Vrije Universiteit Amsterdam

Bachelor Thesis

Property Inference Attacks on Neural Networks: Influence of the Architecture and Implications of the GDPR

Author: Mathias Parisot (2618202)

1st supervisor: Dr. Dayana Spagnuelo
2nd supervisor: Dr. Balazs Pejo
2nd reader: Dr. Michael Cochez

A thesis submitted in fulfillment of the requirements for the VU Bachelor of Science degree in Computer Science

July 9, 2020

Abstract

Machine Learning is a set of techniques to train models on tasks such as classification. Classification models are trained to learn and recognize patterns in the training data to categorize it into classes. While the goal of such models is to learn important properties allowing to make correct predictions, there is also a risk that the model learns properties that are unrelated to the classification task. If the training data is sensitive, accessing such properties could lead to privacy leakage. Property Inference Attacks (PIA) are attacks on models aiming at inferring hidden properties of the training set. We focus on PIA on deep neural network classifiers and perform attacks on models trained on face images to infer whether the dataset is balanced. A complex model can solve more difficult classification tasks by learning complex patterns of the data. Because of that, we also investigate whether such models are more sensitive to PIA by studying the influence of the target classifier architecture on the success of the attack. We discuss the implication of our results on data protection laws such as the General Data Protection Regulation[1] (GDPR).

1 Introduction

Machine Learning (ML) applications have grown in popularity over the last decade. One of the most common uses of ML is to train models, called classifiers, to learn a mapping function between some input data and a set of classes. A popular way of implementing such models is to use artificial neural networks. Deep Learning (DL) is the field studying neural networks containing several hidden layers. DL models require a large amount of training data and only became popular recently with the advances in hardware and algorithms making it possible to process this data efficiently. Due to the huge amount of data generated by the Internet (communication data, social media websites, pictures, among others), corporations became interested in using ML models to automate some of their decision makings. Because of that, cloud providers such as AWS[2], Microsoft Azure[3], Google Cloud AI[4] or IBM Watson[5] started to include ML as a Service to their offer, allowing their users to train and deploy machine learning models given they provide the dataset.

With the increasing amount of data generated and its use by companies to gain insights and improve their processes, governments and authorities have been trying to give control to individuals over how their data is used. An example of such action is the legal act in the European Union concerning data protection and privacy called the General Data Protection Regulation. This regulation enforces entities processing personal data to put in place measures to implement principles (see Art. 5) such as:

- lawfulness, fairness, and transparency
- purpose limitation

[1] https://gdpr-info.eu/
[2] https://aws.amazon.com/machine-learning/
[3] https://azure.microsoft.com/en-us/services/machine-learning/
[4] https://cloud.google.com/ai-platform
[5] https://www.ibm.com/cloud/machine-learning

- data minimization
- accuracy
- storage limitation
- integrity and confidentiality

As the training of ML and DL models requires dealing with plenty of data, and because such models are becoming more and more popular, the risk of privacy leakage increases. Because of regulation such as the GDPR, companies processing data need to make sure their datasets stays confidential. However, ML models are sensitive to malicious attacks. Given a classification model, once trained, it can label a data instance to the appropriate class by learning mapping patterns between the training dataset and the set of classes. The mapping is contained within the model parameters, in the case of a neural network, it is the architecture and weight values. Therefore, an attacker knowing the parameters of a trained model, also know some information about the data it was trained on. If the data was composed of personal data, there is a potential confidentiality leak.

Attacks on ML models can be classified into four categories. Model extraction [1, 2, 3] attacks aim at inferring the behavior of the target model to create a substitute model. A more advanced feat is to create a duplicate by inferring the architecture and the parameter values. Adversarial attacks [4, 5] aim at taking advantage of the weaknesses of the classification boundary of the target model to craft data instances that are wrongly classified. Poisoning attacks [6, 7] are similar to adversarial attacks as their goal is to also influence the prediction of the target model, however, they do that by polluting the training set with malicious samples. Model inversion attacks aim at inferring information about the training data, this can be its composition or specific properties of the dataset. Depending on the goal of the attacker, we can classify model inversion attacks into three categories. Membership inference attacks [8, 9] aim at determining whether a particular data instance was used for training. This creates privacy issues especially when the instance directly maps to an identifiable individual, we can think of a medical records dataset for example. However, such attacks are not always applicable in real-life settings as the attacker needs to get or create a copy of one of the training data beforehand. Property inference attacks [10, 11, 12] (PIA) aim at reconstructing a representative of a particular class of the training set, or at inferring some hidden properties of the dataset. The first type of PIA is particularly useful when all the instances of a single class represent the same entity. For example, if a model is trained to recognize faces of individuals, the data instances of class A all represent the same individual and it is, therefore, possible to reconstruct an image of this person. While such attacks are particularly impressive for face recognition models, they only build the "average" representation of the class and do not work well if the instances are not similar. The second type of PIA, and the one we are focussing on in this paper, aims at inferring properties learned by the model that are independent of the characteristics of any class, and therefore not related to the main classification task. Such properties can be general statistics about the dataset or can reflect biases in the training set.

There have been fewer studies on PIAs: according to He et al. [13] only 4 papers were published on model inversion attacks against 10 for membership inference attacks. [13] also mention that researchers have not yet fully determined the vulnerability of neural network architectures to privacy attacks, PIAs among others. Given the current popularity of computer vision and natural language processing where deep neural networks are the most common model choice, it is reasonable to question to which extend the training datasets are at risk. When training a classifier, the goal is to maximize its performance, usually expressed as its accuracy (ratio of the number of correct classifications to the total number of predictions), recall (true positive rate), and precision (positive predictive value). When the classes are not easily separable, more complex models are required. Because complex models have more parameters and can retain more information about the training set, they usually perform better than less-complex models. Since they retain more information, they could be more sensitive to privacy attacks. In particular, we are interested in the influence of the architecture of a model on its vulnerability to PIAs.

In case of a trained Neural Network, are sensitive properties of the underlying dataset at risk due to property inference attacks, and if so, how does that risk relate to the architecture of the model?

2 Related Works

Zhang et al. [11] present a model inversion attack using Generative Adversarial Networks. They study and theoretically prove the relation between a model predictive power and its vulnerability to model inversion attacks. The influence of the predictive power of a model is a hint that more complex models, which should have greater predictive power, should also be more sensitive to model inversion attacks. However, the result of Zhang et al. [11] was not proven for PIAs. We also focus more specifically on the architecture of the target model.

Several studies [12, 14] performed PIAs in a federated learning [17] setup which allows multiple clients to train a common model without the need to share data. Only the weights and the gradients after each round of training are exchanged which can help tackle privacy issues. Melis et al. [12] managed to infer properties that hold for a subset of the training data and that are independent of the property the target model aims at predicting. Because it is performed in a federated learning setting, the attacker has access to the architecture and weights of the model. Moreover, the attack is performed during the training or, at least, requires the model updates that are exchanged between participants. The attack we focus on does not require the gradients' values after each round of training. We also target properties that are true for the whole dataset and not only a subset of it. Wang et al. [14] propose three kinds of PIAs: class sniffing, quantity inference, and whole determination. Class sniffing detects whether a training label is present within a training round, quantity inference determines how many clients have a given training label in their dataset, whole determination infers the global proportion of a specific label. All of those attacks are extracting properties related to classification labels, and therefore to the main classification task. We focus on properties that are in theory unrelated to the task of the target model.

Geiping et al. [15] study model inversion attacks and analyze the effects of the architecture of the target model on the difficulty of reconstructing input images. They investigate attacks on networks with various widths and depths and found that the width has the greatest influence on the quality of the reconstruction. Their study does not consider PIAs and is restricted to federated learning as they use the gradients' values in their attack.
Ateniese et al. [10] describe the first PIA attack using meta-classifiers, the methodology of the attack we use in our research. Their research is not focussed on the privacy leakage caused by such attack but rather on the impact of the training set properties on the model performance with the commercial benefits that it represents. Moreover, they attack models implemented via Support Vector Machines and Hidden Markov Models using a binary tree meta-classifier but do not experiment with deep neural network models. Ganju et al. [16] extend the research of Ateniese et al. [10] to neural networks and notice that a limitation of PIA performance is due to a property of fully connected networks called invariance to weights permutations within the same layer. They propose two successful strategies to reduce the impact of this property: converting a neural network to a canonical form and using a deep set architecture. They use a pre-trained network to generate an embedding which they feed as input to their target neural network. They perform the attack using the weights of the fully connected network following the pre-trained one and do not study the influence of the type of layers and the architecture of the model on the attack performance.

3 Methodology

To answer the research question, we will perform several PIAs on different target model architectures. For each architecture, we will train attack models and evaluate their performance. This section presents the attack strategy alongside with the assumptions about the target model.

3.1 Threat model

We suppose the target model is a deep neural network classifier. The training dataset of the classifier contains sensitive data according to the definition used in Article 9.1 of the GDPR. The attack is performed in a white-box setting, which means the attacker has access to the full architecture and parameter values of the model. While a white-box model may be seen as a big assumption in some scenario, it is realistic in ours as it is common in some machine learning paradigms such as Federated Learning [17] where the clients, and sometimes the server, have full knowledge of the model, and access is only restricted to the dataset. Moreover, model extraction attacks have been done to perform adversarial attacks on deep neural networks in a black-box setting. The attackers managed to create a substitute model with a similar decision boundary as the target model allowing them to achieve larger than 88% misclassification rates in real-world settings [18]. The specificities of a model extraction attack fall out of the scope of this study. In here, we assume the attacker has access to the model architecture and parameter values, independent of how this knowledge was obtained. The goal of the attacker is to infer general information

about the training dataset such as the proportion of the training data having a property unrelated to the main classification task of the model.

3.2 Attack Strategy: Property Inference Attack

In this subsection, we present the general idea of the attack we used in our experiments. We focus on PIAs which goal is to extract general information about the training dataset of the target model. The information is often presented using a property P which can be true or false. For example, if the dataset used contains images of cars, P could be: *the dataset includes 20% of images of Ferrari*, or any other brand. We can transform a PIA into a classification task where the goal is as follows: given a trained model, determine whether it was trained using a dataset presenting the given target property P. It is then possible to train a classifier to solve the previous classification task. Such a model is called a meta-classifier because the dataset on which it is trained on is composed of models which are themselves classifiers.

The PIA used in this paper is the baseline attack presented in [16]. The main idea is to use a meta-classifier to train an attack model M_a which takes as input the weights of a trained model and outputs the probability that the target property P is true for the training set of the input model. Once the attack model is trained, the attacker can give it the target model M_t as input and know whether the property P is true for the dataset used to train the target model. The main problem is then to find enough trained models to use as the training set for the attack model M_a. This is solved by using shadow models with the same architecture as the target model M_t. The attacker trains k shadow models (M_{s1} ... M_{sk}) on k datasets (D_1 ... D_k respectively) specifically crafted to contain or not the target property P. The general overview of the attack is described in Figure 1.

This PIA is the baseline attack presented in [16]. This paper is not focused on the PIA itself but rather on the behavior of the PIA performed on models with different complexities.

4 Experimental Setup

In this section, we describe the experimental settings: the dataset, the architectures of the target and attack models, and the evaluation metrics. The experiments were performed on a laptop with an Intel i7-8750H (2.20GHz) and 8GB RAM. The operating system is Ubuntu 20.04. The training of the shadow models and the attack models were both done using Pytorch and an Nvidia Quadro P600 GPU.

4.1 Datasets

CelebFaces Attributes (CelebA) [19] is a face attributes dataset containing more than 200000 images of more than 10000 celebrities taken from online. The images are labeled using 40 physical attributes such as hair color, smiling, wearing a hat. We use the dataset to train the shadow models M_s to detect whether the person has their mouth open using the *Mouth_Open* attribute. Although this might seem like an unimportant classification

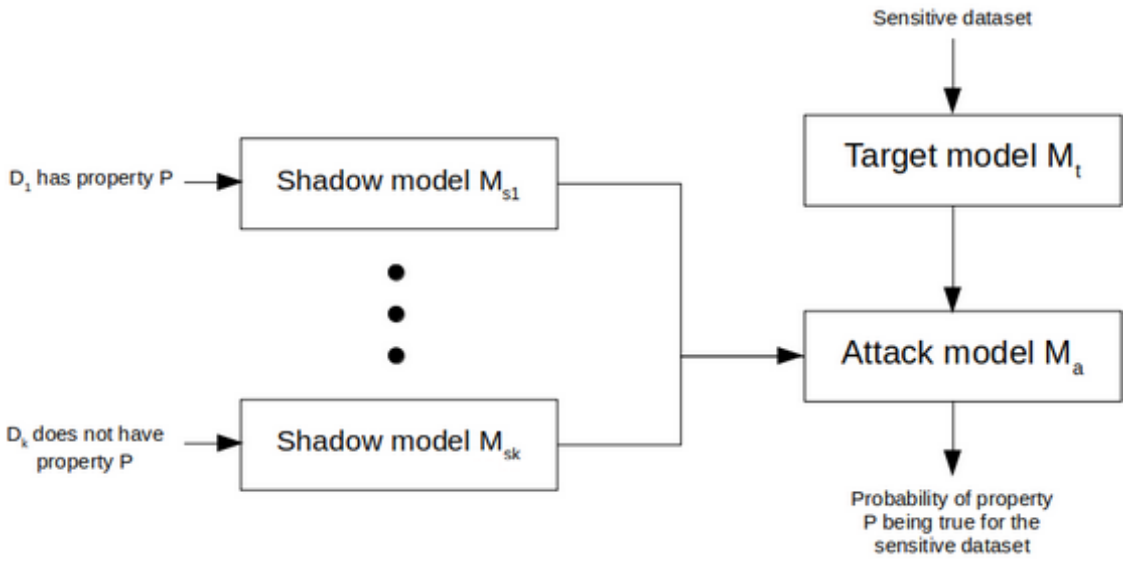

Figure 1: Property Inference Attack using a meta-classifier and a dataset of shadow models.

task, in the time of a COVID-19 pandemic, there have been questions about how technology can help the fight against the virus. Some governments are trying to track the persons who have been in contact with a contaminated subject using mobile applications[6], in some countries facial masks became compulsory in public places and companies are marketing mask-detection software[7]. Detecting whether someone has their mouth open can be associated with detecting whether someone is wearing a face mask. The property targeted by the attacker is the balance of the gender proportion of the training set using the *Male* attribute. The images are centered and resized to 64 by 64 pixels.

4.2 Shadow Models

The shadow models are trained to differentiate between images of persons with and without their mouth open. However, the goal of the attacker is to infer whether the training set of a given model was composed of an unbalanced number of images representing males. The targeted property is a biometric data and is classified as sensitive data according to the GDPR. For a given model, the property P can be formalized as follow: *P is true when the training set of the model is composed of 70% or more images containing males, otherwise, P is false.* It is important to note that P is not related to the classification task of the model and that the target model does not use, at any time during training, the gender attribute.

The shadow models M_{s_k} must have the same architecture as the targeted model, and be

[6]https://www.nature.com/articles/d41586-020-01264-1

[7]https://www.theverge.com/2020/5/7/21250357/france-masks-public-transport-mandatory-ai-surveillance-camera-software

trained to a reasonable level of accuracy for the network to retain some information about the training set. For example, all of our shadow models have at least 85% accuracy on the mouth open classification task when the baseline distribution of the dataset is 51.7%. For the attack to be effective, a large number of shadow models need to be trained. For each target model architecture, we constructed 1800 unique training sets of 2000 images from CelebA, one for each of the 1800 shadow models. The computational cost of such training is not negligible so we decide to not use all images of *CelebA* in each shadow dataset D_{s_k}, but only use 2000 images each randomly selected from the test set of *CelebA*. Out of the 1800 shadow models, 900 were trained on datasets presenting the property P and 900 without. For each dataset, the exact proportion of males was randomly taken from a uniform distribution either above or below 70 respectively.

The shadow model architectures are composed of up to 9 layers which can each be of three kinds: convolution layers, pooling layers, fully connected layers. The description of each layer is presented in Table 1. We trained 9 architectures (A_1 to A_9) which are presented in Table 2. The models take as input 64 by 64 RGB face images and output the probability of each picture representing a person with mouth open. All the networks are composed of 1 to 3 convolution layers, each followed by a max-pooling layer with a ReLU activation, and 1 to 3 fully connected layers with a ReLU activation. The shadow models were trained using the Mean Squared Error loss and the Adam optimizer with a learning rate of 0.001 during 50 epochs.

Layer	Description
Convolution 1	6 filters 5x5
Max-pool	2x2, ReLU
Convolution 2	16 filters 5x5
Max-pool	2x2, ReLU
Convolution 3	32 filters 5x5
Max-pool	2x2, ReLU
Fully-Connected 1	120 neurons, ReLU
Fully-Connected 2	84 neurons, ReLU
Fully-Connected 3	1 neuron

Table 1: Description of the different layers used in the shadow architectures.

4.3 Attack Model and Evalution

The attack model classifies shadow models as models trained on a dataset presenting the property P or not. The dataset used is composed of the 1800 shadow models and is split into training (1500 shadow models), validation (100 models), and test sets (200 models). The training algorithm is presented in Algorithm 1. The attack model is a deep neural network that was tunned using the validation set and finally evaluated on the test set. The architecture of the model is presented in Table 3. The inputs of the attack model are the flattened weights of the model it is trying to classify as having the property P or

	Conv 1	Max-pool	Conv 2	Max-pool	Conv 3	Max-pool	FC 1	FC 2	FC 3
A 1	✓	✓	✓	✓	✓	✓	✓	✓	✓
A 2	✓	✓	✓	✓	✓	✓	✓		✓
A 3	✓	✓	✓	✓	✓	✓			✓
A 4	✓	✓	✓	✓			✓	✓	✓
A 5	✓	✓	✓	✓			✓		✓
A 6	✓	✓	✓	✓					✓
A 7	✓	✓					✓	✓	✓
A 8	✓	✓					✓		✓
A 9	✓	✓							✓

Table 2: Layer-level description of each shadow architectures. The detailed description of the parameters in each layer is presented in Table 1.

not. Therefore, a shadow model architecture with a larger number of parameters induces a wider input layer for the attack model. We created 30 attack models for each shadow model architecture and present the average performance across the 30 repetitions. All the networks are composed of 2 to 4 fully connected layers with ReLU activation. The attack models were trained using the Mean Squared Error loss function and the Adam optimizer with a learning rate of 0.001 during 20 epochs. For each architecture, we compute the accuracy, recall, and precision on the test set containing 200 models.

Algorithm 1 Attack model training

procedure TRAIN_ATTACK(D, n)
  D dataset with images, n number of shadow models to train
  $D_{shadow} \leftarrow \{\}$ ▷ The dataset containing shadow models
  for $k \leftarrow 1, n$ **do**
    $D_{s_k} \leftarrow$ subset of D with or without property P
    $S_k \leftarrow train(D_{s_k})$ ▷ S_k shadow model
    $W_{s_k} \leftarrow getWeights(S_k)$ ▷ W_{s_k} list of weights of model S_k
    $D_{shadow} \leftarrow D_{shadow} \cup \{W_{s_k}\}$
  end for
  $A \leftarrow train(D_{shadow})$ ▷ S_k shadow model
  return A ▷ The attack model
end procedure

5 Results and Discussions

Figure 2 presents the accuracy of the attacks on each target model architecture. The accuracy of the attack is between 56 and 80% depending on the architecture, so the target

Name	Description
Fully-Connected 1	10 neurons, ReLU
Fully-Connected 2	10 neurons, ReLU
Fully-Connected 3	10 neurons, ReLU
Fully-Connected 4	1 neuron

Table 3: Architectures of the attack model. FC1 is the input layer and FC4 the output layer.

models do retain information that they are not intended to learn. We created as many models presenting the property P as ones not presenting it, therefore, the expected baseline is 50% accuracy. Most of the architectures have less than 67% attack accuracy so the actual real-world threat is low. In a real-world setting, the attackers have only one target model to attack, therefore, they must be close to certain (close to 100% attack accuracy) that the model present the property P. However, we did not spend time tuning the meta-classifier architecture and we believe significant improvement could be made at this level. Moreover, Ganju et al. [16] have shown that it is possible to significantly increase the accuracy of this attack using representations that are invariant to node permutations. They managed to get almost perfect accuracy making the attack performable in a real-world setting. Whether using such representations on convolutional neural networks results in similar attack accuracies is still to be determined. One of the goals of this study was to establish whether there is a relationship between a model complexity, which we define by its number of parameters, and its sensibility to PIAs, defined as the accuracy of the attack. According to Figure 3, there does not seem to be a relationship.

We performed PIAs on distinctive neural network architectures presenting convolution layers. Because convolution layers and fully connected ones play different roles in a convolutional neural network, we studied whether the type of layers used has an impact on the accuracy of the attack. To do that, we conducted three PIAs on each of the architectures presented in Table 2. The first PIA uses all the weights of the shadow model, the second PIA only uses the weights of the convolution layers, and the third one only the weights of the fully connected layers. Figure 4 presents the accuracy of the three attacks. For most shadow model architecture, the PIA using only the fully connected weights performs as well, and sometimes better, as the PIA using the weights from both types of layers. The information leaked by a convolutional neural network seems to be contained in the fully connected part of the network.

Our goal was to determine if the dataset used to train a CNN model presented a property P, in our case whether the dataset was unbalanced. This attack can be performed using other properties P as well. Theoretically, any property P could be inferred from the model as long as the attackers can generate datasets with and without it. We could, for example, infer whether the training set was normalized, or whether it contains a picture of a given person. In that case, when the instances of the same class represent the same entity (same person), a PIA can be transformed into a membership inference attack with the difference that it is not focused on a specific data instance but on whether the whole dataset contains

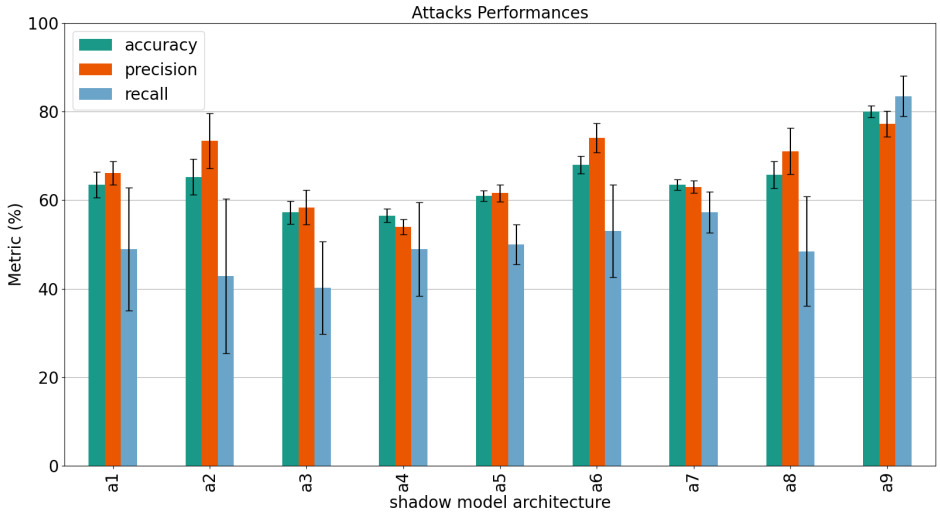

Figure 2: Accuracy, precision (positive predictive value), and recall (true positive rate) of the attacks on each architecture. Each bar corresponds to the median of the 30 attacks, and the error bars are ± the standard deviation.

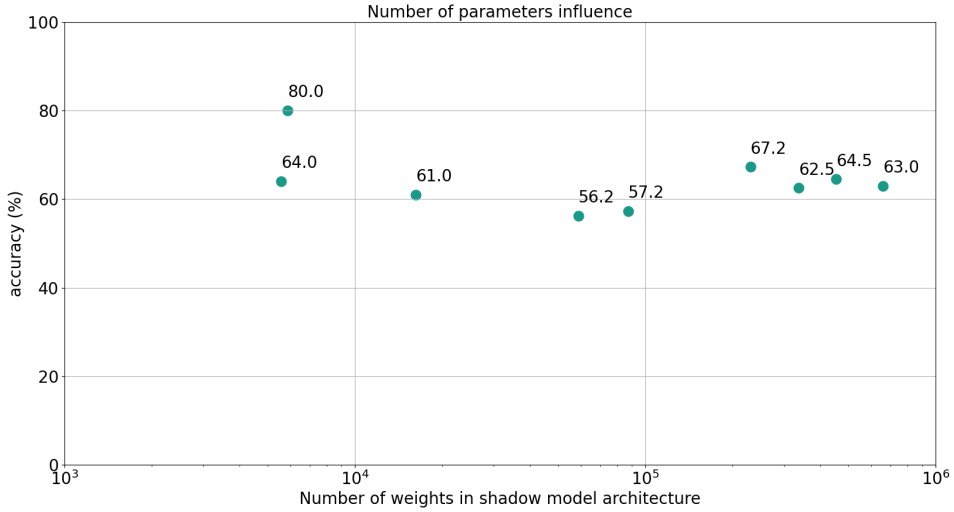

Figure 3: Influence of the complexity of the target model (express as the number of parameters) on the accuracy of the attacks on each architecture. Each dot corresponds to the median of the 30 attacks.

a particular class of image (a particular person). In a real-world setting, this can turn out to be a more important threat. A membership inference attack on portray images risks to fail when the attackers do not possess the exact data instance that is present in the dataset. It is unlikely that the attackers own the same image, however, it is more likely that they

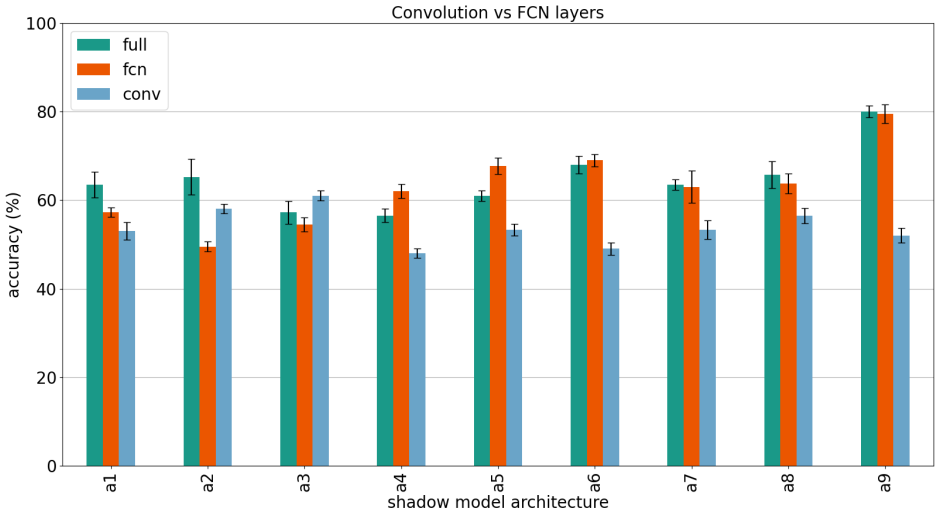

Figure 4: Comparaison of the accuracies of the attacks on each architecture using all the target model weights (full), only the fully connected layers weights (fcn), and only the convolution layers weights (conv). Each bar corresponds to the median of the 30 attacks, and the error bars are $\pm$ the standard deviation.

possess a dataset of pictures of their target person. They can create shadow datasets with and without pictures of their target, use them to train shadow models, and perform a PIA.

Article 5.1 of the GDPR mentions that the processing of personal data shall be limited to what is necessary in relation to the purposes for which they are processed. This property is called data minimization, and it is broken if a data processor shares a model that was trained using personal data. With the PIA discussed, we highlight the surplus of information given to the models and the risk of inference by an attacker. Data processors should consider that information can be extracted from a model and make sure that they train models on data that is only relevant to the classification task. It is the role of data processors to inform users when they give their consent for personal data processing, that there is a risk of privacy leakage.

6 Conclusion

There is a risk of leaking general information about the training set when training a neural network. In this paper, we further studied a PIA methodology by performing several attacks on different deep neural network architectures. We confirmed that convolutional neural networks are also sensible to PIA, and we emphasized the importance of fully connected layers in the information leakage. We did not observe a relation between the number of weights the target model contains and its sensibility to PIA. Our results stress the dangers of sharing models trained on sensitive dataset and question the usage of fed-

erated learning when dealing with personal data.

References

[1] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In 25th {USENIX} Security Symposium ({USENIX} Security 16), pages 601–618, 2016.

[2] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia conference on computer and communications security, pages 506–519, 2017.

[3] Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In 2018 IEEE Symposium on Security and Privacy (SP), pages 36–52. IEEE, 2018.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

[5] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, 2013.

[6] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[7] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In 2018 IEEE Symposium on Security and Privacy (SP), pages 19–35. IEEE, 2018.

[8] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Towards demystifying membership inference attacks. arXiv preprint arXiv:1807.09173, 2018.

[9] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pages 603–618, 2017.

[10] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. International Journal of Security and Networks, 10(3):137–150, 2015. ISSN 17478413. doi: 10.1504/IJSN.2015.071829.

[11] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. 2019. URL http://arxiv.org/abs/1911.07135.

[12] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. Proceedings - IEEE Symposium on Security and Privacy, 2019-May:691–706, 2019. ISSN 10816011. doi: 10.1109/SP.2019.00029.

[13] Yingzhe He, Guozhu Meng, Kai Chen, Xingbo Hu, and Jinwen He. Towards Privacy and Security of Deep Learning Systems: A Survey. pages 1–23, 2019. URL http://arxiv.org/abs/1911.12562.

[14] Lixu Wang, Shichao Xu, Xiao Wang, and Qi Zhu. Eavesdrop the Composition Proportion of Training Labels in Federated Learning. 2019. URL http://arxiv.org/abs/1910.06044.

[15] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting Gradients – How easy is it to break privacy in federated learning? (1), 2020. URL http://arxiv.org/abs/2003.14053.

[16] Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. Proceedings of the ACM Conference on Computer and Communications Security, pages 619–633, 2018. ISSN 15437221. doi: 10.1145/3243734.3243834.

[17] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, pages 1310–1321, 2015.

[18] Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. CoRR, abs/1602.02697, 2016. URL http://arxiv.org/abs/1602.02697.

[19] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), December 2015.