Model Extraction Attacks Revisited

Jiacheng Liang
Stony Brook University
New York, USA
ljcpro@outlook.com

Ren Pang
Penn State University
State College, USA
rbp5354@psu.edu

Changjiang Li
Stony Brook University
New York, USA
meet.cjli@gmail.com

Ting Wang
Stony Brook University
New York, USA
inbox.ting@gmail.com

ABSTRACT

Model extraction (ME) attacks represent one major threat to Machine-Learning-as-a-Service (MLaaS) platforms by "stealing" the functionality of confidential machine-learning models through querying black-box APIs. Over seven years have passed since ME attacks were first conceptualized in the seminal work [75]. During this period, substantial advances have been made in both ME attacks and MLaaS platforms, raising the intriguing question: *How has the vulnerability of MLaaS platforms to ME attacks been evolving?*

In this work, we conduct an in-depth study to answer this critical question. Specifically, we characterize the vulnerability of current, mainstream MLaaS platforms to ME attacks from multiple perspectives including attack strategies, learning techniques, surrogate-model design, and benchmark tasks. Many of our findings challenge previously reported results, suggesting emerging patterns of ME vulnerability. Further, by analyzing the vulnerability of the same MLaaS platforms using historical datasets from the past four years, we retrospectively characterize the evolution of ME vulnerability over time, leading to a set of interesting findings. Finally, we make suggestions about improving the current practice of MLaaS in terms of attack robustness. Our study sheds light on the current state of ME vulnerability in the wild and points to several promising directions for future research.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; • **Security and privacy**;

KEYWORDS

Model Extraction Attacks, Model Extraction Benchmark, Model Extraction Vulnerability Evolution

ACM Reference Format:
Jiacheng Liang, Ren Pang, Changjiang Li, and Ting Wang. 2024. Model Extraction Attacks Revisited. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '24), July 1–5, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3634737.3657002

ASIA CCS '24, July 1–5, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0482-6/24/07
https://doi.org/10.1145/3634737.3657002

1 INTRODUCTION

The remarkable advances in machine learning (ML) technologies in recent years [14, 43, 44, 66, 79, 83, 85] have spurred the expansion of Machine-Learning-as-a-Service (MLaaS) [40, 71], which meets the increasing need for ML capabilities among users who might not possess the requisite expertise or infrastructure. MLaaS platforms offer publicly accessible APIs [18, 40], enabling users to interface with backend ML models and systems seamlessly. Users are often charged on a per-query basis, making advanced ML capabilities both accessible and affordable. Many IT giants (*e.g.*, Google, Amazon, Microsoft, and Face^{++}) have unveiled their MLaaS platforms.

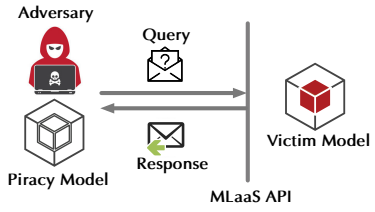


Figure 1: Model extraction attacks.

However, there exists an inherent conflict between the proprietary nature of ML models and the public accessibility of MLaaS APIs, leading to a range of security concerns [25, 38, 39, 41, 45, 46, 48, 49, 52, 53]. One of the most prominent concerns is the threat of model extraction (ME) attacks. As illustrated in Figure 1, such attacks construct a piracy model functionally equivalent or similar to the backend model of MLaaS, via querying the black-box APIs with carefully crafted inputs. Since its conceptualization in [75], a stream of work has been proposed to improve the efficacy of ME attacks [8, 10, 29, 31, 33, 57, 59–65, 71, 75, 76, 82].

Despite the plethora of prior work, our understanding of the vulnerability of real-world MLaaS to ME attacks remains limited. With the exception of [82], most of these studies simulate ME attacks in controlled environments, which may differ significantly from real-world MLaaS scenarios. For instance, some studies (*e.g.*, [33]) assume the adversary possesses prior knowledge about the victim model's architecture, while others (*e.g.*, [76]) create synthetic data to extract models, but MLaaS will initially reject the random noise images, impeding progress. Moreover, since the conceptualization

of ME attacks, substantial advances have been made in ME attacks, ML techniques, and MLaaS platforms. It is unclear whether the conclusions drawn in prior work still hold in this rapidly evolving landscape. Specifically, we have the following intriguing questions:

RQ1 - How vulnerable are today's real-world MLaaS APIs with respect to ME attacks?

RQ2 - How might the adversary exploit such vulnerability effectively (*e.g.*, query-cost reduction)?

RQ3 - How has the vulnerability of MLaaS platforms been evolving in the past years?

Our work – To answer these questions, we design, implement, and evaluate MeBench, the first open-source platform for evaluating the ME vulnerability of MLaaS APIs in a unified and holistic manner. Currently, MeBench has integrated 4 representative ME attacks, 4 attack performance metrics, as well as a suite of 11 piracy models and 6 benchmark datasets. Further, MeBench has implemented a rich set of analysis tools for characterizing the vulnerability, including comparing vulnerability across different APIs, measuring attack transferability, and tracing vulnerability evolution. Our findings can be summarized as follows.

– Leveraging MeBench, we conduct an empirical study on leading MLaaS platforms (*i.e.*, Amazon, Microsoft, Face^{++}, and Google) in the tasks of facial emotion recognition (FER) and natural language understanding (NLU). We show that today's real-world MLaaS APIs still exhibit significant vulnerability to ME attacks, while the characteristics of vulnerability vary greatly across different platforms and tasks.

– We further examine the influential factors on the performance of ME attacks, including optimizers, training regimes, model architectures, and advanced attack strategies. Our evaluation leads to a set of interesting findings, many of which challenge the conclusions in prior work, as summarized in Table 1. For instance, it is found that compared with other factors (*e.g.*, optimizer), the piracy models have a limited impact on the attack performance, challenging the conventional notion that more advanced models lead to more effective attacks. Also, it is shown that adversarial examples offer little boost to or may even negatively impact the efficacy of ME attacks, which contradicts the findings in prior work.

– Finally, by integrating a longitudinal dataset containing over 1.7 million queries to leading MLaaS platforms spanning from 2020 to 2022, we conduct a retrospective study to characterize the vulnerability evolution of these MLaaS platforms. We discover some significant trends in vulnerabilities and identify the impact of model updates on ME attacks. Our analysis leads to some notable findings, including the influence of model updates on ME attack results and the potential lack of investment in model protections. Our contribution highlights the historical perspective of the evolving ME attack in MLaaS platforms, emphasizing the necessity for enhanced security measures and proactive defense against ME attacks.

We envision that the MeBench platform and our findings facilitate future research on ME attacks and shed light on building MLaaS platforms in a more secure manner.

2 BACKGROUND

2.1 Preliminaries

We first introduce fundamental concepts and assumptions used throughout the paper.

Deep neural networks (DNNs) [35] represent a class of ML models to learn high-level abstractions of complex data. In a predictive task, a DNN f_θ (parameterized by θ) encodes a function $f_\theta : \mathcal{X} \to \mathcal{Y}$, which maps an input $x \in \mathcal{X}$ to a class $y \in \mathcal{Y}$. The training of f_θ often involves iteratively updating θ via algorithms such as stochastic gradient descent (SGD) [67], aiming to minimize a loss function (*e.g.*, cross-entropy) that measures the discrepancy between the model's prediction $f_\theta(x)$ and the ground-truth class y. Techniques such as data augmentation, batch normalization, and dropout may also be used during training to accelerate training or prevent overfitting. As their design may require significant engineering effects and their training may involve substantial data and compute resources, DNN models are often considered invaluable intellectual property in various contexts.

Knowledge distillation (KD) [20, 23] is a process where a student model f^{student} is trained to mimic a teacher model f^{teacher}. Typically, KD involves minimizing the discrepancy between the two models, which can be measured in responses (*i.e.*, models' outputs) [23], features (*i.e.*, models' intermediate representations) [28, 68], or relations (*i.e.*, models' modeling of input relationships) [36, 81]. Formally,

$$\min_\theta \mathbb{E}_{x \in \mathcal{D}} \, \Delta(f_\theta^{\text{student}}(x), f^{\text{teacher}}(x)) \tag{1}$$

where Δ measures the discrepancy between two models with respect to a reference dataset $\mathcal{D}$.

Model extraction (ME) aims to infer a victim model's properties typically through black-box query access. The inferred information may include the model's architecture, (hyper)parameters, functionality, and other properties (*e.g.*, attack vulnerability). The existing ME attacks can be roughly categorized as exact or approximate extraction.

Exact extraction aims to infer the victim model's properties *exactly*, for instance, architectures [56], hyperparameters [77], and parameters [75] (with respect to known architectures). For instance, equation-solving attacks [75] allow the extraction of the exact parameters of (multi-class) logistic regression and multi-layer perceptron models. Approximate extraction aims to construct a piracy model similar to the victim model [19, 60, 63, 75, 82], which can be further divided based on its goal: 1) obtaining similar performance as the victim model (measured by accuracy) or 2) acquiring similar behavior as the victim model (measured by fidelity).

In the following study, we primarily focus on KD-based, approximate ME attacks, due to their general applicability and limited assumptions.

2.2 Threat model

We first define the threat model assumed in our study.

Adversary's objective – The adversary's goal is to generate a piracy model f_p that approximates the behavior and/or performance of the MLaaS backend model f_v. The agreement between f_p and f_v

Previous Conclusion	Refined Conclusion	Consistency
Identifying the architectures of victim models and using the same architectures in piracy models substantially boost the effectiveness of ME attacks [13].	The architectures of piracy models have a limited impact on the performance of ME attacks, while more advanced models may not lead to more effective attacks.	◐
Using more complex models tends to achieve better attack performance [3, 33, 70].	The impact of model complexity varies with the concrete tasks (*i.e.*, FER versus NLU) while pre-training is a more dominating factor.	◐
Semi-supervised learning substantially improves the query efficiency of ME attacks, especially when the query budget is low [29].	Semi-supervised learning improves the query efficiency of ME attacks, but the margin of improvement is not as large as reported in local experiments [29]; further, it has a negative impact on adversarial fidelity.	◐
Active learning substantially improves the query efficiency of ME attacks [60].	Active learning yields only marginal improvements in query efficiency and can, in some cases, have a negative effect.	◐
Using adversarial examples improves both the query efficiency and the attack effectiveness [31, 61, 65, 82].	Using adversarial examples has a limited or even negative impact on attack fidelity, but may improve adversarial fidelity.	○
Perturbing output confidence scores effectively mitigates ME attacks [37, 58].	Output quantization weakens ME attacks but is not sufficiently effective.	◐

Table 1. Comparison of conclusions in prior work and MeBench (○ – inconsistent; ◐ – partially inconsistent).

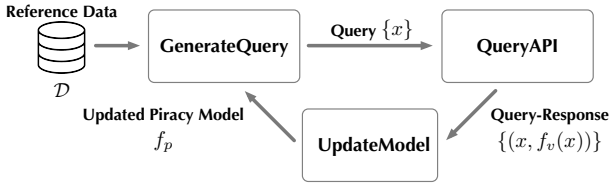


Figure 2: A general framework of ME attacks.

is measured by either accuracy or fidelity with respect to a testing dataset $\mathcal{D}^\star$.

Adversary's knowledge – We assume a black-box setting, in which the adversary has little knowledge about the victim model f_v, including its architecture, (hyper)parameters, and the specific dataset used in its training. Yet, the adversary has access to a reference (unlabeled) dataset $\mathcal{D}$.

Adversary's capability – The adversary is able to access f_v through the MLaaS API, which, for a given query input x, returns f_v's prediction $f_v(x)$. Note that the queries are not restricted to real data and may include synthesized or adversarial inputs. Moreover, as MLaaS often charges users on a per-query basis, we assume the adversary has a limited query budget n_{query}.

2.3 A general attack framework

In MeBench, all the ME attacks are implemented within a general framework, as illustrated in Figure 2. This framework comprises three key functions i) GenerateQuery, ii) QueryAPI, and iii) UpdateModel. Specifically, GenerateQuery generates queries that are either sampled from the reference dataset $\mathcal{D}$ or synthesized; QueryAPI then sends each query x to the MLaaS API and receives its prediction $f_v(x)$; finally, UpdateModel optimizes the piracy model f_p using the query-response pairs $\{(x, f_v(x))\}$. Notably, this process can be executed iteratively: GenerateQuery generates queries based on both the updated piracy model and the previous query results (*i.e.*, using active learning to identify the next batch of queries).

In the default ME attack, we sample n_{query} inputs from $\mathcal{D}$ to query the MLaaS API and use all the query-response pairs $\{(x, f_v(x))\}$ to train the piracy model f_p by minimizing the following cross-entropy loss:

$$\min_{\theta} -\sum_{x \in \mathcal{D}} f_v(x)^T \log f_p(x; \theta) \tag{2}$$

3 STATUS QUO OF ME VULNERABILITY

Leveraging MeBench, we conduct an empirical study on leading MLaaS platforms. Our study is designed to center around 4 questions:

Q_1: How vulnerable are today's MLaaS APIs to ME attacks?
Q_2: How do various factors influence such vulnerability?
Q_3: How does the vulnerability evolve over time?
Q_4: How effective are the defenses (if any) employed on the MLaaS platforms?

3.1 Experiment setting

We begin by introducing the setting of our evaluation.

Datasets – To be succinct yet reveal key issues, we primarily use 6 benchmark datasets: RAFDB [42], EXPW [84], KDEF [51], FER+ [5], IMDB [1], and YELP [2], spanning both the vision and natural language domains. The first 4 correspond to the facial emotion recognition (FER) task, which classifies given facial images into 7 possible expressions (*e.g.*, "calm"). The last 2 correspond to the natural language understanding (NLU) task, which classifies given text into 4 sentiments (*i.e.*, "positive", "negative", "neutral", and "mixed"). The details of the datasets are deferred to Table 18. By fault, following prior work, we partition each dataset into an 80%/20% split, designating the 80% as the reference data $\mathcal{D}$ (for ME attacks) and the remaining 20% as the testing data $\mathcal{D}^\star$ (for performance evaluation). In addition, we also evaluate the scenario in which $\mathcal{D}$ and $\mathcal{D}^\star$ come from different datasets in §3.4.

APIs – We consider the APIs of 4 leading MLaaS service providers: Amazon, Google, Microsoft, and Face^{++}. By default, we assume the adversary has access to the complete query response (*i.e.*, classification labels and confidence scores). Specifically, in the FER task, each query response contains the confidence scores of different expressions (*e.g.*, "calm"). In the NLU task, each query response includes

the confidence scores of different sentiments (*e.g.*, "positive"). The details of each API are deferred to Table 14.

Piracy models – In FER, we consider 4 representative architectures for the piracy model: VGG [72], ResNet [21], DenseNet [26], and ViT [17], while in NLU, we consider 2 Transformer-based architectures for the piracy model: RoBERTa [47] and XLNet [80]. Using models of distinct architectures (*e.g.*, residual blocks versus skip connects), we aim to factor out the influence of individual model characteristics. Besides the backbone model, we use one fully connected layer with softmax activation as the classification head. By default, we assume the piracy models are randomly initialized and trained from scratch. In §3.3.2, we also consider the scenario in which the piracy models are pre-trained on public datasets. In the FER task, the models are pre-trained on the ImageNet-1K dataset; in the NLU task, the models are pre-trained on the BERT dataset [16].

Metrics – To evaluate the effectiveness of ME attacks, we mainly use three metrics:

Accuracy measures the fraction of inputs in the testing set that are correctly classified by the piracy model f_p. Formally,

$$\mathrm{Acc}(f_p) = \frac{\sum_{(x,y)\in\mathcal{D}^\star} \mathbb{1}_{f_p(x)=y}}{|\mathcal{D}^\star|} \tag{3}$$

where $\mathbb{1}_A$ denotes the indicator function that returns 1 if the predicate A is true and 0 otherwise.

Fidelity measures the fraction of inputs in the testing set that receive the same classification by the victim and piracy models [15, 29, 59, 61, 82]. Formally,

$$\mathrm{Fid}(f_p) = \frac{\sum_{(x,y)\in\mathcal{D}^\star} \mathbb{1}_{f_v(x)=f_p(x)}}{|\mathcal{D}^\star|} \tag{4}$$

Adversarial fidelity measures the fraction of adversarial examples (with respect to f_v) that receive the same classification by f_v and f_p. Formally,

$$\mathrm{AdvFid}(f_p) = \frac{\sum_{(x)\in\mathcal{A}(\mathcal{D}^\star)} \mathbb{1}_{f_p(x)=f_v(x)}}{|\mathcal{A}(\mathcal{D}^\star)|} \tag{5}$$

where $\mathcal{A}$ is the adversarial attack (*e.g.*, PGD [54]) that generates adversarial examples with respect to f_p. We mainly use this metric in adversarial ME attacks (§3.3.3).

API	Dataset			
	KDEF	RAFDB	EXPW	FER+
Amazon	+0.49±0.03	-0.98±0.15	-1.23±0.092	+1.99±0.12
Microsoft	/	+1.23±0.19	+2.12±0.17	+4.12±0.18
Face^{++}	+2.12±0.18	+1.78±0.13	+1.97±0.04	+0.45±0.14

Table 2. Difference of attack fidelity w/ and w/o data augmentation.

Data augmentation – In the vision domain, data augmentation (*e.g.*, random cropping and resizing) plays a significant role in model training. Thus, we conduct a pilot study on its effectiveness in the basic ME attack, with results shown in Table 2. Observe that data augmentation improves the attack fidelity in most cases. Thus, we apply data augmentation by default in the FER experiments.

ME attacks – We evaluate a variety of ME attacks. The basic attack is based on knowledge distillation [81], which randomly samples n_{query} inputs from the reference dataset $\mathcal{D}$ as queries, receives query response $f_v(x)$ from the MLaaS API for each query x, and subsequently trains the piracy model f_p based on the query-response pairs $\{(x, f_v(x))\}$. Moreover, we take into account more advanced attacks that aim to minimize the number of queries through various strategies (*e.g.*, semi-supervised learning) in §3.3.3.

3.2 Q1: Overall vulnerability

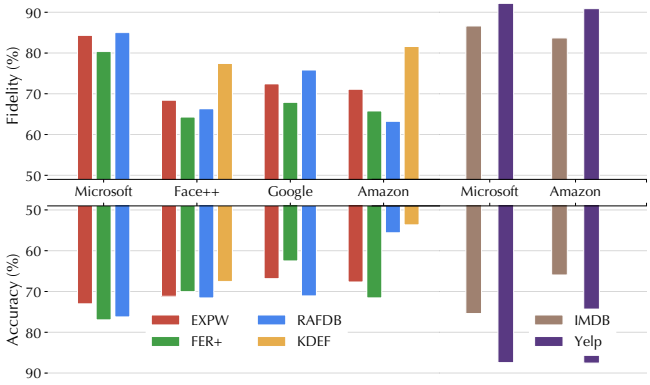


Figure 3: Vulnerability of different MLaaS APIs to ME attacks under the default setting.

3.2.1 Measurements. We first examine the overall vulnerability of MLaaS APIs to ME attacks, with results summarized in Figure 3. We have the following observations.

Platform variations – A detailed evaluation of different APIs on the RAFDB dataset during ME attacks shows significant variations. The attack fidelity varies from 63.53% against the Amazon API to 83.98% against the Microsoft API. These variations can be attributed to several factors:

(*i*) *Label discrepancies* – For instance, Amazon employs unique 8-class labels, diverging from the typical 7-class labels used by other platforms and the dataset itself.

(*ii*) *Training data distributions* – While we lack precise knowledge about the original training data across different platforms, this factor is highly likely to influence the outcomes of ME attacks.

(*iii*) *Pre-processing and post-processing* – The unique pre-processing (*e.g.*, data cleaning, normalization, and augmentation) and post-processing (*e.g.*, confidence score quantization) implemented by different MLaaS platforms may also have a substantial impact on the ME vulnerability.

Task/dataset variations – Our findings also show that the ME vulnerability varies across tasks and datasets. This is evident in the significant fluctuation in the attack fidelity across different tasks and datasets. We attribute these variations to the unique characteristics intrinsic to each dataset.

In the NLU task, take the Amazon API as an example. Recall that the output of Amazon API falls into 4 categories (*i.e.*, "positive", "negative", "mixed", and "neutral"), where "mixed" exists as a separate class. IMDB is a high-polarity dataset, while Yelp is a diverse semantic dataset including a number of instances with mixed sentiments. This unique feature of the Yelp dataset allows the attack to

better capture the "mixed" class, leading to higher attack fidelity compared with the IMDB dataset.

Compared with the NLU task, the FER task requires classifying given facial expressions into 7/8 emotion classes, without ambiguous "mixed" classes. This explains the relatively lower attack fidelity in the FER task. Further, observe that against the Amazon API, the attack fidelity on KDEF is much higher than the other datasets. This is explained by that KDEF explicitly instructs subjects to exhibit exaggerated expressions during data collection, which makes the classification task much easier. However, the piracy model extracted from KDEF does not generalize to other datasets (more details in §3.4).

Fidelity vs. accuracy – Figure 3 also shows that the variations in fidelity (63% to 85%) and accuracy (49% to 77%) across different APIs do not always align with each other. For instance, in the FER task, the attack against the Amazon API attains the highest fidelity yet the lowest accuracy on KDEF across different datasets. This is explained by that the agreement between the victim and piracy models, measured by fidelity, is not necessarily correlated with their performance, measured by accuracy. Given that ME attacks aim to extract the functionality of victim models, rather than attaining high performance, similar to prior work [19, 57, 60, 82], we primarily focus on the metric of attack fidelity in the study below.

The findings above highlight the need for effective mitigation in mainstream MLaaS APIs. Moreover, it reiterates the importance of comprehending different factors (*e.g.*, datasets and tasks) for accurately interpreting the ME vulnerability across different MLaaS platforms.

3.2.2 Comparison with prior work. We also compare our results with prior work on ME vulnerability in both settings of local models and MLaaS APIs.

Local models – Most prior work (*e.g.*, [19, 57, 60]) focuses on ME attacks against local models and uses CIFAR10 as the primary dataset. In particular, InverseNet [19] shows superior performance over the other ME attacks on CIFAR10. If the adversary is able to access the full confidence information, InverseNet attains 45%, 70%, 81%, and 82% attack fidelity under 1K, 5K, 10K, and 15K queries, respectively. The fidelity growth plateaus after 15K queries, indicating a diminishing return from further increasing the query budget.

In Figure 3, the Microsoft API shows the highest vulnerability, with 84% attack fidelity for EXPW (27K queries) and RAFDB (12K queries), respectively. Due to its smaller size, the lower number of queries for FER+ (3K queries) results in slightly lower fidelity than the other two datasets. These findings corroborate with prior work, showing a positive positive correlation between query budget and attack fidelity.

MLaaS APIs – Among prior work, Cloudleak [82] conducts ME attacks against MLaaS APIs and measures attack fidelity with respect to different APIs and query budgets. Specifically, it reports 81% fidelity and 58% accuracy against the Face^{++} API on the KDEF dataset. Furthermore, it reports that the attacks against other platforms such as Microsoft and Clarifai, as well as different tasks including traffic sign recognition, flower, and NSFW, all attain over 82% fidelity and 70% accuracy. This clearly showcases the widespread nature of this vulnerability across various APIs and tasks.

In our experiments, on the same platform and dataset (*i.e.*, KDEF and Face^{++}), we observe that the attack attains similar results (74% fidelity and 66% accuracy), while the performance gap with [82] may be attributed to the scratch model and the difference of testing data. In [82], the testing data is synthesized, while we randomly sample 20% of the KDEF dataset as the testing set. Further, similar to [82], the ME vulnerability varies considerably across different APIs. In particular, the attack attains over 80% fidelity against the Microsoft API across all the datasets, while the other APIs demonstrate more variations from one dataset to another.

Overall, we may conclude:

Observation 1 – Many popular MLaaS APIs continue to be highly vulnerable to ME attacks, while this vulnerability varies greatly with concrete tasks and datasets.

3.3 Q2: Influential factors

Next, we evaluate how different key factors impact the effectiveness of ME attacks with respect to given MLaaS APIs.

3.3.1 Optimizers. Prior studies indicate that the selection of optimizers greatly impacts training dynamics [78]. Thus, we examine how it affects ME attack performance.

We specifically consider 4 popular optimizers: SGD [67], Adam [32], AdamW [50], and Lion (EvoLved Sign Momentum) [12], which is a recently proposed optimizer. We configure the basic ME attack with varied optimizers and evaluate its effectiveness in the FER and NLU tasks.

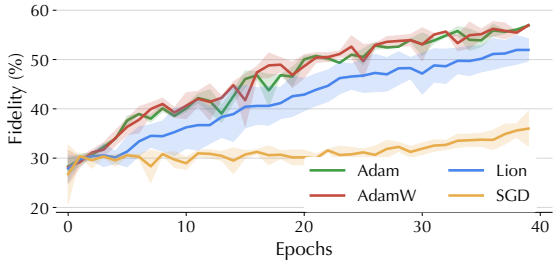
API	Model	Optimizer			
		Lion	AdamW	Adam	SGD
RAFDB in FER					
Amazon	ResNet50	62.89±0.53	64.15±0.17	63.96±0.06	56.06±0.65
Microsoft		84.64±0.20	84.35±0.11	84.35±0.10	81.01±0.65
IMDB in NLU					
Amazon	XLNet	84.30±0.16	82.45±0.20	82.67±0.32	71.69±0.27
	RoBERTa	81.20±0.21	74.72±0.08	74.58±0.12	42.93±0.32
Microsoft	XLNet	86.35±0.23	86.57±0.52	86.57±0.24	84.49±0.36
	RoBERTa	86.55±0.51	85.20±0.54	85.15±0.36	65.75±0.29

Table 3. Attack fidelity of different optimizers in FER and NLU.

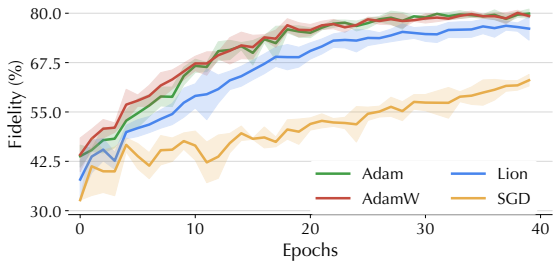
FER – We compare different optimizers using RAFDB on the Amazon and Microsoft APIs. Following [12], we set the hyperparameters (*e.g.*, learning rate LR and decay rate β) for different optimizers as follows: SGD with $\beta = (0.9, 0.999)$ and LR = 3e−2, Adam/AdamW with $\beta = (0.9, 0.999)$ and LR = 3e−4, and Lion with $\beta = (0.9, 0.99)$ and LR = 3e−4. We have the following key observations.

Table 3 shows the attack fidelity at the end of 200 training epochs. Under the same query budget, Lion, Adam, and AdamW attain comparable attack fidelity. There is no distinct advantage between Lion and AdamW; however, it is evident that SGD lags behind the performance of other optimizers significantly. We further examine the training dynamics of different optimizers. Figure 4 summarizes how the attack fidelity varies with the number of training epochs. Despite the claimed superior convergence speed of Lion in supervised

learning [12], we observe that Adam/AdamW actually converges much faster than Lion in the ME attack, while SGD converges fairly slowly.



(a) RAFDB on Amazon



(b) RAFDB on Microsoft

Figure 4: Training dynamics of different optimizers in the FER task.

NLU – We also set the hyper-parameters of optimizers following [12]: SGD with $\beta = (0.9, 0.99)$ and LR = 5e−4, Adam/AdamW with $\beta = (0.9, 0.99)$ and LR = 3e−6, and Lion with $\beta = (0.95, 0.98)$ and LR = 3e−6.

Table 3 summarizes the attack performance corresponding to different optimizers on the IMDB dataset. Lion generally outperforms the other optimizers on both Microsoft and Amazon APIs. Our observations corroborate prior findings [12] that Lion outperforms alternative optimizers in fine-tuning large language models.

Overall, we may conclude:

Observation 2 – Advanced optimizers enhance ME attacks in terms of training efficiency and attack fidelity; the selection of optimizers depends on concrete datasets and tasks.

3.3.2 Piracy models. The architectures of piracy models are also crucial for ME attacks. It is shown in prior work [13] that correctly identifying the architectures of victim models and using the same architectures in piracy models substantially boost the effectiveness of ME attacks. Several studies [3, 33, 57, 70] also demonstrate that the piracy model needs to be at least as complex as the victim model. Additionally, the results of [3, 33, 70] suggest that utilizing a more complex model leads to better attack performance.

In the open-world setting targeted in this study, the victim models are not accessible, limiting us from directly comparing the architectures of piracy and victim models. Therefore, in the study below, we evaluate how different aspects of piracy models may impact the ME attacks with respect to given MLaaS APIs. Specifically, we explore three key aspects of model architectures: model families, model complexity (within the same family), and (non-)use of pre-training. Table 4 summarizes the experimental settings in evaluating the impact of piracy models.

Component	Model	Pre-training	#Params (M)	Model Family
Model Family	GoogLeNet	/	6.7	GoogLeNet
	DenseNet		8	DenseNet
	EfficientNet		56	EfficientNet
	AlexNet		61	AlexNet
	ResNet50		25.6	ResNet
Model Complexity	ResNet18	/	11.7	ResNet
	ResNet50		25.6	
	ResNet101		44.6	
	WResNet50		68.9	
Use of Pre-training	ResNet50	/	25.6	ResNet
		ImageNet1K		
		VGGFace2		

Table 4. Setting of experiments on the impact of piracy models.

Model	Variant	API	
		Amazon	Microsoft
XLNet	Base	84.50	86.19
	Large	84.30	86.35
RoBERTa	Base	78.96	86.12
	Large	81.20	86.62

Table 5. Impact of piracy model architectures on ME attacks (NLU).

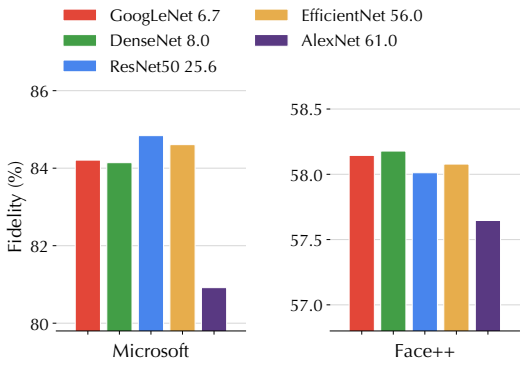


Figure 5: Impact of piracy model architectures on ME attacks (FER).

Model family – For FER, we compare 5 popular architectures, GoogLeNet [73], DenseNet [27], ResNet [22], EfficientNet [74], Alex-Net [34], using RAFDB on the Microsoft and Face^{++} APIs, with results shown in Figure 5. For NLU, we compare 2 popular LLMs, RoBERTa and XLNet, using IMDB on the Amazon and Microsoft APIs, with results summarized in Table 5. We have the following interesting findings.

While prior work suggests that the piracy model architecture greatly impacts the effectiveness of ME attacks (*e.g.*, 10-30% difference in attack fidelity [13]). However, in our evaluation, except for AlexNet, an architecture proposed in 2012, which substantially under-performs the other architectures, the difference among other

architectures is relatively marginal. A similar phenomenon is also observed in the NLU task as shown in Table 5.

It is worth pointing out that the number of parameters in these architectures varies from 6.7K to 61K. Yet, interestingly, GoogleLeNet, with only 6.7K parameters, delivers performance comparable with more complex architectures. To further evaluate the impact of model families, we consider ViT [17], a more advanced architecture with 86K parameters. However, it proves challenging to optimize ViT due to the limited data; even with pre-training, ViT does not match ResNet in terms of attack fidelity. This implies that for ME attacks, overly complex architectures can be counterproductive.

Observation 3 – In real-world MLaaS settings, the piracy model architecture has a limited impact on ME attacks, while more advanced architectures may not lead to more effective attacks.

Model complexity – We further assess the impact of model complexity (within a single model family). In FER, we use ResNet as the piracy model and vary its width and depth, spanning across ResNet18, ResNet50, ResNet101, and Wide-ResNet50. As illustrated in Figure 6, the attack fidelity across different architectures differs by less than 0.60% and 0.20% on Amazon and Face^{++}, respectively. Intuitively, as the model complexity reaches a certain level, further increasing the model width or depth does not lead to a notable improvement in attack performance. Similarly, in NLU, as shown in Table 5, for both RoBERTa and XLNet, using large models yields marginal improvement over base models. It is worth pointing out that compared with the base models, the large models of RoBERTa and XLNet are pre-trained using more data (97GB additional data) [47, 80].

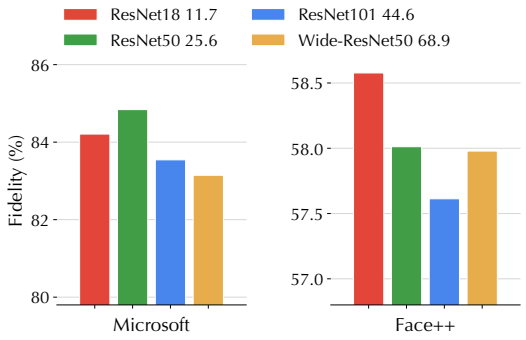


Figure 6: Impact of model complexity on ME attacks (FER).

Observation 4 – Once reaching a certain level, further increasing the model complexity has a marginal impact on ME attacks.

Use of pre-training – Next, we evaluate the influence of pre-training piracy models using public datasets on the performance of ME attacks.

In FER, we compare the attacks using a randomly initialized ResNet50 model as the piracy model with that using ResNet50 models pre-trained on public datasets. We consider two pre-training datasets, ImageNet-1K and VggFace2 [7]. As shown in Figure 7, pre-training marginally improves the attack performance, while

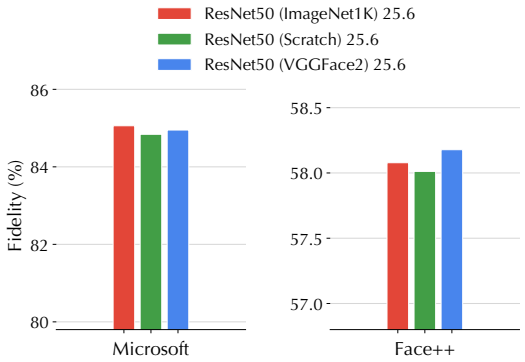


Figure 7: Impact of using pre-training on ME attacks (FER).

Model	Pre-training	API	
		Amazon	Microsoft
XLNet-Base	✗	36.39	65.78
	✓	84.50	86.19
RoBERTa-Base	✗	53.94	73.61
	✓	78.96	86.11

Table 6. Impact of using pre-training on ME attacks (NLU).

the improvement margin varies with the APIs. For instance, compared with VGGFace2, ImageNet-1K leads to more improvement on Microsoft but less so on Face^{++}. In NLU, we compare randomly initialized XLNet and RoBERTa and that pre-trained on the BERT dataset [16], with results shown in Table 6. Surprisingly, unlike the PER task, pre-training language models significantly improves the attack performance. For instance, the pre-training of RoBERTa boosts fidelity by over 25%.

Thus, we may conclude:

Observation 5 – Pre-training generally improves the performance of ME attacks, while the boost is more evident for language models than vision models.

3.3.3 Attack strategies. Thus far, we mainly focus on the basic attack strategy that randomly samples queries from the reference dataset. Next, we investigate advanced strategies that aim to minimize the number of queries. Specifically, we consider three popular strategies: semi-supervised learning, active learning, and adversarial learning.

Semi-supervised learning – Via semi-supervised learning, the adversary queries the API with a small number of samples and mixes the query responses (labeled data) and unlabeled data to train the piracy model. We consider MixMatch [6] as a representative semi-supervised learning method, which, in the context of ME attacks, is employed to reduce the number of queries [29]. At a high level, MixMatch-based ME attack samples a batch of samples to query the API as the labeled data $\mathcal{X}$ and mixes it with an equally-sized batch of unlabeled data $\mathcal{U}$ to produce a batch of augmented labeled data $\mathcal{X}'$ and a batch of augmented unlabeled data with pseudo labels $\mathcal{U}'$, which are used to train the piracy model but with different objective functions.

We evaluate the MixMatch-based ME attack in the FER task. Specifically, using the RAFDB and EXPW datasets on the Microsoft

API	Dataset	Attack	Type	Query Budget (# Batches)							$\overline{\Delta}$
				4	8	16	32	64	128	Full	
Microsoft	RAFDB	Basic	Fid	68.94±0.37	73.8±0.88	77.36±0.13	79.43±0.33	81.42±0.27	83.29±0.16	84.35±0.13 (191 Batches)	+1.12
		MixMatch		70.25±1.35	75.01±0.56	78.67±0.16	80.62±0.18	82.47±0.35	83.93±0.02		
	EXPW	Basic		70.25±1.15	73.42±0.42	76.98±1.7	78.45±0.72	80.17±0.19	81.54±0.31	83.94±0.08 (417 Batches)	+1.44
		MixMatch		72.41(1.03)	74.88±1.2	77.65±1.33	80.05±0.57	81.6±0.03	82.85±0.41		
Amazon	FER+	Basic	Fid	41.41±0.56	48.09±1.22	53.84±0.20	59.66±0.32	/	/	63.20±0.41	+5.91
			AdvFid	18.16±0.45	16.49±0.22	17.26±1.60	18.59±0.61	/	/	18.57±1.12	
		MixMatch	Fid	49.73±1.42	55.39±1.40	59.2±1.15	62.32±0.16	/	/	(50 Batches)	
			AdvFid	16.62±0.74	16.05±1.20	16.07±0.67	15.7±1.07	/	/		

Table 7. Fidelity and adversarial fidelity of basic and MixMatch-based ME attacks (with batch size fixed as 64). In the case of MixMatch, each labeled batch is paired with an equal-sized unlabeled batch.

API, we compare the performance of basic and MixMatch-based ME attacks with varying numbers of queries, ranging from 4 to 128 batches (with batch size fixed as 64). Note that the FER+, RAFDB, and EXPW datasets include 50, 191, and 417 batches, respectively. As summarized in Table 7, we have the following interesting findings.

(*i*) MixMatch generally enhances the effectiveness of ME attacks across different datasets and APIs. However, the improvement is less significant than that reported in [29]. For instance, it improves the fidelity by about 1% against the Microsoft API and about 5% against the Amazon API, under varying query budgets. The divergence from prior work may be attributed to both models and data. For instance, the results in [29] are evaluated using a one-layer fully-connected network on the CIFAR10 data, while the MLaaS backend models often use much more complex architectures trained on a massive amount of data (*e.g.*, millions of images) [4].

(*ii*) Intriguingly, while boosting attack fidelity, MixMatch has a negative impact on adversarial fidelity, which measures the agreement between the victim and piracy models with respect to adversarial examples. For instance, the basic attack achieves adversarial fidelity 2.9% higher than the MixMatch-based attack with $n_{\text{budget}} = 32$ batches. This observation may be explained as follows. MixMatch augments the training data by interpolating different samples, which essentially "smooths" the decision boundaries. While improving query efficiency, it also prevents the piracy model from effectively learning the high-curvature decision boundaries of the victim model, which is essential for adversarial fidelity.

Observation 6 – Semi-supervised learning generally improves attack fidelity under a limited query budget but has a negative impact on adversarial fidelity.

Active learning – The adversary may also employ active learning to choose informative inputs rather than randomly sampled ones to query the API to reduce the query cost.

We adopt k-Center-Greedy [69], a method also used in Active -Thief [60], in ME attacks. At each iteration, ActiveThief picks k central samples from the unlabeled data (determined as cluster centers in the piracy model f_p's latent space) to query the API. These responses are added to the training set $\mathcal{D}$, which is used to train f_p. This process continues until the allocated query budget n_{budget} is exhausted. The results are summarized in Table 8. We have the following findings.

In our experiment combination of Microsoft+RAFDB, the results from Active Learning appear inconsistent. Some seed and query budgets show slight improvements, but these results are unstable, especially with lower query budgets. For instance, Active Learning performs worse with query budgets of 16 and 32 batches than the Basic attack. With a budget of 64 batches, there is a slight average improvement of 0.26%. However, as the query budget rises, the performance becomes steadier, although the improvement is never over 1%. This initial inconsistency might be attributed to the initial model struggles to capture face feature maps well. When the model is capable, the number of queries has been exhausted. As a result, when the model becomes more capable, the number of available queries has already been exhausted, which may prevent it from selecting the best data points. We notice a similar pattern with the Amazon+RAFDB combination. AdvFid and Fid perform worse than the Basic attack at low query budgets. Otherwise, in the Microsoft+EXPW combination, Active Learning always performs worse than the Basic attack.

Back to previous research on active learning, when k-Center-Greedy combines with data augmentation [55], results in only slight accuracy improvements. Specifically, they show an increase of approximately 1 to 2% on CIFAR10 compared to random sampling-based queries. In earlier studies related to ME attacks, [60] demonstrates that using the k-Center-Greedy method leads to an average improvement of 0.976% in attack fidelity on CIFAR10. However, when the query budget increases to 25K, this method underperforms by -0.62%. In previous work, it is effective in most cases but has limited improvement and shows instability.

In conclusion, active learning-based attacks on different APIs and datasets often perform worse than basic ME attacks. They demonstrate lower fidelity and adversarial fidelity, especially when there are fewer data queries. Given its inconsistent performance, active learning is not recommended for ME attacks in real-world scenarios.

Observation 7 – Introducing active learning is likely to acquire negative effects, especially when the amount of queries is small

Adversarial learning – CloudLeak [82] applies adversarial learning to generate examples to reduce the required number of queries in ME attacks. Specifically, the strategy is to perform adversarial attacks on clean data $\mathcal{X}$ and produce adversarial examples $\mathcal{A}$ on the

API	Dataset	Attack	Type	Query Budget (# Batches)					
				16	32	64	96	128	Full
Microsoft	RAFDB	Basic	Fid	76.74±0.76	79.28±0.54	81.76±0.08	82.74±0.15	83.2±0.39	84.35±0.13 (191 Batches)
		ActiveThief		76.45±0.31	79.17±0.4	82.02±0.33	82.93±0.3	84±0.16	
	EXPW	Basic		75.94±0.56	78.52±0.45	79.76±0.08	81.08±0.11	81.57±0.24	83.94±0.08 (417 Batches)
		ActiveThief		74.56±0.68	77.09±0.42	79.42±0.09	80.43±0.47	81.04±0.21	
Amazon	RAFDB	Basic	Fid	53.52±0.57	57.07±0.61	59.41±0.72	61.08±0.8	61.69±0.41	63.53±0.14
			AdvFid	21.61±0.01	21.29±0.02	22.52±1.06	22.2±0.95	23.32±0.41	22.81±0.13
		ActiveThief	Fid	52.14±0.07	56.03±0.28	58.81±0.36	60.32±0.7	61.81±1.17	(191 Batches)
			AdvFid	20.34±0.6	21.26±0.98	22.05±0.8	22.99±0.95	22.55±0.31	

Table 8. Fidelity and adversarial fidelity of basic and active learning-based ME attacks (with batch size fixed as 64).

piracy model f_p. These examples represent highly uncertain regions of f_p. When these regions are queried against the API, they offer valuable insights to refine f_p. Additionally, since $\mathcal{A}$ lies close to f_p's decision boundaries, these queries push f_p towards the decision boundaries of the victim model f_v. We consider two representative adversarial attacks: PGD [54] (with $\epsilon = 4/255$, $\alpha = 2/255$, $n_{\text{iter}} = 7$, and random δ initialization) and CW (with $\kappa = 40$, $n_{\text{step}} = 50$, and LR = 0.01) [9]. The results are summarized in Table 9.

API	Dataset	Query Batches	Attack		
			Basic	PGD	CW
Amazon	RAFDB	8	40.46±0.39	36.97±0.1	37.5±0.22
			17.27±0.27	19.02±0.05	20.35±0.35
		16	43.48±0.47	41.66±0.17	40.39±0.45
			19.66±0.17	19.16±0.29	21.27±0.15
		32	45.94±3.81	42.92±2.02	46.91±2.94
			21.33±0.15	20.71±0.22	23.4±0.48
	FER+	8	35.32±1.83	39.91±0.2	41.52±1.58
			17.06±0.01	18.86±0.97	18.31±1.28
		16	41.5±1.83	42.33±0.02	46.4±3.97
			19.02±0	20.81±0.08	17.75±4.91
		32	44.56±0.2	49.35±0.38	50.29±0.32
			17.87±0.74	19.46±1.34	17.28±0.35
Face^{++}	RAFDB	8	40.92±0.41	40.92±1.27	35.07±2.7
			20.31±0.29	20.79±0.09	20.56±0.14
		16	44.65±3.25	40.01±0.19	41.09±0.07
			18.58±0.34	16.57±0.48	17.15±0.05
		32	47.91±0.46	45.28±0.18	45.48±0.29
			15.54±0.21	16.2±0.43	17.29±0.24
	FER+	8	67.79±0.31	22.58±1.8	51.02±0.01
			14.46±0.2	23.98±0.46	18.74±0.3
		16	72.61±3.39	45.6±0.49	41.65±2.02
			14.46±0.46	18.28±0.2	13.2±0.2
		32	67.79±0.13	47.51±0.01	47.51±0.27
			14.46±0.47	17.79±0.01	17.57±0.22

Table 9. Fidelity and adversarial fidelity of basic and adversarial learning-based ME attacks (with batch size fixed as 64).

It is observed that against the Amazon API, using PGD or CW to enhance ME attacks leads to a considerable improvement in attack fidelity (around 5.61% on average) on FER+, while the effect on RAFDB is fairly mixed. Meanwhile, against Face^{++}, adversarial learning-enhanced ME attack seems not only ineffective but even under-performs the basic attack. However, across all the cases, adversarial learning leads to a consistent improvement in adversarial fidelity. Our findings indicate that clean queries (sampled from the reference dataset) and adversarial queries (generated by adversarial attacks) seem to contribute to ME attacks differently, respectively improving the fidelity and adversarial fidelity of piracy models.

Dataset	Type	Adversarial/Clean Ratio $\lvert\tilde{\mathcal{X}}\rvert : \lvert\mathcal{X}\rvert$				
		1:0	3:1	1:1	1:3	0:1
RAFDB	Fid	42.92	44.85	43.78	45.41	45.94
	AdvFid	20.71	20.57	20.55	20.37	21.33
FER+	Fid	49.35	44.2	43.14	47.00	44.71
	AdvFid	19.46	18.62	20.06	20.07	17.35

Table 10. Fidelity and adversarial fidelity under varying adversarial/clean ratios in queries.

These observations lead us to explore an interesting question: whether combining clean and adversarial inputs in querying the MLaaS API improve the overall efficacy of ME attacks. To investigate this, we vary the ratio of clean $\mathcal{X}$ and adversarial $\mathcal{A}$ (generated by PGD) queries in each batch while keeping the query budget constant. The experimental results are summarized in Table 10.

In some cases (*e.g.*, RAFDB), including adversarial queries, may negatively impact the attack fidelity. In other cases (*e.g.*, FER+), although only using adversarial queries leads to the best fidelity, the proportion of adversarial queries is not consistently correlated with the attack fidelity, which conflicts with the findings in [82]. Similarly, although using more adversarial queries improves adversarial fidelity, increasing the proportion of adversarial queries is not strictly correlated with the increase of adversarial fidelity.

It is possible to explain these observations as follows. There exists intricate dynamics between clean and adversarial queries: while adversarial queries often represent corner cases with respect to the model's decision boundaries, clean queries represent more average cases. Thus, while increasing the number of queries generally improves the agreement between the victim and piracy models, clean and adversarial queries have different focuses.

The results highlight the complexity of operating this attack strategy as well as its dependency on concrete datasets and APIs. Based on the findings, further research is needed to comprehend the possible advantages of applying adversarial learning in ME

attacks and the optimal strategy to combine clean and adversarial queries to achieve high attack efficacy.

Overall, we may conclude:

Observation 8 – It is essential to properly adjust the proportions of clean and adversarial queries in ME attacks to optimize different metrics.

3.4 Q3: Generalizability

Generalizability is a crucial metric for ME attacks: it measures how the extracted piracy model agrees with the victim model on datasets other than that used in the attack. In other words, generalizability indicates how valuable and flexible the extracted piracy model is. Next, under the MLaaS setting, we empirically measure the generalizability of piracy models. We extract the piracy model f_p by querying the API using the reference dataset $\mathcal{D}$ and evaluate f_p's fidelity and accuracy on another dataset $\mathcal{D}^\star$.

3.4.1 FER. We conduct experiments across four datasets on three APIs to analyze the generalizability of ME attacks in the FER task, with results summarized in Table 11. The highlighted cells indicate when the original and evaluation datasets are identical, serving as a baseline. The results under Microsoft+KDEF are unavailable due to the closure of the Microsoft API. We have the following key observations from Table 11.

Evaluation Dataset	API	Origin Dataset			
		KDEF	RAFDB	EXPW	FER+
KDEF	Amazon	77.78	59.90	67.36	55.21
	Microsoft	/	/	/	/
	Face^{++}	73.61	63.72	67.01	50.00
RAFDB	Amazon	33.64	63.53	57.38	48.34
	Microsoft	/	85.01	83.98	75.17
	Face^{++}	37.67	66.76	66.02	57.15
EXPW	Amazon	37.35	48.03	71.09	53.14
	Microsoft	/	72.75	84.31	77.09
	Face^{++}	47.79	56.91	68.40	58.91
FER+	Amazon	33.42	49.68	65.85	63.2
	Microsoft	/	72.80	83.13	80.36
	Face^{++}	40.78	56.06	66.39	64.29

Table 11. Generalizability of different datasets and APIs (FER).

Asymmetric generalizability – A piracy model extracted from dataset A may perform well on dataset B, but the reverse is not always true. For instance, the model extracted from the EXPW dataset demonstrates strong generalizability across datasets, especially matching or surpassing baselines on RAFDB and FER+. However, the models extracted from RAFDB and FER+ do not perform as well when applied to EXPW. We speculate that highly generalizable datasets, such as EXPW, often benefit from their larger query volumes, allowing them to capture the distributions of smaller, less adaptable datasets like RAFDB and FER+. Specifically, with its 27K queries, EXPW substantially overshadows RAFDB's 12K and FER+'s 3K. Given their shared tasks, EXPW's comprehensive nature likely contributes to its superior generalizability.

Generalizability beyond query budgets – Though KDEF and FER+ boast similar query volumes, KDEF's generalizability lags significantly. This can be attributed to its unique data collection methodology, where subjects were instructed to exhibit exaggerated emotions. This induces a distribution shift in KDEF, making it deviate from the normative distributions of facial image datasets. Conversely, RAFDB, FER+, and EXPW, derived from normal photos, have distributions that are more congruent with one another.

3.4.2 NLU. Table 12 shows the generalizability results in the NLU task. ME attacks using IMDB as the reference dataset $\mathcal{D}$ demonstrate strong generalizability, particularly on Microsoft, where fidelity reaches 90.49% compared to the baseline of 92.18% when transferred to Yelp. However, the generalizability is weaker on Amazon, with a fidelity of 80.71% compared to the baseline of 90.88%. The results of Yelp are interesting, showing poor generalizability to Amazon (similar to IMDB) at 68.05%, but high generalizability to Microsoft like IMDB.

In general, the generalizability of ME attacks in NLU is related to the APIs, Amazon is weaker than Microsoft, possibly because Amazon uses 4 classes compared to Microsoft uses 3 classes. Further, IMDB's data is highly polarized, and Yelp's data also contains neutral and mixed sentiments, which might also contribute to the results of generalizability.

Evaluation Dataset	API	Origin Dataset	
		IMDB	Yelp
IMDB	Amazon	84.30	68.05
	Microsoft	86.35	84.60
Yelp	Amazon	80.71	90.88
	Microsoft	90.49	92.18

Table 12. Generalizability of different datasets and APIs of NLU

Observation 9 – The generalizability of piracy models depends on the relationships between original and evaluation datasets, much more than other factors (APIs, victim models, ME attacks).

3.5 Q4: Defenses

It is unclear what specific defense mechanisms are employed by the MLaaS platforms. However, when examining their APIs, Amazon, Microsoft, and Face^{++} will provide exact confidence values for each class. In contrast, Google's FER categorizes its output using five distinct likelihood descriptors, (*i.e.*, "very_unlikely", "unlikely", "possible", "likely" and "very_likely") offering a different approach to presenting its analysis. This approach seems to be an attempt to obscure the confidence levels of the model's output. The central concept behind such quantization is to reduce the granularity of the confidence values returned by the API, thereby making it more challenging for potential attackers to discern intricate details about the model's internal processes. We mimicked Google's approach and assess the effectiveness of quantization by applying it to other APIs. Specifically, We achieve this by discretizing the confidence values into intervals and substituting the confidence value with the midpoint of the corresponding interval (*e.g.*, $[0.5, 0.7)$ is represented as 0.6). We manually reduce the granularity of the returned confidence information to examine its impact on the performance of the Model Extraction (ME) attack.

API	Dataset			
	KDEF	RAFDB	EXPW	FER+
Amazon	-1.54±0.1	-2.76±0.11	-1.98±0.14	-0.45±0.03
Face^{++}	+0.43±0.08	-0.76±0.18	+0.31±0.12	-0.98±0.08

Table 13. Difference of attack fidelity w/ and w/o quantization augmentation.

Table 13 presents the analysis results. When applied to the Amazon API, quantization resulted in an average decrease of 1.68% in the attack fidelity. In contrast, the Face^{++} API exhibits more mixed results in mitigating ME attacks.

Overall, the manual reduction of granularity in the returned confidence information has a somewhat detrimental effect on the results of the ME attack. However, it is not a sufficiently effective defense strategy against model extraction.

Observation 10 – Quantization of query responses mitigates ME attacks to a limited extent but is insufficient.

4 A RETROSPECTIVE STUDY

To comprehend the evolution of ME vulnerability over the years, we integrate a longitudinal dataset into MEBENCH and conduct a retrospective study on the ME vulnerability of leading MLaaS APIs, spanning the period from 2019 until now. The purpose of this study is to understand the evolving security risks associated with ME attacks.

4.1 Study setting

In this retrospective study, we employ HAPI [11], a longitudinal compilation containing 1,761,417 queries submitted to commercial MLaaS APIs including Amazon, Google, IBM and Microsoft. The dataset spans from 2020 to 2022 and covers a range of tasks, such as image tagging, speech recognition, and text mining. Each data point comprises a query input along with the MLaaS's response (*i.e.*, prediction, annotation, and corresponding confidence scores).

Among the range of tasks covered by the HAPI dataset, we focus on two representative tasks due to the inherent constraints of ME: FER (representing computer vision tasks) and NLU (representing natural language processing tasks). Note that the responses of all the APIs include confidence information for all the classes, with the notable exception of Google's API (details in Table 14).

Due to the unique structures of the HAPI dataset, we face a particular challenge: it provides only the highest confidence score among all the classes, without any information about the remaining classes. To address this challenge, we impute the confidence scores of the remaining classes. Specifically, let the prediction about an input x be a probability simplex $[f_v^0(x), \ldots, f_v^{m-1}(x)]$ over m classes, with each element corresponding to one distinct class. Recall that the API only outputs the highest confidence score, $f_v^i(x)$, where $i = \arg\max_j f_v^j(x)$. Following the maximum-entropy principle [30], which suggests choosing a distribution that retains the most significant degree of uncertainty (or entropy) while adhering to known constraints, we thus assign the confidence scores as:

$$f_v^j(x) = \begin{cases} f_v^{\text{hapi}}(x) & \text{if } j = i \\ \frac{1 - f_v^{\text{hapi}}(x)}{m-1} & \text{if } j \neq i \end{cases} \tag{6}$$

That is, the class with the highest confidence score is assigned the score given by HAPI, while the remaining classes receive an equal assignment $(1 - f_v^{\text{hapi}}(x))/(m-1)$. Note that the sum of confidence scores across all classes equals to 1, ensuring a valid probability distribution.

Also note that in the NLU task, as HAPI only records the highest confidence score between "positive" and "negative", we expand the outputs to three classes: "positive", "negative", and "mixed" in the following study.

Task	API	HAPI[1]	Current[2]	# Classes
FER	Amazon	✗	✓	8
	Microsoft[3]	✓	✗[3]	7
	Face++	✓	✓	7
	Google	✓	✓	7
NLU	Amazon	✓	✓	4
	Microsoft	✗	✓	3

Table 14. Details of MLaaS APIs.

[1] "HAPI" represents the data collected by HAPI during 2020-2022 with only the highest confidence scores.
[2] "Current" represents the data collected by us in 2023 with full confidence scores across all classes, except for Google API which only returns the highest confidence scores.
[3] Microsoft has sunset its FER API for emotion prediction to mitigate potential misuse that subject people to stereotyping, discrimination, or unfair denial of services.

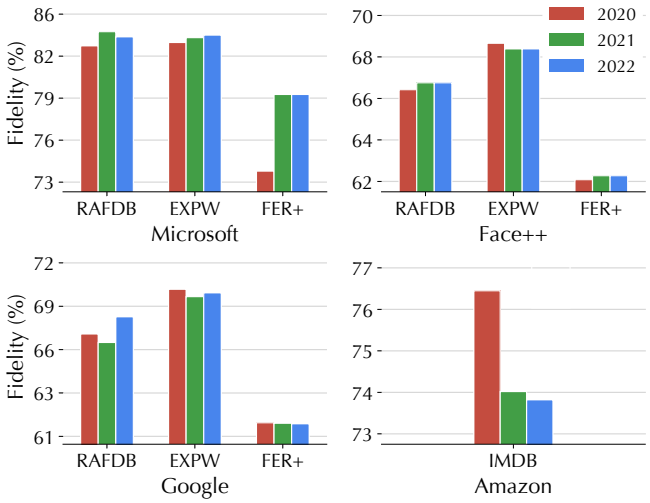


Figure 8: A retrospective study of ME attack vulnerability across different datasets and APIs.

4.2 Results

Our study uncovers interesting trends in the evolving vulnerability of MLaaS APIs to ME attacks. Next, we present our findings in both FER and NLU tasks.

4.2.1 FER. Figure 8 measures the attack fidelity across various APIs and datasets from the year 2020 to 2022. Compared with other platforms, there is a general upward trend for the attack fidelity against the Microsoft API across all three datasets, with a significant shift from 2020 to 2021 on the FER+ dataset. To understand the reason behind this shift, we further conduct a detailed analysis, summarizing the variations in the outcomes of ME attacks and the output data of original APIs in Table 15.

Dataset	ME Attack	API Ouput		
	Fidelity	Accuracy	Predicted Class	Avg. Confidence
FER+	+5.93%	+3.03%	96.07%	0.0169
RAFDB	+1.10%	-0.01%	99.76%	0.0022
EXPW	+0.38%	+0.03%	99.44%	0.0059

Table 15. Change of the Microsoft API from 2020 to 2021 in (i) the fidelity of the ME attack, (ii) the overall accuracy, (iii) predicted classes, and (iv) average confidence scores of the API.

When examining the API outputs with respect to the HAPI dataset, we find that the predictions in 2020 and 2021 overlap more than 99% for RAFDB and EXPW, while this overlap is only around 96% for FER+. It is important to note that these overlaps only match predicted labels and do not account for variations in confidence scores. When considering the average change in confidence scores, FER+ also exhibits much higher variance than other datasets. Finally, the overall accuracy of the Microsoft API in classifying FER+ also increases by 3.03% from 2020 to 2021, while the accuracy for the other two datasets remains little changed.

The substantial changes in the Microsoft API's accuracy and confidence scores with respect to the FER+ and EXPW datasets suggest that the backend model may have undergone significant changes from 2020 to 2021. These alterations likely contribute to the varying fidelity of ME attacks. Specifically, the attack attains 5.93% fidelity increase on FER+, with only minor increments of 1.10% and 0.38% on RAFDB and EXPW, respectively.

While ME attacks appear to become both easier and simpler across different datasets, we believe such changes are mainly due to adjustments in the backend model of the Microsoft API (*e.g.* incorporating new training data to fine-tune the model), rather than any alterations to its defenses against ME attacks. What is particularly interesting is that the significant improvements in model accuracy coincide with the increased attack fidelity. We speculate that after the update, the backend model becomes more aligned with the distribution of the FER+ dataset. As the ME attack specifically targets this dataset, it is easier for the attack to adapt to the backend model. In comparison, Google+EXPW shows an overlap of 90.83% between 2020 and 2021 in HAPI, with a slight fidelity change of -0.47%. All the other dataset-API combinations experience minimal changes, with overlaps consistently exceeding 99%.

In summary, it is evident that in the FER task, the ME vulnerability of various MLaaS APIs has evolved in the past few years. However, we do not observe clear patterns in such evolution and are therefore unable to conclude whether these APIs have strengthened defenses against ME attacks. We tend to believe that most MLaaS platforms have not implemented specific defenses, especially in the case of the Microsoft API, where it exhibits high vulnerability to ME attacks.

4.2.2 NLU. Due to the limitation of NLU datasets in the HAPI dataset, only the combination of Amazon and IMDB is available for analyzing the evolution of ME vulnerability.

Year	ME Attack	API Ouput		
	Fidelity	Accuracy	Predicted Class	Avg. Confidence
2020-2021	-2.43%	-1.08%	83.20%	0.1682
2021-2022	-0.20%	+0.12%	99.41%	0.0070

Table 16. Change of the Amazon API from 2020 to 2022 in (i) the fidelity of the ME attack, (ii) the overall accuracy, (iii) predicted classes, and (iv) average confidence scores of the API.

As shown in Table 16, the attack fidelity against the Amazon API shows a declining trend over the three years, with a decrease of 2.43% from 2020 to 2021. However, since this change is not statistically significant, it does not appear that the API has fortified its defenses against ME attacks in the intervening years.

Despite the growing popularity of MLaaS APIs, our study indicates a potential lack of investment in safeguarding backend models. This poses questions about the commitment of commercial MLaaS providers to addressing ME threats and implementing protective measures. Considering the ongoing expansion and increasing reliance on MLaaS, it becomes imperative for providers to prioritize security. We advocate for a proactive stance by MLaaS providers to bolster their APIs and fortify the protection of their backend models.

5 ADDITIONAL RELATED WORK

Since the concept of ME attacks against MLaaS APIs was first introduced in [75], which applies path-finding attacks to extract classification (*e.g.*, decision trees) and regression (*e.g.*, regression trees) models, over the years, a plethora of ME attacks have been proposed, training the piracy model to mimic the behavior of the victim model by leveraging data labeled by the victim model [8, 10, 29, 31, 33, 57, 59–65, 71, 75, 76, 82].

Two key metrics, accuracy and fidelity, have been proposed in [29] to measure the performance of ME attacks. Fidelity is generally acknowledged as a more critical metric than accuracy for assessing ME attacks, which is also the main metric used in this study. In addition, we also introduce adversarial fidelity, another important metric to assess the agreement between the victim and piracy models with respect to adversarial examples, complementing fidelity and accuracy.

The number of queries is another important metric for ME attacks. Semi-supervised learning has been explored in [29] to reduce the number of queries. Active learning strategies have also been explored in [10, 59, 60, 65, 71, 75]. For instance, in [60], strategies such as uncertainty, K-center, and DeepFool-based active learning are employed to identify the most informative samples. While the experiments span both image and language-based tasks, they are conducted in controlled laboratory environments and not validated on MLaaS platforms. Other work [31, 61, 65, 82] leverages adversarial examples to optimize the number of queries or improve

attack performance. In addition to these studies, which have focused primarily on classification tasks, [24] explores model extraction attacks on GANs. This work examines whether the conclusions in controlled local environments hold for ME attacks against real-world MLaaS APIs, leading to a number of interesting findings that complement existing studies and provide new insights about the vulnerability of real-world MLaaS APIs.

6 CONCLUSION

In this paper, we report a systematic study on the vulnerability of real-world machine-learning-as-a-service (MLaaS) APIs to model extraction (ME) attacks. The evaluation leads to a number of interesting findings that complement the existing studies conducted in controlled laboratory environments: (*i*) Despite their striding progression over the years, leading MLaaS platforms continue to be highly susceptible to ME attacks. (*ii*) The advances in ML techniques (*e.g.*, optimizers) significantly enhance the adversary's capabilities. (*iii*) Attack techniques (*e.g.*, adversarial learning), proven to be effective in controlled environments, may not necessarily exhibit the same level of effectiveness against real-world MLaaS APIs. (*iv*) The existing mitigation (*e.g.*, output quantization) may weaken ME attacks but remain inadequately effective. These findings shed light on developing MLaaS platforms in a more secure manner.

ACKNOWLEDGEMENTS

The work is supported by the National Science Foundation under Grant No. 1951729, 1953813, 2119331, and 2212323. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] [n. d.]. IMDb Datasets. https://www.imdb.com/interfaces/. Accessed: 2023-03-16.
[2] [n. d.]. Yelp Datasets. https://www.yelp.com/dataset. Accessed: 2023-07-16.
[3] Ulrich Aïvodji, Alexandre Bolot, and Sébastien Gambs. 2020. Model extraction from counterfactual explanations. *ArXiv e-prints* (2020).
[4] Amazon. [n. d.]. AWS Rekognition documentation. https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html.
[5] Emad Barsoum, Cha Zhang, Cristian Canton Ferrer, and Zhengyou Zhang. 2016. Training Deep Networks for Facial Expression Recognition with Crowd-Sourced Label Distribution. In *Proceedings of ACM International Conference on Multimodal Interaction (ICMI)*.
[6] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. MixMatch: A Holistic Approach to Semi-Supervised Learning. *ArXiv e-prints* (2019).
[7] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. 2018. Vggface2: A dataset for recognizing faces across pose and age. In *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition (FG)*.
[8] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting Training Data from Large Language Models.. In *Proceedings of USENIX Security Symposium (SEC)*.
[9] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*.
[10] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. 2020. Exploring connections between active learning and model extraction. In *Proceedings of USENIX Security Symposium (SEC)*.
[11] Lingjiao Chen, Zhihua Jin, Sabri Eyuboglu, Christopher Ré, Matei Zaharia, and James Zou. 2022. HAPI: A Large-scale Longitudinal Dataset of Commercial ML API Predictions. *ArXiv e-prints* (2022).
[12] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. 2023. Symbolic Discovery of Optimization Algorithms. *ArXiv e-prints* (2023).
[13] Yufei Chen, Chao Shen, Cong Wang, and Yang Zhang. 2022. Teacher model fingerprinting attacks against transfer learning. In *Proceedings of USENIX Security Symposium (SEC)*.
[14] Ziheng Chen, Fabrizio Silvestri, Jia Wang, Yongfeng Zhang, and Gabriele Tolomei. 2023. The dark side of explanations: Poisoning recommender systems with counterfactual examples. In *Proceedings of the 46th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2426–2430.
[15] Jacson Rodrigues Correia-Silva, Rodrigo F Berriel, Claudine Badue, Alberto F de Souza, and Thiago Oliveira-Santos. 2018. Copycat cnn: Stealing knowledge by persuading confession with random non-labeled data. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*.
[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv e-prints* (2018).
[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *Proceedings of International Conference on Learning Representations (ICLR)*.
[18] Xianglong Feng, Weitian Li, and Sheng Wei. 2021. LiveROI: Region of Interest Analysis for Viewport Prediction in Live Mobile Virtual Reality Streaming. In *Proceedings of the ACM Multimedia Systems Conference (MMSys)*.
[19] Xueluan Gong, Yanjiao Chen, Wenbin Yang, Guanghao Mei, and Qian Wang. 2021. InverseNet: Augmenting Model Extraction Attacks with Training Data Inversion.. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*.
[20] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision* 129 (2021), 1789–1819.
[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[23] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *ArXiv e-prints* (2015).
[24] Hailong Hu and Jun Pang. 2021. Stealing machine learning models: Attacks and countermeasures for generative adversarial networks. In *Annual Computer Security Applications Conference*. 1–16.
[25] Dong Huang, Qingwen Bu, Yuhao Qing, Yichao Fu, and Heming Cui. [n. d.]. ADVERSARIAL FEATURE MAP PRUNING FOR BACK. ([n. d.]).
[26] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of IEEE Conference on*

Computer Vision and Pattern Recognition (CVPR).
[27] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2018. Densely Connected Convolutional Networks. *ArXiv e-prints* (2018).
[28] Zehao Huang and Naiyan Wang. 2017. Like what you like: Knowledge distill via neuron selectivity transfer. *ArXiv e-prints* (2017).
[29] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. 2020. High accuracy and high fidelity extraction of neural networks. In *Proceedings of USENIX Security Symposium (SEC).*
[30] Edwin T Jaynes. 1957. Information theory and statistical mechanics. *Physical review* 106, 4 (1957), 620.
[31] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. 2019. PRADA: protecting against DNN model stealing attacks. In *Proceedings of IEEE European Symposium on Security and Privacy (Euro S&P).*
[32] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *ArXiv e-prints* (2017).
[33] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2019. Thieves on sesame street! model extraction of bert-based apis. *ArXiv e-prints* (2019).
[34] Alex Krizhevsky. 2014. One weird trick for parallelizing convolutional neural networks. *ArXiv e-prints* (2014).
[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
[36] Seung Hyun Lee, Dae Ha Kim, and Byung Cheol Song. 2018. Self-supervised knowledge distillation using singular value decomposition. In *Proceedings of European Conference on Computer Vision (ECCV).*
[37] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. 2019. Defending against neural network model stealing attacks using deceptive perturbations. In *2019 IEEE Security and Privacy Workshops (SPW).* IEEE, 43–49.
[38] Changjiang Li, Shouling Ji, Haiqin Weng, Bo Li, Jie Shi, Raheem Beyah, Shanqing Guo, Zonghui Wang, and Ting Wang. 2021. Towards certifying the asymmetric robustness for neural networks: quantification and applications. *IEEE Transactions on Dependable and Secure Computing* 19, 6 (2021), 3987–4001.
[39] Changjiang Li, Ren Pang, Zhaohan Xi, Tianyu Du, Shouling Ji, Yuan Yao, and Ting Wang. 2023. An Embarrassingly Simple Backdoor Attack on Self-supervised Learning. In *The 2023 International Conference on Computer Vision (ICCV' 23).*
[40] Changjiang Li, Li Wang, Shouling Ji, Xuhong Zhang, Zhaohan Xi, Shanqing Guo, and Ting Wang. 2022. Seeing is living? rethinking the security of facial liveness verification in the deepfake era. *USENIX Security 2022* (2022).
[41] Changjiang Li, Haiqin Weng, Shouling Ji, Jianfeng Dong, and Qinming He. 2019. DeT: Defending against adversarial examples via decreasing transferability. In *Cyberspace Safety and Security: 11th International Symposium, CSS 2019, Guangzhou, China, December 1–3, 2019, Proceedings, Part I 11.* Springer International Publishing, 307–322.
[42] Shan Li and Weihong Deng. 2019. Reliable Crowdsourcing and Deep Locality-Preserving Learning for Unconstrained Facial Expression Recognition. *IEEE Transactions on Image Processing* 28, 1 (2019), 356–370.
[43] Zhenglin Li, Yangchen Huang, Mengran Zhu, Jingyu Zhang, JingHao Chang, and Houze Liu. 2024. Feature Manipulation for DDPM based Change Detection. *arXiv preprint arXiv:2403.15943* (2024).
[44] Jiacheng Liang, Songze Li, Bochuan Cao, Wensi Jiang, and Chaoyang He. 2021. Omnilytics: A blockchain-based secure data market for decentralized machine learning. *arXiv preprint arXiv:2107.05252* (2021).
[45] Han Liu, Yuhao Wu, Zhiyuan Yu, Yevgeniy Vorobeychik, and Ning Zhang. 2023. Slowlidar: Increasing the latency of lidar-based detection using adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 5146–5155.
[46] Han Liu, Yuhao Wu, Zhiyuan Yu, and Ning Zhang. 2024. Please Tell Me More: Privacy Impact of Explainability through the Lens of Membership Inference Attack. In *2024 IEEE Symposium on Security and Privacy (SP).* IEEE Computer Society, 120–120.
[47] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv e-prints* (2019).
[48] Ziyao Liu, Jiale Guo, Kwok-Yan Lam, and Jun Zhao. 2022. Efficient dropout-resilient aggregation for privacy-preserving machine learning. *IEEE Transactions on Information Forensics and Security* 18 (2022), 1839–1854.
[49] Ziyao Liu, Hsiao-Ying Lin, and Yamin Liu. 2023. Long-term privacy-preserving aggregation with user-dynamics for federated learning. *IEEE Transactions on Information Forensics and Security* (2023).
[50] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *ArXiv e-prints* (2019).
[51] Daniel Lundqvist, Anders Flykt, and A Öhman. 2022. The Karolinska directed emotional faces—KDEF, CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet, 1998. *ArXiv e-prints* (2022).
[52] Weimin Lyu, Xiao Lin, Songzhu Zheng, Lu Pang, Haibin Ling, Susmit Jha, and Chao Chen. 2024. Task-Agnostic Detector for Insertion-Based Backdoor Attacks. *arXiv preprint arXiv:2403.17155* (2024).
[53] Weimin Lyu, Songzhu Zheng, Lu Pang, Haibin Ling, and Chao Chen. 2023. Attention-Enhancing Backdoor Attacks Against BERT-based Models. *arXiv preprint arXiv:2310.14480* (2023).
[54] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *Proceedings of International Conference on Learning Representations (ICLR).*
[55] Sudhanshu Mittal, Maxim Tatarchenko, Özgün Çiçek, and Thomas Brox. 2019. Parting with illusions about deep active learning. *ArXiv e-prints* (2019).
[56] Seong Joon Oh, Bernt Schiele, and Mario Fritz. 2019. Towards reverse-engineering black-box neural networks. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (2019), 121–144.
[57] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
[58] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Prediction poisoning: Towards defenses against dnn model stealing attacks. *arXiv preprint arXiv:1906.10908* (2019).
[59] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2019. A framework for the extraction of deep neural networks by leveraging public data. *ArXiv e-prints* (2019).
[60] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2020. Activethief: Model extraction using active learning and unannotated public data. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI).*
[61] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of ACM Symposium on Information, Computer and Communications Security (AsiaCCS).*
[62] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael P Wellman. 2018. Sok: Security and privacy in machine learning. In *Proceedings of IEEE European Symposium on Security and Privacy (Euro S&P).*
[63] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *Proceedings of IEEE Symposium on Security and Privacy (S&P).*
[64] Nicolas Papernot, Patrick D McDaniel, and Ian J Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *ArXiv e-prints* (2016).
[65] Li Pengcheng, Jinfeng Yi, and Lijun Zhang. 2018. Query-efficient black-box attack by active learning. In *Proceedings of IEEE International Conference on Data Mining (ICDM).*
[66] Zhangyang Qi, Jiaqi Wang, Xiaoyang Wu, and Hengshuang Zhao. 2023. OCBEV: Object-Centric BEV Transformer for Multi-View 3D Object Detection. *arXiv preprint arXiv:2306.01738* (2023).
[67] Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The Annals of Mathematical Statistics* (1951), 400–407.
[68] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *ArXiv e-prints* (2014).
[69] Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *ArXiv e-prints* (2017).
[70] Yi Shi, Yalin Sagduyu, and Alexander Grushin. 2017. How to steal a machine learning classifier with deep learning. In *Proceedings of the IEEE International Symposium on Technologies for Homeland Security (HST).*
[71] Yi Shi, Yalin E Sagduyu, Kemal Davaslioglu, and Jason H Li. 2018. Active deep learning attacks under strict rate limitations for online API calls. In *2018 IEEE International Symposium on Technologies for Homeland Security (HST).*
[72] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *ArXiv e-prints* (2014).
[73] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going Deeper with Convolutions. *ArXiv e-prints* (2014).
[74] Mingxing Tan and Quoc V. Le. 2020. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ArXiv e-prints* (2020).
[75] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing Machine Learning Models via Prediction APIs.. In *Proceedings of USENIX Security Symposium (SEC).*
[76] Jean-Baptiste Truong, Pratyush Maini, Robert J. Walls, and Nicolas Papernot. 2021. Data-Free Model Extraction. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
[77] Binghui Wang and Neil Zhenqiang Gong. 2019. Stealing Hyperparameters in Machine Learning. *ArXiv e-prints* (2019).
[78] Wei Wu, Xiaoyuan Jing, Wencai Du, and Guoliang Chen. 2021. Learning dynamics of gradient descent optimization in deep neural networks. *Science China Information Sciences* 64 (2021), 1–15.
[79] Meilong Xu, Xiaoling Hu, Saumya Gupta, Shahira Abousamra, and Chao Chen. 2023. TopoSemiSeg: Enforcing Topological Consistency for Semi-Supervised Segmentation of Histopathology Images. arXiv:2311.16447 [eess.IV]

[80] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 32.
[81] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[82] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin. 2020. CloudLeak: Large-Scale Deep Learning Models Stealing Through Adversarial Examples. In *NDSS*.
[83] Haichao Zhang, Zhe-Ming Lu, Hao Luo, and Ya-Pei Feng. 2021. Restore DeepFakes video frames via identifying individual motion styles. *Electronics Letters* 57, 4 (2021), 183–186.
[84] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. 2016. From Facial Expression Recognition to Interpersonal Relation Prediction. *ArXiv e-prints* (2016).
[85] Qihua Zhou, Song Guo, Jun Pan, Jiacheng Liang, Zhenda Xu, and Jingren Zhou. 2023. PASS: Patch Automatic Skip Scheme for Efficient Real-Time Video Perception on Edge Devices. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 37.

A MORE DETAILS

A.1 Notations

Notation	Definition
f_v, f_v^{real}, f_v^{hapi}	generic, online, HAPI victim model
f_p	piracy model
$\mathcal{X}$	clean labeled data
$\mathcal{U}$	clean unlabeled data
$\mathcal{A}$	adversarial data

Table 17. Notations and symbols.

A.2 Datasets

Task	Dataset	Size	Description
FER	KDEF	562*762	The exaggerated facial expressions made by subjects
	RAFDB	100*100	Great-diverse facial images downloaded from the Internet
	EXPW	224*224	Wild human facial expressions captured at social events
	FER+	48*48	Black and white human face expressions
NLU	IMDB	234 words	Binary sentiment classification dataset, highly polar movie reviews collected from the internet movie database (IMDB)
	Yelp	140 words	Information from user reviews of different restaurants on Yelp. Ratings can be from 1-5, not a polarized dataset

Table 18. Details of datasets.

A.3 Local experiments

To validate the correctness of our experimental setting, we perform local experiments using the CIFAR10 dataset. We train a victim model using the ground-truth labels and use the same strategy in§3.3.3 to perform ME attack against it. The results are summarized in Table 19, 20, and 21.

Attack	Query Budget (# Batches)					
	4	8	16	32	64	Full
Basic	35.05	50.41	49.46	71.98	73.92	92.88
MixMatch	49.35	58.68	79.21	79.22	83.23	

Table 19. Fidelity of basic and MixMatch-based ME attacks (with batch size fixed as 64). In the case of MixMatch, each labeled batch is paired with an equal-sized unlabeled batch.

Attack	Query Budget (# Batches)					
	8	16	32	64	128	Full
Basic	47.85	48.12	68.84	76.63	83.12	92.88
ActiveThief	52.01	58.57	74.22	80.53	86.06	

Table 20. Fidelity of basic and active learning-based ME attacks (with batch size fixed as 64).

Query Batches	Attack		
	Basic	PGD	CW
8	33.94	36.87	35.67
16	41.5	45.16	46.76
32	66.78	69.81	71.2

Table 21. Fidelity of basic and adversarial learning-based ME attacks (with batch size fixed as 64).