ZOO K-lixel: Zeroth Order Optimization with K-means Clustering and One Pixel Attack for Deep Neural Network Black-box Attacks

Group Members: Nicholas Ma (ID# 005035481), Fredrick Tsang (ID# 205068673), Yajie Wang (ID# 205029728), Miles Abel (ID# 005068626)

University of California Los Angeles, CA 90095, USA

Introduction

Adversarial black-box research has an important role in preventing deep neural network misclassification and recognition system obfuscation. Several algorithms, such as Zeroth Order Optimization (ZOO) [1], One Pixel Attack [2], Adversarial Deformation [3], and Universal Perturbation [4] have been developed to create adversarial perturbations on black-box models. In this report, we will explore the Adversarial Deformation, One Pixel Attack, and ZOO algorithms. We then propose a black-box image classification technique that imports the combination of a One Pixel attack and k-means clustering into the ZOO algorithm.

Our method, named ZOO K-lixel ("ZOO" + "K"-means + c"l"ustering + One P"ixel") , reduces ZOO attack time through selectively restricting pixel locations that are subject to attack. Our approach consists of grouping raw images by label; finding similarities between specific images through k-means clustering on each image label; running a 3 pixel attack on each image for multiple images classified to the same cluster; and identifying 40 most frequently selected pixels in each cluster. The data set including 40 most frequently selected pixels of different clusters used to modify the original ZOO algorithm. As compared to ZOO, our algorithm is able to achieve the same 100% attack success rate, comparable distortion (despite slightly higher), and reduced attack time.

General Attack Techniques

All existing attacks we compare are using CIFAR-10 dataset as the image source and LeNet and ResNet models for the image classifiers. Owing to time constraint in this project, results are tested on 100 images. In this project, before developing our own ZOO K-lixel approach, we have implemented three existing black-box attack algorithms.

First, Adversarial Deformation (ADef) addresses modifications to the entire image as opposed to minimizing change to a portion of image [5]. ADef computes a minimal vector field, such that when used to deform the original image, the resulting deformed image is misclassified [3]. As seen in Table 1, ADef has an attack success rate of 80% and 95% when attacking a LeNet and ResNet trained model, respectively. We will revisit the topic of ADef in the Future Work and Conclusion section because of its strong favorable integration potential with our method.

Model	Accuracy	Attack success rate	Smooth factor (σ)
LeNet	0.63	0.8	0.5
ResNet	0.921	0.95	0.5

Table 1: Adversarial Deformation Test Results Using a LeNet Trained Model and a ResNet Trained Model

Second, we have implemented the One Pixel attack, which is a black-box attack that generates image misclassification by modifying as few as a single pixel RGB value. Instead of measuring the overall distortion of the attacked image, One Pixel attack focuses on restricting the number of pixels that can be changed. This attack requires only the output probabilities of the neural network and uses an evolutionary algorithm called differential evolution (DE) to iteratively generate adversarial images [2]. For One Pixel attack, we have attack success rates that vary from 0.61 to 0.92 using one to five pixels on a LeNet trained model. However, for a ResNet trained model, we have lower attack success rates that span 0.37 to 0.78 again with 1 to 5 pixels, as shown in Table 2.

Model	**Accuracy**	**Pixels**	**Attack success rate**
LeNet	0.7488	1	**0.61**
LeNet	0.7488	3	**0.94**
LeNet	0.7488	5	**0.92**
ResNet	0.9231	1	**0.37**
ResNet	0.9231	3	**0.78**
ResNet	0.9231	5	**0.78**

Table 2: One Pixel Attack Test Results on a LeNet Trained Model and a ResNet Trained Model

Finally, ZOO attack plays a critical role in our algorithm. The ZOO algorithm uses zeroth-order stochastic coordinate descent to optimize on the target Deep Neural Networks (DNNs) [1]. Stochastic zeroth-order optimization uses derivative-free optimization or black-box optimization, primarily when gradients are difficult to access [6]. In ZOO, there are targeted and untargeted attacks as well as two selections of solvers: ADAM and Newton [1]. For ZOO K-lixel, we use targeted attacks and ZOO-ADAM for the solver. We have achieved an attack success rate of 100% using a maximum number of iteration of 100.

The ZOO attack achieves a high attack success rate at the expense of attack runtime. In this project, our main objective is explore various possibilities to reduce the attack runtime.

Our Concept: ZOO K-lixel

We devise a novel way to identify pixel importance in order to reduce ZOO computation time, while maintaining comparable performance metrics with respect to attacked image distortion and attack success rate. In exploring different attack algorithms, we discover that the constraints implemented by different attack algorithms are not necessarily mutually exclusive. For example, the goal of the ZOO algorithm is to change the image label while minimizing the amount of distortion. The goal of the One Pixel attack is to change the image label while restricting the number of pixels that the attack can change. We decide to explore the possibility of adding One Pixel attack's constraint to the ZOO algorithm to reduce the number of pixels that ZOO tries to change in each iteration.

The evolution of ZOO K-lixel consists of three stages of development: referred to as Approach 1, Approach 2, and Approach 3. In each approach, we use only 10,000 of the images in the CIFAR-10 dataset (1000 images per label) to improve the original ZOO algorithm. In turn, we test our algorithms using 100 test images. We compare the test results of Approach 2 and Approach 3 to the original ZOO algorithm.

For Approach 1, we perform a 40 pixel attack on each of the images, and generate a unique set of 40 pixel locations for each image. These 40 pixel locations for each image are fed into the ZOO algorithm for the adversarial attack. Figure 1 below provides a block diagram of the first approach.

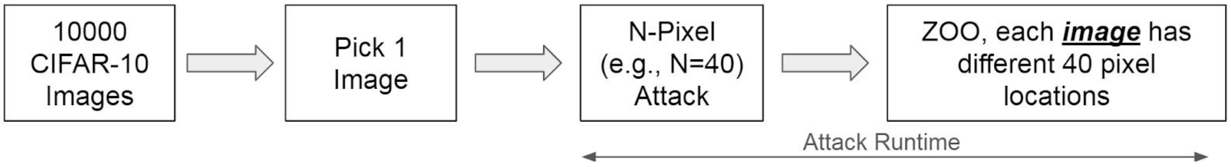
10000 CIFAR-10 Images
Pick 1 Image
N-Pixel (e.g., N=40) Attack
ZOO, each ***image*** has different 40 pixel locations
Attack Runtime

Figure 1: Block Diagram of Approach 1

We quickly found out that Approach 1 produced unsatisfactory results. A cursory look at the block diagram reveals a hint of oversimplification and naivety. There appears to at least two reasons in the failure of Approach 1. First, since the 40-Pixel attack is required for each incoming image, the attack runtime will need to include the time of running the 40-Pixel attack. This is contrary to our goal of reducing runtime. Second, for a 32x32 image in the CIFAR-10, allowing the One Pixel attack algorithm to change 40 pixels is a very loose restriction because the color value at each selected pixel can be changed to any value. As a result, the attack is almost always successful after a small number of iterations. Put differently, instead of providing 40 pixels that are most susceptible to an adversarial attack, the 40-Pixel attack basically randomly chooses 40 pixels because of the loose restriction. The chosen 40 pixels do not provide much insight to the ZOO algorithm.

In Approach 2, the Histogram Approach, the complexity in our selection of pixels increases. First, we group the images by labels (i.e., generating 10 sets of 1,000 images). For each set of 1,000 images of the same label, we perform a single pixel attack on each image. We choose a low number of pixel because we try to avoid the loose restriction problem as discussed in Approach 1. After running the single pixel attack for 1,000 images, we have a count of 1,000 pixels. We generate a histogram plot for the 1,000 pixels. We also repeat this process for other image labels to generate a total of ten histogram plots. Each histogram plot represents the occurrences attack at each pixels. We then select the 40 most frequent pixels for each label and input these 40 pixel locations per label into the ZOO algorithm.

Figure 2 provides a block diagram of the second approach, illustrating the five pre-processing stages we use at this point in an attempt to reduce computation time.

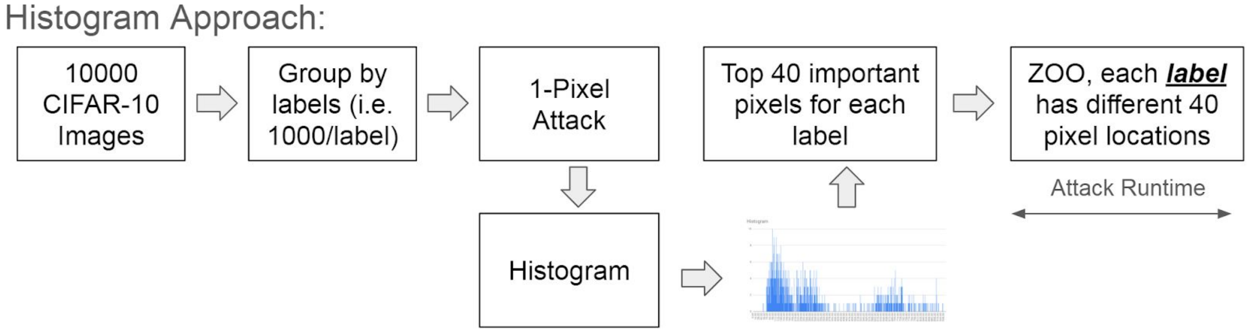
Histogram Approach:
10000 CIFAR-10 Images
Group by labels (i.e. 1000/label)
1-Pixel Attack
Histogram
Top 40 important pixels for each label
ZOO, each ***label*** has different 40 pixel locations
Attack Runtime

Figure 2: Block Diagram of Approach 2

We successfully reduce the computation time to approximately 11 seconds, but the average distortion raises to 1.02, as shown in Table 3. For example, Figure 3 illustrates a high distortion image of 6.0484, which clearly is recognizable by the human eye every time. Although we successfully reduce the execution time, we sacrifice performance by way of increasing image distortion. One of the reasons why Approach 2 does not provide satisfactory results is that the set of 1,000 images, despite having the same image label, do not share sufficiently similar image features. Hence, the set of 40 pixels does not adequately represent pixels that are most susceptible to an adversarial attack for a particular image label. We expected the unsatisfactory results when we saw the histograms for different image labels, but we believed Approach 2 was still worth studying.

Attack success rate	Average Attack Time (seconds)	Attack Time Standard Deviation (seconds)	Average Distortion	Distortion Standard Deviation
100%	11.19	0.49	1.02	1.12

Table 3: Approach 2 ZOO Results on 100 CIFAR-10 Images

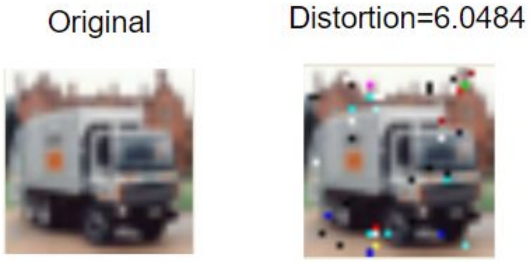

Figure 3: Approach 2 High Image Distortion (6.0484 is an approximate maximum for 100 images)

In our third approach, ZOO K-lixel, we focus on trying to discover image feature similarity among images of the same label because the Approach 2's result indicates that we need to further divide the images. We implement unsupervised learning with k-means clustering to generate five clusters for each of the ten labels (a total of 50 clusters). With these clusters we perform a three pixel attack on each clustered image. We generate a histogram displaying pixel location frequency and select the 40 most frequent pixel locations for each cluster. These 40 pixels per cluster as well as the cluster centroid become part of the modified ZOO algorithm. A cluster centroid is a 3072 (i.e. 32x32x3) dimension vector.

For an incoming image that we try to attack, we will be able to find the unaltered image label from the image classifier. The modified ZOO algorithm then determines the Euclidean distance between the image and a cluster centroid for five different cluster centroids that belong to the image label. As such, the image can be quickly classified to one of the clusters. The computation time of the cluster classification is negligible. We then run the ZOO algorithm with an additional restriction that the attack function can only change the 40 pixels for the particular cluster.

Our block diagram layout in Figure 4 provides a high level overview of our entire algorithm. Note we have a total of eight pre-processing steps identified on the block diagram prior to the execution of the modified ZOO.

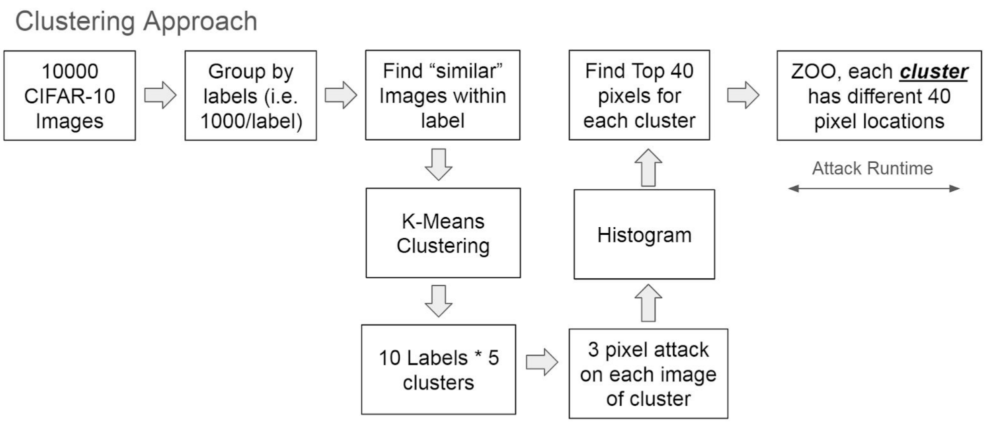
Clustering Approach
10000 CIFAR-10 Images
Group by labels (i.e. 1000/label)
Find “similar” Images within label
K-Means Clustering
10 Labels * 5 clusters
3 pixel attack on each image of cluster
Histogram
Find Top 40 pixels for each cluster
ZOO, each ***cluster*** has different 40 pixel locations
Attack Runtime

Figure 4: Block Diagram of Approach 3

Figure 5 illustrates the effect of k-means clustering for the “Airplane” label from a graphical perspective, and Figure 6 clearly shows the five clusters of the airplane images. We see our k-means clustering, which is unsupervised, selects image clusters in a fashion that appears intelligent even to the human observer. In Figure 6, note the images within a cluster tend to share similar color schemes, shadowing, and backgrounds. It is this intelligent sorting of pixels we aim to use to enhance the ZOO algorithm. However, the clustering does not detect edges well, which we expound upon in the Future Works and Conclusion section.

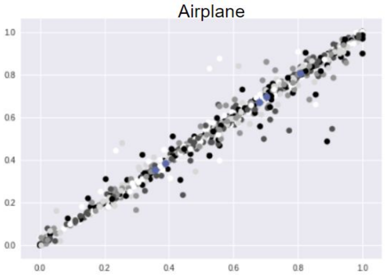

Figure 5: Clustering Data on the Airplane Label

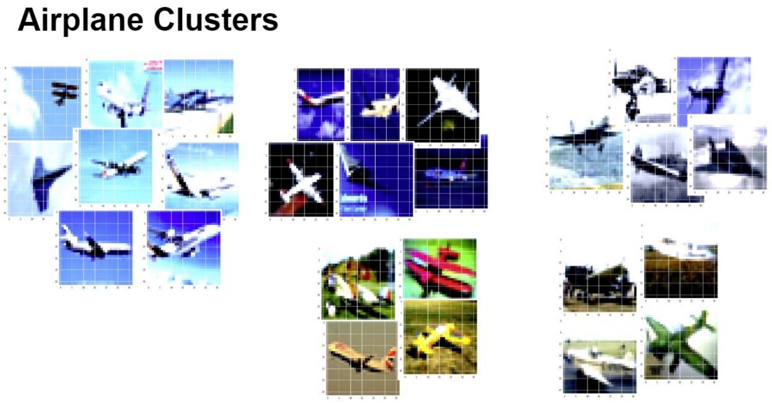

Figure 6: Airplane label Clusters Generated by Our Implementation of K-Means

As previously mentioned, we perform One Pixel attack on each image of a cluster and generate a histogram output of pixel location frequency for each cluster. In Figure 7, we show such two histograms for the “Airplane” image label. The histogram on the left is a histogram generated from Approach 2. The histogram on the right is a histogram generated from Approach 3. In the histogram on the left (Approach 2), the pixels that got attacked are largely uniformly distributed among the 1024 locations. On the contrary, in the histogram on the right, it appears that the pixels that got attacked are more concentrated on an area. Hence, it appears to us, by clustering the images before running the histogram, Approach 3 provides an improved result. The 40 most prevalent pixel location for each cluster in Approach 3 could in fact provide some useful information for the modified ZOO algorithm.

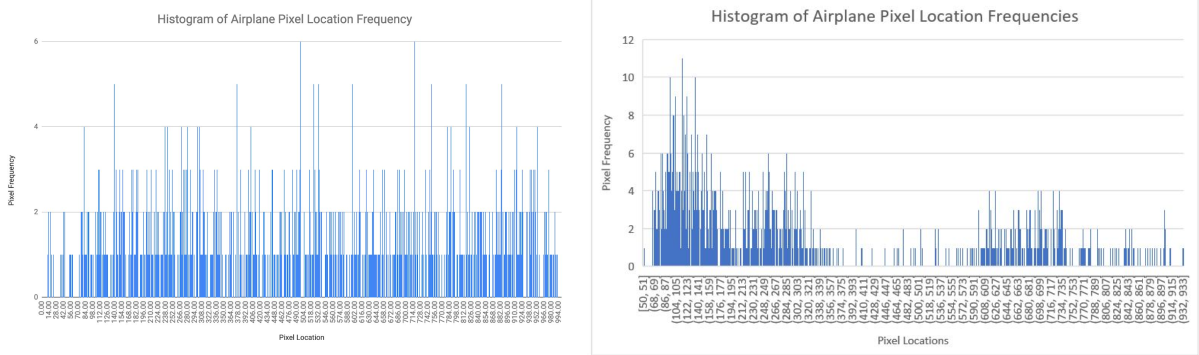

Figure 7: Histograms of Pixel Locations by Frequency

	Attack success rate	Average Attack Time (seconds)	Attack Time Standard Deviation (seconds)	Average Distortion	Distortion Standard Deviation
ZOO K-lixel	100%	11.70	1.04	0.85	0.74
ZOO	100%	20.35	0.16	0.35	0.34

Table 4: Comparison of ZOO K-lixel and Original ZOO Attack on 100 CIFAR-10 Images

The results of our ZOO K-lixel approach compared to the original ZOO performance are shown in Table 4 and Table 5. In Table 4, the image sample size is 100. We see that ZOO K-lixel successfully reduces the average attack time by approximately 50%, but the standard deviation of the attack time for ZOO K-lixel is 6.5 times larger than the original ZOO algorithm. The larger standard deviation is expected because classifying the images of the same label into 5 clusters will not fit the image features of each image in the CIFAR-10 dataset. Some images, despite being classified into a particular cluster, do not share sufficient image features with other members of the same cluster. As a result, the 40-pixel locations selected for that particular cluster might not fit every member of the cluster, leading to some high distortions. ZOO K-lixel effectively doubles the average distortion and standard deviation of the distortion. The results from Table 4 indicate that although our clustering process reduces the attack execution time, it does so at the expense of several other key metrics.

Figure 8 illustrates the effects of distortion on two sample images using our ZOO K-lixel approach. Both images are successfully reclassified from their original image label. The cat image is relabelled as a deer with a distortion of 0.832, which is within 2% of the average distortion level for the 100 selected images for Table 4. Although the distortion on the cat image is detectable by the human eye, it is subtle and likely overlooked. However, for the second image, the ship, a human observer is most likely not able to even find a difference between the original and the misclassified image. The ship has a distortion of 0.0133 and is reclassified as an airplane.

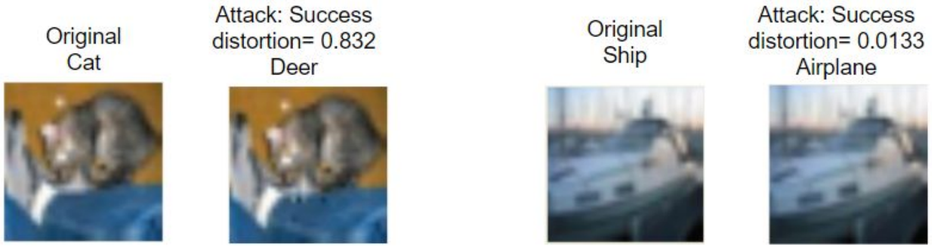

Figure 8: ZOO K-lixel Image Distortion Before and After Attack for an Average and Low-End Distortion Level

And, in Table 5, we see comparisons between the original ZOO algorithm, Approach 2 (the Histogram Approach) and Approach 3 (the Clustering Approach), which is also our final algorithm, ZOO K-lixel. For 10 images of all three attacks, we compare the attack time per image, distortion, and label reclassification. The label reclassification for all three methods match in general. However, the trends of larger standard deviations for both ZOO K-lixel and the Histogram Approach become apparent even for 10 images.

	Attack Time per Image (seconds)			**Distortion**			**Label Reclassification**		
Image	**ZOO**	**Histogram**	**Cluster**	**ZOO**	**Histogram**	**Cluster**	**ZOO**	**Histogram**	**Cluster**
1	21.227	11.65	11.320	0.4511	0.99090	0.8379	3 → 5	3→ 5	3 → 4
2	20.158	10.704	10.062	0.2472	0.78903	0.4472	3 → 1	3→ 1	3 → 1
3	20.256	10.737	10.139	0.0133	0.07077	0.0483	8 → 0	8→ 0	8 → 0
4	20.237	10.737	12.051	0.0636	0.221360	0.2293	0 → 8	0→ 8	0 → 8
5	20.230	10.703	11.785	0.030	0.10384	0.0924	6 → 4	6→ 4	6 → 4
6	20.228	10.746	11.774	0.1843	0.56851	0.3866	6 → 3	6→ 3	6 → 3
7	20.268	10.714	11.778	0 .4231	1.54462	1.2576	1 → 3	1→ 9	1 → 9
8	20.247	10.678	10.484	0.0978	0.26185	0.2288	6 → 2	6→ 2	6 → 2
9	20.229	10.725	11.801	0.2563	0.83560	0.6493	3 → 2	3→ 2	3 → 2
10	20.229	10.700	11.811	0.2782	0.80236	0.7631	1 → 8	1→ 8	1 → 8

Table 5: Comparison of Approach 3, ZOO K-lixel, to the Original ZOO Algorithm

Future Work and Conclusion

This report proposes a new method of improving the ZOO algorithm by pre-determining pixels that the ZOO attack can change to reduce execution time but maintain attack success rate and image distortion levels. By performing k-means clustering on CIFAR-10 classes in conjunction with a 3 pixel attack, we successfully identify pixels of higher importance. Feeding these higher importance pixels into the ZOO algorithm reduces the execution time of ZOO but increases the distortion of the attacked images.

We would like to extend our present work to testing ZOO K-lixel on ImageNet. Additionally, we would like to try different ways of clustering. For example, we can try clustering by RGB value as well as focus on edge detection in clustering as means of improving our pixel pre-selection process. The clustering process for the ZOO K-lixel approach presently does not do well with edge detection since the clustering input are the raw images with all three color channels. As we can see from Figure 6, the unsupervised clustering algorithm largely focuses on the color theme, which might not be the most important features in a Convolution Neural Network. We are interested in testing the impact of changing the images into grayscale before applying k-means clustering to force the k-means clustering to focus on other image features such as edges and shapes. Also, we can try applying a convolution kernel that is used for edge detection before applying k-means clustering. In a black-box attack, we are not supposed to know the structure of the CNN or the first layer of convolution kernel used in the CNN. However, for CIFAR-10, we can try an educated guess that a trained image classifier would likely have a 3x3 kernel for the first layer. We can try using different convolution kernels of different size to process the image before running the k-means clustering to compare the results.

Finally, we want to explore the use of deformation as a means to perform edge detection because of how the deformation algorithm calculates the vector field of the image.

References

[1] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, Cho-Jui Hsieh, “ZOO: Zeroth Order Optimization based Black-box Attacks to Deep Neural Networks without Training Substitute Models,” *ACM Conference on Computer and Communications Security (CCS) Workshop on Artificial Intelligence and Security (AISec),* 2017.

[2] Jiawei Su, Danilo Vasconcellos Vargas, Sakurai Kouichi, “One pixel attack for fooling deep neural networks,” *ArXiv:1710.08864*, 2019.

[3] Yipeng Hu, Eli Gibson, Nooshin Ghavami, Ester Bonmati, et al., “Adversarial Deformation Regularization for Training Image Registration Neural Networks,” *arXiv:1805.10665*, 2018.

[4] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, “Universal adversarial perturbations,” *arXiv:1610.08401*, 2016.

[5] Rima Alaifari, Giovanni S. Alberti, Tandri Gauksson, “ADef: an Iterative Algorithm to Construct Adversarial Deformations,” *arXiv:1804.07729*, 2018.

[6] Yining Wang, Simon Du, Sivaraman Balakrishnan, Aarti Singh, “Stochastic Zeroth-order Optimization in High Dimensions,” *arXiv:1710.10551*, 2017.