Emerging Black Box Attacks: A Case Study

Vignesh Sairaj: 305227643[1], Shruti Sharan: 405228029[1], and Brian Wang: 605229631[1]

[1]CS Department, UCLA

March 2019

1 Introduction

Studies have unveiled the vulnerabilities of well trained Deep Neural Networks by demonstrating the ability of generating barely noticeable adversarial images that lead to misclassification. This has prompted a significant body of work on the area of generating perturbed images that can fool classifiers, especially in the black box probability setting. In this setting, attackers have no knowledge of the model structure - they can only feed in information and retrieve the final scores for each class.

This area of research provides insight into building more secure classification systems. Internet services such as search engines and web content providers are supported heavily by content filters. These filters provide protection against undesirable data such as violent or disturbing content by determining what is appropriate. Adversarial attacks are built as a way to bypass these classification systems. Understanding how these attacks work provides insight into building more robust models that can filter out questionable content more accurately.

Emerging work such as [[2], [1], [3], [4], [6]] have explored black box adversarial attacks in this setting. In this report, we will be exploring three attack methods, namely GenAttack [3], ZOO Attack [2], and the Bandit Attack [6].

In this report, we evaluate these three attack methods on a common dataset (a subset of ImageNet) across three common metrics - attack success rate, runtime, and number of queries made through the model.

2 Background

In the context of images, black box adversarial attacks are concerned with creating imperceptible visual perturbations in an image so a model will classify this image as an incorrect class. A specific formulation is as follows:

Given an input image X_{orig}, a corresponding label y and a classification model G, we are interested in finding a perturbed version of the input image

X_{adv} such that $G(X_{adv}) = t$ where $t \neq y$.
We consider two types of black-box attacks. Given a sample with correct labelling, an untargeted attack refers to crafting an adversarial example leading to misclassification, whereas a targeted attack refers to modifying the example in order ti be classified as a desired class.

3 Related Work

To study robustness of DNNs earlier studies assume model transparency (white-box setting) that allows full control and access to the a targeted DNN for analysis. We studied the two most effective open box attacks FGSM [4] and Carlini and Wagner [1] attacks. FGSM uses the sign of the gradient from the back propagation on a targeted DNN to generate adversarial examples. C&W formulate a targeted adversarial attacks as an optimization problem, and use the L_2 norm to quantify the difference between the adversarial and original examples. The attack is driven by the representation of the logit layer of the targeted DNN. Most real world systems do not release the network architecture and internal configurations, therefore open box attacks cannot be used in practice. Hence we move onto the black-box setting.

4 Adversarial Attacks

We explored three effective adversarial attacks in the black box setting and summarized our findings as follows.

4.1 ZOO-Attack

Zeroth order optimization is a derivative free method [2]. The objective function $f(x)$ is evaluated at two very close points $f(x + hv)$ and $f(x - hv)$ with a small h, such that a proper gradient along the direction of vector v can be estimated. This is done using the finite difference method (instead of actual back-propagation).

$$\hat{g_i} := \frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + he_i) - f(x - he_i)}{2h} \tag{1}$$

$$\hat{h_i} := \frac{\partial^2 f(x)}{\partial x_{ii}^2} \approx \frac{f(x + he_i) - 2f(x) + f(x - he_i)}{h^2} \tag{2}$$

where h=1e-5 is a small constant and e_i is a standard basis vector with only the i-th component as 1. Thus both the gradient and Hessian can be computed without additional function evaluations.
The loss function used is a hinge-loss function based on the output of a DNN, which is defined as:

$$f(x, t) = max\{\underset{i \neq t}{max}\ log[F(x)]_i - log[F(x)]_t, -\kappa\} \qquad (targeted) \tag{3}$$

$$f(x) = max\{log[F(x)]_{t_0} - \max_{i \neq t_0} log[F(x)]_i, -\kappa\} \qquad (untargeted) \qquad (4)$$

where $\kappa \geq 0$ is a tuning parameter for attack transferability. t_0 is the original class label for image x and $max_{i \neq t} log[F(x)]_i$ represents the most probable predicted class other than t_0. The log operator is essential to our black-box attack since very often well trained DNNs yield skewed probability distributions such that the confidence score of one class dominates. Since log is a monotonic operator, it lessens the dominance effect.
Then stochastic coordinate descent algorithm is applied to the estimated gradients for optimization. In each iteration, one variable is randomly chosen and is updated by minimizing the objective function along that coordinate. In first-order methods we find that ZOO-ADAM significantly outperforms vanilla gradient descent update and its variants. ZOO-Newton is used for second-order update using the estimated gradients and Hessian computed using eq(1) and (2). If the Hessian is negative (concave) we update by gradient. Since this method can become computationally very expensive, several techniques including attack-space dimension reduction(reduce the permissible dimension of the adversarial noise without altering the dimension of input image), hierarchical attacks (gradually increase the dimension of the reduced transformation)and importance sampling (pixels closer to the main object are sampled first using max-pooling) are used.

4.2 Bandit Attack

As shown by Chen et al. [2], the first order update rule, despite having its origins in white-box attacks, prove to be a very effective black-box attack strategy and forms the basis for many algorithms ([2], [5], [6]).

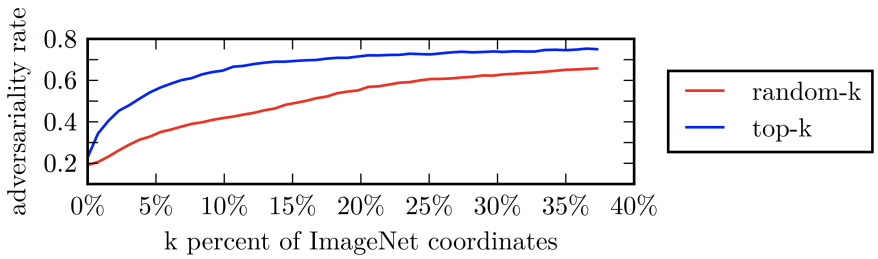

Figure 1: Adversality rate when a few (k) coordinates of the estimated gradient are set in the direction (sign) of the actual (backpropagated) gradient and the rest are randomly set to ± 1

Fig 1 shows that estimating the right signs of the gradient even along a few components is sufficient to generate adversarial examples in a few iterations. Ilyas et al. [6] have shown the optimality of existing algorithms (specifically,

[5]) by showing their equivalence to gradient estimation by least squares. However this optimality was proven in the absence of any prior information. After empirical observations made while running the NES [5] algorithm, certain correlations were observed between gradients across space and time:

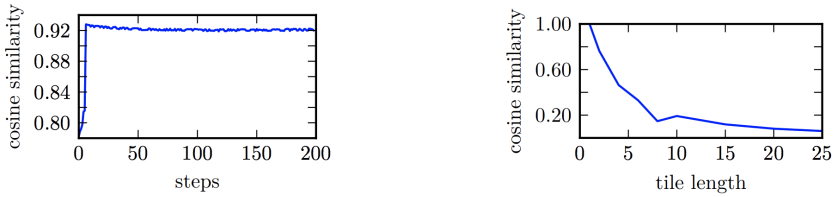

Figure 2: Cosine similarity between gradients across iteration steps (left) and between gradients in pixel locations in the image and a mean-pooled gradient across a window of size k (right) centered around the pixel vs k

As shown in 2, there is a very high correlation in time and space that is not being used in other approaches, i.e., gradients in successive iterations are highly correlated. Also, gradient components in a small k-window are also very similar (This is also the spatial structure exploited by CNNs to share parameters). In [6], Ilyas et al., show that the optimality established for NES [5] can be surpassed if this additional information is taken into account, i.e., the gradient can be better approximated with fewer queries. The authors do this by formulating the problem as a bandit optimization problem.

A bandit optimization problem is one where there is a loss function associated with every action (ℓ_t) in a sequence of actions, and the objective is the average loss across iterations. The loss ℓ_t is just the dot product of the estimated gradient with the true gradient (which isn't known but can be evaluated):

$$\ell_t(g) = -\langle \frac{g}{\|g\|}, \nabla L(x, y) \rangle \tag{5}$$

4.3 GenAttack

Unlike other adversarial attacks which require the calculation of a gradient in order to determine the "direction" to alter the image perturbations, GenAttack [3] uses a genetic algorithm to find the correct image perturbations for successfully attacking a model.

The attack wants to satisfy $argmax f(x_{adv})_c = t$ such that $||x - x_{adv}||_p \leq \delta$ where t is the desired label we wish to misclassify as. p is whatever L-p norm we wish to use, and δ is the maximum norm distance between the original and adversarial image.

At a closer level, the population of adversarial examples is initialized through applying independently and uniformly distributed random noise. Each member

of the population is evaluated for fitness via the following function:

$$ComputeFitness(x) = logf(x)_t - logmax_{c \neq t} f(x)_c \tag{6}$$

where t is the desired adversarial label. In essence, we wish for the probability of the model outputting the desired adversarial label being greater than the highest probability label of any other class. In our case, the target class is randomly chosen.

After population members are evaluated according to their fitness, they are selected given the normalized scores for all of them (sampling from a probability distribution). Those selected will be part of the next generation. Those selected have their features combined to produce additional members of the population. In addition, members may have random noise applied to them, mutating them to explore the search space for adversarial images.

5 Implementation and Evaluation

Keeping in mind that some of the attack methods took a very long time to run for a single image, we decided to evaluate the attacks on a small subset of ImageNet (40 images) that we created. Especially for ZOO Attack, we had to keep the number of iterations limited to 1500 because it takes a long time to run.

We altered some of the attack implementations to use this new dataset. In addition, we added some operations that would allow us to calculate and output similar statistics across different attacks i.e. L2 norm. To be specific, we implemented additional functionality for reporting the attack success rate, average queries run, and the average L2 distance between the original image and the adversarial image, as well as the mean attack time. The Bandit Attack algorithm did not support outside datasets (i.e. it only worked with ImageNet download and file structures). We created our own dataset class for interfacing with the algorithm. We played around with different algorithmic hyperparameters to make the code run more efficiently. For GenAttack and Bandit Attack, we used their image resize ability to limit the search space for adversarial inputs. This is a major shortcoming of our approach - by limiting the search space, some algorithms may have a significant advantage in finding successful images. This imbalance weakens our comparison and we will keep this in mind for the future as we gain access to more powerful computational resources.

We performed our evaluation on a local machine equipped with an AMD Threadripper 1950X CPU with 2 GeForce GTX 1080 Ti. In addition, we also ran the attacks on Google Cloud Platform equipped with a single Nvidia Tesla P100.

Using our dataset, we ran and evaluated each of the attacks on 4 primary metrics: Attack Success Rate, Average number of queries per successful attack, Average L2 distance between the original and adversarial images, as well as the average time taken to generate a specific adversarial example. The results of these attacks are shown in 5

Attack success rate was fairly high for all three methods. However, we should mention that these are the number of successful adversarial attacks divided by the number of successfully classified images. Not all images were classified correctly - ZOO misclassified 4 images, GenAttack misclassified 2 images, and Bandits misclassified a whopping 33 images. Even though all are based off of inception-V3, we were unable to find why there was such high variation in the number of misclassifications.

We did not limit the models by a number of queries - we decided to keep the default settings. ZOO has an upper limit of 1500 queries, GenAttack has an upper limit of 100,000 queries, and Bandits has an upper limit of 10,000 queries. We did bring the limit of all queries to 1500, but found that GenAttack produced 0 successful adversarial attacks, and Bandits produced 1.

The average L2 distance varied highly over algorithms as well. We also attempted to enforce uniform L2 norms, but found two issues. The ZOO attack does not enforce L2 contraints (i.e. it is not a pass/fail condition). GenAttack, although is still able to calculate L2 norms, it uses the L-infinity norm as a multiplying factor for the amount of random noise to generate in each generation. We were not sure how to translate this into L2 norms. Finally, in Bandit Attack, on the other hand, although it had the option to set L-infinity norm contraints so we can compare with GenAttack, it performed very poorly (only 1 successful adversarial attack)

We understand that the average time taken to generate examples is not an appropriate metric to judge the efficiency of the algorithm - after all, our experiments spanned 2 hardware systems and 2 learning frameworks. ZOO Attack, based on Tensorflow, ran on GCP. GenAttack, also based on Tensorflow, ran on the local GPU machine. Bandit Attack, on the other hand, is based on PyTorch and ran on GCP. Although we did attempt to bring these together into a common framework, we were limited by the time it took to compute attacks for some of these algorithms. We did run GenAttack on the GCP cloud instance with the same limited number of queries as ZOO Attack, and found that it performed very poorly (0% Attack success rate).

Figure 3: Original: Gorilla

Figure 4: Adversarial: Chimpanzee

Figure 5: Original: Gorilla

Figure 6: Adversarial: Upright Piano

Figure 7: Original: Gorilla

Figure 8: Adversarial: standard schnauzer

Figure 9: ZOO Attack, Gen Attack, Bandit Attack

Attack	Attack Success Rate	Average Queries	Average L2 Distance	Mean Attack Time (seconds)
ZOO	62.90%	<1500	0.55973	510.15
GenAttack	94.44%	26,097	21.08	2063.23
Bandits	71.43%	2,384	4.9972	26.06

Table 1: Computed Statistics for different Black Box Adversarial Attacks

6 Discussion

We were able to produce successful black-box attacks without training any substitute model as an attack surrogate. All three of the above discussed attacks show comparable performance to the state-of-the-art white box Carlini and Wagner attacks [1]. While all the attacks are based on a similar framework, the objective functions for each method are subtly different.

Some approaches we considered but did not implement (due to the prohibitive run-times of the algorithms discussed above) were:

- Extending the approach to other norms more reflective of human perception. This idea was inspired by the investigations by Sinha et al., [7] where the L_1 and L_2 were studied in the context of human perception. Existing norms, e.g., the L_2 norm of the difference image (between the original and an image that just has intensities altered from the original) is large,

whereas the altered image is still perceived as belonging to the same class by humans. If we had a metric that took this into account, our algorithms would possibly be less constrained in trying to fool the network.

- Run evaluations on the full Imagenet dataset. The images we chose to evaluate the methods may have been a poor representative of the algorithms' performance. For instance, ZOO [2] and GenAttack [3] work very well on the subset we ran evaluation on while BanditAttacks [6] performed rather poorly is comparison. These findings do not line up with the benchmarks provided by the authors.
- Run evaluations on other datasets (e.g. CIFAR-10) for all of the methods discussed above. The accuracies achieved by the state-of-the-art models on CIFAR-10 are higher than that achieved on Imagenet. This might make it harder to fool such architectures. Therefore, the adversariality rates of the attack methods need to evaluated on other datasets as well.
- Run attacks for different architectures. The evaluations were done on the Inception V3 model for the Imagenet dataset. While the transferability of adversarial examples has been shown by Goodfellow et al. [4], the number of queries required to come up with adversarial examples may be different for different architectures. This offers the opportunity to transfer attacks from simpler models to more complex ones.

References

[1] Nicholas Carlini and David Wagner. "Towards evaluating the robustness of neural networks". In IEEE Symposium on Security and Privacy, 2017a.

[2] 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, AISec '17

[3] GenAttack: Practical Black-box Attacks with Gradient-Free Optimization M. Alzantot, Y. Sharma, S. Chakraborty, and M. Srivastava. 2018. "Genattack: Practical black-box attacks with gradient-free optimization".

[4] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples."

[5] Ilyas, A.; Engstrom, L.; Athalye, A.; and Lin, J. 2018. "Black-box adversarial attacks with limited queries and information".

[6] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. "Prior convictions: Black-box adversarial attacks with bandits and priors".

[7] Pawan Sinha, and Richard Russell. "A perceptually based comparison of image similarity metrics." Perception 40, no. 11 (2011): 1269-1281.