Sequential Architecture-Agnostic Black-Box Attack Design and Analysis

Furkan Mumcu[a], Yasin Yilmaz[a,*]

[a]*University of South Florida, Department of Electrical Engineering, 33620, Tampa, Florida, USA*

Abstract

Although adversarial machine learning attacks on image recognition models have been heavily investigated, the rising popularity of vision transformers revitalized the research on this topic. Due to the fundamental architectural differences between CNNs, which still dominate the image recognition applications, and transformers, the state-of-the-art attacks designed for CNNs are not effective against transformers, and vice versa. Such lack of transferability in attacks and the growing architectural heterogeneity in practice make the black-box attack design increasingly challenging. However, skillful attackers can handle the increasing uncertainty in target model architecture following two main approaches: designing transferable attacks that are robust to the architectural uncertainty in target model, and identifying the target architecture for attack selection. In this work, following the latter approach we propose a novel architecture-agnostic black-box attack design and analyze its performance. Experiments show that the proposed method, with a reasonable query overhead, outperforms the recent robust attack designs

*Corresponding author

Email addresses: furkan@usf.edu (Furkan Mumcu), yasiny@usf.edu (Yasin Yilmaz)

that follow the former approach. Different from the existing methods, the proposed method optimizes a trade-off between prior information about the target model and number of queries.
Keywords: adversarial machine learning, black-box attacks, transferability of attacks, vision transformers, sequential hypothesis testing

1. Introduction

Vulnerability of image classification models against adversarial machine learning attacks has been a very popular research topic since the introduction of fast gradient sign method (FGSM) (1). The majority of studies in this field are mainly focused on Convolutional Neural Networks (CNNs) since they constitute the dominant image classifier architecture for many years. But after the recent introduction of vision transformer (ViT) (2), there is an ongoing increase in the usage of transformer based architectures (3; 4; 5; 6) as they are shown to outperform popular CNNs and provide the state-of-the-art performance. While CNNs still dominate the image classification practice, the rapidly rising popularity of transformers cause a sheer architectural heterogeneity for target image classification models and in turn architectural uncertainty for black-box attacks.

Because of the rising popularity of transformers, their vulnerabilities against adversarial machine learning attacks are also being investigated. Latest works show that, due to the architectural differences between CNNs and transformers, existing adversarial attacks targeting CNNs are not effective on transformers since they have been developed with CNNs in mind (7; 8). Therefore several adversarial attacks which were specifically designed for transformers were recently introduced (7; 8; 9). While these attacks are effective on transformers, they are not

as effective against CNNs. This transferability issue, which is explained in Tables 2, 3, and 5, motivates new research on adversarial attacks that are effective on a wide range of CNN and transformer architectures. In this work, we show that a carefully designed *Sequential Attack Strategy Selection (SASS)* approach can enable adversaries to perform effective architecture-agnostic attacks on image recognition systems, as illustrated in Figure 1.

There are two main types of adversarial machine learning attacks, namely white-box attacks and black-box attacks. In white-box attacks, the attacker has full access to the target model including its architecture and parameters. On the other hand, in black-box attacks, the attacker has only access to the output probabilities and/or labels. There are two main black-box attack approaches: (i) transfer learning approach using the prior knowledge on target network and transferability of adversarial images, and (ii) fully data-driven approach using queries to estimate the gradient of the loss function with respect to the input image. While transfer learning-based black-box attacks suffer from low-transferability under architectural heterogeneity, query-based black-box attacks suffer from prohibitive number of queries (on the order of tens of thousands) required for gradient estimation. Minimizing the number of queries is important for an attacker due to several reasons, e.g., there might be stringent limits on the number of queries; queries might be costly; the risk of attack getting detected by the target increases with the number of queries.

In the transfer learning approach, the domain knowledge is used to select a prior model to further train (fine-tune) using queries. For instance, in image classification, CNNs have been typically used as prior model, also called substitute network. While adversarial images learned on one type of CNN (e.g., Inception)

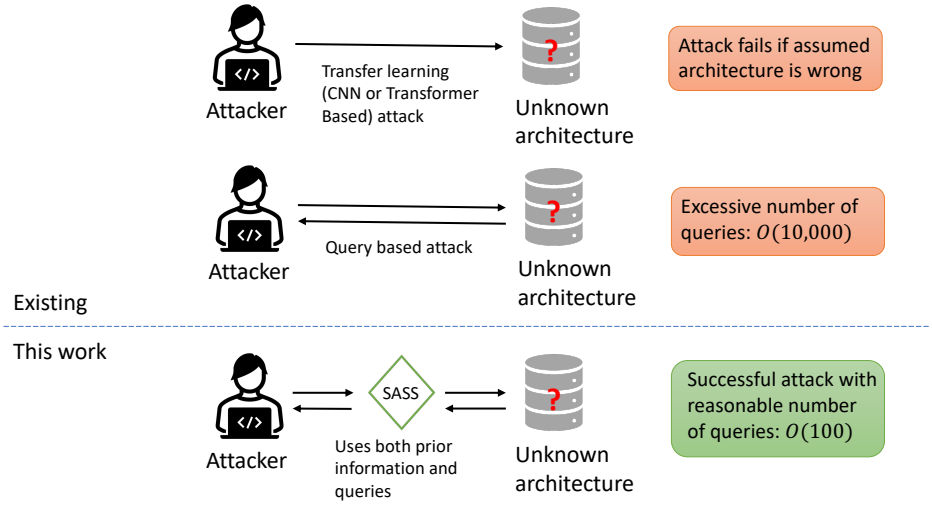

Figure 1: Current state-of-the art attacks are either typically designed for a specific architecture and not effective on others, or require excessive number of queries to design data-driven attacks. In this work, we show that attackers can overcome architecture uncertainty with a reasonable number of queries by following a Sequential Attack Strategy Selection (SASS) approach.

transfers well to another type of CNN (e.g., Resnet-50), the transferability is usually low when there is an architectural domain shift (e.g., Inception as substitute network and ViT as target network), as shown in Tables 2, 3, and 5 in Section 4. With the possibility of such significant domain shifts, it is not clear how to select the substitute network in the existing works which follow the transfer learning approach (10).

In the fully data-driven query-based approach, the black-box attack design methods do not use any prior model, and as a result utilize a large number (tens of thousands) of queries to learn loss gradients of the target network (11; 12). Since a good initialization significantly facilitates solving an optimization problem, we postulate that starting with an appropriate prior can greatly decrease the number of queries to the target network while designing an effective attack. Although the task of selecting an effective prior model became harder with the increasing

popularity of vision transformers, and in turn increasing heterogeneity in target network architectures, attackers still have a good domain knowledge (i.e., popular CNN and transformer networks) which they can use to select an effective prior model. In this work, we present a suitable problem formulation and an effective solution to minimize the average number of queries while maximizing the attack success rate. Our contributions can be summarized as follows:

- We present a novel approach *driven by both prior information and data*, sequential attack strategy selection (SASS), which can effectively deal with the architectural uncertainty of target model (i.e., CNN-based or transformer-based) using a practical number of queries (tens or hundreds of queries). SASS strikes a practical balance between the transfer learning and query-based approaches.

- We analyze the performance of the proposed method and provide guidance on how its parameters can be set.

- We benchmark our method with extensive experiments and analyze its effectiveness compared to the current state-of-the-art attacks. Experimental results show that SASS achieves consistently high attack success rates against both CNN-based and transformer-based target models by sending a feasible number of queries.

The rest of the paper is organized as follows. After reviewing the related works in the literature in Section 2, we present our method and theoretically analyze its performance to guide parameter selection in Section 3. The performance of the proposed method is evaluated with respect to the current state-of-the-art attacks in Section 4, and the paper is concluded in Section 5.

2. Related Work

The robustness of deep neural networks (DNNs) and their vulnerability against adversarial machine learning attacks have been investigated for several years since the seminal work of (1). Many attacks for both white-box and black-box settings were proposed. In white box settings, the attacker has access to the target model including its parameters. Hence, very effective and strong methods were introduced including FGSM(1), C&W attack (13), PGD attack (14). For the black-box setting, in which the adversary does not have any prior information about the target model, different approaches have been proposed. One of the most common approaches is introduced in (10). The idea is to use a substitute model for generating adversarial samples for an unknown target model, utilizing the transferability of adversarial samples to different CNNs. Another black-box attack approach is to estimate gradients to generate adversarial data without using a substitute model, e.g., ZOO (12), NES (11), SPSA (15), SRA (16) and adversarial scratches (17).

In addition to attacks, robustness of models and possible defense methods were investigated. (18) aims to improve classification performance by adding an adversarial attack module to the model. (19) proposes a defense method by analyzing the shared information between clean and perturbed data. (20) tries to eliminate perturbations with the help of adaptive compression and reconstruction. Also some adversarial benchmark datasets, such as (21), were proposed to evaluate robustness of models. Similar to attacks, these robustness efforts and defense methods were developed with CNNs in mind.

After the big success of transformers in natural language processing (NLP) (22), vision transformer (ViT) was introduced for image classification in 2020 (2). Outperforming the state-of-the-art CNNs, ViTs gained popularity rapidly. Many

different derivations of ViTs have been proposed since then, including DeiT (3), PiT (23), TNT (24), Le-ViT (4), and ConVit (6).

Because of the increasing popularity of transformers, their robustness has also become a popular research topic. Some initial research claimed that transformers are more robust because of their ability to catch global interactions with the help of patches while CNNs focus on local features (25; 26; 27; 28). On the other hand, some works claimed that ViTs are also vulnerable to adversarial machine learning attacks (29; 30; 31; 7).

Although most of the initial works that investigate the robustness of transformers used attacks which were initially designed for CNNs, several recent works proposed adversarial machine learning attacks which are designed specifically for transformers. (8) specifically targets the patch structure of transformers by selecting the patches according to how much attention they drew and applying the perturbations to these areas. (9) suggests that traditional gradient-based attacks are ineffective against transformers due to high gradient loss, hence they propose an attack which exploits dot-product mechanism of transformers.

(7) is the closest work to ours. Focusing on the low transferability between attacks designed for CNNs and transformers, authors propose a transferable attack called PNA for both CNNs and transformers by skipping attention gradients. While we study the same low transferability issue, we show that with a limited number of queries attackers can accurately identify the target architecture and select their attack strategy accordingly. Experiments presented in Section 4.2 demonstrate that the proposed method outperforms the PNA attack in (7) with a wide margin at the cost of a small number of queries.

3. Methodology

3.1. Problem Formulation

In a black-box setting, attacker objectives can be formulated as follows: (i) maximizing the probability of detecting target model's architecture to maximize the attack success rate, and (ii) minimizing the expected number of queries sent to the target model. We turn this unconstrained two-objective problem into a constrained single-objective problem:

$$\min \ E[N] \text{ s.t. } P(\text{target identification}) \geq \gamma, \tag{1}$$

where N represents the number of image samples that are queried to target model, $E[N]$ is the expectation of N, and γ is the lower bound for the probability of correct target identification.

This formulation corresponds to a sequential decision making problem in which the attacker tries to choose among a set $\mathcal{S} = \{A_1, \ldots, A_K\}$ of K substitute network architectures by sequentially sending query batches $\{Q_1, \ldots, Q_t, \ldots, Q_T\}$ to the target network. Each query batch consists of M adversarial images designed for K candidate networks. After each query Q_t, attacker decides to either continue with the next query batch Q_{t+1} or stop and choose one of the K candidates (i.e., $K + 1$ possible decisions). Note that the stopping time T, which denotes the number of query batches, and the resulting total number of queries $N = T \times M$ are random variables, which take value according to the stopping criterion.

Before presenting the optimum stopping criterion for this problem, let us first discuss how to form the candidate set $\mathcal{S}$, i.e., choosing its elements and the num-

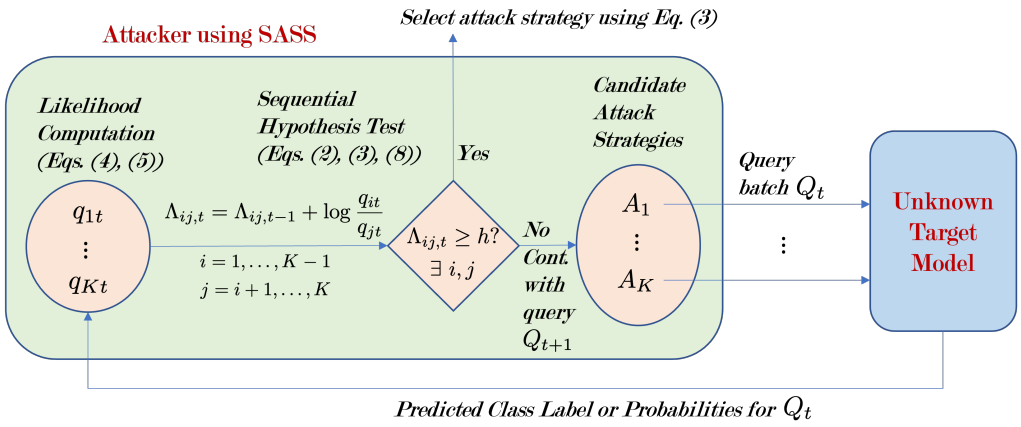

Figure 2: Digram of the proposed SASS method.

ber of elements K. Obviously, larger K means a richer set of options to select from $\mathcal{S}$, and thus a better chance to find a close match to the target network. Nevertheless, larger K also means more queries to the target network to evaluate the potential of candidate networks in $\mathcal{S}$, i.e., M grows with K. Hence, the substitute set size K poses a trade-off between minimizing $E[N]$ and maximizing $P(\text{target identification})$ in Eq. (1). A rule of thumb to decide on the substitute set size can be to include the most popular target network architectures in a parsimonious way and best attack techniques for them. We empirically analyze this trade-off in Section 4.

3.2. Sequential Attack Strategy Selection (SASS)

By selecting a set of popular target network models (e.g., ResNet, Inception, ViT, etc.) and the most effective white-box attack techniques against them (e.g., PGD (14), Patch-Fool (8), etc.) the attacker turns the optimization problem in Eq. (1) into a sequential multi-hypothesis testing problem. Given K hypotheses (i.e., candidate attack strategies), such as PGD for ResNet-50, PGD for Inception,

Patch-Fool for ViT, etc., the attacker aims to *select the best hypothesis as soon as possible and as accurately as possible*. Figure 2 summarizes the proposed SASS method.

For the binary case ($K = 2$), when the true hypothesis is either one of the two competing hypotheses:

$$\begin{aligned} H_0 &: A_1 \text{ is the target architecture} \\ H_1 &: A_2 \text{ is the target architecture}, \end{aligned} \tag{2}$$

the Sequential Probability Ratio Test (SPRT) is known to be optimum in terms of minimizing the expected number of samples $E[N]$ under both hypotheses while satisfying given constraints on the error probabilities $P(\text{selecting } A_2 | A_1 \text{ is true})$ and $P(\text{selecting } A_1 | A_2 \text{ is true})$ (32). The SPRT procedure computes the running log-likelihood ratio

$$\Lambda_t = \Lambda_{t-1} + \log \frac{q_{1t}}{q_{2t}}, \ \Lambda_0 = 0,$$

where q_{1t} and q_{2t} are the likelihoods of hypotheses A_1 and A_2 for query batch Q_t, and compares it with two thresholds to decide

$$\delta_t = \begin{cases} \text{select } A_1 & \text{if } \Lambda_t \geq a \\ \text{select } A_2 & \text{if } \Lambda_t \leq -b \\ \text{continue with query } Q_{t+1} & \text{if } -b < \Lambda_t < a. \end{cases}$$

We use the extension of SPRT, called matrix SPRT (32), for the multi-hypothesis

case ($K > 2$):

$$\begin{aligned} H_0 &: A_1 \text{ is the target architecture} \\ &\vdots \\ H_{K-1} &: A_K \text{ is the target architecture.} \end{aligned} \tag{3}$$

It computes the running log-likelihood ratio between the $K(K-1)/2$ unique pairs of hypotheses

$$\Lambda_{ij,t} = \Lambda_{ij,t-1} + \log \frac{q_{it}}{q_{jt}},\ \Lambda_{ij,0} = 0, \tag{4}$$

for $i = 1, \ldots, K-1,\ j = i+1, \ldots, K$, and compares them with two thresholds to decide

$$\delta_t = \begin{cases} \text{select } A_i & \text{if } \Lambda_{ij,t} \geq a_{ij} \text{ for some } i, j \\ \text{select } A_j & \text{if } \Lambda_{ij,t} \leq -b_{ij} \text{ for some } i, j \\ \text{continue with query } Q_{t+1} & \text{if } -b_{ij} < \Lambda_{ij,t} < a_{ij} \text{ for all } i, j. \end{cases} \tag{5}$$

The proposed SASS method stops the first time when $\Lambda_{ij,t}$ crosses a threshold for an i, j pair, i.e., at the random time

$$T = \min\{t : \Lambda_{ij,t} \notin (-b_{ij}, a_{ij}) \text{ for some } i, j\},$$

and chooses the attack strategy according to Eq. (5). In case more than one i, j pairs stop the test at the same time, among the qualifying attack strategies we select the one with the strongest likelihood.

As noted in Section 3.1, the K value controls a trade-off between minimizing the number of queries and maximizing the probability of finding a close match to

the target network in the candidate set. When the candidate set includes the target network (i.e., no mismatch between the true hypothesis and the set of tested hypotheses), SPRT is known to quickly and accurately identify the true hypothesis. In the case of a mismatch, we expect the proposed SASS based on matrix SPRT to find the closest architecture in the candidate set to the target network. Larger K values increase the probability for finding a close match and forming an effective attack to the target network at the expense of more queries in a batch. However, with a larger K value, the drop in the number of batches to build an effective attack may outweigh the increase in the number of queries in a batch when we look at the total number of queries on average. Comprehensive experimental results are presented in Section 4 to analyze this trade-off.

The likelihood q_{it} of attack strategies can be obtained in two different ways depending on the attacker's access to the target network's output.

Probability-accessible setting: If the attacker has access to the class probabilities for target network predictions [1], then the attack success likelihood of strategy A_i for query Q_t can be computed as

$$q_{it} = 1 - P(\text{true class of} X_{it} | \tilde{X}_{it}), \tag{6}$$

where $\tilde{X}_{it}$ is the adversarial image generated using attack strategy A_i and X_{it} is the original image. Note that, in this case, only one adversarial image is used per attack strategy ($M = K$), and as a result $Q_t = \{\tilde{X}_{1t}, \ldots, \tilde{X}_{Kt}\}$. For example, if the true label of X_{it} is "dog", then the likelihood q_{it} is how far the predicted "dog"

[1] If the target network provides confidence scores for different classes, class probabilities can be obtained by normalizing these scores.

Algorithm 1 SASS algorithm

Input: error probability constraint β, set of attack strategies $\mathcal{S}$ with size K, number of images for each attack strategy in a query batch L (e.g., $L = 1$ for probability-accessible setting and $L > 1$ for label-only setting)

Output: selected attack strategy

Set threshold $h = -\log \frac{\beta}{K-1}$ (Eq. (10))
Initialize $\Lambda_{ij} \leftarrow 0$ for each pair of attack strategies in $\mathcal{S}$
while $-h < \Lambda_{ij} < h$ for all i, j **do**
    Send query batch consisting of L perturbed images for each attack strategy
    Receive class probabilities or only the predicted label for each image and compute likelihoods q_{ij} as in Eqs. (6) or (7), respectively.
    Update $\Lambda_{ij} \leftarrow \Lambda_{ij} + \log \frac{q_i}{q_j}$ for all i, j (Eq. (4))
end while
Consider pair $(ij)^* = \arg\max_{i,j} |\Lambda_{ij}|$
if $\Lambda_{(ij)^*} \geq h$ **then**
    Select attack strategy i
else
    Select attack strategy j
end if

probability for adversarial image $\tilde{X}_{it}$ is from 1.

Label-only setting: On the other hand, if the attacker has access to the predicted label only, but not the predicted class probabilities, then the attack success rate in a mini batch can be used as the likelihood:

$$q_{it} = \frac{\text{\# misclassifications in} \{\tilde{X}_{it}^1, \ldots, \tilde{X}_{it}^L\}}{L}. \tag{7}$$

In this case, a mini batch of L adversarial images is used for each attack strategy in each query ($M = KL$). The proposed SASS algorithm is summarized in Algorithm 1.

3.3. Performance Analysis

While in the binary case there are two error probabilities $\alpha_0 = P_0(\delta_T = A_1)$ and $\alpha_1 = P_1(\delta_T = A_0)$, in the general case there are $K(K-1)$ error cases with the probabilities $\alpha_{ij} = P_i(\delta_T = A_j),\ i = 1, \ldots, K,\ j \neq i$, where P_i denotes the probability under the true hypothesis. If the hypothesis set $\mathcal{S}$ does not include the target network, then P_i denotes the probability under the closest hypothesis in $\mathcal{S}$, i.e., the best performing attack strategy in $\mathcal{S}$ against the target network.

We can find an upper bound for each error probability using the Wald's likelihood ratio identity (32) as follows

$$\begin{aligned}\alpha_{ij} =& P_i(\Lambda_{ij,T} \leq -b_{ij}) = E_i[1\{\Lambda_{ij,T} \leq -b_{ij}\}] = E_j[e^{\Lambda_{ij,T}} 1\{\Lambda_{ij,T} \leq -b_{ij}\}] \\ \leq& e^{-b_{ij}} P_j(\delta_T = A_j) \leq e^{-b_{ij}},\end{aligned} \tag{8}$$

where $1\{\cdot\}$ is the indicator function which takes the value one if its argument is true and zero otherwise. Similarly,

$$\begin{aligned}\alpha_{ji} =& P_j(\Lambda_{ij,T} \geq a_{ij}) = E_j[1\{\Lambda_{ij,T} \geq a_{ij}\}] = E_i[e^{-\Lambda_{ij,T}}, 1\{\Lambda_{ij,T} \geq a_{ij}\}] \\ \leq& e^{-a_{ij}} P_i(\delta_T = A_i) \leq e^{-a_{ij}}.\end{aligned} \tag{9}$$

These relationships between the error probabilities and the thresholds can be used to set the thresholds to a value which will satisfy an error probability constraint. For instance, if α_{ij} is wanted to be at most $\bar{\alpha}$, setting

$$-b_{ij} = \log \bar{\alpha}$$

can satisfy $\alpha_{ij} \leq \bar{\alpha} = e^{-b_{ij}}$. Similarly,

$$a_{ij} = -\log \bar{\alpha}$$

can satisfy $\alpha_{ji} \leq \bar{\alpha} = e^{-a_{ij}}$.

While Eqs. (5), (8), (9) are in a general form with potentially different thresholds for testing each pair of hypotheses, in practice one can typically use the same threshold $h = a_{ij} = b_{ij}$ and the same error probability constraint $\bar{\alpha}$ for all i, j by setting

$$h = -\log \bar{\alpha}$$

Note that given a target model and the corresponding best attack strategy A_i, the total error probability is given by the sum $\sum_{j \neq i} \alpha_{ij}$. If the upper bound for each error case is set the same as $\bar{\alpha}$, then the total error probability must be bounded by $(K-1)\bar{\alpha}$. That is, given a total error probability constraint β, one can set

$$h = -\log \frac{\beta}{K-1}. \tag{10}$$

4. Experiments

In this section we will evaluate our SASS method and compare its performance with different types of state-of-the-art black-box adversarial attacks. First, we will investigate our method's target model detection success within different scenarios. Then, we will compare our approach with transferability-based adversarial attacks. Finally, we will discuss our method's performance when it is compared to query-based black-box attacks. The following experimental settings will be used throughout this section:

Dataset: Following the common practice in literature we have randomly selected 5000 images from the ImageNet 2012 validation set (33) and used this dataset for our experiments, e.g., (8), (9), (11), (29) use radomly selected 2500, 1000, 1000, 1000 images from the ImageNet validation set, respectively.

Target Models: We use the following popular image classification models as target in our experiments: InceptionV3 (34), ResNet-50 (35), DeiT-S (3), DeiT-B (3), Deit-Ti (3), ViT-T (2), ViT-B (2), Vgg16 (36), Vgg19 (36), ResNet-152 (35). These models are publicly available, e.g., in timm (37) and torchvision (38) libraries.

Adversarial attacks: Throughout our experiments we use several adversarial attacks including FGSM (1), C&W (13), PGD (14), PatchFool (8), PNA (7), NES (11).

Evaluation Metrics: The success rate of an attack is defined as the average error percentage of the target model on the generated dataset over 1000 trials. For the proposed SASS method, success rate (SR) is given by

$$SR = \sum_{i=1}^{K} p_i \times SR_i \tag{11}$$

where K is the number of candidate attack strategies (hypotheses), $p_i \in [0, 1]$ is the frequency of trials in which attack strategy i is selected, and SR_i is the success rate of attack strategy i. For SASS, we also define the detection rate (DR) as the selection frequency of best attack strategy.

4.1. Performance of Target Model Identification

We first evaluate our target identification performance considering both the architectural match and mismatch cases.

Attack Strategy	Target	Threshold	AQ	DR	SR
SASS-2	ResNet50	0.8	2.7	100	100
SASS-2	ResNet50	8	5.16	100	100
SASS-2	ResNet50	20	8.78	100	100
SASS-2	ResNet50	40	16.35	100	100
SASS-2 (label only)	ResNet50	0.8	20	100	100
SASS-2 (label only)	ResNet50	8	71.7	100	100
SASS-2 (label only)	ResNet50	20	136.16	100	100
SASS-2 (label only)	ResNet50	40	235.28	100	100
SASS-2	DeiT-S	0.8	9	95.4	88.50
SASS-2	DeiT-S	8	78.9	100	93.78
SASS-2	DeiT-S	20	172.5	100	93.78
SASS-2	DeiT-S	40	348	100	93.78
SASS-2 (label only)	DeiT-S	0.8	23	100	93.78
SASS-2 (label only)	DeiT-S	8	119.4	100	93.78
SASS-2 (label only)	DeiT-S	20	248.5	100	93.78
SASS-2 (label only)	DeiT-S	40	481.2	100	93.78

Table 1: The performance of the proposed SASS method with 2 attack strategies (PGD trained on ResNet-50, PatchFool trained on DeiT-S) against target models ResNet-50 and DeiT-S in terms of average number of queries (AQ), detection rate (DR) and success rate (SR).

4.1.1. Architectural Match Case

In this best-case scenario, SASS hypothesis set $\mathcal{S}$ includes the target model. Table 1 shows the attack success rate (SR) and target model detection rate (DR) as a function of the chosen threshold and the resulting average number of queries (AQ) for a binary hypothesis set, which includes the target model. The two attack strategies in SASS-2 are PGD trained on ResNet-50 and PatchFool trained on DeiT-S. When the target model is ResNet-50, SASS achieves perfect DR and SR even with 2.7 queries on average by accessing the class probability predictions of the target model. When it only accesses the predicted labels, it uses $L = 10$ adversarial images for each attack strategy, i.e., each batch consists of 20 images, which naturally increases AQ compared to the probability-accessible setting. Us-

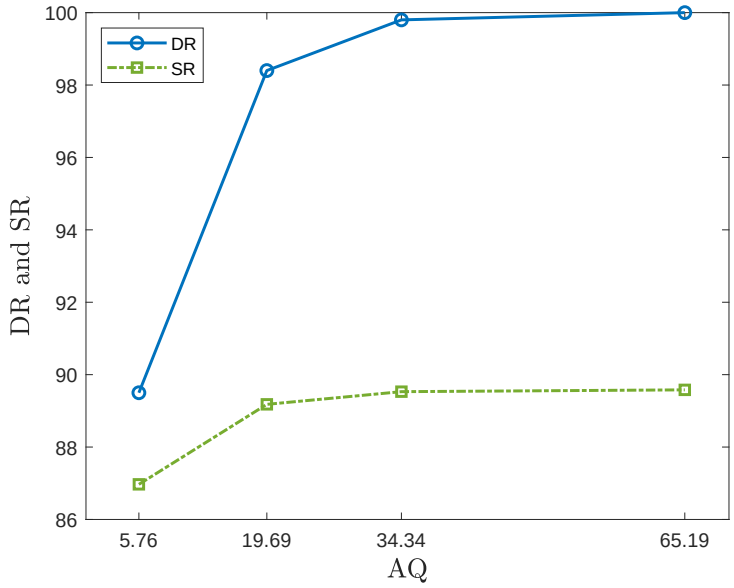

Figure 3: Detection Rate (DR) and Success Rate (SR) of the proposed SASS method with 2 attack strategies (PGD trained on Inception and PNA trained on ViT) against target model VGG-16.

ing only one batch SASS achieves again perfect DR and SR. When the target model is DeiT-S, SASS-2 detects the target model correctly in all trials except for the smallest threshold in the probability-accessible setting, which attains 95.4% DR by using 9 queries on average. Although SASS-2 perfectly detects the target model in other cases, its SR is not 100% because the SR of the PatchFool attack on DeiT-S is 93.78%. By identifying the target model, SASS turns the black-box attack into a white-box attack.

4.1.2. Architectural Mismatch Case

In the case when the hypothesis set $\mathcal{S}$ does not include the target model, we expect SASS to identify the closest architecture in S. In Figure 3, the performance of SASS-2 without an exact match to the target model is shown as a function of the

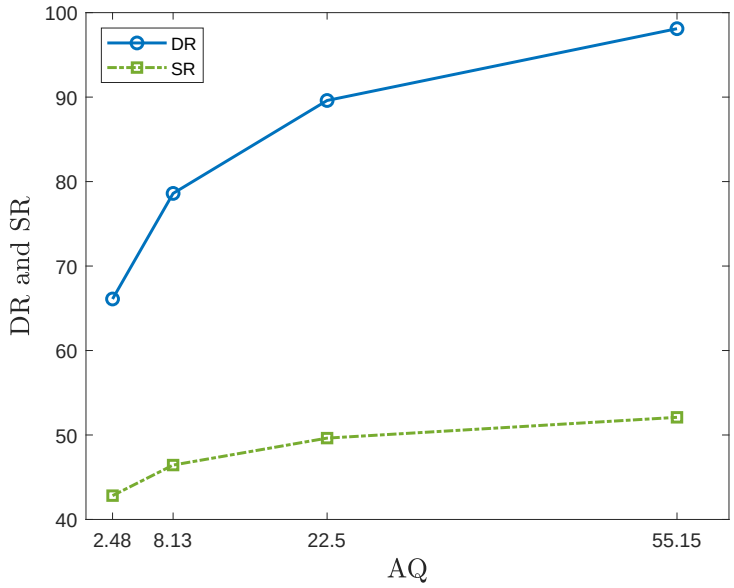

Figure 4: Detection Rate (DR) and Success Rate (SR) of the proposed SASS method with 2 attack strategies (C&W trained on Inception and PNA trained on DeiT-S) against target model ViT-B.

average number of queries. The considered attack strategies is $\mathcal{S}$ are PGD trained on Inception and PNA trained on ViT while the target model is VGG-16. With average query number of 65.19, we achieve 100% percent of DR and 89.58% SR. DR here denotes the detection rate of Inception as the substitute model since it is a CNN-based model like VGG-16. Note that SR is dependent on the chosen attack strategy's transferability performance on the target model. This causes relatively low SR compared to the exact match case (Table 1).

Especially, when we have transformers as target models, transferability is a more serious issue since adversarial attacks against transformers are not as transferable as attacks against CNNs. For example, Figure 4 shows the performance of SASS-2 which includes C&W with Inception and PNA with DeiT-S as attack strategies against target model ViT-B. In this case, DR refers to the selection of

DeiT-S as the substitue model. Even when DR reaches 98.1%, SR stays at 52% due to the low transferability of PNA trained on DeiT-S to ViT-B.

The solution to this problem is enhancing the hypothesis set with more attack strategies. We conducted an extensive experiment in which $\mathcal{S}$ consists of 3, 4, and 5 attack strategies. For SASS-3, we consider PGD with InceptionV3, PGD with DeiT-S, and PGD with ViT-B in $\mathcal{S}$. Then, in addition to these three, we use PGD with ResNet-50 in SASS-4 and also PGD with VGG-16 in SASS-5. Ten different target models (InceptionV3, ResNet-50, DeiT-S, DeiT-B, DeiT-Ti, Vit-T, ViT-B, VGG16, VGG19, ResNet-152) are attacked in the label-only setting using $L = 5$ perturbed images per attack strategy in a query batch and four different thresholds $(0.8, 8, 20, 40)$. The complete SR and AQ values against all target models are given in Table 6. Figure 5 summarizes the SR vs. AQ performances of SASS-3, SASS-4, and SASS-5 averaged over the ten target models. The average SR of SASS-3, SASS-4, and SASS-5 reaches or exceeds 90%. The four-hypothesis case seems to be the ideal setting in this experiment as SASS-4 clearly outperforms SASS-3 and SASS-5. Comparing SASS-4 to SASS-3 we see that the reduction in the average number of batches outweighs the increase in the batch size $M = KL$, where $K = 3, 4$ for SASS-3 and SASS-4, respectively. We see the opposite case in SASS-5, which significantly underperforms due to the increase in the batch size. Note that we tried to add complementary models while expanding the hypothesis set. In this particular scenario, while adding PGD with ResNet-50 in SASS-4 yields improved overall performance, the cost of adding PGD with VGG-16 in SASS-5 outweighs its benefit, resulting in a reduced overall performance.

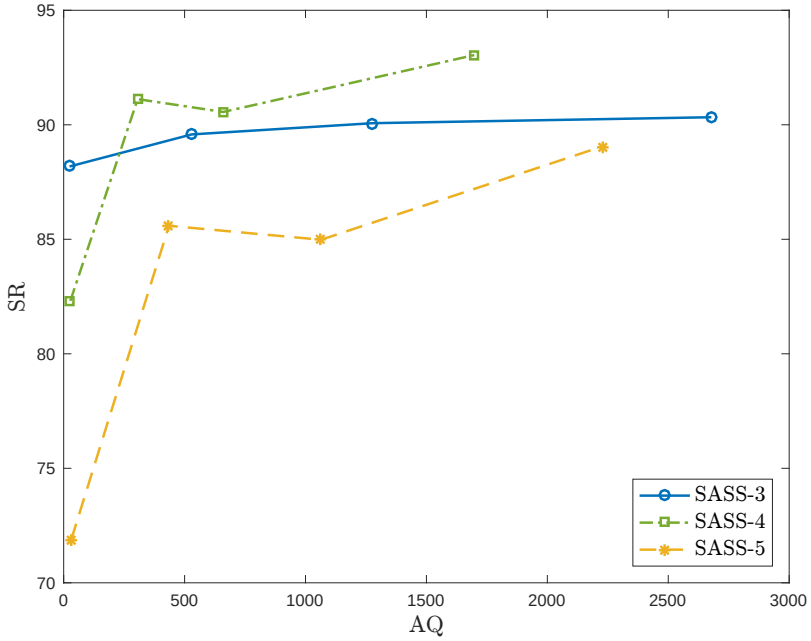

Figure 5: Success Rate (SR) as a function of average query number (AQ) for the proposed SASS method with 3, 4, and 5 attack strategies. PGD with ResNet-50 is added to the three attack strategies (PGD with InceptionV3, DeiT-S, ViT-B) of SASS-3 to obtain SASS-4. And PGD with VGG-16 is further added to obtain SASS-5.

4.2. Comparison against Transfer Learning-Based Black-Box Attacks

Transfer learning-based black-box attacks are practical because of their simple approach. In this type of attacks, an adversarial set is generated using a surrogate model, and assuming transferability this set is used for attacking to the target model. To test our performance against transfer learning-based attacks, we generated eight adversarial datasets by training FGSM, PGD, PatchFool/C&W, and PNA on DeiT-S and InceptionV3 as surrogate models. Then, we used these adversarial sets to attack ten different target models.

Tables 2 and 3 present the attack success rate (SR) of the proposed SASS-4 method (PGD with InceptionV3, DeiT-S, ViT-B, ResNet-50) compared to FGSM,

PGD, PatchFool, PNA trained on DeiT-S and FGSM, PGD, C&W, PNA trained on InceptionV3, respectively. Since PatchFool is specifically designed for transformer models using image patches, it cannot be trained on a CNN-based surrogate model like InceptionV3. Hence, we replaced it with C&W in Table 3. SASS-4-vs and SASS-4-s correspond to the very small (0.8) and small (8) threshold cases. The average query numbers for our method are shown in parentheses. At the cost of a reasonably low number of queries, the proposed method achieves significantly higher SR on average than the popular transfer-based black-box attacks, including the recent PNA attack (7), which focuses on the same low-transferability issue between CNNs and transformers. While the benchmark black-box methods suffer considerable performance loss against several target models (e.g., FGSM against ViT-B, PGD against ViT-B, PatchFool against DeiT-B, PNA against ResNet-152), our approach consistently achieves high SR values against a wide range of target models, including the ones that are not included in the hypothesis set $\mathcal{S}$ (i.e., last six target models). These results indicate that the transferability of the proposed SASS method is much higher than the existing black-box methods, especially when the surrogate model is CNN-based (Table 3).

4.3. Comparison against Query-Based Black-Box Attacks

Following a fully data-driven approach, query-based black-box attacks directly aim at the target model without using a substitute model. Particularly, they aim to estimate the gradient of the target model's classification loss with respect to the input image by sending queries and generate adversarial perturbations via this estimation. While not being limited by a surrogate is an important advantage, the number of queries to target model is typically prohibitive (on the order of tens of thousands) in practice. Query-based attacks are usually studied under two dif-

Target Model	FGSM (1)	PGD (14)	PFool (8)	PNA (7)	SASS-4-vs	SASS-4-s
InceptionV3* (34)	58.5	69.56	33.58	51.52	81.71 (21.8)	**98.27** (174.2)
DeiT-S (3)	41.28	**100**	93.78	95.26	71.15 (20.2)	88.52 (79.4)
ViT-B (2)	25.48	56.94	23.81	52.64	98.6 (20)	**99.56** (45.2)
ResNet-50* (35)	67.14	81.68	26.9	54.3	94.69 (28.4)	**100** (273.6)
VGG-16* (36)	80.72	90.76	32.36	62.16	90.68 (43.2)	**91.85** (939.8)
DeiT-T (3)	53.5	**93.48**	29.87	78.2	74.844 (22.2)	88.44 (134.2)
DeiT-B (3)	34	**86.7**	19.23	73.48	67.84 (20)	79.38 (75)
ResNet-152* (35)	57.16	71.64	24.54	47.9	80.04 (25.8)	**85.44** (252)
VGG-19* (36)	79.56	89.62	30.79	58.7	82.89 (47.2)	**91.06** (897.2)
ViT-T (2)	59.6	**90.1**	53.24	72.36	80.65 (26.8)	88.7 (188.6)
Average	55.69	83.04	36.81	64.65	82.31 (27.56)	**91.12** (305.92)

Table 2: Transferability comparison in terms of success rate (SR) between the popular transfer-based black-box attacks FGSM, PGD, PatchFool (PFoll), PNA, which are trained on DeiT-S, and the proposed SASS method with four candidate attack strategies (PGD with InceptionV3, DeiT-S, ViT-B, ResNet-50). SASS-4-vs and SASS-4-s correspond to the very small (0.8) and small (8) threshold cases. Signficantly higher average SR value of SASS-4-s indicates better transferability. Numbers in parentheses show the average number of queries. Note that the models marked with * are CNN based, while the others are transformer based.

ferent attack objectives, namely targeted and untargeted attacks. While untargeted attacks only aim the misclassication of input into any wrong label, targeted attacks particularly aim misclassification into a specific wrong label.

For untargeted attacks, our results in Sections 4.1 and 4.2 can be used to compare SASS with query-based black-box attacks. For example, ZOO attack (12) achieves 88.9% SR against InceptionV3 with more than $100,000$ queries for 150 images from the ImageNet test set. In the same case, if the hypothesis set includes InceptionV3, as shown in Table 6, SASS-3 achieves 99.98% SR with 17.25 AQ; SASS-4 achieves 98.27% SR with 174.2 AQ; and SASS-5 achieves 98.63% SR with 188.5 AQ. Moreover, if the hypothesis set does not include the target model, as shown in Figure 5, SASS-3 achieves on average 89.58% SR with 529.68 AQ; SASS-4 achieves on average 91.12% SR with 305.92 AQ; and SASS-5 achieves on average 85.59% SR with 435.27 AQ.

For targeted attacks, considering InceptionV3 as the target model, we compare

Target Model	FGSM (1)	PGD (14)	CW (13)	PNA (7)	SASS-4-vs	SASS-4-s
InceptionV3* (34)	48.56	**99.98**	45.1	91.02	81.71 (21.8)	98.27 (174.2)
DeiT-S (3)	36.94	41.5	36.56	31.4	71.15 (20.2)	**88.52** (79.4)
ViT-B (2)	23.78	28.48	23.7	21.98	98.6 (20)	**99.56** (45.2)
ResNet-50* (35)	67.78	81.12	67.47	50.18	94.69 (28.4)	**100** (273.6)
VGG-16* (36)	81.5	89.58	81.28	58.56	90.68 (43.2)	**91.85** (939.8)
DeiT-T (3)	52.08	55.28	51.62	42.88	74.844 (22.2)	**88.44** (134.2)
DeiT-B (3)	31.79	37.52	31.45	26.92	67.84 (20)	**79.38** (75)
ResNet-152* (35)	45.26	30.92	45.92	55.02	80.04 (25.8)	**85.44** (252)
VGG-19* (36)	79.34	87.12	78.96	55.88	82.89 (47.2)	**91.06** (897.2)
ViT-T (2)	57.32	61.44	56.78	41.28	80.65 (26.8)	**88.7** (188.6)
Average	52.435	61.294	51.884	47.512	82.31 (27.56)	**91.12** (305.92)

Table 3: Transferability comparison in terms of success rate (SR) between the popular transfer-based black-box attacks FGSM, PGD, PatchFool (PFoll), PNA, which are trained on InceptionV3, and the proposed SASS method with four candidate attack strategies (PGD with InceptionV3, DeiT-S, ViT-B, ResNet-50). SASS-4-vs and SASS-4-s correspond to the very small (0.8) and small (8) threshold cases. Signficantly higher average SR value of SASS-4-s indicates better transferability. Numbers in parentheses show the average number of queries. Note that the models marked with * are CNN based, while the others are transformer based.

our method with the query-based NES attack (11), which estimates the gradients via queries and then utilizes PGD attack based on the estimated gradients. Table 4 shows that SASS-5 with the same hypothesis set in the previous section, which includes InceptionV3, achieves 98.36% SR in the label-only setting by using only 55.61 queries on average. Whereas, the fully-data driven NES method needs a prohibitive 2.7 million queries to attain 90% SR in the same label-only setting. When all class probabilities are available from the target model, NES achieves 99.2% SR using $11,550$ queries, which corresponds to the most query-limited (QL) performance of NES. And when only the partial information (PI) of top predicted probability is available from the target model, NES can reach an SR of 93.6% with $49,624$ queries. The sheer difference between the average number of queries of the proposed SASS and NES is due to the hypothesis testing mechanism of SASS which effectively utilizes the prior information on target models. The fact that target models are predominantly based on a handful popular architectures

provide valuable prior information. When the target model architecture is in the hypothesis set, SASS is able to drastically reduce the number of queries needed to successfully attack an unknown target model. However, specifically for targeted attacks, mismatch between the target model and hypothesis set can prevent the effective use of such prior information, as discussed next.

For targeted attack, transferability is very low even between similar CNN-based architectures, as seen in Table 5. This is in contrast with untargeted attack, for which transferability is an issue if the target and substitute model architectures are not of similar type. Even in such mismatch cases for untargeted attack, discussed in Tables 2 and 3, the SR values are much higher than the off-diagonal SR values in Table 5. Because of this complete lack of transferability for targeted attacks, our SASS method unfortunately does not work well for targeted attack if the hypothesis set does not include the target model. Hence, for SASS to be successful in targeted attacks, its hypothesis set needs to be large enough to cover the target model, i.e., should cover a high percentage of popular image recognition models. This will inevitably result in loss of data efficiency. However, even if all ten of the popular models considered in Table 6 or even a wider set is included in the hypothesis set, it is reasonable to expect SASS to need less than millions of queries (unlike the NES attack) based on the results in Table 6 for SASS-5, which uses less than 10,000 queries in all cases.

5. Conclusions

In this paper, we showed that a novel architecture-agnostic black-box attack design method, called sequential attack selection strategy (SASS), can address the architectural uncertainty in target image recognition in a query-efficient way.

Attack Method	SR	AQ
NES-QL (11)	**99.2**	11,550
NES-PI (11)	93.6	49,624
NES-LO (11)	90	2.7×10^6
SASS-5	98.36	**55.61**

Table 4: Targeted attack comparison between our attack and query-limited (QL), partial information (PI), and label-only (LO) NES attacks against the target model InceptionV3. Results for SASS-5 are obtained in the label-only setting. The hypothesis set of SASS-5 includes PGD with InceptionV3, DeiT-S, ViT-B, ResNet-50, and VGG-16. NES attacks also use the PGD attack with their gradient estimates. Having the target model in the hypothesis set enables huge query savings compared to the fully-data driven NES method.

Adversarial Dataset	InceptionV3	ResNet50	ViT-B
InceptionV3 & Pgd	98.36	0.26	0.08
ResNet50 & Pgd	0.18	99.34	0.14
ViT-B & Pgd	0.1	0.1	99.32

Table 5: Targeted attack success rates (SRs) of adversarial datasets generated by PGD trained on InceptionV3, ResNet-50, and ViT-B on target models InceptionV3, ResNet-50 and ViT-B. There is a complete lack of transferability for targeted attacks even between similar CNN-based architectures as shown by the very small off-diagonal SRs.

SASS aims to identify the target model architecture as close as possible by sending queries. It chooses its candidate set of substitute models (hypothesis set) using prior information (domain knowledge) on popular image recognition models (i.e., CNNs and transformers), and queries the target model in a data-driven way to identify the closest match in the hypothesis set. This hybrid nature of SASS, which is based on both prior information and queries, provides a trade-off between robustness to architecture heterogeneity and data efficiency. The success of proposed SASS method is limited by the diversity of hypothesis set. As the hypothesis set grows, SASS becomes more robust to architectural heterogeneity, but loses from data efficiency. The number of queries required by a large hypothesis set may be prohibitive for some applications in which queries are costly or

significantly increase the risk of detection and prevention.

Through experiments on real data, we showed that the size of hypothesis set can be set to obtain a balanced treatment by analyzing the Success Rate (SR) vs. Average Query (AQ) number. Particularly, experiments on randomly chosen 5000 images from the ImageNet dataset yielded SASS with 4 hypotheses as the optimum choice. Extensive comparisons with state-of-the-art black-box attacks illustrate that this balanced usage of prior information with a reasonable number of queries gives SASS an edge over the existing methods.

We also noted that, for performing targeted attacks, the trade-off between robustness vs. data efficiency should be severely skewed towards robustness due to the very low transferability of targeted attack techniques even among similar architectures. Larger hypothesis sets will increase the probability of including the target model at the expense of more queries. If the hypothesis set does not include the target model, SASS cannot generate successful targeted attacks due to low transferability.

As future work, we intend to extend our SASS method to video classifiers and work on defense mechanisms against adversarial machine learning attacks targeting different types of architectures. Also, the knowledge base of SASS should be regularly updated with new effective attack and defense methods from the literature, and its hypothesis set should be adjusted accordingly as needed.

6. Acknowledgements

This work was supported by the U.S. National Science Foundation (NSF) [grant number 2029875].

References

[1] I. J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572 (2014).

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations, 2021.
URL https://openreview.net/forum?id=YicbFdNTTy

[3] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, in: International Conference on Machine Learning, PMLR, 2021, pp. 10347–10357.

[4] B. Graham, A. El-Nouby, H. Touvron, P. Stock, A. Joulin, H. Jégou, M. Douze, Levit: a vision transformer in convnet's clothing for faster inference, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 12259–12269.

[5] Z. Chen, L. Xie, J. Niu, X. Liu, L. Wei, Q. Tian, Visformer: The vision-friendly transformer, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 589–598.

[6] S. d'Ascoli, H. Touvron, M. L. Leavitt, A. S. Morcos, G. Biroli, L. Sagun, Convit: Improving vision transformers with soft convolutional inductive bi-

ases, in: International Conference on Machine Learning, PMLR, 2021, pp. 2286–2296.

[7] Z. Wei, J. Chen, M. Goldblum, Z. Wu, T. Goldstein, Y.-G. Jiang, Towards transferable adversarial attacks on vision transformers, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 2668–2676.

[8] Y. Fu, S. Zhang, S. Wu, C. Wan, Y. Lin, Patch-fool: Are vision transformers always robust against adversarial perturbations?, in: International Conference on Learning Representations, 2022.
URL https://openreview.net/forum?id=28ib9tf6zhr

[9] G. Lovisotto, N. Finnie, M. Munoz, C. K. Mummadi, J. H. Metzen, Give me your attention: Dot-product attention considered harmful for adversarial patch robustness, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 15234–15243.

[10] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, A. Swami, Practical black-box attacks against machine learning, in: Proceedings of the 2017 ACM on Asia conference on computer and communications security, 2017, pp. 506–519.

[11] A. Ilyas, L. Engstrom, A. Athalye, J. Lin, Black-box adversarial attacks with limited queries and information, in: International Conference on Machine Learning, PMLR, 2018, pp. 2137–2146.

[12] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, C.-J. Hsieh, 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, pp. 15–26.

[13] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 ieee symposium on security and privacy (sp), Ieee, 2017, pp. 39–57.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, in: International Conference on Learning Representations, 2018.
URL https://openreview.net/forum?id=rJzIBfZAb

[15] J. Uesato, B. O'donoghue, P. Kohli, A. Oord, Adversarial risk and the dangers of evaluating against weak attacks, in: International Conference on Machine Learning, PMLR, 2018, pp. 5025–5034.

[16] C. Lin, S. Han, J. Zhu, Q. Li, C. Shen, Y. Zhang, X. Guan, Sensitive region-aware black-box adversarial attacks, Information Sciences 637 (2023) 118929.

[17] L. Giulivi, M. Jere, L. Rossi, F. Koushanfar, G. Ciocarlie, B. Hitaj, G. Boracchi, Adversarial scratches: Deployable attacks to cnn classifiers, Pattern Recognition 133 (2023) 108985.

[18] S. Yang, J. Li, T. Zhang, J. Zhao, F. Shen, Advmask: A sparse adversarial attack-based data augmentation method for image classification, Pattern Recognition (2023) 109847.

[19] X. Yu, N. Smedemark-Margulies, S. Aeron, T. Koike-Akino, P. Moulin,

M. Brand, K. Parsons, Y. Wang, Improving adversarial robustness by learning shared information, Pattern Recognition 134 (2023) 109054.

[20] Z.-H. Niu, Y.-B. Yang, Defense against adversarial attacks with efficient frequency-adaptive compression and reconstruction, Pattern Recognition 138 (2023) 109382.

[21] M. Pintor, D. Angioni, A. Sotgiu, L. Demetrio, A. Demontis, B. Biggio, F. Roli, Imagenet-patch: A dataset for benchmarking machine learning robustness against adversarial patches, Pattern Recognition 134 (2023) 109064.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[23] B. Heo, S. Yun, D. Han, S. Chun, J. Choe, S. J. Oh, Rethinking spatial dimensions of vision transformers, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 11936–11945.

[24] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, Y. Wang, Transformer in transformer, Advances in Neural Information Processing Systems 34 (2021) 15908–15919.

[25] A. Aldahdooh, W. Hamidouche, O. Deforges, Reveal of vision transformers robustness against adversarial attacks, arXiv preprint arXiv:2106.03734 (2021).

[26] P. Benz, S. Ham, C. Zhang, A. Karjauv, I. S. Kweon, Adversarial robustness comparison of vision transformer and mlp-mixer to cnns, arXiv preprint arXiv:2110.02797 (2021).

[27] M. M. Naseer, K. Ranasinghe, S. H. Khan, M. Hayat, F. Shahbaz Khan, M.-H. Yang, Intriguing properties of vision transformers, Advances in Neural Information Processing Systems 34 (2021) 23296–23308.

[28] R. Shao, Z. Shi, J. Yi, P.-Y. Chen, C.-J. Hsieh, On the adversarial robustness of vision transformers, arXiv preprint arXiv:2103.15670 (2021).

[29] S. Bhojanapalli, A. Chakrabarti, D. Glasner, D. Li, T. Unterthiner, A. Veit, Understanding robustness of transformers for image classification, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10231–10241.

[30] K. Mahmood, R. Mahmood, M. Van Dijk, On the robustness of vision transformers to adversarial examples, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 7838–7847.

[31] M. Naseer, K. Ranasinghe, S. Khan, F. Khan, F. Porikli, On improving adversarial transferability of vision transformers, in: International Conference on Learning Representations, 2022.
URL https://openreview.net/forum?id=D6nH3719vZy

[32] A. Tartakovsky, I. Nikiforov, M. Basseville, Sequential analysis: Hypothesis testing and changepoint detection, CRC Press, 2014.

[33] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang,

A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (3) (2015) 211–252.

[34] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.

[35] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[36] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[37] R. Wightman, Pytorch image models.
URL `https://github.com/rwightman/pytorch-image-models`

[38] Pytorch, Torchvision pre-trained models.
URL `https://pytorch.org/vision/stable/models.html`

Method	InceptionV3	DeiT-S	ViT-B	ResNet-50	VGG-16	DeiT-T	DeiT-B	ResNet-152	VGG-19	ViT-T
SASS-3-vs	99.98	98.83	98.27	80.54	89.83	92.46	83.77	71.58	79.23	87.28
	17.25	15.45	15.15	31.8	43.8	18.45	16.95	25.5	36.6	21.3
SASS-3-s	99.98	97.66	98.27	81.47	90.37	93.48	85.44	73.83	85.58	89.72
	140.85	64.65	43.95	902.85	1764.3	124.8	63.9	597.3	1427.85	166.35
SASS-3-l	99.98	98.83	99.56	81.29	90.65	93.48	85.86	71.56	89.62	89.84
	294.285	128.55	63.75	2342.4	4646.85	205.5	105.0	1188	3497.55	311.25
SASS-3-vl	99.98	100.0	100.0	81.54	88.93	93.48	86.49	73.3	89.62	89.97
	552.63	184.35	89.4	5038.05	9591.15	382.8	184.35	2555.7	7600.8	602.25
SASS-4-vs	81.71	71.15	98.60	94.69	90.68	74.84	67.84	80.04	82.89	80.65
	21.8	20.2	20.0	28.4	43.2	22.2	20.0	25.8	47.2	26.8
SASS-4-s	98.27	88.52	99.56	100	91.85	88.44	79.38	85.44	91.06	88.70
	174.2	79.4	45.2	273.6	939.8	134.2	75.0	252	897.2	188.6
SASS-4-l	96.27	84.97	99.56	100	92.13	89.31	77.64	86.08	91.31	88.33
	369.0	126.2	61.6	616.4	2122	264.4	109.8	532.6	1985.4	411.4
SASS-4-vl	99.98	96.67	100	100	92.38	91.95	82.31	86.08	91.52	89.59
	698.8	186.4	102.0	1264	8059.4	400.0	166.0	1200.4	4192.4	734.4
SASS-5-vs	84.29	53.51	35.32	93.11	91.95	72.28	48.84	82.26	81.30	75.89
	25.75	25.0	25.0	32.0	52.0	26.5	25.25	29.75	41.25	28.5
SASS-5-s	98.63	82.24	61.88	100	92.01	88.98	69.47	85.87	89.23	87.57
	188.5	97.0	62.5	345.5	1529.0	171.0	86.0	331.0	1285.25	257.0
SASS-5-l	95.21	78.37	68.18	99.81	92.25	85.587	64.67	85.93	91.48	88.31
	413.75	136.0	80.5	820.25	4118.5	332.5	128.25	695.0	3436.25	482.25
SASS-5-vl	99.98	93.91	69.04	100	92.36	92.12	76.62	86.08	91.54	88.86
	754.5	224.5	121.25	1636.5	9070.25	585.5	207.25	1254.0	7588.25	817.5

Table 6: SASS-3, SASS-4, SASS-5 attacks against target models Inceptionv3, ResNet-50, DeiT-s, DeiT-b, Deit-B, Vit-T, ViT-B, Vgg16, Vgg19 and ResNet-152 with different thresholds: very small (vs), small (s), large (l), very large (vl) correspond to 0.8, 8, 20, 40 respectively. SASS-3 has inception_pgd, deits_pgd and vit_pgd as attack strategies, SASS-4 has resnet_pgd in addition to SASS-3 and SASS-5 has vgg16_pgd in addition to SASS-4.