Machine Learning (2020) 109:2349–2368
https://doi.org/10.1007/s10994-020-05916-1

Spanning attack: reinforce black-box attacks with unlabeled data

Lu Wang[1,2] · Huan Zhang[3] · Jinfeng Yi[2] · Cho-Jui Hsieh[3] · Yuan Jiang[1]

Received: 16 April 2020 / Revised: 1 August 2020 / Accepted: 19 September 2020 /
Published online: 29 October 2020
© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2020

Abstract
Adversarial black-box attacks aim to craft adversarial perturbations by querying input–output pairs of machine learning models. They are widely used to evaluate the robustness of pre-trained models. However, black-box attacks often suffer from the issue of query inefficiency due to the high dimensionality of the input space, and therefore incur a false sense of model robustness. In this paper, we relax the conditions of the black-box threat model, and propose a novel technique called the *spanning attack*. By constraining adversarial perturbations in a low-dimensional subspace via *spanning* an auxiliary unlabeled dataset, the spanning attack significantly improves the query efficiency of a wide variety of existing black-box attacks. Extensive experiments show that the proposed method works favorably in both soft-label and hard-label black-box attacks.

Keywords Adversarial machine learning · Adversarial robustness · Black-box attacks · Query efficiency

Editors: Kee-Eung Kim, Vineeth N Balasubramanian.

✉ Yuan Jiang
jiangy@lamda.nju.edu.cn

Lu Wang
wangl@lamda.nju.edu.cn

Huan Zhang
huanzhang@ucla.edu

Jinfeng Yi
yijinfeng@jd.com

Cho-Jui Hsieh
chohsieh@cs.ucla.edu

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

[2] JD AI Research, JD.com, Beijing 100020, China

[3] Department of Computer Science, University of California, Los Angeles, Los Angeles, CA 90095, USA

1 Introduction

It has been shown that machine learning models, especially deep neural networks, are vulnerable to small adversarial perturbations, i.e., a small carefully crafted perturbation added to the input may significantly change the prediction results (Szegedy et al. 2014; Goodfellow et al. 2015; Biggio and Roli 2018; Fawzi et al. 2018). Consequently, the problem of finding those perturbations, also known as *adversarial attacks*, has become an important way to evaluate model robustness: the more difficult to attack a model, the more robust the model is.

Depending on the type of information available to the adversary, adversarial attacks can be categorized into *white-box attacks* and *black-box attacks*. In the white-box setting, the *target model* (the model to attack) is completely exposed to the attacker, and adversarial perturbations could be crafted by exploiting the first-order information (or any higher order information), i.e., gradients with respect to the input (Carlini and Wagner 2017; Madry et al. 2018). Despite its efficiency and effectiveness, the white-box setting often stands for an overly strong and pessimistic *threat model*, and white-box attacks are usually not practical when attacking real-world machine learning systems due to the fact that their gradient information is often invisible to the attacker.

In this paper, we focus on the problem of *black-box* attacks: the case where the model structure and parameters (weights) are not available to the attacker (Chen et al. 2017). The attacker can only gather necessary information by means of (iteratively) making input queries to the model and obtaining the corresponding outputs. The black-box setting is a more realistic threat model, and furthermore, crucial in the sense that they could serve as a general way to evaluate the robustness of machine learning models beyond neural networks, even when the model is not differentiable (e.g., evaluating the robustness of tree-based models (Chen et al. 2019) and nearest neighbor models (Wang et al. 2019, 2020)).

Black-box attacks have been extensively studied in the past few years. Depending on what kind of outputs the attacker could derive, black-box attacks could be broadly grouped into two categories: *soft-label attacks* (Chen et al. 2017) and *hard-label attacks* (Brendel et al. 2018). Soft-label attacks assume that the attacker has access to real-valued scores (logits or probabilities) for all labels, while hard-label attacks assume that the attacker only has access to the final discrete decision (the predicted label). However, black-box attacks, especially hard-label attacks, usually require a large number of (typically $> 10K$) queries for each adversarial perturbation. High query complexity limits the scope of application of black-box attacks, and also incurs a false sense of model robustness.

We notice that the convergence rates of the zeroth-order optimization methods used for black-box attacks are shown to be proportional to the dimensionality of the input space (Nesterov and Spokoiny 2017; Wang et al. 2017; Tu et al. 2019). As a consequence, we have a natural conjecture: the query complexity of black-box attacks is also dependent on the dimensionality of the input space, and thus reducing its dimensionality *in a certain delicate way* can enhance the query efficiency of black-box attacks.

Based on the idea above, in this paper we propose a method—the *spanning attack*—to constrain the search space of black-box attacks for the purpose of tackling the inefficiency issue. The spanning attack is motivated by our theoretical analysis that *minimum adversarial perturbations* for a variety of machine learning models prove to be in the *subspace* of the training data. Specifically, we relax the conditions of the black-box threat model by additionally assuming that a small *auxiliary unlabeled dataset* is available to the attacker. The assumption is reasonable: imagine that before attacking an image classification model,

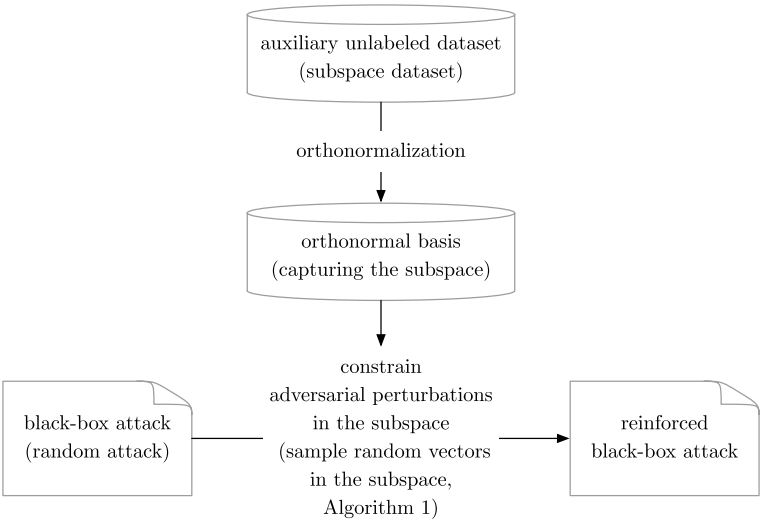


Fig. 1 Workflow of the spanning attack

the attacker just needs to "collect" some unlabeled images, from the Internet for instance. This auxiliary unlabeled dataset plays as a substitute for the original training data: this dataset *spans* a subspace of the input space. Then, we constrain the search of adversarial perturbations only in this subspace, of which the dimensionality is much smaller than the one of the original input space. The overall workflow of the spanning attack is illustrated in Fig. 1.

We also show that the spanning attack method is general enough to apply to a wide range of existing black-box attack methods, including both soft-label attacks and hard-label attacks. Our experiments verify that the spanning attack could significantly improve query efficiency of black-box attacks. Furthermore, we show that even a very small and *biased* unlabeled dataset sampled from a distribution different from the training data suffices to perform favorably in practice. This finding suggests that the assumption of the spanning attack (about the auxiliary unlabeled dataset) is not too strict to be satisfied.

In summary, this paper makes the following contributions:

1. We present the *random attack* framework which captures most existing black-box attacks in various settings including both soft-label attacks and hard-label attacks. It is a novel and intuitive interpretation on the mechanism of black-box attacks from the perspective of random vectors.
2. We propose a method to regulate the resulting adversarial perturbation of any random attack to be constrained in a predefined subspace. This is a general method to reduce the dimensionality of the search space of black-box attacks.
3. We make preliminary theoretical analysis about the subspace in which the *minimum adversarial perturbation* is guaranteed to be placed. Motivated by our analysis, we propose to reinforce black-box attacks (random attacks) by means of constraining a subspace spanned by an auxiliary unlabeled dataset. In our experiments across various black-box attacks and target models, the reinforced attack typically requires less than 50% queries while improves success rates in the meantime.

The remainder of the paper is organized as follows: Sect. 2 discusses related work about black-box attacks; Sect. 3 introduces the basic preliminaries and our motivation; Sect. 4 presents our framework for black-box attacks and proposes our general method to improve query efficiency; Sect. 5 reports empirical evaluation results; Sect. 6 concludes this paper.

2 Related work

Transfer-based black-box attacks The first practical black-box attack is the transfer-based attack (Papernot et al. 2017). A substitute model is trained with synthetic instances labeled by the target model (solf labels or hard labels). Then, the adversarial perturbation is crafted to fool the target model by attacking the substitute model. The effectiveness highly depends on transferability of adversarial perturbations (Papernot et al. 2016; Liu et al. 2017). Accordingly, the attack performance is severely degraded with poor transferability (Su et al. 2018). Therefore, we mainly talk about black-box attacks based on zeroth-order optimization as below.

Soft-label black-box attacks Chen et al. (2017) showed that soft-label black-box attacks can be formulated as solving an optimization problem in the zeroth-order scenario, in which one can query the function itself but not its gradients. Since then, many black-box attack methods based on zeroth-order optimization have been proposed such as ZO-Adam (Chen et al. 2017), NES (Ilyas et al. 2018), ZO-SignSGD (Liu et al. 2019), AutoZOOM (Tu et al. 2019), and Bandit-attack (Ilyas et al. 2019).

Hard-label black-box attacks Hard-label black-box attacks are more challenging since it is non-trivial to define a smooth objective function for attacks based only on the hard-label decisions. Brendel et al. (2018) proposed a method based on reject sampling and random walks. Cheng et al. (2019) reformulated the attack as a real-valued optimization problem and the objective function is estimated via coarse-grained search and then binary search. Chen et al. (2019) proposed an unbiased estimator of the gradient direction at the decision boundary, and presented an attack method with a convergence analysis. Cheng et al. (2020) proposed a query-efficient sign estimator of the gradient.

Improve query efficiency of black-box attacks Recently, the idea of relaxing the threat model to improve query efficiency of black-box attacks has attracted increasing attention. Some work captured the idea of transfer-based attacks (Papernot et al. 2016; Liu et al. 2017): adversarial examples of a surrogate model also tend to fool other models. Brunner et al. (2018) and Cheng et al. (2019) both assumed that a *surrogate model* is available to the attacker. Therefore, the attacker could employ the gradients of the surrogate model as a prior for the true gradient of the target model. Another work (Yan et al. 2019) proposed a soft-label black-box attack method which employs an auxiliary *labeled* datasets. Multiple *reference models* are trained with the *labeled* datasets, and a subspace is spanned by perturbed gradients of these reference models. Then the true gradients of the target model are estimated in the subspace. The major difference from our work is that their auxiliary dataset has to be labeled, whereas ours is unlabeled. Moreover, their auxiliary dataset is much larger than ours owing to the need for training reference models: in the ImageNet case, we only need less than 1000 unlabeled instances, whereas Yan et al. (2019) require 75,000 labeled instances. Finally, our method is more general, and can be applied to both soft-label and hard-label black-box attacks.

3 Background and motivation

We introduce the notations regarding black-box adversarial attacks. Let $\mathbb{X} = \mathbb{R}^D$ denote the input space where $D \in \mathbb{N}^+$ is the number of dimensions, and let $\mathbb{Y} = [C]$ denote the output space where $C \in \mathbb{N}^+$ is the number of labels. The function $c : \mathbb{X} \to \mathbb{Y}$ is a classifier (the target model) and makes decisions by

$$c(\boldsymbol{x}) = \underset{i \in [C]}{\arg\max}\ f(\boldsymbol{x})_i,$$

where $f : \mathbb{X} \to \mathbb{R}^C$ is the score function of the classifier, which outputs scores of all labels for any given input.

Given a radius $\epsilon > 0$ and a correctly-classified labeled instance $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$, the *untargeted attack* aims to find an *adversarial perturbation* $\boldsymbol{\delta} \in \mathbb{X}$ with the norm $\|\boldsymbol{\delta}\| \leq \epsilon$ such that the classifier predicts a label for the perturbed instance $\boldsymbol{x} + \boldsymbol{\delta}$ different from the original instance $\boldsymbol{x}$, i.e., $c(\boldsymbol{x} + \boldsymbol{\delta}) \neq y$. In comparison, the *targeted attack* aims to make the classifier predict a pre-specified label. Our paper will focus on untargeted attacks, while it is easy to extend to targeted attacks. Besides, we focus on the ℓ_2 norm (the Euclidean norm) perturbation: the magnitude of adversarial perturbations are measured by the ℓ_2 norm, and further research on general norms are deferred for future work.

In the soft-label setting, the attacker has access to the score (logit or probability) output for any input $\boldsymbol{x}$ in $\mathbb{X}$, i.e., $f(\boldsymbol{x})$. Therefore, any loss function defined on the the pair of the score and the ground-truth label is also available to the attacker. We denote the loss function as $\ell_f(\boldsymbol{x}, y)$. In contrast, in the hard-label setting, the attacker only has access to the final decision (the predicted label) for any input $\boldsymbol{x}$ in $\mathbb{X}$, i.e., $c(\boldsymbol{x})$. It is more challenging than soft-label attacks due to less information available. The number of queries, to $f(\cdot)$ or $c(\cdot)$, is the cost of black-box attacks. It is crucial to reduce the number of queries required when applying attack methods in real applications.

In practice, the input space $\mathbb{X}$ is usually high-dimensional: for instance, the typical input image for an ImageNet model has $224 \times 224 \times 3 = 150,528$ pixels. It is suspected that requiring such a large amount of queries, often $> 10K$, when searching for an adversarial perturbation $\boldsymbol{\delta}$ in $\mathbb{X}$ is probably owing to the high dimensionality of $\mathbb{X}$. To verify our conjecture, a natural question for black-box attacks is as below:

> "Is it possible to reduce the number of queries for general black-box attacks by reducing the dimensionality of the search space?"

In this paper, we provide a positive answer to this question by proposing a method reinforcing black-box attacks with a small set of unlabeled data.

4 Proposed method

We first introduce the technique on constraining (transforming) adversarial perturbations into a predefined subspace for general black-box attacks, and then propose a method which utilizes an auxiliary unlabeled dataset to select an appropriate subspace.

4.1 Subspace transformation

Definition 1 (subspace attack) A subspace attack is an adversarial attack which returns adversarial perturbations in a predefined subspace $\mathbb{V} \subseteq \mathbb{X}$.

Intuitively, the predefined subspace $\mathbb{V}$ can be seen as a prior for perturbations of adversarial examples. If the subspace is small enough while still being able to capture most of small adversarial perturbations, then due to the reduced dimensionality, it can significantly reduce the number of queries required for black-box attacks.

Algorithm 1: Random vectors in a subspace

Input: orthonormal vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M \in \mathbb{X}$ which spans the subspace $\mathbb{V}$, and the sampling routine `sample` for isometric random vectors
Output: a random vector in the subspace $\mathbb{V}$
1 $\boldsymbol{w} \leftarrow \texttt{sample}(M)$
2 **return** $\sqrt{\frac{D}{M}} \sum_{i=1}^{M} w_i \boldsymbol{e}_i$

We will focus on "one type" of black-box attacks, the *random attack*, which captures a wide range of (nearly all existing) black-box attacks, and is convenient to incorporate the prior knowledge about the subspace, and thus easy to be transformed into a subspace attack. Examples of random attacks will be shown in Sects. 4.1.1 and 4.1.2

Definition 2 (Random attack) The resulting adversarial perturbation of a random attack is a linear combination of random vectors.

The following lemma highlights an intuition on how to transform a random attack into a subspace attack:

Lemma 1 *If all random vectors sampled by a random attack is constrained to be in a predefined subspace $\mathbb{V}$, then the random attack is a subspace attack with respect to $\mathbb{V}$.*

The proof is straightforward: a linear combination of vectors in a subspace is also in the subspace.

Random vectors of random attacks are typically sampled from isometric distributions: all elements of the random vector are independent and identically distributed. Typical examples of these distributions include the isometric Gaussian distribution and the Rademacher distribution (uniform over $\{\pm 1\}$). Let $\texttt{sample}(d)$ denote the sampling routine for such a random vector with the dimension $d \in \mathbb{N}^+$. (Thus $\texttt{sample}(D)$ will sample a random vector in the original input space $\mathbb{X}$.) It follows that if we could constrain the sampling routine in a subspace, by Lemma 1 the resulting attack would be a subspace attack. Specifically, Algorithm 1 displays how to sample a random vector in a subspace. The subspace $\mathbb{V}$ is characterized by an orthonormal basis (see Sect. 4.2 for details on how to derive the orthonormal basis), and the term $\sqrt{\frac{D}{M}}$ guarantees that the returned random vector has the same expected length as the original random vector $\texttt{sample}(\mathtt{D})$.

Note that the returned random vector of Algorithm 1 is a linear combination of the orthonormal vectors. Therefore we have the following lemma:

Lemma 2 *The returned random vector of Algorithm 1 is constrained in the subspace* $\mathbb{V} = \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_M)$*, where* $\text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_M)$ *returns the smallest space* $\mathbb{V}$ *that contains all the input vectors* $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_M$.

Therefore, by applying Algorithm 1 to any random attack, we have a subspace attack as the following corollary implies:

Corollary 1 *Given a set of orthonormal vectors* $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_M \in \mathbb{X}$*, any random attack using isometric random vectors can be transformed into a subspace attack with the corresponding subspace* $\mathbb{V} = \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_M)$ *by means of replacing the sampling routine* `sample`*(D) via Algorithm 1.*

Corollary 1 introduces a particular method to transform a random attack (black-box attack) into a subspace attack. It is noteworthy that the transformation is performed only by means of replacing sampling routines. It *does not require to project adversarial perturbations from the input space* $\mathbb{X}$ *to the subspace* $\mathbb{V}$ *explicitly*, and therefore causes as little negative impact as possible on the original random attack.

4.1.1 Case study: soft-label black-box attacks

We investigate a soft-label black-box attack framework within which the attack is composed of a gradient-based optimization method and a *backend* zeroth-order gradient estimation method. This framework is summarized in Algorithm 2, and captures a wide range of soft-label black-box methods (Ilyas et al. 2018; Liu et al. 2019; Uesato et al. 2018; Tu et al. 2019; Cheng et al. 2019).

Algorithm 2: Soft-label black-box attack framework

Input: score function $f : \mathbb{X} \to \mathbb{R}^C$, and corresponding classifier $c : \mathbb{X} \to \mathbb{Y}$, correctly-classified labeled instance $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$, and budget $B \in \mathbb{N}^+$
Output: adversarial perturbation $\boldsymbol{\delta}$ or NULL

$\boldsymbol{\delta} \leftarrow$ `initializer()`
while $B > 0$ **do**
  if $c(\boldsymbol{x} + \boldsymbol{\delta}) \neq y$ **then**
    return $\boldsymbol{\delta}$ `// successful`
  end
  $\boldsymbol{g} \leftarrow$ `gradient_estimator`$(f, \boldsymbol{x} + \boldsymbol{\delta}, y)$
  $\boldsymbol{\delta} \leftarrow$ `gradient_based_optimizer`$(\boldsymbol{\delta}, \boldsymbol{g})$
  $B \leftarrow B -$ `query_sum_within_the_iteration`$()$
end
return *NULL* `// failed`

In this framework, random vectors could be introduced when initializing the perturbation (the all-zero vector or a random vector) and estimating gradients by the zeroth-order method. A typical example of estimating gradients is the random gradient-free (RGF) method (Nesterov and Spokoiny 2017), which returns the estimated gradient in the form below:

$$\hat{\boldsymbol{g}} = \sum_i \frac{\ell_f(\boldsymbol{x} + \sigma \boldsymbol{u}_i, y) - \ell_f(\boldsymbol{x}, y)}{\sigma} \boldsymbol{u}_i,$$

where $\boldsymbol{u}_i$s are *unit* Gaussian random vectors (Gaussian random vectors of length 1). Therefore, $\hat{\boldsymbol{g}}$ is a linear combination of random vectors.

Then, the resulting adversarial perturbation is calculated by gradient-based optimization methods such as projected gradient descent (Madry et al. 2018), all of which return linear combinations of the estimated gradients. It follows that these attacks are random attacks and could be easily transformed into a subspace attack via Algorithm 1.

4.1.2 Case study: hard-label black-box attacks

Hard-label black-box attacks could be separated into two categories: methods based on random walks (Brendel et al. 2018; Chen et al. 2019) and methods based on direction estimation (Cheng et al. 2019, 2020). In the first case, a random walk consists of a succession of random vectors, i.e., the sum of random vectors; in the second case, the gradient with respect to the direction towards the boundary is estimated by RGF or its variant based on the sign of the finite difference. As we discuss before, these gradient estimation methods typically return linear combinations of random vectors. In both cases, the resulting adversarial perturbation is also a linear combination of random vectors, and as a consequence they could also be transformed into subspace attacks obviously.

4.2 Spanning attack

The subspace $\mathbb{V}$ is a prior for the subspace attack. To make a subspace attack perform well, it has to be easier to find an adversarial perturbation in the subspace $\mathbb{V}$ than in the original input space $\mathbb{X}$. The crux of the subspace attack is how to locate an appropriate subspace $\mathbb{V}$. We propose to utilize an auxiliary unlabeled dataset to span the subspace, which is motivated by the theoretical analysis regarding the *minimum adversarial perturbation* as below.

The minimum adversarial perturbation is the adversarial perturbation with the minimum norm. Formally, given a classifier $c : \mathbb{X} \to \mathbb{Y}$ and a labeled instance $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$, the minimum adversarial perturbation is defined as

$$\boldsymbol{\delta}^* = \arg\min_{\boldsymbol{\delta}} \ \|\boldsymbol{\delta}\| \ \text{ s.t. } c(\boldsymbol{x} + \boldsymbol{\delta}) \neq y.$$

Let $\mathbb{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ be the training dataset, and $\mathbb{D}_{\mathbb{X}} = \{\boldsymbol{x}_i\}_{i=1}^N$ be the training instances without labels. We have the following theorem on the minimum adversarial perturbation of the K-nearest neighbor classifier (K-NN):

Theorem 1 *For every $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$, there exists $\boldsymbol{w} \in \mathbb{R}^N$ such that the minimum adversarial perturbation of K-NN satisfies*

$$\boldsymbol{\delta}^* = \sum_{i=1}^N w_i \boldsymbol{x}_i.$$

In other words, the minimum adversarial perturbation of K-NN is in the subspace $\text{span}(\mathbb{D}_{\mathbb{X}})$.

Proof Given $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$ and $\mathbb{T} \subseteq \mathbb{D}_{\mathbb{X}}$ with $|\mathbb{T}| = K$, consider to add a small perturbation $\boldsymbol{\delta} \in \mathbb{X}$ such that $\mathbb{T}$ is the K nearest neighbors of $\boldsymbol{x} + \boldsymbol{\delta}$. This problems could be formalized as the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\| \\ &\text{s.t.} \|\boldsymbol{x} + \boldsymbol{\delta} - \boldsymbol{x}^+\| \leq \|\boldsymbol{x} + \boldsymbol{\delta} - \boldsymbol{x}^-\| \\ &\quad \forall \boldsymbol{x}^+ \in \mathbb{T},\ \forall \boldsymbol{x}^- \in \mathbb{D}_{\mathbb{X}} - \mathbb{T}. \end{aligned}$$

It is equivalent to the following problem:

$$\begin{aligned} &\min_{\boldsymbol{\delta}} \frac{1}{2} \boldsymbol{\delta}^\top \boldsymbol{\delta} \\ &\text{s.t.} (\boldsymbol{x}^- - \boldsymbol{x}^+)^\top \boldsymbol{\delta} \leq \frac{1}{2} (\|\boldsymbol{x} - \boldsymbol{x}^-\|^2 - \|\boldsymbol{x} - \boldsymbol{x}^+\|^2) \\ &\quad \forall \boldsymbol{x}^+ \in \mathbb{T},\ \forall \boldsymbol{x}^- \in \mathbb{D}_{\mathbb{X}} - \mathbb{T}. \end{aligned}$$

The constraints could be rewritten in the matrix form:

$$\begin{aligned} &\min_{\boldsymbol{\delta}} \frac{1}{2} \boldsymbol{\delta}^\top \boldsymbol{\delta} \\ &\text{s.t.} \boldsymbol{A}\boldsymbol{\delta} \leq \boldsymbol{b}. \end{aligned}$$

Obviously, it is a convex quadratic programming problem. Let $\boldsymbol{\delta}^*_{\mathbb{T}}$ and $\boldsymbol{\lambda}^*_{\mathbb{T}}$ be the optimal points of the primal problem and the dual problem respectively. By the primal-dual relationship, we have

$$\boldsymbol{\delta}^*_{\mathbb{T}} = -\boldsymbol{A}^\top \boldsymbol{\lambda}^*_{\mathbb{T}}.$$

Considering the form of $\boldsymbol{A}$, it is obvious that $\boldsymbol{\delta}^*_{\mathbb{T}}$ is a linear combination of instances in $\mathbb{D}_{\mathbb{X}}$.

Note that the minimum adversarial perturbation $\boldsymbol{\delta}^*$ of K-NN has to be $\boldsymbol{\delta}^*_{\mathbb{T}}$ for a certain $\mathbb{T}$. Therefore, $\boldsymbol{\delta}^*$ has to be in the subspace span($\mathbb{D}_{\mathbb{X}}$). □

Similar results on the minimum adversarial perturbation also hold for support vector machine (SVM) classifiers (Cortes and Vapnik 1995) as follows:

Theorem 2 *For every $(\boldsymbol{x}, y) \in \mathbb{X} \times \mathbb{Y}$, there exists $\boldsymbol{w} \in \mathbb{R}^N$ such that the minimum adversarial perturbation of SVM satisfies*

$$\boldsymbol{\delta}^* = \sum_{i=1}^{N} w_i \boldsymbol{x}_i.$$

In other words, the minimum adversarial perturbation of SVM is also in the subspace span($\mathbb{D}_{\mathbb{X}}$).

Proof For simplicity, we only consider the binary case, which can be easily extended to the multi-class case by strategies such as one-vs-one and one-vs-rest. Let $\boldsymbol{w}^*$ be the optimal solution of SVM. Based on the primal-dual relationship, we have

$$\boldsymbol{w}^* = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i$$

for some $\boldsymbol{\alpha} \in \mathbb{R}^N$. When predicting a perturbed instance, SVM calculates

$$\begin{aligned}\langle \boldsymbol{w}, \boldsymbol{x} + \boldsymbol{\delta} \rangle &= \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \langle \boldsymbol{w}, \boldsymbol{\delta} \rangle \\ &= \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \|\boldsymbol{w}\| \|\boldsymbol{\delta}\| \cos(\theta),\end{aligned}$$

where θ is the angle between $\boldsymbol{w}$ and $\boldsymbol{\delta}$. Therefore, the minimum adversarial perturbation that flips the sign of $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ has to be in the direction of $\boldsymbol{w}$ with $\cos(\theta) = 1$ or in the opposite direction of $\boldsymbol{w}$ with $\cos(\theta) = -1$. □

It is inspiring that K-NN and SVM are very different whereas share the same property:

"Minimum adversarial perturbations prove to be in the subspace spanned by the training data."

Consequently, Theorems 1 and 2 motivate us to *search for adversarial perturbations in the space* span($\mathbb{D}_{\mathbb{X}}$), which is the theoretical foundation of our spanning attack. Nevertheless, before diving into details of the spanning attack, it is worth mentioning that computing minimum adversarial perturbations for neural networks and tree-based ensemble models has shown to be NP-hard (Katz et al. 2017; Kantchelian et al. 2016), and it is an open problem in what conditions minimum adversarial perturbations for neural networks and tree-based ensemble models are also in the space span($\mathbb{D}_{\mathbb{X}}$).

In practice, it is not reasonable to assume the training data are available to attackers. To make a relaxation, we assume that the attacker only has access to an auxiliary unlabeled dataset $\mathbb{S}$. By this means, subspace attackers search for adversarial perturbations in span($\mathbb{S}$), namely the ***spanning attack***, i.e., *the subspace attack by spanning an auxiliary unlabeled dataset*. For convenience, we simply term the auxiliary unlabeled dataset as the *subspace dataset*.

By Corollary 1, given a subspace dataset $\mathbb{S}$, the spanning attack requires a set of orthonormal vectors which is a basis for span($\mathbb{S}$) so as to transform a random attack into a subspace attack. We could make it by the standard process of *orthonormalization*, which can be performed by the Gram-Schmidt process, the Householder transformation etc (Cheney and Kincaid 2010). Therefore, the overall procedures of our spanning attack is as below (also shown in Fig. 1):

1. Compute a basis of $\mathbb{S}$ by orthonormalization;
2. Transform the random attack into a subspace attack by Algorithm 1;
3. Attack the target model with the resulting subspace attack.

4.3 Selective spanning attack

In this section, we talk about an extension of the spanning attack. The spanning attack searches for adversarial perturbations in the space span($\mathbb{S}$), which is a subspace of the input space $\mathbb{X}$. A natural question is *whether it is possible to benefit more by means of explicitly selecting a subspace of* span($\mathbb{S}$) *instead of using* span($\mathbb{S}$) *directly*. We term the method which searches for adversarial perturbations in a non-trivial subspace of span($\mathbb{S}$) as the *selective spanning attack*, as it selects a subspace from span($\mathbb{S}$).

In the case of the selective spanning attack, the Gram-Schmidt process or Householder transformation is not instructive to select a subspace of span(𝕊), since there is no significant difference among the derived orthonormal vectors.

Instead, we employ the singular value decomposition (SVD) to derive a set of orthonormal vectors which is a basis of span(𝕊). In particular, assume the subspace dataset has M' different instances $\mathbb{S} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{M'}\}$ and $\boldsymbol{S} \in \mathbb{R}^{M' \times D}$ is the matrix of which the j-th row is $\boldsymbol{x}_j^\intercal$. (We use M' here because M denotes the number of orthonormal vectors as Algorithm 1.) By SVD, $\boldsymbol{S}$ can be decomposed into the form

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\intercal,$$

where $\boldsymbol{U} \in \mathbb{R}^{M' \times M'}$ and $\boldsymbol{V} \in \mathbb{R}^{D \times D}$ are orthogonal matrices and $\boldsymbol{\Sigma} \in \mathbb{R}^{M' \times D}$ is a diagonal matrix, of which diagonal entries are singular values.

It could be proved that the right singular vectors (columns of $\boldsymbol{V}$) satisfy the following property:

Lemma 3 *Right singular vectors of which the corresponding singular values are larger than zero are an orthonormal basis for* span(𝕊).

Proof Let M denote the number of non-zero singular values. Then, we have the compact SVD as

$$\boldsymbol{S} = \boldsymbol{U}_M \boldsymbol{\Sigma}_M \boldsymbol{V}_M^\intercal.$$

Let $\boldsymbol{e}_i$ denote the i-th column of $\boldsymbol{V}_M$. The objective is to prove

$$\text{span}(\mathbb{S}) = \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M),$$

which is equivalent to

(i) $\text{span}(\mathbb{S}) \subseteq \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M)$ and
(ii) $\text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M) \subseteq \text{span}(\mathbb{S})$.

For any $\boldsymbol{a} \in \text{span}(\mathbb{S})$, by definition there exists $\boldsymbol{b} \in \mathbb{R}^{M'}$ such that

$$\boldsymbol{a} = \boldsymbol{b}^\intercal \boldsymbol{S} = (\boldsymbol{b}^\intercal \boldsymbol{U}_M \boldsymbol{\Sigma}_M) \boldsymbol{V}_M^\intercal.$$

Therefore, we have (i) $\text{span}(\mathbb{S}) \subseteq \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M)$.

By the compact SVD, we also have

$$\boldsymbol{\Sigma}_M^{-1} \boldsymbol{U}_M^\intercal \boldsymbol{S} = \boldsymbol{V}_M^\intercal.$$

Thus, for any $\boldsymbol{a} \in \text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M)$, there exists $\boldsymbol{b} \in \mathbb{R}^M$ such that

$$\boldsymbol{a} = \boldsymbol{b}^\intercal \boldsymbol{V}_M^\intercal = (\boldsymbol{b}^\intercal \boldsymbol{\Sigma}_M^{-1} \boldsymbol{U}_M^\intercal) \boldsymbol{S}.$$

Therefor, we have (ii) $\text{span}(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M) \subseteq \text{span}(\mathbb{S})$. □

We denote these right singular vectors as $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_M$ with corresponding singular values larger than zero, and they are sorted according to the corresponding singular values such that the singular values have $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_M$. Then, we have roughly two options

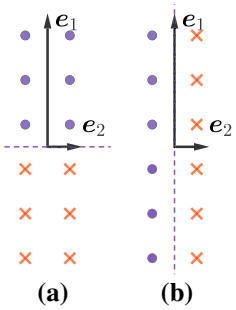


Fig. 2 Illustration of top and bottom spanning attacks. The only difference between (**a**) and (**b**) is the ground-truth labels. $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are right singular vectors, and their lengths represent the corresponding singular values. The top spanning attack, i.e., selecting span($\boldsymbol{e}_1$), is better than the bottom spanning attack, i.e., selecting span($\boldsymbol{e}_2$), in (**a**); whereas (**b**) is the opposite case

for the selective spanning attack: selecting the top singular vectors, and selecting the bottom singular vectors. We term these two options as the *top spanning attack* and the *bottom spanning attack* respectively.

The top spanning attack and the bottom spanning attack have their own advantageous situations depending on the labels of the underlying data distribution. We illustrate two toy cases in Fig. 2. In the first case, the top spanning attack is favorable since we can find adversarial perturbations along $\boldsymbol{e}_1$, and the second case is the exact opposite. Roughly speaking, top singular vectors represent directions along the manifold of the dataset, and bottom singular vectors represent directions out of the manifold. It is believed that for high-dimensional datasets adversarial examples widely exist in directions out of the manifold (Stutz et al. 2019). Therefore, the bottom spanning attack would be a better choice in practice, which is validated in our experiments.

5 Experiments

In this section, we empirically validate the performance of the proposed spanning attack.[1] Specifically, we select three representative black-box (random) attacks as baselines and employ the spanning attack method to reinforce them:

- The RGF attack (Cheng et al. 2019): a soft-label attack with Gaussian random vectors within the framework considered in the case study for soft-label attacks;
- The SPSA attack (Uesato et al. 2018): a soft-label attack similar to the RGF attack but with Rademacher random vectors instead of Gaussian random vectors; (In our implementation, SPSA has all hyper-parameters the same as RGF except the distribution for random vectors.)
- The boundary attack (Brendel et al. 2018): a pioneering widely-used hard-label attack based on random walks;
- The Sign-OPT attack (Cheng et al. 2020): a state-of-the-art hard-label attack based on direction estimation.

We perform untargeted black-box attacks on the ImageNet dataset (Deng et al. 2009). Attacks are performed against the pre-trained ResNet-50 (He et al. 2016),

[1] Our code is available at https://github.com/wangwllu/spanning_attack.

VGG-16 (Simonyan and Zisserman 2015) and DenseNet-121 (Huang et al. 2017) from the PyTorch model zoo (Steiner et al. 2019), since these architectures are diverse and representative, and many real-world deployed models are based on them. Correctly-classified images are randomly sampled from the validation set as the evaluation dataset, of which every labeled image is the instance to attack. The size of the evaluation dataset for soft-label attacks is 1000, and the one for hard-label attacks is 100 for computational efficiency. Another 1000 unlabeled images are sampled from the validation set as the subspace dataset. The evaluation dataset and the subspace dataset are mutually exclusive. We set the perturbation radius $\epsilon = \sqrt{0.001D}$ and the budget $B = 10,000$ by convention (Cheng et al. 2019). (We also study whether the radius parameter ϵ impacts performance in Sect. 5.3.) If an attack method finds an adversarial perturbation $\boldsymbol{\delta}$ within B queries such that $\|\boldsymbol{\delta}\| \leq \epsilon$ holds, then this attack is *successful*; otherwise it is failed. Therefore, we have two criteria for a black-box attack: (i) whether it is successful and (ii) the number of queries it executes when successful.

All hyper-parameters of the spanning attacks are the same as the corresponding baselines. The only difference is introducing an appropriate subspace via our methods. We refer to Cheng et al. (2019), Uesato et al. (2018), Brendel et al. (2018) and Cheng et al. (2020) for details of the baseline black-box methods.

5.1 Main results

Success rates, query means and query medians on the evaluation dataset are reported in Table 1. By convention only successful adversarial perturbations are counted for query means and query medians. On the one hand, this criterion is favorable for the method with a lower success rate and a lower query number on successful adversarial perturbations. On the other hand and more importantly, if a method has a higher success rate and a lower query number on successful perturbations than the other, we would have sufficient confidence to conclude that the first method performs better.

Our results show that the spanning attack method reinforces the baselines significantly in terms of both success rates and query numbers, consistently across *all* of the baseline methods and *all* of the pre-trained target models.

In particular, in the case of the RGF attack and the Sign-OPT attack, the spanning attack only needs approximately *half* the queries of the baseline for a successful attack, and increases success rates in the meantime. For example, in the Sign-OPT case against ResNet-50, the spanning attack improves the success rate to 100%, and more crucially, it only requires 44% queries in terms of the query mean and 30% queries in terms of the query median!

In the case of the boundary attack, while success rates of the baseline attack within the given budget is not satisfying, our spanning attack improves the success rates by a wide margin. For example, in the Boundary case against VGG-16, the spanning attack improves the success rate from 81% to 94%.

Visualization of the subspace basis A sample of vectors of the resulting orthonormal basis are visualized in Fig. 3. Note that these vectors reflect low-dimensional structures of the subspace rather than white Gaussian noise in the input space.

Examples of adversarial images Several adversarial images, crafted by the baseline method and the spanning attack, are displayed in Fig. 4. All these adversarial images does not show any significant difference from the original images due to the fact that they have the same constraint on the perturbation norm.

Table 1 Comparison between the baseline black-box attacks and the resulting spanning attacks

			Success rate	Query mean	Query median
ResNet-50	RGF (soft-label)	Baseline	0.971	589.575	358.0
		Spanning attack	0.991	329.541	205.0
	SPSA (soft-label)	Baseline	0.972	584.772	358.0
		Spanning attack	0.991	330.725	205.0
	Boundary (hard-label)	Baseline	0.720	4133.903	3291.0
		Spanning attack	0.880	3197.557	2569.5
	Sign-OPT (hard-label)	Baseline	0.970	2392.175	2143.0
		Spanning attack	1.000	1053.220	647.0
VGG-16	RGF (soft-label)	Baseline	0.966	389.519	256.0
		Spanning attack	0.975	261.335	154.0
	SPSA (soft-label)	Baseline	0.968	386.187	256.0
		Spanning attack	0.975	262.905	154.0
	Boundary (hard-label)	Baseline	0.810	3467.086	2787.0
		Spanning attack	0.940	2972.755	2263.0
	Sign-OPT (hard-label)	Baseline	1.000	1665.080	1450.0
		Spanning attack	1.000	840.900	572.5
DenseNet-121	RGF (soft-label)	Baseline	0.981	528.312	358.0
		Spanning attack	0.995	272.043	154.0
	SPSA (soft-label)	Baseline	0.984	552.982	358.0
		Spanning attack	0.997	299.941	154.0
	Boundary (hard-label)	Baseline	0.670	3806.687	3389.0
		Spanning attack	0.890	3063.449	2261.0
	Sign-OPT (hard-label)	Baseline	0.980	2407.398	1863.5
		Spanning attack	1.000	1014.280	688.5

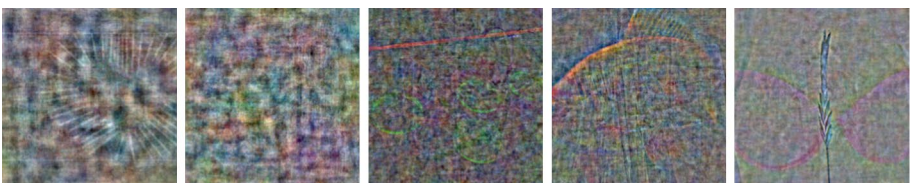
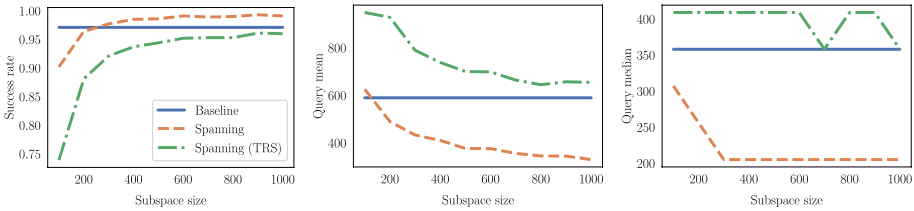


Fig. 3 Visualization for vectors of the orthonormal basis

Comments on improvement of success rates The black-box attack is a non-convex zeroth-order optimization problem; there is always a chance that the baseline method is trapped in some local areas, and as a consequence fails to attack. For instance, when RGF estimates gradients, informative random vectors could be too sparse to find an accurate gradient, as these random vectors are sampled from a large space. In contrast, the spanning attack employs prior knowledge (encoded in the subspace) about adversarial perturbations, and hence reduces the possibility of being trapped. That's why the spanning attack could improve success rates and query efficiency simultaneously. It is

Fig. 4 Examples of the adversarial images. The first row is the original images; the second row is the adversarial images crafted by the baseline attack (Sign-OPT against ResNet-50); the third row is the adversarial images crafted by the corresponding spanning attack



(a) Success rate **(b)** Query mean **(c)** Query median

Fig. 5 Attack performance with different sizes of the subspace. TRS stands for Totally Random Subspace

noteworthy that the capability of the spanning attack depends on the subspace dataset (the prior knowledge encoded); we will carefully investigate it in the next section.

Since the results are consistent across all baseline methods and target models, in the following we take the RGF attack against ResNet-50 as illustration by default to avoid unnecessary repetition.

5.2 Investigation on the subspace

In this section, we study to what extent the subspace impacts on the performance of the spanning attack, and furthermore how we could establish the subspace dataset in practice.

Table 2 Results of the spanning attack with label-biased subspace datasets, spanning attack with Flickr8k subspace datasets, spanning attack with totally random subspace (a subspace without any prior), bottom spanning attack and top spanning attack. TRS stands for Totally Random Subspace

	Success rate	Query mean	Query median
Baseline	0.971	589.575	358.0
Spanning attack	0.991	329.541	205.0
Spanning attack (label biased)	0.991	316.572	205.0
Spanning attack (Flickr8k)	0.990	318.333	205.0
Spanning attack (TRS)	0.960	654.491	358.0
Bottom spanning attack	0.991	298.817	154.0
Top spanning attack	0.991	354.346	205.0

5.2.1 Size of the subspace dataset

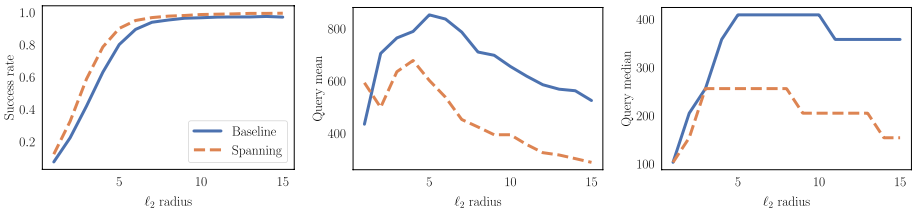
We show attack performance with different sizes of the subspace dataset in Fig. 5. (We will talk about TRS in Sect. 5.2.2.) In our experiments, the minimum size is 100 and the maximum size is 1000. In this scope, the larger the subspace size, the better the performance of the spanning attack. In contrast, the baseline method is the extreme case where the subspace size is $D = 224 \times 224 \times 3 = 150,528$. Therefore, it is expected that the performance of the spanning attack will reach the peak and then slide down as the subspace size increases. It is noteworthy that *even a small subspace dataset,* ≈ 400 *as shown in Fig.* 5, *would help the spanning attack defeat the baseline methods.*

5.2.2 Distribution of the subspace dataset

We investigate whether it is necessary to sample the subspace dataset from the same distribution as the training data. Specifically, we further try three settings for the subspace dataset:

(1) instances of top 50 classes from the ImageNet validation set. Note that there are 1000 classes in total, and hence it is a *label-biased* setting.
(2) instances from the Flickr8k dataset (Hodosh et al. 2015), which is much different from the ImageNet dataset.
(3) instances sampled from a uniform distribution. In other words, the spanned subspace could be seen as a *totally random subspace* (TRS) without any prior knowledge.

The results are displayed in the middle area of Table 2. On the one hand, the results of the label-biased and the Flickr8k spanning attack are still better than the baseline, and competitive with the spanning attack in Sect. 5.1 (see the upper area of Table 2 for convenience), where the subspace dataset is sampled i.i.d. (without any bias) from the ImageNet validation set. It suggests that *even a biased subspace dataset suffices to work*, which extends the application range of the spanning attack. In a word, the subspace dataset does not necessarily has to be sampled from the same distribution with the training data.



(a) Success rate **(b)** Query mean **(c)** Query median

Fig. 6 Attack performance with different radii

On the other hand, the spanning attack with a totally random subspace performs even worse than the baseline. (To better illustrate this issue, we also show attack performance for different subspace sizes with totally random subspaces in Fig. 5.) In other words, the totally random subspace plays a *negative* role on performance. The result validates that *prior knowledge given by the subspace dataset is necessary, rather than an arbitrary low-dimensional subspace.*

Discussion on establishing the subspace dataset in practice The experimental results of Sects. 5.2.1 and 5.2.2 jointly suggest that the conditions which the subspace dataset has to obtain is not too strict in practice: the size of the subspace dataset could be very small, and the distribution of the subspace could be different from the one of the training data. Therefore, when applying the subspace attack method in real-world applications, we only need to collect a *small* set of *unlabeled* data *related* to the target model. For instance, if the task is attacking a face recognition system, one possibility is to crawl the web and find some face pictures in advance.

5.2.3 Bottom and top spanning attack

We investigate whether the selective spanning attack could further improve performance. In our experiments, the bottom 800 singular vectors (remember the total number of the singular vectors is 1000) are used for the bottom spanning attack, and the top 800 singular vectors are used for the top spanning attack. The comparison among the original spanning attack, bottom spanning attack and top spanning attack are shown in the lower area of Table 2. The results show that the bottom spanning attack could further improve performance, whereas the top spanning attack has a negative impact. This is an empirical validation that adversarial perturbations are more likely to appear in directions out of the data manifold, rather than along the data manifold, as discussed in Sect. 4.3.

5.3 Sensitivity to radii

We investigate whether the given radius affects the capability of the spanning attack over the baseline method. We report success rates, query means and query medians with varying radii. The results are illustrated in Fig. 6, and show that the spanning attack improves the baseline method consistently across different radii. Note that in all the other experiments the radius is set $\epsilon = \sqrt{0.001D} \approx 12.27$.

5.4 More discussion with related work

Although the work of Yan et al. (2019) has a setting different from ours as discussed in Sect. 2, for completeness we try to adapt their methods for comparison. They require an auxiliary *labeled* dataset, focus on ℓ_∞ norm and only considers the soft-label black-box attack. In order to have a comparison, we let the subspace dataset be *labeled* with size 1000. We notice that *with such a small subspace dataset* in our setting, Yan et al. (2019)'s method does not perform well. For instance, when attacking ResNet-50, it has the success rate 58.7% and the query mean 641.283. The results for attacking VGG-16 and DenseNet-121 are similar (VGG-16: success rate 68.6%, query mean 558.044; DenseNet-121: success rate 59%, query mean 623.603). It is primarily due to the fact that their method trains substitute models with labeled data. As a consequence, when the dataset is too small, it is difficult to train reliable substitute models.

6 Conclusion

We propose a general technique named the spanning attack to improve efficiency of black-box attacks. The spanning attack is motivated by the theoretical analysis that minimum adversarial perturbations of machine learning models incline to be in the subspace of the training data. In practice, the spanning attack only requires a small auxiliary unlabeled dataset, and is applicable to a wide range of black-box attacks including both the soft-label black-box attacks and hard-label black-box attacks. Our experiments show that the spanning attack can significantly improve the query efficiency and success rates of black-box attacks simultaneously.

Funding This work is jointly supported by NSFC 61673201, NSFC 61921006, NSF IIS-1901527, NSF IIS-2008173 and the program A for Outstanding PhD candidate of Nanjing University.

Availability of data and material Our methods are validated on public datasets and public models.

Code availability Our code is available at https://github.com/wangwllu/spanning_attack.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Biggio, B., & Roli, F. (2018). Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, *84*, 317–331.

Brendel, W., Rauber, J., & Bethge, M. (2018). Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International conference on learning representations (ICLR)*.

Brunner, T., Diehl, F., Truong-Le, M., & Knoll, A. (2018). Guessing smart: Biased sampling for efficient black-box adversarial attacks. In *IEEE international conference on computer vision (ICCV)* (pp. 4958–4966).

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *IEEE symposium on security and privacy (SP)* (pp. 39–57).

Chen, J., Jordan, M. I., & Wainwright, M. J. (2019). Hopskipjumpattack: A query-efficient decision-based attack. CoRR abs/1904.02144.

Chen, H., Zhang, H., Boning, D., & Hsieh, C. J. (2019). Robust decision trees against adversarial examples. In *International conference on machine learning (ICML)* (pp. 1122–1131).

Chen, P. Y., Zhang, H., Sharma, Y., Yi, J., & Hsieh, C. J. (2017). ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *ACM conference on computer and communications security (CCS) workshop on artificial intelligence and security (AISec)* (pp. 15–26).

Cheney, W., & Kincaid, D. R. (2010). *Linear algebra: Theory and applications*. Washington, DC: The Saylor Foundation.

Cheng, S., Dong, Y., Pang, T., Su, H., & Zhu, J. (2019). Improving black-box adversarial attacks with a transfer-based prior. In *Advances in neural information processing systems (NeurIPS)*.

Cheng, M., Le, T., Chen, P. Y., Yi, J., Zhang, H., & Hsieh, C. J. (2019). Query-efficient hard-label black-box attack: An optimization-based approach. In *International conference on learning representations (ICLR)*.

Cheng, M., Singh, S., Chen, P. Y., Liu, S., & Hsieh, C. J. (2020) Sign-opt: A query-efficient hard-label adversarial attack. In *International conference on learning representations (ICLR)*.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009) Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 248–255).

Fawzi, A., Fawzi, O., & Frossard, P. (2018). Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning*, *107*(3), 481–508.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International conference on learning representations (ICLR)*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 770–778).

Hodosh, M., Young, P., & Hockenmaier, J. C. (2015). Framing image description as a ranking task: data, models and evaluation metrics. In *International conference on artificial intelligence (IJCAI)* (pp. 4188–4192).

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2261–2269).

Ilyas, A., Engstrom, L., Athalye, A., & Lin, J. (2018) Black-box adversarial attacks with limited queries and information. In *International conference on machine learning (ICML)* (pp. 2142–2151).

Ilyas, A., Engstrom, L., & Madry, A. (2019) Prior convictions: Black-box adversarial attacks with bandits and priors. In *International conference on learning representations (ICLR)*.

Kantchelian, A., Tygar, J., & Joseph, A. (2016). Evasion and hardening of tree ensemble classifiers. In *International conference on machine learning (ICML)* (pp. 2387–2396).

Katz, G., Barrett, C., Dill, D. L., Julian, K., & Kochenderfer, M. J. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. In *International conference on computer aided verification* (pp. 97–117).

Liu, S., Chen, P. Y., Chen, X., & Hong, M. (2019). signSGD via zeroth-order oracle. In *International conference on learning representations (ICLR)*.

Liu, Y., Chen, X., Liu, C., & Song, D. (2017) Delving into transferable adversarial examples and black-box attacks. In *International conference on learning representations (ICLR)*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018) Towards deep learning models resistant to adversarial attacks. In *International conference on learning representations (ICLR)*.

Nesterov, Y., & Spokoiny, V. G. (2017). Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, *17*(2), 527–566.

Papernot, N., McDaniel, P. D., & Goodfellow, I.J. (2016) Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. CoRR abs/1605.07277.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). Practical black-box attacks against machine learning. In *Asia conference on computer and communications security* (pp. 506–519).

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International conference on learning representations (ICLR)*.

Steiner, B., DeVito, Z., Chintala, S., Gross, S., Paszke, A., Massa, F., Lerer, A., Chanan, G., Lin, Z., Yang, E., Desmaison, A., Tejani, A., Kopf, A., Bradbury, J., Antiga, L., Raison, M., Gimelshein, N., Chilamkurthy, S., Killeen, T., Fang, L., & Bai, J. (2019) Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems (NeurIPS)* (pp. 8024–8035).

Stutz, D., Hein, M., & Schiele, B. (2019). Disentangling adversarial robustness and generalization. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 6976–6987).

Su, D., Zhang, H., Chen, H., Yi, J., Chen, P. Y., & Gao, Y. (2018). Is robustness the cost of accuracy?—A comprehensive study on the robustness of 18 deep image classification models. In *European conference on computer vision (ECCV)* (pp. 644–661).

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., & Fergus, R. (2014). Intriguing properties of neural networks. In *International conference on learning representations (ICLR)*.

Tu, C. C., Ting, P., Chen, P. Y., Liu, S., Zhang, H., Yi, J., et al. (2019). Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. *AAAI Conference on Artificial Intelligence (AAAI)*, *33*, 742–749.

Uesato, J., O'Donoghue, B., Kohli, P., van den Oord, A. (2018). Adversarial risk and the dangers of evaluating against weak attacks. In *International conference on machine learning (ICML)* (pp. 5025–5034).

Wang, Y., Du, S. S., Balakrishnan, S., Singh, A. (2017). Stochastic zeroth-order optimization in high dimensions. In *International conference on artificial intelligence and statistics (AISTATS)* (pp. 1356–1365).

Wang, L., Liu, X., Yi, J., Jiang, Y., & Hsieh, C. J. (2020). Provably robust metric learning. CoRR abs/2006.07024.

Wang, L., Liu, X., Yi, J., Zhou, Z. H., & Hsieh, C. J. (2019) Evaluating the robustness of nearest neighbor classifiers: A primal-dual perspective. CoRR abs/1906.03972.

Yan, Z., Guo, Y., Zhang, C. (2019). Subspace attack: Exploiting promising subspaces for query-efficient black-box attacks. In *Advances in neural information processing systems (NeurIPS)*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.