Feature Space Hijacking Attacks against Differentially Private Split Learning

Grzegorz Gawron,[1] Philip Stubbings[1]

[1] LiveRamp, Privacy Tech R&D
greg.gawron@liveramp.com, phil.stubbings@liveramp.com

Abstract

Split learning and differential privacy are technologies with growing potential to help with privacy-compliant advanced analytics on distributed datasets. Attacks against split learning are an important evaluation tool and have been receiving increased research attention recently. This work's contribution is applying a recent feature space hijacking attack (FSHA) to the learning process of a split neural network enhanced with differential privacy (DP), using a client-side off-the-shelf DP optimizer. The FSHA attack obtains client's private data reconstruction with low error rates at arbitrarily set DP ϵ levels. We also experiment with dimensionality reduction as a potential attack risk mitigation and show that it might help to some extent. We discuss the reasons why differential privacy is not an effective protection in this setting and mention potential other risk mitigation methods.

Introduction

Privacy compliant data processing has been receiving an increasing amount of attention recently. Take a consortium of hospitals as a motivational example. Each owns a data set describing treatments given to patients, together with the treatments' outcomes. There is a potential great value in learning from all of the data sets taken together. However, the hospitals are bound by privacy regulations that prevent them from sharing the data in raw form. For a thorough discussion of distributed learning from patient data see Vepakomma et al. (2018a).

Split learning (see Gupta and Raskar (2018)), a type of federated learning, has been proposed to address scenarios like this. It is a neural network training mechanism, which **splits** the network into modules corresponding to the private data silos, so that each module is trained within the silo and no raw data is exposed outside. Another reason for growing popularity of split learning is it's performance. Since only the processed or **smashed** data of lower dimensions is being exchanged among silos and their coordinator, the process is more efficient than some of the more established mechanisms of distributed learning, such as federated averaging. Yet, relying on smashing the data as the only mechanism of protecting privacy has been shown by Erdogan, Kupcu, and Cicek (2021b) to be misguided and that even an honest-but-curious actor having access to smashed data can reconstruct the private inputs. Vepakomma et al. (2018b) have surveyed various methods to deal with split learning privacy leaks having, what they call a **no peek** learning, as their target.

Figure 1: Plots comparing the private input (top row) and its corresponding reconstruction (bottom row) in a successful FSHA attack on MNIST.

In this paper we focus on another technique which can be used to enhance split learning's privacy compliance: differential privacy (DP). Its promise is to provide a rigorous mathematical privacy guarantee (see Dwork et al. (2014)). Abadi et al. (2016) showed that training with DP is feasible, while having control over model performance degradation and privacy cost. There has also been work by Papernot et al. (2020) on fine tuning deep learning with DP to minimise the performance cost with some good results. Geyer, Klein, and Nabi (2017) is another example of the work on optimizing the DP deep learning process.

Theoretical privacy guarantees, when applied in practice, need to be backed by automatic evaluation, which helps to guard against plain bugs or other implementation issues. There are multiple, already well established libraries available, which provide the necessary evaluation components, e.g. ADVERSARIAL-ROBUSTNESS-TOOLBOX[1] or TENSORFLOW-PRIVACY[2]. For a comprehensive overview of attack and defences literature and evaluation best practices we refer the reader to Carlini et al. (2019).

Attack based evaluation is particularly relevant for newly emerging methods like split learning. There have already been first papers published on this subject. One recent attack

[1] https://github.com/Trusted-AI/adversarial-robustness-toolbox
[2] https://github.com/tensorflow/privacy

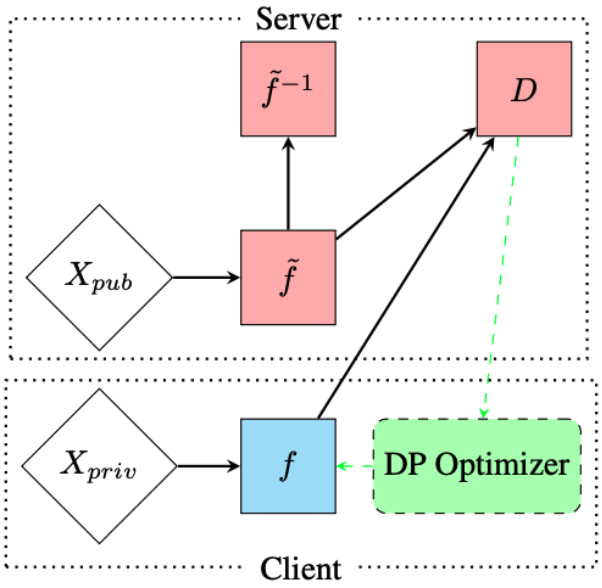


Figure 2: FSHA split network architecture with the DP logic implemented by a DPOptimizer from TENSORFLOW-PRIVACY package

method is feature space hijacking attack FSHA by Pasquini, Ateniese, and Bernaschi (2021), where a malicious attacker taking over the server module is able to reconstruct all of the client private input data with arbitrarily low reconstruction error on MNIST and other visual datasets. See Figure 1 for an illustration of a successful FSHA attack on a MNIST dataset, based on an experiment reproduction from the original paper - without DP applied.

As a testament to the rapid pace of research progress in the field, methods guarding against such attacks have already been presented, see Erdogan, Kupcu, and Cicek (2021a).

Our contributions. This work contributes by applying differential privacy to the client module of a split neural network, running a suite of FSHA attacks against it and drawing conclusions for differentially private split learning system implementations. To our knowledge this is the first FSHA style attack attempt at such a setup. Hitaj, Ateniese, and Perez-Cruz (2017) have shown the limits of applying DP to general collaborative deep learning and we validate the vulnerability of split learning in that respect.

We reuse the software prototype from Pasquini, Ateniese, and Bernaschi (2021) found at https://github.com/pasquini-dario/SplitNN_FSHA and aim to introduce minimal code changes to reach our goal. We also follow the authors' choice of using TENSORFLOW and use its sub-package TENSORFLOW-PRIVACY for the application of DP clipping and noise. See McMahan et al. (2019) for the description of this package.

Architecture

In what follows we briefly describe the original FSHA attack split learning architecture and refer the reader to the original paper of Pasquini, Ateniese, and Bernaschi (2021) for a complete discussion of the components (we use the same notation as in the original paper). We restrict our investigation to a situation where the attacker takes over only the server. The clients are not taken over and believe they are part of a legitimate cooperation. However, as highlighted in the original paper, the same attack would work successfully by taking over one of the clients only. The example neural network in Figure 2 is split into two main components: client and server (in general there could be multiple clients). The client owns their private data stored in a silo X_{priv} and is responsible for training the f model component of the network. The server runs the attacker's code, which instead of running its part of the split learning protocol, trains two components: discriminator D and an auto-encoder $\tilde{f}$ and $\tilde{f}^{-1}$. The auto-encoder, when given inputs from X_{pub}, encodes them using $\tilde{f}$ to an internal representation (feature space) and decodes using $\tilde{f}^{-1}$ with the loss set, so as to learn to reconstruct the original input data. Finally, discriminator D is responsible for classifying the input as coming from either f model or the $\tilde{f}$ model. The ultimate attacker's target is for the discriminator D to force the private model f's output to come from the same feature space as the outputs of $\tilde{f}$ and the $\tilde{f}^{-1}$ to be able to decode it with minimal error, just as it learned to decode the $\tilde{f}$.

The standard way of applying differential privacy during neural network training is on the gradients during back propagation. In our scenario the most natural place to apply the DP clipping and noise to the f gradients is just before applying the gradient updates sent back by server's discriminator D as illustrated by the green fragments in Figure 2. This allows the client data owner to control the noise application process and configure it according to their privacy preferences.

Experiments

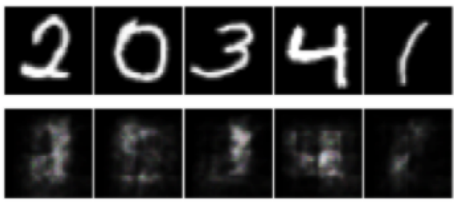
Experimental Setup

We enhanced the FSHA code with DP by introducing only minimal changes, with the aim of being able to easily compare the results. We used **TensorFlow-privacy** python package to add a differentially private optimizer **DPAdamGaussianOptimizer**. We parametrised DP with $\delta = 0$ and various levels of ϵ from 1 to 100.

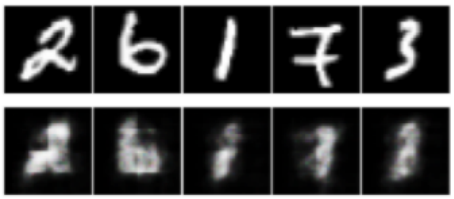
We deployed several GCP Notebooks instances with the following hardware: (a) a2-highgpu-1g (Accelerator Optimized: 1 NVIDIA Tesla A100 GPU, 12 vCPUs, 85GB RAM); and (b) n1-standard-4 (4 vCPUs, 15 GB RAM, 1 NVIDIA Tesla T4). GPU load was 30 percent on (a) and 80 percent on (b) with similar run times, so we settled on the latter to optimize the experiment cost.

MNIST Results

We start with the MNIST dataset because it is widely used for benchmarking and its simplicity helps to clearly show the impact of the application of DP. Note, that we focus on showing the reconstruction visualisations, as they nicely communicate the results. The diagrams of the actual reconstruction errors can be found in the Appendix.

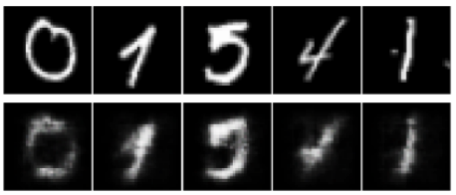


(a) 10 000 iterations, DP with $\epsilon = 10$.



(b) 10 000 iterations, DP with $\epsilon = 100$.

Figure 3: Private input (top row) and its corresponding reconstruction (bottom row) for various epsilon settings and 10 000 iterations.



(a) 100 000 iterations, DP with $\epsilon = 0.5$.

(b) 100 000 iterations, DP with $\epsilon = 10$.

Figure 4: Plots comparing the private input (top row) and its corresponding reconstruction (bottom row) for various epsilon settings and 100 000 iterations.

We reproduced the results from the original paper, without any differential privacy applied. Figure 1 shows a successful FSHA reconstruction attack after 10 000 iterations of 64 element mini-batches. This corresponds to 10 epochs of training, since there are $N = 60\,000$ samples in the training set. The next step is to run the same number of iterations using DP with $\epsilon = 10$ and $\delta = 1/N$. Figure 3a shows that the reconstruction was far less successful, however the reconstruction error was still declining after the training stopped. This suggested that more iterations might make the attack successful. As expected the attack against a split learning with $\epsilon = 100$ is able to reconstruct the private data with much less error (Figure 3b). We continued with the training process running for 100 000 iterations (100 epochs) and at $\epsilon = 10$ it was enough to produce very low-error reconstructions, as illustrated in Figure 4b. Finally, we set ϵ to a much lower value 0.5 and obtained the reconstruction results shown in Figure 4a. Although the error is much higher than with the previous experiment (where $\epsilon = 10$) the reconstruction error was still steadily declining, so by increasing the number of iterations we could expect the error to go down as per previous experiments.

See the Appendix for a more complete record of reconstruction plots.

FASHION-MNIST Results

Beyond the FASHION-MNIST attacks reproduced from Pasquini, Ateniese, and Bernaschi (2021) we have experimented with removing some of the image classes from the public data set and applying differential privacy in both settings.

We first removed the category 0 representing t-shirts/tops and ran the attacks with and without the DP. We present the relevant results in Figure 5a and Figure 5b. In both cases the shapes were mostly visible, but the DP version had less logo details visible (see the two pictures to the right).

There are, however, other categories in the public data set (see Figure 10) that are similar to the t-shirts, so the process had more chance to learn the relevant features to also reconstruct t-shirts. In an attempt to minimise this effect, we remove the category which is least similar to others: bags (category = 8). The result is illustrated in Figures 5c and 5d. Interestingly, the version without DP reconstructs some of the bags as shoes, perhaps because they activate similar latent features learned by the attack model. However we can recognise certain features coming from the private data e.g., the circle on the right t-shirt. With DP added we see no 'bag as shoe' effect. The images are more blurred, but the general shapes are reconstructed. This might be due to regularisation effect of DP. It is important to note that it should be possible to improve this results by running more training iterations (we stopped at 100 000).

To summarise, FASHION-MNIST data set with some of the classes removed from the public data set achieves lower reconstruction precision, especially with DP applied. This helps demonstrate the point that the reconstructions are really generated out of the feature space driven by the access to similar public data. The closer the public and private data distributions are, the more successful the attacks.

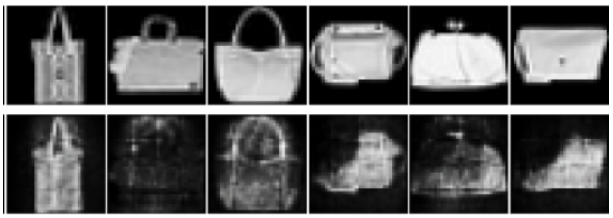
Mitigating the attack with dimensionality reduction

We explored the application of DP during the model training process as a means to mitigate the attack and demonstrated that DP can at most delay FSHA convergence.

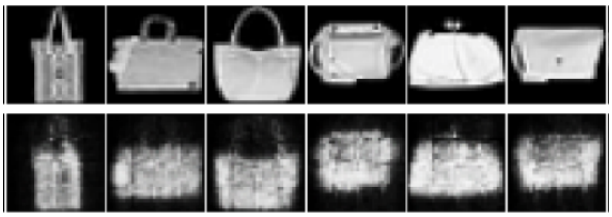
In an un-trusted environment in which it can be assumed a client or server will attempt to exploit the learning protocol by means of the FSHA method, additional steps should be taken to protect the underlying data. One possibility is to apply dimensionality reduction techniques directly to the data prior to training to a level which still yields acceptable model accuracy.

(a) no DP, iterations $= 100\,000$

(b) $\epsilon = 10$, iterations $= 100\,000$



(c) no DP, iterations $= 100\,000$



(d) $\epsilon = 10$, iterations $= 100\,000$

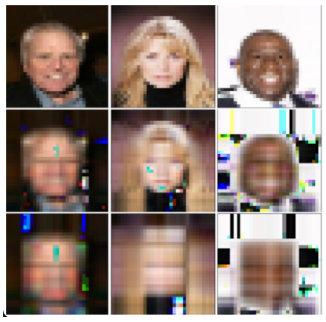
Figure 5: FSHA against FASHION-MNIST with category 0 (tops) missing from the public data set in Figures 5a and 5b and category 8 (bags) missing in Figures 5c and 5d.

To test this hypothesis, we re-run the FSHA attack up to 100k iterations on image data which has been compressed using various levels of PCA principal components as shown in Figure 6a (also in Appendix Figure 11a). We make use of the CELEBA dataset and define the classification task to predict whether the celebrity is likely male or female.

The resulting recovered images shown in Figure 6b (also in Appendix Figure 11b) demonstrate that applying dimensionality reduction to the data can mitigate the attack to a degree in which the celebrities are no longer identifiable. One should note though, that some reconstructed image features might still be obvious and unique to certain individuals.

Conclusions

The FSHA mechanism used is very robust. It only requires that the client part of the model can be successfully trained; and for any useful usage that must be the case. Once the attacker obtains a sufficiently good client model they can reconstruct the private data which are passed from the client to server during the training process. As Pasquini, Ateniese, and Bernaschi (2021) notice, they could also infer certain features out of the private data, if that's what they choose to do.



(a) Image compression

(b) FSHA recovered images

Figure 6: PCA image compression and recovery. 1^{st} row - no compression, 2^{nd} row - principal components = 4, 3^{rd} row - principal components = 2.

It might seem surprising that the DP doesn't give enough protection against FSHA. However, the DP noise applied during the training is designed to prevent the model from memorizing data so that the private data cannot be retrieved from the trained model itself. It must still allow for training useful models. After a successful training, a good DP model must be able to achieve high inference accuracy. The inferred values might well be sensitive and constitute a privacy leak. Hence, having access to the model being trained allows an attacker to either reconstruct or infer properties of the private data. We refer the reader to Hitaj, Ateniese, and Perez-Cruz (2017) for a more thorough discussion of this topic.

Employing some dimensionality reduction techniques might make the potential reconstructions useless, but we must remember that it might not always produce a model with good enough utility and even more importantly the attacker might still infer sensitive properties from the private data.

Detection methods such as ones proposed by Erdogan, Kupcu, and Cicek (2021a) might also be used, but they are not bullet proof and require constant attention to new detection method workarounds used by potential attackers.

As for now, to ensure that the split learning process is safe we should mitigate the risk of someone taking over the server (or indeed any other split network component). The risk mitigation might rely on enforcing a secure compute environment with e.g. only a certified version of the training code running on all the modules of the split network. It also follows that the application of the DP trained split neural network model should happen in an environment of trust. That is, the owners of private data should trust that inferred outputs are appropriately taken care of. It is safe, for instance, if the model is applied directly by the data owner on their own data and they own the inferred results.

Acknowledgements

This work has benefited from valuable input during the 2021 OpenDP Fellows Program[3].

References

Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep Learning with Differential Privacy. *Proceedings of the ACM Conference on Computer and Communications Security*, 24-28-Octo: 308–318.

Carlini, N.; Athalye, A.; Papernot, N.; Brendel, W.; Rauber, J.; Tsipras, D.; Goodfellow, I.; Madry, A.; and Kurakin, A. 2019. On Evaluating Adversarial Robustness. arXiv:1902.06705.

Dwork, C.; Roth, A.; Dwork, C.; and Roth, A. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9: 211–407.

Erdogan, E.; Kupcu, A.; and Cicek, A. E. 2021a. SplitGuard: Detecting and Mitigating Training-Hijacking Attacks in Split Learning. *IACR Cryptol. ePrint Arch.*, 2021: 1080.

Erdogan, E.; Kupcu, A.; and Cicek, A. E. 2021b. UnSplit: Data-Oblivious Model Inversion, Model Stealing, and Label Inference Attacks Against Split Learning. *IACR Cryptol. ePrint Arch.*, 2021: 1074.

Geyer, R. C.; Klein, T.; and Nabi, M. 2017. Differentially Private Federated Learning: A Client Level Perspective. arXiv:1712.07557.

Gupta, O.; and Raskar, R. 2018. Distributed learning of deep neural network over multiple agents. arXiv:1810.06060.

Hitaj, B.; Ateniese, G.; and Perez-Cruz, F. 2017. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. *Proceedings of the ACM Conference on Computer and Communications Security*, 603–618.

McMahan, H. B.; Andrew, G.; Erlingsson, U.; Chien, S.; Mironov, I.; Papernot, N.; and Kairouz, P. 2019. A General Approach to Adding Differential Privacy to Iterative Training Procedures. arXiv:1812.06210.

Papernot, N.; Chien, S.; Song, S.; Thakurta, A.; and Erlingsson, U. 2020. Making the Shoe Fit: Architectures, Initializations, and Tuning for Learning with Privacy. https://openreview.net/forum?id=rJg851rYwH. Accessed: 2021-11-22.

Pasquini, D.; Ateniese, G.; and Bernaschi, M. 2021. Unleashing the Tiger: Inference Attacks on Split Learning. arXiv:2012.02670.

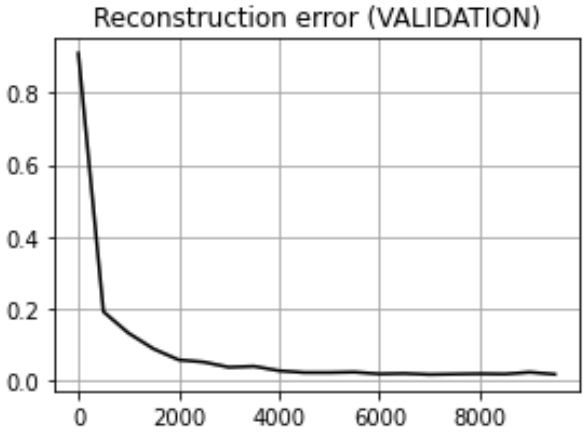
Vepakomma, P.; Gupta, O.; Swedish, T.; and Raskar, R. 2018a. Split learning for health: Distributed deep learning without sharing raw patient data. arXiv:1812.00564.

Vepakomma, P.; Swedish, T.; Raskar, R.; Gupta, O.; and Dubey, A. 2018b. No Peek: A Survey of private distributed deep learning. arXiv:1812.03288.

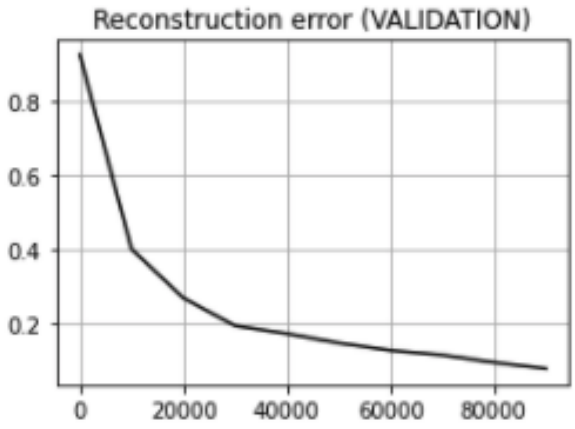
Appendix

Below we include the complete figures from our experiments, where they didn't fit in the main body of the paper.

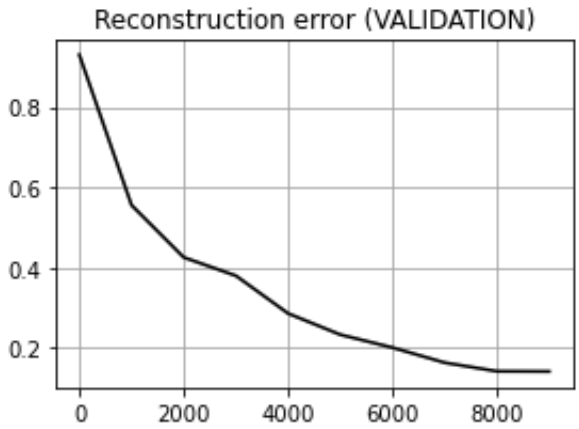
[3]https://opendp.org/fellows-program



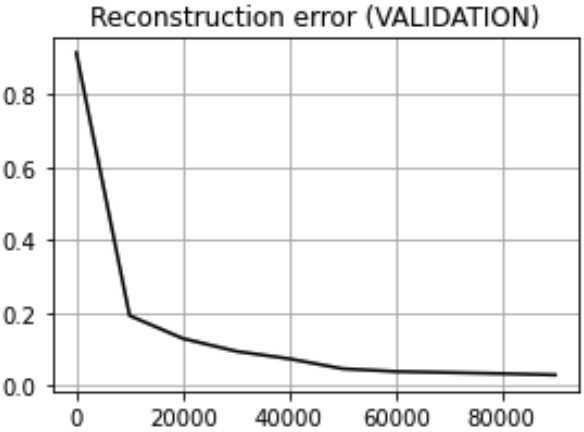
(a) 10 000 iterations, no DP applied (original reconstruction).



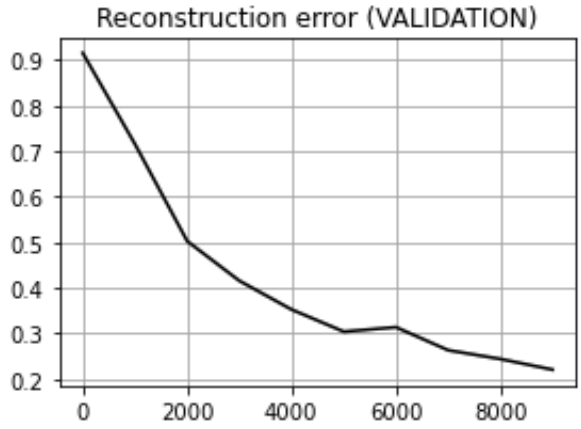
(b) 100 000 iterations, DP with $\epsilon = 0.5$.



(c) 10 000 iterations, DP with $\epsilon = 100$.



(d) 100 000 iterations, DP with $\epsilon = 10$.



(e) 10 000 iterations, DP with $\epsilon = 10$.

Figure 7: Reconstruction errors (vertical axis) against iterations (horizontal axis) for various iteration levels and DP application parameters.

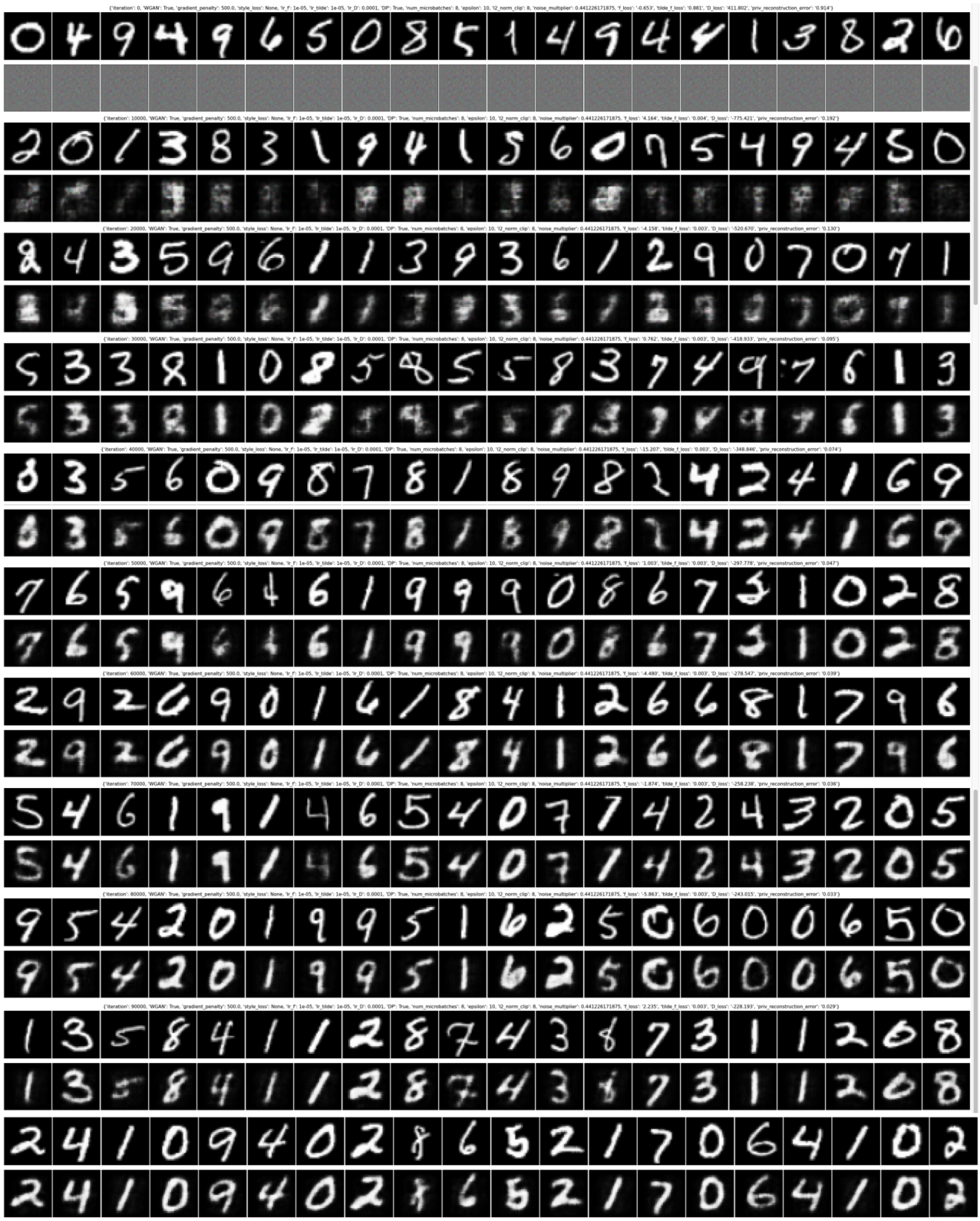


Figure 8: Reconstruction attempts for $\epsilon = 10$ and 100 000 iterations. Odd rows show the private data fed to the private local f model and the even rows below show the corresponding images reconstructed using attacker's f^{-1} model. The lower the line, the more training iterations there were.

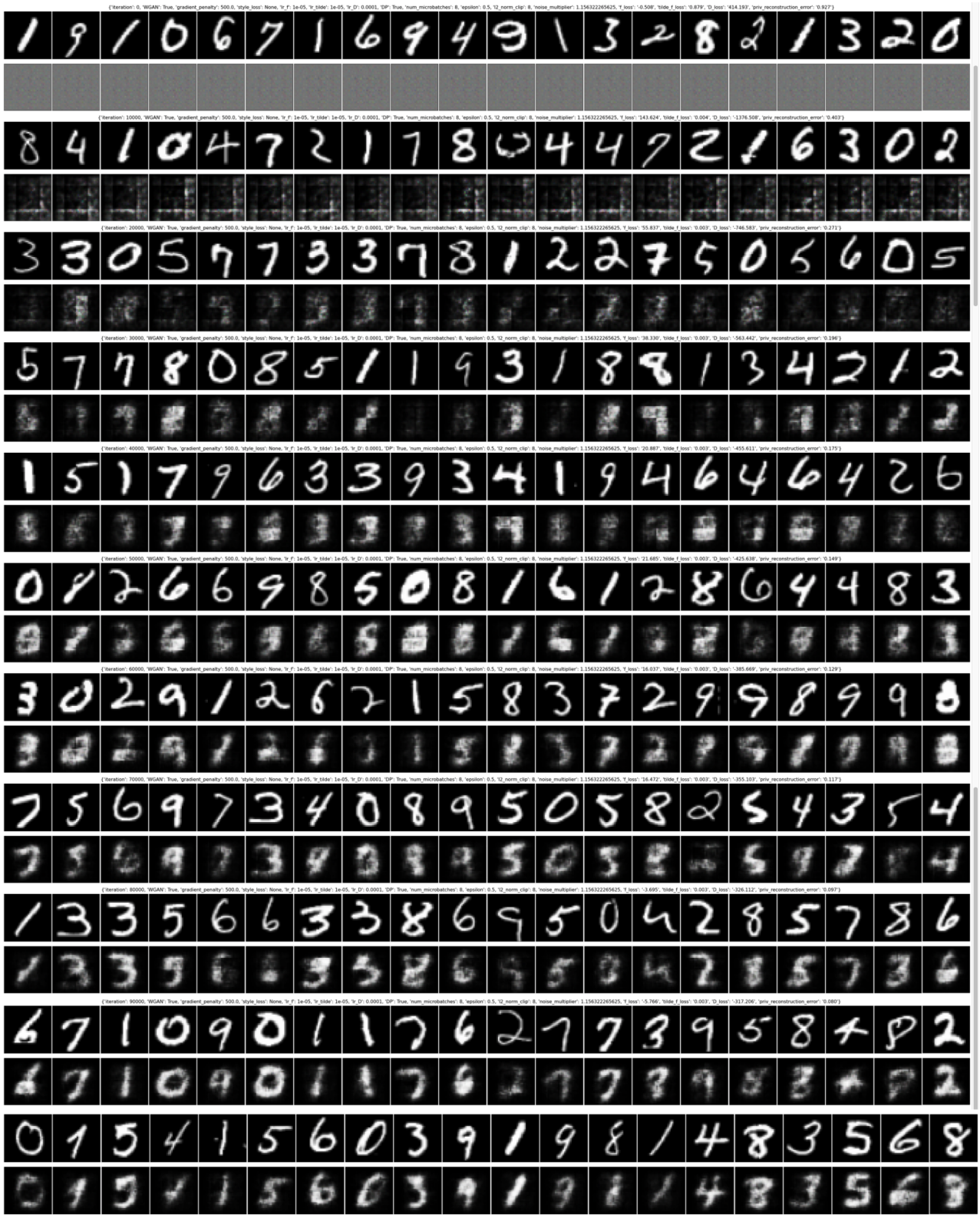
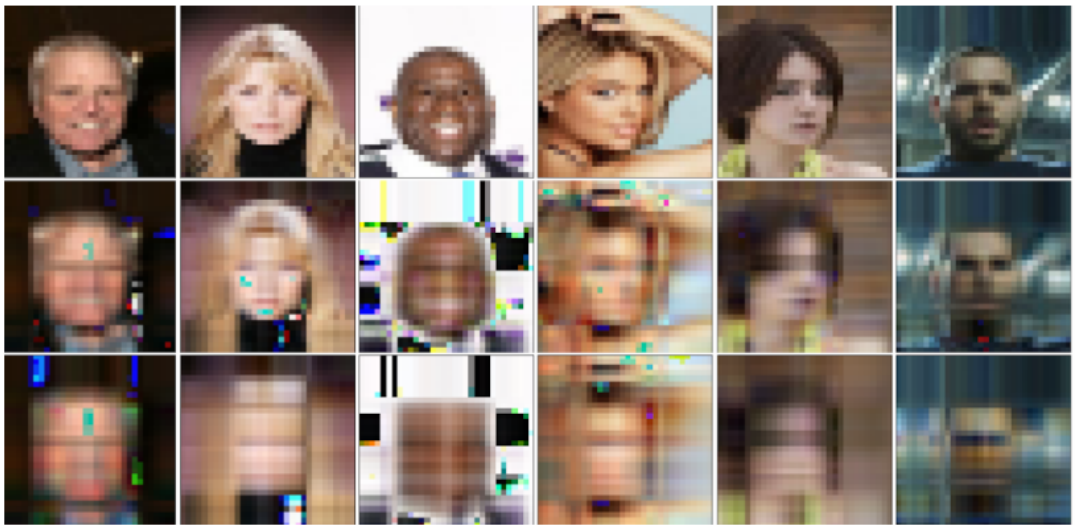
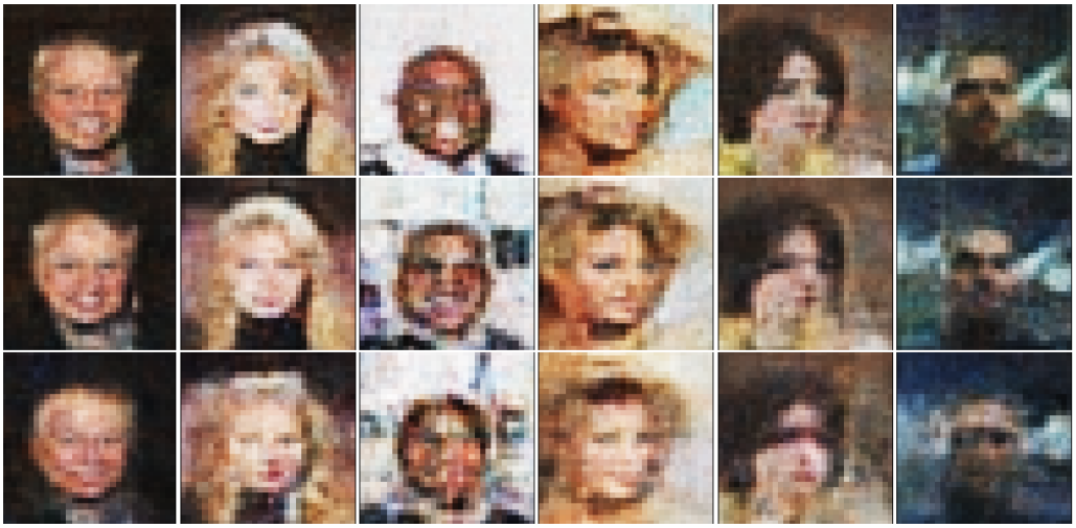


Figure 9: Reconstruction attempts for $\epsilon = 0.5$ and 100 000 iterations. Odd rows (with little titles) show the private data fed to the private local f model and the even rows below show the corresponding images reconstructed using attacker's f^{-1} model. The lower the line, the more training iterations there were.

Figure 10: A few examples per each of the ten FASHION-MNIST categories



(a) Image compression



(b) FSHA recovered images

Figure 11: PCA image compression and recovery. 1^{st} row - no compression, 2^{nd} row - principal components = 4, 3^{rd} row - principal components = 2.