Parsimonious Black-Box Adversarial Attacks via Efficient Combinatorial Optimization

Seungyong Moon [*] [1] [2] **Gaon An** [*] [1] [2] **Hyun Oh Song** [1] [2]

Abstract

Solving for adversarial examples with projected gradient descent has been demonstrated to be highly effective in fooling the neural network based classifiers. However, in the black-box setting, the attacker is limited only to the query access to the network and solving for a successful adversarial example becomes much more difficult. To this end, recent methods aim at estimating the true gradient signal based on the input queries but at the cost of excessive queries. We propose an efficient discrete surrogate to the optimization problem which does not require estimating the gradient and consequently becomes free of the first order update hyperparameters to tune. Our experiments on Cifar-10 and ImageNet show the state of the art black-box attack performance with significant reduction in the required queries compared to a number of recently proposed methods. The source code is available at `https://github.com/snu-mllab/parsimonious-blackbox-attack`.

1. Introduction

Understanding the vulnerability of neural network based classifiers to adversarial perturbations (Szegedy et al., 2013; Carlini et al., 2016) designed to fool the classifier predictions have emerged as an important research area in machine learning. Recent studies have devised highly successful attacks in the *white-box* setting (Goodfellow et al., 2014; Madry et al., 2017; Carlini & Wagner, 2017), where the attacker has access to the network model parameters and the corresponding loss gradient with respect to the input perturbation. However, in more realistic settings (Watson visual recognition, 2019; Google vision API, 2019; Clarifai API, 2019), the attacker is limited to the access of input queries and the corresponding model predictions. These web services offer various commercial recognition services such as image classification, content moderation, and face recognition backed up by pretrained neural network based classifiers. In this setting, the inference network receives the query image from the user and only exposes the inference results, so the model weights are hidden from the user.

To this end, *black-box* methods construct adversarial perturbations without utilizing the model parameters or the gradient information. Some recent works on black-box attacks compute the gradient using a substitute network (Papernot et al., 2016; 2017) but it has been shown that the method does not always transfer to the target network. On the other hand, another line of works aim at estimating the gradient based on the model predictions from the input queries and apply first order updates with the estimated gradient vector (Chen et al., 2017; Tu et al., 2018; Bhagoji et al., 2018; Ilyas et al., 2018a;b). However, the robustness of this approach can be susceptible to the choice of the hyperparameters such as the learning rate, decay rates, and the update rule since the performance of the method hinges on the first order updates with approximated ascent directions.

We first consider a discrete surrogate problem which finds the solution among the *vertices* of the ℓ_∞ ball and show that this unlocks a new class of algorithms which constructs adversarial perturbations without the need to estimate the gradient vector. This comes with the benefit that the algorithm becomes free of the update hyperparameters and thus is more applicable in black-box settings. Intuitively, our proposed method defines and maintains upper bounds on the *marginal gain* of attack locations, and this leads to a parsimonious characteristic of the algorithm to terminate quickly without having to perform excessive queries.

Our results on Cifar-10 (Krizhevsky & Hinton, 2009) and ImageNet (Russakovsky et al., 2015) show the state of the art attack performance under ℓ_∞ noise constraint demonstrating significantly higher attack success rates while making considerably less function queries compared to the recent baseline methods (Chen et al., 2017; Tu et al., 2018;

[*] Equal contribution [1] Department of Computer Science and Engineering, Seoul National University, Seoul, Korea [2] Neural Processing Research Center. Correspondence to: Hyun Oh Song <hyunoh@snu.ac.kr>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

Ilyas et al., 2018a;b) in both the untargeted and targeted black-box attack settings. Notably, our method achieves attack success rate comparable to the white-box PGD attack (Madry et al., 2017) in some settings (attacks on the adversarially trained network on Cifar-10), although the method uses more queries being a black-box attack method.

2. Related works

There has been a line of work on adversarial attacks after the recent discovery of network vulnerability to the attacks from Biggio et al. (2012); Szegedy et al. (2013). In our paper, we focus on black-box attacks under ℓ_∞ constraint with access to the network prediction scores only. Although attacks on more limited settings (access to the network decision only) have been explored (Brendel et al., 2017; Ilyas et al., 2018a; Cheng et al., 2018), the methods typically require up to $1M$ queries per images which can be difficult to apply in practical settings.

Black-box attacks with substitute networks Papernot et al. (2016; 2017) utilize separate substitute networks trained to match the prediction output of the target network similar to model distillation (Hinton et al., 2015). The idea then is to craft adversarial perturbations by using the backpropagation gradients from the substitute network and transfer the adversarial noise to the target network. The follow-up work from Liu et al. (2016) showed that black-box attacks with substitute networks tend not to transfer well for targeted attacks but can be improved with ensemble networks. However, the attack success rates for these methods are outperformed by another line of techniques which directly estimate the gradient of the target network based on the input queries.

Black-box attacks with gradient estimation Chen et al. (2017) computes the coordinate-wise numerical gradient of the target network by repeatedly querying for the central difference values *at each pixels per each ascent steps*. This can be prohibitive as it would require approximately half million queries on moderate sized images. Bhagoji et al. (2018) mitigates the issue by grouping the pixels at random or via PCA but still requires computing the group-wise numerical gradients per each ascent steps.
In contrast, Ilyas et al. (2018a); Tu et al. (2018) compute the vector-wise gradient estimate with random vector u_i by computing $\frac{1}{\sigma n}\sum_i^n(f(x+\sigma u_i)-f(x-\sigma u_i))u_i$. Ilyas et al. (2018b) extends the approach to incorporate time-dependent prior which acts similar to the momentum term (Nesterov, 1983) in first order optimization and data-dependent prior which exploits the spatial regularity (also in Tu et al. (2018)) for query efficiency.

3. Methods

Suppose we have a classifier with a corresponding loss function $\ell(x, y)$. In black box attacks, the goal is to craft imperceptible adversarial perturbations (x_{adv}) typically under small ℓ_∞ radius in a limited query budget. Furthermore, the attacker only has access to the loss function (zeroth order oracle). This is a challenging setup as the attacker does not have access to the gradient information (first order oracle) with respect to the input.

3.1. Problem formulation

First order methods tend to have strong attack performance by formulating a constrained optimization problem and querying the gradient of the loss with respect to the input perturbation. Fast gradient sign method (FGSM) (Goodfellow et al., 2014) first derives the following first order Taylor approximation to the loss function. Note, PGD is a multi-step variant of FGSM (Madry et al., 2017).

$$\ell(x_{adv}, y) \approx \ell(x, y) + (x_{adv} - x)^\intercal \nabla_x \ell(x, y)$$

Then, the optimization problem becomes,

$$\underset{\|x_{adv}-x\|_\infty \le \epsilon}{\text{maximize}} \;\; \ell(x_{adv}) \implies \underset{x_{adv}}{\text{maximize}} \;\; {x_{adv}}^\intercal \nabla_x \ell(x, y) \quad (1)$$
$$\text{subject to} \quad -\epsilon\mathbf{1} \preceq x_{adv} - x \preceq \epsilon\mathbf{1},$$

where $\mathbf{1}$ denotes the vector of ones, $\preceq$ denotes the element-wise inequality. Thus, we can interpret FGSM as finding the solution to the above linear program (LP) in Equation (1) with the gradient vector evaluated at the original image x as the cost vector in LP. Similarly, we can interpret that PGD sequentially finds the solution to the above LP at each step with the updated gradient vector $\nabla_{x_{adv}}\ell(x_{adv}, y)$ as the cost vector.

Since the feasible set in Equation (1) is bounded, the solution of the LP is attained at an extreme point of the feasible set (Schrijver, 1986), and we can theoretically characterize that an optimal solution will be attained at a vertex of the ℓ_∞ ball. Figure 1 shows an example statistics of the adversarial noise $(x_{adv} - x)$ obtained by running the PGD algorithm until convergence on Cifar-10 dataset. This shows that the empirical solution from PGD is mostly found on vertices of ℓ_∞ ball as well. We also found that running PGD until convergence on ImageNet dataset also produces similar results.

This characterization together with the fact that in many realistic scenarios, the access to the true gradient is not readily available, motivates us to consider a discrete surrogate to the problem as shown in Equation (2).

$$\underset{x_{adv}\in\mathbb{R}^p}{\text{maximize}} \;\; f(x_{adv}) \qquad \implies \qquad \underset{x_{adv}}{\text{maximize}} \;\; f(x_{adv}) \qquad (2)$$
$$\text{subject to} \;\; \|x_{adv} - x\|_\infty \le \epsilon \qquad \text{subject to} \;\; x_{adv} - x \in \{\epsilon, -\epsilon\}^p,$$

where p denotes the number of pixels in the image x, $f(x) = \ell(x, y_{gt})$ for untargeted attacks with ground truth label y_{gt}, and $f(x) = -\ell(x, y_{target})$ for targeted attacks with target label y_{target}.

Optimizing the discrete surrogate problem does not require estimating the gradient and thus removes all the hyperparameters crucial for gradient update based attacks (Ilyas et al., 2018a;b; Chen et al., 2017; Tu et al., 2018). Furthermore, we show in Section 3.4 how the surrogate exploits the underlying problem structure for faster convergence (early-termination in black-box attacks).

Equivalently, the discrete problem in Equation (2) can be reformulated as the following set maximization problem in Equation (3).

$$\underset{\mathcal{S}\subseteq\mathcal{V}}{\text{maximize}} \left\{ F(\mathcal{S}) \triangleq f\left(x + \epsilon \sum_{i\in\mathcal{S}} e_i - \epsilon \sum_{i\notin\mathcal{S}} e_i \right) \right\}, \tag{3}$$

where e_i denotes the i-th standard basis vector, $\mathcal{V}$ denotes the ground set which is the set of all pixel locations ($|\mathcal{V}| = p$), $\mathcal{S}$ denotes the set of *selected* pixels with $+\epsilon$ perturbations, and $\mathcal{V} \setminus \mathcal{S}$ indicates the set of remaining pixels with $-\epsilon$ perturbations. The goal in Equation (3) is to find the set of pixels $\mathcal{S}$ with $+\epsilon$ perturbations (and vice versa for $\mathcal{V} \setminus \mathcal{S}$) which will maximize the objective function. However, finding the exact solution to the problem is NP-Hard as naïve exhaustive solution requires combinatorial ($2^{|\mathcal{V}|}$) queries. Subsequent subsections discuss how we exploit the underlying problem structure for efficient computation.

3.2. Approximate submodularity

In general, maximizing functions over sets is usually NP-hard (Krause & Golovin, 2014; Bach et al., 2013). However, many set functions that arise in machine learning problems often exhibit *submodularity*.

Definition 1. *For a set function $F : 2^{\mathcal{V}} \to \mathbb{R}, \mathcal{S} \subseteq \mathcal{V}$, and $e \in \mathcal{V}$, let $\Delta(e \mid \mathcal{S}) := F(\mathcal{S} \cup \{e\}) - F(\mathcal{S})$ be the marginal gain (Krause & Golovin, 2014) of F at $\mathcal{S}$ with respect to e.*

Definition 2. *A function $F : 2^{\mathcal{V}} \to \mathbb{R}$ is submodular if for every $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and $e \in \mathcal{V} \setminus \mathcal{B}$ it holds that*

$$\Delta(e \mid \mathcal{A}) \geq \Delta(e \mid \mathcal{B})$$

Intuitively, submodular functions exhibit a diminishing returns property where the marginal gain diminishes as the augmenting set size increases. In the context of machine learning problems, the implication of submodularity is that we can efficiently compute an approximately optimal solution to the submodular set functions with a suite of greedy style algorithms. Furthermore, submodularity allows us to establish $(1 - \frac{1}{e})$-approximation (Nemhauser et al., 1978) for monotone submodular functions and $\frac{1}{3}$-approximation (Feige et al., 2011) for non-monotone submodular functions.

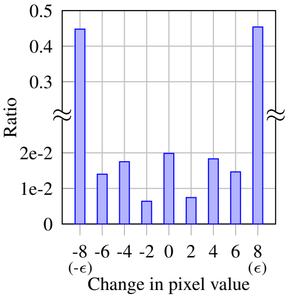


Figure 1. Distribution of adversarial noise with white box PGD attack on Cifar-10 dataset with wide Resnet w32-10 adversarially trained network at ℓ_∞ ball radius $\epsilon = 8$ in [0, 255] scale.

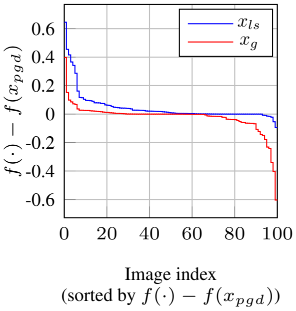


Figure 2. $f(\cdot) - f(x_{pgd})$ values on random 100 samples on the same experiment setting as Figure 1, where x_{ls}, x_g and x_{pgd} each denotes image perturbed by local search, greedy insertion, and PGD method.

Unfortunately, we can construct a counterexample showing $F(\mathcal{S})$ is not submodular. Let $f(x) = -\log\left(\frac{1}{1+e^{-w^\intercal x}}\right)$, $w = (-1, -1)^\intercal$, $\epsilon = 1$, and $x = (0, 0)^\intercal$. For $\mathcal{A} = \emptyset$, $\mathcal{B} = \{1\}$, and $e = 2$ the inequality on Definition 2 does not hold because $0.57 = \Delta(e \mid \mathcal{A}) < \Delta(e \mid \mathcal{B}) = 1.43$, which implies $F(\mathcal{S})$ is not submodular. Regardless, if the submodularity is not severely deteriorated (Zhou & Spanos, 2016), submodular maximization algorithms still work to a substantial extent. Figure 2 shows the result of running local search (Feige et al., 2011) and greedy insertion (Krause & Golovin, 2014) in comparison to the function value of the solution obtained by white box PGD method on Cifar-10 dataset. The plots show that greedy insertion solution, in general, has comparable objective value to PGD solution and that local search solution shows slightly higher value than PGD solution.

For non-monotone set functions, local search type algorithms achieve better theoretical and practical solutions than greedy insertion algorithms by providing modifications to the working set with alternating insertion and deletion processes. In the following subsection, we establish the approximation bound for performing improved local search procedure for non-monotone approximate submodular functions and then propose an accelerated variant of the algorithm for practical black-box adversarial attack. We first make a slight detour and introduce the local search procedure and a proof of local optimality.

3.3. Local search optimization for black-box attack on approximately submodular functions

Local search algorithm (Feige et al., 2011) alternates between greedily inserting an element while the marginal gain is strictly positive ($\Delta(e \mid \mathcal{S}) > 0$) and removing an element while the marginal gain is also strictly positive. Feige et al. (2011) showed that once the algorithm converges it converges to a *local optimum of the set function F*. When the set function is submodular, the local search solution has $\frac{1}{3}$-approximation with respect to the optimal solution.

Definition 3. *Given a set function F, a set $\mathcal{S}$ is a local optimum, if $F(\mathcal{S}) \geq F(\mathcal{S} \setminus \{a\})$ for any $a \in \mathcal{S}$ and $F(\mathcal{S}) \geq F(\mathcal{S} \cup \{a\})$ for any $a \notin \mathcal{S}$.*

Lemma 1. *Let $\mathcal{S}$ be the solution obtained by performing the local search algorithm. Then $\mathcal{S}$ is a local optima.*

Proof. See supplementary A.1. □

Note that Lemma 1 holds regardless of submodularity. The following theorem states an approximation bound for the local search solution for approximately submodular set functions. Note, we assume non-negativity[1] of the set function for the proof.

Theorem 1. *Let $\mathcal{C}$ be an optimal solution for a function F and $\mathcal{S}$ be the solution obtained by the local search algorithm. Then,*

$$2F(\mathcal{S}) + F(\mathcal{V} \setminus \mathcal{S}) \geq F(\mathcal{C}) + \xi\lambda_F(\mathcal{V}, 2),$$

where

$$\xi = \binom{|\mathcal{S} \setminus \mathcal{C}|}{2} + \binom{|\mathcal{C} \setminus \mathcal{S}|}{2} + |\overline{\mathcal{S} \cup \mathcal{C}}| \cdot |\mathcal{S}| + |\mathcal{C} \setminus \mathcal{S}| \cdot |\mathcal{S} \cap \mathcal{C}|$$

Proof. See supplementary A.2. □

Finally, from Theorem 1, we get the following corollary.

Corollary 1. *If $F(\mathcal{C}) + \xi\lambda_F(\mathcal{V}, 2) \geq 0$, one of the following holds*

1. $F(\mathcal{S}) \geq \frac{1}{3}\Big(F(\mathcal{C}) + \xi\lambda_F(\mathcal{V}, 2)\Big)$
2. $F(\mathcal{V} \setminus \mathcal{S}) \geq \frac{1}{3}\Big(F(\mathcal{C}) + \xi\lambda_F(\mathcal{V}, 2)\Big)$

As the set function becomes submodular, the submodularity index (λ_F) becomes close to zero (Zhou & Spanos, 2016), recovering the $\frac{1}{3}$-approximation bound (Feige et al., 2011). From the local search algorithm, we obtain a set $\mathcal{S}$ of pixels to perturb the input image x with $+\epsilon$ and the complement set $\mathcal{V} \setminus \mathcal{S}$ of pixels to perturb with $-\epsilon$. Concretely, the perturbed image is computed as $x_{adv} \triangleq x + \epsilon \sum_{i \in \mathcal{S}} e_i - \epsilon \sum_{i \notin \mathcal{S}} e_i$. Figure 3 shows some examples of the perturbed images produced by the algorithm.

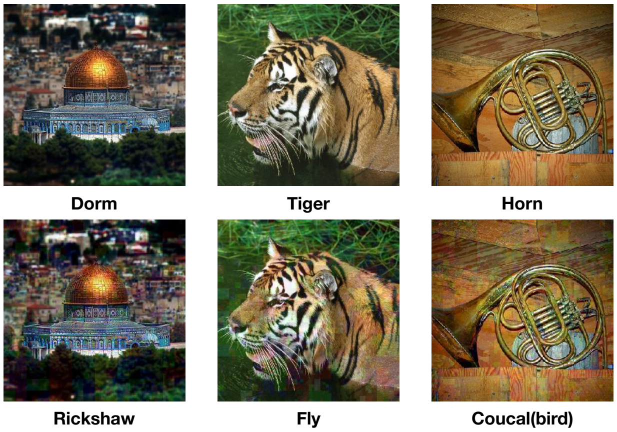


Figure 3. Adversarial examples from ImageNet in the targeted setting. The top row shows the original images and the bottom row shows the corresponding perturbed images from our method.

3.4. Acceleration with lazy evaluations

Naïvely applying the local search algorithm (Feige et al., 2011) for black-box adversarial attack poses a challenge because each calls to the greedy insertion or deletion algorithms make $O(|\mathcal{V}| \cdot |\mathcal{S}|)$ queries (since at each step greedy algorithm finds the element that maximizes the marginal gain) and would become impractical for query limited black-box attacks which we are interested in.

Algorithm 1 Lazy Greedy Insertion

```
input  Objective set function $F$, Working set $\mathcal{S}$, Ground set $\mathcal{V}$
initialize  Max heap $Q = \emptyset$
for each element $e \in \mathcal{V} \setminus \mathcal{S}$ do
    Calculate $\Delta(e \mid \mathcal{S}) := F(\mathcal{S} \cup \{e\}) - F(\mathcal{S})$
    Push $(e, \Delta(e \mid \mathcal{S}))$ into $Q$
end for
while $|Q| > 0$ do
    Pop the top element $\hat{e}$ from $Q$, update its upper bound $\rho(\hat{e})$
    Peek the top element $\tilde{e}$ and its upper bound $\rho(\tilde{e})$ in $Q$
    if $\rho(\hat{e}) > \rho(\tilde{e})$ and $\rho(\hat{e}) > 0$ then
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{\hat{e}\}$
    else if $\rho(\hat{e}) > \rho(\tilde{e})$ and $\rho(\hat{e}) \leq 0$ then
        break
    else
        Push $(\hat{e}, \rho(\hat{e}))$ into $Q$
    end if
end while
output  $\mathcal{S}$;
```

Thus, we employ the accelerated greedy algorithm often called Lazy-Greedy (Minoux, 1978). Instead of computing the marginal gain $\Delta(e \mid \mathcal{S}_i)$ for each item $e \in \mathcal{V} \setminus \mathcal{S}_i$ at each iteration $i + 1$, the algorithm keeps an upper bound $\rho(e)$ on the marginal gain for each item in a max-heap. In each iteration $i + 1$, it evaluates the marginal gain for the top element $\hat{e}$ in the heap and updates the upper bound $\rho(\hat{e}) := \Delta(\hat{e} \mid \mathcal{S}_i)$. If $\rho(\hat{e}) \geq \rho(e) \;\; \forall e$, then submodularity

[1] A standard trick in discrete optimization is to add a constant offset term to the set function.

guarantees that $\hat{e}$ is the element with the largest marginal gain. If the top element does not satisfy this condition, the algorithm inserts it again into the heap with the updated upper bound.

While the number of function evaluations in the worst case is the same as the standard greedy algorithm, the algorithm provides *several orders of magnitude speedups* in practice (Leskovec et al., 2007; Gomez-Rodriguez et al., 2012; Lin & Bilmes, 2011; Wei et al., 2013; Mirzasoleiman et al., 2013). Note, however, if strict upper bound on function evaluation is required, another variant, Stochastic-Greedy (Mirzasoleiman et al., 2015) offers a stochastic algorithm with linear time upper bound on function evaluations ($|\mathcal{V}| \log \frac{1}{\theta}$).

Algorithm 2 Lazy Greedy Deletion

```
input Objective set function $F$, Working set $\mathcal{S}$
initialize Max heap $Q = \emptyset$
for each element $e \in \mathcal{S}$ do
    Calculate $\Delta^-(e \mid \mathcal{S}) := F(\mathcal{S} \setminus \{e\}) - F(\mathcal{S})$
    Push $(e, \Delta^-(e \mid \mathcal{S}))$ into $Q$
end for
while $|Q| > 0$ do
    Pop the top element $\hat{e}$ from $Q$, update its upper bound $\rho(\hat{e})$
    Peek the top element $\tilde{e}$ and its upper bound $\rho(\tilde{e})$ in $Q$
    if $\rho(\hat{e}) > \rho(\tilde{e})$ and $\rho(\hat{e}) > 0$ then
        $\mathcal{S} \leftarrow \mathcal{S} \setminus \{\hat{e}\}$
    else if $\rho(\hat{e}) > \rho(\tilde{e})$ and $\rho(\hat{e}) \leq 0$ then
        break
    else
        Push $(\hat{e}, \rho(\hat{e}))$ into $Q$
    end if
end while
output $\mathcal{S}$;
```

Algorithm 3 Accelerated Local Search w/ Lazy Evaluations

```
input Objective set function $F$, Working set $\mathcal{S}$, Ground set $\mathcal{V}$
for $t = 1, \ldots,$ MAXITER do
    Insert elements of $\mathcal{V}$ into $\mathcal{S}$ using Algorithm 1
    $\mathcal{S} \leftarrow \textsc{LazyGreedyInsertion}(F, \mathcal{S}, \mathcal{V})$
    Delete elements from $\mathcal{S}$ using Algorithm 2
    $\mathcal{S} \leftarrow \textsc{LazyGreedyDeletion}(F, \mathcal{S})$
end for
output $\operatorname*{argmax}_{\mathcal{A} \in \{\mathcal{S}, \mathcal{V} \setminus \mathcal{S}\}} F(\mathcal{A})$;
```

The resulting algorithm for performing local search optimization with lazy evaluations is presented in Algorithm 3. The lazy insertion and deletion algorithms are presented in Algorithm 1 and Algorithm 2 respectively.

3.5. Hierarchical lazy evaluation

Most of the recent black-box approaches (Chen et al., 2017; Tu et al., 2018; Ilyas et al., 2018b) exploit the spatial regularities inherent in images for query efficiency. The underlying idea is that natural images exhibit locally regular structure (Huang & Mumford, 1999) and is far from random matrices of numbers. Tu et al. (2018); Ilyas et al. (2018b) exploit the idea to estimate the gradient in blocks of pixels and perform interpolation to compute the full gradient image.

Algorithm 4 Split Block

```
input Set of blocks $\mathcal{B}$, Block size $k$
initialize $\mathcal{B}' = \emptyset$
for each block $b \in \mathcal{B}$ do
    Split the block $b$ into 4 blocks $\{b_1, b_2, b_3, b_4\}$ with size $k/2$
    $\mathcal{B}' \leftarrow \mathcal{B}' \cup \{b_1, b_2, b_3, b_4\}$
end for
output $\mathcal{B}'$;
```

Algorithm 5 Hierarchical Accelerated Local Search

```
input Objective set function $F$, Block size $k$, Ground set $\mathcal{V}$ of
    size $|\mathcal{V}| = h/k \times w/k \times c$ where the image size is $h \times w \times c$
initialize Working set $\mathcal{S} = \emptyset$
repeat
    Run Local Search Algorithm on $\mathcal{S}$ and $\mathcal{V}$ using Algorithm 3
    $\mathcal{S} \leftarrow \textsc{LocalSearch}(F, \mathcal{S}, \mathcal{V})$
    if $k > 1$ then
        Split the blocks into finer blocks using Algorithm 4
        $\mathcal{S} \leftarrow \textsc{SplitBlock}(\mathcal{S}, k)$, $\mathcal{V} \leftarrow \textsc{SplitBlock}(\mathcal{V}, k)$
        $k \leftarrow k/2$
    end if
until $F$ converges
output $\mathcal{S}$;
```

We take a hierarchical approach and perform the accelerated local search in Algorithm 3 on a coarse grid (large blocks) and use the results to define the initial working set in the subsequent rounds on finer grid (smaller blocks) structure. Figure 4 illustrates the process for several rounds. Algorithm 5 shows the overall hierarchical accelerated local search algorithm. The ground set $\mathcal{V}$ now becomes the set of all blocks, not pixels. It is important to note that most of the attacks terminate in early stages and rarely run until the very fine scales. Even in the case when the algorithm proceeds into finer granularities, the algorithm pre-terminates when it reaches the maximum allowed query limit following the experimental protocol in Ilyas et al. (2018a;b).

4. Implementation details

We assume only the cross-entropy loss is available by model access, following Ilyas et al. (2018a;b). On Cifar-10, We set the initial block size to $k = 4$. On ImageNet, we set the initial block size to $k = 32$. To make blocks divisible by 2, we set the noise size to be 256×256 and resize it to the image size (299×299) using nearest neighbor interpolation. Since our method runs in query-limited setting, we fix MAXITER (in Algorithm 3) to 1, reducing unnecessary query counts for calculating marginal gains.

Our method needs $O(\frac{|\mathcal{V}|}{k^2})$ queries for calculating the initial marginal gains. This can consume excessive amount of queries before the actual perturbation. To address this, instead of running the algorithm on the whole ground set

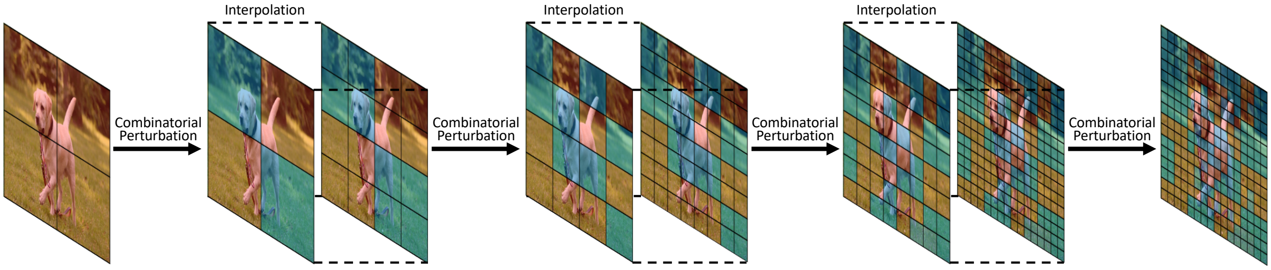

Interpolation
Interpolation
Interpolation
Combinatorial Perturbation
Combinatorial Perturbation
Combinatorial Perturbation
Combinatorial Perturbation

Figure 4. Illustration of hierarchical lazy evaluation process. Blue area represents the blocks currently in the working set ($\mathcal{S}$), while red area represents blocks outside the working set ($\mathcal{V} \setminus \mathcal{S}$).

$\mathcal{V}$, we split $\mathcal{V}$ into a partition of mini-batches $\{\mathcal{V}_i\}_{i=1}^n$ and split the working set $\mathcal{S}$ into $\{\mathcal{S}_i\}_{i=1}^n$ where $\mathcal{S}_i = \mathcal{S} \cap \mathcal{V}_i$. Then we update $\mathcal{S}_i$ subject to $\mathcal{V}_i$ for $i = 1, ..., n$ sequentially. Concretely, at i-th step we insert elements of $\mathcal{V}_i$ into $\mathcal{S}_i$ and remove the elements from the updated $\mathcal{S}_i$. We fix the mini-batch size to 64 throughout all the experiments.

5. Experiments

We evaluate the performance comparing against the NES method (Ilyas et al., 2018a) and the Bandits method (Ilyas et al., 2018b), which is the current state of the art in black-box attacks, on both untargeted and targeted attack settings. We consider the ℓ_∞ threat models on Cifar-10 and ImageNet datasets and quantify the performance in terms of success rate, average queries, and median queries. We further investigate the average queries on samples that NES, the weakest attack among the baselines, successfully fooled. Note that this is a fairer measure for evaluating an attack method's performance, since naïve average query measure can be affected by the method's success rate. More accurate methods can be disadvantaged by successfully fooling more difficult images slightly below the max query budget. We also show the white-box PGD results from Madry et al. (2017) as the *upper bound* experiment.

5.1. Experiments on Cifar-10

To evaluate the effectiveness of the method on the adversarially trained network, which is known to be robust to adversarial perturbations, we tested the attacks on wide Resnet w32-10 classifier (Zagoruyko & Komodakis, 2016) adversarially trained on Cifar-10 dataset (Madry et al., 2017). We use the pretrained network provided by MadryLab[2]. We then use 1,000 randomly selected images from the validation set that are initially correctly classified. We set the maximum distortion of the adversarial image to $\epsilon = 8$ in $[0, 255]$ scale, following the experimental protocol in Madry et al. (2017); Athalye et al. (2018); Bhagoji et al. (2018). We restrict the maximum number of queries to 20,000.

We run 20 iterations of PGD with constant step size of 2.0, as done in Madry et al. (2017). We performed grid search for hyperparameters in NES and Bandits. For NES, we tuned $\sigma \in \{0.0001, 0.001, 0.01\}$, size of NES population $n \in \{50, 100, 200\}$, learning rate $\eta \in \{0.001, 0.005, 0.01\}$, and momentum $\beta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. For Bandits, we tuned for OCO learning rate $\eta \in \{0.01, 0.1, 1, 10, 100\}$, image learning rate $h \in \{0.001, 0.005, 0.01\}$, bandit exploration $\delta \in \{0.01, 0.1, 1\}$, and finite difference probe $\eta \in \{0.01, 0.1, 1\}$. The hyperparameters used are listed in supplementary B.1.

The results are presented in Table 1 and Figure 5a. We found that our algorithm has about 10% higher success rate than Bandits, with 33% less average queries. Notably, this success rate is higher than the white-box PGD method. The efficiency of our algorithm is more evident in average queries on samples that NES successfully fooled. On this measure, our method needs 53% less queries on average compared to Bandits, and 91% less compared to NES.

Method	Success rate	Avg. queries	Med. queries	Avg. queries (NES success)
PGD (white-box)	47.2%	20	-	-
NES	29.5%	2872	900	2872
Bandits	38.6%	1877	459	520
Ours	**48.0%**	**1261**	**356**	**247**

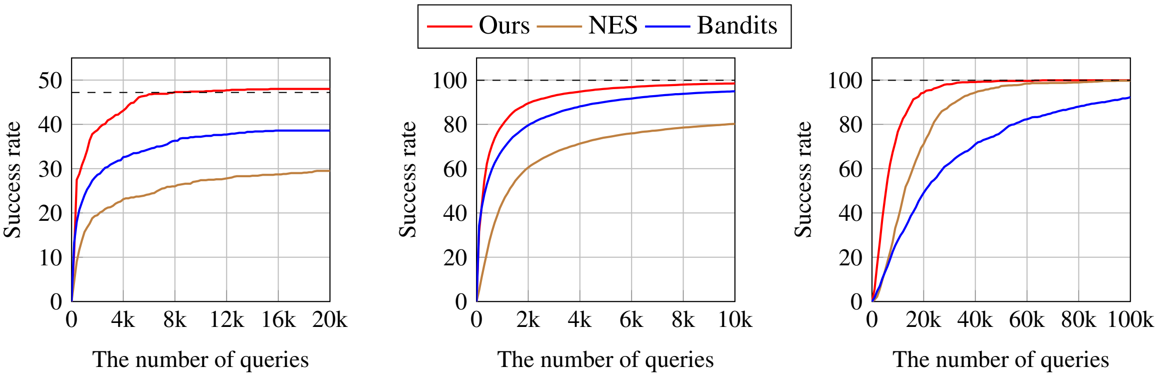
Table 1. Results for ℓ_∞ untargeted attacks on Cifar-10. Maximum number of queries set to 20,000.

5.2. Untargeted attacks on ImageNet

On ImageNet, we attack the pretrained Inception v3 classifier from Szegedy et al. (2015) provided by Tensorflow[3]. We use 10,000 randomly selected images (scaled to [0, 1]) that are initially correctly classified. We set ϵ to 0.05 and the maximum queries to 10,000, as done in Ilyas et al. (2018b).

[2] https://github.com/MadryLab/cifar10_challenge

[3] https://github.com/tensorflow/models/tree/master/research/slim



(a) Cifar-10, untargeted (b) ImageNet, untargeted (c) ImageNet, targeted

Figure 5. The cumulative distribution of the number of queries required for (a) untargeted attack on Cifar-10, (b) untargeted attack on ImageNet, and (c) targeted attack on ImageNet. The dashed line indicates the success rate of white-box PGD. The results show that our method consistently finds successful adversarial images faster than the baseline methods.

We ran PGD for 20 steps at the constant step size of 0.01. We communicated with the authors of Ilyas et al. (2018b), and the authors provided the up-to-date hyperparameters for Bandits. Hyperparameters for NES were referred from the paper. NES† and Bandits† denote the results copied from the paper (Ilyas et al., 2018b) for comparison. The hyperparameters used are listed in supplementary B.2.

The results are presented in Table 2 and Figure 5b. We found that our method again outperforms other black-box methods by a significant margin. Our method achieves about 4% higher success rate than Bandits, with 30% less queries. Also, note that our method requires 38% less average queries on samples that NES successfully attacked than Bandits.

Method	Success rate	Avg. queries	Med. queries	Avg. queries (NES success)
PGD (white-box)	99.9 %	20	-	-
NES†	77.8%	1735	-	1735
NES	80.3%	1660	900	1660
Bandits†	95.4%	1117	-	703
Bandits	94.9%	1030	286	603
Ours	**98.5%**	**722**	**237**	**376**

Table 2. Results for ℓ_∞ untargeted attacks on ImageNet. Maximum number of queries set to 10,000.

5.3. Targeted attacks on ImageNet

For targeted attacks, we use the same Inception v3 network used in the untargeted attack setting. We attack 1,000 randomly selected images (scaled to [0, 1]) that are initially correctly classified. Targeted classes were chosen randomly for each image, and each attack method chose the same target classes for the same images for fair comparison. We limit ϵ to 0.05 and the maximum number of queries to 100,000.

We ran PGD for 200 steps at the constant step size of 0.001. Hyperparameters for NES were adjusted from Ilyas et al. (2018a) to match the results in the paper (given in supplementary B.3). Since Ilyas et al. (2018b) does not report targeted attack for Bandits, we performed hyperparameter tuning for Bandits. We tuned for the image learning rate $h \in \{0.0001, 0.001, 0.005, 0.01, 0.05\}$ and OCO learning rate $\eta \in \{1, 10, 100, 1000\}$. Details of the experiment's results can be found in supplementary C. NES†[4] indicates the result reported in the paper (Ilyas et al., 2018a) for comparison. The results are presented in Table 3 and Figure 5c. We can see that our method achieves a higher success rate (near 100%), with about 55% less queries than NES.

Method	Success rate	Avg. queries	Med. queries	Avg. queries (NES success)
PGD (white-box)	100%	200	-	-
NES†	99.2%	-	11550	-
NES	99.7%	16284	12650	16284
Bandits	92.3%	26421	18642	26421
Ours	**99.9%**	**7485**	**5373**	**7371**

Table 3. Results for ℓ_∞ targeted attacks on ImageNet. Maximum number of queries set to 100,000.

5.4. Untargeted attacks on ImageNet with smaller ϵ

To evaluate the performance of our method in a more constrained perturbation limit, we conducted experiments on ImageNet with the maximum perturbation $\epsilon \in \{0.01, 0.03\}$. The experiments are done in the untargeted attack setting. We restrict the maximum number of queries to be 10,000.

For NES and Bandits, which are gradient estimation based algorithms, the hyperparameters could be sensitive to the change of ϵ. For this reason, we re-tuned for the hy-

[4] In the original paper, the query limit was 1,000,000.

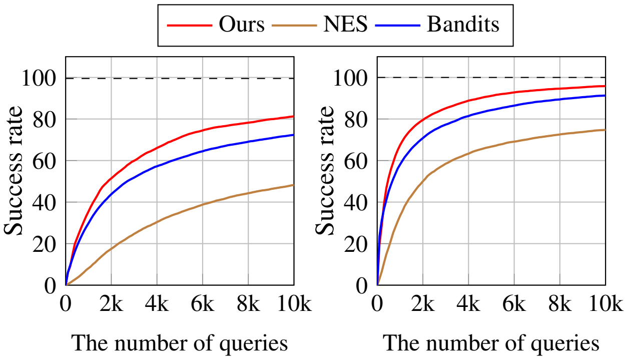


Figure 6. The cumulative distribution for the number of queries required for untargeted attack on ImageNet with $\epsilon = 0.01$ (left) and $\epsilon = 0.03$ (right).

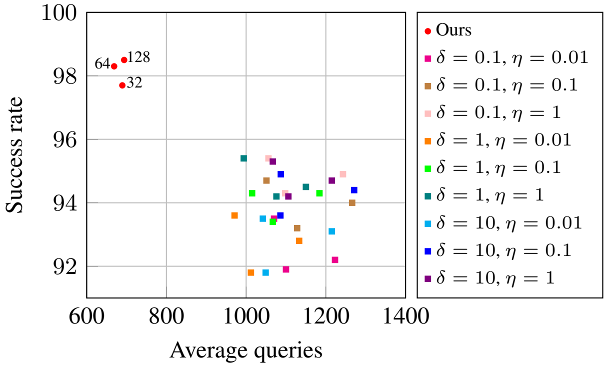


Figure 7. Success rate against the average number of queries with different hyperparameters. The square markers indicate the results of Bandits method. The numbers at round markers show the values of initial block size.

perparameters for these methods. For NES, we tuned for samples per step $n \in \{50, 100, 200\}$, finite difference probe $\eta \in \{0.01, 0.1, 1\}$, and learning rate $h \in \{0.001, 0.005, 0.01\}$. For Bandits, we tuned for OCO learning rate $\eta \in \{1, 10, 100\}$, image learning rate $h \in \{0.001, 0.005, 0.01\}$, bandit exploration $\delta \in \{0.01, 0.1, 1\}$, and finite difference probe $\eta \in \{0.01, 0.1, 1\}$. The hyperparameters used are listed in supplementary B.4.

The results are shown in Table 4 and Figure 6. We can see that, for all ϵ, our success rate is higher than the baseline methods. The results show that the margin in the success rate with respect to Bandits gets larger as ϵ decreases, up to 10%, while maintaining the query efficiency lead.

	Method	**Success rate**	**Avg. queries**	**Med. queries**	**Avg. queries (NES success)**
$\epsilon = 0.01$	PGD	99.5 %	20	-	-
	NES	48.2%	3598	3000	3598
	Bandits	72.4%	2318	1374	1052
	Ours	**81.3%**	**2141**	**1249**	**852**
$\epsilon = 0.03$	PGD	99.9 %	20	-	-
	NES	74.8%	2049	1200	2049
	Bandits	91.3%	1382	520	774
	Ours	**95.9%**	**1129**	**420**	**537**
$\epsilon = 0.05$	PGD	99.9 %	20	-	-
	NES	80.3%	1660	900	1660
	Bandits	94.9%	1030	286	603
	Ours	**98.5%**	**722**	**237**	**376**

Table 4. Results for ℓ_∞ untargeted attacks on ImageNet with $\epsilon \in \{0.01, 0.03, 0.05\}$.

5.5. Hyperparameter sensitivity

We measure the robustness of our method to hyperparameters compared to Bandits. Each method's robustness was measured by sweeping through hyperparameters and plotting their success rate and average queries, on ImageNet untargeted attack setting. For our method, we swept the initial block size $k \in \{32, 64, 128\}$. Note that k is the only hyperparameter for our method. For Bandits, we swept through bandit exploration $\delta \in \{0.1, 1, 10\}$, finite difference probe $\eta \in \{0.01, 0.1, 1\}$, and tile size $\in \{25, 50, 100\}$. Also note that Bandit still has two other hyperparameters, which were left as the original setting for the experiment.

The results are shown in Figure 7. The figure shows that the proposed method maintains a high success rate with low variance as the hyperparameter k changes. On the contrary, Bandits method shows relatively higher variance in both success rate and average queries. In our opinion, gradient estimation based methods, in general, are sensitive to the first order update hyperparameter settings as the ascent direction is approximated under limited query budget.

6. Conclusion

Motivated by the observation that finding an adversarial perturbation can be viewed as computing solutions of linear programs under bounded feasible set, we have developed a discrete surrogate problem for practical black-box adversarial attacks. In contrast to the current state of the art methods, our method does not require estimating the gradient vector and thus becomes free of the update hyperparameters. Our experiments show the state of the art attack success rate at significantly lower average/median/NES_success queries on both untargeted and targeted attacks on neural networks.

Acknowledgements

This work was partially supported by Samsung Advanced Institute of Technology and Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01367, BabyMind). Hyun Oh Song is the corresponding author.

References

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Bach, F. et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3), 2013.

Bhagoji, A. N., He, W., Li, B., and Song, D. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *ECCV*, 2018.

Biggio, B., Nelson, B., and Laskov, P. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017.

Carlini, N., Mishra, P., Vaidya, T., Zhang, Y., Sherr, M., Shields, C., Wagner, D., and Zhou, W. Hidden voice commands. In *USENIX Security Symposium*, 2016.

Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J. 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*. ACM, 2017.

Cheng, M., Le, T., Chen, P.-Y., Yi, J., Zhang, H., and Hsieh, C.-J. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457*, 2018.

Clarifai API. https://clarifai.com/, 2019.

Feige, U., Mirrokni, V. S., and Vondrak, J. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4), 2011.

Gomez-Rodriguez, M., Leskovec, J., and Krause, A. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data*, 5(4), 2012.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Google vision API. https://cloud.google.com/vision/, 2019.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Huang, J. and Mumford, D. Statistics of natural images and models. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1. IEEE, 1999.

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

Ilyas, A., Engstrom, L., and Madry, A. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv preprint arXiv:1807.07978*, 2018b.

Krause, A. and Golovin, D. Submodular function maximization, 2014.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., Van-Briesen, J., and Glance, N. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007.

Lin, H. and Bilmes, J. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011.

Liu, Y., Chen, X., Liu, C., and Song, D. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Minoux, M. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization techniques*. Springer, 1978.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, 2013.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. Lazier than lazy greedy. In *AAAI*, 2015.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functions. *Mathematical programming*, 14(1), 1978.

Nesterov, Y. E. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. In *Dokl. Akad. Nauk SSSR*, volume 269, 1983.

Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 2017.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

Schrijver, A. *Theory of Linear and Integer Programming*. Wiley-Interscience, New York, 1986.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*, 2015.

Tu, C.-C., Ting, P., Chen, P.-Y., Liu, S., Zhang, H., Yi, J., Hsieh, C.-J., and Cheng, S.-M. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. *arXiv preprint arXiv:1805.11770*, 2018.

Watson visual recognition. https://www.ibm.com/watson/services/visual-recognition/, 2019.

Wei, K., Liu, Y., Kirchhoff, K., and Bilmes, J. Using document summarization techniques for speech data subset selection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhou, Y. and Spanos, C. J. Causal meets submodular: Subset selection with directed information. In *NIPS*, 2016.