Learning Transferable Adversarial Examples via Ghost Networks

Yingwei Li[1] Song Bai[2] Yuyin Zhou[1] Cihang Xie[1] Zhishuai Zhang[1] Alan Yuille[1]
[1]Johns Hopkins University [2]University of Oxford
yingwei.li@jhu.edu, {songbai.site, zhouyuyiner, cihangxie306, zhshuai.zhang, alan.l.yuille}@gmail.com

Abstract

Recent development of adversarial attacks has proven that ensemble-based methods outperform traditional, non-ensemble ones in black-box attack. However, these methods generally require a family of diverse models, and ensembling them together afterward, both of which are computationally expensive. In this paper, we propose Ghost Networks to generate transferable adversarial examples. The critical principle of ghost networks is to apply feature-level perturbations to an existing model to potentially create a huge set of diverse models. After that, models are subsequently fused by longitudinal ensemble. Compared to traditional ensemble methods, our work significantly improves the transferability of adversarial examples, and in the meantime, requires almost no extra time and memory consumption. By reproducing the NeurIPS 2017 adversarial competition, our method outperforms the No.1 attack submission by a large margin, demonstrating its effectiveness and efficiency. The code will be publicly available.

1. Introduction

In recent years, Convolutional Neural Networks (CNNs) have advanced performance in various vision tasks. However, it has been observed that attacking CNNs by adding human imperceptible perturbations to input images can cause networks to make incorrect predictions [24]. These perturbed images are termed as adversarial examples.

Two attack settings are later developed, *i.e.*, white-box attack and black-box attack. In white-box attack, attackers can fully access the model [4, 13]. By contrast, in black-box attack, the target model is invisible to attackers. A typical solution is to generate adversarial examples with strong transferability (the same input remain malicious for different models [24]). Meanwhile, defense techniques based on randomization [15, 29], input transformation [8], and training [17, 25] are also studied.

Focusing on the transferability, many attempts have been made, such as attacking a substitute model [19] or an ensemble of multiple substitute models [6, 16, 31]. In particular, the ensemble-based attacks obtain much better performance than the non-ensemble ones, and thus has attracted many attentions. Almost all top-ranked entries in competitions use ensemble-based attacks [14].

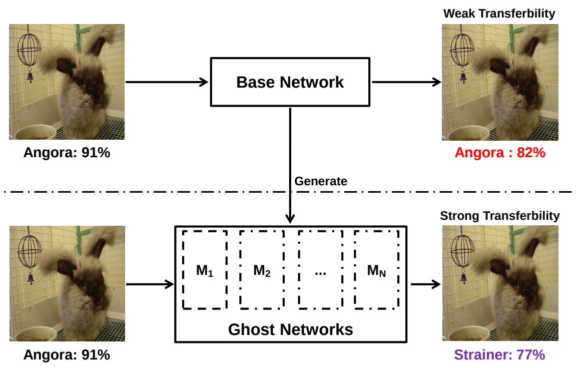


Figure 1. An illustration of the capacity of the proposed ghost networks in learning transferable adversarial examples.

However, the ensemble-based attacks suffer from expensive computational overhead, making it difficult to generate transferable adversarial examples efficiently. First, in order to acquire good (*i.e.*, low test error) and diverse (*i.e.*, converge at different local minima) models, people usually independently train them from scratch. Second, to leverage their complementarity, existing methods adopt an intensive aggregation way to fuse the outputs of those networks (*e.g.*, logits). These two points limit the number of ensembled models in practice, *e.g.*, attacking methods in competitions (like [14]) generally ensemble at most only ten networks.

How to generate strong transferable adversarial examples without additional cost remains a challenging task. [1, 20, 28] suggest that re-training networks can achieve high transferability. [2, 5, 7] propose query-based methods to attack black-box model without substitute models, which require extensive information from the target model. In conclusion, acquiring and integrating information from various models is the key to achieving better transferability. However, most works are inefficient and inadequate to learn adversarial examples with strong transferability.

In this paper, we propose a highly efficient alternative called Ghost Networks to address this issue. As shown in Fig. 1, the basic principle is to generate a vast number of virtual models built on a base network (a network trained from scratch). The word "virtual" means that these ghost networks are not stored or trained. Instead, they are generated by imposing erosion on certain intermediate structures of the base network on-the-fly. However, with an increasing number of models we have, a standard ensemble [16] would be problematic owing to its complexity. Accordingly, we propose Longitudinal Ensemble, a specific method for ghost networks, which conducts an implicit ensemble during attack iterations. Consequently, adversarial examples can be generated without sacrificing computational efficiency.

Extensive experimental results demonstrate our method improves the transferability of adversarial examples, acting as a computationally cheap plug-in. In particular, by reproducing NeurIPS 2017 adversarial competition [14], our work outperforms the No.1 attack submission by a large margin, demonstrating its effectiveness and efficiency.

2. Backgrounds

This section introduces two iteration-based methods, Iterative Fast Gradient Sign Method (I-FGSM) [13] and Momentum I-FGSM (MI-FGSM) [6]. **I-FGSM** initializes an adversarial example $I_0^{\text{adv}} = I$ and iteratively updates it by

$$I_{n+1}^{\text{adv}} = \text{Clip}_I^{\epsilon}\left\{I_n^{\text{adv}} + \alpha \text{sign}\left(\nabla_I L(I_n^{\text{adv}}, y^{\text{true}}; \theta)\right)\right\}, \tag{1}$$

where L is the loss function of a network with parameter θ. The clip function Clip_I^{ϵ} ensures the generated adversarial example within the ϵ-ball of the original image I with ground-truth y^{true}. n is the iteration number, and α is the step size. Based on I-FGSM, **MI-FGSM** integrates the momentum decay term μ into the attack process to stabilize update directions and escape from poor local maxima.

3. Ghost Networks

The goal of this work is to learn transferable adversarial examples. Given an image I, we want to find an adversarial example $I^{\text{adv}} = I + r$, which is visually similar to I after adding adversarial noise $\|r\|_\infty < \epsilon$ but fools the classifier.

Without additional cost, we generate a huge number of ghost networks from a single trained model for later attack by applying feature-level perturbations to non-residual and residual based networks in Sec. 3.1 and Sec. 3.2, respectively. These ghost networks are efficiently emsembled by our customized method, logitudinal ensemble, see Sec. 3.3.

3.1. Dropout Erosion

Revisit Dropout. Dropout [21] is one of the most popular techniques in deep learning. Let x_l be the activation in the l^{th} layer, at the training time, the output y_l after a dropout layer can be defined as

$$y_l = r_l * x_l, \quad r_l \sim \text{Bernoulli}(p), \tag{2}$$

where $*$ denotes an element-wise product and Bernoulli(p) denotes the Bernoulli distribution with the probability p of elements in r_l being 1.

Perturb Dropout. Dropout provides an efficient way of approximately combining different neural network architectures and thereby prevents overfitting. Inspired by this, we propose to generate ghost networks by inserting the dropout layer. To make ghost networks as diverse as possible, we **densely** apply dropout to every block throughout the base network, *rather than simply enable default dropout layers [3]*. Form our preliminary experiments, the latter cannot provide transferability. Therefore, diversity is not limited to high-level features but applied to all feature levels.

Let f_l be the function between the i^{th} and $(i+1)^{\text{th}}$ layer, *i.e.*, $x_{l+1} = f_l(x_l)$, then the output of f_l after applying dropout erosion, *i.e.*, $g_l(x_l)$, is

$$g_l(x_l) = f_l\left(\frac{r_l * x_l}{1-\Lambda}\right), \quad r_l \sim \text{Bernoulli}(1-\Lambda), \tag{3}$$

where $\Lambda = 1 - p$, and p has the same meaning as in Eq. (2), indicating the probability that x_l is preserved. To keep the expected input of $f_l(\cdot)$ consistent after erosion, the activation of x_l should be divided by $1-\Lambda$.

During the inference, the output feature after $(L-1)$-th dropout layer $(L > l)$ is

$$x_L = g_{L-1} \circ g_{L-2} \circ g_{L-3} \circ \cdots \circ g_l(x_l). \tag{4}$$

$\circ$ denotes composite function, *i.e.*, $g \circ f(x) = g\left(f(x)\right)$.

By combining Eq. (3) and Eq. (4), we observe that when $\Lambda = 0$ (means $p = 1$), all elements in r_l equal to 1. In this case, we do not impose any perturbations to the base network. When Λ gradually increases to 1 (p decreases to 0), the ratio of elements dropped out is Λ. In other words, $(1-\Lambda)$ of elements can be back-propagated. Hence, larger Λ implies a heavier erosion on the base network. Therefore, we define Λ to be the *magnitude of erosion*.

When perturbing dropout layers, the gradient in back-propagation can be written as

$$\frac{\partial x_L}{\partial x_l} = \prod_{l \le i < L}\left(\frac{r_i}{1-\Lambda} * \frac{\partial}{\partial x_i} f_i\left(\frac{r_i * x_i}{1-\Lambda}\right)\right). \tag{5}$$

As shown in Eq. (5), deeper networks with larger L are influenced more easily according to the product rule. Sec. 4.2 will experimentally analyze the impact of Λ.

Generate Ghost Network. The generation of ghost networks via perturbing dropout layer proceeds in three steps: 1) randomly sample a parameter set from the Bernoulli distribution $r = \{r_1, r_2 ..., r_l, ..., r_L\}$; 2) apply Eq. (3) to the base network with the parameter set r and get the perturbed network; 3) repeat step 1) and 2) to independently sample r for N times and obtain a pool of ghost networks $M = \{M_1, M_2, ..., M_N\}$ which can be used for attacks.

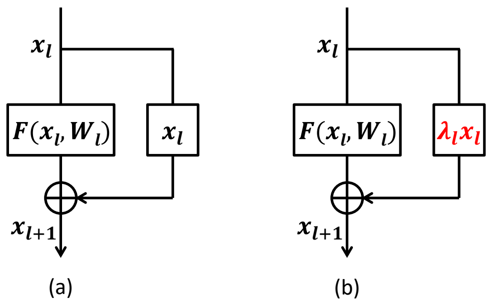


Figure 2. An illustration of skip connection (a, Eq. (6)) and skip connection erosion (b, Eq. (7)).

3.2. Skip Connection Erosion

Revisit Skip Connection. [9] propose skip connections in CNNs, which makes it feasible to train very deep neural networks. The residual block is defined by

$$x_{l+1} = x_l + F(x_l, W_l), \tag{6}$$

where x_l and x_{l+1} are the input and output of the l-th block. $F(\cdot)$ denotes the residual function with the weights W_l.

Perturb Skip Connection. We propose to perturb skip connections to generate ghost networks. Specifically, the network weights are first learned using identity skip connections, then switched to the randomized skip connection (see Fig. 2). To this end, we apply randomized modulating scalar λ_l to the l-th residual block by

$$x_{l+1} = \lambda_l x_l + F(x_l, W_l), \tag{7}$$

where λ_l is drawn from the uniform distribution $U[1 - \Lambda, 1 + \Lambda]$. One may have observed several similar formulations on skip connection to improve the classification performance, *e.g.*, the gated inference in [26] and lesion study in [27]. However, our work focuses on attacking the model with a randomized perturbation on skip connection, *i.e.*, the model is not trained via Eq. (7).

During inference, the output after $(L-1)$th layer is

$$x_L = (\prod_{i=l}^{L-1} \lambda_i) x_l + \sum_{i=l}^{L-1} (\prod_{j=i+1}^{L-1} \lambda_j) F(x_i, W_i). \tag{8}$$

The gradient in back-propagation is then written as

$$\frac{\partial x_L}{\partial x_l} = (\prod_{i=l}^{L-1} \lambda_i) + \sum_{i=l}^{L-1} (\prod_{j=i+1}^{L-1} \lambda_j) \frac{\partial F(x_i, W_i)}{\partial x_l}. \tag{9}$$

Similar to the analysis in Sec. 3.1, we conclude from Eq. (8) and Eq. (9) that a larger Λ will have a greater influence on the base network and deeper networks are easily influenced.

Generate Ghost Network. This step is similar to that via perturbing the dropout layer. The only difference is we need to sample a set of modulating scalars $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_L\}$.

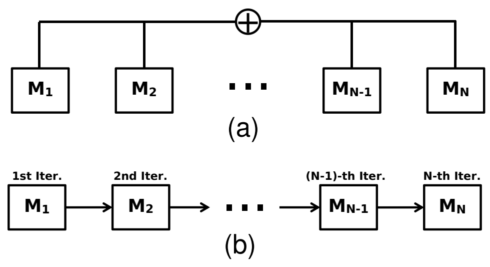


Figure 3. The illustration of the standard ensemble (a) and the proposed longitudinal ensemble (b).

3.3. Longitudinal Ensemble

The existing iteration-based ensemble-attack approach requires averaging the outputs (*e.g.*, logits) of different networks [16]. However, this standard ensemble is inefficient when we can readily obtain a huge candidate pool of qualified models by using Ghost Networks.

To remedy this, we propose longitudinal ensemble, a specially designed fusion method for Ghost Networks, which constructs an implicit ensemble of the ghost networks by randomizing the perturbations during iterations of adversarial attacks. Suppose we have a base model B, which can generate a pool of networks $M = \{M_1, M_2, ..., M_N\}$, where N is the model number. The critical step of longitudinal ensemble is that at the j-th iteration, we attack the ghost network M_j only. In comparison, for each iteration, standard ensemble methods require fusing gradients of all the models in the model pool M, leading to high computational cost. We illustrate the difference between two ensemble methods in Fig. 3.

The longitudinal ensemble shares the same prior as [16] that if an adversarial example is generated by attacking multiple networks, it is more likely to transfer to other networks. However, longitudinal ensemble method removes duplicated computations by sampling only one model from the pool rather than using all models in each iteration.

There are three noteworthy comments here. First, ghost networks are never stored or trained, reducing both additional time and space cost. Second, it is evident from Fig. 3 that attackers can combine [16] and longitudinal ensemble of ghost networks. Finally, it is easy to extend longitudinal ensemble to multi-model attack by treating each base model as a branch (details are in experimental evaluations).

4. Experiments

In this section, we give a comprehensive experimental evaluation of the proposed Ghost Networks. In order to distinguish models trained from scratch and the ghost networks, we call the former one base network/model in the rest of this paper. We provide additional experimental results in the appendix.

Methods	Settings			Res-50		Res-101		Res-152		IncRes-v2		Inc-v3		Inc-v4		CC
	MT	#S	#L	I-	MI-	I-	MI-	I-	MI-	I-	MI-	I-	MI-	I-	MI-	
Exp. S1	Base	1	1	16.3	29.4	17.8	31.3	16.7	29.6	8.3	20.0	5.3	13.7	7.3	18.4	1
Exp. S2	Ghost	1	1	8.4	17.4	6.1	19.9	6.4	17.9	5.7	15.2	1.7	5.6	1.9	7.2	1
Exp. S3	Ghost	1	10	**23.4**	**39.4**	**23.7**	**40.1**	**21.1**	**38.0**	**11.2**	**26.8**	**6.3**	**17.6**	**10.0**	**22.4**	1
Exp. S4	Ghost	10	1	28.8	44.5	29.9	43.2	25.6	41.9	13.1	30.4	6.3	17.9	9.3	25.6	10
Exp. S5	Ghost	10	10	**35.9**	**50.6**	**35.9**	**51.4**	**60.1**	**64.9**	**14.6**	**33.3**	**12.3**	**28.3**	**19.4**	**37.4**	10

Table 1. The average black-box attack rate (%) comparison of different methods over two iterative methods, "I-" for I-FGSM and "MI-" for MI-FGSM. MT denotes model type, and #S (or #L) denotes the number of models for standard (or longitudinal) ensemble in each iteration (branch). CC denotes the computational cost, which is a relative value and we set the CC of Exp. S1 as 1. We marked all highest attack success rate under the same CC in **boldface**.

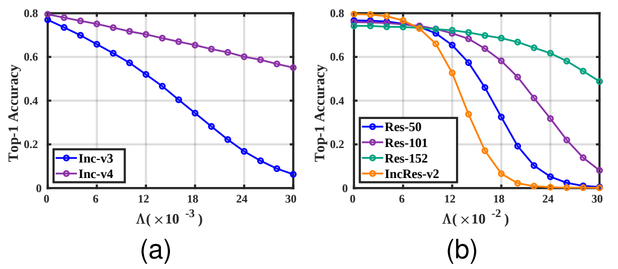


Figure 4. The top-1 accuracy of dropout erosion (a) and skip connection erosion (b) with different magnitude Λ.

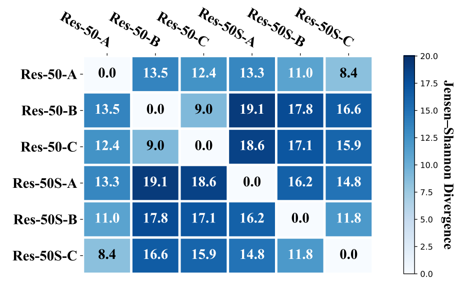


Figure 5. The illustration of the mean diversity $(\times 10^{-2})$ of any pair of networks over the ILSVRC 2012 validation set.The higher value indicates larger diversity.

4.1. Experimental Setup

Base Networks. 9 base models are used in our experiments, including 6 normally trained models, *i.e.*, Resnet v2-{50, 101, 152} (Res-{50, 101, 152}) [10], Inception {v3, v4} (Inc-{v3, v4}) [22, 23] and Inception Resnet v2 (IncRes-v2) [23], and 3 adversarially-trained models [25] , *i.e.*, Inc-v3$_{\text{ens3}}$, Inc-v3$_{\text{ens4}}$ and IncRes-v2$_{\text{ens}}$.

Datasets. Following [30], we select 5000 images from the ILSVRC 2012 validation set, which can be correctly classified by all the 9 base models.

Attacking Methods. We employ two iteration-based attack methods mentioned in Sec. 2 to evaluate the adversarial robustness, *i.e.*, I-FGSM and MI-FGSM.

Parameter Specification. If not specified otherwise, we follow the default settings in [13], *i.e.*, step size $\alpha = 1$ and the total iteration number $N = \min(\epsilon + 4, 1.25\epsilon)$. We set the maximum perturbation $\epsilon = 8$ ($N = 10$ in this case). For MI-FGSM, the decay factor μ is set to be 1 as in [6].

4.2. Analysis of Ghost Networks

In order to learn adversarial examples with good transferability, there are generally two requirements for the intrinsic models – models should have low test error rates and be diverse. We experiment with the whole ILSVRC 2012 validation set to show the ghost networks' qualification.

Descriptive Capacity. To measure the descriptive capacity of the ghost networks, we plot the relationship between the magnitude of erosion Λ and top-1 classification accuracy.

We apply dropout erosion in Sec. 3.1 to non-residual networks (Inc-{v3, v4}) and skip connection erosion in Sec. 3.2 to residual networks (Res-{50, 101, 152} and IncRes-v2). Fig. 4 (a) and (b) present the accuracy curves of the dropout and skip connection erosion, respectively.

The classification accuracies of different models are negatively correlated to the magnitude of erosion Λ as expected. By choosing the performance drop approximately equal to 10% as a threshold, we can determine the value of Λ individually for each network. Specifically, in our following experiments, Λs are 0.006, 0.012, 0.22, 0.16, 0.12 and 0.08 for Inc-{v3, v4}, Res-{50, 101 ,152}, and IncRes-v2 respectively. As emphasized throughout this paper, it is extremely cheap to generate a huge number of ghost networks that still preserve relative low error rates.

Model Diversity. To measure diversity, we use Res-50 as the backbone model. We denote the base Res-50 described in Sec. 4.1 as Res-50-A, and independently train two additional models with the same architecture, denoted as Res-50-{B, C}. Meanwhile, we apply skip connection erosion to Res-50-A, then obtain three ghost networks denoted as Res-50S-{A, B, C} respectively.

We employ the Jensen-Shannon Divergence (JSD) as the evaluation metric for model diversity. Concretely, we compute the pairwise similarity of the output probability distribution for each pair of networks as in [11]. Given an image, X and Y denotes the softmax outputs of two networks, then

$$\text{JSD}(X\|Y) = \frac{1}{2}D(X\|Z) + \frac{1}{2}D(Y\|Z), \quad (10)$$

where Z is the average of X and Y, *i.e.*, $Z = (X + Y)/2$. $D(\cdot)$ is the Kullback-Leibler divergence.

In Fig. 5, we report the averaged JSD for all pairs of networks over the ILSVRC 2012 validation set. As can be

Methods	Black-box Attack				White-box Attack			
	TsAIL	iyswim	Anil Thomas	Average	Inc-v3_adv	IncRes-v2_ens	Inc-v3	Average
No.1 Submission	13.60	43.20	43.90	33.57	94.40	93.00	**97.30**	94.90
No.1 Submission+**ours**	**14.80**	**52.28**	**51.68**	**39.59**	**97.62**	**96.00**	95.48	**96.37**

Table 2. The attack rate (%) comparison in the NeurIPS 2017 Adversarial Challenge.

drawn, the diversity among ghost networks is comparable or even more significant than independently trained networks.

Based on the analysis above, we can see that ghost networks can provide accurate yet diverse descriptions of the data manifold, which is beneficial to learn transferable adversarial examples as we will experimentally prove below.

4.3. Single-model Attack

Firstly, we evaluate the ghost networks in single-model attack, where attackers can only access one base model B trained from scratch. We design five experimental comparisons. The setting, performance and properties are shown in Table 1. The difference among five experiments is the type of model to attack, the number of models ensembled by standard ensemble [16] in each iteration, and by longitudinal ensemble in each branch of [16]. For example, Exp. S5 combines two ensemble methods, that is, we do a standard ensemble of 10 models for each iteration and a longitudinal ensemble of 10 models for each standard ensemble branch. Therefore, Exp. S5 intrinsically ensembles 100 models.

We attack 6 normally-trained networks and test on all the 9 networks (include 3 adversarially-trained networks). The attack rate is shown in Table 1. To save space, we report the average attack rate for black-box models. All the individual cases are shown in the appendix.

As can be drawn from Table 1, a single ghost network is worse than the base network (Exp. S2 *vs.* Exp. S1), due to the inferior descriptive power of ghost networks. However, by leveraging the longitudinal ensemble, our work achieves a higher attack rate in most settings (Exp. S3 *vs.* Exp. S1).

This observation firmly demonstrates the effectiveness of ghost networks in learning transferable adversarial examples. It should be mentioned that the computational cost of Exp. S3 remains the same as Exp. S1 since 1) the 10 ghost networks used in Exp. S3 are not trained but eroded from the base model and used on-the-fly, and 2) multiple ghost networks are fused via the longitudinal ensemble.

The proposed ghost networks can also be fused via the standard ensemble method, as shown in Exp. S4, and achieve a higher attack rate at the sacrifice of computational efficiency. This observation inspires us to combine two ensemble methods (Exp. S5). As we can see, Exp. S5 consistently beats all the compared methods in all the black-box settings. Of course, Exp. S5 is as computational expensive as Exp. S4. However, the overhead stems from the standard ensemble rather than longitudinal ensemble.

Note that in all the experiments presented in Table 1, we use only one individual base model. Even in the case of Exp. S3, all the to-be-fused models are ghost networks. However, the generated ghost networks are never stored or trained, meaning no extra space complexity. Therefore, the benefit of ghost networks is obvious. Especially when comparing Exp. S5 and Exp. S1, ghost networks can achieve a substantial improvement in black-box attack.

Based on the experimental results above, we arrive at a similar conclusion as [16]: the number of intrinsic models is essential to improve the transferability of adversarial examples. However, a different conclusion is that it is less necessary to train different models independently. Instead, ghost networks is a computationally cheap alternative enabling good performance. When the number of intrinsic models increases, the attack rate will increase.

4.4. NeurIPS 2017 Adversarial Challenge

Then, we evaluate our method in a benchmark test of the NeurIPS 2017 Adversarial Challenge [14], a typical multi-model attack setting. For evaluation, we use the top-3 defense submissions (black-box models), *i.e.*, TsAIL, iyswim and Anil Thomas, and three official baselines (white-box models), *i.e.*, Inc-v3$_{\text{adv}}$, IncRes-v2$_{\text{ens}}$ and Inc-v3.

Our settings are exactly the same with the No.1 attack submission [6], but replace each clean trained model to a ghost model in each attack iteration, which incurs almost neither extra computational cost nor additional memory usage. The results are summarized in Table 2. Consistent with previous experiments, we observe that by using ghost networks, the performance of the No. 1 submission can be significantly improved, especially with black-box attack. This suggests that our method generalize well to other defenses.

5. Conclusion

This paper focuses on learning transferable adversarial examples for adversarial attacks. We propose, for the first time, to exploit network erosion to generate a kind of virtual models called ghost networks. Ghost networks, together with the coupled longitudinal ensemble strategy, requiring almost no additional time and space consumption, is an effective tool to improve existing methods in learning transferable adversarial examples. Extensive experiments have firmly demonstrated the efficacy of ghost networks. Meanwhile, one can potentially **densely** erode other typical layers (*e.g.*, batch norm [12] and relu [18]) through a neural network. We suppose these methods could improve the transferability as well, and leave these issues as future work.

References

[1] Shumeet Baluja and Ian Fischer. Learning to attack: Adversarial transformation networks. In *AAAI*, 2018. 1

[2] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *ECCV*, 2018. 1

[3] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *AISec*, 2017. 2

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE S&P*, 2017. 1

[5] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017. 1

[6] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Xiaolin Hu, Jianguo Li, and Jun Zhu. Boosting adversarial attacks with momentum. In *CVPR*, 2018. 1, 2, 4, 5

[7] Chuan Guo, Jared S Frank, and Kilian Q Weinberger. Low frequency adversarial perturbation. *arXiv preprint arXiv:1809.08758*, 2018. 1

[8] Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering adversarial images using input transformations. In *ICLR*, 2018. 1

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 4

[11] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. In *ICLR*, 2017. 4

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 5

[13] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017. 1, 2, 4

[14] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, et al. Adversarial attacks and defences competition. *arXiv preprint arXiv:1804.00097*, 2018. 1, 2, 5

[15] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *ECCV*, 2018. 1

[16] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *ICLR*, 2017. 1, 2, 3, 5

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 1

[18] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 5

[19] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016. 1

[20] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *CVPR*, 2017. 1

[21] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014. 2

[22] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017. 4

[23] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 4

[24] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1

[25] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *ICLR*, 2018. 1, 4

[26] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *ECCV*, 2018. 3

[27] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *NIPS*, 2016. 3

[28] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. In *IJCAI*, 2018. 1

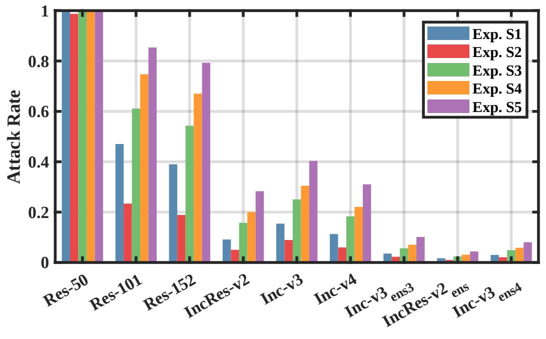
[29] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *ICLR*, 2018. 1

[30] Cihang Xie, Zhishuai Zhang, Jianyu Wang, Yuyin Zhou, Zhou Ren, and Alan Yuille. Improving transferability of adversarial examples with input diversity. *arXiv preprint arXiv:1803.06978*, 2018. 4

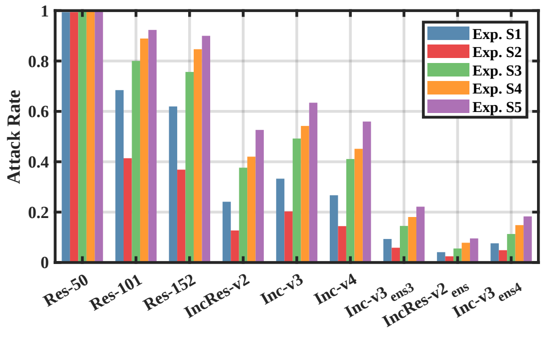
[31] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *ECCV*, 2018. 1

A. Detail Results

Due to the space limitation, the comparison of performance under the setting of single-model attack has to be simplified in the main manuscript. The goal of this section is to present a detailed comparison rather than the average values in the Table. 1. Therefore, in Fig. 6, Fig. 7, Fig.8, Fig.9, Fig. 10 and Fig. 11, we present the comparison of attack rates of adversarial examples generated by a single base model (Res-50, Res-101, Res-152, IncRes-v2, Inc-v3 and Inc-v4 for each figure).

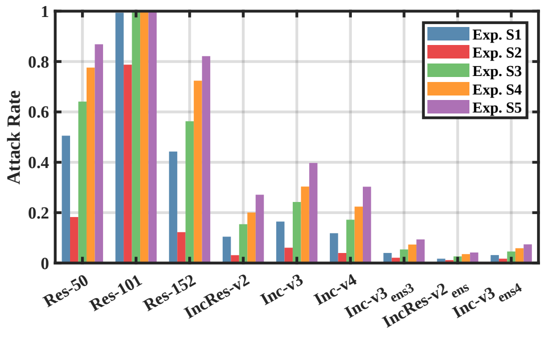


(a) I-FGSM

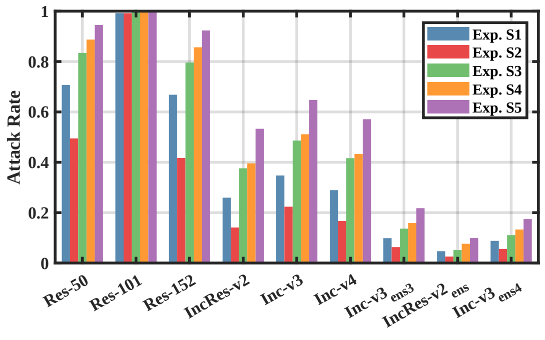


(b) MI-FGSM

Figure 6. The attack rate comparison when attacking Res-50, and testing on all the base models.

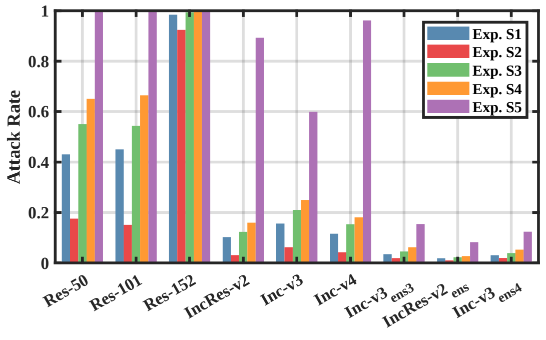


(a) I-FGSM

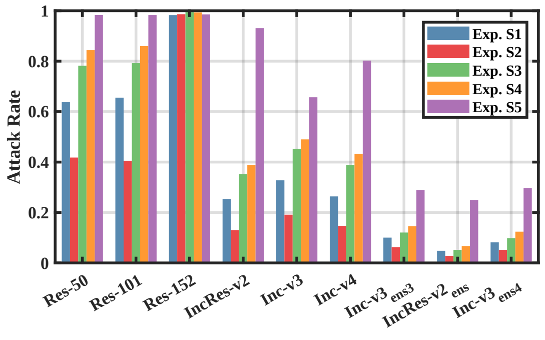


(b) MI-FGSM

Figure 7. The attack rate comparison when attacking Res-101, and testing on all the base models.

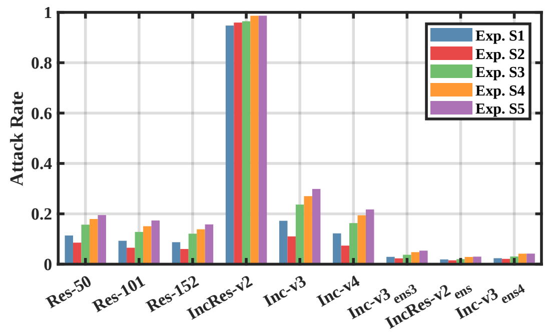


(a) I-FGSM

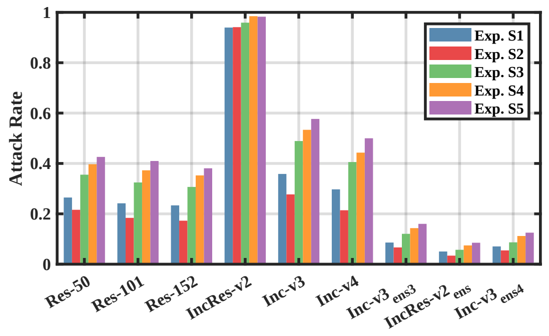


(b) MI-FGSM

Figure 8. The attack rate comparison when attacking Res-152, and testing on all the base models.

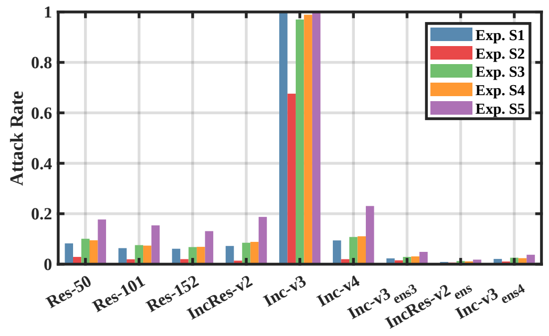


(a) I-FGSM

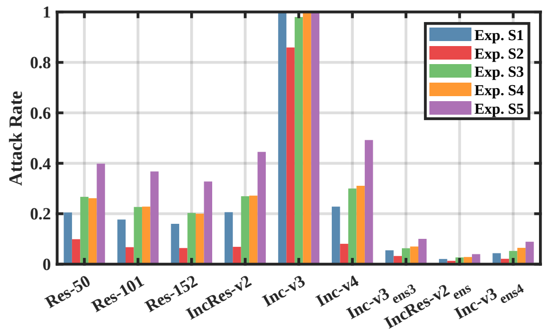


(b) MI-FGSM

Figure 9. The attack rate comparison when attacking IncRes-v2, and testing on all the base models.

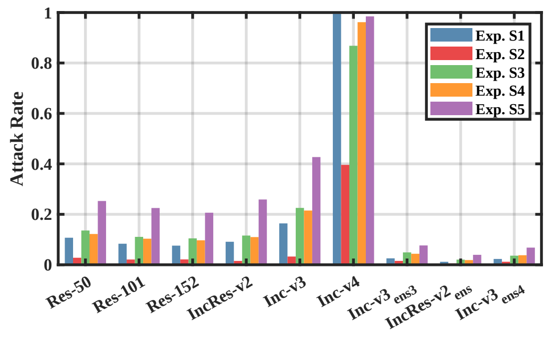


(a) I-FGSM

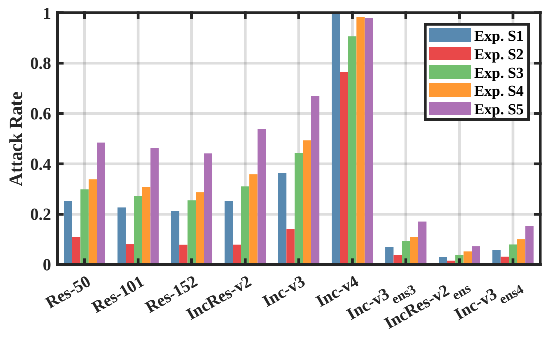


(b) MI-FGSM

Figure 10. The attack rate comparison when attacking Inc-v3, and testing on all the base models.



(a) I-FGSM



(b) MI-FGSM

Figure 11. The attack rate comparison when attacking Inc-v4 with, and testing on all the base models.