Efficient and Transferable Adversarial Examples from Bayesian Neural Networks

Abstract

An established way to improve the transferability of black-box evasion attacks is to craft the adversarial examples on an ensemble-based surrogate to increase diversity. We argue that transferability is fundamentally related to uncertainty. Based on a state-of-the-art Bayesian Deep Learning technique, we propose a new method to efficiently build a surrogate by sampling approximately from the posterior distribution of neural network weights, which represents the belief about the value of each parameter. Our extensive experiments on ImageNet and CIFAR-10 show that our approach improves the transfer rates of four state-of-the-art attacks significantly (up to 62.1 percentage points), in both intra-architecture and inter-architecture cases. On ImageNet, our approach can reach 94% of transfer rate while reducing training computations from 11.6 to 2.4 exaflops, compared to an ensemble of independently trained DNNs. Our vanilla surrogate achieves 87.5% of the time higher transferability than 3 test-time techniques designed for this purpose. Our work demonstrates that the way to train a surrogate has been overlooked, although it is an important element of transfer-based attacks. We are, therefore, the first to review the effectiveness of several training methods in increasing transferability. We provide new directions to better understand the transferability phenomenon and offer a simple but strong baseline for future work.

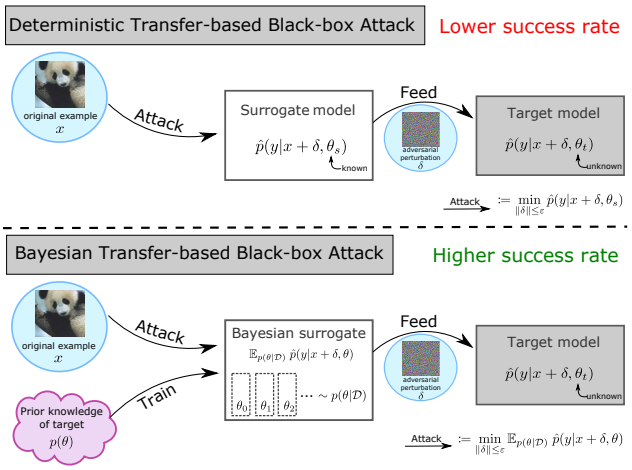


Figure 1: Illustration of the proposed approach.

1 INTRODUCTION

Deep Neural Networks (DNNs) have caught a lot of attention in recent years thanks to their capability to solve efficiently various tasks, especially in computer vision [Dargan et al., 2019]. However, a common pitfall of these models is that they are vulnerable to adversarial examples, i.e., misclassified examples that result from slightly altering a well-classified example at test time [Biggio et al., 2013, Szegedy et al., 2013]. This constitutes a critical security threat, as a malicious third party may exploit this property to enforce some desired outcome.

Such *adversarial attacks* have been primarily designed in white-box settings, where the attacker is assumed to have complete knowledge of the target DNN (incl. its weights). While studying such worst-case scenarios is essential for proper security assessment, in practice the attacker should have limited knowledge of the target model. In such a case, the adversarial attack is applied to a surrogate model, with the hope that the crafted adversarial examples *transfer to* (i.e., are also misclassified by) the target DNN.

Achieving transferability remains challenging, though. This is because adversarial attacks were designed to optimize the loss function of a specific model [Goodfellow et al., 2014, Kurakin et al., 2019], different from that of the target model. As a result, Liu et al. [2017] improved transferability by attacking an *ensemble* of architectures. The key intuition is

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

that adversarial examples that fool a diverse set of models are more likely to generalize. While ensemble-based attacks typically report significantly higher success rates than their single-model counterparts, their computational cost is prohibitive due to the necessity to independently train several surrogate models (to form a diverse ensemble).

In this paper, we analyse the unknown target model with a probabilistic eye, and relate transfer-based attacks to uncertainty. We propose a new method to improve the transferability of adversarial examples using approximate Bayesian inference to build a surrogate – and do so with less computation overhead compared to ensemble-based methods. Our approach, shown in Figure 1, leans upon recent results in Bayesian Deep Learning. More precisely, we train our surrogate with a cyclical variant of Stochastic Gradient Markov Chain Monte Carlo (i.e., *cSGLD* [Zhang et al., 2020]) to sample from the posterior distribution of neural network weights. We then perform efficient approximate Bayesian model averaging during the attack with minimal modifications of the attack algorithms.

We evaluate our approach on the CIFAR-10 and the ImageNet datasets (with five DNN architectures each), four adversarial attacks, and three test-time transformations. Overall, our results indicate that applying cSGLD significantly improves the success rate compared to training single DNNs and outperforms classical ensemble-based attacks in terms of computation cost. Deep Ensemble requires at least 2.51 times more flops to achieve the same success rates as cSGLD when the targeted architecture is known. This can represent, on ImageNet, a saving of 3.56 exaflops (2.36 vs 5.92). At the same computation budget, our method increases the intra-architecture transfer rates between 12.1 and 49.2 percentage points and the inter-architecture transfer rates between -2.3 and 62.1 percentage points at constant computation costs. cSGLD always raises the effectiveness of test-time techniques designed for transferability between 3.8 and 56.2 percentage points. Applied alone, it is more effective than these techniques applied to a single DNN in 105/120 cases.

To summarize, our contributions are:

- We relate uncertainty and transferability of adversarial examples with a Bayesian perspective. We conclude that the posterior predictive approximates an optimal surrogate model under some conditions.
- We propose the first method based on a Bayesian Deep Learning technique to generate transferable adversarial examples. Existing iterative attacks can be easily modified to perform approximate Bayesian model averaging at no additional computational cost.
- We pave the way for improving surrogates at train-time by evaluating six Bayesian and ensemble techniques. cSGLD is a strong competitor, though other techniques open promising avenues.
- We advocate the use of a new metric, T-DEE, to compare the effectiveness of transferability techniques with the strong baseline of Deep Ensemble.
- Our evaluation on CIFAR-10 and ImageNet reveals significant improvements over the single-DNN and Deep Ensemble baselines in diverse experimental settings. Our train-time method improves existing test-time techniques, and is better in most cases on a competitive basis. We open new ways to understand transferability.

2 BACKGROUND AND RELATED WORK

Adversarial attacks. We consider 4 gradient-based attacks, which aims to maximise the prediction loss $L(x, y, \theta)$ with a p-norm constraint: $\arg\max_{\|\delta\|_p \leq \varepsilon} L(x + \delta, y, \theta)$. FGSM [Goodfellow et al., 2014] is a L∞ single-step attack defined by $\delta_{\text{FGSM}} = \varepsilon\, \text{sign}(\nabla_x L(x, y, \theta))$. FGM is its L2 equivalent: $\delta_{\text{FGM}} = \varepsilon \frac{\nabla_x L(x,y,\theta)}{\|\nabla_x L(x,y,\theta)\|_2}$. The adversarial example is then clipped in $[0, 1]$. I-FGSM [Kurakin et al., 2019] applies iteratively FGSM with a small step-size α: $\delta_0 = 0$ and $\delta_{i+1} = \text{proj}_{B_\varepsilon}(\delta_i + \alpha\, \text{sign}(\nabla_x L(x, y, \theta)))$, where $\text{proj}_{B_\varepsilon}(\bullet)$ projects the perturbation in the Lp ball of radius ε. The L2 I-FGM attack is derived similarly. MI-FGSM attack [Dong et al., 2018] adds a momentum term with decay factor μ to the previous attack: $g_{i+1} = \mu\, g_t + \frac{\nabla_x L(x,y,\theta)}{\|\nabla_x L(x,y,\theta)\|_1}$ and $\delta_{i+1} = \text{proj}_{B_\varepsilon}(\delta_i + \alpha\, \text{sign}(g_{t+1}))$. PGD [Madry et al., 2018] adds random restarts to I-FG(S)M and δ_0 is sampled uniformly inside the ball B_ε. Figure 4 illustrates their relations.

Ensemble surrogate. Liu et al. [2017] show the benefit of ensembling architectures for inter-architecture transfer-based black-box attacks. Our work leans on theirs and complements it by demonstrating that attacking models sampled with cSGLD (performing Bayesian model averaging on a unique architecture) achieves better transferability at lesser computation cost.

Input and model transformations. Other approaches have been developed to improve transferability of adversarial examples. They transform the model or the input at test time (i.e., after training, when performing the attack). *Ghost Networks* (GN) [Li et al., 2018] use Dropout and Skip Connection Erosion to generate on-the-fly diverse sets of surrogate models from one or more base models. *Input Diversity* (DI) [Xie et al., 2019] applies random transformations (random resize followed by random padding) to the input images at each attack iteration. *Skip Gradient Method* (SGM) [Wu et al., 2020] favours the gradients from skip connections rather than residual modules through a decay factor applied to the latter during the backward pass. These techniques can naturally be combined to ours: (i) cSGLD can provide at a low computation cost a diverse set of base models to build GN; (ii) DI applies transformations to adversarial inputs independently of the surrogate models; (iii)

SGM modifies backward passes during the attack, independently of the training method. As our evaluation will reveal, our train-time method further improves the transferability of the above three techniques and outperforms them 87.5% of the time. It is also compatible with other test-time approaches not considered in this paper, such as linear backpropagation [Guo et al., 2020], intermediate level attack [Huang et al., 2019], Nesterov accelerated gradient and scale invariance [Lin et al., 2020], and serial mini-batch ensemble attack [Che, 2020].

Bayesian Neural Network (BNN) and adversarial examples. Though not our goal, past research aimed at generating adversarial examples for BNNs (we rather use Bayesian Deep Learning as a way to attack classical DNNs). Grosse et al. [2018] show that BNN uncertainty measures are vulnerable to high-confidence-low-uncertainty adversarial examples crafted on Gaussian Processes. Palacci and Hess [2018] show that several SG-MCMC sampling schemes are not secure against white-box attacks. Wang et al. [2018] used SGLD and Generative Adversarial Network to detect adversarial examples instead of crafting them.

The recent work of Carbone et al. [2020] shows that in the large-data, overparametrized limit for BNN, the gradient of the posterior predictive distribution reaches zero, which makes white-box gradient-based attacks unsuccessful. Our experiments reveal opposite conclusions: our surrogates DNNs suffer more often from vanished gradients than on BNN surrogates. Our setting differs from Carbone et al. [2020] in several aspects: (i) the attacker controls the BNN surrogate and can exploit the cyclical nature of cSGLD to avoid fully vanished gradients; (ii) our cheaper SG-MCMC approximation of the posterior predictive distribution (compared to the Hamiltonian Monte Carlo & Variational Inference used in Carbone et al. [2020]) may produce correlated MCMC samples, which explains that the averaged gradient is not zero; (iii) we consider larger datasets (ImageNet and CIFAR-10) than them (MNIST and Fashion-MNIST), where convergence to flat loss is harder to reach.

3 APPROACH

A Bayesian perspective on transferability. Under a specified threat model, we relate uncertainty and posterior predictive distributions to transferability. We consider a classification problem with a training dataset $\mathcal{D} = \{(x_i, y_i) \sim p(x, y)\}_{i=1}^{N}$ and C class labels. A probabilistic classifier parametrized by θ maps x_i into a predictive distribution $\hat{p}(y|x_i, \theta)$. A white-box adversarial perturbation of a test example $(x, y) \sim p(x, y)$ against such classifier is defined as:

$$\delta_\theta \in \underset{\|\delta\|_p \leq \varepsilon}{\arg\min}\, \hat{p}(y|x + \delta, \theta).$$

In practice, this optimization problem is solved by replacing the predictive distribution with a loss function (see section 2). The *transferability* phenomenon is the empirical observation that an adversarial example for one model is likely to be adversarial for another one Goodfellow et al. [2014]. Black-box attacks can leverage this property by crafting adversarial examples using white-box attacks against a surrogate model to target an unseen model Papernot et al. [2016].

Assumption 1 (Threat model)**.** We define our threat model with the following assumptions on the targeted classifier:

1. its architecture is known and so is its prediction function $\hat{p}(y|x, \bullet)$ [1]
2. its training set $\mathcal{D}$ is known
3. its parameters θ_t, estimated by maximum likelihood, are unknown
4. a reasonable prior on its parameters $p(\theta_t)$ is known[2]
5. no oracle access (test-time feedback).

Assuming the threat model described in Assumption 1, *uncertainty on target parameters arises from the stochastic nature of training*, and more specifically from two sources of randomness: (i) every Stochastic Gradient Descent (SGD) update depends on a random batch of training examples[3], (ii) weights are randomly initialized at the beginning of training[4]. From the attacker subjective view, the target parameters obtained at the end of training are random variables.

We argue that θ_t is approximately distributed according to the posterior distribution $p(\theta|\mathcal{D})$. [Mingard et al., 2020] observes a strong correlation between the probability to obtain with SGD or its variants a function consistent with a training set and the Bayesian posterior probability of this function. Mandt et al. [2017] shows that SGD with constant learning rate has a stationary distribution centred on an optimum, which approximates a posterior. Marginalizing over local optima, we obtain a posterior that is the distribution of SGD endpoints with a step decay learning rate schedule (as widely used).

Then, *the best transferable adversarial example approximately minimizes the Bayesian posterior predictive distribution* $p(y|x, \mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}\, \hat{p}(y|x, \theta)$ and our black-box attack objective is:

$$\delta^* \in \underset{\|\delta\|_p \leq \varepsilon}{\arg\min}\, \mathbb{E}_{\theta_t \sim p(\theta|\mathcal{D})}\, \hat{p}(y|x + \delta, \theta_t). \tag{1}$$

[1] We discuss the unknown architecture case further on.

[2] In practice, it corresponds to knowing the weight decay hyperparameter, see discussion below.

[3] The same argument holds for SGD variants.

[4] Despite being independent and identically distributed random variables, weights initialization values play an important role in guiding the SGD trajectory Frankle and Carbin [2019].

As previously stated, usually in adversarial learning, transferable adversarial examples are optimized against one surrogate model. This is similar to solving problem (1) deterministically by approximating the expectation of the posterior predictive with a "plug-in" estimation of the parameters, $\hat{\theta}_{\text{MAP}}$ the maximum a posteriori probability (MAP) estimate: $\delta^* \approx \delta_{\hat{\theta}_{\text{MAP}}}$. To avoid overfitting to the surrogate model, random transformations of inputs or prediction functions were developed in the literature (see related work section).

A fundamental issue is that the closed form of the posterior predictive distribution is intractable for DNNs. Our contribution lies in *sampling from the posterior distribution to build a surrogate in black-box adversarial attacks*. We replace the crude MAP approximation of the posterior predictive distribution with a more accurate one to generate transferable adversarial examples. Therefore, we focus on the training phase by considering the methods and the computational costs of obtaining the surrogate model, whereas most previous work searches to optimize adversarial examples crafting at the time of the attack ("test time").

SG-MCMC & cSGLD. In practice, we perform Bayesian model averaging using samples obtained from Stochastic Gradient-Markov Chain Monte Carlo (SG-MCMC). SG-MCMC is a family of approximate Bayesian inference techniques, inaugurated by SGLD [Welling and Teh, 2011], that combines SGD with MCMC. Adding noise during training allows to sample from the posterior distribution of parameters. Our method aims to solve the following optimization problem:

$$\delta_{\{\theta_s\}} \in \underset{\|\delta\|_p \leq \varepsilon}{\arg\min} \frac{1}{S} \sum_{s=1}^{S} \hat{p}(y|x+\delta, \theta_s) \ , \tag{2}$$

where $\{\theta_s \sim p(\theta|\mathcal{D})\}_{s=1}^{S}$ are samples of the posterior.

We choose to apply the recently proposed *cyclical Stochastic Gradient Langevin Dynamics* (cSGLD) [Zhang et al., 2020], a state-of-the-art SG-MCMC technique. cSGLD performs warm restarts by dividing the training into cycles that all start from the initial learning rate value (cf. Figure 3). Each cycle consists of (1) an exploration stage with larger learning rates which corresponds to the burn-in period of MCMC algorithms; (2) a sampling stage that samples parameters at regular intervals and operates with smaller learning rates and added noise. Starting a new cycle with a large learning rate allows the exploration of another local maximum of the loss landscape. Contrary to most SG-MCMC methods, cSGLD has the compelling benefit of sampling from both several modes of the posterior distributions and locally inside each mode, avoiding mode collapse. Another major advantage of cSGLD is that its computation overhead compared to SGD/Adam is negligible (0.019% flops for one epoch on PreResNet110 on CIFAR-10 and 0.015% for ResNet50 on ImageNet).

Difference with Ensembling. Our work differs from previous research Liu et al. [2017], Li et al. [2018], Xie et al. [2019] that relates diversity with transferability in the same way that Ensembling and Bayesian Model Averaging do Minka Thomas P. [2002]. The latter "assumes that the true model lies within the hypothesis class of the prior, and performs soft model selection [...]. In contrast, ensembles [...] combine the models to obtain a more powerful model; ensembles can be expected to be better when the true model does not lie within the hypothesis class" Lakshminarayanan et al. [2016]. Under Assumptions 1, the unknown target model, our true model here, lies within the hypothesis class of its prior by definition. Therefore, we argue that under these conditions, a Bayesian approach is a more natural way to select a surrogate model.

Target prior. We express the prior of a standard target DNN. Deterministic DNNs are classically trained using the cross-entropy loss regularized by weight decay:

$$\min_{\theta_t} -\frac{1}{N} \sum_{i=1}^{N} \log \hat{p}(y_i|x_i, \theta_t) + \frac{\lambda}{2} \|\theta_t\|^2,$$

with λ the weight decay hyperparameter. This maximum likelihood estimation (MLE) procedure corresponds to the maximum a posteriori inference (MAP) of this implied probabilistic model:

$$p(y, \theta_t|x) = p(y|x, \theta_t) p(\theta_t),$$

where $p(y|x, \theta_t)$ is the likelihood function and $p(\theta_t) = \mathcal{N}(\theta_t|0, \frac{1}{N\lambda} I)$ a Gaussian prior. Therefore, in this standard setting, the hypothesis 4 reduces to knowing the weight decay hyperparameter λ.

Extension to unknown architecture. Let $\mathcal{A} = \{a_i\}_i$ be a countable set of candidate architectures, $p(a)$ a prior on $\mathcal{A}$, θ^a the parameters of the architecture a and $\hat{p}^a(y|x, \theta^a)$ its predictive distribution. Discarding hypothesis 1 of Assumption 1 on the knowledge of the architecture, the architecture of the target a becomes a random variable. We perform *Bayesian Model Comparison* to compute the posterior over models:

$$p(a|\mathcal{D}) \propto p(\mathcal{D}|a) p(a). \tag{3}$$

We marginalize over architectures to express the complete posterior predictive distribution as the average across architectures weighted by their posterior probabilities:

$$\begin{aligned} p(y|x, \mathcal{D}) &= \sum_{a \in \mathcal{A}} p(a|\mathcal{D}) p(y|x, \mathcal{D}, a) \\ &\propto \mathbb{E}_{p(a)}\, p(\mathcal{D}|a)\, \mathbb{E}_{p(\theta^a|\mathcal{D})}\, \hat{p}^a(y|x, \theta^a) \end{aligned} \tag{4}$$

If $\mathcal{A}$ is finite and small, we can approximate this quantity with a weighted average of one cSGLD empirical posterior

predictive distribution per architecture. Otherwise, we estimate it with MCMC by sampling according to $p(a)$ a finite subset $A = \{a_i \sim p(a)\}_{i=1}^{S_A} \subset \mathcal{A}$ of architectures, where the number of architectures S_A is fixed by the computational budget. We sample S parameters $\{\theta_s^a\}_{s=1}^{S}$ for all $a \in A$. Then, our inter-architecture attack that minimizes our approximation of $p(y|x, \mathcal{D})$ becomes:

$$\delta_A \in \underset{\|\delta\|_p \leq \varepsilon}{\arg\min} \frac{1}{S_A S} \sum_{a \in A} p(\mathcal{D}|a) \sum_{s=1}^{S} \hat{p}^a(y|x+\delta, \theta_s^a) \quad (5)$$

Various methods exist to approximate model evidence Friel and Wyse [2012]. To simplify empirical conclusions, we assume that all architectures in $\mathcal{A}$ have approximately equal evidence. This strong assumption is reasonable here, since we select widely used architectures which are well-specified on the standard benchmark datasets evaluated. For fairness to ensemble baselines, our experiments on unknown architectures do not include the target architecture in the set $\mathcal{A}$.

Attack algorithm. One can approximate the solution of Equations 2 and 5 with minor modifications of existing adversarial attack algorithms, i.e. simply cycling surrogate models throughout iterations. To efficiently approximate Bayesian model averaging during iterative attacks, we compute the gradient of every iteration on a single model sample per architecture. If multiple architectures are attacked, we average their gradients (see Algorithm 1). The cost of iterative attacks, measured as the number of backward passes, does not increase with the number of samples S.

4 EXPERIMENTS

The goal of our approach is to increase the transferability of adversarial examples by using a surrogate sampled from the posterior distribution to attack a deterministic DNN.

Setup summary. The target models are deterministic DNNs and are never used as a surrogate. For a fair comparison between DNNs and cSGLD, we train the surrogate DNNs on CIFAR-10 using the same process as the target models. ImageNet targets are third-party pretrained models. Each cSGLD cycle lasts 50 epochs and samples 5 models on CIFAR-10, and 45 epochs/3 models on ImageNet. We report the success rate (misclassification rates of untargeted adversarial examples) averaged over three attack runs. We craft adversarial examples from correctly predicted test examples (all examples for CIFAR-10, and a random subset of 5000 examples for ImageNet). The iterative attacks (I-FG(S)M, MI-FG(S)M, and PGD) perform 50 iterations such that the transferability rates plateaus (Figures 6 and 7). Each attack computes the gradient of 1 model per architecture. Therefore, their computation cost and volatile memory are not multiplied by the size of the surrogate, except for FG(S)M which computes its unique gradient against all available models. The source code is publicly available[5]. Appendix A.1 presents the experimental setup in details.

Table 1: Number of DNNs (T-DEE) and training computation budget (in flops) to achieve the intra-architecture transferability of cSGLD with Deep Ensemble. Higher is better. ">15" means that 15 DNNs always transfer less than cSGLD.

Dataset	Attack	Norm	T-DEE	Flops Ratio
CIFAR-10	I-FG(S)M	L2	>15 ±nan	>15 ±nan
		L∞	3.76 ±0.08	3.76 ±0.08
	MI-FG(S)M	L2	5.56 ±0.80	5.56 ±0.80
		L∞	2.88 ±0.03	2.87 ±0.03
	PGD	L2	>15 ±nan	>15 ±nan
		L∞	3.74 ±0.12	3.74 ±0.12
	FG(S)M	L2	>15 ±nan	>15 ±nan
		L∞	8.72 ±0.01	8.72 ±0.01
ImageNet	I-FG(S)M	L2	4.91 ±0.11	2.84 ±0.06
		L∞	4.34 ±0.13	2.51 ±0.08
	MI-FG(S)M	L2	4.69 ±0.18	2.71 ±0.10
		L∞	4.38 ±0.03	2.53 ±0.02
	PGD	L2	5.00 ±0.11	2.89 ±0.06
		L∞	4.42 ±0.16	2.56 ±0.09
	FG(S)M	L2	5.81 ±0.34	3.35 ±0.19
		L∞	5.98 ±0.03	3.46 ±0.02

4.1 INTRA-ARCHITECTURE TRANSFERABILITY

Since SG-MCMC methods sample the weights of a given architecture, we expect our approach to work particularly well in settings where the architecture of the target model is known, but not its weights. To demonstrate this, we compare the intra-architecture transfer success rates of cSGLD with the ones of Deep Ensemble surrogates (using 1 up to 15 independently trained DNNs). Architectures are PreResNet110 (CIFAR-10) and ResNet-50 (ImageNet).

Table 7 provides the detailed results for four classical gradient-based attacks. In summary, for a similar computation cost, cSGLD systematically increases the success rate of iterative attacks by 13.8 (ImageNet, MI-FGSM, L∞) to 49.2 (CIFAR-10, I-FGM, L2) percentage points, and of FG(S)M by 12.18 to 22.2. One explanation for the highest improvements is that DNN-based L2 norm attacks suffer from vanishing gradients on CIFAR-10, whereas cSGLD avoids it thanks to fast convergence and warm restarts (cf. Table 6 for proportions of vanished gradients).

Inspired by DEE [Ashukha et al., 2020], we propose the **Transferability-Deep Ensemble Equivalent (T-DEE)** metric as the number of independently trained DNNs needed to achieve the same success rate as the technique considered (computed with linear interpolation). Under some assump-

[5]URL retracted for anonymous submission.

Table 2: Transfer success rates of I-FG(S)M attack on CIFAR-10 hold-out architectures. The $\star$ symbol indicates that 1 DNN per architecture is better than 1 cSGLD per architecture. Higher is better.

Norm	Surrogate	−PreResNet110	−PreResNet164	−VGG16bn	−VGG19bn	−WideResNet	Nb epochs
L2	1 cSGLD per arch.	**95.56% ±0.04**	**95.72% ±0.06**	**45.96% ±0.07**	**42.60% ±0.08**	**84.04% ±0.05**	4×300
	1 DNN per arch.	60.38% ±1.09	60.93% ±1.06	29.97% ±0.48	27.57% ±0.66	57.86% ±0.74	4×300
	4 DNNs per arch.	77.12% ±1.32	77.21% ±1.14	40.89% ±0.63	40.18% ±0.76	77.54% ±0.93	4×1200
L∞	1 cSGLD per arch.	96.38% ±0.06	96.51% ±0.08	49.19% ±0.06	45.17% ±0.03	84.75% ±0.01	4×300
	1 DNN per arch.	87.02% ±0.04	88.86% ±0.04	44.99% ±0.10	$\star$45.55% ±0.02	74.84% ±0.03	4×300
	4 DNNs per arch.	**96.50% ±0.01**	**97.01% ±0.02**	**59.80% ±0.01**	**59.08% ±0.01**	**89.23% ±0.04**	4×1200

Table 3: Transfer success rates of I-FG(S)M attack on ImageNet hold-out architectures. Higher is better.

Norm	Surrogate	−ResNet50	−ResNeXt50	−DenseNet121	−MNASNet	−EffNetB0	Nb epochs
L2	1 cSGLD per arch.	**93.28% ±0.12**	**90.61% ±0.24**	**92.25% ±0.26**	**95.98% ±0.19**	**81.88% ±0.38**	4×135
	1 DNN per arch.	72.99% ±0.52	72.31% ±0.44	64.72% ±0.59	84.21% ±0.18	53.99% ±0.76	4×135
L∞	1 cSGLD per arch.	**92.21% ±0.23**	**89.83% ±0.22**	**90.86% ±0.19**	**95.85% ±0.46**	**79.40% ±0.42**	4×135
	1 DNN per arch.	69.65% ±0.47	69.01% ±0.70	61.00% ±0.66	82.25% ±0.03	49.71% ±1.37	4×135

tions[6], Deep Ensemble samples exactly from the distribution of target parameters, and is thus optimal for intra-architecture transferability with infinite computing power.

Table 1 reports the T-DEE and the *computing ratio*, i.e., the total number of flops to train such DNNs ensemble divided by the number of flops used to trained cSGLD. This ratio represents the computational gain factor achieved by our approach[7]. In the worst case across the two datasets, an ensemble of 3 surrogate DNNs is required to beat the cSGLD surrogate, while requiring at least 2.51 times more flops during the training phase. On CIFAR-10 and considering L2 attack specifically (MI-FGM aside), it even outperforms the ensemble of 15 DNNs by a significant factor (up to 30.3 percentage points). On ImageNet, cSGLD achieves the same transfer rate as 4.38–5.98 DNNs, which corresponds to dividing the number of flops by 2.51–3.46.

Then, the uncertainty on parameter estimation captured by cSGLD is useful to discover generic adversarial directions.

4.2 INTER-ARCHITECTURE TRANSFERABILITY

We now focus on black-box settings where the architecture of the target model is unknown (and not used to build the surrogate model). We consider ten architectures (five for both datasets). Following Liu et al. [2017], Xie et al. [2019], Li et al. [2018], Dong et al. [2018], we hold-out one architecture to act as the target model and use the four remaining ones as surrogates. We apply I-FG(S)M with 1 model per surrogate architecture per iteration to keep attack cost constant. Due to computational limitations, we limit the training to 135 epochs on ImageNet (3 cycles of 45 epochs for cSGLD). For every architecture, cSGLD and 1 DNN are trained for the same number of epochs.

As shown in Tables 2 (CIFAR-10) and 3 (ImageNet), our method significantly improved transferability on all 5 hold-out architectures for both datasets, except for the L∞ VGG19bn target (with a difference of 0.4 percentage point). On CIFAR-10, the differences range from 15.0 to 35.2 percentage points (2-norm), and from -0.4 to 9.9 (∞-norm). Our method outperforms 4 DNNs per architecture on the L2 attack, despite been trained for 4 times fewer epochs. On ImageNet, cSGLD improves over the one DNN counterpart by 11.8 and 29.9 percentage points of transfer rate at constant computational train and attack budget.

Tables 8 and 9 present the results for an alternative protocol where we use a single architecture as surrogate. In summary, in this setup cSGLD achieves a higher inter-architecture success rate in 38/40 cases on CIFAR-10, and 39/40 cases on ImageNet, compared to a single DNN trained for the same number of epochs. Differences range between -2.3 and 62.1 percentage points on CIFAR-10 and -0.3 and 44.8 on ImageNet.

We conclude that our method improves transferability even when the target architecture is unknown. This tends to indicate that the adversarial directions against posterior predictive distribution are partially aligned across different architectures. In other words, given a common classification task, the variability of an architecture parameters might be informative of the variability of another architecture parameters.

4.3 TEST-TIME TRANSFERABILITY TECHNIQUES

Given that our approach works at train time, we evaluate its combination with test-time techniques. We apply three test-

[6]Besides Assumptions 1, we suppose that Deep Ensemble uses the target optimizer, and that the minimum in Eq. 2 is reached, i.e. that the attack doesn't fail due to vanished or obfuscated gradients Athalye et al. [2018].

[7]The ImageNet computing ratios don't equal to T-DEE since 1 DNN is trained for 130 epochs and cSGLD for 225.

Table 4: Transfer success rates of (M)I-FG(S)M improved by our approach combined with test-time techniques on ImageNet. Target in column. ResNet50 is intra-architecture transferability, others are inter-architecture. Bold is best. Symbols ⋆ are DNN-based techniques better than our vanilla cSGLD surrogate, † are techniques that degrades their vanilla surrogate. All techniques improve with cSGLD compared to 1 DNN.

Norm	Surrogate	ResNet50	ResNeXt50	DenseNet121	MNASNet	EffNetB0
L2	1 DNN	56.60% ±0.71	41.09% ±0.61	29.73% ±0.30	28.13% ±0.17	16.64% ±0.33
	+ Input Diversity	83.15% ±0.30	73.17% ±0.80	61.24% ±0.58	58.16% ±0.36	⋆42.10% ±0.36
	+ Skip Gradient Method	65.64% ±0.88	52.75% ±0.42	38.58% ±0.55	43.40% ±0.61	29.11% ±0.30
	+ Ghost Networks	78.84% ±0.46	62.46% ±0.38	45.76% ±0.02	41.44% ±0.58	25.77% ±0.11
	+ Momentum (MI-FGSM)	†52.53% ±0.80	†37.15% ±0.76	†26.33% ±0.48	†25.21% ±0.42	†14.74% ±0.31
	+ Input Diversity	80.81% ±0.72	69.55% ±0.83	56.73% ±0.39	54.16% ±0.05	37.07% ±0.03
	+ Skip Gradient Method	65.65% ±0.95	53.25% ±0.18	38.79% ±0.62	44.33% ±0.63	29.45% ±0.28
	+ Ghost Networks	71.50% ±0.12	53.45% ±0.65	37.39% ±0.47	34.53% ±0.69	20.29% ±0.36
	cSGLD	84.83% ±0.55	74.73% ±0.82	71.45% ±0.56	60.14% ±0.44	39.71% ±0.20
	+ Input Diversity	**93.87% ±0.19**	**89.12% ±0.24**	**88.52% ±0.16**	**82.78% ±0.28**	**66.13% ±0.35**
	+ Skip Gradient Method	†83.17% ±0.85	†72.79% ±1.06	†66.19% ±0.89	71.71% ±0.41	52.66% ±0.31
	+ Ghost Networks	92.99% ±0.13	85.69% ±0.24	82.81% ±0.42	72.88% ±0.30	50.30% ±0.29
	+ Momentum (MI-FGSM)	†82.44% ±0.19	†70.93% ±1.04	†66.19% ±0.56	†55.51% ±0.59	†34.49% ±0.59
	+ Input Diversity	93.48% ±0.23	87.87% ±0.15	86.81% ±0.33	80.37% ±0.20	60.26% ±0.02
	+ Skip Gradient Method	†82.35% ±0.10	†71.54% ±0.58	†64.50% ±0.18	70.47% ±0.22	50.80% ±0.23
	+ Ghost Networks	90.11% ±0.18	80.35% ±0.61	75.10% ±0.67	64.08% ±0.12	39.85% ±0.52
L∞	1 DNN	47.81% ±1.09	32.29% ±0.64	23.43% ±0.32	22.52% ±0.45	12.77% ±0.32
	+ Input Diversity	76.55% ±1.01	62.57% ±0.56	50.17% ±0.33	49.31% ±0.18	⋆32.64% ±0.09
	+ Skip Gradient Method	66.36% ±0.50	51.60% ±0.36	39.05% ±0.24	45.60% ±0.72	30.69% ±0.03
	+ Ghost Networks	67.02% ±0.17	46.74% ±0.63	32.57% ±0.17	31.12% ±0.77	17.68% ±0.05
	+ Momentum (MI-FGSM)	55.12% ±0.82	38.47% ±0.82	28.19% ±0.14	27.55% ±0.67	16.34% ±0.37
	+ Input Diversity	⋆82.47% ±0.41	⋆69.69% ±0.81	57.79% ±0.57	⋆55.99% ±0.37	⋆38.63% ±0.29
	+ Skip Gradient Method	68.39% ±0.53	54.57% ±0.60	41.48% ±0.37	47.97% ±0.41	⋆33.16% ±0.37
	+ Ghost Networks	71.27% ±0.54	51.46% ±0.84	36.91% ±0.48	34.54% ±0.32	20.51% ±0.30
	cSGLD	78.71% ±1.19	65.11% ±1.45	61.49% ±0.59	51.81% ±1.45	31.11% ±0.99
	+ Input Diversity	90.03% ±0.10	82.13% ±0.45	81.19% ±0.34	74.48% ±0.39	53.51% ±0.39
	+ Skip Gradient Method	81.37% ±0.72	69.88% ±1.31	65.20% ±0.75	71.68% ±0.53	52.15% ±0.32
	+ Ghost Networks	87.33% ±0.73	76.00% ±1.33	71.67% ±0.97	61.45% ±0.25	37.19% ±0.68
	+ Momentum (MI-FGSM)	82.89% ±0.70	70.42% ±1.26	66.39% ±0.74	56.68% ±0.97	36.00% ±1.15
	+ Input Diversity	**93.97% ±0.26**	**87.69% ±0.44**	**86.78% ±0.16**	**81.08% ±0.14**	**60.87% ±0.48**
	+ Skip Gradient Method	84.19% ±0.21	73.14% ±0.99	67.35% ±0.26	74.36% ±0.47	55.30% ±0.16
	+ Ghost Networks	89.53% ±0.05	78.69% ±0.19	73.33% ±0.58	63.56% ±0.35	39.79% ±0.52

time transformations to cSGLD samples and one DNN obtained with the same number of epochs (300 for CIFAR-10, 135 for ImageNet). The CIFAR-10 surrogates are PreResNet110 (respect. ResNet50 on ImageNet). The targets are the same as in Section 4.2. Following [Li et al., 2018, Xie et al., 2019, Wu et al., 2020], we also combine every test-time technique with momentum[8].

Table 4 shows the results on ImageNet (detailed results on CIFAR-10 are provided in Table 10). We observe that our approach and the test-time techniques complement well to each other. Indeed, the best success rates are always achieved by a technique applied on cSGLD (in bold). All three techniques combined with momentum applied on cSGLD achieve a systematically higher success rate than the same technique applied on 1 DNN, with differences ranging from 10.7 to 41.7 percentages points on ImageNet and from 3.8 to 56.2 on CIFAR-10. Overall, the addition of a technique (excluding momentum alone) to our vanilla cSGLD surrogate never decrease the success rate on CIFAR-10 and only in 10% of the averaged cases considered on ImageNet, as indicated by the † symbols.

Besides, our vanilla cSGLD surrogate achieves better transferability than any of the test-time techniques applied to 1 DNN in 90% of the cases on CIFAR-10 and 93.3% on ImageNet, using the I-FG(S)M attack. Similarly, for MI-FG(S)M, we observe 76.7% for the former and 90% for the latter. This demonstrates that despite previous efforts in providing effective test-time techniques for transferability (see Section 2), *improving the training of the surrogate – in our case, through efficient sampling from the posterior distribution – yields significantly higher improvements*. Hence, while training approaches have been overlooked, canonical elements that have been related to transferability, ie. skip connections [Wu et al., 2020], input [Xie et al., 2019] and model diversity [Li et al., 2018], should be put into

[8] All rows with momentum correspond to MI-FG(S)M, an attack variant designed to improve transferability [Dong et al., 2018].

perspective compared to the importance that the posterior distribution appears to have.

4.4 BAYESIAN AND ENSEMBLE TECHNIQUES

In addition to cSGLD and Deep Ensemble, we explore the use of other training techniques to improve transferability: two other Bayesian techniques – Variational Inference (VI) and Stochastic Weight Averaging-Gaussian (SWAG) – and two other ensembling techniques – Snapshot ensembles (SSE) and Fast Geometric Ensembling (FGE). We train each for an equivalent computational cost of 3 DNNs on CIFAR-10 and 2 DNNs on ImageNet (except for VI and SWAG, see discussion in Appendix A.2). Figure 2 presents the success rate of L∞ I-FS(S)M attack with the corresponding training computational cost (in flops), as we increase the number of models in each surrogate. Appendix A.2 contains details on the methods and the results for L2 I-FS(S)M.

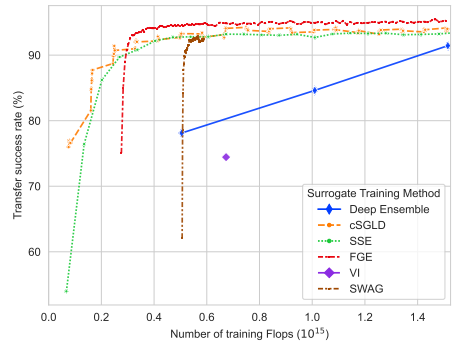
On CIFAR-10, the success rate of the first 4 cycles of cSGLD increases substantially from one cycle to the next (from 76.58% to 81.56% for the first to the second cycle) and within a single cycle (from 81.56% to 87.20% between the start and the end of the second cycle). This reveals that exploring modes of the posterior plays an important role to generate transferable adversarial examples, and that there is some local geometric discrepancy of the loss landscape among local maxima. On ImageNet, transferability improves mainly by sampling from several local optima.

Interestingly, even though FGE and SWAG build an ensemble around a single local optimum, their flexibility allows capturing general adversarial directions. The FGE surrogates trained for more than 0.30 petaflops have systematically higher success rates than cSGLD and SSE on CIFAR-10. However, the opposite is observed on ImageNet: FGE is not competitive with methods exploring several local optima (cSGLD, SSE, and Deep Ensemble). We hypothesize that modes are not as well-connected on larger datasets.

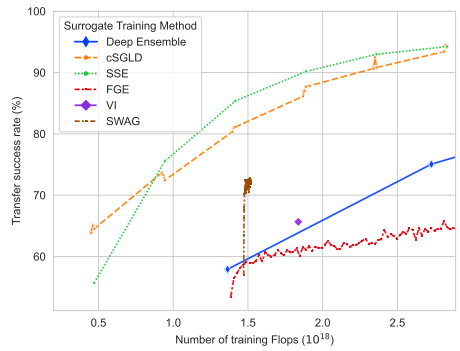
The efficiency of SWAG on both datasets opens new directions to create hybrid attacks based on few additional iterations over the training set. SWAG approximates the posterior with a Gaussian fitted on some additional SGD epochs from a pretrained DNN. It captures well the shape of the true posterior [Maddox et al., 2019], reinforcing our views on the strong relationship between the posterior and transferability. The success rate gap between cSGLD/SSE and SWAG on ImageNet suggests higher geometrical discrepancies between local loss maxima on larger datasets.

VI fails to compete with Deep Ensemble on both success rate and computational efficiency for the L∞ attack on CIFAR-10, but beats it on L2 bound and on ImageNet.

On CIFAR-10, the marginal impact beyond 6 cSGLD cycles, 17 SWAG samples, 7 SSE models, and 35 FGE models becomes noisy. We hypothesize that correlated samples produce these limitations. Hence, the use of multiple runs is a promising direction for greater transferability.



(a) CIFAR-10



(b) ImageNet

Figure 2: Intra-architecture L∞ I-FG(S)M success rate with respect to the training computational complexity of an increasing number of samples from six training techniques.

5 CONCLUSION

We are the first to extensively investigate training-time approaches to enhance transferability. We discover a strong connection between the posterior predictive distribution and both intra- and inter-architecture transferability. Our Bayesian surrogate is efficient and effective to craft adversarial examples transferable to deterministic DNNs. Our approach further improves existing adversarial attacks and test-time transferability techniques, as one can use it on top of them to perform approximate Bayesian model averaging efficiently and with minimal modifications. We show that our simple training-time approach improves transferability more than previous test-time techniques. We, therefore, cast an important yet overlooked direction to explain transferability and pave the way for new hybrid attacks. Overall, we provide new evidence that the Bayesian framework is a promising direction for research on adversarial examples.

References

Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov. Pitfalls of In-Domain Uncertainty Estimation and Ensembling in Deep Learning. *ICLR*, 2 2020. ISSN 2331-8422. URL http://arxiv.org/abs/2002.06470.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. *ICML*, 2 2018.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Lecture Notes in Computer Science*, volume 8190 LNAI, pages 387–402, 8 2013. doi: 10.1007/978-3-642-40994-3{_}25.

Ginevra Carbone, Matthew Wicker, Luca Laurenti, Andrea Patane, Luca Bortolussi, and Guido Sanguinetti. Robustness of Bayesian neural networks to gradient-based attacks. In *NeurIPS*. arXiv, 2 2020.

Zhaohui Che. A New Ensemble Adversarial Attack Powered by Long-Term Gradient Memories. *AAAI*, pages 3405–3413, 2020.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, volume PartF16814, pages 2184–2194, 3 2020. ISBN 9781713821120. URL http://arxiv.org/abs/2003.01690.

Shaveta Dargan, Munish Kumar, Maruthi Rohit Ayyagari, and Gulshan Kumar. A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning. *Archives of Computational Methods in Engineering*, 27 (4):1071–1092, 9 2019.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting Adversarial Attacks with Momentum. In *CVPR*, pages 9185–9193, 10 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00957. URL http://arxiv.org/abs/1710.06081.

Jonathan Frankle and Michael Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. *ICLR*, 2019.

Nial Friel and Jason Wyse. Estimating the evidence - a review. *Statistica Neerlandica*, 66(3):288–308, 8 2012. ISSN 00390402.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry Vetrov, and Andrew Gordon Wilson. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs. *NeurIPS*, pages 8789–8798, 2 2018. URL https://arxiv.org/abs/1802.10026v4.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. 12 2014.

Kathrin Grosse, David Pfaff, Michael Thomas Smith, and Michael Backes. The Limitations of Model Uncertainty in Adversarial Settings. *NeurIPS Workshop*, 12 2018. URL http://arxiv.org/abs/1812.02606.

Yiwen Guo, Qizhang Li, and Hao Chen. Backpropagating Linearly Improves Transferability of Adversarial Examples. Technical report, 2020. URL https://github.com/qizhangli/linbp-attack.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *ICCV*, pages 770–778. IEEE Computer Society, 12 2016a. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity Mappings in Deep Residual Networks. *Lecture Notes in Computer Science*, 9908 LNCS:630–645, 3 2016b.

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*. International Conference on Learning Representations, ICLR, 3 2017. URL http://arxiv.org/abs/1704.00109.

Qian Huang, Isay Katsman, Zeqi Gu, Horace He, Serge Belongie, and Ser Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *ICCV*, pages 4732–4741, 7 2019. doi: 10.1109/ICCV.2019.00483. URL http://arxiv.org/abs/1907.10823.

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009. ISSN 1098-6596. doi: 10.1.1.222.9220.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 7 2019. URL http://arxiv.org/abs/1607.02533.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *NeurIPS*, pages 6403–6414, 2016.

Yingwei Li, Song Bai, Yuyin Zhou, Cihang Xie, Zhishuai Zhang, and Alan Yuille. Learning Transferable Adversarial Examples via Ghost Networks. *AAAI*, 34(07), 12 2018. ISSN 2374-3468. doi: 10.1609/aaai.v34i07.6810. URL http://arxiv.org/abs/1812.03413.

Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks. In *ICLR*, 8 2020. URL http://arxiv.org/abs/1908.06281.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *ICLR*, 11 2017. URL https://arxiv.org/abs/1611.02770.

Wesley J. Maddox, Timur Garipov, Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *NeurIPS*, volume 32. arXiv, 2 2019. URL http://arxiv.org/abs/1902.02476.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 6 2018. URL http://arxiv.org/abs/1706.06083.

Stephan Mandt, Matthew D. Hof Fman, and David M. Blei. Stochastic Gradient Descent as Approximate Bayesian Inference. *Journal of Machine Learning Research*, 18: 1–35, 4 2017. ISSN 15337928. URL https://arxiv.org/abs/1704.04289v2.

Chris Mingard, Guillermo Valle-Pérez, Joar Skalse, and Ard A. Louis. Is SGD a Bayesian sampler? Well, almost. *Journal of Machine Learning Research*, 22, 6 2020. ISSN 15337928.

Minka Thomas P. Bayesian model averaging is not model combination. Technical report, 2002.

Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial Robustness Toolbox v1.2.0. *CoRR*, 1807.01069, 2018.

Henri Palacci and Henry Hess. Scalable Natural Gradient Langevin Dynamics in Practice. 6 2018. URL http://arxiv.org/abs/1806.02855.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. 5 2016. URL http://arxiv.org/abs/1605.07277.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, pages 8024–8035. Curran Associates, Inc., 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. 12 2013. URL http://arxiv.org/abs/1312.6199.

Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ICML*, pages 10691–10700, 5 2019.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. MnasNet: Platform-Aware Neural Architecture Search for Mobile. *ICCV*, 2018.

Kuan Chieh Wang, Paul Vicol, James Lucas, Li Gu, Roger Grosse, and Richard Zemel. Adversarial distillation of Bayesian neural network posteriors. In *ICML*, volume 12, pages 8239–8248, 6 2018. ISBN 9781510867963. URL http://arxiv.org/abs/1806.10317.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *ICML 2011*, pages 681–688, 2011. ISBN 9781450306195.

Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip Connections Matter: On the Transferability of Adversarial Examples Generated with ResNets. *ICLR*, 2 2020.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L. Yuille. Improving transferability of adversarial examples with input diversity. In *ICCV*, 3 2019. ISBN 9781728132938. doi: 10.1109/CVPR.2019.00284. URL http://arxiv.org/abs/1803.06978.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995, 11 2017.

Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. *British Machine Vision Conference 2016, BMVC 2016*, 2016-Septe:1–87, 5 2016.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical Stochastic Gradient MCMC for Bayesian Deep Learning. *ICLR*, 2 2020. URL http://arxiv.org/abs/1902.03932.

A SUPPLEMENTARY MATERIALS

In the supplementary materials for the paper, the following are provided:

- The detailed experimental setup section
- The description and experimental setup of Bayesian and Ensembling training techniques
- Additional results, including:
  - The natural accuracy of target Neural Networks;
  - The proportions of vanishing gradients of cSGLD surrogate compared to DNN surrogate;
  - The intra-architecture transfer success rates on cSGLD and Deep Ensemble of 1, 2, 5 and 15 DNNs surrogates;
  - The inter-architecture transfer success rates of single architecture surrogates;
  - The intra-architecture transfer success rate of 6 Bayesian and Ensemble training methods attacked by L2 I-FG(S)M;
  - The intra-architecture transfer success rate combined with test-time transformations on CIFAR-10;
  - The transfer rate of cSGLD with respect to the number of cycles and samples per cycle;
- An illustration of the cSGLD cyclical learning rate schedule;
- A diagram of the relationships between gradient-based attacks;
- The algorithm applied to perform approximate Bayesian model averaging efficiently;
- Details on hyperparameters, including:
  - The transfer success rate of iterative attacks with respect to the number of iterations;
  - The tuning of the hyperparameter of the Skip Gradient Method technique to extend it to PreResNet110;
  - The hyperparameters used to train and attack models;

A.1 EXPERIMENTAL SETUP

Datasets. We consider CIFAR-10 [Krizhevsky, 2009] and ImageNet (ILSVRC2012; Russakovsky et al. 2015). In all cases, we train the surrogate and target models on the entire training set. For each CIFAR-10 target model, we select all the examples from the test set that are correctly predicted by it. In the case of ImageNet, we use a random subset of 5000 correctly predicted test images.

Architectures. Following the work of Ashukha et al. [2020], we consider the following five architectures for CIFAR-10: PreResNet110, PreResNet164 [He et al., 2016b], VGG16BN, VGG19BN [Simonyan and Zisserman, 2015], and WideResNet28x10 [Zagoruyko and Komodakis, 2016]. For ImageNet, we select five architectures with $3 \times 224 \times 244$ input size. Three classical architectures: ResNet-50 [He et al., 2016a][9], ResNeXt-50 32x4d [Xie et al., 2017] and Densenet-121 [Xie et al., 2017]; and 2 mobile architectures: MNASNet 1.0 [Tan et al., 2018] and EfficientNet-B0 [Tan and Le, 2019]. In doing so, we cover a diverse set of architectures in terms of heterogeneity (similar and different families of architecture), computation cost, and release date.

Target models. The target models are deterministic DNNs. In the case of CIFAR-10, they are trained using Adam optimizer for 300 epochs with step-wise learning rate decay that divides it by 10 every 75 epochs (the architecture depends on the experiments). For ImageNet, we use the pre-trained models provided by PyTorch [Paszke et al., 2019] and the pre-trained EfficientNet-B0 provided by PyTorch Image Models (*timm*). The benign accuracy of all target models exceeds 83% (CIFAR-10) and 73% (ImageNet); see Table 5 below exact values.

Surrogate models (Deep Ensemble). For CIFAR-10, the DNNs used to form surrogate ensembles are trained using the same process as the target models. Therefore the comparison between deterministic DNNs and cSGLD is fair since one can expect the deterministic DNNs surrogate to be "close" to the target. As for ImageNet, we retrieve an ensemble of 15 ResNet-50 models trained independently by Ashukha et al. [2020] using SGD with momentum during 130 epochs. For the RQ2 experiments, we train similarly one model for every 4 other ImageNet architectures.

Surrogate models (cSGLD). Following the work of Ashukha et al. [2020] and Zhang et al. [2020], we train models with cSGLD on CIFAR-10 for 6 learning rate cycles (which, as our RQ4 experiments reveal, is where the transfer rate starts plateauing). cSGLD performs 5 cycles on ImageNet. The learning rate is set with cosine annealing schedule for fast convergence. Each cycle lasts 50 epochs on CIFAR-10 and 45 on ImageNet. The last epochs of every cycle form the sampling phase: noise is added and one sample is drawn at the end of each epoch. On CIFAR-10, we obtain 5 samples per cycle (resp. 3 on ImageNet), so 30 samples in total (resp. 15). An illustration of a cSGLD cyclical learning rate schedule is in supplementary materials. To train ResNet-50 models on ImageNet, we re-use the original cSGLD hyperparameters.

Surrogate models (other training methods). Additionally to Deep Ensemble cSGLD and following Ashukha et al. [2020], we consider 2 Bayesian Deep Learning techniques (SWAG and VI) and 2 Ensemble ones (SSE and FGE). We train every technique on CIFAR-10 and cSGLD and SWAG

[9] Ashukha et al. [2020] study ResNet-50 only on ImageNet. We used their shared trained models as surrogate DNNs.

on ImageNet. We retrieve trained Deep Ensemble, SSE, FGE and VI ImageNet models from Ashukha et al. [2020]. Technique descriptions and experimental setup of surrogates trained with SWAG, VI, FGE, or SSE are detailed below in the Bayesian and Ensemble Training Techniques section.

Adversarial attacks. We applied our variant of 4 gradient-based attacks as described in the approach section. The attacker's goal is misclassification (untargeted adversarial examples). We perform both 2-norm and ∞-norm bounded adversarial attacks, and report means and standard deviations computed on 3 random seeds. In accordance to values commonly used in the literature [Croce and Hein, 2020], the maximum perturbation norm ε is set respectively to 0.5 and $\frac{4}{255}$ on CIFAR-10, and respectively to 3 and $\frac{4}{255}$ on ImageNet. The step-size α is set to $\frac{\varepsilon}{10}$. We choose to perform 50 iterations such that the transferability rates plateaus for all iterative attacks (I-FGSM, MI-FGSM and PGD) on both norms and both datasets (see Figures 6 and 7 below). PGD runs with 5 random restarts. FG(S)M aside, *every iteration computes the gradient of 1 model per architecture*. Therefore, the attack computation cost and volatile memory are not multiplied by the size of the surrogate, except for FG(S)M which computes its unique gradient against all available models. cSGLD samples are attacked in random order. The MI-FG(S)M decay factor is set to 0.9.

Test-time Transformations. In the dedicated section, we consider three test-time transformations applied during attack designed for transferability (see related work section): Ghost Networks [Li et al., 2018], Input Diversity [Xie et al., 2019] and Skip Gradient Method [Wu et al., 2020]. We implemented the first two in PyTorch with their original hyperparameters. To extend Input Diversity to the smaller input sizes of CIFAR-10, we keep the same maximum resize ratio of 0.9. We reuse original implementation of the third one on ResNet50, and extend it to PreResNet110 (we set its hyperparameter via grid-search, see Figure 10 below).

Implementation. The source code of the experiments are publicly available on GitHub[10]. Our attack is built on top of the Python ART library [Nicolae et al., 2018]. cSGLD, VI, SSE, and FGE models were trained thanks to the implementation of Ashukha et al. [2020] available on GitHub[11]. All models were trained with PyTorch [Paszke et al., 2019]. We use EfficientNet-B0 from timm[12]. We train SWAG on ImageNet with the original implementation [Maddox et al., 2019]. We use the following software versions: Python 3.8.8, Pytorch 1.7.1 (1.9.0 for Flops measurement), torchvision 0.8.2, Adversarial Robustness Toolbox 1.6.0, and timm 0.3.2.

[10] `https://github.com/Framartin/transferable-bnn-adv-ex`

[11] `https://github.com/bayesgroup/pytorch-ensembles`

[12] `https://github.com/rwightman/pytorch-image-models`

Flops. We measure the training computational complexity in Flops using the PyTorch profiler. The computation overhead of one epoch with cSGLD compared to one with SGD/Adam is negligible. The main difference is the addition of noise to the weights during the sampling phase. On CIFAR-10, the overhead of 1 cSGLD epoch of PreResNet110 with added noise compared to one of a DNN trained with Adam (SGD) is 0.0187% Flops (respectively 0.0146% for ResNet50 on ImageNet).

Infrastructure. Experiments were run on Tesla V100-DGXS-32GB GPUs. The server has the following specifications: 256GB RDIMM DDR4, CUDA version 10.1, Linux (Ubuntu) operating system.

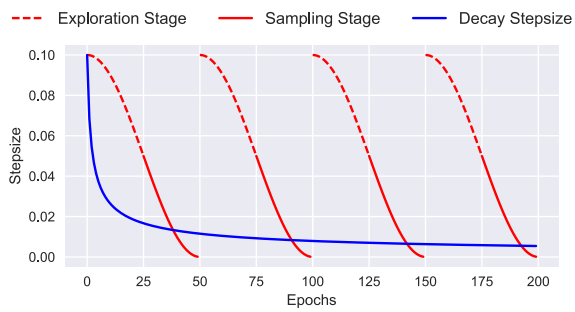


Figure 3: Illustration of the cSGLD cyclical learning rate schedule (red) and the traditional decreasing learning rate schedule (blue). Each cSGLD cycle is composed of an exploration phase (burn-in period of MCMC algorithms — red doted) and of a sampling phase (red plain). Figure taken from Zhang et al. (2020).

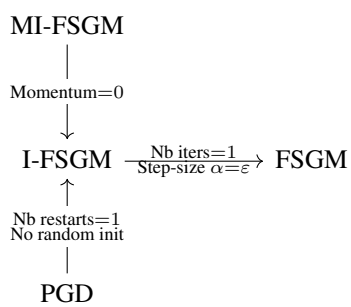


Figure 4: Relationships between gradient-based attacks.

A.2 BAYESIAN AND ENSEMBLE TRAINING TECHNIQUES

Following the work of Ashukha et al. [2020], we consider the following training techniques: Deep Ensemble [Lakshminarayanan et al., 2016], cSGLD [Zhang et al., 2020], SWAG [Maddox et al., 2019], VI, SSE [Huang et al., 2017],

Algorithm 1 Variant of I-FG(S)M attack to perform approximate Bayesian Model Averaging efficiently on numerous models from several architectures

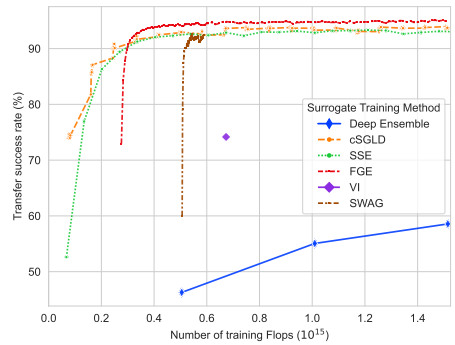
Input: original example (x, y), S_A ordered sets of model parameters $(\theta_s^1)_{s=1}^S, \ldots, (\theta_s^{S_A})_{s=1}^S$ sampled from the corresponding posterior distribution $\theta_s^i \sim p(\theta_s|\mathcal{D})$, number of iterations n_{iter}, perturbation p-norm ε, step-size α
Output: adversarial example x_{adv}
Shuffle each ordered set of model samples $(\theta_s^1)_{s=1}^S, \ldots, (\theta_s^{S_A})_{s=1}^S$
$x_{\text{adv}} \leftarrow x$
for $i = 1$ **to** n_{iter} **do**
  $x_{\text{adv}} \leftarrow x_{\text{adv}} + \frac{\alpha}{S_A} \sum_{a=1}^{S_A} \nabla \mathcal{L}(x_{\text{adv}};\, y, \theta^a_{i \bmod S})$
  $x_{\text{adv}} \leftarrow \text{project}(x_{\text{adv}}, B_\varepsilon[x])$
  $x_{\text{adv}} \leftarrow \text{clip}(x_{\text{adv}})$
end for

and FGE [Garipov et al., 2018]. For computational limitations, we evaluate them on a single attack run (one random seed) of 5000 images.

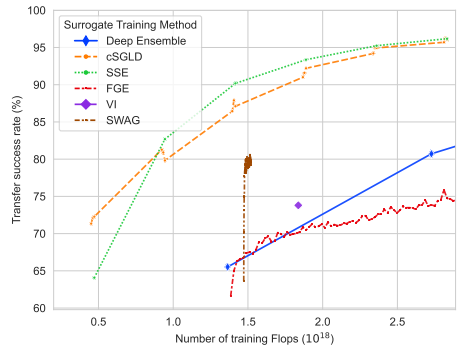
Deep Ensemble. Deep Ensemble [Lakshminarayanan et al., 2016] simply trains several DNNs independently with random initialization and random subsampling (mini-batch on shuffled data in practice). All DNNs have the same standard hyperparameters for training. For classification, predictions of individual DNNs are averaged. We train 15 PreResNet110, 4 PreResNet164, 4 VGG16bn, 4 VGG19bn, and 4 WideResNet28x10 DNNs on CIFAR-10. We retrieve 15 ResNet50 DNNs trained by Ashukha et al. [2020] on ImageNet, and trained on our own 1 DNN for each of the remaining studied architectures (ResNeXt50 32x4d, DenseNet121, MNASNet 1.0, and EfficientNet-B0).

cSGLD. We refer the reader to the approach section for a detailed description of cyclical Stochastic Gradient Langevin Dynamics. Figure 3 joined as supplementary materials illustrates both the cyclical cosine annealing learning rate schedule and the separation of each cycle into an exploration phase (called the burn-in period of MCMC algorithm) and a sampling phase.

SWAG. Stochastic Weight Averaging-Gaussian (SWAG) [Maddox et al., 2019] is a Bayesian Deep Learning method that fits a Gaussian onto SGD iterates to approximate the posterior distribution over weights. Its first moment is the SWA solution, and its second moment a diagonal plus low-rank covariance matrix. Both are estimated from SGD iterates with constant learning rate (0.001 on ImageNet and 0.01 on CIFAR-10). On ImageNet, SWAG performs 10 additional epochs to collect SGD iterates from one of the Deep Ensemble DNNs. On CIFAR-10, a regular pre-training phase of 160 epochs precedes 140 epochs to collect checkpoints. Once fitted, models are sampled from the Gaussian distribution. For every sample, batch normalization statistics are updated in a forward pass over the entire CIFAR-10 train set and over a random subset of 10% on ImageNet. Apart from the fixed initial cost, the marginal computational cost to obtain a sample is very low. We sample a maximum of 50 models because iterative attacks perform 50 iterations of one model per iteration, and further samples would be discarded. Thus, the lines corresponding to SWAG in Figures 2 and 5 are shorter than the ones of other methods. The rank of the estimated covariance matrix is 20. Batch-size is 128 on CIFAR-10, and 256 on ImageNet.



(a) CIFAR-10



(b) ImageNet

Figure 5: Intra-architecture L∞ I-FG(S)M success rate with respect to the training computational complexity of 6 Bayesian and Ensemble methods. Every curve starts with one model, and each successive point is obtained by forming an ensemble with one more model.

VI. Variational Inference (VI) approximates the true posterior distribution with a variational approximation, here a fully-factorized Gaussian distribution, and maximizes a corresponding lower bound. A Gaussian prior is chosen. Once trained, the variational approximation is used as the posterior. There is no additional sampling phase to perform Bayesian model averaging. Therefore, we cannot tune the number of samples and a single VI point is plotted in Figures 2 and 5. We follow the solutions of Ashukha et al. [2020] to avoid underfitting: pre-training and annealing of β. The first moment of the Gaussian variational approximation is initially set to a DNN pre-trained similarly to Deep Ensemble

(300 epochs on CIFAR-10 with initial learning rate of 10^{-4}, and 130 epochs on ImageNet starting at 10^{-3}). The log of its second moment is initially set to -5 on CIFAR-10 and -6 on ImageNet, and further optimized for 100 epochs (45 on ImageNet) with Adam and a learning rate of 10^{-4}. β is set to 10^{-5} on CIFAR-10 and 10^{-4} on ImageNet. Batch-size is 128 on CIFAR-10, and 256 on ImageNet.

SSE. Snapshot ensembles technique [Huang et al., 2017] is the foundation of cSGLD. The learning rate is cyclical with a cosine annealing schedule. Contrary to cSGLD, SSE saves a single snapshot per cycle and does not add gradient noise. The cycles are 40 epochs long on CIFAR-10, 45 on ImageNet. The maximum learning rate is 0.2, batch size is 64 on CIFAR-10, respectively 0.1 and 256 on ImageNet.

FGE. Fast Geometric Ensembling [Garipov et al., 2018] is a method developed after the empirical observation of Mode Connectivity on CIFAR-10 and CIFAR-100: it's possible to find a path in the parameters space that connects two independently trained DNNs such that the models along the path have low loss and high test accuracy. In practice, it uses a cyclical triangular learning rate and collects one model during each cycle. It is quite similar to SSE, except for the learning rate schedule, the much shorter cycles (4 epochs on CIFAR-10, 2 epochs on ImageNet), and a pre-training phase. Pre-training lasts for 160 epochs on CIFAR-10. On ImageNet, FGE is initialized from one Deep Ensemble checkpoint. The learning rate varies between 5×10^{-5} and 5×10^{-3} on CIFAR-10 and 10^{-6} and 10^{-4} on ImageNet. Batch-size is 128 on CIFAR-10, and 256 on ImageNet.

Table 5: Top-1 accuracy on natural test examples for every target deterministic Neural Network.

Dataset	Target DNN	Benign Test Accuracy
CIFAR-10	PreResNet110	93.26 %
	PreResNet164	93.03 %
	VGG16bn	83.68 %
	VGG19bn	83.62 %
	WideResNet28x10	92.13 %
ImageNet	ResNet50	76.15 %
	ResNeXt50 32x4d	77.62 %
	Densenet121	74.65 %
	MNASNet 1.0	73.51 %
	EfficientNet-B0	77.70 %

Table 6: Proportion of gradients with an L2 norm lower than 10^{-8}, the numerical tolerance used by the library Adversarial Robustness Toolbox in L2 attacks to avoid division by zero in norm scaling. Gradients are of the models of RQ1 (PreResNet110 for CIFAR-10, ResNet-50 for ImageNet) and on 10 000 original examples of the test sets. Means and standard deviations of 15 models are reported.

Dataset	Surrogate	Vanishing gradients
CIFAR-10	cSGLD	10.93% ±1.31
	DNN	64.96% ±0.56
ImageNet	cSGLD	0.06% ±0.06
	DNN	0.11% ±0.03

Table 7: Intra-architecture transfer success rates of 4 attacks on PreResNet110 (CIFAR-10) and ResNet50 (ImageNet). Bold is best. Higher is better.

Dataset	Attack	Surrogate	L2 Attack	L∞ Attack	Nb training epochs	Nb backward passes
CIFAR10	I-FG(S)M	cSGLD	**92.38% ±0.23**	92.74% ±0.33	300	50
		1 DNN	43.17% ±0.97	77.59% ±0.01	300	50
		2 DNNs	52.08% ±1.03	84.75% ±0.20	600	50
		5 DNNs	58.74% ±0.98	94.81% ±0.17	1500	50
		15 DNNs	62.08% ±0.92	**97.83% ±0.03**	4500	50
	MI-FG(S)M	cSGLD	92.29% ±0.25	94.20% ±0.14	300	50
		1 DNN	72.34% ±0.23	80.43% ±0.04	300	50
		2 DNNs	84.10% ±0.33	90.70% ±0.07	600	50
		5 DNNs	91.66% ±0.26	97.04% ±0.07	1500	50
		15 DNNs	**93.87% ±0.30**	**98.30% ±0.11**	4500	50
	PGD (5 restarts)	cSGLD	**91.65% ±0.33**	92.10% ±0.25	300	250
		1 DNN	51.08% ±0.10	77.58% ±0.38	300	250
		2 DNNs	60.60% ±0.06	83.67% ±0.27	600	250
		5 DNNs	67.55% ±0.21	94.19% ±0.07	1500	250
		15 DNNs	70.42% ±0.23	**97.37% ±0.06**	4500	250
	FG(S)M	cSGLD	**43.13% ±0.00**	58.85% ±0.01	300	30
		1 DNN	20.92% ±0.00	38.89% ±0.01	300	1
		2 DNNs	23.75% ±0.00	45.83% ±0.01	600	2
		5 DNNs	25.60% ±0.00	54.62% ±0.01	1500	5
		15 DNNs	26.71% ±0.00	**61.81% ±0.00**	4500	15
ImageNet	I-FG(S)M	cSGLD	94.41% ±0.46	90.77% ±0.09	225	50
		1 DNN	64.95% ±0.54	57.79% ±0.17	130	50
		2 DNNs	80.39% ±0.83	74.25% ±0.71	260	50
		5 DNNs	94.53% ±0.43	92.81% ±0.45	650	50
		15 DNNs	**98.51% ±0.11**	**98.28% ±0.16**	1950	50
	MI-FG(S)M	cSGLD	93.42% ±0.73	93.61% ±0.41	225	50
		1 DNN	61.11% ±0.35	63.70% ±0.21	130	50
		2 DNNs	77.93% ±0.44	79.27% ±0.76	260	50
		5 DNNs	94.41% ±0.47	95.32% ±0.25	650	50
		15 DNNs	**98.89% ±0.13**	**99.19% ±0.13**	1950	50
	PGD (5 restarts)	cSGLD	91.81% ±0.38	88.76% ±0.24	225	250
		1 DNN	57.47% ±0.52	53.79% ±0.45	130	250
		2 DNNs	74.04% ±0.47	70.90% ±0.41	260	250
		5 DNNs	91.99% ±0.41	91.27% ±0.59	650	250
		15 DNNs	**97.83% ±0.20**	**97.65% ±0.21**	1950	250
	FG(S)M	cSGLD	58.91% ±0.11	67.17% ±0.26	225	15
		1 DNN	37.37% ±0.19	44.55% ±0.72	130	1
		2 DNNs	46.73% ±0.34	53.91% ±0.60	260	2
		5 DNNs	58.17% ±0.18	65.53% ±0.10	650	5
		15 DNNs	**68.48% ±0.52**	**76.57% ±0.62**	1950	15

Table 8: Inter-architecture transfer success rates of I-FG(S)M of single architecture surrogate on CIFAR-10. All combinations of surrogate and targeted architectures are evaluated. Diagonals are intra-architecture. Symbols $\star$ indicate 1 DNN having higher transferability than cSGLD. 1 DNN and cSGLD have similar computation budget (300 epochs). Bold is best. Higher is better.

Attack	Surrogate	Target	PreResNet110	PreResNet164	VGG16bn	VGG19bn	WideResNet
L2	PreResNet110	cSGLD	**88.96% ±0.02**	**88.57% ±0.00**	26.18% ±0.02	24.38% ±0.00	**63.35% ±0.01**
		1 DNN	34.42% ±0.00	34.39% ±0.01	12.66% ±0.01	12.54% ±0.00	26.29% ±0.00
		4 DNNs	50.50% ±0.00	50.49% ±0.00	**27.45% ±0.01**	**27.30% ±0.00**	46.10% ±0.00
	PreResNet164	cSGLD	**88.28% ±0.01**	**87.52% ±0.01**	25.83% ±0.01	23.64% ±0.01	**62.79% ±0.01**
		1 DNN	33.89% ±0.00	34.36% ±0.01	11.93% ±0.00	12.07% ±0.01	25.95% ±0.01
		4 DNNs	50.36% ±0.01	50.45% ±0.00	**26.79% ±0.01**	**27.13% ±0.00**	45.94% ±0.00
	VGG16bn	cSGLD	**69.22% ±0.06**	**69.03% ±0.03**	43.70% ±0.04	38.54% ±0.02	**55.62% ±0.07**
		1 DNN	27.22% ±0.04	27.23% ±0.05	29.28% ±0.08	28.73% ±0.02	22.22% ±0.00
		4 DNNs	55.14% ±0.06	54.96% ±0.04	**73.65% ±0.00**	**71.24% ±0.04**	44.89% ±0.09
	VGG19bn	cSGLD	**69.82% ±0.05**	**68.27% ±0.07**	44.59% ±0.10	39.76% ±0.13	**54.40% ±0.08**
		1 DNN	18.09% ±0.10	18.09% ±0.06	$\star$44.63% ±0.03	$\star$46.76% ±0.03	14.38% ±0.03
		4 DNNs	34.30% ±0.06	33.77% ±0.01	**66.20% ±0.03**	**68.87% ±0.05**	27.44% ±0.02
	WideResNet	cSGLD	**82.25% ±0.03**	**85.06% ±0.02**	**26.34% ±0.08**	**23.81% ±0.03**	**69.31% ±0.07**
		1 DNN	22.14% ±0.01	23.00% ±0.00	9.43% ±0.00	9.54% ±0.00	26.85% ±0.00
		4 DNNs	41.07% ±0.00	41.75% ±0.04	22.91% ±0.04	22.65% ±0.03	43.00% ±0.01
L∞	PreResNet110	cSGLD	88.70% ±0.00	88.48% ±0.01	26.32% ±0.00	24.27% ±0.01	62.95% ±0.01
		1 DNN	72.73% ±0.00	74.57% ±0.00	22.26% ±0.00	20.98% ±0.00	47.59% ±0.01
		4 DNNs	**91.98% ±0.00**	**92.25% ±0.00**	**38.24% ±0.00**	**35.56% ±0.00**	**72.64% ±0.01**
	PreResNet164	cSGLD	87.99% ±0.01	87.74% ±0.00	26.33% ±0.00	23.67% ±0.01	61.83% ±0.02
		1 DNN	68.97% ±0.01	71.76% ±0.00	20.29% ±0.00	18.86% ±0.00	45.07% ±0.00
		4 DNNs	**90.67% ±0.00**	**92.22% ±0.00**	**37.62% ±0.00**	**35.23% ±0.00**	**73.18% ±0.00**
	VGG16bn	cSGLD	**66.97% ±0.13**	**67.48% ±0.11**	42.91% ±0.05	37.91% ±0.02	**50.52% ±0.01**
		1 DNN	35.57% ±0.02	35.89% ±0.03	38.35% ±0.00	35.82% ±0.00	26.77% ±0.02
		4 DNNs	52.59% ±0.00	53.12% ±0.00	**70.89% ±0.00**	**68.53% ±0.00**	41.34% ±0.00
	VGG19bn	cSGLD	**67.11% ±0.00**	**66.55% ±0.02**	43.50% ±0.01	38.72% ±0.02	**49.69% ±0.02**
		1 DNN	20.50% ±0.02	20.97% ±0.00	$\star$45.90% ±0.02	$\star$48.60% ±0.02	16.37% ±0.01
		4 DNNs	32.43% ±0.06	32.25% ±0.04	**63.11% ±0.07**	**65.64% ±0.06**	25.34% ±0.02
	WideResNet	cSGLD	**81.99% ±0.01**	**85.63% ±0.01**	27.04% ±0.02	23.46% ±0.01	68.43% ±0.01
		1 DNN	49.24% ±0.16	52.84% ±0.03	20.23% ±0.04	18.53% ±0.02	60.84% ±0.09
		4 DNNs	77.45% ±0.01	79.55% ±0.13	**36.33% ±0.13**	**33.60% ±0.22**	**83.24% ±0.00**

Table 9: Inter-architecture transfer success rates of I-FG(S)M of single architecture surrogate on ImageNet. All combinations of surrogate and targeted architectures are evaluated. Diagonals are intra-architecture. 1 DNN and cSGLD have similar computation budget (135 epochs). Bold is best. Higher is better.

Attack	Surrogate	Target	ResNet50	ResNeXt50	DenseNet121	MNASNet	EfficientNetB0
L2	ResNet50	cSGLD	**84.93% ±0.59**	**74.70% ±0.91**	**71.32% ±0.63**	**60.09% ±0.60**	**39.70% ±0.29**
		1 DNN	56.98% ±0.62	41.13% ±0.97	29.81% ±0.33	27.90% ±0.43	16.39% ±0.46
	ResNeXt50	cSGLD	**79.25% ±0.24**	**77.34% ±0.39**	**68.53% ±0.19**	**62.16% ±0.19**	**43.51% ±0.62**
		1 DNN	37.48% ±0.52	36.35% ±0.22	23.77% ±0.41	23.69% ±0.21	14.32% ±0.24
	DenseNet121	cSGLD	**63.23% ±1.16**	**59.89% ±1.12**	**73.28% ±0.45**	**60.84% ±0.33**	**40.27% ±0.44**
		1 DNN	32.61% ±0.29	32.06% ±0.61	39.18% ±0.47	32.01% ±0.44	17.72% ±0.49
	MNASNet	cSGLD	**7.81% ±0.19**	**5.97% ±0.37**	**9.81% ±0.31**	30.41% ±1.45	**15.46% ±0.44**
		1 DNN	7.04% ±0.51	5.29% ±0.36	8.41% ±0.20	**32.65% ±0.22**	13.13% ±0.06
	EfficientNetB0	cSGLD	**18.93% ±2.17**	**14.16% ±1.69**	**19.89% ±1.21**	**65.97% ±3.60**	**49.41% ±3.64**
		1 DNN	15.15% ±0.30	13.33% ±0.33	16.12% ±0.71	58.73% ±0.25	48.85% ±0.56
L∞	ResNet50	cSGLD	**78.67% ±1.19**	**65.21% ±1.48**	**61.54% ±0.83**	**51.75% ±1.39**	**31.11% ±1.13**
		1 DNN	48.03% ±0.94	32.17% ±0.43	23.37% ±0.34	22.60% ±0.40	12.59% ±0.21
	ResNeXt50	cSGLD	**71.67% ±1.00**	**69.33% ±0.85**	**59.18% ±1.14**	**54.75% ±1.33**	**35.13% ±0.71**
		1 DNN	31.19% ±0.42	28.68% ±0.76	19.12% ±0.07	19.53% ±0.51	11.20% ±0.33
	DenseNet121	cSGLD	**54.13% ±1.70**	**50.66% ±1.62**	**65.80% ±0.66**	**53.43% ±1.30**	**32.49% ±0.36**
		1 DNN	25.49% ±0.81	23.73% ±0.59	30.78% ±0.21	26.05% ±0.66	13.41% ±0.20
	MNASNet	cSGLD	**6.77% ±0.29**	4.72% ±0.27	**8.26% ±0.36**	25.27% ±1.83	**12.21% ±0.84**
		1 DNN	6.52% ±0.23	**5.06% ±0.12**	7.83% ±0.13	**29.19% ±0.05**	11.13% ±0.16
	EfficientNetB0	cSGLD	**17.81% ±1.58**	**13.91% ±1.45**	**19.71% ±1.29**	**63.67% ±3.16**	46.91% ±3.44
		1 DNN	15.83% ±0.32	13.51% ±0.52	16.78% ±0.38	60.14% ±0.37	**50.16% ±0.64**

Table 10: Transfer success rates of (M)I-FG(S)M attack improved by our approach combined with test-time transformations on CIFAR-10. Columns are targets. PreResNet110 columns are intra-architecture transferability, others are inter-architecture. Bold is best. Symbols ⋆ are DNN-based techniques better than our vanilla cSGLD surrogate, and † are techniques that doesn't improve the corresponding vanilla surrogate. The success rate for every cSGLD-based technique is better than its counterpart with 1 DNN.

Attack	Surrogate	PreResNet110	PreResNet164	VGG16bn	VGG19bn	WideResNet
L2	1 DNN	34.42% ±0.00	34.39% ±0.01	12.67% ±0.00	12.54% ±0.00	26.29% ±0.01
	+ Input Diversity	59.63% ±0.80	59.79% ±0.75	24.37% ±0.16	23.25% ±0.12	46.09% ±0.47
	+ Skip Gradient Method	57.00% ±0.00	57.66% ±0.04	20.87% ±0.03	20.10% ±0.09	41.80% ±0.04
	+ Ghost Networks	79.22% ±0.30	80.38% ±0.16	⋆32.03% ±0.25	⋆28.63% ±0.17	56.65% ±0.24
	+ Momentum	67.12% ±0.07	67.80% ±0.00	20.49% ±0.02	19.15% ±0.01	44.11% ±0.04
	+ Input Diversity	81.44% ±0.32	82.69% ±0.29	27.64% ±0.03	25.82% ±0.42	57.29% ±0.12
	+ Skip Gradient Method	73.52% ±0.00	75.23% ±0.01	24.52% ±0.00	22.76% ±0.00	49.73% ±0.00
	+ Ghost Networks	77.44% ±0.28	79.13% ±0.12	⋆28.98% ±0.57	25.74% ±0.18	54.06% ±0.04
	cSGLD	90.67% ±0.39	89.74% ±0.31	28.05% ±0.33	26.12% ±0.14	67.27% ±0.89
	+ Input Diversity	92.45% ±0.14	91.80% ±0.14	33.69% ±0.28	31.35% ±0.28	72.41% ±0.76
	+ Skip Gradient Method	92.46% ±0.17	92.10% ±0.28	31.96% ±0.53	29.84% ±0.34	71.04% ±1.23
	+ Ghost Networks	**92.73% ±0.21**	**92.20% ±0.07**	**36.17% ±0.39**	**33.08% ±0.32**	**74.77% ±0.10**
	+ Momentum	†90.35% ±0.37	89.77% ±0.28	†26.89% ±0.37	†25.02% ±0.29	†65.98% ±0.52
	+ Input Diversity	92.31% ±0.33	91.58% ±0.23	31.92% ±0.49	29.72% ±0.46	70.94% ±0.31
	+ Skip Gradient Method	92.33% ±0.34	91.94% ±0.41	31.95% ±0.29	29.85% ±0.28	70.96% ±0.65
	+ Ghost Networks	92.42% ±0.16	91.93% ±0.25	33.02% ±0.60	29.77% ±0.14	72.28% ±0.53
L∞	1 DNN	72.73% ±0.00	74.58% ±0.01	22.26% ±0.00	20.98% ±0.00	47.59% ±0.01
	+ Input Diversity	81.29% ±0.18	82.77% ±0.12	28.10% ±0.22	26.17% ±0.25	57.04% ±0.10
	+ Skip Gradient Method	77.92% ±0.00	79.50% ±0.01	27.43% ±0.00	25.31% ±0.01	53.39% ±0.00
	+ Ghost Networks	74.92% ±0.08	77.23% ±0.26	⋆29.61% ±0.19	26.31% ±0.30	52.93% ±0.05
	+ Momentum	76.12% ±0.01	78.05% ±0.00	23.77% ±0.02	22.33% ±0.01	50.49% ±0.01
	+ Input Diversity	84.66% ±0.19	86.38% ±0.12	⋆31.47% ±0.05	⋆28.89% ±0.31	61.60% ±0.16
	+ Skip Gradient Method	79.72% ±0.02	80.80% ±0.02	28.75% ±0.01	26.12% ±0.00	55.74% ±0.00
	+ Ghost Networks	80.34% ±0.34	82.59% ±0.42	⋆34.17% ±0.48	⋆29.37% ±0.18	60.62% ±0.40
	cSGLD	90.98% ±0.40	90.26% ±0.35	29.26% ±0.53	26.97% ±0.43	67.18% ±1.03
	+ Input Diversity	92.46% ±0.14	91.62% ±0.16	33.81% ±0.25	30.84% ±0.34	71.15% ±0.92
	+ Skip Gradient Method	93.38% ±0.50	92.84% ±0.25	35.68% ±0.61	32.43% ±0.52	73.55% ±1.08
	+ Ghost Networks	91.66% ±0.40	91.32% ±0.19	34.77% ±0.09	31.01% ±0.27	71.60% ±0.40
	+ Momentum	92.84% ±0.18	92.18% ±0.28	32.03% ±0.49	28.53% ±0.38	71.56% ±0.25
	+ Input Diversity	94.05% ±0.31	93.53% ±0.21	37.31% ±0.38	33.23% ±0.23	75.40% ±0.25
	+ Skip Gradient Method	**94.64% ±0.26**	**94.29% ±0.31**	**38.08% ±0.27**	**34.28% ±0.17**	**76.62% ±0.50**
	+ Ghost Networks	93.76% ±0.14	93.75% ±0.13	38.01% ±0.44	33.15% ±0.36	76.23% ±0.29

Table 11: Hyperparameters used to train cSGLD or Deep Ensemble. The $\star$ symbols refer to the inter-architecture and test-time techniques sections, and $\star\star$ to the Bayesian and Ensemble training methods section. We do not include target DNNs on ImageNet, since they are pretrained models from PyTorch and timm.

Method	Hyperparameter	CIFAR-10 cSGLD	CIFAR-10 DNN Surrogate	CIFAR-10 DNN Target	ImageNet cSGLD	ImageNet DNN Surrogate
All	Number epochs	50 per cycle	300	300	45 per cycle	130 (135 for $\star$)
	Initial learning rate	0.5	0.01	0.01	0.1	0.1
	Learning rate schedule	Cosine Annealing	Step size decay ($\times$0.1 each 75 epochs)	Step size decay ($\times$0.1 each 75 epochs)	Cosine Annealing	Step size decay ($\times$0.1 each 30 epochs)
	Optimizer	cSGLD	Adam	Adam	cSGLD	SGD
	Momentum	0	0.9	0.9	0.9	0.9
	Weight decay	5e-4 (3e-4 for PreResNet)	1e-4	1e-4	1e-4	1e-4
	Batch-size	64	128	128	256 for ResNet50, 64 for others	256 for ResNet50, 64 for others
cSGLD	Sampling interval	1 sample per epoch	-	-	1 sample per epoch	-
	Nb cycles	6 (18 for $\star\star$)	-	-	5 (3 for $\star$, 6 for $\star\star$)	-
	Nb samples per cycle	5	-	-	3	-
	Nb epochs with noise	5	-	-	3	-

Table 12: Hyperparameters of attacks and test-time transferability techniques.

Attack / Technique	Hyperparameter	CIFAR-10	ImageNet
All attacks	Perturbation 2-norm ε	0.5	3
	Perturbation ∞-norm ε	$\frac{4}{255}$	$\frac{4}{255}$
Iterative Attacks	Step-size α	$\frac{\varepsilon}{10}$	$\frac{\varepsilon}{10}$
	Number iterations	50	50
MI-FG(S)M	Momentum term	0.9	0.9
PGD	Number random restarts	5	5
Ghost Network	Skip connection erosion random range	[1-0.22, 1+0.22]	[1-0.22, 1+0.22]
Input Diversity	Minimum resize ratio	90 %	90 %
	Probability transformation	0.5	0.5
Skip Gradient Method	Residual Gradient Decay γ	0.7 (PreResNet110)	0.2 (ResNet50)

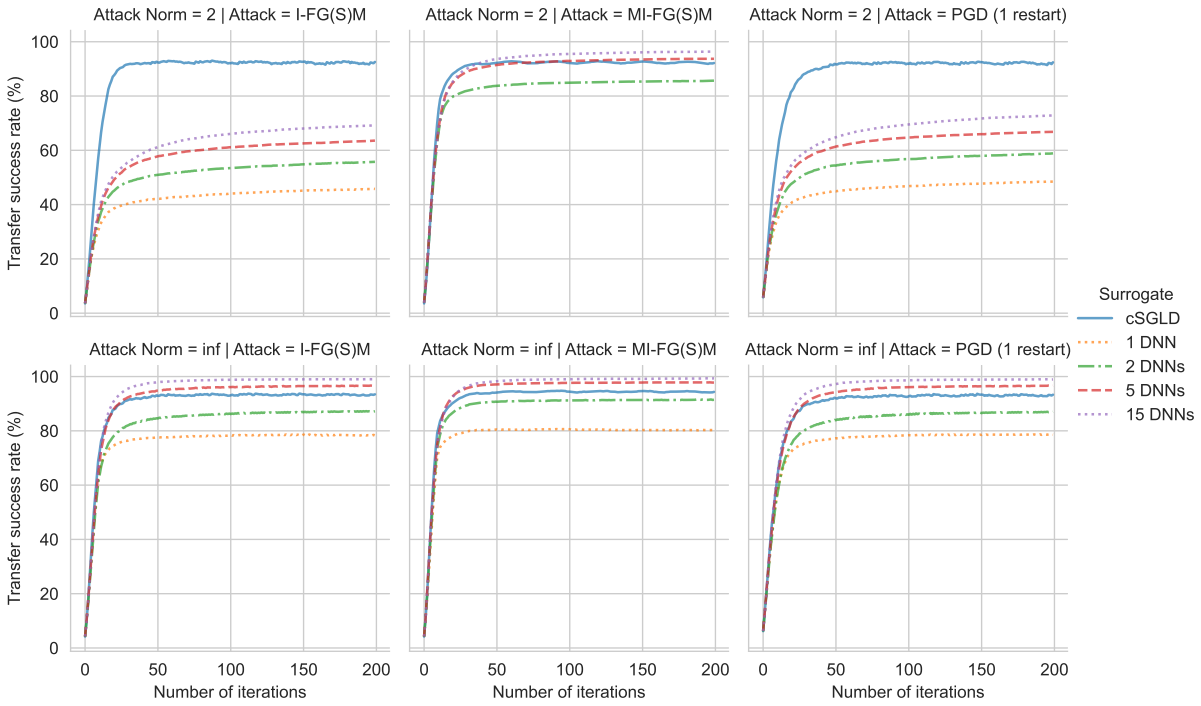


Figure 6: Transfer success rates on CIFAR-10 of three iterative gradient-based attacks on the same architecture (PreResNet110) with respect to the number of iterations.

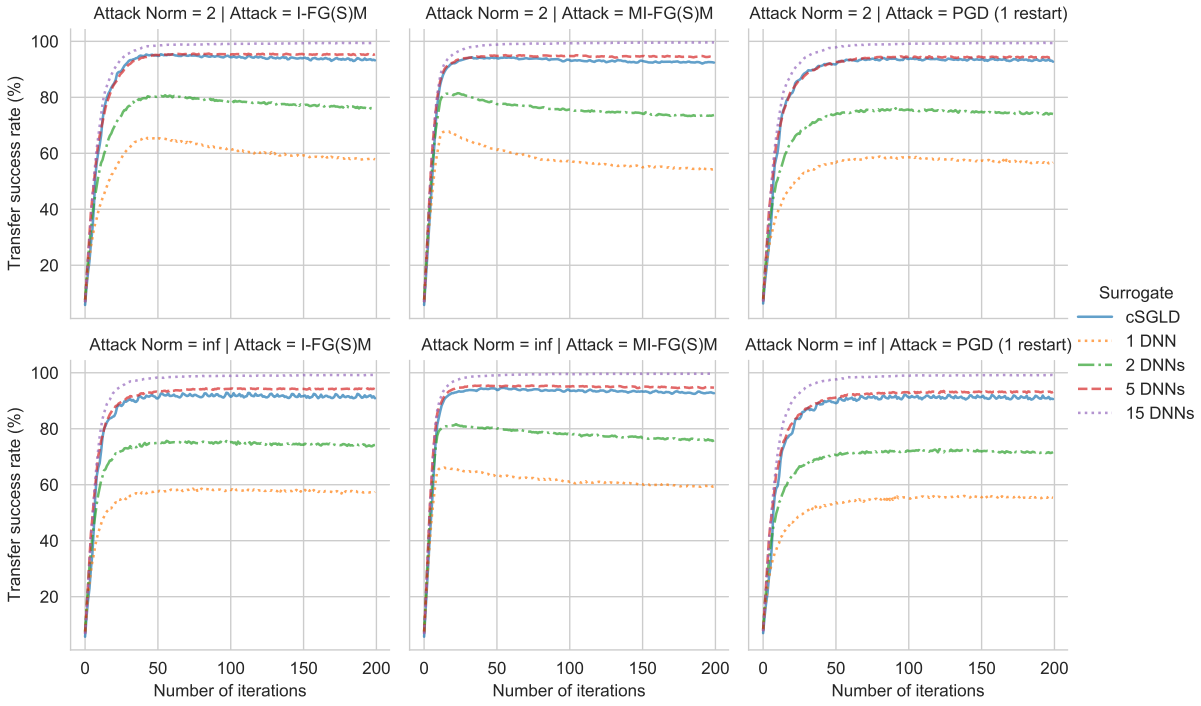


Figure 7: Transfer success rates on ImageNet of three iterative gradient-based attacks on the same architecture (ResNet-50) with respect to the number of iterations.

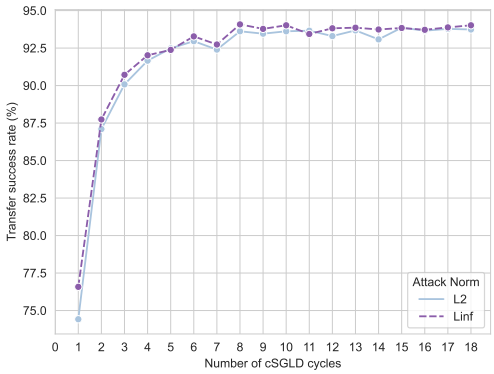


Figure 8: Intra-architecture transfer success rate of I-FG(S)M with respect to the number of cSGLD cycles on CIFAR-10 (PreResNet110).

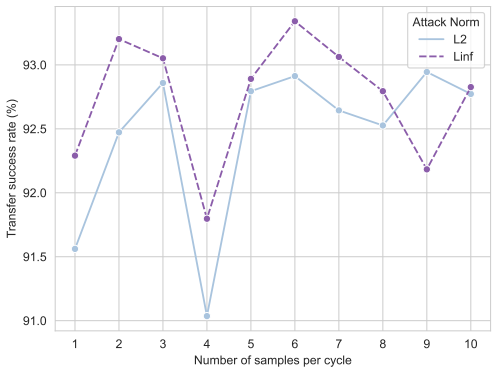


Figure 9: Intra-architecture transfer success rate of I-FG(S)M with respect to the number of cSGLD samples per cycle. We train one PreResNet110 cSGLD on CIFAR-10 for every number of cycles, from 1 to 10 samples per cycle. Each additional sample per cycle increases the training cost by 1 epoch per cycle (starting at 48 epochs per cycle). A fixed number of 5 cSGLD cycles is used.

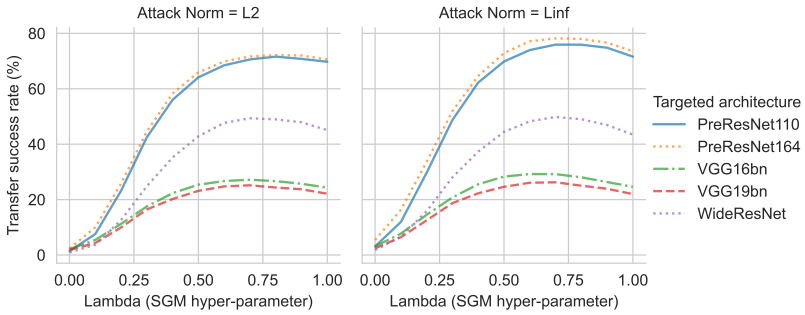


Figure 10: Transfer success rates of the test-time transferability technique Skip Gradient Method with varying values of its hyperparameter γ between 0 and 1 with 0.1 steps. The surrogate is a PreResNet110 DNN trained on CIFAR-10 and evaluated on 1 independently trained DNN for every targeted architecture. The plain line represents the intra-architecture transferability, and the dotted ones the inter-architecture transferability. Adversarial examples are crafted from a validation set randomly sampled from the train set. $\gamma = 0.7$ is selected in the rest of the paper for PreResNet110.