# EvoBA: An Evolution Strategy as a Strong Baseline for Black-Box Adversarial Attacks

**Catalin-Andrei Ilie** ( cilie@fmi.unibuc.ro )
University of Bucharest
**Marius Popescu**
University of Bucharest
**Alin Stefanescu**
University of Bucharest

**Research Article**

**Keywords:**

**Posted Date:** November 23rd, 2022

**DOI:** https://doi.org/10.21203/rs.3.rs-2294355/v1



**Additional Declarations:** No competing interests reported.

# EvoBA: An Evolution Strategy as a Strong Baseline for Black-Box Adversarial Attacks

Andrei Ilie *, Marius Popescu, and Alin Stefanescu

University of Bucharest, Romania
{cilie, marius.popescu, alin}@fmi.unibuc.ro

**Abstract.** Recent work has shown how easily white-box adversarial attacks can be applied to state-of-the-art image classifiers. However, real-life scenarios resemble more the black-box adversarial conditions, lacking transparency and usually imposing natural, hard constraints on the query budget.
It is usual for the black-box adversarial literature to assume that the attacker, while constrained by the number of model invocations and total image perturbation, has huge resources in terms of computation power and time. Therefore, most black-box adversarial attacks take a long time to run and require heavy computational processes. This makes it unfeasible for day-to-day computer vision systems developers to run continuous robustness checks against their models, by playing the attacker role and seeing when and how their model fails.
Therefore, we propose a class of black-box attacks based on simple evolutionary strategies: **EvoBA**, **EvoBA-E**, and **EvoBA-ET**. These are fast, query-efficient attacks, aiming to minimize the $L_0$ adversarial perturbations, and do not require any form of training.
We benchmark these attacks on the CIFAR-100 dataset and observe that they bear different trade-offs in terms of the count of queries required to fool the models and the $L_0$ adversarial distance. However, **EvoBA** strikes out as being the fastest of the three, so we further focus on comparing it against well-known adversarial attacks. We also build and open-source [1] a tool that can be used to run quick robustness checks of generic computer vision classifier systems.
**EvoBA** shows efficiency and efficacy through results that are in line with much more complex state-of-the-art black-box attacks such as **AutoZOOM**. It is more query-efficient than **SimBA**, a simple and powerful baseline black-box attack, and has a similar level of complexity. Therefore, we propose it both as a new strong baseline for black-box adversarial attacks and as a fast and general tool for gaining empirical insight into how robust image classifiers are with respect to $L_0$ adversarial perturbations.
There exist fast and reliable $L_2$ black-box attacks, such as **SimBA**, and $L_\infty$ black-box attacks, such as **DeepSearch**. We propose **EvoBA** as a query-efficient $L_0$ black-box adversarial attack which, together with the aforementioned methods, can serve as a generic tool to assess the empirical robustness of image classifiers. The main advantages of such methods are that they run fast, are query-efficient, and can easily be integrated in image classifiers development pipelines.
While our attack minimises the $L_0$ adversarial perturbation, we also report $L_2$, and notice that we compare favorably to the state-of-the-art $L_2$ black-box attack, **AutoZOOM**, and of the $L_2$ strong baseline, **SimBA**.

* Corresponding author

[1] https://github.com/andreiilie1/BBAttacks

## 1 Introduction

With the increasing performance and applicability of machine learning algorithms, and in particular deep learning, the safety of these methods has become more relevant than ever. There has been growing concern over the course of the last few years regarding adversarial attacks, i.e., algorithms which are able to fool machine learning models with minimal input perturbations, as they were shown to be very effective.

Ideally, theoretical robustness bounds should be obtained in the case of critical software involving image classification components. There has been important recent research in this direction [10, 13, 7, 25], but most often the algorithms generating these bounds work for limited classes of models, do not scale well with larger neural networks, and require complete knowledge of the target model's internals. Therefore, complementary empirical robustness evaluations are required for a better understanding of how robust the image classifiers are. In order to achieve this, one has to come up with effective adversarial attacks that resemble real-life conditions, such as in the black-box query-limited scenario.

In general, adversarial attacks are classified as either white-box or black-box. White-box adversarial attacks, where the attacker has complete knowledge of the target model, were shown to be particularly successful, most of them using gradient-based methods [31, 9, 2]. In the case of black-box adversarial attacks, the attacker can only query the model, and has no access to the model internals and to the data used for training. These restrictions make the black-box adversarial setup resemble more real-life scenarios. Furthermore, the attacker usually has to minimise the number of queries to the model, either due to time or monetary constraints (such as in the case of some vision API calls).

Previous state-of-the-art black-box adversarial attacks focused on exploiting the transferability phenomenon, which allowed the attackers to train substitute models imitating the target one, and perform white-box attacks on these [22, 23]. More recently, a class of black-box adversarial attacks, called Zeroth Order Optimization (**ZOO**) [5], has gained momentum, providing one of the current state-of-the art attacks, **AutoZOOM** ([32]). Interestingly, a much simpler algorithm, **SimBA** (Simple Black-box Attack) [11], achieves a similar, slightly lower success rate than state-of-the-art attacks, including **AutoZOOM**, and requires a lower number of queries. Therefore, **SimBA** is proposed to be used as a default baseline for any adversarial attack, but it is itself an unexpectedly powerful algorithm.

We propose **EvoBA**, **EvoBA-E**, and **EvoBA-ET**: a class of untargeted black-box adversarial attacks that make use of simple evolutionary search strategies. Namely, **EvoBA** is a simple mutation-based greedy strategy, while **EvoBA-E** and **EvoBA-ET** are variations of the classic $\epsilon -$ greedy strategy. These attacks only require access to the output probabilities of the target model for a given input and need no extra training.

We benchmark the three attacks we introduce on the CIFAR-100 dataset and comment on the results and advantages of each. While from the query efficiency and perturbation norm points of view there are some trade-offs to be made, from the running time point of view **EvoBA** is the clear winner. **EvoBA** is usually on par or very close to the best results obtained by **EvoBA-E** or **EvoBA-ET**, but at least three times faster. Therefore, we focus our further analysis on **EvoBA**. We also build an open source tool for

fast assessments of arbitrary computer vision classifiers and provide usage examples. The code can be found at https://github.com/andreiilie1/BBAttacks.

**EvoBA** is more query-efficient than **SimBA** and **AutoZOOM**, and has a perfect success rate, surpassing **SimBA** and being aligned with **AutoZOOM** from this point of view. We designed **EvoBA** to minimize the $L_0$ adversarial perturbation, however we also report the $L_2$ norms of the perturbations it generates and compare them with the $L_2$ norms of **AutoZOOM** and **SimBA**, which are $L_2$ adversarial attacks. Despite our algorithm aiming to minimize a different metric than these methods, we achieve similar $L_2$ norms in the perturbations we generate, and a significantly better $L_0$.

**EvoBA** is a model-agnostic empirical tool that can be used as a test to assess how robust image classifier systems are with respect to $L_0$ noise. Therefore, **EvoBA** is both a simple and strong baseline for black-box adversarial attacks, but also a generic $L_0$ robustness evaluation method that can be used in a fast and reliable way with any system involving image classification.

While $L_2$ (Euclidean distance) and $L_\infty$ (maximum absolute pixel-wise distance) adversarial attacks are more commonly studied, $L_0$ (count of modified pixels distance) adversarial attacks can fit better real-life, physical settings, such as in the well-known cases of graffiti perturbations on stop-signs from [8] and of adversarial eyeglass frames from [27]. As [3] notes, "physical obstructions in images or malicious splicing of audio or video files are realistic threats that can be modeled as $L_0$ noise, whereas $L_2$ attacks may be more difficult to carry out in the physical world".

The main property of adversarially perturbed images is that they are very close from a human point of view to the original benign image, but trick the model into predicting the wrong class. While the notion of "closeness from a human perspective" is hard to quantify, there exists general consensus around using the $L_0, L_2,$ and $L_\infty$ norms as proxies for measuring the adversarial perturbations [33, 23, 22, 11, 34].

**EvoBA** focuses on the $L_0$ norm, is fast, query-efficient, and effective. Therefore, we propose using it together with similarly fast and efficient methods that focus on different norms, such as **SimBA**, which focuses on $L_2$, and **DeepSearch** ([34]), which focuses on $L_\infty$, to empirically evaluate the robustness of image classifiers. These methods can act together as a fast and general toolbox used along the way of developing systems involving image classification models. **EvoBA** can easily be incorporated in development pipelines for gaining fast insights into a model's $L_0$ empirical robustness, only requiring access to the classifier's prediction method and to a sample of target images together with their corresponding labels.

Moreover, many adversarial training methods focus on improving the robustness with respect to $L_0, L_2,$ and $L_\infty$ perturbations [26]. Therefore, it is important to be able to empirically evaluate how much the robustness of target models improves due to adversarial training with respect to these norms. A toolbox consisting of efficient and reliable attacks such as **EvoBA**, **DeepSearch**, and **SimBA** can serve this purpose.

**EvoBA** is most surprising through its dual nature, acting both as a strong and fast black-box adversarial attack, such as **SimBA**, but also achieving results which are in line with much more complex, state-of-the-art black-box attacks, such as **AutoZOOM**.

To wrap it up, we propose **EvoBA**, **EvoBA-E**, and **EvoBA-ET** as standard strong baselines for any black-box adversarial attacks. We focus on **EvoBA** due to being the

fastest and making it most suitable to be used as a tool that can provide empirical insight into how robust image classifiers are. Its main advantages are that it is as effective as state-of-the-art black-box attacks, such as **AutoZOOM**, and more query-efficient. While it is an $L_0$ adversarial attack, it achieves $L_2$ perturbations of similar magnitudes with state-of-the-art $L_2$ black-box attacks and requires no training. Furthermore, **EvoBA** is highly parallelisable, which allows it to run significantly faster than **SimBA**, the other powerful baseline black-box attack.

**Our contributions**:

- We propose **EvoBA**, **EvoBA-E**, and **EvoBA-ET**, which are fast, query-efficient, and effective black-box $L_0$ adversarial attacks. We compare the three attacks on a standard dataset (CIFAR-100) and decide to focus on **EvoBA**, due to being the fastest, while not trading off too much query efficiency. **EvoBA** can be used as a generic test for assessing the $L_0$ robustness of image classifiers.
- We show how **EvoBA** compares favourably to the state-of-the-art black-box $L_2$ attack **AutoZOOM**, while being significantly simpler.
- We show how **EvoBA** (an $L_0$ attack) serves a similar purpose to the other strong and simple baselines such as **SimBA** (an $L_2$ attack) and **DeepSearch** (an $L_\infty$ attack). These methods are suitable to be run together, as part of a robustness testing toolbox, while developing any image classification model.
- We open-source our work at https://github.com/andreiilie1/BBAttacks, where we provide code examples about how to use **EvoBA** as a tool on custom models and datasets.

**Paper outline**:

- In Section 2 we do a quick literature review around related work and position especially **EvoBA** (which will be the subject of head-to-head comparison against the attacks from the literature review) with respect to the presented methods.
- In Section 3 we introduce **EvoBA** together with the threat model we use. We provide pseudocode for **EvoBA** and explain the main ideas behind it. We also analyze its space and time complexities.
- In Section 4 we present our experimental methodology and setup, provide and analyse the quantitative results of **EvoBA** in comparison with other methods, and present some qualitative results.
- In Section 6 we wrap-up the main findings and propose future research directions for our work.

## 2 Related Work

While the black-box adversarial attack settings are far more restrictive than the white-box ones, their similarity to real-life scenarios has increasingly brought them into the spotlight of machine learning safety.

One general approach for black-box attacks exploits the transferability property of adversarial examples [22, 23]. The attacker can train their own substitute model and perform white-box attacks on it, which usually yield good adversarial samples for the

target model as well. However, [29] showed the limitations of this method, highlighting how not all white-box attacks and not all architectures transfer well.

A class of approaches that do not rely on transferability is based on Zeroth Order Optimization (ZOO), which tries to estimate the gradients of the target neural network in order to generate adversarial examples. One of the early algorithms in this direction, **ZOO** [5] managed to reach similar success rates with state-of-the-art white-box attacks, producing adversarial samples of comparable visual quality. In general, Zeroth Order Optimization refers to any functional optimization where gradients are not available, so the term has commonly been adapted in the field of black-box adversarial attacks as a general category of methods that estimate the gradients.

The main disadvantage of traditional ZOO-type attacks is that they usually require plenty of queries for approximating the coordinate-wise gradients. **AutoZOOM** [32] solved this issue by performing a dimensionality reduction on top of the target image, and then using the ZOO approach in the reduced space. It achieved results that are aligned with previous ZOO state-of-the-art methods, but managed to reduce the query count by up to $93\%$ on datasets such as MNIST, ImageNet, and CIFAR-10. Our method is comparable to **AutoZOOM** in terms of performance, achieving a similar success rate with a slightly lower query count, but has the main advantage of being considerably simpler. Our approach does not need to estimate gradients at all and, compared to **AutoZOOM-AE** - the more powerful attack from [32], it does not require any form of training or knowledge about the training data. In addition, **AutoZOOM** demands access to the output probabilities over all classes, unlike our method, which only requires the output class and its probability. While **AutoZOOM** is an attack that minimizes the $L_2$ perturbations norm, the three **EvoBA** attacks are focused on minimising the $L_0$ norm. However, we also report the $L_2$ of the attack we focus on (**EvoBA**) and notice that it is comparable to **AutoZOOM**'s results.

**SimBA** (Simple Black-box Adversarial Attack, [11]) is a very simple strategy that has comparable performance to the significantly more complex **AutoZOOM**. The method randomly iterates through all pixels, perturbing them with fixed noise amounts if this makes the model output a lower probability for the correct class. **EvoBA** has a similar complexity to **SimBA**, both of them being good candidates for strong adversarial attack baselines. **SimBA** is mainly powered up by the randomness of choosing which pixel to perturb, and the only information it uses is whether the correct probability decreases. In comparison, **EvoBA** strikes a better balance between exploration and exploitation, which is common for evolutionary algorithms. This makes **EvoBA** achieve a slightly better success rate than **SimBA**, with a lower query budget. Similarly to **AutoZOOM**, **SimBA** is an $L_2$ adversarial attack, while **EvoBA** focuses on $L_0$. Nevertheless, the average $L_2$ perturbation norm of **EvoBA** is slightly higher, but comparable to **SimBA**'s results. We also run **SimBA** and remark that the $L_0$ perturbation norms that **EvoBA** achieves are significantly better. Furthermore, **SimBA** does not allow for any kind of parallelisation, while the evolution strategy we use in **EvoBA** is highly parallelisable and, accordingly, faster.

**DeepSearch** ([34]) is another simple, yet very efficient black-box adversarial attack, which achieves results in line with much more complex methods. It is an $L_\infty$ attack, perturbing with very high probability all the pixels (maximal $L_0$ norm of the

perturbations), which makes it incomparable with the $L_0$ attack **EvoBA**. Similarly, while **EvoBA** optimizes the $L_0$ norm of the perturbations, it most often produces high $L_\infty$ distortions, with sometimes near-maximal perturbations for the few chosen pixels. While both **DeepSearch** and **EvoBA** have non-complex implementations, **EvoBA** is conceptually simpler. **DeepSearch** is based on the idea of linear explanations of adversarial examples ([9]), and exploits three main aspects: it devises a mutation strategy to perturb images as fast as possible, it performs a refinement on top of the earliest adversarial example in order to minimise the $L_\infty$ norm, and adapts an existing hierarchical-grouping strategy for reducing the number of queries ([21]). Furthermore, **EvoBA** has the advantage of being highly parallelisable, while **DeepSearch** is inherently sequential. **DeepSearch**, **EvoBA**, and **SimBA** are complementary methods that serve the similar purpose of efficient and reliable black-box attacks working under the query-limited scenario, each optimising the produced perturbations under a different norm.

As black-box adversarial attacks are ultimately search strategies in obscured environments, it has been natural to also explore the path of evolutionary algorithms. One notable example is **GenAttack** [1], an approach that follows the classic pattern of genetic algorithms. While it was developed at the same time with **AutoZOOM**, the authors report similar results for the targeted versions of the two methods, without providing any untargeted attack results. We focused on the untargeted scenario, and our results are also in line with **AutoZOOM**. **GenAttack** focuses on minimising the $L_\infty$ perturbation norm, and, in expectation, the $L_0$ it achieves is equal to the count of all pixels in the image. In comparison, **EvoBA** is an $L_0$ attack, achieving considerably small perturbations under this norm. In addition, **EvoBA** is less complex and more suitable for a strong baseline attack.

A related approach to ours, which also makes use of evolution strategies, is [20], which tries to minimise the $L_\infty$ norm of adversarial perturbations. It proposes different evolution strategies applied on top of a tiling approach inspired by [16], where the authors use a Bandits approach. The attacks they propose focus on minimising the $L_\infty$ norm of the perturbations and the authors do not report any other results regarding different norms. The $L_2$ cost of this approach is not clear, one of the main issues being that it can become rather high. The $L_0$ is equal in general to the number of pixels in the entire image, in comparison with **EvoBA**, which is $L_0$-efficient. Qualitatively, the applied adversarial tiles it generates are easily perceivable by a human, yielding grid-like patterns on top of the target image, while the samples produced by **EvoBA** are imperceptible (Figure 7, ImageNet) or look like benign noise (Figure 4, CIFAR-10).

**EvoBA-E** and **EvoBA-ET** are also based on evolutionary algorithms but are significantly less complex than **GenAttack** and the work introduced in [20], being simple alterations of the well-known $\epsilon-$ greedy strategy. **EvoBA-E** and **EvoBA-ET** are most similar to the work introduced in [20], as $\epsilon-$ greedy are naturally similar to most Bandits approaches. Our 2 attacks also take a tile-based approach in generating the perturbations, but the tiles are significantly smaller, not allowing for very visible artifacts, and focus on minimising $L_0$ (as opposed to $L_\infty$).

# 3 The Methods

## 3.1 Notation and threat model

We work under black-box adversarial settings, with limited query budget and $L_0$ perturbation norm. We consider the untargeted attack scenario, where an adversary wants to cause perturbation that changes the original, correct prediction of the target model for a given image to any other class.

We denote by $\mathbf{F}$ the target classifier. By a slight abuse of notation, we let $\mathbf{F}(\mathbf{x})$ be the output distribution probability of model $\mathbf{F}$ on input image $\mathbf{x}$ and $\mathbf{F}_k(\mathbf{x})$ be the output probability for class $k$. Then, $\mathbf{F}$ can be seen as a function $\mathbf{F} : \mathbb{I} \mapsto \mathbb{R}^K$, where $\mathbb{I}$ is the image space (a subset of $\mathbb{R}^{h \times w \times c}$) and $K$ is the number of classes.

As we are working under black-box conditions, we have no information about the internals of $\mathbf{F}$, but we have query access to it, i.e., we can retrieve $\mathbf{F}(\mathbf{x})$ for any $\mathbf{x} \in \mathbb{I}$. In fact, we will see that for our method we just need access to $\arg\max_k \mathbf{F}_k(\mathbf{x})$ and to its corresponding probability.

Let us consider an image $\mathbf{x}$, which is classified correctly by $\mathbf{F}$. The untargeted attack goal is to find a perturbed version $\tilde{\mathbf{x}}$ of $\mathbf{x}$ that would make

$$\arg\max_k \mathbf{F}_k(\tilde{\mathbf{x}}) \neq \arg\max_k \mathbf{F}_k(\mathbf{x}), \tag{1}$$

constrained by the query and $L_0$ bounds.

## 3.2 Solving a surrogate problem

It is usual for black-box attacks to deal with a surrogate optimization problem that tries to find a perturbed version $\tilde{\mathbf{x}}$ of $\mathbf{x}$ that minimizes $\mathbf{F}(\tilde{\mathbf{x}})$. This is clearly not equivalent to the formulation at (1), but it often yields good adversarial examples and is easier to use in practice.

In loose terms, this surrogate optimization problem can be formulated as follows for an image $\mathbf{x}$ with true label $y$:

$$\min_{\boldsymbol{\delta} \in \mathbb{R}^{h \times w \times c}} \mathbf{F}_y(\mathbf{x} + \boldsymbol{\delta}), \text{w.r.t. queries} \leq Q, \|\delta\|_0 \leq \epsilon. \tag{2}$$

In order to tackle (2), we adopt simple evolution strategies that yields results in line with state-of-the-art black-box attacks.

## 3.3 EvoBA

Our method (Algorithm 1, **EvoBA**) works by iteratively creating generations of perturbed images according to the following process: it selects the fittest individual in each generation (with lowest probability to be classified correctly), starting from the unperturbed image, then samples small batches of its pixels and randomly perturbs them, stopping when the fittest individual is either no longer classified correctly or when one of the constraints no longer holds (when either the query count or the distance become too large).

**Algorithm 1:** EvoBA

**Data:** black-box model $\mathbf{F}$, image $\mathbf{x}$, correct class $k$, query budget $Q$, $L_0$ threshold $\epsilon$, pixel batch size $B$, generation size $G$

PARENT $\leftarrow \mathbf{x}$
PREDICTION $\leftarrow \mathbf{F}_k(\mathbf{x})$
QUERY_CNT $\leftarrow 1$
**while** $\|\text{PARENT} - \mathbf{x}\|_0 < \epsilon$ *and* QUERY_CNT $< Q$ **do**
  OFFSPRING $\leftarrow [\,]$
  FITNESSES $\leftarrow [\,]$
  PIXELS $\leftarrow$ SAMPLE_PIXELS(PARENT, $B$)
  **for** IDX $\leftarrow 1 \ldots G$ **do**
    CHILD $\leftarrow$ PARENT
    **for** PIXEL $\leftarrow$ PIXELS **do**
      CHILD[PIXEL] $\leftarrow$ SAMPLE_VALUES()
    OFFSPRING $\leftarrow$ OFFSPRING $+$ [CHILD]
    PRED_CHILD $\leftarrow \mathbf{F}$(CHILD)
    QUERY_CNT $\leftarrow$ QUERY_CNT $+ 1$
    **if** $\arg\max$ PRED_CHILD $\neq k$ **then**
      **return** CHILD
    FITNESSES $\leftarrow$ FITNESSES $+ [1 -$ PRED_CHILD$_k$(CHILD)$]$
  BEST_CHILD $\leftarrow \arg\max$(FITNESSES)
  PARENT $\leftarrow$ OFFSPRING[BEST_CHILD]
**return** PERTURBATION FAILED

The function SAMPLE_PIXELS(PARENT, B) does a random, uniform sample over the pixels of PARENT, and returns a list of size at most $B$ (the sampling is done with repetition) containing their coordinates. Its purpose is to pick the pixels that will be perturbed. The function SAMPLE_VALUES generates pixel perturbed values. For our $L_0$ objective, we let it pick uniformly a random value in the pixel values range.

The algorithm follows a simple and general structure of evolution strategies. The mutation we apply on the best individual from each generation is selecting at most $B$ pixels and assigning them random, uniform values. The fitness of an individual is just the cumulative probability of it being misclassified.

One important detail that is not mentioned in the pseudocode, but which allowed us to get the best results, is how we deal with multi-channel images. If the target image has the shape $h \times w \times c$ (height, width, channels), then we randomly sample a position in the $h \times w$ grid and add all of its channels to the PIXELS that will be perturbed. We hypothesise that this works well because of a "inter-channel transferability" phenomenon, which allows **EvoBA** to perturb faster the most sensible zones in all the channels. Note that this yields a cost of $c$ for every grid-sample to the $L_0$ perturbation norm, so in the case of ImageNet or CIFAR-10 images it counts as 3, and in the case of MNIST it counts as 1.

The query budget and $L_0$ constraints impose a compromise between the total number of generations in an **EvoBA** run and the size $G$ of each generation. The product of these is approximately equal to the number of queries, so for a fixed budget we have to

strike the right balance between them. The bigger $G$ is, the more we favour more exploration instead of exploitation, which should ultimately come at a higher query cost. The smaller $G$ is, the search goes in the opposite direction, and we favour more the exploitation. As each exploitation step corresponds to a new perturbation, this will result in bigger adversarial perturbations. Furthermore, lacking proper exploration can even make the attack unsuccessful in the light of the query count and $L_0$ constraints.

The batch size $B$ allows selecting multiple pixels to perturb at once in the mutation step. It is similar to the learning rate in general machine learning algorithms: the higher it is, the fewer queries (train steps, in the case of machine learning algorithms) we need, at the cost of potentially missing local optimal solutions.

The space complexity of Algorithm 1 is $\mathcal{O}(G \times \text{size}(\mathbf{x}))$. As the size of $\mathbf{x}$ is in general fixed, or at least bounded for specific tasks, we can argue that the space complexity is $\mathcal{O}(G)$. However, **EvoBA** can be easily modified to only store two children at a moment when generating new offspring, i.e., the currently generated one and the best one so far, which makes **EvoBA**'s space complexity $\mathcal{O}(1)$.

The dominant component when it comes to time complexity is given by $\mathbf{F}$ queries. In the current form of Algorithm 1, they could be at most $\min(\frac{\epsilon}{B} \times G, Q)$. Assuming that the budget $Q$ will generally be higher, the time complexity would roughly be $\mathcal{O}(\frac{\epsilon}{B} \times G \times (\text{query cost of } \mathbf{F}))$. This is merely the sequential time complexity given by the unoptimised pseudocode, however the $G$ $\mathbf{F}$-queries can be batched, yielding much faster, parallelised runs.

### 3.4 EvoBA-E and EvoBA-ET

**EvoBA-E** (Algorithm 2) is based on the well-known $\epsilon-$greedy strategy. It is parametrised by a set of pixel groups $\{P_1, P_2, \ldots, P_G\}$ that form a partition (i.e $P_i \cap P_j = \emptyset$ for any $i \neq j$ and $\bigcup_i P_i$ represents all pixel indices of an image) and an $\epsilon$ that represents the trade-off between exploration and exploitation that **EvoBA-E** will make (higher $\epsilon$ means higher exploration).

We explore attacking a random pixel group from the $P_i$'s with probability $\epsilon$ and attack one of the pixel groups with maximum historical average reward (decrease of probability of an image to be classified correctly) with probability $1 - \epsilon$.

For attacking a random pixel group $P_i$ we randomly sample a pixel from it and change its value according to a randomised schema. In our experiments, we will choose new pixels according to uniform perturbations. Note that this is prone to generate high $L_2$ perturbations, but our goal is to minimise the $L_0$ distortion, modifying as few pixels as possible.

We note that this is not a standard $\epsilon$ − greedy / multi-armed bandits setup, as the expected rewards we compute are not independent of the history of samples. Pixels changed in the past will affect the future probabilities of attacking randomly selected groups in the future.

Unlike the classic multi-armed bandits approaches, we exploit the advantage of being able to check the potential rewards without necessarily taking their inflicted loss. Therefore, we assess whether the random pixel modifications we do decrease the probabilities of an image being classified correctly, and only in this case update the image

we are attacking. The cost we pay for any operation, no matter if it has a positive reward or not, is consuming our query budget.

In our experiments, the chosen pixel groups $\{P_1, P_2, \ldots, P_G\}$ form a grid over the image. There are other non-trivial choices, such as segmentation-based pixel groups. One could even use light explainability methods such as LIME ([24]) that also provides a simple image segmentation schema, further utilising the importance scores as priors for sampling over the attacked pixel groups. We leave this for further research.

**EvoBA-ET** works similarly to **EvoBA-E**, with a single change in the way it updates the currently attacked image. Instead of just requiring the reward to be positive in order to update the image, it requires the reward to be greater than a small threshold. Every $max\_rounds\_until\_decay$ the image hasn't been updated, we multiply the threshold by a $decay\_factor < 1$, which is close to 1. This will allow the search to escape local plateaus. Once we succeed in updating the image, we multiply the threshold by an $increase\_factor > 1$, which is also close to 1. We denote the initial probability threshold by $init\_threshold$.

The main disadvantage of **EvoBA-E** and **EvoBA-ET** against **EvoBA** is that they are not easily parallelisable. Their logic is highly iterative, with exploration alternating exploitation randomly. In the case of **EvoBA**, exploration and exploitation happens naturally at each iteration, and the exploration is highly parallelisable.

## 4 Experiments

### 4.1 Experimental Setup

We used TensorFlow/Keras for all our experiments. All the experiments were performed on a MacBook with 2,6 GHz 6-Core Intel Core i7, without a GPU.

We ran locally **SimBA**, the strong and simple $L_2$ black-box adversarial attack, and compared **EvoBA** to both our local **SimBA** results and reported results from the paper introducing it ([11]).

We also monitored and compared against the **AutoZOOM** results, but for these we have used different target models, as the main focus was on comparing to **SimBA** (which already achieves results which are in line with state-of-the-art approaches, such as **AutoZOOM**, being more query-efficient), so we adopted their models.

We don't do a head-to-head comparison with the efficient $L_\infty$ black-box baseline **DeepSearch**, as it is a direct consequence of their approach that they get near-maximal $L_0$ perturbations, while **EvoBA** aims to optimize the $L_0$ norm. Similarly, **EvoBA** creates high $L_\infty$ perturbations, as it modifies very few pixels with random, possibly big quantities. Therefore, **DeepSearch** and **EvoBA** are complementary methods that should be used together, but a direct comparison of their results is not suitable, as the $L_0$ and $L_\infty$ objectives are partly contradictory.

In order to reduce our algorithm selection bias, we initially run a head-to-head comparison of our three proposed methods on top of CIFAR-100, attacking a classic VGG-16 model ([28]). For this, we fix 1000 randomly sampled CIFAR-100 images, and run various configurations of **EvoBA**, **EvoBA-E**, and **EvoBA-ET** against them. We will comment on why it makes the most sense to choose **EvoBA** as the engine of

**Algorithm 2:** EvoBA-E

**Data:** black-box model $\mathbf{F}$, image $\mathbf{x}$, correct class $k$, query budget $Q$, $L_0$ threshold $\epsilon$, count of pixel groups $G$, pixel groups (a partition of pixel indices) PIXEL_GROUPS, exploration ratio $\epsilon$

EXPLORATION_COUNT $\leftarrow [0] \times G$
EXPLORATION_VALUES $\leftarrow [0] \times G$
PARENT $\leftarrow \mathbf{x}$
PREDICTION $\leftarrow \mathbf{F}_k(\mathbf{x})$
QUERY_CNT $\leftarrow 1$
**while** $\|\text{PARENT} - \mathbf{x}\|_0 < \epsilon$ *and* QUERY_CNT $< Q$ **do**
    EXPLORE $\leftarrow$ (UNIFORM_RANDOM(0,1) $< \epsilon$)
    **if** EXPLORE **then**
        GROUP $\leftarrow$ UNIFORM_DISCRETE_RANDOM(0,G)
    **else**
        GROUP $\leftarrow \arg\max$(EXPLORATION_VALUES)
    SIZE_SAMPLE $\leftarrow size$(PIXEL_GROUPS[GROUP])
    ATTACK_PIXEL_IDX $\leftarrow$ UNIFORM_DISCRETE_RANDOM(0, SIZE_SAMPLE)

    CHILD $\leftarrow$ COPY(PARENT)
    CHILD[ATTACK_PIXEL] $\leftarrow$ RANDOM_PIXEL()

    PRED_CHILD $\leftarrow \mathbf{F}$(CHILD)
    QUERY_CNT $\leftarrow$ QUERY_CNT $+ 1$
    REWARD $\leftarrow$ PREDICTION $-$ PRED_CHILD$_k$

    N $\leftarrow$ EXPLORATION_COUNT[GROUP]
    EXPLORATION_COUNT[GROUP] $\leftarrow$ N $+ 1$

    V $\leftarrow$ EXPLORATION_VALUES[GROUP]
    EXPLORATION_VALUES[GROUP] $\leftarrow$ V $\cdot \frac{N}{N+1} +$ REWARD $\cdot \frac{1}{N+1}$

    **if** $\arg\max$ PRED_CHILD $\neq k$ **then**
        **return** CHILD

    **if** REWARD $> 0$ **then**
        PARENT $\leftarrow$ CHILD
        PREDICTION $\leftarrow$ PRED_CHILD$_k$
**return** PERTURBATION FAILED

a robustness-checking tool, and then move on to compare it against other black-box adversarial attacks on different datasets.

We ran multiple experiments over four other datasets: MNIST [18], CIFAR-10 [17], and ImageNet [6].

On MNIST we have been running our experiments on a classic target LeNet architecture [19], while **SimBA** does not report any results on this dataset, and **AutoZOOM** uses a similar architecture to ours, with additional dropout layers (taken from [4]). However, their target models are trained by default with distillation, which is likely to make perturbations harder. For comparing with **SimBA** on MNIST, we ran the attack ourselves and aggregated various metrics.

We initially used MNIST to validate **EvoBA** against a completely random black-box adversarial attack, similar to the one introduced in [14]. The purely random strategy iterates by repeatedly sampling a bounded number of pixels from the original target image and changing their values according to a random scheme. While the purely random method introduced in [14] achieves surprising results for such a simple approach, it does a very shallow form of exploration, restarting the random perturbation process with each miss (i.e., with each perturbed image that is still classified correctly). We will refer to the completely random strategy as **CompleteRandom** in the experiments below.

For CIFAR-10, we use a ResNet-50 [12] target model, similarly to **SimBA**, and we compare to both their reported results and to our local run of their attack.

For ImageNet, we have used a similar target ResNet-50 to the one used in **SimBA**, while **AutoZOOM** used InceptionV3 [30].

We will use the following shorthands in the results below: **SR** (Success Rate), **QA** (Queries Average), **L0** (Average of L0 successful perturbations), **L2** (Average L2 norm of successful perturbations).

We will refer to **EvoBA** that perturbs at most $B$ pixels at once and that has generation size $G$ as **EvoBA**$(B, G)$. We will explicitly mention in each section which thresholds were used for the experiments.

For all the local runs of **SimBA**, we have been using $\epsilon = 0.2$ (a hyperparameter specific to **SimBA**), which was also used in the paper [11]. We only replicated locally the results of the Cartesian Basis version of **SimBA**, which resembles more an $L_0$ adversarial attack, but which is less efficient than the Discrete Cosine Transform (DCT) version from the paper. Therefore, we will use **SimBA-LCB** to refer to the local run of SimBA on top of the exact same target models as **EvoBA**, with $\epsilon = 0.2$. We will use **SimBA-CB** to refer to the results of the Cartesian Basis paper results, and **SimBA-DCT** for the DCT paper results.

In the cases where the **AutoZOOM** paper [32] provides data, we will only compare to **AutoZOOM-BiLIN**, the version of the attack which requires no additional training and data, and which is closer to our and **SimBA**'s frameworks.

### 4.2 Benchmarks on CIFAR-100

As introduced in the experimental setup, we denote the **EvoBA** that perturbs at most $B$ pixels at once and that has generation size $G$ by **EvoBA**$(B, G)$.

For both **EvoBA-E** and **EvoBA-ET** we use pixel groups given by $4 \times 4$ patches, yielding a chess-like $8 \times 8$ grid. We denote the **EvoBA-E** with an exploration ratio of $\epsilon$ by **EvoBA-E**$(\epsilon)$. We denote the **EvoBA-ET** with an exploration ratio of $\epsilon$, rounds until decay $R$, decay factor $\alpha$, and increase factor $\beta$ by **EvoBA-ET**$(\epsilon, R, \alpha, \beta)$. We fix the initial threshold to $0.1$, which represents requiring perturbations to decrease the correct probability by $10\%$. This likely means the threshold will be decreased fast through the decaying mechanism, with the advantage of allowing to capture early the obvious perturbations with huge impact.

We introduce our CIFAR-100 benchmark results in Table 1.

Increasing the number of individuals in a generation $G$ for **EvoBA**$(B, G)$ decreases the $L_0$ distance of the adversarial samples we find, favouring exploration, at the cost of more queries. Increasing the count of pixels to be attacked in a batch $B$ has the effect of finding adversarial examples faster, through lower counts of queries, but at larger $L_0$ distances.

In the case of **EvoBA-E**, there is no obvious performance disadvantage for either the query count or the $L_0$ distance by growing $\epsilon$ up to $0.8$, point at which the performance plateaus. Therefore, **EvoBA-E**$(0.8)$ seems to provide the best results, both in terms of efficiency and of average perturbation $L_0$ distance. This means that the optimal **EvoBA-E** is highly explorative, learning and updating the potential rewards over the image groups $80\%$ of the time, and targeting the highest reward groups in only $20\%$ of the time. We observe that **EvoBA-E**$(0.8)$ is competitive with the best **EvoBA**$(B, G)$ results.

**EvoBA-E** is able to achieve decent $L_0$ average distances (order of $60$) at the cost of very few queries (order of $40 - 50$). On the other side, **EvoBA** is able to achieve significantly lower $L_0$ distances (even down to the order of $10 - 40$), by using significantly more queries (in the order of $100 - 200$). This means that there is no clear winner in **EvoBA** versus **EvoBA-E** and one has to choose the appropriate trade-off.

**EvoBA-ET** works best when favouring a balanced split of exploration and exploitation ($\epsilon = 0.4$ in our results) and when allowing a very fast decrease of the reward threshold that it applies ($\alpha = 0.5$) and a slow increase ($\beta = 1.05$). While the results it gets are competitive, there exist **EvoBA** configurations which are strictly better from every point of view than **EvoBA-ET**. Therefore, in our search for the perfect attack to apply as part of a generic robustness tool, we focus our attention on **EvoBA-E** and **EvoBA**.

Inspecting some of the best **EvoBA** runs, we notice that **EvoBA**$(1, 40)$ ran in $0.415s$ per image sample, **EvoBA**$(8, 40)$ in $0.22s$ per image, and **EvoBA**$(4, 10)$ in $0.335s$ per image. In comparison, **EvoBA-E**$(0.1)$ ran in $1.25s$ per image, and **EvoBA-E(0.8)** in $1.19s$ per image.

This means that the best **EvoBA** runs are faster by three times than the best runs of **EvoBA-E**.

As our main motivation is to provide a fast and reliable black-box attack that developers can use for fast robustness iterations on their computer vision classifiers, we resort to using the method that is definitely faster: **EvoBA**. From now on, we will focus our attention on **EvoBA** experiments against other well-known black-box adversarial attacks. **EvoBA** is also the core of our fast and reliable robustness-checking tool.

| | SR | QA | L0 |
|---|---|---|---|
| **EvoBA(1,10)** | 99.9% | 95.08 | 28 |
| **EvoBA(1,20)** | 100% | 147.5 | 21.82 |
| **EvoBA(1,40)** | 100% | 236.4 | 17.56 |
| **EvoBA(1,80)** | 100% | 400 | 14.9 |
| **EvoBA(2,10)** | 100% | 70.09 | 40.97 |
| **EvoBA(2,20)** | 100% | 107.8 | 31.8 |
| **EvoBA(2,40)** | 100% | 180.6 | 26.72 |
| **EvoBA(2,80)** | 100% | 311.4 | 23.12 |
| **EvoBA(4,10)** | 100% | 49.59 | 57.2 |
| **EvoBA(4,20)** | 100% | 80.18 | 46.8 |
| **EvoBA(4,40)** | 100% | 135.7 | 39.91 |
| **EvoBA(4,80)** | 100% | 236.8 | 35.07 |
| **EvoBA(8,10)** | 100% | 35.96 | 81.69 |
| **EvoBA(8,20)** | 100% | 58.3 | 67.44 |
| **EvoBA(8,40)** | 100% | 101.4 | 59.24 |
| **EvoBA(8,80)** | 100% | 182 | 53.55 |
| **EvoBA-E(0)** | 99.9% | 55.88 | 66.05 |
| **EvoBA-E(0.01)** | 100% | 55.82 | 65.6 |
| **EvoBA-E(0.1)** | 100% | 55.07 | 64.94 |
| **EvoBA-E(0.2)** | 100% | 51.08 | 63.73 |
| **EvoBA-E(0.4)** | 100% | 46 | 61.9 |
| **EvoBA-E(0.6)** | 100% | 42.76 | 58.6 |
| **EvoBA-E(0.8)** | 100% | 42 | 57.68 |
| **EvoBA-ET(0.2, 20, 0.9, 1.05)** | 100% | 610.7 | 36.09 |
| **EvoBA-ET(0.2, 50, 0.5, 1.05)** | 100% | 247.8 | 38.77 |
| **EvoBA-ET(0.4, 20, 0.5, 1.01)** | 100% | 148.8 | 40.79 |
| **EvoBA-ET(0.2, 20, 0.5, 1.05)** | 100% | 158.4 | 43.33 |

**Table 1.** Benchmarks of our proposed methods on top of CIFAR-100.

### 4.3 Results on MNIST

We only experiment with **EvoBA**$(B, G)$ with $B > 1$ on MNIST, and focus on $B = 1$ for subsequent experiments, for which we impose an $L_0$ perturbation limit of $100$ and a query threshold of $5000$.

Running **SimBA** (**SimBA-LCB**) on a local machine with the mentioned specifications, requires an average of **93s** per MNIST sample. In comparison, all the **EvoBA** experiments on MNIST took between **1.94s** and **4.58s** per sample.

We also report the **CompleteRandom** results, for which we impose an $L_0$ perturbation limit of $100$ and a query threshold of $5000$, similarly to the constraints we use for **EvoBA**.

We randomly sampled $200$ images from the MNIST test set and ran **SimBA-LCB**, **EvoBA**, and **CompleteRandom** against the same LeNet model [19]. For reference, we also add the results of **AutoZOOM**, which are performed on a different architecture. Therefore, the results are not directly comparable with them. $L_0$ data is not available for **AutoZOOM**, but it is usual for ZOO methods to perturb most of the pixels, so it is very likely that the associated $L_0$ is very high.

| | SR | QA | L0 | L2 |
|---|---|---|---|---|
| **EvoBA(1,10)** | 100% | 301.4 | 29.32 | 3.69 |
| **EvoBA(1,20)** | 100% | 549.4 | 26.88 | 3.58 |
| **EvoBA(1,40)** | 100% | 894.2 | 21.92 | 3.33 |
| **EvoBA(2,20)** | 100% | 312.6 | 30,72 | 3.65 |
| **EvoBA(2,30)** | 100% | 265.4 | 38,6 | 3.89 |
| **SimBA-LCB** | 56% | 196.86 | 48.16 | 2.37 |
| **AutoZOOM-BiLIN** | 100% | 98.82 | - | 3.3 |
| **CompleteRandom** | 60.5% | 576.1 | 93.98 | 5.59 |

**Table 2.** MNIST results. **SimBA-LCB** has a very low success rate in the case of a LeNet architecture. If we take a look at the 56% images perturbed by **EvoBA(1,10)** for example, the QA becomes 192.8, which is not apparent from the table, but which is more efficient than **SimBA-LCB**.

We remark that all the **EvoBA** configurations have a $100\%$ success rate, and **SimBA-LCB** achieves $56\%$. While **SimBA** generally achieves near-perfect success rates on other tasks, one could argue that attacking a simple target model such as LeCun on a relatively easy task such as MNIST is much harder than performing attacks for more intricate target models and tasks. This is a natural trade-off between the complexity of a model and its robustness.

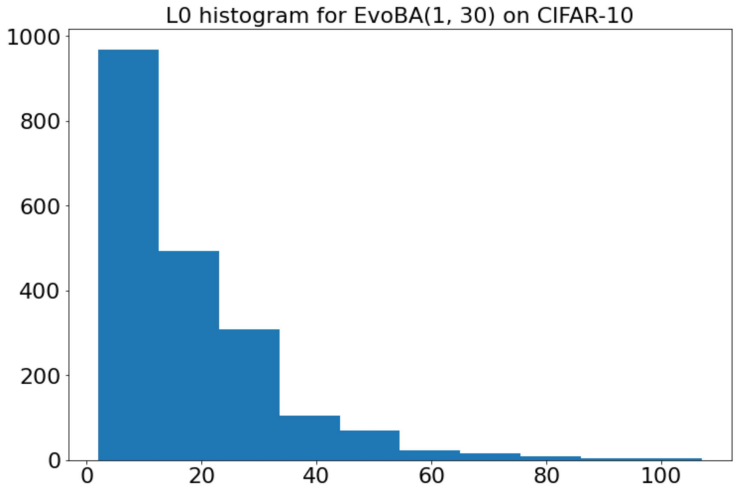


**Fig. 1.** Histogram of $L_0$ perturbation norms obtained by **EvoBA(1,30)** on CIFAR-10 with target model ResNet-50. The distribution is very heavy on small values, with few outliers.

If we restrict **EvoBA(1,10)** to its top $56\%$ perturbed images in terms of query-efficiency, it achieves an average of $192.79$ queries, which is below **SimBA-LCB**'s queries average of $196.86$. Similarly, if we restrict **EvoBA(1,10)** to its top $56\%$ per-

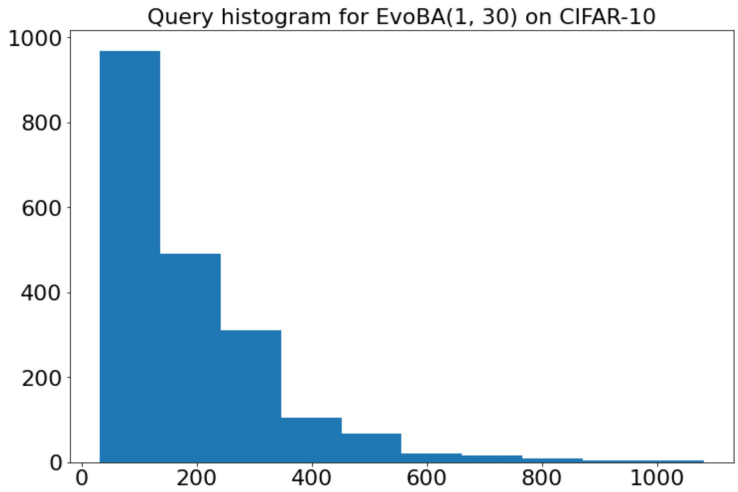


**Fig. 2.** Histogram of query counts obtained by **EvoBA(1,30)** on CIFAR-10 with target model ResNet-50. The distribution is very heavy on small values, with few outliers.

turbed images in terms of $L_2$-efficiency, it achieves an $L_2$ of $3.07$, which is higher, but closer to **SimBA-LCB**'s $L_2$ result of $2.37$.

**CompleteRandom** achieves a success rate of $60.5\%$, far below **EvoBA**'s $100\%$, but surprisingly above **SimBA-LCB**'s $56\%$. However, the nature of **CompleteRandom**'s perturbations is to lie at high distances, achieving $L_0$ and $L_2$ distances that are significantly higher than the distances achieved by the other methods.

### 4.4 Results on CIFAR-10

We impose an $L_0$ perturbation limit of $100$, and a query threshold of $2000$ for both **EvoBA** and **CompleteRandom**. We randomly sample $2000$ images for **EvoBA** and $50$ images for **SimBA-LCB**. **EvoBA** and **SimBA-LCB** are run on the exact same target ResNet-50 model, while **SimBA-DCT** and **SimBA-CB** also run on a target ResNet-50 model. **AutoZOOM-BiLIN** targets an InceptionV3 model.

**SimBA-LCB** required an average of **26.15s** per CIFAR-10 sample, while **EvoBA** required **1.91s** per sample.

All the attacks achieved $100\%$ success rate in the CIFAR-10 experiments, with the sole exception of **CompleteRandom**, which only got $69.5\%$. **EvoBA(1,30)** has a better query average when compared to all the **SimBA** approaches, which targeted the same ResNet-50 architecture. While **EvoBA(1,30)** targeted a different architecture than **AutoZOOM-BiLIN**, we still remark how the latter is twice more query efficient. However, **EvoBA(1,30)** surprisingly achieves an $L_2$ metric which is better than the reported numbers of **SimBA-CB** and **SimBA-DCT**, which are $L_2$ adversarial attacks.

It is not as much of a surprise the fact that **EvoBA(1,30)** achieves a considerably better $L_0$ metric when compared to **SimBA-LCB** ($17.67$ vs $99.46$).

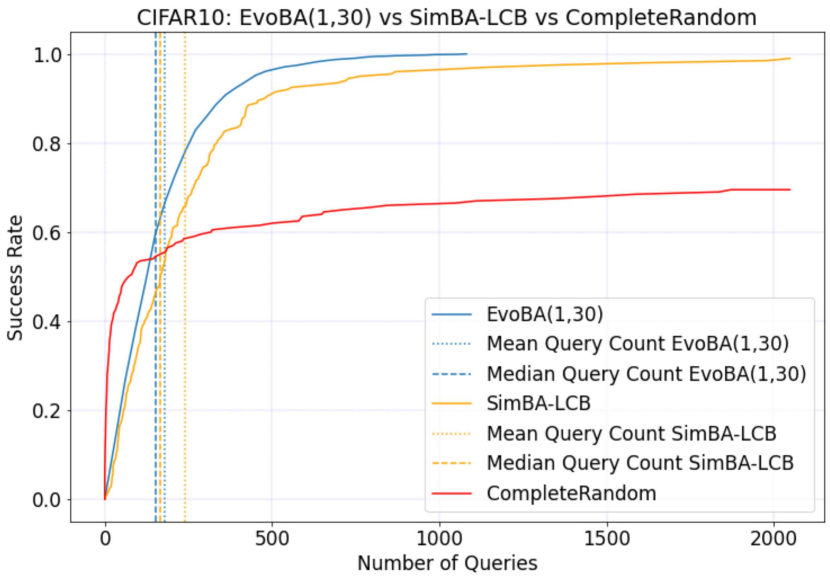


**Fig. 3.** The success rate (ratio of perturbed images) as a function of the maximum query budget. We compare **EvoBA** with the strong baseline **SimBA** and with **CompleteRandom** on CIFAR-10.

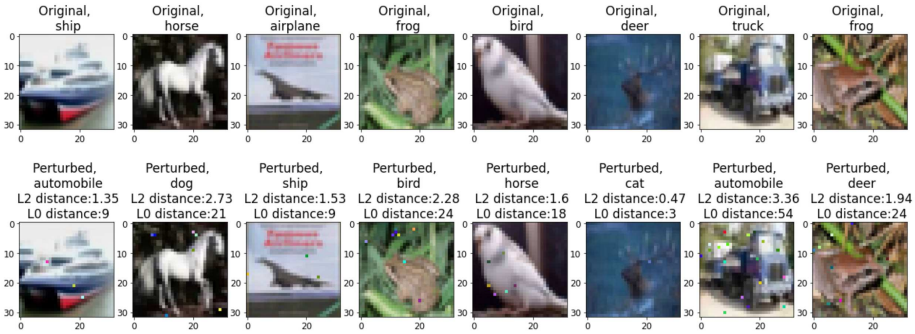


**Fig. 4.** The first row contains original CIFAR-10 samples, which are classified correctly by ResNet-50. The second row contains adversarial examples created by **EvoBA**, and are labelled with the corresponding ResNet-50 predictions. Furthermore, we also provide the $L_2$ and $L_0$ distances between the unperturbed and perturbed samples.

In Figures 1 and 2 we plot the histograms of the $L_0$ perturbation norms, respectively of the query counts obtained by **EvoBA(1,30)**. Both are highly skewed towards low values, showing how **EvoBA** does well in finding quick small perturbations with respect to the $L_0$ norm.

The success rate of **CompleteRandom** ($69.5\%$) and its low average query count ($161.2$) are surprisingly good results for the trivial nature of the method, outlining once again the lack of robustness in complex image classifiers. However, these come at the

| | SR | QA | L0 | L2 |
|---|---|---|---|---|
| **EvoBA(1,30)** | 100% | 178.56 | 17.67 | 1.82 |
| **SimBA-LCB** | 100% | 206.5 | 99.46 | 1.73 |
| **SimBA-CB** | 100% | 322 | - | 2.04 |
| **SimBA-DCT** | 100% | 353 | - | 2.21 |
| **AutoZOOM-BiLIN** | 100% | 85.6 | - | 1.99 |
| **CompleteRandom** | 69.5% | 161.2 | 97.17 | 3.89 |

**Table 3.** CIFAR-10 results. **AutoZOOM**, **SimBA-CB**, and **SimBA-DCT** do not report the $L_0$ metrics. However, we have discussed already why it is very likely that **AutoZOOM** perturbs most of the pixels.

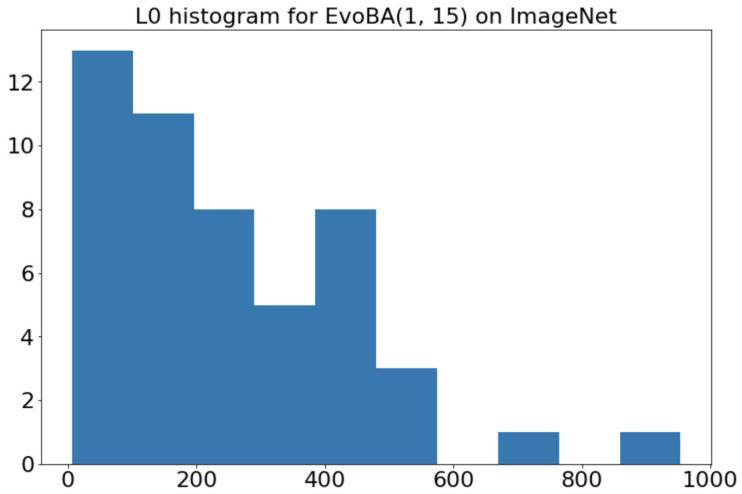


**Fig. 5.** Histogram of $L_0$ perturbation norms obtained by **EvoBA(1,15)** on ImageNet with target model ResNet-50. The distribution is very heavy on small values, with few outliers.

cost of an average $L_0$ that is roughly $5.5$ times higher and of an average $L_2$ that is roughly $1.4$ times higher in comparison with **EvoBA(1,30)**'s average results.

In Figure 3 we compare **EvoBA(1,30)** with **SimBA-LCB**, while also providing the **CompleteRandom** results. We plot the success rate as a function of the number of queries in order to understand how each method behaves for different query budgets. **EvoBA(1,30)** has a better success rate than **SimBA-LCB** for any query budget up to $2000$. For very low query budgets (under $112$ queries), **CompleteRandom** has a better success rate than **EvoBA(1,30)**, but it starts converging fast after their intersection point to the success rate of $69.5\%$. It is natural for the **CompleteRandom** strategy to find quick perturbations for the least robust images, as it performs bulk perturbations of many pixels at once, while **EvoBA** does all perturbations sequentially. This illustrates the natural trade-off between exploration and exploitation that any black-box optimization problem encounters.

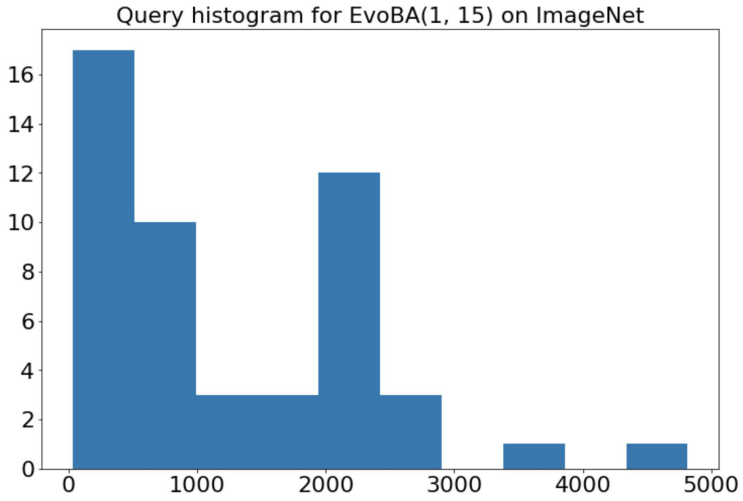


**Fig. 6.** Histogram of query counts obtained by **EvoBA(1,15)** on ImageNet with target model ResNet-50. The distribution is very heavy on small values, with few outliers.

### 4.5 Results on ImageNet

We adopt a similar framework to **AutoZOOM**: we randomly sample 50 correctly classified images and run **EvoBA** on top of them. For **EvoBA**, similarly to **SimBA**, we use a ResNet50 model, while **AutoZOOM** uses an InceptionV3. We impose an $L_0$ perturbation limit of $1000$, and a query threshold of $10000$. The $L_0$ limit we impose is reasonable, as each time we perturb a pixel we actually modify all of its three channels, therefore the $1000$ limit stands for approximately $0.66\%$ of the image pixels.

| | SR | QA | L0 | L2 |
|---|---|---|---|---|
| **EvoBA(1,15)** | 100% | 1242.4 | 247.3 | 6.09 |
| **EvoBA(1,20)** | 100% | 1412.51 | 211.03 | 5.72 |
| **SimBA-CB** | 98.6% | 1665 | - | 3.98 |
| **SimBA-DCT** | 97.8% | 1283 | - | 3.06 |
| **AutoZOOM-BiLIN** | 100% | 1695.27 | - | 6.06 |

**Table 4.** ImageNet results. **AutoZOOM**, **SimBA-CB**, and **SimBA-DCT** do not report the $L_0$ metrics. However, we have discussed already why it is very likely that **AutoZOOM** perturbs most of the pixels.

The median number of queries for **EvoBA(1,15)** is surprisingly low: $728.5$. Its median $L_0$ is $200$, and its median $L_2$ is $5.69$. This shows once more how we met our goal to minimise the query count and $L_0$ perturbation, as the medians are significantly smaller than the average values, showing how the distributions are biased towards low values.

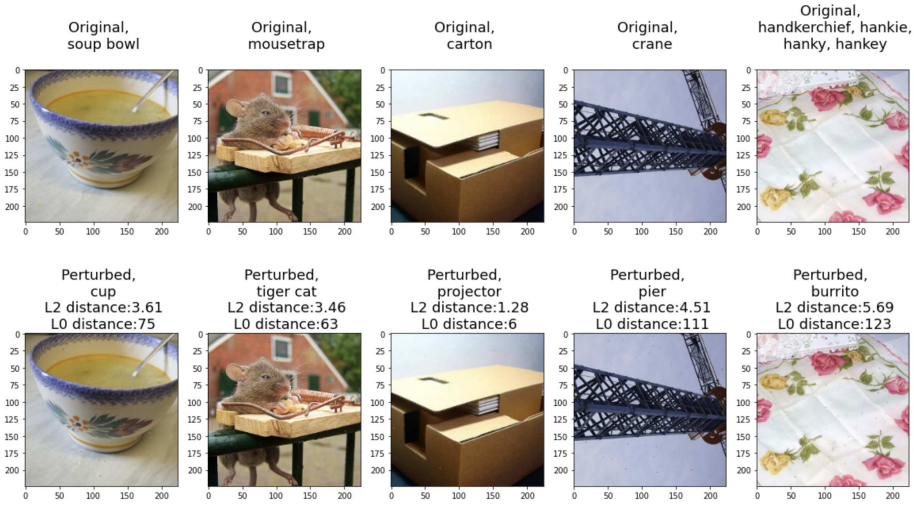


**Fig. 7.** The first row contains original ImageNet samples, which are classified correctly by ResNet-50. The second row contains adversarial examples created by **EvoBA**, and are labelled with the corresponding ResNet-50 predictions. Furthermore, we also provide the $L_2$ and $L_0$ distances between the unperturbed and perturbed samples.

In comparison, the $L_2$ mean and median are close, indicating that the good $L_2$ results we get are a consequence of our other constraints rather than an actual objective.

**EvoBA(1,15)** achieves the best average query metric among the given experiments. Surprisingly, its $L_2$ is almost equal to the one of **AutoZOOM-BiLIN**. **EvoBA(1,15)** also achieves a $100\%$ success rate, which is in line with **AutoZOOM-BiLIN**, and better than **SimBA**'s results.

### 4.6 Qualitative results

In Figures 4 and 7, we provide samples of CIFAR-10, respectively of ImageNet perturbed images, together with their initial and perturbed labels. The perturbations are almost imperceptible to the human eye, and they look mostly like regular noise. Considering the highly-biased nature of the $L_0$ histogram in Figures 1 and 5 towards small values, it is natural to expect well-crafted perturbations.

## 5 Using the tool

At https://github.com/andreiilie1/BBAttacks we provide a script **run_robustness_checks.py** that runs **EvoBA** against custom classifiers and datasets.

The script only requires custom implementations for loading the data to be adversarially perturbed. We provide an example of the implementation for the CIFAR-100 task. This is required to allow custom data loaders: unloading methods provided by packages, local data dumps, external data sources (e.g Amazon's S3), etc. In a future

implementation, we will provide options to run the attacks against all of these without any custom implementation on the user side.

The script can be run as follows:

```
python run_robustness_checks.py
--model_path "models/cifar100vgg/cifar100vgg.py"
--model_class_name "cifar100vgg"
--task "cifar100"
--sample_size 100
```

The $model\_path$ argument points to the Python inference logic. $Model\_class\_name$ specifies the class in the inference logic that has a black-box predict method. We provide multiple examples of usage in our repository.

The tool will print attack statistics like:

```
EvoBA STATS (L0 attack)
____________________
Perturbed successfully 100/100 images
Average query count: 59.16
Average l0 distance: 72.83
Average l2 distance per pixel: 0.0010281872578653908

Median query count: 43.0
Median l0 dist: 54.0

Max query count: 214
Max l0 dist: 239
____________________
```

The tool will also save comprehensive attack logs, including $json$ and $npy$ objects containing the indices of successfully perturbed images, the query counts used for each, and the $L_0$ and $L_2$ distances. It will furthermore generate histograms of the $L_0$ distances and query counts required to successfully perturb images, such as the ones introduced in our paper (Figures 5, 1, etc). The tool provides flags to enable or disable the $L_0$ and $L_2$ attacks used for providing fast robustness benchmarks.

A sample collection of output artifacts will look like below:

```
evoba_l0_hist.png
evoba_l0_stats.json
simba_l2_hist.png
simba_l2_stats.json
simba_stats.json
evoba_l0_queries_hist.png
evoba_l0_stats.npy
simba_l2_queries_hist.png
simba_l2_stats.npy
```

We furthermore provide custom utils for logging all results, including original and perturbed images, to MLflow (https://mlflow.org/). Using an MLflow server to then visualise the results make it extremely convenient and easy to document results and has been of tremendous help for us. This is only provided as custom scripts in our utils and examples of usage are documented in the provided notebooks. We will add the MLflow logging options to the tool in a future release.

To illustrate the utility of using MLflow as integrated part of these experiments, we show the UI of some of our experiments in Appendix C.

## 6 Conclusion

We proposed **EvoBA**, **EvoBA-E**, and **EvoBA-ET**, three black-box adversarial attacks based on evolution strategies which can serve as powerful baselines for black-box adversarial attacks.

While all provide competitive query efficiency and small perturbation distances, we focus on **EvoBA**, as it also provides significantly faster running times.

**EvoBA** achieves results in line with state-of-the-art approaches, such as **AutoZOOM**, but is far less complex. Simple yet efficient methods, such as **EvoBA**, **SimBA**, **DeepSearch**, and even **CompleteRandom** shed a light on the research potential of the black-box adversarial field, outlining the inherent security issues in many machine learning applications.

We provide a robustness tool built around **EvoBA**, which also allows running **SimBA**, at our repository. This allows for a quick robustness assessment of one's computer vision system. As the running times are surprisingly small and **EvoBA** is very effective, we propose that computer vision developers run our tool as part of their continuous development of models, iterating on improving not only the accuracy but also the robustness of their solutions.

**EvoBA** could be easily extended for classification tasks in other fields, such as natural language processing. One would have to come up with the right perturbation scheme in our approach, but we leave this for future research.

**Acknowledgement** This work was partially supported by the Romanian Ministry of Research and Innovation UEFISCDI 401PED/2020 and PN-III-P2-2.1-PTE-2019-0820, and by the European Regional Development Fund, Competitiveness Operational Program 2014-2020 through project IDBC (code SMIS 2014+: 121512).

**Authorship contribution statement** The first author, Andrei Ilie, has written most material and conducted the experiments in this paper under the supervision of the second and third authors, Marius Popescu and Alin Stefanescu. All three authors have contributed significantly to the research introduced in this paper. All three authors approved the final form and took active part in the drafting and reviewing process. No other people were involved in writing this paper or conducting the research behind it.

**Data Availability Statement** The datasets generated during and/or analysed during the current study are well-known, standard datasets: CIFAR-10 and CIFAR-100 ([17], available in standard TensorFlow datasets and at https://www.cs.toronto.edu/~kriz/cifar.html), ImageNet ([6], available at https://www.image-net.org/), and MNIST ([18], available in standard TensorFlow datasets and at https://yann.lecun.com/exdb/mnist/). The weights and architectures of the models we used for benchmarking are available in either the Tensorflow pretrained models collection, or at our repository: https://github.com/andreiilie1/BBAttacks/tree/main/models.

**Conflict of Interest Statement** The authors have no relevant financial or non-financial interests to disclose.

The authors have no conflicts of interest to declare that are relevant to the content of this article.

All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

The authors have no financial or proprietary interests in any material discussed in this article.

## References


1. Alzantot, M., Sharma, Y., Chakraborty, S., Zhang, H., Hsieh, C.J., Srivastava, M.B.: GenAttack: Practical black-box attacks with gradient-free optimization. In: Proc. of the Genetic and Evolutionary Comp. Conf. (GECCO'18). pp. 1111–1119 (2019)
2. Athalye, A., Carlini, N., Wagner, D.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In: Proceedings of Int. Conf. on Machine Learning (ICML'18). pp. 274–283 (2018)
3. Bafna, M., Murtagh, J., Vyas, N.: Thwarting adversarial examples: An $l\_0$-robust sparse fourier transform. Advances in Neural Information Processing Systems **31**, 10075–10085 (2018)
4. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: IEEE Symp. on Security and Privacy (SP'17). pp. 39–57. IEEE (2017)
5. Chen, P.Y., Zhang, H., Sharma, Y., Yi, J., Hsieh, C.J.: Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In: Proc. of the 10th ACM Workshop on Artificial Intelligence and Security. pp. 15–26 (2017)
6. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: ImageNet: A large-scale hierarchical image database. In: CVPR'09. pp. 248–255 (2009)
7. Dvijotham, K., Stanforth, R., Gowal, S., Mann, T.A., Kohli, P.: A dual approach to scalable verification of deep networks. In: UAI. vol. 1, p. 3 (2018)
8. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., Song, D.: Robust physical-world attacks on deep learning visual classification. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 1625–1634 (2018)
9. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (2015)
10. Gopinath, D., Katz, G., Păsăreanu, C.S., Barrett, C.: Deepsafe: A data-driven approach for assessing robustness of neural networks. In: International symposium on automated technology for verification and analysis. pp. 3–19. Springer (2018)
11. Guo, C., Gardner, J.R., You, Y., Wilson, A.G., Weinberger, K.: Simple black-box adversarial attacks. In: Proceedings of Int. Conf. on Machine Learning. pp. 2484–2493 (2019)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
13. Huang, X., Kwiatkowska, M., Wang, S., Wu, M.: Safety verification of deep neural networks. In: International conference on computer aided verification. pp. 3–29. Springer (2017)
14. Ilie, A., Popescu, M., Stefanescu, A.: Robustness as inherent property of datapoints. AISafety Workshop, IJCAI (2020)
15. Ilyas, A., Engstrom, L., Athalye, A., Lin, J.: Black-box adversarial attacks with limited queries and information. In: Proceedings of Int. Conf. on Machine Learning (ICML'18). pp. 2142–2151 (2018)
16. Ilyas, A., Engstrom, L., Madry, A.: Prior convictions: Black-box adversarial attacks with bandits and priors. arXiv:1807.07978 (2018)
17. Krizhevsky, A.: Learning multiple layers of features from tiny images. Master's thesis, University of Toronto (2009)
18. LeCun, Y.: The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/ (1998)
19. LeCun, Y., et al.: LeNet-5, convolutional neural networks. http://yann.lecun.com/exdb/lenet
20. Meunier, L., Atif, J., Teytaud, O.: Yet another but more efficient black-box adversarial attack: tiling and evolution strategies. arXiv:1910.02244 (2019)
21. Moon, S., An, G., Song, H.O.: Parsimonious black-box adversarial attacks via efficient combinatorial optimization. In: International Conference on Machine Learning. pp. 4636–4645. PMLR (2019)

22. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv:1605.07277 (2016)
23. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: IEEE European Symp. on Security and Privacy (EuroS&P'16). pp. 372–387. IEEE (2016)
24. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should I trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. pp. 1135–1144 (2016)
25. Ruan, W., Huang, X., Kwiatkowska, M.: Reachability analysis of deep neural networks with provable guarantees. arXiv:1805.02242 (2018)
26. Sharif, M., Bauer, L., Reiter, M.K.: On the suitability of lp-norms for creating and preventing adversarial examples. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 1605–1613 (2018)
27. Sharif, M., Bhagavatula, S., Bauer, L., Reiter, M.: Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In: Proc. of ACM SIGSAC Conf. on Computer and Comm. Security (CCS'16). pp. 1528–1540 (2016)
28. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Proc. of Int. Conf.e on Learning Representations (ICLR'15) (2015)
29. Su, D., Zhang, H., Chen, H., Yi, J., Chen, P.Y., Gao, Y.: Is robustness the cost of accuracy? - a comprehensive study on the robustness of 18 deep image classification models. In: Proc. of the European Conf. on Computer Vision (ECCV'18). pp. 631–648 (2018)
30. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. pp. 2818–2826 (2016)
31. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv:1312.6199 (2013)
32. Tu, C.C., Ting, P., Chen, P.Y., Liu, S., Zhang, H., Yi, J., Hsieh, C.J., Cheng, S.M.: AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In: Proc. of the AAAI Conf. vol. 33, pp. 742–749 (2019)
33. Wiyatno, R.R., Xu, A., Dia, O., de Berker, A.: Adversarial examples in modern machine learning: A review. arXiv:1911.05268 (2019)
34. Zhang, F., Chowdhury, S.P., Christakis, M.: Deepsearch: a simple and effective blackbox attack for deep neural networks. In: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 800–812 (2020)

## A Consistency of the results

As our attack is stochastic and relies heavily on a randomness generator, we did 10 independent runs of **EvoBA** on a sample of 200 CIFAR-10 images.

The **EvoBA** results from Table 5 show how robust our method is. The query standard deviation is only $4.87$ (roughly $3\%$ of the query average $178.29$). Similarly, the standard deviation is very small for **EvoBA** across all metrics.

The time required to perturb one CIFAR-10 sample was $1.91s$ on average, with a standard deviation of $0.1s$.

| **EvoBA** | Run 1 | Run 2 | Run 3 | Run 4 | Run 5 | Run 6 | Run 7 | Run 8 | Run 9 | Run 10 | **Mean** | **Std** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SR** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 0 |
| **L0** | 17.37 | 18.29 | 17.94 | 17.31 | 17.27 | 17.33 | 17.76 | 17.77 | 18.43 | 16.89 | 17.63 | 0.489 |
| **L2** | 1.81 | 1.87 | 1.86 | 1.83 | 1.83 | 1.82 | 1.81 | 1.84 | 1.87 | 1.79 | 1.83 | 0.027 |
| **Q** | 175.6 | 184.9 | 181.3 | 174.7 | 174.55 | 175.45 | 179.8 | 184.5 | 186.1 | 170.95 | 178.285 | 4.871 |

**Table 5.** Ten independent random runs of **EvoBA**.

## B More baselines

We show how **EvoBA** compares with other well-known $L_2$ black-box adversarial attacks which work under the query-limited scenario. For this, we reuse the results from [11], where the same model (ResNet-50) is targeted, and extend the metrics we presented in Table 4. The methods are **QL-attack**, introduced in [15], and **Bandits-TD**, introduced in [16].

| | SR | QA | L0 | L2 |
|---|---|---|---|---|
| **EvoBA(1,15)** | 100% | 1242.4 | 247.3 | 6.09 |
| **EvoBA(1,20)** | 100% | 1412.51 | 211.03 | 5.72 |
| **SimBA-CB** | 98.6% | 1665 | - | 3.98 |
| **SimBA-DCT** | 97.8% | 1283 | - | 3.06 |
| **QL-attack** | 85.4% | 28174 | - | 8.27 |
| **Bandits-TD** | 80.5% | 5251 | - | 5.00 |
| **AutoZOOM-BiLIN** | 100% | 1695.27 | - | 6.06 |

**Table 6.** ImageNet results which show how **EvoBA** is comparable with $L_2$ attacks from the $L_2$ distance point of view, while being an $L_0$ attack.

## C MLflow

We show the UI of a subset of our experiments logged in MLflow in Figure 8 and 9.


**Fig. 8.** MLflow home page view of some of our experiments. It allows choosing between historic Experiments (CIFAR-100 and ImageNet). Within each experiment, we can scroll through individual attack runs and observe the metrics and parameters of each.

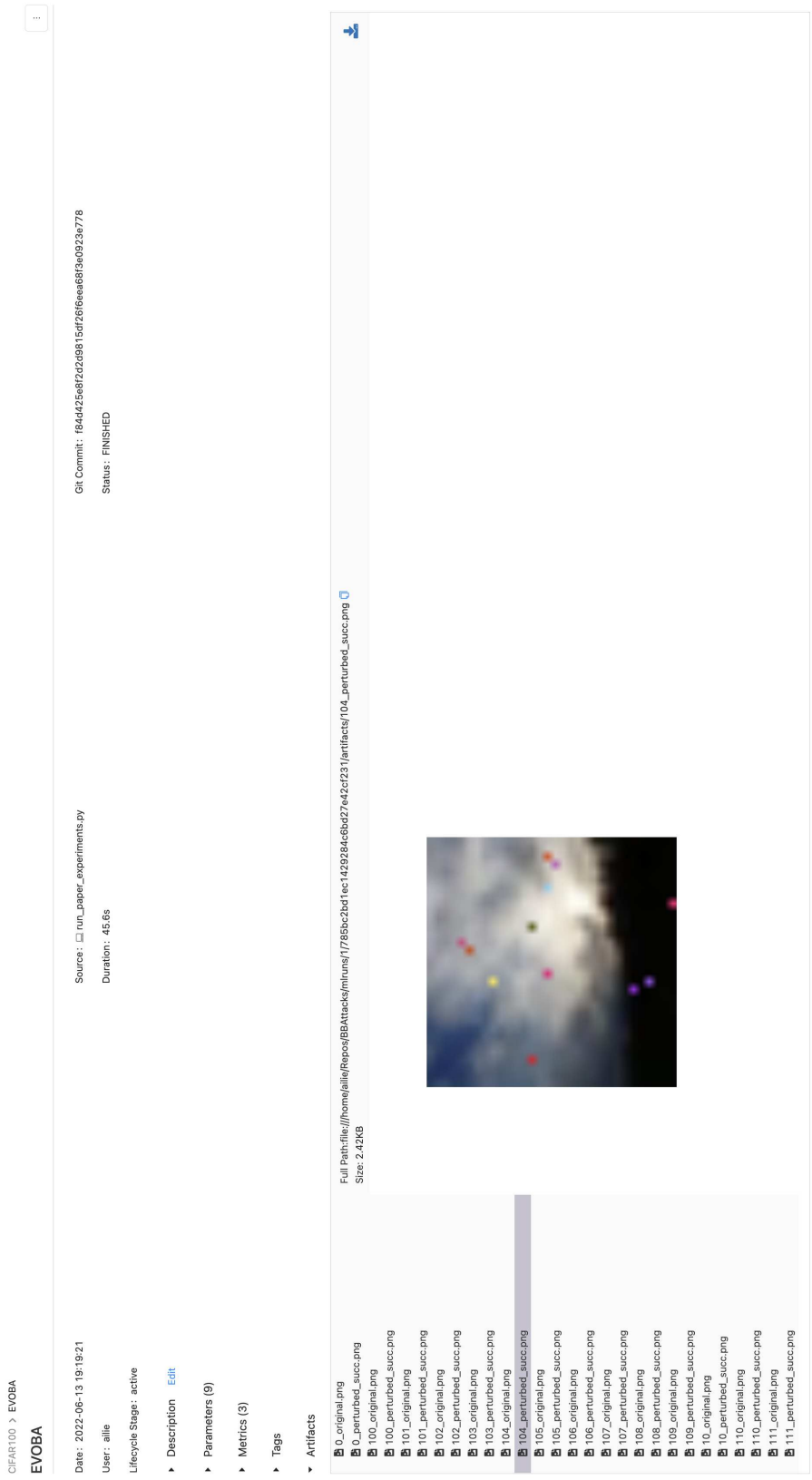


**Fig. 9.** MLflow view of one of our CIFAR-100 **EvoBA** experiments. Parameters used and resulting metrics are available, together with original and perturbed images in an interactive view.