An Empirical Look at Gradient-based Black-box Adversarial Attacks on Deep Neural Networks Using One-point Residual Estimates

Joost Jansen
Supervisors: Stefanie Roos, Jiyue Huang and Chi Hong
EEMCS, Delft University of Technology, The Netherlands
June 19, 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

An Empirical Look at Gradient-based Black-box Adversarial Attacks on Deep Neural Networks Using One-point Residual Estimates

Joost Jansen[1]
Supervisors: Stefanie Roos[1] , **Jiyue Huang**[1] , **Chi Hong**[1]
[1]EEMCS, Delft University of Technology, The Netherlands

Abstract

In recent years, there has been a great deal of studies about the optimisation of generating adversarial examples for Deep Neural Networks (DNNs) in a black-box environment. The use of gradient-based techniques to get the adversarial images in a minimal amount of input-output correspondence with the attacked model has been extensively studied. However, existing studies have not been discussing the effect of different gradient estimation techniques coherently. In this paper, a new one-point residual estimate is compared to the known two-point estimates. The findings in this paper show that the one-point residual estimate is not a viable option to decrease the number of queries to the attacked model. The accuracy of the attacks with the use of an one-point residual estimate maintains the same for weaker models. For stronger models, there is a slight decrease in accuracy at identical distortion levels. All estimates are tested on different PGD attacks on the MNIST and F-MNIST datasets using a 3-layer convolutional network.

1 Introduction

As Deep Neural Networks (DNNs) are introduced more and more in our everyday life, their dangers become of greater interest. Therefore, it is truly important to foresee what kind of exposure a DNN has and how to train a DNN to overcome these pitfalls. In the last couple of years, there has been a lot of research about a specific exposure called adversarial attacks, specifically in image classifiers. These adversarial attacks on a DNN provide an image that seems the same for the human eye but is misclassified by the DNN as another class (See Figure 1 as an example). This misclassification can be either a specific class (targeted attack) or any other class (untargeted attack). The practical dangers of these adversarial attacks are already widely discovered. For example, it has been shown that a stop sign can be misclassified to a yield sign by a DNN [18] or an image classifier phone application can be fooled with printed images to be misclassified to other classes [16].

There are numerous methodologies to find such an adversarial image given that the model is known to the attacker [10; 16; 3]. These methods are called white-box methods and give excellent results. However, in practicality, the model is most of the time not known to the attacker. Only the input image and the output class probabilities are known to an attacker. Therefore, the question arises of how an attacker can make an adversarial image without knowing the model. These so-called black-box methods can be divided into two approaches. With the first method, it's possible to train a substitute model using the input and output of the attacked model and perform a white-box attack on the substitute model. The second method, as displayed in Figure 2, is a direct attack on the model by estimating the gradient and performing gradient descent to decrease the probability of the correct class. In the first case, the problem emerges that the accuracy of the attack heavily relies on the transferability between the substitute model and the attacked model [23]. Therefore, this paper mainly focuses on the second gradient-based attack method. The ZOO [4], AutoZOOM [24] and other attack algorithms [12; 2; 13; 21] have gradient-descent methods for adversarial attacks without knowing the true gradient. However, these methods require a lot of query-result pairs to the attacked model that take a substantial amount of time and could raise awareness of the intruder. Hence, this paper will focus on improving such an attack by reducing the number of queries while maintaining the same accuracy. Granted that this is true, it shows that we should caution when to use DNNs and further focus on how to resolve these attacks.

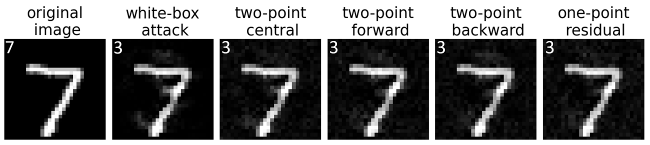


Figure 1: A L_2-PGD attack with $\epsilon = 2.5$ on a Deep Neural Network (DNN) using different estimates. In the top left corner of every image is the estimated class by the DNN. The first image is the correctly classified original image and aside from this picture is an adversarial image made with the white box version of the L_2-PGD attack. The images on the right of the white-box attack are the adversarial images made with the use of different estimates in a black-box environment. In Appendix A more examples are shown.

Recently a new paper came out about determining the gradient with the use of an one-point residual estimate instead of

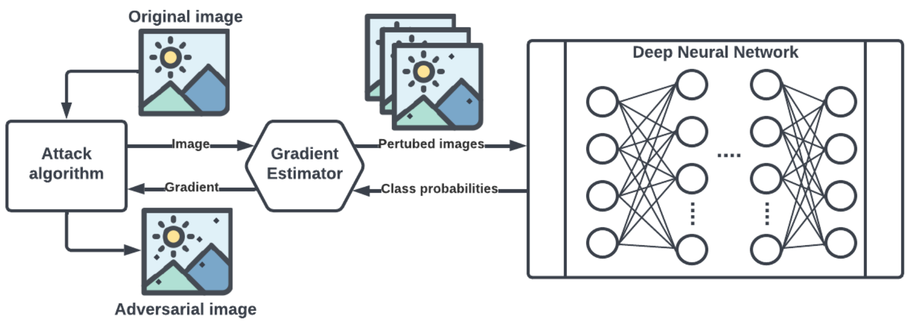


Figure 2: A global view of a gradient-based black-box adversarial attack on a Deep Neural Network. The attack algorithm receives an image that is sent to the gradient estimator. The gradient estimator sends multiple identical images with a small perturbation to the attacked model. It then receives the class probabilities back and calculates the gradient of every pixel with these probabilities. The estimated gradient is sent back to the attack algorithm and changes the image according to the gradient. This process is repeated multiple times until a misclassified adversarial image arises.

the known two-point estimates [26]. This paper showed that with other comparable problems this new estimate remained relatively the same accuracy. The one-point residual estimate uses fewer points to determine the gradient, which might result in a reduction in the overall number of queries. Therefore, we will answer the following research question in this empirical study:

Do one-point residual estimates improve untargeted gradient-based adversarial attacks in terms of reducing the number of queries while maintaining accuracy?

In pursuance of the aforementioned research question, our contributions can be summarised as follows:

- A brief survey about adversarial attacks and different kinds of gradient estimators.
- A novel algorithm for gradient-based attacks that uses an one-point residual estimate instead of a two-point estimate.
- A new inside in the performance of the one-point residual estimate compared to the different two-point estimates.

2 Related Work

2.1 White-box attacks

Gradient-based adversarial attacks fall mostly within the white-box setting. In this white-box setting, the gradient is directly calculated through the back-propagation of the model. The first adversarial attack, called Fast Gradient Sign Method (FGSM) [10], used the gradient of the image to get a quick and dirty adversarial image. Not long after that, methods like the Basic Iterative Method (BIM) [16] and Projected Gradient Descent (PGD) [17] performed this method in an iterative way to improve accuracy. Others have used a greedy algorithm that modifies pixels one at a time using the gradient as a saliency map [20]. The authors of [3] incorporated the distance between the original and the adversarial image in the gradient function to minimise the perturbation at the same time. In this paper, we'll be using the black-box version of the PGD attack.

2.2 Black-box attacks

Since a couple of years, gradient-based adversarial attacks are also possible in a black-box environment. Instead of calculating the gradient directly, it's possible to estimate the gradient using zeroth order (ZO) methods as used in [4] . In section 3 further explanation about different ZO methods will be given. These ZO methods use the difference between the output of the same image but with a small perturbation to calculate the gradient. This makes it possible to use gradient-based attacks in a black-box environment. However, as listed before, the estimation of the gradient is quite inaccurate and needs a lot of queries to the attacked model. Therefore others have tried to enhance this ZO method through the use of offline or online optimisations. For example, the authors of [24] used an AutoEncoder to preprocess the adversarial images before sending them to the attacked model. In terms of online optimisation techniques researchers have used the momentum of gradients to decrease the number of queries [7]. The Adam optimiser [15] is a widely known gradient-based optimiser that has also been used by multiple articles [4; 24]. Another main contribution to improving query efficiency is the use of random vector-based estimations instead of a coordinate-wise estimation [24]. A recent study also used Natural Gradient descent, which uses an estimation of the second-order derivative to improve accuracy [27].

2.3 Non-gradient-based attacks

Aside from gradient-based attacks, other attacks are also possible in the black-box scenario. The use of a genetic algorithm as used in [1] seemed to greatly reduce the number of queries with relatively good accuracy. Random perturbation in different directions in the images is also a non-gradient-based method that is used by the SimBA attack [11]. Another attack in the black-box area is the use of a substitute model instead of attacking the model directly [18; 19; 23].
None of the related work covers the effect of the use of different two-point estimations or the use of an one-point residual estimate of the gradient. The effect of these methods is tested on the algorithms of the authors of [17].

Estimate name	Formula	Mini-batch formula	Number of queries to the attacked model
Two-point central	$\frac{f(x_t+\beta u_t)-f(x_t-\beta u_t)}{2\beta}u_t$	$\frac{1}{b}\sum_{j=1}^{b}\frac{f(x_t+\beta u_{tj})-f(x_t-\beta u_{tj})}{2\beta}u_{tj}$	$2Tb$
Two-point forward	$\frac{f(x_t+\beta u_t)-f(x_t)}{\beta}u_t$	$\frac{1}{b}\sum_{j=1}^{b}\frac{f(x_t+\beta u_{tj})-f(x_t)}{\beta}u_{tj}$	$Tb+T$
Two-point backward	$\frac{f(x_t)-f(x_t-\beta u_t)}{\beta}u_t$	$\frac{1}{b}\sum_{j=1}^{b}\frac{f(x_t)-f(x_t-\beta u_{tj})}{\beta}u_{tj}$	$Tb+T$
One-point residual	$\frac{f(x_t+\beta u_t)-f(x_{t-1}+\beta u_{t-1})}{\beta}u_t$	$\frac{1}{b}\sum_{j=1}^{b}\frac{f(x_t+\beta u_{tj})-f(x_{t-1}+\beta u_{(t-1)j})}{\beta}u_{tj}$	Tb

Table 1: Different gradient estimates with their corresponding formula, mini-batch formula and their total number of queries to the model that is being attacked. Here T is total amount of iterations.

3 Methodology

3.1 Notation for Deep Neural Networks

Deep Neural Networks are able to classify images into certain classes through a network of multiple layers of artificial neurons. A DNN is trained with numerous image-class pairs and may be thought of as a function $Z(x) = y$ with input $x \in [0,1]^d$ and $y \in \mathbb{R}^C$. The model Z is also dependent on some model parameters θ. In our study, the model is fixed, thus we don't indicate the dependency on θ for convenience. d is in this case the number of pixels in the input image x and the value between 0 and 1 gives the intensity of the pixel. For example, a colour image contains 3 dimensions of RGB colours. Therefore, there are $d = width * height * 3$ number of pixels. The last layer, which is called the *logits*, outputs y and gives a number to the likeliness of the input belonging to the C classes. This last layer of a DNN is then normalised and is the only information given in the black-box environment. It gives all the probabilities of the known classes which are calculated through the cross-entropy function of the *logits* and can be stated as:

$$Prob(x,c) = \frac{exp([Z(x)]_c)}{\sum_{i=1}^{C} exp([Z(x)]_i)} \in [0,1] \qquad (1)$$

Where x is the input image, c is a given class in all class possibilities C and $[Z(x)]_c$ gives the output of the *logits* for class c. This function will give the probability of x being class c between 0 and 1.

3.2 Deep Neural Network attacks

The goal of an adversarial attack is to make the attacked model misclassify the original image by making an imperceptible change for the human eye in the pixels of the image. Thereby the attack minimises the change between the original picture and the adversarial image. As stated by [22], this comes down to the following constrained optimisation problem:

$$\begin{aligned} f(x) &= \min(Prob(x,c_0)) \\ & w.r.t.||x_{adv} - x_0||_{L_p} \end{aligned} \qquad (2)$$

c_0 is the original class of the image that should be misclassified. Thus, by minimising the chance of c_0, other classes will be the most probable outcome and the adversarial image x_{adv} will be wrongly classified. x_0 is the original image and is compared to x_{adv} to measure the difference between the two images. L_p is the distance metric and can be calculated in multiple ways:

1. L_0 distance measures the number of pixels that have been altered in the image and can be written as: $\sum_i^d x_{adv}^i \neq x_0^i$ for all pixels $i \in d$. This value will be minimal if only a few pixels are changed. However, these pixels can be changed by a lot and still receive a low L_0 distance
2. L_1 is the Manhatten distance and it measures the absolute distance between two points. This value stays small if a lot of pixels are changed a little bit and can be written as: $\sum_i^d |x_{adv}^i - x_0^i|$ for all pixels $i \in d$.
3. L_2 is the Euclidean distance measure and is also known as the root-mean-square. It measures the squared distance between the adversarial image and the original image. This value also stays small if a lot of pixels are changed a little.
4. L_∞ measures the maximum distance of all pixels between the original and adversarial image. This means all pixels can be changed a slight bit, but there can be no outliers.

All types of L_p measurements have an effect on the amount of distortion in the image. Many of these measurements try to give a number to the human perceptual similarity between two images. These metrics all have a different effect on formula 2 and therefore give a different adversarial image when applied. We will use the L_2 and L_∞ distance measures since these are the most widely used distance measures for adversarial attacks and also have been used in the articles about the adversarial attack method we use.

3.3 White-box gradient-based attacks

Most adversarial attacks are based on formula 2. A specific domain of adversarial attacks is the gradient-based attack. They calculate with backward propagation the gradient of the loss function and use it with gradient-descent to minimise formula 1 as formula 2. FGSM was the first method to use the sign of the gradient to get an adversarial image [10]. After that came the iterative versions of FGSM, called

BIM [16] and PGD [17]. The last method is the most efficient of the three and suffices to perform in a black-box environment. PGD uses the gradient of the image to perform gradient-descent iteratively and can be stated as:

$$x_{t+1} = proj(x_t + \alpha * sign(\nabla f(x_t))) \\ w.r.t. ||x_{t+1} - x_0||_{L_p} \leq \epsilon \quad (3)$$

$\nabla f(x_t)$ is the gradient vector of the current image x_t. The sign function is then taken from this gradient and is multiplied by a learning rate α. This vector is then added to the image to become x_{t+1}. All pixels in the image change by a certain amount α in each iteration. However, the pixels must stay within their [0,1] bound and therefore it's projected to keep within these bounds. The L_p distance between the original and the distorted image x_{t+1} can never be bigger than ϵ. This process is repeated t times. After t iterations it is not guaranteed that the image will be misclassified by the attacked model.

Although there are other efficient gradient-based techniques in the field, we will mainly focus on gradient estimation with the PGD approach and let other techniques be covered in later research.

3.4 Gradient estimations in a black-box scenario

To calculate the gradient of the PGD attack in a black-box environment an estimation needs to be made. These estimations of the gradient are called zeroth-order (ZO) methods. The first efficient ZO adversarial generator was the ZOO attack and it uses the central finite difference estimation of the gradient [4]:

$$\tilde{\nabla} f(x_t) = \frac{f(x_t + he^i) - f(x - he^i)}{2h} \quad (4)$$

Here e^i is a null-vector of length d with on the ith place an one. h is a small constant and i is determined stochastically. In this way, the gradient is determined by a single pixel. However, a single pixel change has little influence and therefore this procedure must be repeated an inefficient amount of times. The NES-PGD attack introduced another method [12]. This attack uses the natural evolutionary strategy as an estimation with the combination of a projected gradient-descent. It uses a central difference of a random vector base.

$$\tilde{\nabla} f(x_t) = \frac{f(x_t + \beta u_t) - f(x_t - \beta u_t)}{2\beta} u_t \quad (5)$$

Here u is a unit-length vector that is uniformly drawn at random from a unit Euclidean sphere and $\beta > 0$ is a smoothing parameter. In this way, all pixels in the image are slightly changed and the gradient of all pixels is determined at the same time. The authors of AutoZOOM [24] suggested a forward difference estimate instead of the central difference to increase query efficiency and can be written as:

$$\tilde{\nabla} f(x) = \frac{f(x_t + \beta u_t) - f(x_t)}{\beta} u_t \quad (6)$$

To reduce variance, both AutoZOOM and NES-PGD use a mini-batch version which takes b samples and takes the average of all current gradients. It can be written as:

$$\tilde{g}(x_t) = \frac{1}{b} \sum_{j=1}^{b} \tilde{\nabla} f(x_{tj}) \quad (7)$$

While the mini-batch version of formula 5 needs $2 * b$ queries to the attacked model to get a single gradient approximation, the mini-batch version of formula 6 needs to query b + 1 times. b times for all $f(x_t + \beta u_t j)$ and one time for the $f(x_t)$ query. Since the variance is too big in a non-mini-batch way, we will focus from hereon solely on the mini-batch versions of all formulas written as formula 7.

While the finite difference formula 4 is known to be too inefficient [4], the difference in terms of accuracy and number of queries between the forward, central and backward difference is yet to be discovered for adversarial gradient-based attacks.

3.5 Proposed one-point residual estimate

In this part, we propose the use of an one-point residual estimate from the authors of [26] instead of the various two-point estimates. The one-point residual estimate uses the difference between the perturbed estimation of the last iteration $t-1$ and the current iteration t as listed in formula 8.

$$\tilde{\nabla} f(x_t) = \frac{f(x_t + \beta u_t) - f(x_{t-1} + \beta u_{t-1})}{\beta} u_t \quad (8)$$

This version only uses one query to the attacked model per gradient estimation and b queries for the mini-batch version. Compared to the two-point central estimate, it would reduce the queries by half. For the two-point forward and backward difference, it reduces the number of queries by $\frac{1}{b}$. However, it should be noted that the image between t and $t-1$ is altered and the random vector distribution is also different. This can result in an inaccurate calculation of the gradient and thus lead to more variance. Therefore, we will compare the mini-batch version of formula 8 with the other mini-batch estimates stated in Table 1 in an empirical way and see if the one-point residual estimate can make a positive impact in terms of queries or accuracy.

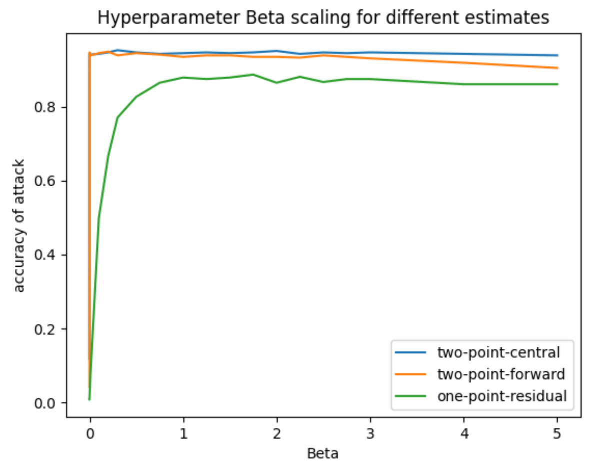


Figure 3: Different use of β at L_∞-PGD attack with ϵ=0.3 and a learning rate α of 0.025. The number of steps and the number of samples taken per gradient estimation are both 50. The two-point backward estimation is dismissed from this graph as it's highly similar to the two-point forward estimate.

4 Experimental Setup and Results

4.1 Hyperparameters

The different estimations as listed in Table 1 are tested in a mini-batch way. The number of samples drawn for each gradient estimation is 50 and the number of steps the L_∞- and L_2-PGD attack are 50 and 100, respectively. The learning rate is different per distance measure as listed in Appendix B in Table 3. These hyperparameters are mainly taken from the authors of the PGD [17] and NES-PGD [12] attacks, who have argued the reason for these parameters. During our research, we've noticed that the one-point residual estimate is heavenly dependent on the β in formula 8. Therefore we've taken a look at the optimal β in terms of accuracy for one-point residual estimates and two-point forward and central estimates in Figure 3. We've tested the optimal β for both L_∞- and L_2-PGD attacks and received similar results as is visible in Appendix C. Since β does not affect the performance of the two-point estimate and the one-point accuracy does not increase after 1.5, we have concluded the optimal to be $\beta = 1.5$.

4.2 Setup

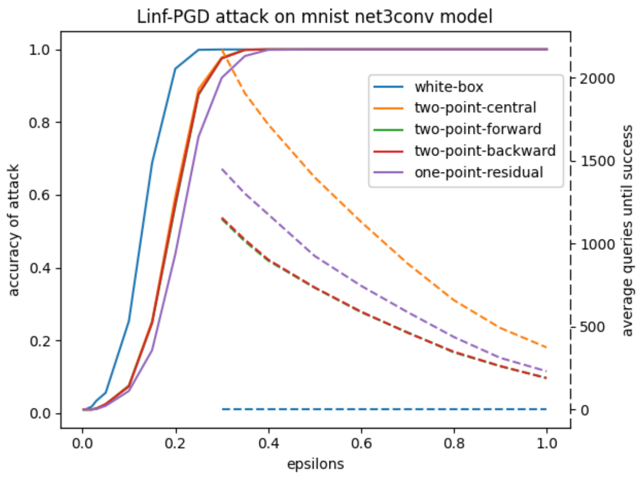
The L_∞- and L_2-PGD attack are tested both on 500 randomly selected MNIST [6] and Fashion-MNIST (F-MNIST) [25] images and the attacked model is a 3-layer convolutional network with 99.55% and 90.50% accuracy respectively for each dataset[1]. The average is then taken of three runs to reduce the randomness. The MNIST and F-MNIST dataset both have the same image format. However, the MNIST images are less complex than the F-MNIST images, and thus the used model has a higher prediction accuracy. In that way, we can see if the accuracy of the attacked model has any effect on the accuracy of the attack. The one-point residual estimate is compared to the white-box version and the two-point forward, backward and central estimates versions of the different PGD-attacks.

4.3 Evaluation metrics

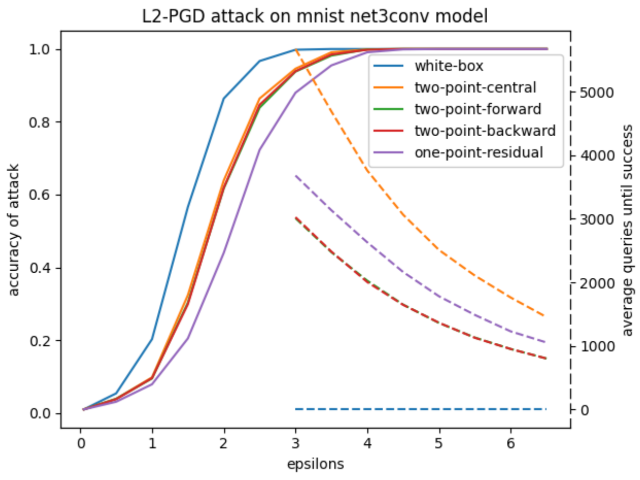
Accuracy: The different estimates are tested on three metrics. The first metric is the accuracy of the attack. This is the number of correctly misclassified adversarial images divided by the total number of images.

Average queries until success: Another metric used, is the average number of queries until a successful adversarial image is created. At every PGD step, it checks if the attack has created a misclassified image and adds the number of queries to the total if it's not misclassified. The total number of queries is then divided by the number of images at the end. The average number of queries until success is only measured from a higher attack accuracy than 90%. As the total amount of tests is 500 images, an accuracy lower than 90% will result in few successful attacks and thus an inaccurate average of queries until success. To get a broad view of the effect of the estimates, the attacks are tested on multiple maximum distortion levels called epsilons. The specific points and further details about the parameters are listed in Appendix B.

[1]https://github.com/Joost-Jansen/PGD_attack_estimators



(a) L_∞-PGD attack on MNIST



(b) L_2-PGD attack on MNIST

Figure 4: Results of L_∞ and L_2 PGD attacks on a 3-layer convolutional model with MNIST dataset. On the left y-axis is the accuracy of the attack and on the right dashed axis is the average number of queries until a successful adversarial image is made. The x-axis displays the amount of distortion the adversarial image contains.

Null hypothesis: One of the key question is if the one-point residual estimate will maintain the same accuracy. Therefore we've proposed the following hypothesis:

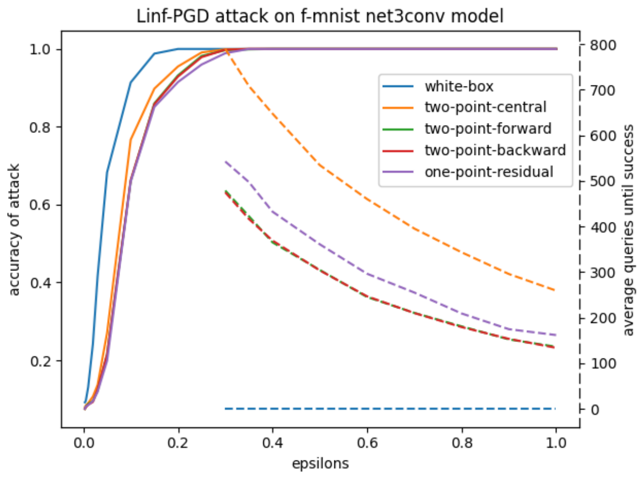
$$\begin{aligned} H_0 &: p_{opr} == p_{est} \\ H_1 &: p_{opr} < p_{est} \end{aligned} \quad (9)$$

Here p is the accuracy of the estimate, opr is the one-point residual estimate and est is a different estimate. For this one-tailed test, we'll be testing the hypothesis with a 95% confidence level. Which results in a critical region of $z_{95\%} = 1.65$. Therefore the H_0 hypothesis will be rejected if the test-statistic z is $|z| > z_{95\%}$. This means that the one-point residual estimate does not have statistically the same accuracy. z

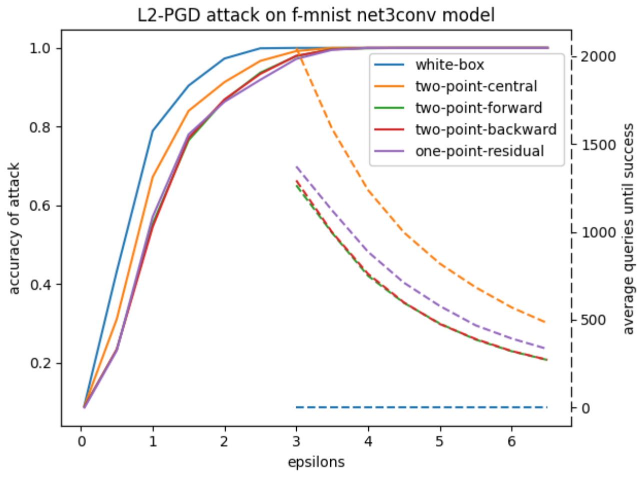
can be calculated in the following way:

$$z = \frac{p_{opr} - p_{est}}{\sqrt{\frac{p_{opr}(1-p_{opr})+p_{est}(1-p_{est})}{n}}} \quad (10)$$

Here n is the number of samples tested. We'll be testing the H_0 hypothesis for $\epsilon_\infty = 0.25$ and $\epsilon_2 = 2.5$ as at these points the average accuracy gap between the one-point residual estimate and other estimates is at its maximum.



(a) L_∞-PGD attack on F-MNIST



(b) L_2-PGD attack on F-MNIST

Figure 5: Results of L_∞ and L_2 PGD attacks on a 3-layer convolutional model with the F-MNIST dataset. On the left y-axis is the accuracy of the attack and on the right dashed axis is the average number of queries until a successful adversarial image is made. The x-axis displays the amount of distortion the adversarial image contains.

4.4 Results

Our findings on the accuracy and the average number of queries until success of PGD-attacks with different estimates are listed in Figure 4 and 5. A more accurate inside in the results of the estimates at $\epsilon_\infty = 0.25$ and $\epsilon_2 = 2.5$ are visible in Table 2.

Maintaining accuracy: The white-box estimate is the optimal accuracy for each PGD-attack. It can be pointed out that the one-point residual estimate has a slightly lower accuracy than the other estimates for the MNIST dataset, but performs relatively the same for the F-MNIST dataset. For the specific points $\epsilon_\infty = 0.25$ and $\epsilon_2 = 2.5$ the H_0 hypothesis is rejected for the MNIST dataset. However, it is not rejected for the F-MNIST dataset compared to the two-point forward and backward estimates. A reason for maintaining the same attack accuracy could be that the attacked model has a lower prediction accuracy for the F-MNIST dataset in general. In turn, this would mean the attack algorithm needs a less accurate gradient estimator as it can fool the attacked model more easily.

Fewer queries: The average amount of queries until a successful attack is noticeably lower than the two-point central estimate. However, the two-point backward and forward estimates have a lower query average than the one-point residual for all tests. This outcome can be deduced by the lower precision of estimating the gradient with the one-point residual estimate. Although the one-point residual estimate uses fewer queries per iteration, the poor precision results in the need for more iterations to achieve an adversarial image. Therefore, the average queries of the one-point residual estimate lie considerably higher than for the two-point forward and backward estimate.

5 Discussion

The results shown in section 4 are limited in some aspects. In terms of model, datasets and kind of attack, the estimates could behave differently. Due to the limited computational power and time, all attacks are only tested on one model. The estimates could have a different effect on other models. The same accounts for different datasets.

5.1 Simple datasets:

Currently, we are only testing our attacks on the MNIST and F-MNIST datasets, which are relatively simple low-dimensional datasets. Recent research has discovered that adversarial vulnerability is more likely to be caused by high-dimensional data [8; 9; 14]. Therefore, testing different estimates on more complex datasets could result in closing the accuracy gap between the two-point and one-point residual estimates. The two-point central estimate is tested against a more high-dimensional dataset called Imagenet in [12] and produces a fairly good outcome. It could therefore be discussed if one-point residual estimates produce the same outcome for the Imagenet dataset. The pictures from Imagenet have far more pixels and are RGB coloured instead of a single grayscale. This could influence the need for accurate gradients. As more pixels could be changed, the model will rely less on specific pixels and therefore could decrease the need for accurate estimations.

Attack	Estimate	MNIST			F-MNIST		
		Average queries until success	Accuracy	H_0 rejection	Average queries until success	Accuracy	H_0 rejection
L_∞-PGD	white-box	-	99.9%	True	-	100%	True
L_∞-PGD	two-point central	2170	89.1%	True	789	99.1%	True
L_∞-PGD	two-point forward	1150	87.5%	True	479	98.2%	False
L_∞-PGD	two-point backward	1150	87.8%	True	475	99.9%	False
L_∞-PGD	**one-point residual**	1500	76.0%	-	543	96.2%	-
L_2-PGD	white-box	-	96.7%	True	-	99.9%	True
L_2-PGD	two-point central	5670	86.4%	True	2630	96.7%	True
L_2-PGD	two-point forward	3030	83.9%	True	1530	93.7%	False
L_2-PGD	two-point backward	3010	84.6%	True	1580	93.4%	False
L_2-PGD	**one-point residual**	3680	72.3%	-	1520	91.9%	-

Table 2: Results for different PGD attacks at $\epsilon_\infty = 0.25$ and $\epsilon_2 = 2.5$. The average queries until success display the mean number of queries until a valid adversarial image is made. The H_0 hypothesis is rejected (True) when there is a significant difference between the accuracy of the one-point residual estimate and the estimate of that current row. The hyperparameters of all attacks are listed in section 4.1

5.2 Single attack algorithm:

The effect of an one-point residual estimate could further be worse or better with the use of other attacks such as ZOO [4] or AutoZOOM [24]. These attacks use other offline and online optimisation techniques that could influence the need for accurate gradient estimations. They also use a different function to calculate the gradient.

5.3 Distribution and hyperparameters:

For all our estimators we've chosen to use the uniform distribution to add noise to the picture and thereby follow the article of the NES-PGD attack [12]. Other distributions, such as the Gaussian distribution, could have an influence on the performance. On top of that, finding optimal values for the hyperparameters other than the most influential β could improve the accuracy. Then again, due to insufficient computational power, this research was limited to finding one optimal hyperparameter.

Although there are some limitations in different scenarios, the current results are in line with some of the work of [26]. Here the one-point residual estimate performs slightly worse than the two-point version.

6 Conclusions and Future Work

6.1 Conclusion

With the results in mind of Figure 4, Figure 5 and Table 2, we can answer the question of one-point residual estimates improving query efficiency while maintaining accuracy. With the use of the MNIST dataset, the accuracy of the two-point estimates outperforms the one-point residual estimate in all cases significantly. However, the one-point residual estimate compared to the two-point forward and backward estimate maintains statistically the same accuracy with the use of the F-MNIST dataset. This can be deduced to the attacked model having a lower accuracy on the F-MNIST dataset. In terms of the number of queries used, the one-point residual estimate outperforms the two-point central estimate. Despite the fact that it uses fewer queries per iteration, it performs worse compared to the two-point forward and backward estimate. Therefore, we can conclude that one-point residual estimates perform worse in accuracy and number of queries when used on high-accuracy models. For a model with a lower accuracy rate, the one-point residual estimate can perform attacks with relatively the same accuracy. However, it does have a worse average number of queries until success. Since DNNs are constantly being improved and high accuracy models are the norm today, the use of one-point residual estimates will not be of any use. Be that as it may, it shows that the two-point forward and backward estimates remain favourite as gradient estimators for black-box gradient-based adversarial attacks.

6.2 Future work

Although these empirical results implicate the use of one-point residual estimates will not be significant, further research should be done on the theoretical side of the implementation of adversarial attacks with one-point residual estimates. A theoretical-proven optimal for β could improve the efficiency to have the same accuracy as two-points estimates. Aside from the empirical view, this research was limited to PGD attacks and further research could also be done on other attacks with more complex datasets and models. Furthermore, a grid search could be used to find all optimal hyperparameters. At last, more research could be done on the use of different distributions with the one-point residual estimator.

7 Ethics of adversarial attacks

The effect of studying and improving adversarial attacks is a controversial topic. Emphasising this subject can lead to more attention and thereby lead to the use of such algorithms for misconduct. However, it can also lead to better protection as users of such DNN can improve their algorithms with adversarial training. A concept called Ethical Adversarial Attacks (EAA) can help mitigate this problem. The authors of [5] proposed the EAA concept, which is the proposition to debate and recognise ethical adversarial machine learning. In turn, this would be used to halt, deceive, or effectively attack adversarial algorithms built for harmful goals and damaging society. Such tools and procedures should be developed alongside suitable legal and ethical frameworks. With this kind of framework, attacks could be prevented and rightfully persecuted when occurred. This paper would like to make a contribution to these ethical aspects by adding a new inside in gradient-based black-box adversarial attack and thereby expanding the knowledge about adversarial attacks.

Acknowledgement

We wish to thank our supervisors Stefanie Roos, Jiyue Huang and Chi Hong for providing tips and feedback throughout the process of making this paper.

References

[1] Moustafa Alzantot, Yash Sharma, Supriyo Chakraborty, Huan Zhang, Cho-Jui Hsieh, and Mani B Srivastava. Genattack: Practical black-box attacks with gradient-free optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1111–1119, 2019.

[2] Arjun Bhagoji, Warren He, Bo Li, and Dawn Song. *Practical Black-Box Attacks on Deep Neural Networks Using Efficient Query Mechanisms: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part XII*, pages 158–174. 09 2018.

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[4] Pin Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho Jui Hsieh. Zoo: Zeroth order optimization based black-box atacks to deep neural networks without training substitute models. pages 15–26. Association for Computing Machinery, Inc, 11 2017.

[5] Michał Choraś and Michał Woźniak. The double-edged sword of ai: Ethical adversarial attacks to counter artificial intelligence for crime. *AI and Ethics*, 10 2021.

[6] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[8] Simant Dube. High dimensional spaces, deep learning and adversarial examples. *arXiv preprint arXiv:1801.00634*, 2018.

[9] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

[10] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[11] Chuan Guo, Jacob Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Weinberger. Simple black-box adversarial attacks. In *International Conference on Machine Learning*, pages 2484–2493. PMLR, 2019.

[12] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

[13] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In *International Conference on Learning Representations*, 2019.

[14] Marc Khoury and Dylan Hadfield-Menell. On the geometry of adversarial examples. *ArXiv*, abs/1811.00525, 2018.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[16] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *ICLR Workshop*, 2017.

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[18] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. pages 506–519. Association for Computing Machinery, Inc, 4 2017.

[19] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[20] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.

[21] Ankur Sarker, Haiying Shen, and Tanmoy Sen. A context-aware black-box adversarial attack for deep driving maneuver classification models. volume 2021-July. IEEE Computer Society, 7 2021.

[22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, D. Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.

[23] Jean-Baptiste Truong, Pratyush Maini, Robert J. Walls, and Nicolas Papernot. Data-free model extraction. *CoRR*, abs/2011.14779, 2020.

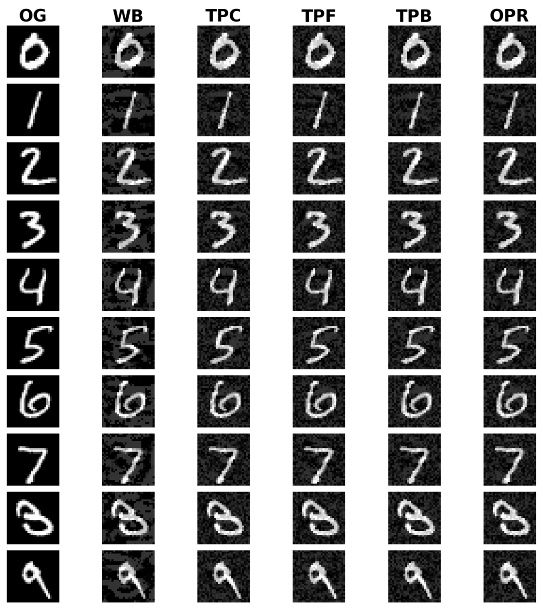
[24] Chun-Chen Tu, Paishun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:742–749, 07 2019.

[25] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. cite arxiv:1708.07747Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/.

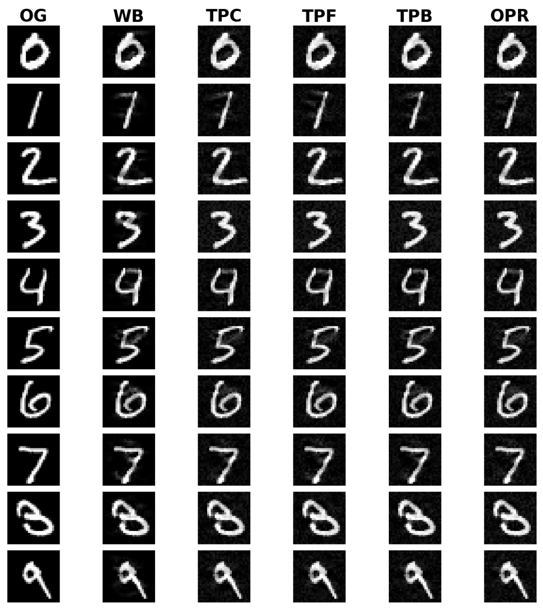
[26] Yan Zhang, Yi Zhou, Kaiyi Ji, and Michael M. Zavlanos. A new one-point residual-feedback oracle for black-box learning and control. *Automatica*, 136, 2 2022.

[27] Pu Zhao, Pin-Yu Chen, Siyue Wang, and Xue Lin. Towards query-efficient black-box adversary with zeroth-order natural gradient descent. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6909–6916, 2020.

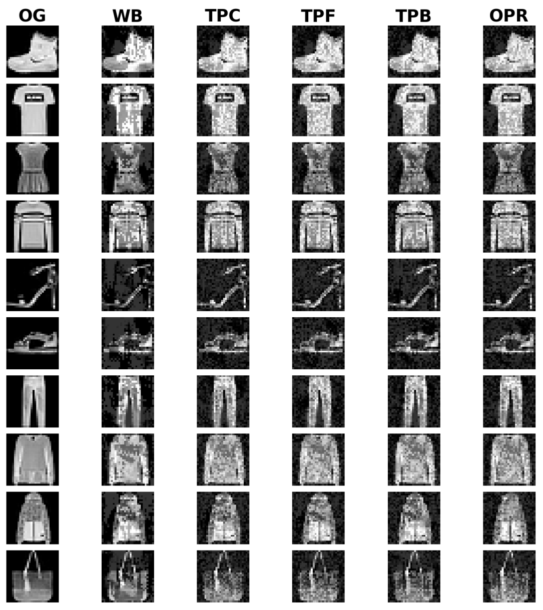
A Image Examples



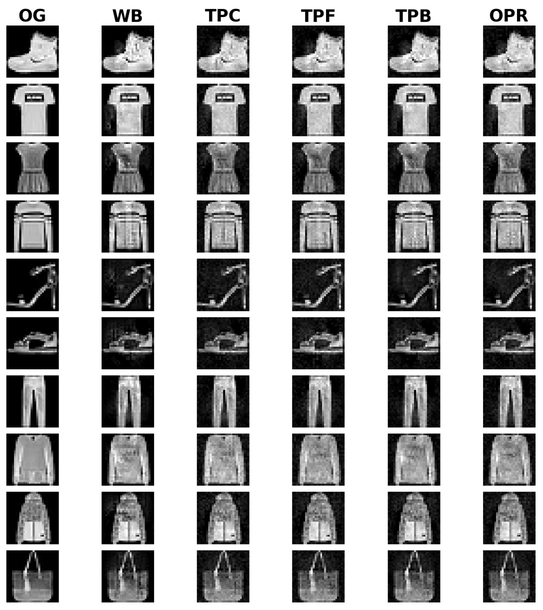
(a) L_∞-PGD attack with MNIST dataset



(b) L_2-PGD attack with MNIST dataset



(c) L_∞-PGD attack with F-MNIST dataset



(d) L_2-PGD attack with F-MNIST dataset

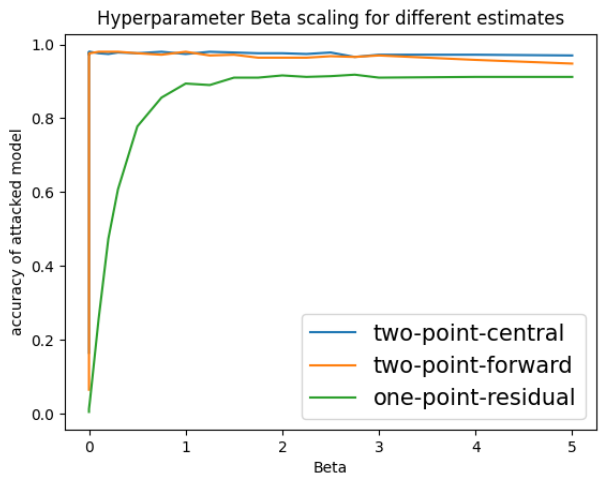
Figure 6: L_∞- and L_2-PGD attack with $\epsilon_\infty = 0.25$ and $\epsilon_2 = 2.5$ on a 3-layer convolutional network with MNIST and F-MNIST dataset. OG is the correctly classified original image and WB is an adversarial image made by the white-box version of the L_2-PGD attack. TPC, TPF, and TPB are the black-box attacks with two-point central, forward and backward estimates respectively. OPR is the attack using the one-point residual estimate.

B Setup details

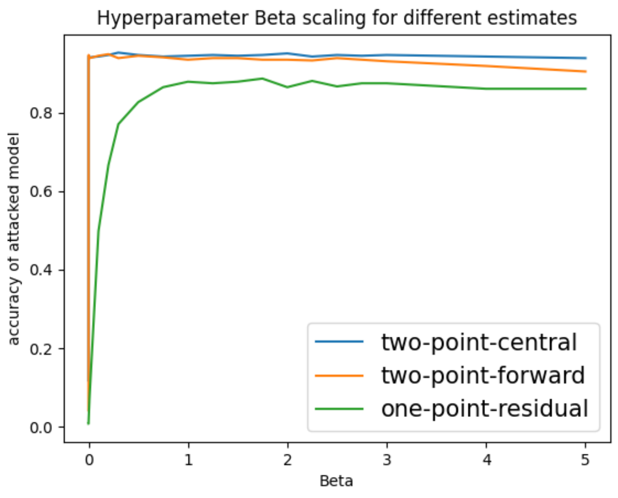
Model	
Model	3-layer convolutional network with max pooling (2,2) and 2 fully connected layers
Activation function	ReLu
Datasets	MNIST (99.55% accuracy), F-MNIST (90.05% accuracy)
Number of test images	500
L_∞ PGD parameters	
Number of PGD steps	50 [12]
ϵ tested	[0.003, 0.005, 0.01, 0.02, 0.03, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0]
α	0.025 [12]
L_2 PGD parameters	
Number of PGD steps	100 [17]
ϵ tested	[0.05, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0]
α	$\frac{2.5*\epsilon}{100}$ [17]
Estimate parameters	
Estimates	'white-box', 'one-point-residual', 'two-point-forward', 'two-point-backward', 'two-point-central'
Number of samples	50 [12]
Optimal β	1.5

Table 3: Hyperparameters used to get results.

C Hyperparameter tuning



(a) L_∞-PGD attack



(b) L_2-PGD attack

Figure 7: Different use of β at L_∞- and L_2-PGD attack with ϵ_∞=0.3, ϵ_2=3 and a learning rate α of 0.025 and 0.0075 respectively. The number of steps and the number of samples taken per gradient estimation are both 50. We've dismissed the two-point backward estimation from this graph as it's highly similar to the two-point forward estimate.