Transfer Attacks Revisited: A Large-Scale Empirical Study in Real Computer Vision Settings

Yuhao Mao*†, Chong Fu*, Saizhuo Wang*, Shouling Ji*(✉), Xuhong Zhang*‡(✉),
Zhenguang Liu**, Jun Zhou§, Alex X. Liu§, Raheem Beyah¶, Ting Wang‖
*Zhejiang University, †ETH Zürich, ‡Zhejiang University NGICS Platform, **Zhejiang Gongshang University, §Ant Group,
¶Georgia Institute of Technology, ‖Pennsylvania State University
{yuhaomao, fuchong, szwang, sji, zhangxuhong}@zju.edu.cn, liuzhenguang2008@gmail.com, zhoujun@antfin.com,
alexliu@antgroup.com, rbeyah@ece.gatech.edu, inbox.ting@gmail.com

***Abstract*—One intriguing property of adversarial attacks is their "transferability" – an adversarial example crafted with respect to one deep neural network (DNN) model is often found effective against other DNNs as well. Intensive research has been conducted on this phenomenon under simplistic controlled conditions. Yet, thus far there is still a lack of comprehensive understanding about transferability-based attacks ("transfer attacks") in real-world environments.**

To bridge this critical gap, we conduct the first large-scale systematic empirical study of transfer attacks against major cloud-based MLaaS platforms, taking the components of a real transfer attack into account. The study leads to a number of interesting findings which are inconsistent to the existing ones, including: (*i*) Simple surrogates do not necessarily improve real transfer attacks. (*ii*) No dominant surrogate architecture is found in real transfer attacks. (*iii*) It is the gap between posterior (output of the softmax layer) rather than the gap between logit (so-called κ value) that increases transferability. Moreover, by comparing with prior works, we demonstrate that transfer attacks possess many previously unknown properties in real-world environments, such as (*i*) Model similarity is not a well-defined concept. (*ii*) L_2 norm of perturbation can generate high transferability without usage of gradient and is a more powerful source than L_∞ norm. We believe this work sheds light on the vulnerabilities of popular MLaaS platforms and points to a few promising research directions. [1]

I. Introduction

Deep neural networks (DNNs) achieve tremendous success in a variety of application domains [20], [47], but they are inherently vulnerable to adversarial examples (AEs) which are malicious samples crafted to deceive target DNNs [26], [33], [35], [44]. This vulnerability significantly hinders their use in security-sensitive domains.

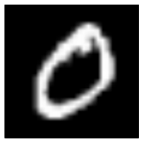
Among the many properties of AEs, the transferability, that an AE crafted with respect to one DNN also works against other DNNs, is particularly intriguing. Leveraging this property, the adversary could forge AEs using a surrogate DNN to attack the target DNN without knowledge of the target. This is highly dangerous because an increasing number of cloud-based MLaaS platforms are deploying DNNs with public API. Furthermore, given that years' advances in technology did not eliminate this vulnerability, it is highly likely that transferability stems from the intrinsic properties of DNNs. Understanding the phenomenon improves the interpretability of DNNs in its own right as well.

Therefore, understanding the deciding factors of transferability and their working mechanisms has attracted intensive researches [15], [19], [29], [34], [36], [39], [44]. However, to the best of our knowledge, all systematic empirical studies are conducted under controlled "lab" environments that are too ideal to make the derived conclusions reliable in the real environment. For instance, many studies [36], [37] give the adversary access to the training data of the target model, and some others [41] discuss surrogates with similar complexity to the target model. Although there are studies that attack deployed DNNs known as cloud models [29], their conclusions are neither systematic nor informative enough, suffering from the lack of adequate observations to do statistical tests and suitable metrics to account for the difference between lab and real settings. The difference between lab and real environment includes:

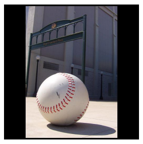
- **Complexity and architecture of the target.** In the real environment, neither the target's complexity nor its architecture is known to the attacker. The cloud model could be arguably far more complicated than a surrogate and of an uncommon architecture. Under this circumstance, the conclusions derived from academic target models may not hold, calling for thorough examinations.
- **Training of the target.** In the real environment, the adversary knows nothing about the training details of the target, including the optimization hyperparameters and algorithms. Additionally, the training datasets are far more complicated and noisy in the real environment[2], increasing the difficulty in simulating a similar training environment at local.
- **Structure of the input.** The cloud models are designed for high resolution images. Therefore, the cloud models may not produce meaningful outputs for inputs from academic datasets that are extremely low resolution like MNIST [25] and CIFAR10 [22], as shown in Figure 1a. Conclusions made on these toy datasets may not hold under this condition.
- **Structure of the output.** Cloud models usually return multiple predictions and their corresponding confidences,

[1]Code & Results: https://github.com/AlgebraLoveme/Transfer-Attacks-Revisited-A-Large-Scale-Empirical-Study-in-Real-Computer-Vision-Settings

[2]Amazon declares they have access to billions of images daily and continue to learn from new data [3].



(a) 97.2% Text; 96.5% Number; 96.5% Symbol.



(b) 91.7% Sports; 86.7% Sphere; 78.9% Baseball.

Figure 1: AWS Rekognition's predictions to two sample images. The left one does not contain the desired prediction (digit zero) and the right one's ground truth label (baseball) does not get the highest score. They are not mistakes but metrics like top-k accuracy would consider them as not good.

which are different from the logits returned by classifiers in lab settings (*multiple returns problem*). In addition, cloud models usually have a significantly larger set of labels[3] that are hard to estimate and categorize locally (*class inconsistency problem*). These differences, as depicted in Figure 1, add to the difficulty in measuring the effectiveness of transfer attacks in a real-world scenario.

To address these limitations, we conduct a large-scale systematic empirical study on transfer attacks in the real settings. To comprehensively explore the possible factors that may affect the transferability, we combine different settings of the attack components at local environment where AEs are generated. To thoroughly evaluate the effectiveness of transfer attacks and the robustness of real-world models, we select four leading cloud-based Machine-Learning-as-a-Service (MLaaS) platforms, namely Aliyun (Alibaba Cloud), Baidu Cloud, AWS Rekognition and Google Cloud Vision as the targets.

Our contributions include:

- We identify the main gaps of evaluating transfer attack in the real world, namely *multiple returns problem* and *label inconsistency problem*, and extend existing metrics to address these difficulties. To the best of our knowledge, these are the first metrics that can be reasonably applied to analyze transfer attack on MLaaS systems.
- We conduct a systematic evaluation on the transferability of adversarial attacks against four leading commercial MLaaS platforms on two computer vision tasks: object classification and gender classification, using datasets consisting of real-world pictures in order to get meaningful insights (ImageNet [16] and Adience [18], respectively). Based on the results of our evaluation, we measure the robustness of the discussed platforms and point out that while on average transfer attack performs poorly, they can be systematically designed to achieve stable success rates, *e.g.*, use FGSM for the object classification. More detailed guidelines for the industry is provided in Appendix X-F. We hope this will help the industry fortify their models and improve their services.
- We explore the possible factors that may affect the transferability using 180 different settings as the combination of components in a transfer attack. 36,000 AEs in total generated from 200 seed images for each task are evaluated, making the conclusions statistically adequate. We find that some former conclusions can be generalized to the real settings while some others are inapplicable. A comparison between the former conclusions and their real-world counterparts is shown in Table I. Additionally, we demonstrate that transfer attacks possess many previously unknown properties in real-world environments. We believe that our findings will provide new insights for future research.

[3]Google shares an open dataset consisting of about 60M labels [7].

II. Background

In this section, we introduce the preliminaries of transfer attacks, including the components that may affect the transferability.

A. Transfer Attack

Transfer attack involves *adversarial attack* and *transferability*. Specifically, a classifier with parameters θ, namely f_θ, accepts an input $x \in R^n$ and makes a prediction $f_\theta(x)$. The adversary wants to generate a maliciously perturbed input $\hat{x} = x + \delta$ so that $f_\theta(\hat{x}) \neq f_\theta(x)$. Typically, δ is required to be human imperceptible, *i.e.*, the size of δ is small under L_p norm. Attack algorithms of this form are called *adversarial attacks* [12], [19], [27], [30], [32], [38], [44], [46].

Formally, if an AE $\hat{x}$ crafted on f_θ deceives another model $\phi_{\theta'}$ with unknown parameters and architecture, *i.e.*, $\phi_{\theta'}(\hat{x}) \neq \phi_{\theta'}(x)$, then we say this AE *transfers* from model f_θ to $\phi_{\theta'}$. Utilizing this property, an adversary may generate AEs based on f_θ and transfer them to the target black-box model $\phi_{\theta'}$. Typically, the label sets of the source model and the target model are the same. This process is called *transfer attack* [29], [36], [37]. Although query-based attacks [13], [21] that request partial information about the targets are more powerful, transfer attacks do not require any information about the target and thus are more stealthy and economic.

According to the process of transfer attack, there are three major components affecting the transferability of AEs: *surrogate model*, *surrogate dataset* and *adversarial algorithm*. We elaborate on these components next.

B. Surrogate Model

Surrogate model is the model on which the adversary performs white-box attacks. In general, it is expected to resemble the target model to increase the transferability of the generated AEs. The main factors affecting the similarity include:

(a) *Pretraining*. Target models, such as the ones on MLaaS platforms, are trained on a sufficiently large dataset to achieve a superior performance. Since attackers may not have enough computing resources to train the surrogate model on a large dataset, they may fine-tune the surrogate model based on a public model that is pretrained on a large dataset, as pretraining generally improves a model's accuracy [17], [50].

(b) *Model Architecture.* Various DNN architectures have been designed to address different problems and improve performance [20], [40], [43]. The fitting functions that models learned under different architectures could be different even with the same training data, which might influence the transferability of generated AEs.

Table I: Conclusions comparison between lab settings and real settings.

Former conclusion in labs	Refined conclusion in reality	Consistency
Untargeted Attacks are easy to transfer and targeted attacks almost never transfer. [29]	All kind of transfer attacks discussed has a significantly positive success rate.	Partial
FGSM $\geq$ *PGD* $\geq$ *CW* in the sense of transferability. [41]	(1) Targeted algorithms are worse than untargeted algorithms in both targeted and untargeted transfer attack. (2) Single-step algorithms are better than iterative algorithms.	✓
Surrogates with less complexity, defined by smaller variability of the loss landscape, are better choices for transfer attack. Experimentally, a surrogate with simpler structure and stronger regularization is better. [15]	The effect of surrogate complexity, defined by the number of layers and parameters of the neural network, is non-monotonic. A surrogate with a good depth can outperform simpler and deeper surrogates.	✗
AEs crafted from VGG surrogates transfer well to all targets, while others almost never transfer to targets in different families. [41]	No dominant architecture is found to craft AEs that transfer better to cloud models.	✗
Relaxing L_∞ norm perturbation constraint largely increases transferability. [41]	L_2 norm of perturbation has far stronger correlation with transferability. Increasing L_2 norm while keeping L_∞ fixed makes good transferability but large L_∞ norm with small L_2 norm can give poor success rate.	Partial
A larger logit gap between the adversarial class and the second likely class, called κ value, increases untargeted attack transferability. [12] [41]	The gap between the posterior of logits is a better representative for measuring transferability. A large κ cannot increase transferability in many cases.	✗

(c) *Model Complexity.* For one specific architecture, models with different complexities usually have different performance. This phenomenon indicates that they have different capabilities in capturing the distribution of the training data, thus affecting the transferability of AE when used as surrogate models. In this paper, since we are in the context of neural networks, we define the model complexity by the number of layers and parameters. On the contrary, Demontis *et al.* [15] define the model complexity by the variability of the loss landscape and the input gradient size. We experimentally confirm their conclusions on the input gradient size on local targets because the gradients of the MLaaS models are non-transparent. For more details, please refer to Appendix X-J. The variability of the loss landscape is computationally expensive for the large surrogates we use, thus we do not use this as the complexity metric.

C. Surrogate Dataset

Surrogate dataset is the dataset used to train the surrogate model. In general, the distribution of the surrogate data is expected to approximate the distribution of the data used to train the target. In this way, the similarity between the surrogate and the target model is expected to increase so that the AEs crafted on the surrogate are easier to transfer. However, the surrogate datasets that an attacker possesses are usually much smaller than those owned by the MLaaS systems. To make the surrogate dataset capture the distribution of the target model's training data as well as possible, there are two common data enrichment methods:

(a) *Data Augmentation.* Data augmentation is to apply transformations on an image while maintaining the primary patterns critical to the classification. Applying augmentation enforces a model to learn patterns under extended situations and focus on the dominant patterns.

(b) *Adversarial Training.* Adversarial training is to enrich the training data with their adversarial counterparts so that the surrogate model can focus on robust features which are not changeable by small perturbations [23], [45].

These methods allow the surrogate to learn better features from the limited surrogate dataset.

D. Adversarial Algorithm

Adversarial algorithm is the white-box attack algorithm (WBA) used to generate AEs on the surrogate model. Among the WBAs, there are several common properties separating them into different categories:

(a) Depending on the goal, WBAs can be categorized into *targeted* and *untargeted*. Targeted algorithms aim to let the model predict the target class, *i.e.*, $f_\theta(\hat{x}) = c_t$, where c_t denotes the target class. Untargeted algorithms merely want the model to misclassify, *i.e.*, $f_\theta(\hat{x}) \neq c_o$, where c_o denotes the original predicted class.

(b) Depending on the optimization process, WBAs can be categorized into *single-step attacks* which add perturbations to the original input only once, and *iterative attacks* which perturb the original input repeatedly until a certain condition is satisfied.

III. Threat Model

Figure 2 provides an overview of transfer attacks on MLaaS platforms. The model structure and training dataset of the target platform model is obscure to an attacker. Additionally, an attacker can only access the target platform model by sending images to the MLaaS platform and getting their predictions via the platform's API. Therefore, an attacker can only manipulate the input of the target platform model to perform an attack. For ease of understanding, we formally present the following attack-related definitions: (1) *targeted AE* is defined as the AE generated locally via targeted algorithms; (2) *untargeted AE* is defined as the AE generated locally via untargeted algorithms; (3) *targeted transfer attack* is defined as the targeted attack against the MLaaS platforms using locally generated AEs; (4) *untargeted transfer attack* is defined as the untargeted attack against the MLaaS platforms using locally generated AEs;

Following Chen *et al.* [14], we consider the case that attackers maintain a pool of highly transferable AEs as well. Under such condition, we are actually viewing transferability as a property of AEs and every AE can be used to launch any kind of transfer attack. It immediately leads to the following two key differences: (1) Both targeted and untargeted AEs can

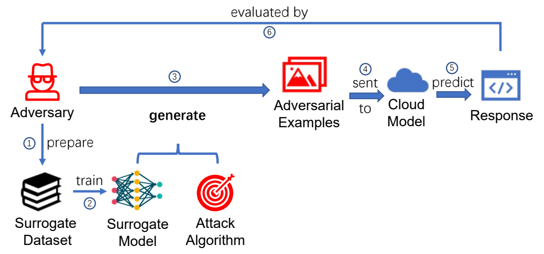


Figure 2: Pipeline of transfer attack against MLaaS platforms.

Table II: Settings for each component of the surrogate model.

Component	**Settings**
Architecture	Inception, VGG, ResNet
Complexity	(ResNet Depth) 18, 34, 50
Pretraining	True, False

be used to perform targeted and untargeted transfer attacks. For targeted transfer attacks, we take the predicted label of an AE by the surrogate as the target label. (2) The correlation between transferability and sample-level properties such as adversarial confidence and perturbation size can be easily examined.

IV. Evaluation Settings and Metrics

A. Evaluation Settings

1) Surrogate Model Settings: For each surrogate model, we evaluate multiple settings for each of its components discussed in Section II-B. We use the implementations of models in PyTorch [8] and TorchVision [9] libraries to conduct our experiments. For different model complexities, we apply three ResNet models with different depths: ResNet-18, ResNet-34 and ResNet-50. These depths are chosen because they are officially implemented. We apply Inception V3 [42], VGG-16 [40] and ResNet-18 [20] to study the impact of architecture. For pretraining, we compare the raw ResNet models without pretraining and the ResNet models pretrained on the whole ImageNet dataset. For the object classification task, we only finetune the last fully connected layer and fix other layers. For the gender classification task, we use the pretrained parameters as an initialization and finetune the whole model. Empirical result shows all pretrained surrogates have a significant improvement on accuracy. Table II summarizes each component.

2) Surrogate Dataset Settings: For the image classification task, we use a subset of ImageNet [16] consisting of 10 classes. Each class contains approximately 500 images cropped to 512×512×3. It is not an important reduction in the number of samples because these classes have roughly 500-600 images per class in the original ImageNet data. We do this because the number of targets' classes is less meaningful in attacking, while the structure of the data is critical, as shown in Figure 1. Training the surrogate with a subset is to lower the cost and have a better simulation of real attacks with limited resources. Similarly, for gender classification, we use a subset of the Adience dataset [18]. We randomly pick 10,000 images from the Adience dataset with half male and half female and crop them to 384×384×3. For each of the two surrogate datasets, we randomly split it into training set, validation set and test set with the split ratio 8:1:1.

We apply the two data enlargement techniques discussed in Section II-C. Specifically, for data augmentation, we use color jitter, random affine, random horizontal flip, random perspective, random rotation and random vertical flip. They are imposed on an image independently with a probability of 0.5. The detailed parameter settings can be found in Appendix X-G. For adversarial training, we employ the naive adversarial training (NAT) algorithm [23]. Thus, we have three variants for each dataset: *raw* (the original dataset), *augmented* and *adversarial*.

3) Adversarial Algorithm Settings: In total, we employ nine representative adversarial attacks. Among them there are two targeted algorithms, BL-BFGS (simplified as BLB) [44] and CW2 [12]. BLB and CW2 generate AEs by solving the corresponding optimization problem using iterative methods. Adversarial perturbations generated by them are optimized in size and thus by nature small. Apart from these targeted algorithms, we employ six untargeted adversarial algorithms as well. DeepFool [32] is the only untargeted algorithm that optimize the perturbation size, realized by approximating the decision boundary. A generalized version of DeepFool is the UAP algorithm [31], adding image-level adversarial perturbations given by DeepFool together to obtain a universal adversarial perturbation. Algorithms remained are built upon another idea that AEs should be generated with L_p norm of perturbation less than a predetermined norm budget. The root algorithm is FGSM [19] which perturbs the original image for once along the opposite direction of the gradient, *i.e.*, $\hat{x} = x - \epsilon \cdot sign(\nabla loss(f_\theta(x); c)$, where $loss(f_\theta(x); c)$ is the loss function between the model prediction and the ground truth, $sign(\cdot)$ denotes the sign function and ϵ denotes the perturbation budget. FGSM has its improved version RFGSM [46] which adds a random perturbation to the image before imposing gradient-based perturbation. This preprocessing is claimed to penetrate the defense of gradient masking. In addition, FGSM has an iterative counterpart PGD [30] which executes the FGSM step iteratively for predetermined times. Another variation of FGSM called Step-LLC and its iterative counterpart LLC [24] minimize the loss w.r.t. the least likely class rather than maximize the loss w.r.t. the correct class. The perturbing process then becomes $\hat{x} = x + \epsilon \cdot sign(\nabla loss(f_\theta(x); c^*)$, where c^* denotes the class the model predicts with the least confidence. These attacks have L_p norm of adversarial perturbation less than or equal to the budget. We use L_∞ norm in the paper.

For algorithm implementations, we employ the codes from the open source framework DEEPSEC [28]. We select ten independent classes from the ImageNet and randomly sampled 20 images for each class as the seed image. For gender classification, we randomly select 200 original images as the seed image, half male and half female. The combinations of surrogate settings and attack algorithms form 180 different settings in total for our evaluation. Under each setting, AEs are generated on the same set of seed images and sent to the target platform. Then metrics are computed based on the responses.

4) Cloud Experiment Settings: We conduct our experiments on four leading commercial MLaaS platforms: Google Cloud Vision [6], AWS Rekognition [2], Aliyun (Alibaba Cloud) [1] and Baidu Cloud [4]. We send the AEs to these clouds using the official API provided by each platform and save the responses for evaluation.[4]

B. Evaluation Metrics

It is not trivial to determine whether a transfer attack is successful in real settings for the multi-class classification tasks like ImageNet. Basically, a MLaaS platform's response to a certain input image can be denoted as $P = \{(l, s)|l \in L, s \in [0, 1]\}$, where l represents a label from the class set L on the MLaaS platform and s denotes the confidence score for the prediction l. Consequently, two challenging problems come up in matching a local class with the response from a MLaaS platform:

(a) *Label inconsistency problem*: Since L is significantly different from the local class set C, l cannot be directly associated with a local label c because l can either be a sub-class or super-class of c. For example, the local class "weapon" may correspond to classes in L that are sub-classes of it, such as "gun" and "knife", and the local class "baseball" may be predicted as a super-class, such as "sport". Additionally, L differs across MLaaS platforms as well.

(b) *Multiple predictions problem*: While P consists of multiple predictions, the ground truth is a single label. Hence, it is difficult to fairly judge whether the MLaaS platform is correct, since the expected predication might have a fairly high score but is not a top-k label, as what we have seen in Figure 1b. As a result, metrics like top-k accuracy can not be reasonably applied because it only considers the prediction with the top k scores.

To address the label inconsistency problem, we construct a class mapping $M_c = \{l_c|l_c \in L\}$ for each local class c and each MLaaS platform. The content of the constructed mapping can be found in the code repository. To construct M_c, we inspect all the responses of a MLaaS platform to the original image belonging to class c and manually select the relevant classes from the responses. We then build an equivalence dictionary $T = \{(c, M_c)|c \in C\}$ for each MLaaS platform. We discuss the validity and potential bias introduced by the equivalence dictionary in Appendix X-A.

To address the multiple predictions problem, we select a confidence threshold σ for each platform to filter out the predictions with low scores. The value of the threshold σ is critical to the fairness of our evaluation and should be meticulously chosen. Our principle is that the prediction accuracy for clean images should not drop significantly and low confidence predictions are ruled out as many as possible. To set an appropriate threshold for each cloud platform, we measure the prediction accuracies of the studied cloud platforms on the original ImageNet data under different thresholds. As shown

[4]We exclude Google for gender classification because it does not support this function.

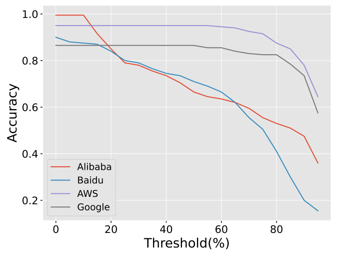


Figure 3: The accuracies of clouds with respect to different confidence thresholds.

in Figure 3, the threshold value largely affects the prediction accuracy and the trend varies across platforms. We find that Google and AWS mainly set high scores for their responded predictions, while Alibaba and Baidu attach low scores to many responded predictions. Therefore, we use $\sigma = 50\%$ for AWS and Google and $\sigma = 10\%$ for Alibaba and Baidu. The difference in threshold mitigates the unfairness of setting a global confidence threshold to evaluate their robustness and leaves the analysis of other factors, assumed to be independent to the target model, unharmed. More discussions about threshold cutting is provided in Appendix X-I.

Based on the discussions above, the matching between a local class c and a response P from a MLaaS platform can be formalized as: c match $P \Leftrightarrow \exists(l, s) \in P, s > \sigma \wedge l \in M_c$, where M_c is the equivalence dictionary of c. For an AE with an original label c_o but classified to be c_t by the surrogate model, we call it *misclassified* on the MLaaS platform if c_o fails to match the response P and *matched* if c_t matches P. Note that since it is possible that both the correct label and the target label are present in the response, a matched AE is not necessarily misclassified. Therefore, there is no strict relationship between the misclassification rate and the matching rate.

Following the extended definition of misclassified and matched AEs, We further introduce two metrics for evaluating the effectiveness of transfer attacks, which is aligned to the previous studies in the lab settings. In short, transfer attack aims to find "similar mistakes" and what we do is to redefine what a mistake is. Note that the class mapping technique and threshold cutting are used to evaluate the transfer success rate rather than launching attacks.

Definition 1. We define misclassification rate to be the number of AEs that are misclassified on the MLaaS system divided by the number of AEs sent to the MLaaS system. Similarly, matching rate is defined to be the number of AEs that are matched on the MLaaS system divided by the number of AEs sent to the MLaaS system.

For gender classification, every cloud has a different response format. Following what we have done to ImageNet, we take the prediction with the higher confidence as the final prediction. Since gender is binary, the misclassification rate and matching rate degenerate into the same one. Thus, we further examine the Male2Female rate (M2F rate) and Female2Male rate (F2M rate).

Definition 2. We define M2F rate to be the number of misclassified male AEs divided by the number of male AEs sent to the MLaaS system and F2M rate to be the number of misclassified female AEs divided by the number of female AEs sent to the MLaaS system.

V. Results and Analysis

This section is divided into two parts, discussing two kinds of influencing factors respectively. The first part is about factors concerning the threat setting, *i.e.*, platform factors, surrogate factors, adversarial algorithm factors and their joint effects. In this part, we run hierarchical ordinary least squares (OLS) regressions, on the results obtained from the ImageNet and the Adience. These regressions show how threat setting factors influence the transferability metrics, leading to a bunch of empirical observations. The reason and implication of using OLS and correlation analysis is discussed in Appendix X-B and X-C. The second part is about the factors concerning the properties of AE. In this part, we consider how the norm of adversarial perturbation, the adversarial confidence and the classification hardness affect the transferability.

The observations stated in the first part, if not explicitly claimed, are obtained from the ResNet surrogates. They generalize if we presume the impact of surrogate architecture to transfer attacks is additively separable[5] to the impact of other discussed factors. For ease of writing, we use subscript for relations to show the p-value of relation tests and "$\approx$" to show the null hypothesis that two values are equal cannot be rejected ($p > 0.1$). For example, $A <_{p=0.01} B$ means that the null hypothesis $A \geq B$ is rejected with p-value 0.01 and $B \approx C$ means that null hypothesis $B = C$ cannot be rejected. When we write compound inequalities, the "$\approx$" is treated as equivalence relation and statistical tests are done on each of them respectively. It means that $A <_{p=0.01} B \approx C$ confirms $A <_{p\leq 0.01} C$ as well. In addition, we use the variable name to refer to its coefficient in the regression.

A. Threat Setting Factors

As discussed in Section IV-A, target models on MLaaS platforms, surrogates and adversarial algorithms are the main factors in the threat setting. We use hierarchical regression to decompose the effect of these factors. Due to the large cost of adding the architecture dimension to all the experiments, we cannot run all tests on various surrogate architectures and thus the OLS regression does not include the architecture dimension in order to balance the data and avoid biased conclusions. Instead, we include a separate study of the impact of the surrogate architecture.

Table III shows the OLS regression result on the ImageNet dataset. Each column (regression group) is a different OLS regression with different independent variable (IV) group. For each column, variables left blank are not included in the regression. The table follows a standard representation of hierarchical regression and readers who are unfamiliar with this representation can find more explanations on how to read this table in Appendix X-D. Detailed codes for conducting the OLS analysis is included in the released code repository. Regression $\mathbb{A}$ and $\mathbb{F}$ decompose the effect of target models, revealing how well different target MLaaS platforms behave to defend transfer attacks. Regression $\mathbb{B}$ and $\mathbb{G}$ further decompose the effect of pretraining and data enrichment factors, intending to reveal how surrogate training affects the transferability. Adversarial algorithm factors' effect are further decomposed in regression $\mathbb{C}$ and $\mathbb{H}$. In regression $\mathbb{D}$ and $\mathbb{I}$, surrogate depth's effect is decomposed as well. In the result of these regressions, data enrichment and surrogate depth are not clearly correlated to transfer attack, thus regression $\mathbb{E}$ and $\mathbb{J}$ are designed to reveal if there are joint effects between surrogate model setting and training techniques. Table IV shows the result on the Adience dataset, and is organized similarly.

In the regression, multicollinearity exists among platform factors, adversarial algorithm factors and surrogate depth factors. For example, one and only one of the platform factors is 1, which makes their coefficients not unique in the regression. To avoid this problem, we choose a baseline setting: UAP attack for adversarial algorithm factors and ResNet-18 for surrogate depth factors. This means that for the settings with UAP attack, all indicator variables for the adversarial algorithms are zero. Similarly, for the settings with ResNet-18 surrogate, all indicators for the surrogate depth are zero. No baseline platform is set because the constant term of the regression can be merged to the platform factors.

1) Platform Factors: Platform factors represent the vulnerability of the target model. A larger coefficient for a platform means that transfer attacks are more likely to succeed on this platform. However, as we use a restricted range of classes and a manually designed class mapping to decide whether a transfer attack is successful (see Section IV-B for details), the following discussions do not provide any guarantee nor comparison for the robustness of these platforms when different settings are applied. We are careful not to generalize our conclusions, and readers are discouraged to compare the MLaaS systems by their performance in these specific tasks and datasets.

Regression $\mathbb{A}$ and Regression $\mathbb{F}$ decomposes the platform factors in Table III and Table IV, respectively. Since these two regressions run on all settings but only include platform factors in the IV group, the coefficient of the platform factors represents an average of metric values over all settings. In other words, if attackers pick their settings uniformly at random, then they are expecting the coefficient as their metric value. This provides us with some knowledge about "inexperienced attackers" and is a good representation of the robustness of the target model.

By comparing the coefficients, we get the following relations (lower is better):

(i) In the object classification task, Aliyun $\approx$ AWS $<_{p<0.001}$ Baidu $<_{p<0.001}$ Google on the misclassification rate, and Baidu $<_{p=0.068}$ AWS $<_{p<0.001}$ Aliyun $\approx$ Google on the matching rate.

[5]For the definition of additively separation, please refer to [5].

Table III: The OLS regression result of misclassification rate and matching rate with regard to different groups of factors on the data obtained from ResNet surrogates in the image classification task. For each entry, the first number is the regression coefficient and the second number, presented in parentheses, is the corresponding standard deviation. Variables with "is_" as prefix are boolean virtual variables that take the value 1 iff the corresponding conditions are true. Uninteresting values are grayed since they do not contain additional information.

	misclassification rate					matching rate				
IV \ Group	$\mathbb{A}$	$\mathbb{B}$	$\mathbb{C}$	$\mathbb{D}$	$\mathbb{E}$	$\mathbb{F}$	$\mathbb{G}$	$\mathbb{H}$	$\mathbb{I}$	$\mathbb{J}$
is_Google	0.232*** (0.004)	0.236*** (0.006)	0.235*** (0.005)	0.190*** (0.005)	0.234*** (0.006)	0.100*** (0.003)	0.089*** (0.004)	0.090*** (0.005)	0.090*** (0.006)	0.082*** (0.007)
is_AWS	0.071*** (0.004)	0.075*** (0.006)	0.074*** (0.005)	0.047*** (0.005)	0.073*** (0.006)	0.038*** (0.003)	0.027*** (0.004)	0.028*** (0.005)	0.027*** (0.006)	0.020*** (0.007)
is_Baidu	0.193*** (0.004)	0.197*** (0.006)	0.196*** (0.005)	0.200*** (0.005)	0.194*** (0.006)	0.029*** (0.003)	0.019*** (0.004)	0.020*** (0.005)	0.019*** (0.006)	0.011* (0.007)
is_Aliyun	0.064*** (0.004)	0.068*** (0.006)	0.067*** (0.005)	0.102*** (0.005)	0.065*** (0.006)	0.097*** (0.003)	0.086*** (0.004)	0.087*** (0.005)	0.086*** (0.006)	0.078*** (0.007)
is_pretrained		-0.018*** (0.004)	-0.018*** (0.002)	-0.048*** (0.005)	-0.014*** (0.005)		0.020*** (0.003)	0.020*** (0.003)	0.020*** (0.003)	0.029*** (0.006)
is_adversarial		0.010* (0.005)	0.010* (0.003)	-0.003 (0.006)	0.005 (0.006)		0.007* (0.004)	0.007** (0.003)	0.007** (0.003)	0.022** (0.007)
is_augmented		0.004 (0.005)	0.004 (0.003)	0.010 (0.006)	0.014** (0.006)		-0.006 (0.004)	-0.006* (0.003)	-0.006* (0.003)	0.003 (0.007)
is_PGD			0.003 (0.005)	0.003 (0.005)	0.003 (0.005)			0.020*** (0.006)	0.020*** (0.006)	0.020*** (0.006)
is_FGSM			0.073*** (0.005)	0.073*** (0.005)	0.073*** (0.005)			0.021*** (0.006)	0.021*** (0.006)	0.021*** (0.006)
is_BLB			-0.048*** (0.005)	-0.048*** (0.005)	-0.048*** (0.005)			-0.025*** (0.006)	-0.025*** (0.006)	-0.025*** (0.006)
is_CW2			-0.047*** (0.005)	-0.047*** (0.005)	-0.047*** (0.005)			-0.026*** (0.006)	-0.026*** (0.006)	-0.026*** (0.006)
is_DEEPFOOL			-0.050*** (0.005)	-0.050*** (0.005)	-0.050*** (0.005)			0.015*** (0.006)	0.015*** (0.006)	0.015*** (0.006)
is_STEP_LLC			0.078*** (0.005)	0.078*** (0.005)	0.078*** (0.005)			-0.010* (0.006)	-0.010* (0.006)	-0.010* (0.006)
is_RFGSM			-0.000 (0.005)	-0.000 (0.005)	-0.000 (0.005)			0.024*** (0.006)	0.024*** (0.006)	0.024*** (0.006)
is_LLC			0.002 (0.005)	0.002 (0.005)	0.002 (0.005)			-0.028*** (0.006)	-0.028*** (0.006)	-0.028*** (0.006)
is_depth_34				0.002 (0.003)	0.001 (0.006)				0.002 (0.003)	0.006 (0.007)
is_depth_50				0.002 (0.003)	-0.003 (0.006)				0.001 (0.003)	0.007 (0.007)
is_pre×adv					-0.003 (0.006)					-0.016** (0.007)
is_pre×aug					-0.014** (0.006)					-0.011 (0.007)
is_pre×depth34					-0.004 (0.006)					0.001 (0.007)
is_pre×depth50					-0.010* (0.006)					-0.001 (0.007)
is_adv×depth34					0.012* (0.007)					-0.005 (0.008)
is_adv×depth50					0.007 (0.007)					-0.013 (0.008)
is_aug×depth34					-0.001 (0.007)					-0.006 (0.008)
is_aug×depth50					-0.0206 (0.007)					-0.004 (0.008)
R^2	0.645	0.656	0.899	0.899	0.902	0.382	0.430	0.587	0.587	0.594
Adjusted R^2	0.644	0.653	0.896	0.896	0.898	0.379	0.425	0.578	0.577	0.578

$^{*}p < .1$, $^{**}p < .05$, $^{***}p < .01$. Number of observation is 648.

(ii) In the gender classification task, AWS $<_{p<0.001}$ Baidu $<_{p=0.032}$ Aliyun on the F2M rate, and AWS $<_{p=0.013}$ Aliyun $<_{p<0.001}$ Baidu on the M2F rate.

It shows that although these platforms have similar accuracies on the clean dataset (see Figure 3 for details), their robustness against transfer attack varies. In particular, the ranking of the robustness is not the same to the ranking of their accuracy. This result suggests that model accuracy does not necessarily guarantee robustness against transfer attack, even in the real applications. In addition, no single platform has superior robustness in different kinds of transfer attacks (untargeted vs targeted, F2M vs M2F), although AWS is the best in the gender classification. In particular, when compared with other platforms, Aliyun is good at defending the untargeted attacks but not the targeted attacks. On the contrary, Baidu is good at defending the targeted attcks but not the untargeted attacks. This means that a model's robustness cannot be naively measured by its robustness against specific kind of attacks. Therefore, since the coefficients are significantly positive, some even exceeding 0.2, the platforms should take transfer attack more seriously due to their ignorable marginal cost because even inexperienced attackers can get over 20% work done with a very small cost.

Furthermore, we observe that the matching rate is significantly positive for all the platforms. This is different to the conclusion of Liu *et al.* [29] that targeted transfer attack almost never transfer, by showing that targeted transfer attack can succeed in the real applications.

Observation 1. *In the real transfer attack, the difficulty of attacking a target model is not directly related to its accuracy,* i.e.*, a target with higher accuracy is possible to be more vulnerable to transfer attacks. No single platform has superior robustness in different kinds of transfer attacks (untargeted vs targeted, F2M vs M2F). Therefore, the threat of transfer attacks in the real applications should be treated seriously because it has a non-trivial success rate and a low cost.*

2) Pretraining and Surrogate Dataset Factors: Arguably, pretraining plays an important role in training a decent model with few efforts, thus is attempting to be applied to train

Table IV: The OLS regression result of F2M rate and M2F rate with regard to different groups of factors on data obtained from ResNet surrogates in the gender classification. This table is formatted similarly to Table III.

Group / IV	F2M rate					M2F rate				
	$\mathbb{A}$	$\mathbb{B}$	$\mathbb{C}$	$\mathbb{D}$	$\mathbb{E}$	$\mathbb{F}$	$\mathbb{G}$	$\mathbb{H}$	$\mathbb{I}$	$\mathbb{J}$
is_AWS	0.028*** (0.004)	0.023*** (0.006)	-0.008 (0.007)	0.002 (0.007)	0.001 (0.009)	0.036*** (0.004)	0.031*** (0.005)	0.002 (0.007)	0.012 (0.007)	0.005 (0.009)
is_Baidu	0.069*** (0.004)	0.064*** (0.006)	0.033*** (0.007)	0.043*** (0.007)	0.042*** (0.009)	0.117*** (0.004)	0.112*** (0.005)	0.083*** (0.007)	0.092*** (0.007)	0.086*** (0.009)
is_Aliyun	0.082*** (0.004)	0.077*** (0.006)	0.047*** (0.007)	0.056*** (0.007)	0.055*** (0.009)	0.049*** (0.004)	0.045*** (0.005)	0.016** (0.007)	0.025*** (0.007)	0.019** (0.009)
is_pretrained		0.019*** (0.005)	0.019*** (0.004)	0.019*** (0.004)	0.020*** (0.006)		0.013*** (0.004)	0.013*** (0.004)	0.013*** (0.004)	0.016* (0.008)
is_adversarial		-0.021*** (0.006)	-0.021*** (0.004)	-0.021*** (0.004)	-0.033*** (0.006)		-0.016*** (0.005)	-0.016*** (0.005)	-0.016*** (0.005)	-0.019** (0.009)
is_augmented		0.008 (0.006)	0.008* (0.004)	0.008* (0.004)	0.014* (0.006)		0.010* (0.005)	0.010* (0.005)	0.010** (0.005)	0.011 (0.009)
is_PGD			0.056*** (0.008)	0.056*** (0.008)	0.056*** (0.007)			0.047*** (0.008)	0.047*** (0.008)	0.047*** (0.008)
is_FGSM			0.092*** (0.008)	0.092*** (0.008)	0.092*** (0.007)			0.070*** (0.008)	0.070*** (0.008)	0.070*** (0.008)
is_BLB			-0.001 (0.008)	-0.001 (0.008)	-0.001 (0.007)			0.015* (0.008)	0.015* (0.008)	0.015* (0.008)
is_CW2			0.001 (0.008)	0.001 (0.008)	0.001 (0.007)			0.014* (0.008)	0.014* (0.008)	0.014* (0.008)
is_DEEPFOOL			-0.013 (0.008)	-0.013 (0.008)	-0.013* (0.007)			0.008 (0.008)	0.008 (0.008)	0.008 (0.008)
is_STEP_LLC			0.070*** (0.008)	0.070*** (0.008)	0.070*** (0.007)			0.052*** (0.008)	0.052*** (0.008)	0.052*** (0.008)
is_RFGSM			0.039*** (0.008)	0.039*** (0.008)	0.039*** (0.007)			0.034*** (0.008)	0.034*** (0.008)	0.034*** (0.008)
is_LLC			0.030*** (0.008)	0.030*** (0.008)	0.030*** (0.007)			0.022*** (0.008)	0.022*** (0.008)	0.022*** (0.008)
is_depth_34				-0.009** (0.004)	0.000 (0.009)				-0.008* (0.005)	0.008 (0.009)
is_depth_50				-0.019*** (0.004)	-0.023*** (0.009)				-0.020*** (0.005)	-0.001 (0.009)
is_pre×adv					-0.002 (0.009)					0.005 (0.009)
is_pre×aug					0.010 (0.009)					0.027*** (0.009)
is_pre×depth34					-0.003 (0.009)					-0.014 (0.009)
is_pre×depth50					-0.009 (0.009)					-0.026*** (0.009)
is_adv×depth34					0.002 (0.010)					-0.004 (0.011)
is_adv×depth50					0.035*** (0.010)					0.006 (0.011)
is_aug×depth34					-0.026** (0.010)					-0.022** (0.011)
is_aug×depth50					-0.010 (0.010)					-0.022** (0.011)
R^2	0.150	0.218	0.558	0.575	0.600	0.352	0.395	0.531	0.550	0.575
Adjusted R^2	0.147	0.210	0.546	0.562	0.580	0.350	0.389	0.518	0.535	0.554

${}^{*}p < .1$, ${}^{**}p < .05$, ${}^{***}p < .01$. Number of observation is 486.

surrogate models. Regression $\mathbb{B}$ and Regression $\mathbb{G}$ decompose the effect of pretraining in Table III and Table IV, respectively. By looking at its coefficients, we get the following relations:

(i) In the object classification task, pretraining has a significantly negative impact on the misclassification rate but a significantly positive effect on the matching rate.

(ii) In the gender classification task, pretraining has a significantly positive effect on both the F2M rate and the M2F rate.

The observation in the object classification task is intuitively confusing. In the research common sense, pretrained surrogates should be more similar to the target model because they both have a low error rate. Therefore, better similarity should guarantee better transferability of the crafted AEs. The empirical result, on the contrary, tells us that pretrained surrogates are worse choices for untargeted transfer attack but better for targeted transfer attack. The result is unlikely caused by the potential joint effects because further decomposition shown in Table III supports it as well. In addition, we check the surrogate accuracy and confirm that pretrained surrogates have a roughly 5% improvement on accuracy for all model types.

This finding shows that the "surrogate similarity" concept is not suitable, at least not straightforward, to describe transfer attacks. On the one hand, suppose pretraining does improve the similarity between the surrogate and target, then both the misclassification rate and matching rate should be improved because better similarity ensures better transferability. On the other hand, suppose pretraining reduces the similarity, then both metrics should be lowered. However, neither can explain this result. Therefore, if we want to define "surrogate similarity", we should not define it via the accuracy gap.

In the gender classification task, however, pretraining improves both metrics. This suggests that a simpler task may benefit more from pretraining. Although we are not aware of any existing result that can explain this, we have an intuitive hypothesis about why pretraining works in simple tasks. Bahri *et al.* [10] mentioned a result that in the random Gaussian landscape, the number of local optima increases exponentially when the loss decreases. If we generalize this result and hypothesize that the number of local optima increases exponentially when the task complexity increases, then the empirical finding follows. This is because under this hypothesis, the ten-class object classification task is more complex than the binary gender classification task and thus much more local optima is present. Therefore, although pretraining improves the model's accuracy, it leads the surrogate into a local optima that is different to the local optima in which the target model locates, which makes the boundary of the surrogate and the target model less similar. The targeted transfer attack, however, relies less on the boundary. Instead, it relies more on the correct direction of the perturbation towards the target class, and a similar direction towards the target class is enforced by performing better on the task. This rationale may explain why the matching rate is improved while the misclassification rate is decreased by pretraining.

Regression $\mathbb{B}$ and Regression $\mathbb{G}$ decompose the effect of surrogate dataset factors as well. However, adversarial training and data augmentation do not show strong correlation to the transfer attack. Therefore, we leave the discussion about these factors to the later section which considers the joint effect of surrogate-level factors.

Observation 2. *Pretraining improves targeted transfer attacks but not untargeted transfer attacks. This implies an appropriate definition of model similarity is extremely difficult.*

3) Adversarial Algorithm Factors: Since the finding of the adversarial property of neural networks, adversarial algorithm is the main object being studied, making it relatively well-understood among all components of a transfer attack. However, their transferability in the real attack scenario is under-explored.

Regression $\mathbb{C}$ and Regression $\mathbb{H}$ decompose the effect of adversarial algorithms in Table III and Table IV, respectively. By comparing its coefficients, we get the following relations:

(i) In the object classification task, DeepFool $\approx$ BLB $\approx$ CW2 $<_{p<0.001}$ LLC $\approx$ RFGSM $\approx$ UAP $\approx$ PGD $<_{p<0.001}$ FGSM $\approx$ Step-LLC on the misclassification rate, and LLC $\approx$ CW2 $\approx$ BLB $<_{p=0.008}$ Step-LLC $<_{p=0.068}$ UAP $<_{p=0.008}$ DeepFool $\approx$ PGD $\approx$ FGSM $\approx$ RFGSM.

(ii) In the gender classification task, DeepFool $\approx$ BLB $\approx$ UAP $\approx$ CW2 $<_{p<0.001}$ LLC $\approx$ RFGSM $<_{p=0.031}$ PGD $<_{p=0.062}$ Step-LLC $<_{p=0.004}$ FGSM on the F2M rate. For the M2F rate, we have UAP $\approx$ DeepFool, CW2 $\approx$ BLB $\approx$ LLC and RFGSM $<_{p=0.092}$ PGD $\approx$ Step-LLC $<_{p=0.021}$ FGSM.

Two patterns are especially significant. First, the strong adversarial algorithms (BLB and CW2) transfer worse than most of the algorithms on all the metrics, which is unexpected. In particular, although they are targeted, their matching rates are lower than untargeted algorithms. In addition, FGSM performs well in all kind of transfer attacks, although it uses less information from the surrogate. Second, single-step algorithms have a higher misclassification rate than their iterative counterparts (Step-LLC > LLC and FGSM > PGD). Su *et al*. [41] confirms that FGSM transfers better than PGD as well. This result is surprising because iterative algorithms are more powerful and exploit more information from the surrogate model. Combining these two facts, we can see that attacks that use too much information of the surrogate are less likely to transfer, and the probably most transferable information is the gradient with regard to the seed image.

Observation 3. *In the real applications, strong adversarial algorithms,* e.g.*, CW2, might have weak transferability. In addition, single-step algorithms transfer better than iterative algorithms,* e.g.*, FGSM > PGD. This suggests the probably most transferable information is the gradient with regard to the seed image.*

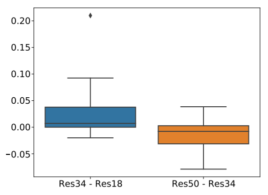


Figure 4: The box plot of the difference in misclassification rate against the local VGG target. The diamond represents an outlier.

4) Surrogate Depth Factors: Choosing the surrogate depth is important for a good surrogate model. Demontis *et al*. [15] pointed out that simple surrogates are better than complex surrogates. However, in practice, this brings up a question: a too simple surrogate cannot perform reasonably on the task. For example, the most complex surrogate that Demontis *et al*. applied was a two-layer neural network. However, this particular architecture is too simple to be the surrogate in our tasks, getting roughly 15% accuracy on the test data for the object classification task. Therefore, the impact of surrogate depth in the real scenario is still under-explored.

Regression $\mathbb{D}$ and $\mathbb{I}$ decompose the effect of surrogate depth in Table III and Table IV, respectively. By comparing the coefficients, we can see that for the gender classification, ResNet-18 indeed performs better than ResNet-34 and ResNet-50. However, for the object classification task, ResNet-18, ResNet-34 and ResNet-50 essentially have the same performance. This shows simpler surrogates do not necessarily have better transferability in the real transfer attack, especially for complex tasks. We further conduct two experiments for the object classification task. One of them aims to show that on local targets, an appropriate ResNet surrogate preserves better transferability than both simpler and deeper surrogates. The other aims to show that for VGG surrogates, the same phenomenon is observed as well.

(i) **Experiments on Local Targets**

To make our settings more similar to the previous study by Demontis *et al*. [15], we apply AEs crafted from ResNet surrogates with different depths against a local VGG-16 target model. We calculate the difference in misclassification rate among various depths for each threat setting which consists of different pretraining factors and attack algorithm factors. The surrogate dataset is fixed to raw, *i.e*., no data enrichment is used. Figure 4 shows the distribution of the difference between the ResNet-34 and the ResNet-18 surrogates and the difference between the ResNet-50 and the ResNet-34 surrogates.

It can be seen from Figure 4 that the ResNet-34 surrogates have higher transferability than ResNet-18 but the ResNet-50 surrogates have lower transferability than ResNet-34. By performing Wilcoxon test [48], we get that the first conclusion has p-value 0.028 and the second has p-value 0.035, which are

Table V: The OLS results on VGG surrogates.

	coef	std err	p-value
is_google	0.1711	0.018	0.000
is_aws	0.0101	0.018	0.570
is_baidu	0.1444	0.018	0.000
is_aliyun	0.0240	0.018	0.177
is_PGD	0.0482	0.019	0.014
is_FGSM	0.2016	0.019	0.000
is_BLB	-0.0006	0.019	0.977
is_CW2	-0.0157	0.019	0.421
is_DEEPFOOL	-0.0207	0.019	0.288
is_STEP_LLC	0.1575	0.019	0.000
is_RFGSM	0.0749	0.019	0.000
is_LLC	0.0244	0.019	0.210
is_13	**0.0006**	0.013	0.960
is_16	**0.0780**	0.013	0.000
is_19	**-0.0003**	0.013	0.983

statistically sufficient. This result agrees with our hypothesis that an appropriate surrogate complexity is better than lower and higher complexity.

(ii) **Experiments on VGG Surrogates**

To make sure that this phenomenon is not restricted to ResNet surrogates, we extend the result to VGG surrogates. We use VGG-11, VGG-13, VGG-16 and VGG-19 as surrogates and evaluate the transferability of the crafted AEs on the MLaaS systems. These surrogates are obtained from pretrained models and the raw dataset is used to fine-tune them, *i.e.*, *is_pretrained* is fixed to True and *is_augmented* and *is_adversarial* are fixed to False.

Table V shows the result of OLS regression. The emphasized numbers in the table show that using VGG-13, 16 and 19 has a 0.0006, 0.078 and -0.0003 improvement in the misclassification rate respectively when compared to VGG-11. By looking at the p-values, we observe that VGG-11 $\approx$ VGG-13 $\approx$ VGG-19 $<_{p<0.001}$ VGG-16, making VGG-16 a better surrogate than its simpler and deeper versions. This result shows that the phenomenon that a surrogate with appropriate depth is better than other surrogates is not restricted to ResNet surrogates but holds for VGG surrogates as well.

Observation 4. *Surrogate complexity, defined by the depth of the surrogate, has a non-monotonic effect on the transferability. A surrogate with appropriate depth is better than both simpler and deeper surrogates.*

5) Joint Effect of Surrogate-Level Factors: Various factors are included to train a surrogate model, *i.e.*, surrogate dataset, pretraining and the choice of surrogate depth. Therefore, their interactions are of particular interest, since these factors affect the attack via the surrogate as a whole.

Regression $\mathbb{E}$ and Regression $\mathbb{J}$ decompose the joint effect of surrogate-level factors in Table III and Table IV, respectively. Before we continue, some interpretation of regression $\mathbb{E}$ and $\mathbb{J}$ should be explained to avoid misunderstanding of the result. First, the indicator variable of the decomposed factor have a different statistical interpretation when compared to other regressions due to the joint terms included. For example, in regression $\mathbb{E}$ and $\mathbb{J}$, the coefficient of *is_pretrained* represents the influence of pretraining when ResNet-18 surrogate is used without data enrichment. This is because the joint effect is characterized by the joint terms while previously the coefficient is not conditioned on the choice of ResNet-18 and the absence of data enrichment. Similar rules apply to the coefficients of *is_adversarial*, *is_augmented* and surrogate depth factors. Second, a significantly positive coefficient of the joint term does not necessarily mean that using them jointly is good for transfer attack, vice versa. This is because the joint terms should be added with the individual terms to characterize the joint effect. For example, for the Regression $\mathbb{E}$ shown in Table III, the coefficient of *is_pre*$\times$*adv* is -0.003. To recover the effect of the joint usage of pretraining and adversarial training, we should add this term with *is_pretraining* and *is_adversarial*, which is (-0.003) + (-0.014) + (+0.005) = -0.012. Third, we cannot conclude that two factors are approximately independent, even though the joint term is not significant. Instead, this should be digested as that the joint effect is not significant enough to be observed.

We only discuss joint terms with a significantly non-zero coefficient. The first interesting phenomenon is that almost no joint term has consistent effects on different kinds of transfer attack, *i.e.*, if they show significant improvement on untargeted transfer attack, then they almost always do not show significant improvement on targeted transfer attack. This phenomenon further supports Observation 2 by showing that pretraining is not the only one that has this feature. Second, the interaction between the surrogate-level factors is extremely complex, as no common effect is observed in both the Table III and Table IV. For example, in Table III we observe significantly negative coefficient for *is_pre*$\times$*aug* on the misclassification rate. However, in Table IV this coefficient becomes insignificantly positive. Therefore, choosing a good surrogate model actually needs trial and error, and is highly task-specific.

Observation 5. *The interaction between surrogate-level factors is highly chaotic and task-specific. Training a good surrogate needs trial and error.*

6) Importance of Different Factors: We have discussed the effect of many factors, but we still want to know which factor contributes the most to the transferability of AEs. The contribution of factors can be measured by the changes of R^2 when the factors are included.[6]

By comparing the ΔR^2 of hierarchical regressions in Table III and Table IV, we can see the most important factors are the platform factors and the adversarial algorithm factors, with a ΔR^2 ranging from 0.150 to 0.645 and from 0.136 to 0.340, respectively. This means that attackers benefit the most from setting a good target platform and a good adversarial algorithm. While the target platform is predetermined, the best improvement for the attack is to choose an appropriate attack algorithm, e.g., FGSM.

Observation 6. *Among all the discussed factors, the easiest and most beneficial practice to improve a transfer attack is to apply an appropriate adversarial algorithm,* e.g.*, FGSM.*

[6] R^2 is a statistic that measures the ratio of model's explained variance to the total variance. When additional independent variables are included in a linear regression, ΔR^2 measures the contribution of additional variables.

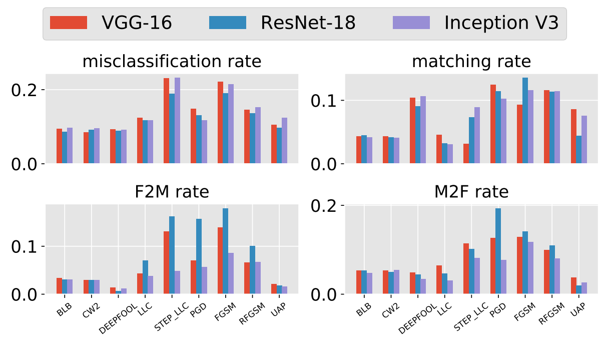
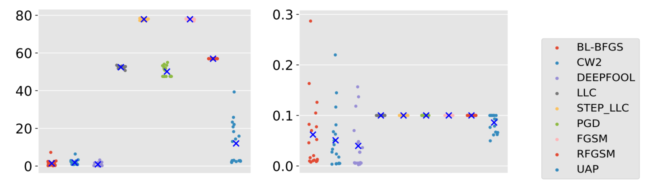


Figure 5: The transfer attack results grouped by surrogate architectures and attack algorithms. The result of each bar is averaged on the four platforms.



(a) L_2 norm (b) L_∞ norm

Figure 6: The L_2 and L_∞ norm of perturbation for AEs.

7) Surrogate Architecture Factors: We have completed the OLS analysis of transfer attack in the image classification and the gender classification, leaving an important factor behind: the impact of surrogate architecture. Figure 5 plots metrics against adversarial algorithms grouped by surrogate architectures. Unlike the conclusion from Su *et al.* [41] that AEs crafted from most surrogates only transfer in their own surrogate family except VGG, we can see from Figure 5 that no dominant architecture exists in the real scenario. The result suggests surrogate families other than VGG worth attention in the real transfer attack as well.

Observation 7. *No dominant architecture family is found in the real transfer attack.*

B. Sample Property Factors

In this section, we dive deeper into the correlations between transferability and sample-level properties. Specifically, we study the following three properties: the norm of adversarial perturbation, the adversarial confidence (*e.g.*, κ in CW attack [12]) and the intrinsic classification hardness of seed images.

1) Connection Between Transferability and the Norm of the Adversarial Perturbation: So far, relaxing the perturbation norm budget of adversarial attacks has been believed to be a general method to increase transferability. We find this conclusion still holds in the real setting, hence we aim to provide an answer for a deeper question: does the type of norm matter in deciding transferability?

Figure 6 plots the average norm of the adversarial perturbations generated by each attack algorithm in the image classification task. It is interesting to notice that the order of adversarial algorithms in terms of L_2 norm in Figure 6a is roughly aligned to the order in terms of misclassification

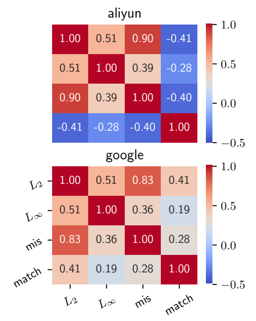


Figure 7: The correlation of norm and transferability.

rate shown in Section V-A3. Figure 7 further computes the correlation matrix between transferability metrics and norms, from which we observe the correlation between L_2 norm and misclassification rate is extremely high, exceeding 0.8. The correlation between L_∞ and misclassification rate is far weaker, up to 0.41. In fact, we prove in Appendix X-E that even this correlation is largely due to the dependence of L_∞ on L_2 norm. This finding motivates us to hypothesize that a perturbation with larger L_2 norm and fixed L_∞ norm improves transferability. As a validation of this hypothesis, we repeat to generate perturbations sampled from set $\{-0.1, 0.1\}^d$ randomly and stop until either an adversarial perturbation that deceives the surrogate is found or a maximum iteration, 1000 times, is reached. In this way, we get adversarial perturbation with the largest L_2 norm while keeping the L_∞ norm fixed at 0.1. These AEs, though unaware of the explicit gradient information, achieve an average misclassification rate of 39.1% on Google, 16.5% on AWS, 25.4% on Aliyun and 36.7% on Baidu, respectively, which is higher than many attack algorithms with the same L_∞ norm budget. On the contrary, AEs with large L_∞ norm of perturbations but small L_2 norm, such as those generated by BLB and CW, only have trivial transferability as shown in Section V-A3. Therefore, we conclude that transfer attacks are closely related to the L_2 norm of the adversarial perturbation but not the L_∞ norm, although human vision systems are believed to be insensitive to L_2 norms [51].

Observation 8. *The transferability of the AEs is very closely related to the L_2 norm of the perturbation but not the L_∞ norm. Increasing the L_2 norm while keeping L_∞ norm fixed can increase the transferability of the AEs.*

2) Connection between Transferability and the Adversarial Confidence: Some attacks are given the ability to control adversarial confidence defined in many ways, one of which is the κ value, the logit gap between the adversarial class and the second most likely class. Su *et al.* [41] empirically found that imposing a stricter κ constraint on CW attack slightly increases transferability on untargeted attacks. However, this measurement of adversarial confidence is scaling-sensitive because the κ value scales if all parameters of the ReLU network is scaled, which is undesired because ReLU network is scaling-invariant. A straightforward extension of the κ value is to compute the softmaxed logit gap between the adversarial class and the

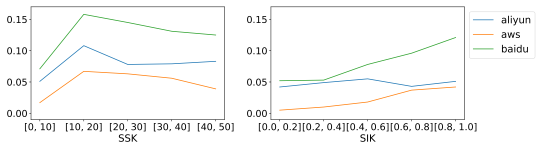


Figure 8: $P(\text{success} \mid \kappa)$ across platforms when κ is SSK and SIK, respectively. Both figures are scaled by a constant factor $P(\text{success})$. While all platforms show more vulnerability to a larger SIK, platforms except for Baidu do not show clear vulnerability to a large SSK.

Table VI: Linear regression results for SSK and SIK respectively. Adjusted R-squared are 0.653 for SSK regression and 0.821 for SIK regression. SSK and SIK are discretized as Figure 8 and revalued from one to five. For each entry, the upper is the result of SSK regression and the lower is of SIK regression.

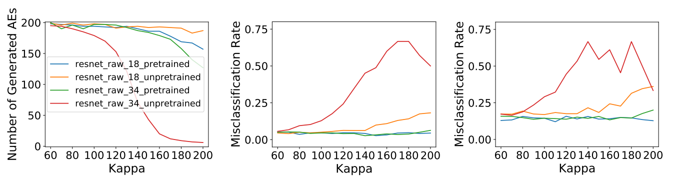
IV	coefficient	p-value	5% interval
is_Alibaba	0.065	0.003***	[0.026,0.103]
	0.019	0.050**	[0.000,0.039]
is_Baidu	0.112	0.000***	[0.073,0.150]
	0.049	0.000***	[0.030, 0.068]
is_AWS	0.030	0.117	[-0.009,0.068]
	-0.006	0.482	[-0.026,0.013]
SSK / SIK	**0.005**	**0.303**	[-0.005,0.015]
	0.010	**0.002*** **	[0.004,0.014]

$^{*}p < .1$, $^{**}p < .05$, $^{***}p < .01$.

second most likely class, called *Scaling-Insensitive* κ *(SIK)* in this paper. By definition, scaling the logit will cause a much smaller change on SIK than SSK. The original κ value is thus renamed *Scaling-Sensitive* κ *(SSK)*. For ease of understanding, SIK and SSK are mentioned collectively as κ . To make AEs crafted from different algorithms and surrogates comparable, we measure SSK and SIK by direct calculation for all AEs. Then we divide them into two groups based on whether the AE successfully fools the cloud. Invoking the Bayesian rule: $P(\text{success} \mid \kappa) = P(\text{success}) \times P(\kappa \mid \text{success})/P(\kappa) \propto P(\kappa \mid \text{success})/P(\kappa)$, we can see that the change in transfer success rate is proportional to the change in $P(\kappa \mid \text{success})/P(\kappa)$, a metric easier to compute.

Figure 8 plots $P(\kappa \mid \text{success})/P(\kappa)$ for SSK and SIK. It indicates that the conclusion from Su *et al.* [41] does not generalize as Aliyun and AWS do not show higher vulnerability to larger SSK. However, Figure 8 shows transferability of AEs is nearly linear to SIK. This suggests that the conclusion from Su *et al.* [41] is only a special case of a more general property. They might get their conclusion because increasing SSK probably leads to a larger SIK. We further run a linear regression to quantify the impact, shown in Table VI. The result shows $P(\text{SSK} \mid \text{success})/P(\text{SSK})$ is not strongly explained by SSK ($p = 0.303$), but $P(\text{SIK} \mid \text{success})/P(\text{SIK})$ strongly depends on SIK ($p = 0.002$). Therefore, we conclude that SIK is a good choice for measuring adversarial confidence in transfer attack while SSK is not representative enough.

In practice, we want to know whether increasing the SSK for the CW attack is sufficient to increase the transferability as well. To answer this question, we perform CW attack with different κ thresholds. The result is shown in Figure 9. We can see from Figure 9b and Figure 9c that a larger SSK does not always increase transferability. For example, for the blue line



(a) Number of AEs (b) CW on AWS (c) CW on Baidu

Figure 9: The misclassification rate for CW attack with different κ thresholds, measured on AWS and Baidu using four different surrogates. The number of AEs drops because we use a fixed iteration budget, but for large kappa this budget no longer guarantees an adversarial perturbation.

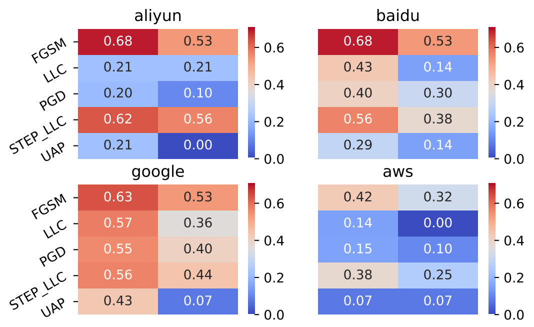


Figure 10: Comparison of the transferability between AEs generated from natural AEs and normal seed images. For each plot, the left column is the misclassification rate of AEs generated from natural AEs and the right column is the misclassification rate of AEs generated from other seed images. It shows that the left column is always greater or equal to the right column.

in Figure 9, while the number of AEs drops at roughly 25% when we increase SSK from 60 to 200, the transfer rate against the AWS platform does not improve. Therefore, increasing SSK is not useful enough for a real transfer attack.

Observation 9. *The logit difference between the adversarial class and the second most likely class, called* κ *value, is neither a good measurement for the transferability nor a good tool to increase the transferability.*

3) Connection Between Transferability and Intrinsic Classification Hardness: An image is defined to be a *natural* AE w.r.t. a surrogate model if it can fool the surrogate without any disguise. When conducting transfer attack in the real world, natural AEs are unavoidable since the surrogate cannot achieve 100% accuracy. We are thus motivated to find out if perturbing natural AEs can provide better transferability.

We compare the transferability of AEs generated from natural AEs with AEs generated from other seed images in Figure 10. The natural AEs disguised by adversarial perturbation achieve higher success rate than AEs generated from other seed images consistently. It suggests that the intrinsic classification hardness is important in deciding transferability and real world attackers should prefer natural AEs in transfer attacks. The result is intuitive because natural-AE seeds are believed to be closer to the theoretical classification border and an adversarial perturbation is more likely to transfer on these images.

Observation 10. *AEs generated from seed images that are hard to classify on surrogates have better transferability.*

VI. Discussion

A. Ethic

When evaluating attacks on the platforms, we conduct normal queries with the official API and pay the usage fee as required. Therefore, our evaluations are legitimate and do no harm to the operations of the cloud services. The related companies are informed about our experiment to exclude any potential harm, *e.g.*, do not use the uploaded images in any form. We have received confirmation from AWS and Baidu.

B. Limitation and Future Work

Our goal is to provide a systematic view of factors that affect transferability. Therefore, we leave some conclusions not fully understood. For example, we do not validate the hypothesis that the number of the local optimal increases exponentially when the task complexity increases. These arguments are beyond the scope of this paper and require further study in the future. In addition, we do not find the best hyperparameter setting in the real settings. Although these answers are useful, it requires massive resources which we cannot afford, thus left for future work. In particular, our conclusions about the attack algorithms are specific to the chosen hyperparameters, not representing their ability under different hyperparameters. For example, we set $\kappa = 0$ for CW attack, but choosing a large κ may be preferable for attackers. In our experiment, however, we set $\kappa = 0$ for a fair comparison of CW with other attacks which cannot manipulate kappa.

VII. Related Works

Early works on adversarial attacks [12], [19], [30], [32], [44], [46] mainly focus on white-box attacks, where the adversary has full access to the target model. Considering the lack of information in black-box attacks, efforts have been made in two directions: 1) *query-based attacks* that estimate the gradient information used to facilitate the generation of AEs [13], [21] and 2) *transfer attacks* that generate AEs using a surrogate model and expect them to transfer to the unknown target model [36], [37]. Our paper focuses on transfer attacks.

The transferability of AE in deep learning models was first discovered by Szegedy *et al.* [44]. Papernot *et al.* [36], [37] utilized this property to perform black-box attack to real MLaaS systems, which proves its possibility for real attacks. Carlini *et al.* [12] proposed that high-confidence adversarial examples increase the transferability.

Efforts have been made to understand transferability of AEs. Liu *et al.* [29] found that unlike untargeted AEs, targeted AEs almost never transfer and proposed to use an ensemble of surrogates to increase transferability of targeted attacks. Wu *et al.* [49] studied influencing factors of transferability and point out that reduced variance of gradients results in better transferability. Su *et al.* [41] conducted a comprehensive analysis of transfer attack using 18 different surrogates. They found that relaxing norm constraint of adversarial perturbation generates better transferability. They concluded that FGSM $>$ PGD $>$ CW in the sense of transferability as well. Demontis *et al.* [15] further highlighted that simpler surrogates and better gradient alignment provide better transferability. This conclusion was based on adding different levels of regularization and found stronger regulation provides better transferability. In addition, they concluded that targets which have large gradients for the input are more vulnerable to transfer attack as well. Our work aims at examining whether their conclusions generalize to transfer attack in the real applications and extending their results to get more insights. In particular, when evaluating the relation between surrogate complexity and transferability, we believe that a better way to examine the impact of complexity in the real settings is to directly use surrogates with different depths but the same architecture family, rather than changing regularization. Our conclusion about the non-monotonicity of the effect of surrogate depth on transferability is a complement to Demontis *et al.*

VIII. Conclusion

In this paper, we identify the difficulties of evaluating transferability of adversarial example (AE) in the real world and propose customized metrics to address these difficulties. Based on the proposed metrics, we conduct a systematic evaluation of real-world transfer attacks on four popular MLaaS systems, Aliyun, Baidu Cloud, Google Cloud Vision and AWS Rekognition. The evaluation leads to the following new conclusions on top of the existing ones made in lab settings: **(1)** model similarity concept is ill-suited for transfer attack; **(2)** surrogates of suitable complexity can exceed simpler and deeper counterparts; **(3)** MLaaS systems have different level of robustness to transfer attack and could use more effort to improve; **(4)** Strong adversarial algorithms do not necessarily transfer better, and single-step algorithms transfer better than iterative algorithms; **(5)** adversarial algorithm and the target platform are the most important factors for transfer attacks, and the most beneficial practice is to choose an appropriate adversarial algorithm; **(6)** no dominant surrogate architecture exists in the real transfer attack. **(7)** large L_2 norm of adversarial perturbation can be a more direct source of transferability than L_∞ norm. **(8)** larger gap between the posterior of logits makes better transferability. **(9)** classification hardness is preferred when choosing seed images of transfer attack.

IX. Acknowledgement

We would like to thank our shepherd and the anonymous reviewers for their valuable suggestions. This work is partly supported by the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, NSFC under No. 62102360, and U1836202, and the Fundamental Research Funds for the Central Universities (Zhejiang University NGICS Platform). Ting Wang is partially supported by the National Science Foundation under Grant No. 1951729, 1953813, and 1953893.

References

[1] Alibaba cloud. https://aliyun.com.

[2] AWS rekognition. https://aws.amazon.com/rekognition.

[3] AWS Rekognition documentation. https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html.

[4] Baidu cloud. https://bce.baidu.com.

[5] Definition of additively separable functions. https://calculus.subwiki.org/wiki/Additively_separable_function.

[6] Google cloud vision. https://cloud.google.com/vision.

[7] Google Open Images V6+. https://storage.googleapis.com/openimages/web/index.html.

[8] PyTorch. https://pytorch.org/.

[9] pytorch/vision. https://github.com/pytorch/vision, June 2020. original-date: 2016-11-09T23:11:43Z.

[10] Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics 11*, 1 (2020), 501–528.

[11] Box, G. E. P., and Cox, D. R. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological) 26*, 2 (1964), 211–252.

[12] Carlini, N., and Wagner, D. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP)* (May 2017), pp. 39–57. ISSN: 2375-1207.

[13] Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J. ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security* (2017), AISec '17, Association for Computing Machinery, pp. 15–26.

[14] Chen, S., He, Z., Sun, C., and Huang, X. Universal adversarial attack on attention and the resulting dataset damagenet, 2020.

[15] Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., Nita-Rotaru, C., and Roli, F. Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks. In *28th USENIX Security Symposium (USENIX Security 19)* (2019), pp. 321–338.

[16] Deng, J., Dong, W., Socher, R., Li, L., Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 248–255.

[17] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (June 2019), Association for Computational Linguistics, pp. 4171–4186.

[18] Eidinger, E., Enbar, R., and Hassner, T. Age and gender estimation of unfiltered faces. *IEEE Transactions on Information Forensics and Security 9*, 12 (2014), 2170–2179.

[19] Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations* (2015).

[20] He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), IEEE, pp. 770–778.

[21] Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box Adversarial Attacks with Limited Queries and Information. In *ICML 2018* (July 2018). arXiv: 1804.08598.

[22] Krizhevsky, A. Learning multiple layers of features from tiny images.

[23] Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial machine learning at scale, 2016.

[24] Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings* (2017), OpenReview.net.

[25] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (1998), 2278–2324.

[26] Li, C., Wang, L., Ji, S., Zhang, X., Xi, Z., Guo, S., and Wang, T. Seeing is living? rethinking the security of facial liveness verification in the deepfake era. *CoRR abs/2202.10673* (2022).

[27] Li, X., Ji, S., Han, M., Ji, J., Ren, Z., Liu, Y., and Wu, C. Adversarial examples versus cloud-based detectors: A black-box empirical study. *IEEE Trans. Dependable Secur. Comput. 18*, 4 (2021), 1933–1949.

[28] Ling, X., Ji, S., Zou, J., Wang, J., Wu, C., Li, B., and Wang, T. DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model. In *2019 IEEE Symposium on Security and Privacy (SP)* (2019), pp. 673–690.

[29] Liu, Y., Chen, X., Liu, C., and Song, D. Delving into Transferable Adversarial Examples and Black-box Attacks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings* (2017), OpenReview.net.

[30] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings* (2018), OpenReview.net.

[31] Moosavi-Dezfooli, S., Fawzi, A., Fawzi, O., and Frossard, P. Universal Adversarial Perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 86–94.

[32] Moosavi-Dezfooli, S., Fawzi, A., and Frossard, P. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 2574–2582.

[33] Pang, R., Xi, Z., Ji, S., Luo, X., and Wang, T. On the security risks of automl. *CoRR abs/2110.06018* (2021).

[34] Pang, R., Zhang, X., Ji, S., Luo, X., and Wang, T. Advmind: Inferring adversary intent of black-box attacks. In *KDD '20* (2020), pp. 1899–1907.

[35] Pang, R., Zhang, Z., Gao, X., Xi, Z., Ji, S., Cheng, P., and Wang, T. TROJANZOO: everything you ever wanted to know about neural backdoors (but were afraid to ask). *CoRR abs/2012.09302* (2020).

[36] Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* (2016).

[37] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical Black-Box Attacks against Machine Learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (2017), ASIA CCS '17, Association for Computing Machinery, pp. 506–519. event-place: Abu Dhabi, United Arab Emirates.

[38] Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)* (2016), IEEE, pp. 372–387.

[39] Shen, L., Ji, S., Zhang, X., Li, J., Chen, J., Shi, J., Fang, C., Yin, J., and Wang, T. Backdoor pre-trained models can transfer to all. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, November 15 - 19, 2021* (2021), Y. Kim, J. Kim, G. Vigna, and E. Shi, Eds., ACM, pp. 3141–3158.

[40] Simonyan, K., and Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv:1409.1556 [cs]* (Apr. 2015). arXiv: 1409.1556.

[41] Su, D., Zhang, H., Chen, H., Yi, J., Chen, P., and Gao, Y. Is robustness the cost of accuracy? - A comprehensive study on the robustness of 18 deep image classification models. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII* (2018), V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., vol. 11216 of *Lecture Notes in Computer Science*, Springer, pp. 644–661.

[42] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), IEEE, pp. 2818–2826.

[43] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 1–9.

[44] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations* (2014).

[45] Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. 6th International Conference on Learning Representations, ICLR 2018 ; Conference date: 30-04-2018 Through 03-05-2018.

[46] Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I. J., Boneh, D., and McDaniel, P. D. Ensemble Adversarial Training:

Attacks and Defenses. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings* (2018), OpenReview.net.

[47] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, L., AND POLOSUKHIN, I. Attention is all you need, 2017.

[48] WILCOXON, F. Individual comparisons by ranking methods. *Biometrics Bulletin 1*, 6 (1945), 80–83.

[49] WU, L., ZHU, Z., TAI, C., AND E, W. Understanding and Enhancing the Transferability of Adversarial Examples. *arXiv:1802.09707 [cs, stat]* (Feb. 2018). arXiv: 1802.09707.

[50] YANG, Z., DAI, Z., YANG, Y., CARBONELL, J., SALAKHUTDINOV, R. R., AND LE, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 5753–5763.

[51] ZHAO, H., GALLO, O., FROSIO, I., AND KAUTZ, J. Loss functions for image restoration with neural networks. *IEEE Trans. Computational Imaging 3*, 1 (2017), 47–57.

X. APPENDIX

A. Validity and Bias of Equivalence Dictionaries

An equivalence dictionary consists of different label mappings. Each label mapping can be viewed as a new class for a MLaaS model without making critical changes.

However, building a good equivalence dictionary is important for getting the real performance. Although we use human knowledge to create it, bias can be introduced. We formally discuss the level of bias introduced using the following proposition.

Proposition 1. *Suppose $\mathcal{D}$ contains all acceptable MLaaS classes for a local class c. M is the class mapping we construct. If M satisfies $M \subset \mathcal{D}$, then the measured matching rate is lower and the measured misclassification rate is higher than their true value. If $\mathcal{D} \subset M$, then the measured matching rate is higher and the measured misclassification rate is lower than their true value. When $\|\mathcal{D} - M\|$ shrinks to empty set, the bias decreases to zero as well.*

The proof of Proposition 1 is straightforward based on its definition. To make $\|\mathcal{D}-M\|$ as small as possible, we construct the equivalence dictionaries as described in Section IV-B. This method is good at capturing acceptable predictions in practice, reducing the number of missed acceptable predictions to a very low level, *i.e.*, $\|\mathcal{D} - M\|$ is nearly empty. In our experiment, we find some class mappings include super-classes, such as including *sports* in the class mapping of *baseball*, which makes $\mathcal{D} \subset M$. As shown in Proposition 1, this leads to a larger matching rate and smaller misclassification rate, which is potentially destructive to our evaluations. However, as we see from the results, all the matching rates are small and most of the misclassification rates are significant, indicating that the bias should be of little harm to the conclusions we make.

B. Impact of Irrelevant Factors on the Conclusions

Traversing all the possible factor settings in transfer attacks is expensive. In fact, to ensure balanced observations, the experiment settings are $\Omega = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_n$, where Ω_i is the setting space of one factor, *e.g.*, pretraining, surrogate dataset, adversarial algorithm, *etc.*, and n is the total number of factors. This means the total setting space is exponential. The large setting space makes it extremely hard to derive conclusions that are generalizable in a reasonable sense.

We address this problem by taking empirical expectation over the irrelevant setting space, thus minimizing the influence of the specific settings of irrelevant factors. To be exact, we conclude the influence of one specific factor by averaging over settings with this factor fixed and other factors varied in a balanced way. This is realized by running regression on balanced data which can be viewed as grid samples from Ω. For example, when concluding the effect of a target platform, the setting space of the target platform is the considered factor space (denoted as Ω_{i^*}), and the setting space of pretraining, dataset and surrogate model, *etc.*, are the irrelevant space $\prod_{i \neq i^*} \Omega_i$. The regression methodology provides automatic empirical expectation over the irrelevant settings based on the provided data, and we use balanced data in regression to further make the expectation taken on $\prod_{i \neq i^*} \Omega_i$ to be unbiased. Therefore, the effect of a specific factor is in fact eliminated by taking expectation. In other words, the conclusions made on one factor, *e.g.*, pretraining, are not affected by the specific settings of other factors, *e.g.*, adversarial algorithm. Since what we actually do is an empirical expectation, the effect of factors might not be totally eliminated, but still minimized.

C. Correctness and Generalization of Applying OLS Analysis

In multi-factor analysis, OLS analysis is commonly adopted to decompose the influence of each factor. However, due to its simplicity, it sometimes only reflects some aspects of the underlying influence function.

First, since the underlying influence of factors might be non-linear, the result of OLS which is linear is not suitable to be taken as the ground truth of the underlying influence function. The correct application is to use the "direction" and "size" of impact, *i.e.*, whether the impact is positive and how large the impact is, which is the central point of our observations and analyses. However, the reader should note that these directions are local and are technically restricted to the applied linear model.

Second, the OLS analysis finds "relationship" from the data and is a correlation analysis naturally. This means that all results are "correlations" but not "causation" unless there are additional experiments targeting the causation. In our paper, the only causation analysis is in Section V-B1 where we generate random AEs with largest L_2 and fixed L_∞ to test the influence of L_2 norm.

Third, the p-values are computed based on the assumption that the residual of linear model is Gaussian-distributed. In practice, this assumption may not hold, leading to biased p-values. Specifically, in our experiment, we find that the residual is roughly t-distributed with a degree of freedom 16 which is non-Gaussian. However, fixing this issue using techniques such as Cox-Box transformation [11] is ill-suited for binary variables and makes the result difficult to follow. In addition, for large degrees of freedom, t-distribution converges to Gaussian distribution. Therefore, since it has little bias on the result, we use the original OLS for clarity.

D. Detailed Explanations for OLS Tables

Due to the page limit of the main text, we do not include detailed explanations on how to read the formatted OLS tables in Section V-A. This section provides these materials for readers unfamiliar with OLS analysis.

To conclude a factor's impact on the dependent variable (called response variable as well), one need to look at all regressions on the dependent variable that include this factor. If the coefficient changes dramatically when an extra factor is included, it usually suggests that strong interaction exists between this factor and the extra factor. If the coefficient is not changed when other factors are included, it means that the interaction is weak, if exists. When strong interaction is detected, it is preferable to perform another regression and include the interaction term, *e.g.*, *is_pre*×*is_adv*, simplified as *is_pre*×*adv* in this paper. This kind of technique, known as hierarchical regression, provides more information about the impact of factors.

The tables are formatted to include and directly compare results of multiple OLS regressions. Since Table III and Table IV are formatted in the same way, we use Table III as an example. In Table III, regression $\mathbb{A}$ to $\mathbb{E}$ regress misclassification rate and regression $\mathbb{F}$ to $\mathbb{J}$ regress matching rate on the corresponding variables. Each regression from $\mathbb{A}$ to $\mathbb{E}$ includes different regressors. For one regression, if coefficients and standard deviations are provided in the table for a regressor, then it means that this regressor is included in this regression and the corresponding result is the provided value. Otherwise, if the space is left blank, it means that this regressor is not included in the regression. For example, in regression $\mathbb{A}$, the regressors are platform factors (from *is_Google* to *is_Aliyun*). In regression $\mathbb{B}$, the regressors are platform factors, pretraining factors (*is_pretrained*) and dataset factors (*is_adversarial* and *is_augmented*).

To avoid randomness introduced by the data on the result, p-values are provided. If a coefficient has a small p-value, highlighted by asterisks, it means that this coefficient is significantly different from zero, *i.e.*, the impact is "real". In practice, only values highlighted with asterisks are considered as important and worth attention. Apart from p-values, coefficients of the OLS regression and their standard deviations are provided in the table. A positive coefficient indicates that this factor has a positive impact on the dependent variable and a negative coefficient indicates the opposite. In addition, a larger absolute value of the coefficient indicates a larger effect. The standard deviation represents how much uncertainty is involved in computing the coefficient, and is mainly used as a supplement to p-values. Details about the OLS regressions can be found in the released code repository.

E. Further Discussion about L_∞ and Misclassification Rate

From Figure 7 in Section V-B1, we have shown that L_2 has large correlation with misclassification rate while the correlation between L_∞ and misclassification rate is relatively small. Since L_2 and L_∞ has a correlation 0.51, it is possible that even this roughly 0.4 correlation between L_∞ and misclassification rate is largely due to the dependence between L_2 and L_∞. We prove this claim in this discussion.

For notational reasons, we define X_1 to be L_2 norm and X_2 to be L_∞ norm, divided by their standard deviation, respectively. We further define Y to be the misclassification rate divided by its standard deviation. Thus, $\operatorname{Var} X_1 = \operatorname{Var} X_2 = \operatorname{Var} Y = 1$ and the correlation between them are equivalent to the covariance. Let $\epsilon = X_2 - X_1 \times \operatorname{Cov}(X_1, X_2)$. Since $\operatorname{Cov}(\epsilon, X_1) = \operatorname{Cov}(X_1, X_2) - \operatorname{Cov}(X_1 \times \operatorname{Cov}(X_1, X_2), X_1) = 0$, we know ϵ is actually uncorrelated with X_1. Therefore, $X_2 = \epsilon + X_1 \times \operatorname{Cov}(X_1, X_2)$ decomposes the effect of X_1 and ϵ is the "L_2-free" term of L_∞. Now we claim that this L_2-free term has a very small correlation to misclassification rate. Indeed, $\operatorname{Cov}(\epsilon, Y) = \operatorname{Cov}(X_2, Y) - \operatorname{Cov}(X_1 \times \operatorname{Cov}(X_1, X_2), Y) = \operatorname{Cov}(X_2, Y) - \operatorname{Cov}(X_1, X_2) \times \operatorname{Cov}(X_1, Y)$. Plug in all the correlation values in Figure 7, we get the correlation between ϵ and Y is -0.069 for Aliyun, -0.0439 for Baidu, -0.0633 for Google and -0.0682 for AWS. Therefore, the L_2-free contribution of L_∞ to the misclassification rate is slightly negative, almost negligible.

F. Guidelines for Defending Transfer Attacks

• (Observation 3) Use FGSM first when evaluating the robustness to transfer attacks.

• (Section V-B1 and V-B2) As long as the attack algorithm is able to control adversarial confidence (SIK) and perturbation norm of AEs, use a large SIK value and a large L_2 perturbation norm for better transferability.

• (Section V-B3) Maintain a pool of images that are intrinsically hard to classify. Use these images as seeds to test the robustness of the target model.

G. Parameter Settings for White-box Attacks and Augmentation

The detailed settings of white-box adversarial attacks on the surrogate models are shown in Table VII. We set kappa to zero for CW2 to make it comparable to other attacks, and it achieves a 100% success rate on surrogates for all cases. The iteration times of iterative attacks, though might improve the attack when increased further, are sufficient in our experiments because all of them reach the ϵ bound, as shown in Figure 6b. The choice of specific ϵ bound is based on the principle that an attacker would like to set ϵ as large as possible while maintaining visual quality, *i.e.*, being unnoticed by human, because it would lead to better transferability. Since visual quality is hard to quantify, we try different values for ϵ, and decide that 0.1 is a good choice for ImageNet under the condition that human eyes should not perceive the difference. Actually, with $\epsilon = 0.1$, there are already some mild noise patterns that are noticeable to human eyes. Therefore, our experiments can be viewed as a stress test of MlaaS systems.

The detailed setting of data augmentation is described below. Color jitter has brightness=0.1, contrast=0.1 and saturation=0.1, which means the augmented image will be in the range of these deviations. Random affine has maximum degree 30 and a maximum translate (0.1, 0.1). Random rotation has a

Table VII: Hyperparameter setting of adversarial algorithms. ϵ is all set to be 0.1 for ImageNet and Adience.

Attacks	Configuration
BL-BFGS	initial const=0.01
	search steps=10
	max iterations=40
CW2	κ=0
	initial const=0.001
	learning rate=0.02
	max iterations=100
DeepFool	max iterations=100
	overshoot=0.02
PGD	number of steps=10
RFGSM	α=0.5

maximum degree 30. Random horizontal flip, random vertical flip and random perspective have no parameter. They are applied with a probability of 0.5 independently. After finishing an epoch of training, the augmentation is redone, which means the dataset is enlarged by the number of epoch times. In our case, it is 50.

H. Implementation Changes on DEEPSEC

The original DEEPSEC implementation presented by Ling *et al.* [28] has issues about numerical stability which have been fixed by the original authors. We apply this new version of DEEPSEC in our experiments. Furthermore, we removed the random initialization in the PGD code, which is proposed in RFGSM, to make it directly comparable to FGSM. This action is not a fix to DEEPSEC but specific to our usage, and users are still encouraged to use DEEPSEC but not our changed version. Apart from some auxiliary code, what we include in our repo is mostly the test code which sends the data to MLaaS and analyzes the feedback.

I. Discussion about the Cutting Threshold

As discussed in Section IV-B, we address the multiple predictions problem by applying a cutting threshold estimated on the returns of original images. In fact, it is worthwhile to consider whether these estimated thresholds are applicable to other datasets and whether real users will apply the threshold cutting mechanism. We discuss these two issues below, respectively.

First, the estimation is conditioned on the chosen classes and no refinement can be made because no knowledge about the "real" threshold is known. However, we estimate the thresholds on the used dataset and apply it on this dataset. Therefore, the threshold does not need to "transfer". In addition, as stated in Section IV-B, the choice of thresholds only affects the comparison of MLaaS systems which is limited to the discussed tasks but leaves the discussion of other factors unharmed. Therefore, although the estimation might be biased, it does not affect conclusions about other factors but only platform factors. For attacks with seed images from undiscussed classes, they should do the threshold searching again and not directly apply our thresholds.

Second, the threshold cutting mechanism is a reasonable way to be adopted by users of MLaaS systems since the number of highly confident labels may vary. Furthermore, if the users apply threshold cutting, then the chances are that the thresholds derived from our methods are good estimations for the applied thresholds. This is because these users would like to focus on a few predictions without significant accuracy drop, which is aligned to our principles. Therefore, our conclusions provide an important insight of transfer attack for these users.

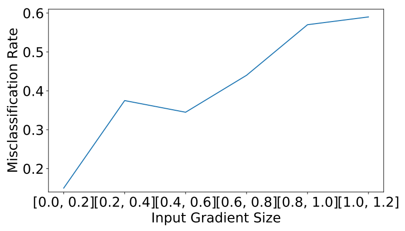


Figure 11: The relation between transferability and the input gradient size on the target model.

J. Discussion about Input Gradient Size

We would like to bring back that the input gradient size on our target models, MLaaS platform models, is not available. Nevertheless, in order to explore the relation between transferability of AEs and their input gradient size on the target model, we build up some local models to simulate the process of transfer attacks, i.e., use one local model as the target model and the others as the surrogate models. The experiments are conducted on the Adience dataset, and we choose the classic pretrained VGG-16 model as the target model. Then we use AEs generated on other local models to attack the target model and measure their input gradient size on the target model. Note that these AEs are generated with multiple model architectures, multiple surrogate dataset settings and multiple adversarial attack methods, following the evaluation settings in Section V-B2. Specifically, the surrogate datasets include raw and augmented datasets; the surrogate models may be trained with or without the adversarial training; the surrogate models' architectures include ResNet, VGG-16 and Inception V3; depths of the ResNet include 18, 34 and 50; the adversarial attack methods include BL-BFGS, CW2, DeepFool, FGSM, LLC, PGD, RFGSM, STEP-LLC and UAP.

The experiment results are shown in Figure 11. It is evident that AEs with larger input gradient size on the target model tend to have better transferability, regardless of how the AE is generated, which is consistent with the conclusion of Demontis *et al.* [15].