A Covert Channel Detector for Machine Learning Models

John Stein
Indiana University
Bloomington, Indiana
jodstein@iu.edu

ABSTRACT

It has been shown that malicious model training code can augment sensitive training data with specially-crafted samples to enable later exfiltration of the sensitive data by an attacker who has black-box access to the completed model, but no direct access to the sensitive training data. [4] This is known as a Covert Channel for machine learning models. It has also been shown an attacker can infer whether a specific sample was included in the training set for a black-box model if the attacker has knowledge of the model's structure or can generate models with similar structure. [3] This is known as Membership Inference on machine learning models. We demonstrate that a variation of the Membership Inference 'attack' can be used to detect whether a given machine learning model may have a covert channel. We review some performance factors of the detector, and discuss how performance is directly related to model capacity and sparsity of the sensitive and maliciously-crafted data.

KEYWORDS

machine learning, models, covert channels, detector

1 INTRODUCTION

The relevant threat scenario we will discuss involves four parties: the machine learning (ML) provider who developed the ML software, the client who owns the data and needs the model, an outsider, and an optional ML marketplace which provides secure transaction services between clients and ML providers.

A covert channel attack occurs when the client obtains black-box use of a malicious ML provider's software (via a ML marketplace or download of the executable(s)) and generates a model from the client's own private training and test data. Within the ML software, the private training data is encoded into the model for future ex-filtration by the malicious ML provider (the attacker). In the 'Capacity Abuse' version of this attack, the private training data is augmented with malicious samples which have known (to the attacker) feature values mapped to labels that represent an encoding of the private training data, and the model is trained in the normal way. If the attacker has black-box access to the model at some future point in time, they can present the known feature values to the model and obtain an encoding of the sensitive data as the prediction result. [4]

A membership inference attack occurs when an outsider gains black-box access to a target ML model as well as either (a) information about the model's structure or (b) access to the same ML service, resulting in the ability to generate models with similar structure using known training data. The attacker generates several shadow models using known training data, then tests the shadow models with data that was included in their associated training set and data that was not included. The data, the models' response, and a label of 'in' or 'out' is then used to train an attack model to predict whether a model's response to a given input is characteristic to that of a model being tested on its training data, or a model being tested on unseen data. The attacker can now present any data to the target model, then present the model's response to the attack model to learn whether that data was included in the training set for the target model. [3]

Our contributions. We observe that in the capacity abuse attack, use of sparse data as the known feature values in the malicious samples improves the model's performance for both the ostensible classification task as well as the malicious ex-filtration task. Essentially, when the malicious data is sparse, it allows the model to generalize in the (dimensional) region of data occupied by the client's training data, while allowing the model to over-fit to those malicious data specified by the attacker. We use a variation of membership inference to classify a model's response as being that of a model which was trained using sparse data, or one which was not. This technique can be used to detect covert channels in ML models that were created in this way.

2 METHODOLOGY

2.1 Computing Environment

The work described in this paper was performed using GPU compute nodes on the Big Red II high-performance computing system. The code was run within an Anaconda virtual environment using the following packages:

- python 2.7.13
- tensorflow-gpu 1.1.0
- cudnn 5.1
- keras 2.0.5
- pypng 0.0.16
- pydot 1.0.28

2.2 Client Task & Data Set

We chose a photo classification task, using CIFAR-10 data [2], as the task the client wishes to perform. CIFAR-10 was chosen because it is a benchmark data set which was used in the previous papers which this work extends [3, 4] and therefore provides a basis for comparison.

2.3 Benign & Malicious Models

For the benign classification model, we used a 34-layer Residual Network based largely on previous work from Kaiming He et al [1].

In my implementation, if the number of categories isn't a power of two, then there will be some categories that are never used as labels by the malicious encoder. This is a red herring for a malicious model classifier, but is easily circumvented by either choosing by an

encoding number base that is equal to the number of categories (if the attacker knows the number of categories a priori) or else adding a pseudo-random number to the malicious labels (then modulo the number of categories) so that each category gets equivalent representation.

3 CONCLUSIONS

ACKNOWLEDGMENTS

The authors acknowledge the Indiana University Pervasive Technology Institute for providing Big Red II resources that have contributed to the research results reported within this paper [5].

This research was supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute, and in part by the Indiana METACyt Initiative. The Indiana METACyt Initiative at IU was also supported in part by Lilly Endowment, Inc.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *CoRR* abs/1512.03385 (2015). arXiv:1512.03385 http://arxiv.org/abs/1512.03385

[2] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report. University of Toronto.

[3] Reza Shokri, Marco Stronati, and Vitaly Shmatikov. 2016. Membership Inference Attacks against Machine Learning Models. *CoRR* abs/1610.05820 (2016). arXiv:1610.05820 http://arxiv.org/abs/1610.05820

[4] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. 2017. Machine Learning Models that Remember Too Much. *CoRR* abs/1709.07886 (2017). arXiv:1709.07886 http://arxiv.org/abs/1709.07886

[5] C.A. Stewart, V. Welch, B. Plale, G. Fox, M. Pierce, and T. Sterling. 2018. Indiana University Pervasive Technology Institute. https://doi.org/10.5967/K8G44NGB