy offers a computationally efficient alternative to homomorphic encryption, current differential privacy approaches, e.g., [13], and [14], establish a dependency between the obfuscation model and the classification model. This dependency influences the accuracy of the system when either one of the models is modified. Specifically, when an obfuscation neural network is trained using a specific classification model at an specific state, the obfuscation model learns to retain the attributes of this particular classification model. If the classification model is later modified, as is the case when further training occurs, the attributes in the classification model may change. This leads to a dramatic impact on the accuracy of the classification based on the obfuscated images.

In this paper, we implement and evaluate the differentially-private neural network proposed by [14]. In particular, we first implement a local obfuscation neural network to distort input images before offloading them to a cloud-hosted pre-trained classification neural network at the cloud. The obfuscation model must retain the image attributes observed by the pre-trained classification model, so that the obfuscated image can be properly classified with a high degree of accuracy. Once the obfuscated image is classified, the remote server responds with the classification results. We thoroughly evaluate our implementation, and find that additional training on the classification neural network has a significant impact on the accuracy of the obfuscation model resulting in a decrease in accuracy of nearly 50%.

II. System Architecture

We consider a resource-rich cloud server and a resource-limited mobile client who aims to use deep learning neural networks to classify images as seen in Figure 2. The classification neural network is assumed to be previously trained and located in the cloud. The cloud may observe the uploaded data, and it may further train the classification neural network without notification. The obfuscation neural network is implemented by the user and it is assumed to be previously trained, but available for as further training.

TABLE I
Obfuscator and deobfuscator network architecture

Input Size	Module	Output Channels	Stride
3 x 32 x 32	Conv2D	32	1
32 x 16 x 16	Bottleneck	32	2
64 x 16 x 16	Bottleneck	64	2
128 x 4 x 4	Bottleneck	128	2
128 x 4 x 4	Upsample Bottleneck	64	1
64 x 8 x 8	Upsample Bottleneck	32	1
32 x 16 x 16	Upsample Bottleneck	3	1

III. Performance Evaluation of a Differentially-Private Neural Network

A. Experiment Setup

In this section, we empirically quantify the dependency between the obfuscation model and the classification model outlined by Leroux et al. [14] to investigate the feasibility of relying on a potentially mutable third-party model. Specifically, their obfuscation neural network is implemented using a Generative Adversarial Network (GAN) [15] which use two neural networks to compete against each other to further improve the models as seen in Figure 3. Leroux et al. uses a deobfuscation neural network that trains alongside the obfuscation neural network, whose sole purpose is to reconstruct the original image closely as possible from the obfuscated image. The obfuscation neural network then considers the accuracy of the classification from the obfuscated image, and the accuracy of the reconstructed image to improve its obfuscation. The result is an obfuscation model that produces an obfuscated image that can still be classified while not being able to be restored by the deobfuscation model. The network architecture for the obfuscation and deobfuscation models can be seen in Table I and is taken directly from [14]. The classification model uses the ResNet18 [17] neural network and resides locally to simulate a cloud platform since the training results will be the same. All models are implemented using PyTorch [18].

The dataset used to train the neural networks is the CIFAR-10 dataset [16], which consists of 32x32 RGB images, each containing 1 of 10 different objects or animals with its associated label. There are 40,000 images for training and 10,000 images for testing the accuracy. To allow for further improvement, the baseline models are trained using only the first 20,000 unaltered images of the CIFAR-10 dataset. For a baseline, he classification model is trained for 50 epochs to reach an accuracy of 70.82%. The obfuscation model is also implemented using PyTorch with the GAN introduced by Leroux et al., and for a baseline, is trained for 100 epochs to reach an accuracy of 40.68%. To evaluate the impact of the baseline obfuscation accuracy when the baseline classification model is further trained, the baseline classification model is trained for another 50 epochs using the full CIFAR-10 dataset including random horizontal image flips. Each five epochs, the accuracy of the baseline obfuscation model is tested and recorded with the updated classification model. Then the impact of the recovery rate of the obfuscation model

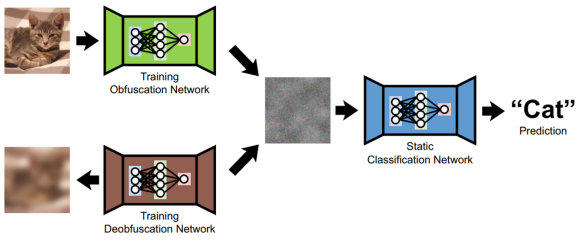

Fig. 3. Training via Generative Adversarial Network (GAN).

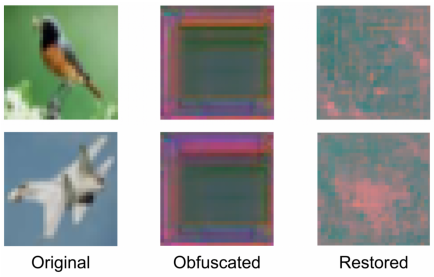

Fig. 4. Sample images processed by obfuscation and deobfuscation models.

is evaluated by training the baseline obfuscation model using the final classification model for another 100 epochs with the full CIFAR-10 dataset including random horizontal image flips. This will result in a new final classification model and a final obfuscation model, both with a different accuracy than their baseline equivalents which can be seen in Figure 5.

B. Results

We quantify the dependency of the obfuscation model using two metrics:

1) The impact of the baseline obfuscation accuracy when the baseline classification model is further trained.
2) The recovery rate of the obfuscation model as it is further trained with the final classification model.

For the first experiment, the accuracy of the classification model, as seen in Figure 6, jumped from 70.82% to approximately 80% within the first 5 epochs trained with the new dataset. The final accuracy after 50 epochs is 81.53% which is a jump of approximately 10%. The baseline obfuscation model's accuracy dramatically decreased from 40.68% to 20.52% within the first 10 epochs of the classification training, resulting in nearly a 50% decrease.

The second experiment demonstrated how quickly the baseline obfuscation model can recover after further training using the latest classification model as seen in Figure 7. After 5 epochs of training, the accuracy surpassed the original baseline obfuscation accuracy using the baseline classification model, then after 100 epochs, the accuracy increased to 73%. Figure 5 displays the baseline accuracy of the models after being trained for the initial epochs, and the accuracy of the final models after the continued training. The final result of the obfuscated images and restored images produced by the GAN can be seen in Figure 4.

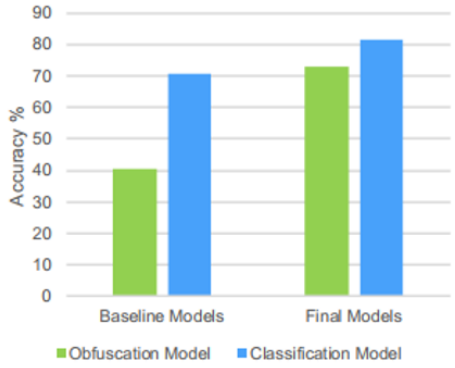

Fig. 5. Accuracy of models before and after further training.

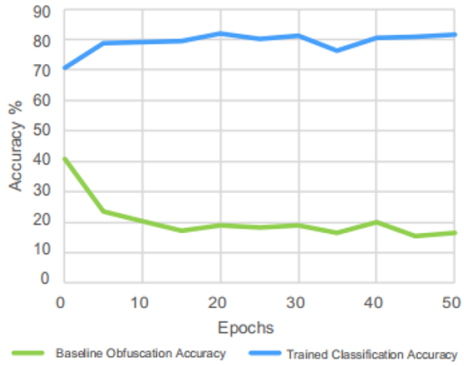

Fig. 6. Experiment 1: Obfuscation model accuracy while Classification model is trained.

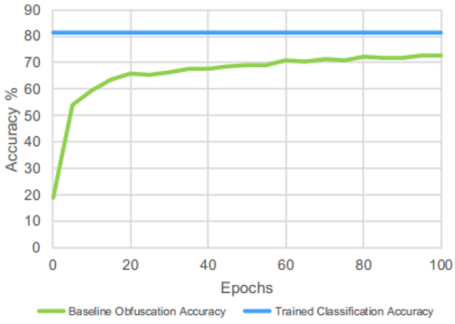

Fig. 7. Experiment 2: Obfuscation model accuracy while training with final Classification model.

IV. Conclusion

The results obtained clearly show that the even small changes in the classification model result in significant accuracy degradation for the the obfuscation model. However, the obfuscation model can quickly recover if it is possible to re-train it with the latest classification model. In the future, the baseline obfuscation accuracy is too low to be viable in a realistic sense, so another baseline obfuscation model with a more realistic accuracy would provide better results. That is, given more time and resources, the user could create a better obfuscation model. Another area of interest would be

to conduct the experiments on other models, whether that be with various obfuscation models or entirely new systems that which contain dependent neural networks.

REFERENCES

[1] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[2] A.-r. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[3] J. Hauswald, T. Manville, Q. Zheng, R. Dreslinski, C. Chakrabarti, and T. Mudge, "A hybrid approach to offloading mobile image classification," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 8375–8379.

[4] A. Vision, "Image content analysis/google cloud platform," 2017.

[5] N. Schlitter, "A protocol for privacy preserving neural network learning on horizontal partitioned data," *PSD*, 2008.

[6] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[7] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[8] 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*, 2016, pp. 201–210.

[9] M. Al-Rubaie and J. M. Chang, "Privacy preserving machine learning: Threats and solutions," *arXiv preprint arXiv:1804.11238*, 2018.

[10] J. Yuan and S. Yu, "Privacy preserving back-propagation neural network learning made practical with cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 212–221, 2014.

[11] Q. Zhang, L. T. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1351–1362, 2016.

[12] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*. Springer, 2011, pp. 338–340.

[13] Z. Ren, Y. J. Lee, and M. S. Ryoo, "Learning to anonymize faces for privacy preserving action detection," *arXiv preprint arXiv:1803.11556*, 2018.

[14] S. Leroux, T. Verbelen, P. Simoens, and B. Dhoedt, "Privacy aware offloading of deep neural networks," *arXiv preprint arXiv:1805.12024*, 2018.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[16] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.