Towards Lightweight Black-Box Attacks against Deep Neural Networks

Chenghao Sun[1] **Yonggang Zhang**[2] **Wan Chaoqun**[3] **Qizhou Wang**[2]
Ya Li[4] **Tongliang Liu**[5] **Bo Han**[2] **Xinmei Tian**[1*]
[1]University of Science and Technology of China [2]Hong Kong Baptist University
[3]Alibaba Cloud Computing Ltd [4]iFlytek Research
[5]The University of Sydney

Abstract

Black-box attacks can generate adversarial examples without accessing the parameters of deep neural networks (DNNs), largely exacerbating the threats of deployed models. However, previous works state that black-box attacks fail to mislead DNNs when their training data and outputs are inaccessible. In this work, we argue that black-box attacks can pose practical attacks in this highly restrictive scenario where only several test samples are available. Specifically, we find that attacking the shallow layers of DNNs trained on a few test samples can generate powerful adversarial examples. As only a few samples are required, we refer to these attacks as *lightweight* black-box attacks. The main challenge to promoting lightweight attacks is to mitigate the adverse impact caused by the approximation error of shallow layers. As it is hard to mitigate the approximation error with few available samples, we propose Error TransFormer (ETF) for lightweight attacks. Namely, ETF transforms the approximation error in the parameter space into a perturbation in the feature space and alleviates the error by disturbing features. In our experiments, lightweight black-box attacks with the proposed ETF achieve surprising results. For example, even if only 1 sample per category is available, the attack success rate achieved by lightweight black-box attacks is only about 3% lower than that of the black-box attacks using complete training data [1].

1 Introduction

Black-box attack methods [12, 58, 62] can mount successful attacks without accessing the parameters of deep neural networks (DNNs), posing great challenges to deep learning in safety-critical situations. Existing black-box attack methods implicitly assume that the training data and/or the outputs of target models are available to adversaries. For scenarios where training data are available, adversarial examples can be generated by attacking surrogate models constructed for approximate target models. Using the target models' outputs for estimating gradients is another practical approach to crafting adversarial examples. In general, due to the low dependency on target model information, the black-box attacks raise realistic threats to real-world deep learning systems, including semantic segmentation [2, 37], object detection [18, 47, 46], and automatic driving [56].

However, these assumptions can be violated in many practical scenarios [3, 48], where both the training data and the outputs of target models are inaccessible. This realistic attack scenario is introduced in [3], generally termed as the *no-box setting*. Under such a restrictive setting, Li et

*Corresponding author (xinmei@ustc.edu.cn)

[1]Code is available at `https://github.com/sunch-ustc/Error_TransFormer/tree/ETF`

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

al. [35] stated that existing black-box attacks fail to mislead target models because it is now forbidden to access previously essential information in mounting black-box attacks. Therein, the failure of existing black-box attacks is not surprising. The reason is that constructing reliable surrogate models typically relies on accessing the complete training data of target models, which is forbidden in such a restrictive setting with only a few test samples available.

In this work, we aim to reveal the potential threats of black-box attacks in the no-box setting, as mounting black-box attacks with highly limited accessible information leaves many models exposed to attack. Specifically, we challenge the previous believes by raising the following question: *Can black-box attacks success in the no-box setting?* The doubts about the potential threats of the black-box attack are not groundless. Specifically, the potential threats of black-box attacks stem from two facts: a) adversarial examples can be generated by perturbing representations at shallow layers of DNNs [28, 27, 50]; b) regarding the representation of shallow layers, there do not exist critical differences between those models learned from a few data and that of the whole training data. [1]

Building upon the above facts, it is actually possible that an adversary can successfully mount attacks using limited samples, as powerful adversarial examples can be generated by attacking the shallow layers of DNNs trained on few samples. Concretely, the adversary can leverage available samples to construct a surrogate model to approximate the shallow layers of target models within acceptable errors. Consequently, adversarial examples can be generated by perturbing the features obtained from the shallow layers of the constructed surrogate model. As merely a few samples are required for attacking, we refer to black-box attacks in the no-box setting as *lightweight black-box attacks*. Intuitively, the closer the surrogate model is to the target model, the higher the attack success rate the crafted adversarial examples have [31]. If the approximation error of shallow layers is alleviated, the lightweight black-box attack can be as powerful as black-box attacks with complete training data. Hence, the main challenge to mounting a lightweight attack is to mitigate the adverse impact caused by the approximation error of shallow layers.

However, it is challenging to mitigate the approximation error, especially when the number of available samples is limited. Fortunately, it is straightforward to identify which kind of perturbations are preferred: if perturbations applied to a feature contribute to fooling the surrogate model, the perturbation is preferred. Therefore, bridging the connection between the parameter space and the feature space is the key to mitigating the approximation error. Accordingly, we propose transforming the approximation error in the parameter space as the perturbation in the feature space, dubbed Error TransFormer (ETF). Namely, ETF transforms the worst-case approximation error to the worst-case feature perturbation, leading to a min-max scheme that generates adversarial examples under the worst feature perturbations. We verify the attack success rate of lightweight black-box attacks using 7 models trained on the ImageNet dataset [49] and find that existing attack methods are much more potent than previously claimed. Moreover, the performance of lightweight black-box attacks can be further promoted by the proposed ETF, i.e., the attack success rate is only 3% lower than that achieved by black-box attacks having complete training data of target models.

2 Related works

2.1 Adversarial attack

According to the amount of accessible information, existing adversarial attacks can be roughly divided into two categories: white-box attacks and black-box attacks. White-box attacks [19, 43, 15, 16] assume that the target model is transparent to adversaries, i.e., adversaries can access all information about the target model. Nevertheless, the assumption of transparent target models can be violated in many practical scenarios [11]. Hence, black-box attacks [14, 38, 66, 7] are proposed and applied to the scenario where relatively limited information about the target model is accessible, i.e., only a certain number of model queries or the training data of target tasks are available.

Among these black-box attacks, intermediate-level attacks [50, 28, 24] are widely explored to improve the adversarial transferability. The core idea of these attacks is based on the empirical observation that well-trained models' intermediate features are transferable [60]. This is consistent with the recent works showing that adversarial examples comprise spurious features having the transferable property [65, 6]. Hence, Inkawhich et al. [28] propose to perturb the feature space of neural networks to create more transferable adversarial examples. These methods provide an interesting empirical conclusion that disturbing the shallow layers of DNNs can also generate adversarial examples.

Existing black-box attacks assume that the large-scale training data of target models and the feedback from querying target models are accessible, but these assumptions can be violated in many scenarios, e.g., the no-box setting [3]. Li et al. [35] stated that existing black-box attacks cannot be successfully mounted because existing DNNs require large-scale training data for generalization. Given only small-size data available, obtaining a surrogate model with strong generalization is challenging. Hence, Li et al. propose replacing black-box attacks with their proposed no-box attack [35], where they train a classical auto-encoder model instead of the supervised classification model due to the constraint of a small-scale dataset. Concretely, they train 20 auto-encoders for each category and generate adversarial examples by attacking these auto-encoders, which is time- and computational-consuming.

Different from existing black-box attacks, lightweight black-box attacks aim to reveal the potential risk of black-box attacks under the no-box setting. Because one main challenge to perform lightweight black-box attack is to mitigate the adverse impact caused by the approximation error, we propose error transformer that is a min-max strategy transforming the approximation error in the weight space to the feature space. The min-max strategy is different from that introduced in [57], where min-max strategy is proposed to flatten the loss landscape in the weight space, see Appendix F.

2.2 Approximation of Shallow Networks

A recent study [60] confirms the intuition that shallow layers in DNNs can be seen as low-level feature extractors, provided that strong data augmentation is used [1]. Motivated by this observation, Asano et al. [1] design a method to explore the information in every layer of DNNs. Asano et al. [1] train a supervision model on ImageNet LSVRC-12, and a self-supervision model on a small-scale dataset. Then, they apply linear probes [63] to all intermediate layers of networks, where a linear classifier is trained on the top of pre-trained and fixed feature representations. In this way, they evaluate the quality of the representation learned at different depths of the networks. The results show that, given heavily synthetic transformations, the shallow layers of DNNs learned from a few images can approximate that of DNNs trained on millions of images. According to the intriguing empirical observation, it is possible to construct a surrogate model with limited samples, where its shallow layers are similar to that of the target models. Besides, it is known that the neural networks prefer to make the decision through the spurious feature [65] on which are focused by models to correlate to the true label. Since shallow layers based on the small-scale or large-scale data set can acquire the similar spurious feature [1], it is an effective way to leverage the shallow model to mount an attack in the spurious features when the number of the data is limited.

3 Preliminaries

3.1 No-Box Threat Model

The no-box setting [3] denotes a threat model where available information about the target model is extremely limited, making it challenging to generate adversarial examples. According to the description in [3], the no-box threat model can be defined as follows.

Definition 1 (No-Box Threat Model) *A threat model is called the no-box threat model if the adversary in this scenario is not allowed to access the target model's training data and outputs, so it only has some samples that can be correctly predicted with high probability by the target model.*

The definition is consistent with the no-box setting introduced in [3], where the target model's training data and outputs are inaccessible to the adversary. Moreover, the available samples used for generating adversarial examples are expected to be limited. This is because if the number of samples is similar to or even larger than the number of samples used for training target models, the adversary can leverage these samples to construct a surrogate model similar to target models, making the condition of inaccessible training data meaningless. It is intuitive that target models can correctly predict the labels of these available samples with high probability. For example, it is meaningless to perform attacks using samples misclassified by the target model.

3.2 Lightweight Black-Box Attack

Definition 2 (Lightweight Black-Box Attack) *An attack is called the lightweight black-box attack if an adversary aims to perform an attack in the context of a no-box threat model.*

The definition of the lightweight black-box attack shows that lightweight attack is a special kind of black-box attacks. The extremely limited accessible information distinguishes it from other black-box attacks. For example, query-efficient black-box attacks require the feedback information of target models [25, 26], while lightweight attacks do not. To our best knowledge, how to perform a lightweight black-box attack is still lacking in the literature. For example, Li et al. [35] claim that black-box attacks fail to fool target models when their training data and outputs are inaccessible.

4 Approach

This section gives a detailed description of how to perform lightweight black-box attacks and the proposed Error TransFormer (ETF) to alleviate to adverse impact caused by approximation error.

4.1 Lightweight Surrogate Model

Surrogate models are widely used in black-box attacks since adversarial examples are usually generated by attacking surrogate models when querying target models is forbidden. Since the lightweight black-box attack is a special kind of black-box attack, performing the attack also requires constructing surrogate models. We call surrogate models employed in lightweight black-box attacks the *lightweight surrogate model*, building upon the fact that the number of available samples in the no-box threat model is typically limited.

Similar to existing black-box attacks, we assume that the label information of samples used for generating adversarial examples are accessible. In this scenario, the lightweight surrogate model is trained for a classification task. Specifically, the lightweight model is realized by training a classification DNN equipped via conventional supervised learning, using cross-entropy loss $\ell(\cdot,\cdot)$:

$$\min_{w} \mathbb{E}_{(x,y)\sim\hat{D}}\ \ell(f(x;w),y), \tag{1}$$

where w stands for the parameters of the surrogate model f, and (x, y) denotes the sample label pair of random variables sampled from the distribution of natural data $\hat{\mathcal{D}}$.

In black-box attacks, adversary only leverages the shallow layers of DNNs, leading to a straightforward approach to constructing lightweight surrogate models. Specifically, we can train the lightweight surrogate model in a contrastive manner, where the supervised information is no longer necessary:

$$\min_{w}\ \frac{1}{|\hat{D}|}\sum_{\mathbf{x}\in\hat{D}} -\log\frac{\sigma(f(\mathbf{x};w),f(\mathcal{T}(\mathbf{x});w))}{\sum_{\mathbf{x}'\sim\hat{D}\backslash\mathbf{x}}\ \sigma(f(\mathbf{x};w),f(\mathcal{T}(\mathbf{x}');w))/(|\hat{D}\backslash\mathbf{x}|-1)}, \tag{2}$$

where σ is a specific similarity metric, e.g., cosine similarity, $\mathcal{T}$ denotes the data transformation operation, and $\hat{D}\backslash\mathbf{x}$ means that $\mathbf{x}$ is removed from the dataset $\hat{D}$. The contrastive strategy in Eq. 2 makes it possible to perform attacks when the label information is unavailable. Unless otherwise specified, we mainly uses Eq. 1 to train the surrogate models.

4.2 Feature Space Perturbation

Built upon the empirical observation [60], using limited samples can approximate the shallow layers of DNNs within acceptable errors. This suggests that deep layers other than shallow layers can cause large approximation error. Accordingly, merely the shallow layers of the learned lightweight surrogate models are used for generating adversarial examples, which is similar to the feature space attacks [50]. For brevity, we define the shallow layers as φ and denote the feature as $\varphi(x;w)$.

In feature space attacks, guide images are usually required for generating adversarial examples [50]. A natural image used for generating adversarial example is usually called source image. Given a source image x_s, we generate an adversarial example x_{adv} by perturbing x_s such that its feature is similar to that of a guide image x_g, where the labels of these two images are different. Specifically, the adversarial examples are generated as follows:

$$x_{adv} = \arg\min_{\|x'-x\|_p\leq\epsilon} d(\varphi(x_g;w),\varphi(x';w)), \tag{3}$$

where d stands for a specific distance, ϵ controls the strength of adversarial perturbation, and x' denotes the perturbed version of the source image.

4.3 Error Transformer

Although applying feature space attacks to lightweight surrogate models can generate transferable adversarial examples, the approximation error will cause adverse impact to mounting attacks. The reason is that large dissimilarity between the surrogate models and the target models reduces the attack success rate. Therefore, the main challenge to mounting lightweight attacks is to mitigate the adverse impact caused by the approximation error. However, available samples are usually limited in the no-box threat model, making it hard to alleviate the approximation error.

To address the challenge, we propose transforming the parameter space's approximation error as the feature space's perturbation. Specifically, we seldom know which perturbations can point (from the surrogate model) to the target model, making it challenging to alleviate the approximation error in the weight space. In contrast, we have the prior that samples with different labels should have distinguishable representations. Thus, we can leverage such prior knowledge to select preferred perturbations in the feature space, i.e., we prefer perturbations that can make representations of samples with different labels indistinguishable. Similarly, it is also straightforward to define the "bad" perturbations, leading to the design of the min-max optimization to identify the "worst" model. Thus, connecting the parameter space and feature space makes it possible to mitigate the adverse impact caused by approximation error. To bridge the connection, we introduce the following key identity, which is also used in [45]:

$$\varphi(x; \{w^1 + w^1 A\} \cup \{w \backslash w^1\}) = g((w^1 + w^1 A)x; w \backslash w^1) = g(w^1(x + Ax); w \backslash w^1) = \varphi(x + Ax; w), \quad (4)$$

where w^1 stands for the first layer parameters of model φ, A is a transformation matrix used for perturbing w^1, $\{w^1 + w^1 A\} \cup \{w \backslash w^1\}$ means that the first layer's parameters are perturbed, the other layers' parameters keep unchanged, and g is the function parameterized with $w \backslash w^1$ used for processing the first layer's outputs. According to Eq. 4, we can transform a perturbation in the parameter space as the perturbation in the data space, i.e., $\varphi(x; \{w^1 + w^1 A\} \cup \{w \backslash w^1\}) = \varphi(x + Ax; w)$. Similarly, we can apply the transformation operation to the hidden layers, see Appendix A for more details.

Built upon the connection in Eq. 4, we can transform the first layer parameters' discrepancy between the target model and the surrogate model. Taking the first layer as an example, let w_t be the parameters of the target model and w_t^1 be its first layer parameters, where the other parameters are the same as the surrogate model. Assume [2] that there exists a transformation matrix A applied to the first layer parameters of the surrogate model such that $w_t^1 = w^1 + w^1 A$, we have:

$$\varphi(x; w_t^1 \cup \{w_t \backslash w_t^1\}) = \varphi(x; \{w^1 + w^1 A\} \cup \{w_t \backslash w_t^1\}) = \varphi(x + Ax; \{w^1\} \cup \{w_t \backslash w_t^1\}) = \varphi(x + Ax; w). \quad (5)$$

The identity in Eq. 5 shows that we can find a perturbation in the data space to alleviate the difference in the first layer parameters between these two models. Built upon the transformation, we can transform the worst approximation error to the worst perturbation in the data space. Therefore, we can employ a min-max strategy to mitigate the adverse impact of approximation error through generating adversarial examples under the worst perturbation. Thus, lightweight black-box attacks with ETF generate adversarial examples as follows:

$$x_{adv} = \arg \min_{\|x' - x\|_p \leq \epsilon} \max_{\|\Delta_s\|_p \leq \tau,\ \|\Delta_g\|_p \leq \tau} d(\varphi(x_g + \Delta_g; w), \varphi(x' + \Delta_s; w)), \quad (6)$$

where $\|\cdot\|_p$ is the ℓ_p-norm, ϵ and τ control the strength of perturbations, Δ_g and Δ_s are the data space perturbations for mitigating the approximation error in parameter space, and x' denotes the perturbed version of source image. In this way, we reduce the error in parameter space to approximate the target model. We solve the inner maximization problem by generating perturbations in the feature space, given a perturbed adversarial example. This step aims to mitigate the approximation error. The outer minimization problem is solved by finding adversarial perturbations in the input space, the same as the adversarial example generation. After iterative generation of perturbations, adversarial examples are generated by attacking a model with a reduced approximation error.

5 Experiments

In this section, we conduct extensive experiments to verify the power of lightweight black-box attacks augmenting with ETF.

[2] Discussion about the assumption can be found in Appendix E.

Table 1: The accuracy (%) of 7 normally trained target models evaluated on 1000 adversarial examples generated by lightweight black-box attacks or existing black-box attacks, under $\epsilon \leq 0.1$. Shallow-(PGD, MI, DI, TI) means applying PGD, MI, DI and TI to the shallow layers of the model. Deep-(PGD, MI, DI and TI) means applying PGD, MI, DI and TI to the model's output. EFT-(PGD, MI, DI and TI) means applying ETF combined with PGD, MI, DI or TI to the shallow layers. (The lower, the better)

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Autoattack[9]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deep-PGD	49.01±0.23	52.26±0.25	60.71±0.74	57.92±0.37	27.94±0.18	60.18±0.64	44.20±0.63	50.31±0.52
Deep-MI	38.92±0.43	42.37±0.37	49.53±0.49	49.06±0.89	19.44±0.75	49.11±0.82	33.46±0.80	40.69±0.96
Deep-DI	43.34±0.40	43.13±0.52	53.78±0.38	55.41±0.53	23.53±0.52	51.77±0.48	38.14±0.74	44.15±0.60
Deep-TI	49.46±0.52	49.64±0.27	58.89±0.71	58.75±0.30	26.19±0.16	56.31±0.58	44.02±0.46	49.03±0.51
Shallow-MI	22.62±0.25	30.83±0.48	34.05±0.27	35.74±0.76	12.31±0.41	29.98±0.65	17.72±0.31	26.17±0.56
Shallow-DI	22.14±0.39	29.78±0.17	35.51±0.33	35.79±0.61	8.99±0.42	30.61±0.88	16.88±0.47	25.67±0.55
Shallow-TI	21.82±0.45	28.54±0.34	34.78±0.15	34.71±0.39	7.96±0.48	30.14±0.85	15.77±0.51	24.81±0.37
Shallow-PGD	22.93±0.33	31.07±0.58	34.71±0.67	36.20±0.87	13.08±0.36	32.16±0.66	16.65±0.54	26.69±0.49
ETF-PGD	**14.11**±0.24	20.22±0.29	24.20±0.34	24.74±0.37	6.96±0.44	**20.73**±0.28	**10.66**±0.31	**17.37**±0.35
ETF-MI	15.32±0.52	19.97±0.28	26.25±0.14	28.10±0.65	7.02±0.43	22.21±0.66	12.23±0.32	18.72±0.45
ETF-DI	14.77±0.35	20.63±0.32	23.71±0.83	25.70±0.51	7.23±0.37	20.22±0.64	11.53±0.50	17.68±0.47
ETF-TI	15.45±0.37	**18.03**±0.34	**22.63**±0.45	**24.20**±0.68	**6.94**±0.41	21.53±0.25	12.88±0.34	17.38±0.71
Deep*-PGD	12.43±0.51	28.15±0.43	16.54±0.49	12.61±0.22	7.09±0.32	13.33±0.54	9.64±0.28	14.25±0.37
Deep*-MI	11.77±0.75	25.14±0.56	18.10±0.64	13.72±0.34	4.26±0.35	14.61±0.37	8.30±0.37	13.70±0.68
Deep*-DI	7.61±0.41	18.17±0.45	8.23±0.33	9.90±0.57	6.66±0.34	9.72±0.42	7.91±0.46	9.74±0.55
Deep*-TI	9.55±0.48	23.48±0.86	13.51±0.46	10.63±0.64	6.46±0.26	10.92±0.61	9.55±0.35	12.01±0.43

Deep* refers to the attacks mounted in the model trained on the large-scale training data.

5.1 Experimental Setup

Models Architectures. All surrogate models are based on the ResNet-18 [21]. In the ordinary settings of the black-box attack, the general surrogate models are trained through the whole training set of ImageNet [49]. On the contrary, the lightweight surrogate models adopt only $1,000$ images randomly sampled from the validation set of ImageNet, considering that images in the training set are probably inaccessible in practice [35]. Results evaluated on the CIFAR10 dataset can be found in Appendix D. Please refer to Table 5 in the Appendix B for the detailed information on the lightweight surrogate model and the general surrogate model. Regarding the target models, various model architectures are selected for full comparison, including, VGG-19 [53], Inception v3 [55], ResNet-152 [21], DenseNet [23], SENet [22], wide ResNet (WRN) [61] and MobileNet v2 [51]. All these target models are well-trained on ImageNet.

Implement Details. The lightweight surrogate models are randomly initialized. The batch size is 128, the epoch is 500, and the initial learning rate is 0.4, linearly decreasing to 0.008. To better approximate the low-level feature of shallow layers, we refer to the self-supervision work [5] and apply various random augmentations to the data in each epoch for training. To mount attacks, the classic methods, e.g., PGD [42], MI [11], DI [58], and TI [12], are applied to all the surrogate models. Unless otherwise specified, our attack is mounted on the first layer of ResNet-18 for lightweight surrogate models. Following previous works on intermediate-level attacks, the metric d in Eq. 3 of Shallow- in Table 1 is instantiated as the ℓ_2-norm. Meanwhile, to maximize the potential of ETFs, we apply contrastive loss to ETF-$(\cdot)$ in Table 1. The Top-1 prediction accuracy is adopted as the evaluation metric, i.e., lower the classification accuracy means better attack success rate. All the experiments are run for 5 individual trials with different random seeds. In our experiments, target models are evaluated using ℓ_∞-norm adversarial examples with maximum distortion $\epsilon = 0.1$, results evaluated under mores settings, e.g., more strict constraints $\epsilon = 0.05$, ℓ_2-norm adversarial examples, and more test images 5,000 images, can be found in Appendix B.4.

5.2 Main Results

Results on normally trained models. The results of classification accuracy are shown in Table 1, which exhibits the evaluations for both lightweight and general surrogate models. The last four rows

Table 2: The accuracy (%) of 7 normally trained target models evaluated on 1000 adversarial examples generated by no-box attacks and lightweight black-box attacks with different intermediate-level attacks. The best results are in **bold**. (The lower, the better)

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
No-box[35]	18.74	33.68	34.72	26.06	42.36	33.16	16.34	29.29
ILA[24]	20.13	28.01	35.72	35.14	9.16	29.97	14.31	24.63
AA[28]	22.76	31.21	36.67	37.94	7.72	33.16	18.17	26.81
ETF	**14.11**	**20.22**	**24.20**	**24.74**	**6.96**	**20.73**	**10.66**	**17.37**

show the ordinary results where the general surrogate models are trained through all data in the training set. The black-box attacks are mounted based on the whole model, termed "Deep/Deep*". As the size of the training data is reduced to $1,000$, this conventional method obviously loses its aggressiveness. Specifically, the averaged accuracies of 3-6 rows are among 40-50%, revealing limited attacking capability compared to 66.99% of the clean accuracy. It demonstrates the necessity of large amount of data for classic black-box attacks. In rows 7-10, shallow layers without ETF are used to generate adversarial examples. The averaged accuracies of around 25% illustrates the less dependence of shallow networks on the amount of data. In rows 11-14, attacks on shallow layers are further enhanced by ETF, decreasing the averaged accuracies to about 17%. The narrow gap between the proposed ETF and the general surrogate model indicates the potential capability of the black-box attack in the no-box settings.

Results on Adversarially Trained Models. Here, we consider the models that are trained through adversarial training [42], which is more persuasive to evaluate the effectiveness of attacking methods. Following [42], the target models are trained on the whole training set of ImageNet. During adversarial training, the adversarial examples are generated under $\epsilon = 0, 4/255, 8/255$, where ϵ means the ℓ_∞-norm constraint. Table 3 exhibits the attacking results mounted by ETF, black-box attacks (general surrogate models with all training data avaliable), and no-box attacks. For simplicity, only the results of PGD attacks are reported. As analysed above, the black-box attacks shows the best attacking results, while ETF only has a narrow gap with it. However, when $\epsilon = 4/255$ or $8/255$, ETF exceeds the black-box attacks (29.13/26.14 vs. 48.11/38.24). It is interesting that black-box attacks are even worse than lightweight attacks. We believe that exploiting the phenomenon can bring something new, but we leave it as our future work due to the limited space.

5.3 Further Analysis and Ablation Study

Intermediate-Level Perturbations. Since the lightweight black-box attacks are implemented mainly on intermediate features, two classic attacks mounted on intermediate features, i.e. ILA [24] and AA [28], are selected for experiments. Comparing the ILA and AA in Table 2 and Shallow-MI/DI/TI/PGD in Table 1, the difference between their average accuracies is marginal. It indicates those intermediate-level feature perturbation has limited benefit to lightweight black-box attacks. The reason is that the intermediate-level attacks highly rely on the low-level features extracted by the lightweight model to generate adversarial examples and it can weaken the role of high-level information when attacking. The proposed ETF achieves better performance. On one hand, ETF does not rely on the features from high-level layers.

Table 3: The performance of different attacks on the adversarial trained ResNet-50 [13]. Therein, ϵ refers to the constraint ℓ_∞ in adversarial examples for adversarial training. The accuracy (%) is evaluated on 1000 adversarial examples. $\epsilon = 0.1$ (the lower the better). White-box refers to Auto-Attack [9].

Adv_model	Clean	ETF ours	Black-box [42]	No-box [35]	White-box [9]
$\epsilon = 0/255$	69.43	16.97	**8.20**	24.53	0.00
$\epsilon = 4/255$	55.62	**29.13**	48.11	39.62	0.00
$\epsilon = 8/255$	41.68	**26.14**	38.24	35.87	0.48

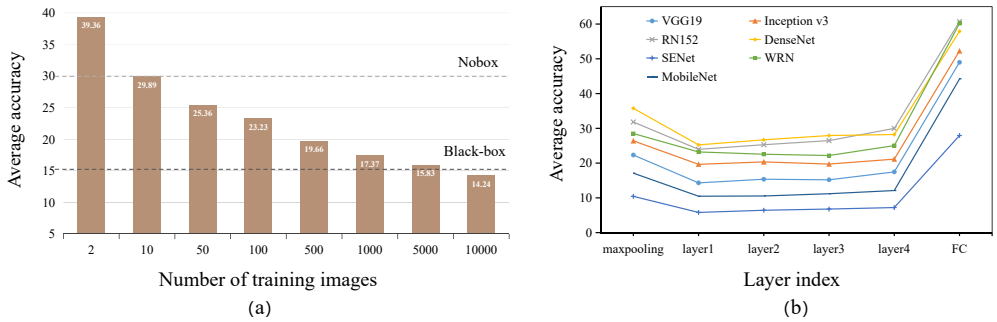

Figure 1: (a) How the lightweight attack performance of our approach varies with the number of images used for training the surrogate model. (b) The influence of low-level feature extraction at different layers of ResNet-18 on lightweight black-box attack performance. (The lower, the better)

On the other hand, ETF introduces a min-max scheme to mitigate the adverse impact induced by the approximation error from the shallow layers.

Comparison with No-box Attacks. No-box attack is the first work to explore how to mount attacks in the no-box setting. Since Li et. al [35] stated that black-box attacks do not work in this setting, surrogate models are instantiated as auto-encoders rather than models for approximating target models. Namely, they design the pretext task to train 20 auto-encoders per category to generate adversarial examples. However, the difference between the pretext tasks and the target tasks may significantly reduce the transferability of adversarial examples. In contrast, our method trains the lightweight surrogate model for tasks similar to or same as the task of target models. Hence, though only one sample per category is available, we achieve better performance than no-box attacks, as shown in Table 2.

Number of Training Samples. One key difference between the lightweight black-box attack and existing black-box attacks mainly lies in the number of samples n used for training surrogate models. Therefore, we further explore how lightweight attack performance varies with different number of samples, i.e., n. We conduct experiments on randomly selected n images from the validation set of ImageNet. The experimental results are shown in Figure 1. The average accuracy decreases as more samples are available, thus achieving better attacking results. More data makes it possible for the lightweight black-box attack to approximate the shallow layers of the target model more accurately. Especially the lightweight black-box attacks with $10,000$ test images can achieve the same performance as general black-box attacks which train the surrogate model on $1,200,000$ training images. Also, our method can still mount attacks even when only 2 or 10 images are available.

Surrogate Model. As analyzed above, we utilize the first few layers of the lightweight surrogate model to approximate the shallow layers of the target model to mount attacks. Hence, it would be interesting to study how the layers selection for lightweight models impacts the attack performance. We adopt different layers of ResNet-18 to generate adversarial examples utilizing ETF and attack the different target models. The results are summarized in Fig. 1. It can be seen that low-level information in the first block of the model is sufficient for achieving promising attack performance. With more block information on the model, the performance of the attack does not improve. In particular, the transferability of adversarial examples significantly decreases when we attack the fully-connected layer, verifying our analysis that the high-level layer of the lightweight model cannot approximate the target model well. More results for architecture selection can be found in Appendix B.4.

Adversarial Example Visualization. To verify that adversarial examples are truly imperceptible, we provide visualization in Figure 2, where Deep*-PGD attack (using training images), Deep-PGD attack (using test images), and lightweight black-box attack are considered.

6 Discussion

Though the no-box threat model provides a weak assumption, many realistic scenarios may be more complex. For example, adversaries have access to only data without supervision information due to security [36] or privacy [48] issues. In more extreme cases, e.g., autonomous vehicle [56], adversaries even have no access to the test samples from the distribution over which the model is trained. Instead,

Table 4: Model accuracy (%) under lightweight black-box attacks under challenging scenarios, where supervision information or the in-distribution data are unavailable, named Unsupervised and OOD.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Supervised	14.11	20.22	**24.20**	24.74	6.96	**20.73**	10.66	17.37
Unsupervised	15.54	**19.16**	26.27	23.75	7.66	22.79	11.43	18.08
OOD	**6.13**	21.72	25.44	**21.89**	**5.02**	24.33	**7.16**	**15.96**

(a) ETF (b) Deep (c) Deep*

Figure 2: Adversarial examples crafted by: a) ETF, b) Deep, and c) Deep* attacks.

the adversaries have access to samples from a similar (but not exact) distribution (because the adversary knows the deployment environment). Hence, we consider two more challenging scenarios in which we provide solutions to mount lightweight black-box attacks.

Insufficient supervision information. We design an experiment for mounting attacks using $1,000$ unlabeled samples randomly selected from the ImageNet validation set to explore the power of ETF under this challenging setting with unlabeled data. Thanks to the framework of lightweight attack, it is feasible to train the unsupervised model on unlabeled data for attacks using Eq. 2. Specifically, we adopt contrastive learning [5] to train a lightweight surrogate model on the small-scale unlabeled dataset and generate adversarial examples by attacking the trained lightweight surrogate model. The performance is shown in Table 4, where ETF still achieves similar performance without supervision information as that of the results under supervision. Intuitively, we can also employ other self-supervised learning method, e.g., rotation prediction, to train shallow layers, see details of experimental settings and results in Appendix G.

Consideration of Out-of-distribution. We further study the possibility for mounting attacks under a more strict scenario, where only OOD (out-of-distribution, OOD) data are available. Specifically, we utilize the model pre-trained on OOD samples to mount lightweight black-box attacks. The rationality lies in that only low-level information is required to mount lightweight black-box attacks and the acquisition of low-level information. We load pre-trained ResNet-50 on STL-10 [39] dataset as the surrogate model. Without further fine-tuning, we experiment with two test in-distribution samples, and the results are recorded in Table 4, named OOD. As we can see, the lightweight black-box attack with models trained on OOD data can achieve a surprising attack success rate, showing that black-box attacks enhanced with ETF can pose practical threats to deployed models.

7 Conclusion

In this paper, we propose the conception of the lightweight black-box attack to reveal the potential risk of black-box attacks under the no-box threat model. To mount effective lightweight attacks, we find it crucial to leverage DNNs‘ shallow layers because they can learn similar features regardless of

the number of samples. This is based on the empirical observations that the model trained with limited samples can approximate the shallow layers of the target models within acceptable errors, which can be used for crafting adversarial examples. Therefore, to further enhance the performance, we propose Error TransFormer (EFT) to decrease the approximation error by transforming the approximation error in the parameter space into the feature space. The experiments show that the proposed method achieves a surprising attack success rate under the no-box threat model using only one image per category, i.e., only 3% lower than black-box attacks with complete training data of the target model.

8 Acknowledgements

This work was supported by NSFC No. 61872329 and No. 62222117, and the Fundamental Research Funds for the Central Universities under contract WK3490000005. YZ and BH was supported by NSFC Young Scientists Fund No. 62006202 and Guangdong Basic and Applied Basic Research Foundation No. 2022A1515011652. TL was partially supported by Australian Research Council Projects DP180103424, DE-190101473, IC-190100031, DP-220102121, and FT-220100318.

References

[1] Y. M. Asano, C. Rupprecht, and A. Vedaldi. A critical analysis of self-supervision, or what we can learn from a single image. In ICLR, 2020.

[2] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell., 40(4):834–848, 2018.

[3] P. Chen, H. Zhang, Y. Sharma, J. Yi, and C. Hsieh. ZOO: zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In B. M. Thuraisingham, B. Biggio, D. M. Freeman, B. Miller, and A. Sinha, editors, Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pages 15–26, 2017.

[4] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In AAAI, 2018.

[5] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton. A simple framework for contrastive learning of visual representations. In ICML, 2020.

[6] Y. Chen, C. Wei, A. Kumar, and T. Ma. Self-training avoids using spurious features under domain shift. In NeurIPS, 2020.

[7] Y. Chen, H. Yang, Y. Zhang, K. Ma, T. Liu, B. Han, and J. Cheng. Understanding and improving graph injection attack by promoting unnoticeability. In ICLR, 2022.

[8] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), 2021.

[9] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In ICML, 2020.

[10] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In NeurIPS, 2014.

[11] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. In CVPR, 2018.

[12] Y. Dong, T. Pang, H. Su, and J. Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In CVPR, 2019.

[13] L. Engstrom, A. Ilyas, H. Salman, S. Santurkar, and D. Tsipras. Robustness (python library), 2019.

[14] L. Gao, Q. Zhang, J. Song, X. Liu, and H. T. Shen. Patch-wise attack for fooling deep neural network. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, ECCV, 2020.

[15] R. Gao, F. Liu, J. Zhang, B. Han, T. Liu, G. Niu, and M. Sugiyama. Maximum mean discrepancy test is aware of adversarial attacks. In M. Meila and T. Zhang, editors, ICML, 2021.

[16] R. Gao, J. Wang, K. Zhou, F. Liu, B. Xie, G. Niu, B. Han, and J. Cheng. Fast and reliable evaluation of adversarial robustness with minimum-margin attack. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, ICML, 2022.

[17] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In ICLR, 2018.

[18] R. B. Girshick. Fast R-CNN. In ICCV, 2015.

[19] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. ICLR, 2015.

[20] S. Gowal, S. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann. Improving robustness using generated data. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, NeurIPS , 2021.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[22] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In CVPR, 2018.

[23] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In CVPR, 2017.

[24] Q. Huang, I. Katsman, Z. Gu, H. He, S. J. Belongie, and S. Lim. Enhancing adversarial example transferability with an intermediate level attack. In ICCV, 2019.

[25] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. In ICML, 2018.

[26] A. Ilyas, L. Engstrom, and A. Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In ICLR, 2018.

[27] N. Inkawhich, K. Liang, B. Wang, M. Inkawhich, L. Carin, and Y. Chen. Perturbing across the feature hierarchy to improve standard and strict blackbox attack transferability. In NeurIPS, 2020.

[28] N. Inkawhich, W. Wen, H. H. Li, and Y. Chen. Feature space perturbations yield more transferable adversarial examples. In CVPR, 2019.

[29] Q. Kang, Y. Song, Q. Ding, and W. P. Tay. Stable neural ODE with lyapunov-stable equilibrium points for defending against adversarial attacks. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, NeurIPS , 2021.

[30] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. ICLR, 2016.

[31] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In ICLR, 2017.

[32] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In ICLR, 2017.

[33] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In NeurIPS, 2018.

[34] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, NeurIPS, 2018.

[35] Q. Li, Y. Guo, and H. Chen. Practical no-box adversarial attacks against dnns. In NeurIPS , 2020.

[36] K. J. Liang, G. Heilmann, C. Gregory, S. O. Diallo, D. Carlson, G. P. Spell, J. B. Sigman, K. Roe, and L. Carin. Automatic threat recognition of prohibited items at aviation checkpoint with x-ray imaging: a deep learning approach. In Anomaly Detection and Imaging with X-Rays (ADIX) III, volume 10632, page 1063203. International Society for Optics and Photonics, 2018.

[37] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[38] Y. Long, Q. Zhang, B. Zeng, L. Gao, X. Liu, J. Zhang, and J. Song. Frequency domain model augmentation for adversarial attack. CoRR, abs/2207.05382, 2022.

[39] D. G. Lowe. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis., 60(2):91–110, 2004.

[40] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In ICLR, 2018.

[41] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. N. R. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In ICLR, 2018.

[42] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In ICLR, 2018.

[43] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In CVPR, 2016.

[44] T. Pang, M. Lin, X. Yang, J. Zhu, and S. Yan. Robustness and accuracy could be reconcilable by (proper) definition. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, ICML, 2022.

[45] H. Petzka, M. Kamp, L. Adilova, C. Sminchisescu, and M. Boley. Relative flatness and generalization. In NeurIPS, 2021.

[46] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In CVPR, 2016.

[47] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In NeurIPS, 2015.

[48] M. Rigaki and S. Garcia. A survey of privacy attacks in machine learning. CoRR, abs/2007.07646, 2020.

[49] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. Int. J. Comput. Vis., 115(3):211–252, 2015.

[50] S. Sabour, Y. Cao, F. Faghri, and D. J. Fleet. Adversarial manipulation of deep representations. In Y. Bengio and Y. LeCun, editors, ICLR, 2016.

[51] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In CVPR, 2018.

[52] V. Sehwag, S. Mahloujifar, T. Handina, S. Dai, C. Xiang, M. Chiang, and P. Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? arXiv preprint arXiv:2104.09425, 2021.

[53] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In Y. Bengio and Y. LeCun, editors, ICLR, 2015.

[54] J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. IEEE Transactions on Evolutionary Computation, 23(5):828–841, 2019.

[55] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In CVPR, 2016.

[56] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski. Simulation-based adversarial test generation for autonomous vehicles with machine learning components. In 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018.

[57] D. Wu, S.-T. Xia, and Y. Wang. Adversarial weight perturbation helps robust generalization. In NeurIPS, 2020.

[58] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille. Improving transferability of adversarial examples with input diversity. In CVPR, 2019.

[59] K. Yang, T. Zhou, Y. Zhang, X. Tian, and D. Tao. Class-disentanglement and applications in adversarial detection and defense. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, NeurIPS , 2021.

[60] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In NIPS, 2014.

[61] S. Zagoruyko and N. Komodakis. Wide residual networks. In R. C. Wilson, E. R. Hancock, and W. A. P. Smith, editors, BMVC, 2016.

[62] Q. Zhang, X. Li, Y. Chen, J. Song, L. Gao, Y. He, and H. Xue'. Beyond imagenet attack: Towards crafting adversarial examples for black-box domains. In ICLR, 2022.

[63] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In CVPR, 2017.

[64] X. Zhang, J. Zou, K. He, and J. Sun. Accelerating very deep convolutional networks for classification and detection. IEEE Trans. Pattern Anal. Mach. Intell., 38(10):1943–1955, 2015.

[65] Y. Zhang, M. Gong, T. Liu, G. Niu, X. Tian, B. Han, B. Schölkopf, and K. Zhang. Adversarial robustness through the lens of causality. In ICLR, 2022.

[66] Y. Zhang, X. Tian, Y. Li, X. Wang, and D. Tao. Principal component adversarial example. IEEE Trans. Image Process., 29:4804–4815, 2020.

A ETF on Hidden Layers

Before giving the details of applying ETF to hidden layers, we revisit the key identity:

$$\varphi(x; \{w^1 + w^1 A\} \cup \{w \backslash w^1\}) = g((w^1 + w^1 A)x; w \backslash w^1) = g(w^1(x + Ax); w \backslash w^1) = \varphi(x + Ax; w), \quad (7)$$

where w^1 stands for the first layer (convolution layer) parameters of model φ, A is a transformation matrix used for perturbing w^1, $\{w^1 + w^1 A\} \cup \{w \backslash w^1\}$ means that the first layer's parameters are perturbed and the other layers' parameters keep the same, and g is the function parameterized with $w \backslash w^1$ used for processing the first layer's outputs. Built upon Eq. 7, we have $\varphi(x; \{w^1 + w^1 A\} \cup \{w \backslash w^1\}) = \varphi(x + Ax; w)$.

The model φ can be decomposed: $\varphi = g^l \circ \varphi^l$, where $\varphi^l(x)$ denotes the hidden feature at layer l. Then the output feature at layer L is expressed as $\varphi^L(x; w) = g^l(w^l \varphi^l(x))$, where w^l is the parameter used for processing the feature $\varphi^l(x)$. Then, we can apply the identity at layer l:

$$\varphi^L\left(x; \{w^l + w^l A^l\} \cup \{w \backslash w^l\}\right) = g^l\left((w^l + w^l A^l)\varphi^l(x)\right) = g^l\left(w^l(A^l \varphi^l(x) + \varphi^l(x))\right), \quad (8)$$

where A^l is the transformation matrix at layer l. According to Eq. 8, we can transform a perturbation in the parameter space (Aw^l) as the perturbation in the feature space ($A\varphi^l(x)$). Thus, lightweight black-box attacks with ETF generate adversarial examples as follows:

$$x_{adv} = \arg \min_{\|x' - x\|_p \leq \epsilon} \max_{\|\Delta_s^l\|_p \leq \tau,\ \|\Delta_g^l\|_p \leq \tau, l \in \{0,1,\ldots,L-1\}} d(\varphi(x_g, \cup_l \Delta_g^l; w), \varphi(x', \cup_l \Delta_s^l; w)), \quad (9)$$

where Δ_s^l (Δ_g^l) denotes the feature space perturbation ($l = 0$ means the data space), $\cup_l \Delta_s^l$ ($\cup_l \Delta_g^l$) stands for all perturbations in the feature space, and $\varphi(x_g, \cup_l \Delta_g^l; w)$ ($\varphi(x_g, \cup_l \Delta_g^l; w)$) stands for the output feature with perturbed features, where the feature of source (guide) image at layer l is perturbed by Δ_s^l (Δ_g^l). Although we can perform the error transformation in the feature space, the features obtained using weights with approximation errors make it challenging. Specifically, we merely know that the input distribution (can be seen as the feature map) is not biased, but the feature map obtained using any weights will cause bias. Thus, we merely apply ETF to the input layer and leave further exploitation of applying it to hidden layers as future work.

B Additional Detail Description

B.1 Simplified Architecture

The architecture of surrogate models is modified to avoid overfitting. Considering the limited amount of data, we employ a network with a small model capacity to instantiate the feature extractor. In particular, ResNet-18 is simplified by reducing the number of blocks in each layer, i.e., only one block is used in each layer of ResNet-18.

B.2 Classification Ability of the Surrogate Model.

To demonstrate that the lightweight black-box attack performance does not rely on the generalization of classification, we show the classification accuracy of the lightweight surrogate model used for ETF attacks in Table 5. To make the conclusion clearer, we also report the performance of a general surrogate model, which is trained on the training set of target models. We can see that the test accuracy of the lightweight surrogate model

Table 5: The classification performance of the lightweight surrogate model and the general surrogate model.

	Lightweight surrogate model	General surrogate model
Number of samples for training	1000	1 200 000
Training accuracy	96.38	72.71
Test accuracy	2.36	63.24

(about 2) is drastically lower than that of the general model, bringing fresh air for black-box attacks. Specifically, the common sense in black-box attacks is that mounting attacks requires a surrogate model, which generalizes well on the test set. However, the experimental results in Table 5 show that the lightweight surrogate model has poor classification performance, suggesting that the generalizability of surrogate models is not a necessary condition for performing black-box attacks.

B.3 Approximation to more layers.

Besides applying ETF in the first layer, we also conduct experiments that apply ETF to all layers except layer L. The results are reported in Table 6. It can be seen that applying ETF to other layers can marginally promote the attack success rate. The phenomenon may result from the fact that the approximation error can accumulate with depth. Moreover, considering that applying ETF will cause more computational overhead, we merely give the results of applying ETF to the first layer in the main paper.

Table 6: Apply ETF to all layers (except layer L) to further approximate the target model. This experiment is called "All" below. Similarly, "First" means only applying ETF to the first layer. (The lower, the better)

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
No-box[35]	18.74	33.68	34.72	26.06	42.36	33.16	16.34	29.29
First	14.11	20.22	24.20	24.74	6.96	20.73	10.66	17.37
All	13.33	20.21	22.66	25.49	5.40	21.74	9.88	**16.95**

B.4 Hardware Configuration and Computation Costs.

We conduct experiments using GEFORCE RTX 2080 Ti, CPU AMD Ryzen 7 3700X @3.6 GHz. As merely 1,000 samples are required for the training of the lightweight surrogate model, the computational overhead is much less than the training of general surrogate models.

C Further Experiments on ImageNet

Strict Constraint. We conduct experiments under smaller ϵ, i.e., $\epsilon = 0.05$. The results are given in Table 7, demonstrating that ETF can generate powerful adversarial examples even with meeting more strict constraints, i.e., smaller ϵ.

Table 7: The accuracy of 7 normally trained target models evaluated on 1,000 adversarial examples generated by lightweight black-box attacks or existing black-box attacks, under $\epsilon \leq 0.05$. The Shallow-(PGD, MI, DI, TI) mean applying PGD, MI, DI and TI to the shallow layers of the model. Deep-(PGD, MI, DI and TI) mean applying PGD, MI, DI and TI to the model's output. EFT-(PGD, MI, DI and TI) mean applying ETF combined with PGD, MI, DI or TI to the shallow layers.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Autoattack	0.00	0.20	0.00	0.00	0.00	0.10	0.00	0.04
Deep-PGD	61.14	63.05	65.78	62.31	34.50	68.17	56.65	58.8
Shallow-PGD	46.55	49.13	56.78	58.34	28.50	55.82	37.94	47.58
ETF-PGD	41.76	46.74	48.55	50.79	24.68	53.11	32.65	42.61
Deep*-PGD	16.23	36.71	25.36	24.62	18.16	31.42	13.34	23.69

More Validation Images. Besides the widely used setting on the number of samples, i.e., 1,000 images, we also evaluate different methods using more samples, i.e., 5,000 images, and report the results in Table 8. The conclusion drawn from Table 8 is consistent with that drawn from Table 1, e.g., EFT outperforms "shallow" attack methods, demonstrating that ETF can generate powerful adversarial examples under various scenarios.

Table 8: The accuracy of 7 normally trained target models evaluated on 5,000 adversarial examples generated by lightweight black-box attacks or existing black-box attacks, under $\epsilon \leq 0.1$. The Shallow-(PGD, MI, DI, TI) mean applying PGD, MI, DI and TI to the shallow layers of the model. Deep-(PGD, MI, DI and TI) mean applying PGD, MI, DI and TI to the model's output. EFT-(PGD, MI, DI and TI) mean applying ETF combined with PGD, MI, DI or TI to the shallow layers. Auto-attack[23] is used for testing the robustness of the target models, so it adopts the white-box setting to mount the target models.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Autoattack[9]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deep-PGD	55.86	56.08	64.48	65.44	35.92	63.54	51.10	56.06
Deep-MI	38.02	44.70	52.56	52.98	13.22	49.74	28.92	40.02
Deep-DI	51.32	51.10	61.44	61.60	33.34	60.36	47.70	52.41
Deep-TI	55.00	54.94	64.60	64.48	36.80	63.86	51.50	55.88
Shallow-PGD	19.42	25.12	31.04	31.70	9.28	29.16	16.64	23.19
Shallow-MI	22.47	28.14	34.69	35.76	11.42	31.65	17.13	25.89
Shallow-DI	19.68	24.62	30.26	32.17	10.02	28.24	16.08	23.01
Shallow-TI	20.40	23.96	29.00	31.04	9.82	28.26	17.08	22.79
ETF-PGD	13.56	17.66	23.68	24.60	**4.54**	20.68	9.42	16.31
ETF-MI	15.94	20.32	26.28	26.74	5.52	22.72	9.70	18.17
ETF-DI	**13.16**	25.72	22.32	22.76	4.68	19.84	**8.58**	15.29
ETF-TI	13.30	**14.60**	**20.48**	**22.38**	5.22	**19.06**	9.50	**14.93**
Deep*-PGD	12.43	28.15	16.54	12.61	7.09	13.33	9.64	14.25
Deep*-MI	11.77	25.14	18.10	13.72	4.26	14.61	8.30	13.70
Deep*-DI	7.61	18.17	8.23	9.90	6.66	9.72	7.91	9.74
Deep*-TI	9.55	23.48	13.51	10.63	6.46	10.92	9.55	12.01

ℓ_2-norm Perturbation. We mainly conduct experiments with ℓ_∞ perturbation since it is widely adopted in many previous works [58, 25, 12]. To further demonstrate the power of our ETF, we further evaluate different methods using ℓ_2-norm perturbation. The results are reported in Table 9, which further demonstrate the effectiveness of our proposal. Considering that ℓ_1 and ℓ_0 perturbations require careful design [54, 4], it is beyond the scope of this work, so we leave it as our future work.

Architecture Selection. We further exploit whether the architecture of surrogate models have significant impact on the performance of ETF. Specifically, we instantiate the shallow layers with different model architectures containing ResNet [21], VGG [53], and SENet [22]. The results are reported in Table 10, demonstrating that our EFT is powerful across various model architectures.

Heavy Data Augmentation. We follow the empirical conclusion suggested in [1], where heavy data augmentation is vital for training appropriate shallow models. Because appropriate shallow models are necessary for mounting lightweight black-box attacks, data augmentation plays a crucial role and is heavily used in our experiments. This is supported by results shown in Table 11, where we report the performance of lightweight black-box attacks with and without data augmentation.

Capacity to Evade Adversarial Detectors. Adversarial Detection [40, 33, 59] aims to distinguish adversarial examples from natural examples, which is also an effective way to test the robustness of adversarial attacks. Therefore, we further exploit the capacity of ETF in evading adversarial example detectors. Specifically, we employ a detection method [40, 33] to detect adversarial examples generated by different attack methods, e.g., FGSM [19], PGD [42], BIM [11], and ETF. All settings are the same as that used in the paper, and the results are reported in Table 12. We can see that ETF performs better than the baselines, i.e., having a high probability of evading detection methods.

Table 9: The classification accuracy evaluation on ℓ_2-norm attacks. The experiment is conducted on the ImageNet validation. Following the previous work[25] about ℓ_2-norm attacks, the maximum disturbance ε is set to 16 $\sqrt[2]{N}$ where N is the dimension of input to attacks.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Autoattack[9]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deep-PGD	37.73±0.31	42.75±0.34	51.04±0.77	51.96±0.62	17.48±0.34	50.61±0.49	31.07±0.55	40.38±0.54
Deep-MI	40.40±0.44	45.02±0.51	54.53±0.46	54.13±0.53	17.59±0.47	53.22±0.63	32.47±0.41	42.48±0.55
Deep-DI	38.73±0.53	38.63±0.49	50.34±0.35	48.79±0.48	17.66±0.43	47.53±0.57	27.34±0.33	38.43±0.47
Deep-TI	37.89±0.23	37.86±0.38	46.52±0.46	45.62±0.31	18.54±0.44	46.44±0.37	30.75±0.52	37.66±0.46
Shallow-PGD	25.74±0.64	31.51±0.56	44.96±0.54	43.72±0.55	8.58±0.51	40.62±0.24	18.73±0.48	30.55±0.55
Shallow-MI	37.46±0.94	42.28±0.87	51.56±0.79	50.77±0.63	16.58±0.67	52.06±0.86	28.02±0.74	39.82±0.67
Shallow-DI	28.75±0.55	28.36±0.64	38.11±0.49	40.23±0.41	15.54±0.56	34.42±0.75	24.08±0.77	29.93±0.66
Shallow-TI	30.28±0.36	31.55±0.40	37.69±0.39	38.44±0.48	14.52±0.48	35.26±0.27	23.54±0.19	30.18±0.38
ETF-PGD	**22.16±0.54**	27.03±0.36	34.87±0.48	37.94±0.59	**11.28±0.37**	29.63±0.41	**16.17±0.46**	25.58±0.28
ETF-MI	32.76±0.95	33.05±0.87	45.91±0.91	44.22±0.88	14.38±0.76	41.54±0.78	20.76±0.69	33.23±0.74
ETF-DI	23.71±0.46	**23.45±0.55**	**33.29±0.56**	**34.25±0.49**	12.49±0.34	**29.23±0.24**	18.54±0.48	**24.99±0.53**
ETF-TI	25.23±0.37	25.73±0.68	34.15±0.73	37.34±0.43	12.56±0.66	30.07±0.56	21.53±0.45	26.65±0.69
Deep*-PGD	7.65±0.42	22.88±0.34	11.44±0.12	11.23±0.44	4.56±0.71	9.69±0.78	8.03±0.46	10.78±0.45
Deep*-MI	11.26±0.65	26.08±0.92	17.47±0.34	15.73±0.56	4.78±0.48	14.52±0.41	8.58±0.88	14.06±0.57
Deep*-DI	1.04±0.34	11.04±0.54	1.68±0.48	1.39±0.51	0.77±0.32	3.01±0.29	0.56±0.41	2.78±0.42
Deep*-TI	5.56±0.44	18.09±0.36	9.94±0.43	10.42±0.37	3.23±0.74	8.27±0.43	6.54±0.43	8.86±0.49

Table 10: Model accuracy under ETF attack with different architectures, containing SENet, VGG11, and ResNet18.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
SENet [22]	23.44	28.42	35.07	31.64	6.73	28.19	11.80	23.61
VGG11 [53]	18.20	22.65	27.24	26.33	**6.47**	23.16	12.69	19.53
Resnet [21]	**14.11**	**20.22**	**24.20**	**24.74**	6.96	**20.73**	**10.66**	**17.37**

Table 11: The impact of augmentation to ETF attacks. "No-Aug" means the effect of the attack on the ETF using the surrogate model without augmentation for training. This experiment is conducted on the ImageNet validation. The best results are in bold.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
No-Aug	34.58	39.17	46.25	50.06	10.42	45.10	22.92	35.50
Aug	**14.11**	**20.22**	**24.20**	**24.74**	**6.96**	**20.73**	**10.66**	**17.37**

Table 12: Performance of adversarial detection against four attacks, metric to evaluate the detection performance can be found in [34, 41].

Mahalanobis [34]					
Method	TNR	AUROC	DTACC	AUIN	AUOUT
BIM [32]	99.99	99.99	99.86	99.86	99.71
FGSM [19]	98.89	99.88	98.89	99.66	99.24
Deep*-PGD [42]	97.22	99.58	97.92	99.64	99.05
ETF	**96.67**	**98.73**	**96.94**	**98.75**	**97.98**
LID[41]					
Method	TNR	AUROC	DTACC	AUIN	AUOUT
Deep*-BIM [32]	99.99	**98.81**	98.33	99.77	99.33
Deep*-FGSM [19]	99.99	99.99	99.99	99.72	99.44
Deep*-PGD [42]	99.99	99.99	99.99	99.86	99.72
ETF	**97.78**	99.58	**97.22**	**99.51**	**98.68**

D Results on CIFAR10

We conduct the experiments on the CIFAR10 dataset, see Table 13, and evaluate the robustness of models downloaded from RobustBench [8], see Table 14. The conclusion drawn from Table 13 and Table 14 is consistent with that drawn from Table 1 evaluating on ImageNet dataset.

Table 13: Evaluate the performances of different attacks on CIFAR10. Here, experiments of "Deep-, Shallow-, ETF-" are conducted in the no-box threat model. "Deep*" means the black-box setting where the surrogate models are trained on the training data the same as the seven target models. "PGD [42], MI [11], DI [58], TI [12]" is applied to the different settings and methods. Auto-attack[23] is used for testing the robustness of the target models, so it adopts the white-box setting to mount the seven target model. $\varepsilon \leq 0.1$ in ℓ_∞-norm.

Model	VGG19[53]	RN56[21]	MobileNet[51]	ShuffleNet[22]	Avg
clean	93.91	94.37	93.72	92.98	93.74
Auto-attack [9]	0.00	0.00	0.00	0.00	0.00
Deep-PGD	59.45 ±0.34	57.58 ±0.46	45.21 ±0.27	52.32 ±0.37	53.64 ±0.78
Deep-MI	53.44 ±0.75	52.17 ±0.65	44.25 ±0.34	49.80 ±0.35	49.91 ±0.58
Deep-DI	60.24 ±0.19	58.63 ±0.34	47.67 ±0.31	54.34 ±0.62	55.22 ±0.52
Deep-TI	64.51 ±0.38	59.85 ±0.60	48.80 ±0.59	56.88 ±0.44	57.51 ±0.42
Shallow-PGD	27.17 ±0.74	31.06 ±0.55	22.83 ±0.66	28.14 ±0.76	27.30 ±0.81
Shallow-MI	32.43 ±0.98	36.42 ±1.01	31.84 ±0.79	30.76 ±0.94	32.86 ±0.94
Shallow-DI	25.65 ±0.56	30.27 ±0.51	22.61 ±0.38	27.22 ±0.55	26.43 ±0.45
Shallow-TI	28.66 ±0.45	31.35 ±0.33	27.20 ±0.44	29.48 ±0.63	29.17 ±0.56
ETF-PGD	21.27 ±0.27	25.85 ±0.84	**20.03** ±0.65	22.37 ±0.44	22.38 ±0.53
ETF-MI	**20.75** ±0.55	**24.36** ±0.35	20.51 ±0.34	**19.68** ±0.23	**21.32** ±0.42
ETF-DI	21.37 ±0.37	26.46 ±0.27	21.11 ±0.69	23.14 ±0.36	23.02 ±0.55
ETF-TI	25.48 ±0.41	30.26 ±0.23	23.37 ±0.51	26.34 ±0.25	26.36 ±0.39
Deep*-PGD	4.63 ±0.54	0.81 ±0.74	3.79 ±0.28	3.21 ±0.32	3.11 ±0.47
Deep*-MI	4.72 ±0.20	0.96 ±0.36	4.36 ±0.12	3.78 ±0.25	3.45 ±0.33
Deep*-DI	4.63 ±0.17	0.81 ±0.67	2.38 ±0.53	3.34 ±0.43	2.79 ±0.47
Deep*-TI	4.66 ±0.18	0.84 ±0.25	3.78 ±0.46	3.67 ±0.31	3.23 ±0.32

Table 14: The attacks on the most robust models from CIFAR10 RobustBench. The robustness model is trained by the different adversarial defense method,$\varepsilon \leq 0.1$ in ℓ_∞-norm.

Model	Gowal2021 [20]	Kang2021 [29]	Pang2022 [44]	Sehwag2021 [52]	Avg
clean	89.00	92.00	87.50	86.50	88.75
Auto-attack [9]	0.00	0.00	0.00	0.00	0.00
ETF-PGD	72.01	72.86	72.50	67.44	71.20
Deep*-PGD	83.53	88.06	83.17	79.44	83.55

E Target Model Approximation Assumption

Taking the first layer as an example, let w_t^1 and w^1 stand for the parameters of the target and surrogate models, respectively. In many practical scenarios, w_t^1 and w^1 usually have different dimensions, leading to intractable parameters' discrepancy alleviation. Fortunately, we can find an appropriate low-rank approximation for parameters of deep neural networks [10, 64, 30]. Specifically, we can approximate either w_t^1 or w^1 to make these two matrices have the same dimensions, so we can consider that the dimensions of the two models are the same. Consequently, we can find a transformation matrix A such that the approximation error is minimized, i.e., $A = \arg\min_{\tilde{A}} |w_t^1 - w^1 - w^1\tilde{A}|_F$, where $|\cdot|_F$ is the Frobenius norm. In this paper, we assume the approximation error is infinitesimal, i.e., $|w_t^1 - w^1 - w^1 A|_F = 0$. Then, we leverage w^1 and A to represent the target model, i.e., $w_t^1 = w^1 + w^1 A$.

F Perturbation in Different Space

In the no-box setting, performing the min-max strategy in the feature space is more appropriate than the weight space optimization [57] for the no-box threat model. This is because we know which perturbations are preferred in the feature space, e.g., towards features of guide images, but we have no idea about which perturbations are preferred in the weight space, i.e., no "guide models", which is supported by our experiments, see Table 15.

Table 15: The model accuracy under ETF attacks, where "Feature_space" means the feature space perturbation and "Weight_space" represents the min-max strategy in the weight space [57]. This experiment is conducted on the ImageNet validation. The best results are in bold.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Weight-space	29.43	32.44	40.11	41.88	10.12	35.41	19.27	29.81
Feature-space	**14.11**	**20.22**	**24.20**	**24.74**	**6.96**	**20.73**	**10.66**	**17.37**

G Different Self-supervised Learning Approach

It is straightforward that exploring different strategies to train the shallow model is exciting for further improvement of the performance of lightweight black-box attacks, as shallow layers play an important role in lightweight black-box attacks. Thus, we generate adversarial examples using EFT with shallow layers trained with a rotation prediction task [17] and report the results in Table 16. We can see that shallow layers trained with the rotation prediction task is slightly worse than using the contrastive strategy, but the performance can also reduce the model accuracy significantly.

H Social Impact

The motivation of this work is to provide an approach to evaluate the adversarial robustness in a more practical scenario, the no-box setting. Defenses can be assessed with fewer constraints

Table 16: The model accuracy under ETF attacks, where "rotation" means that the shallow layers are trained using the rotation task and "classification" represents that shallow layers are trained with the classification task. This experiment is conducted on the ImageNet validation. The best results are in bold.

Model	VGG19 [53]	Inception v3[55]	RN152 [21]	DenseNet [23]	SENet [22]	WRN [61]	MobileNet v3[51]	Average
Clean	67.43	64.36	74.21	73.34	51.28	73.22	65.06	66.99
Rotation [17]	19.07	21.79	27.30	28.85	7.66	23.94	12.51	20.16
Classification	**14.11**	**20.22**	**24.20**	**24.74**	**6.96**	**20.73**	**10.66**	**17.37**

through lightweight black-box attacks, i.e., without accessing training samples and any queries. We can develop defensive models robust against lightweight black-box attacks and attack algorithms to mislead deployed models. We believe the development of lightweight black-box attacks can help better access the robustness of deployed models and hope the proposed ETF can promote the development of corresponding defense methods.