Fastest Heist in the West: Model Extraction Attacks

Daniel Richards Ravi Arputharaj
KTH Royal Institute of Technology
drra@kth.se

Adhithyan Kalaivanan
KTH Royal Institute of Technology
adhkal@kth.se

Aishwarya Ganesan
KTH Royal Institute of Technology
aganesan@kth.se

Abstract

Large machine learning models are valuable assets with high costs associated with training. We explore and demonstrate model extraction attacks where an adversary with just input - output access to a black box model could "steal" it. We study the required query budget to mount a successful attack and methods to reduce it. We ask if the adversary could gain significant advantages with further knowledge of the original model and lastly, demonstrate how well privacy attacks on an extracted model transfer to the original. We extract models that match up to 82.88% of the victim's predictions on CIFAR-10 dataset with $25k$ queries.

1 Introduction

With ML-as-a-service (MLaaS) gaining popularity as a commercially viable business, the machine learning models deployed by a company are critical financial assets that provide a competitive advantage. Model extraction attacks (Rigaki and Garcia [1]) are a class of privacy attacks where an adversary who can query the model and access its outputs, can "steal" the model, i.e., create a surrogate that replicates the original's output characteristics, with no additional information such as the model architecture and training procedure. We shall refer to the adversary's model as the attacker and the deployed black box as the victim. So, the relevant metric we care about is the attacker accuracy assuming the true label is one predicted by the victim on samples not seen during training, which is simply referred to as accuracy unless mentioned otherwise. We study how viable are extraction attacks considering models trained on CIFAR-10 and CIFAR-100 datasets by observing the query budget required to reach a desired accuracy. We then study the effects of increased adversarial access to the victim such as access to training data, model architecture and the complete output layer instead of just predicted labels. Finally, we demonstrate the capabilities an adversary gains after having extracted the victim. Specifically, we explore if white box attacks such as adversarial examples and membership inference can be mounted on the victim using the attacker as a surrogate. We briefly investigate completely data free extraction methods, the scope of which is limited due to increased computational demands. In Section 2, methods used to answer the above and brief note on experimental setup is presented. After which we discuss in detail the analysis of the experimental outcomes in Section 3. Finally, in Section 4, we provide our conclusion with a few interesting open questions.

2 Methods

For majority of our experiments, we take an open-source implementation of RestNet-50 model pretrained on CIFAR-10 (Phan [2]) as our victim and also trained a ResNet-50 model on CIFAR-100 from scratch. The attacker is built from scratch using PyTorch library, with a ResNet-34 like

architecture and trained on labels obtained by querying the victim. We use a cross entropy loss and perform gradient descent using the Adam optimiser with cyclic learning rates.

We begin by assuming the adversary has access to the training data and focus on the query budget required to extract the victim. Our baseline is built by querying the victim with a random subset of the data, to train the attacker. We aim to decrease the query budget by identifying the "coreset" (Coleman et al. [3]), a representative subset without affecting performance. This is done by training a small ResNet-18 model for 20 epochs and sorting the images by the output entropy of the classifier. The high entropy images are sliced to form our coreset. The key advantage is the inexpensive training of this step.

We explore increased levels of adversarial access and ask whether it provides an advantage for model extraction. We check if it is important for the attacker to match the victim's architecture or number of parameters. We do this by choosing different architecture and victim model configurations between ResNet-34, ResNet-50, and VGG19-BN. We also investigate what if the victim provides top-k labels with confidences, instead of just the most likely label. This may improve the attacker, as it gains further insight into the victim's predictions. We then remove the adversary's access to training data and query the victim with out-of-distribution (OOD) samples. The motive is to verify if actual training data is essential, or any public dataset would do. We build our own OOD dataset with same labels as CIFAR by both sampling from ImageNet and scrapping the web, then resizing the images to be 32×32. These are manually inspected to ensure the correctness of labels. Lastly, we briefly discuss a scenario where it is restrictive to obtain any data that matches the training data of the victim. Here, we investigate data-free model extraction where a generator is introduced which is trained to produce and query images that are "valuable" for the attacker. It is important to note that the trade-off here is to have an exceptionally large query budget, in our case $20M$.

We finally evaluate what kind of attacks can be mounted by the adversary using the attacker model. We choose two which would have required white box access to the model, but here the attacker acts as a surrogate of the victim. What we are interested in, is to see if the attack transfers well to the victim. We generate adversarial examples for images correctly classified by the attacker using the Fast Gradient Sign Method (FSGM), for its simplicity (Goodfellow et al. [4]). If the attacker misclassifies an adversarial example, we check if the victim also misclassifies it and whether does so with the same label. If the examples transfer well, this provides a powerful tool to break the victim in predictable ways. We also perform a membership inference attack, a method to evaluate if an input is part of the training data, using entropy of the softmax outputs. Note that an adversary would not have direct access of the victim's complete softmax output, so once again, the attacker acts as a surrogate. The key idea is that a classifier is much more "confident" on its predictions about seen samples than unseen ones. So, assuming the adversary has access to a portion of victim's training data, they can query the attacker with both these seen and other unseen images. From the observed entropy distribution, it would be possible to obtain a threshold-based classifier which infers the membership of an image.

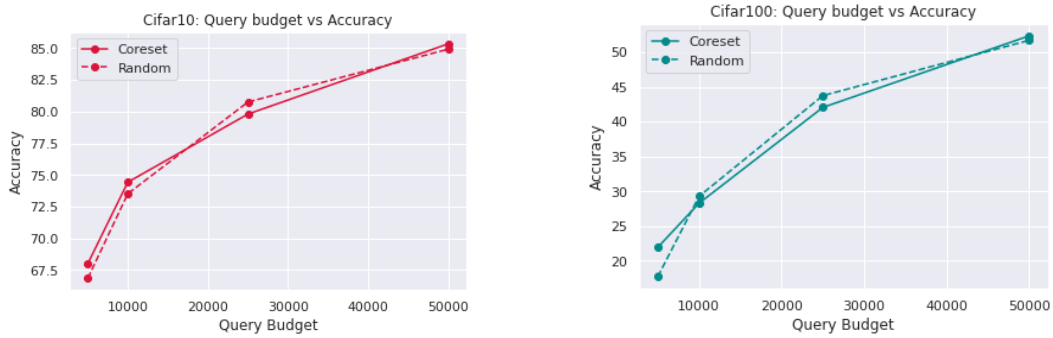
The complete information on model architectures, query budget, query sampling algorithm, level of output access and others used for each of above experiments is listed in the Table 3 available in Appendix A. The source code can be found at [5].

3 Results

3.1 Query Budget and Sampling Algorithm

We identify the "coreset" of both CIFAR-10 and CIFAR-100 as described above. Using pretrained models as victims, we query and train the attackers with different query budgets. In one case, we query a random subset of training data and in the other, slice from the ordered coreset list. The accuracy reached by the attacker with different query budget on both these datasets is shown in Figure 1.

We notice that coreset provides an exceedingly small advantage over random at a low query budget, and this effect quickly vanishes. We also observe that the gap is larger for CIFAR-100, as the number of classes increases, it becomes a lot more crucial to sample representative images than random ones. We also notice the attacker accuracy is a lot worse in CIFAR-100 compared to CIFAR-10. It could be that large number of classes require higher query budget to get enough samples for each class. But it



(a) (b)

Figure 1: Attacker accuracy over different query budgets and sampling algorithm

is also worth noting that the CIFAR-100 victim has a test accuracy of 70.0% compared of 93.6% of the CIFAR-10 victim. So, the labels we obtain from the CIFAR-100 victim are much noisier which in turn significantly degrades the attacker.

3.2 Increased Victim Access

We fix the sampling algorithm as "coreset", the query budget as $25k$ and choose the CIFAR-10 data for the following experiments. To evaluate if it is essential for the attacker to match the victim architecture, we perform model extraction with different combinations of victim and model architecture. The accuracy reached by the attackers are presented in Table 1a. We observe that the ResNet-34 attacker extracts all the victims well. This is in line with the model distillation literature where a small student is enough to extract knowledge from a large teacher. We also note that architectural differences are not factor of concern as ResNet-34 manages to extract the VGG19-BN victim well.

We now freeze the victim as ResNet-50 and attacker as ResNet-34 and vary the access to the victim's output. Three different cases are considered - just the top label, top 3 labels with confidences, confidences on all the outputs. Here "confidence" just refers to the softmax outputs. While cross entropy loss is used to train the attacker in the label case, this does not work when the victim provides confidences. So, with the aim of matching the distribution, we use the KL divergence loss for these two cases. The results of these runs are summarised in Table 1b. Unsurprisingly, the attacker extracts better when provided with the confidences. With images that are ambiguous, the victim now provides information on what other labels it believes the image to be. Also note how going from top-3 values to all values provides no advantage. This is because the softmax layer diminishes all the lower logits, so they add little to no information.

Table 1: Results for increased victim access

(a) Accuracy for varying Attacker-Victim models

Victim Attacker	**ResNet34**	**ResNet50**	**Vgg19_BN**
ResNet34	80.27	79.8	81.34
ResNet50	79.38	79.42	77

(b) Accuracy for varying output access

Output Access	**Accuracy**
Labels	79.8
Top 3 softmax	82.88
All softmax	82.15

If the adversary does not have access of any part of the training data, they are forced to query the victim with similar images which may lie out of distribution of the training data. To simulate this case, we build our own OOD dataset, with same labels as CIFAR-10 as described in the above section. When the victim is queried with $10k$ and $15k$ images from the OOD dataset, the attacker accuracy reaches 56.83% and 59.43% respectively. While this is significantly lower than when querying with train data, the attacker performs vastly better than random. In cases where it is not feasible to construct even such a similar dataset, we explore implementing a generative model-based approach (Truong et al. [6]). The same setup is used with ResNet-50 as victim and ResNet-34 as attacker. After

performing $20M$ queries, we reach a attacker accuracy of 30%. This method is not explored further as the query budget to reach reasonable accuracy is extremely high.

3.3 Transfer of Privacy Attacks

We finally investigate what are the capabilities an adversary gains after having extracted the attacker model. Specifically, we evaluate if adversarial examples of the attacker transfer to the victim. The results are summarised in Table 2. We observe that very few of the adversarial images transfer to the victim, and most of them are not labelled with the same incorrect label as the attacker. This is very promising for the victim as the adversary may not be able to "break" the victim in a predictable manner. But one should be cautious before concluding the victim to be robust, as this may just be due to the low dimensionality of CIFAR images.

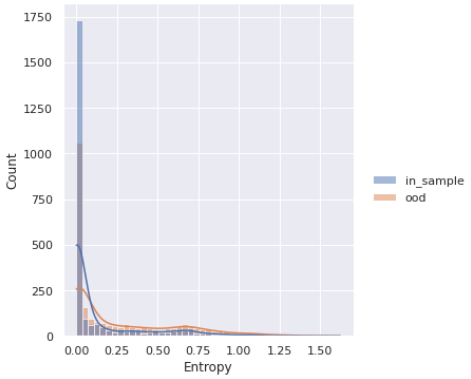
Table 2: Adversarial Transfer with $\epsilon = 0.01$

	Percentage of test images
Correctly classified by attacker	79.32%
Adversarial for attacker	47.84%
Adversarial for victim	3.47%
Same adversarial label	1.94%

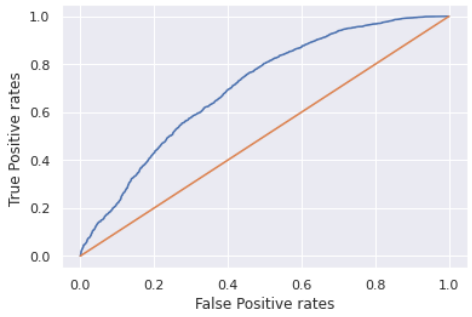
(a) Real label: Cat

(b) Predicted label: Dog

Figure 2: Adversarial Examples from Cifar-10 dataset



(a) Entropy distribution of the images

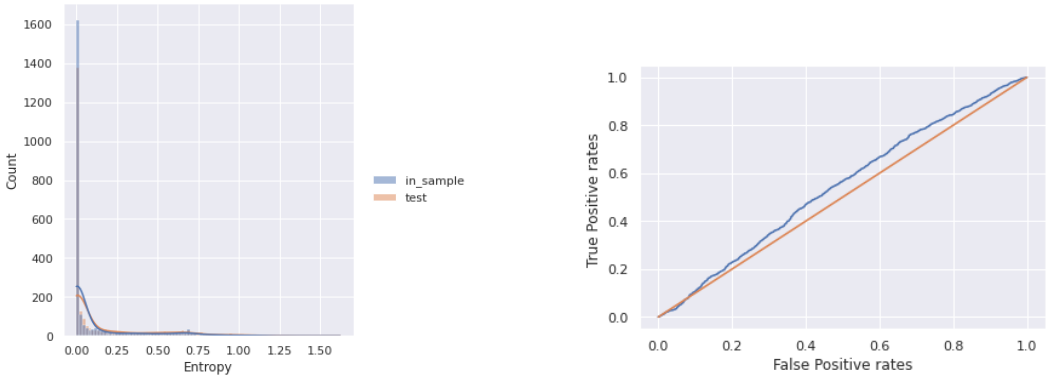


(b) ROC-AUC plot

Figure 3: Membership Inference Attack with OOD

We then run a membership inference attack, by building a simple threshold-based classifier from the attacker's output entropy of 2500 seen and 2500 unseen images by the victim. When the unseen images are completely out-of-distribution, the ROC-AUC curve as in Figure 3, shows that we can obtain a classifier that is much better than random at inferring the membership of an image. But this task only requires finding whether the image is in distribution, not if it is a training sample. So, we

also conduct a run where the unseen images are from CIFAR-10 test dataset, which are in distribution but not used in training the victim. From Figure 4, we see that this task is much harder, and the classifier is nearly random. This could be due to the ease of CIFAR-10 dataset but does not indicate complete robustness of the victim. Being able to classify if an image is OOD may still be enough to infer the membership with reasonable confidence.



(a) Entropy distribution of the images

(b) ROC-AUC plot

Figure 4: Membership Inference Attack with Test

4 Summary and Conclusions

We find that an adversary can successfully extract the victim with significantly lower number of queries than the complete training set size, even in cases where they do not have complete output access of the victim. Further we find that it is not essential for the attacker to have knowledge of the victim's architecture, and any domain appropriate model architecture with sufficient capacity will do. The obtained attacker is better than random even when access to parts of the training data is removed by trading off with a higher query budget, and the performance significantly increases with further access to victim's outputs. But we also make few observations which are reassuring for the victim, where adversarial examples do not transfer well enough for low dimensional inputs. The victim is also prone to membership inference attacks but only in cases were detecting out-of-distribution is enough to reasonably infer membership. Further work can explore if the generator used in model free extraction produces images that are increasingly higher entropy for a classifier as it trains, thus unifying the idea behind coreset identification.

References

[1] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *arXiv preprint arXiv:2007.07646*, 2020.

[2] Huy Phan. huyvnphan/pytorch_cifar10, January 2021. URL `https://doi.org/10.5281/zenodo.4431043`.

[3] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019.

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

[5] Daniel Richards Ravi Arputharaj, Adhithyan Kalaivanan, and Aishwarya Ganesan. Deep Learning - Model Extraction, 05 2022. URL `https://github.com/the-nihilist-ninja/dl-model-extraction`.

[6] Jean-Baptiste Truong, Pratyush Maini, Robert J. Walls, and Nicolas Papernot. Data-free model extraction. *CoRR*, abs/2011.14779, 2020. URL `https://arxiv.org/abs/2011.14779`.

A Appendix

A.1 Experiment Test Cases

Table 3: Details of experiment test cases

Test cases	Dataset	Victim	Attacker	Output Access	Query Algorithm	Query Budget
Case 1	Cifar-10 Cifar-100	ResNet50	ResNet34	Labels	Random Coreset	$5k$ $10k$ $25k$ $50k$
Case 2	Cifar-10	ResNet34 ResNet50 VGG19_BN	ResNet34 ResNet50 VGG19_BN	Labels	Coreset	$25k$
Case 3	Cifar-10	ResNet50	ResNet34	Labels Top 3 softmax All softmax	Coreset	$25k$
Case 4	Cifar-10 Custom Dataset	ResNet50	ResNet34	Labels	Coreset	$10k$ $15k$
Case 5	Cifar-10	ResNet50	ResNet34	Labels	Coreset	$25k$
Case 6	Cifar-10	ResNet50	ResNet34	Labels	Coreset	$25k$
Case 7	Generated	ResNet50	ResNet34	Labels	-	$20M$

A.2 Parameters used for Attacker Training

Table 4: Parameters for Attacker Training

Parameters	
Learning cycles	2
Learning step size	1000
Minimum learning rate	10^{-5}
Maximum learning rate	10^{-1}
Batch size	500