Appendix A. Notation

Table 4. Notations Table.

Variable	Description
$\mathbf{x}_{\text{orig}}$	Original, unperturbed image
$\mathbf{x}$	Adversarial example
δ	Perturbation
$\|\mathbf{x} - \mathbf{x}_{\text{orig}}\|_2$	Euclidean distance
f_θ	Neural network
y	Label
$F(x)$	Fitness function
$l(\mathbf{x})$	Adversarial loss term
t_{adv}	Threshold for Initial Search Strategy
q	Current query count
Q	Number of queries
g	Current generation
n	Perturbation dimension
λ	Population size
$\mathbf{m}^{(g)}$	Mean at generation g
$\sigma^{(g)}$	Step-size at generation g
$\mathbf{C}^{(g)}$	Covariance matrix at generation g
$\mathbf{x}_i^{(g+1)}$	Sampled individual i at generation $g+1$
μ	Parent population size
μ_{eff}	Effective population size
c_σ	Damping factor for step-size adaptation
d_σ	Damping factor for step-size update
c_{m}	Learning rate for the mean
e_σ	Step-size enlarger of Initial Search Strategy
w_i	Weight of the i-th individual
$\mathbf{1}$	Indicator function
F	Fitness value
l	Penalty function
$\mathbb{R}^n$	n-dimensional real space
$\mathcal{N}$	Gaussian distribution
$\mathbf{x}_{1:\lambda}^{(g+1)}$	Best instance of the population at $g+1$
$\mathbf{x}_{i:\lambda}^{(g+1)}$	i-th best instance of the population at $g+1$

Appendix B. Automated Algorithm Configuration

The implementation of CMA-ES we use has 16 parameters, including the initial and maximum step-size, the population size or the learning rate for updating the mean.

And due to our adaptions, we have two more parameters, namely the step-size enlarger of the ISS e_σ and, for ImageNet pictures, the scaling factor, *i.e.*, the size of the subspace in which adversarial examples are searched, making the task of finding a good configuration difficult. To avoid tedious manual search for good parameters and to make our experiments easier to replicate, we use AAC for setting the parameters. We use the multi-fidelity facade of SMAC3 [22], which uses Hyperband (HB) [19] to find how many configurations should be evaluated with given budgets. This method is based on the assumption that the performance of an algorithm with a low budget approximates the performance when run to convergence (which is typically computationally expensive). In other words, it evaluates different configurations for the algorithm on different budgets to find the best-performing parameters with limited resources.

Table 5. Search space and tuned values for DACES-NNI.

Hyperparameter	Symbol	Type	Lower bound	Upper bound	Default	Result
stdev_init	σ_{init}	Float	0.05	0.1	0.075	0.085
stdev_max	σ_{max}	Float	2.5	3.5	3.0	2.61
e_sigma	e_σ	Float	0.1	0.5	0.3	0.40
c_m	c_m	Float	0.25	0.75	0.5	0.57
c_sigma	c_σ	Float	0.005	0.01	0.0075	0.0083
scaling_factor	s_f	Integer	3	19	15	14
popsize	λ	Integer	20	100	60	23
c_sigma_ratio	$c_{\sigma,\text{ratio}}$	Float	1.5	2.0	1.75	1.88
damp_sigma	d_σ	Float	1.0	1.01	1.005	1.0083
damp_sigma_ratio	$d_{\sigma,\text{ratio}}$	Float	0.5	1.0	0.75	0.58
c_c	c_c	Float	0.01	0.1	0.05	0.066
c_c_ratio	$c_{c,\text{ratio}}$	Float	1.0	1.5	1.25	1.21
c_1	c_1	Float	0.0005	0.001	0.0007	0.00095
c_1_ratio	$c_{1,\text{ratio}}$	Float	1.25	2.5	1.75	2.35
c_mu	c_μ	Float	0.001	0.01	0.005	0.0042
c_mu_ratio	$c_{\mu,\text{ratio}}$	Float	1.5	3.0	2.25	2.68

Both the budget that is allocated to the configurations and the share of configurations that is retained in each round of elimination are controlled by the parameter η. For our CMA-ES-based attack, the budget is the number of queries we allow for the attack. Particularly, each round of elimination retains $1/\eta$ configurations and the budget B is determined by $B = \left(\lfloor \log_\eta(R) \rfloor + 1\right) R$, where R is the maximum budget [19]. The larger the value of η, the more configurations will be discarded, resulting in fewer rounds of elimination. It is usually set to 3, and larger values result in a more aggressive elimination schedule. We set $\eta = 7$ to reduce the computational burden of the AAC. We decided to use $\eta = 7$ because we found that configurations should be evaluated using a budget of at least 500 queries and by setting $\eta = 7$, the minimum amount of queries for each configuration is $\frac{30\,000}{7*7} = 612$, the medium budget becomes $\frac{30\,000}{7} = 4285$ and the maximum budget 30 000 queries. This yields three brackets in SMAC3, where the first bracket contains 49, 7 and 0 runs with the low, medium and high budget, respectively. Bracket 2 contains 11 and 1 runs with the medium and maximum budget, while the last run contains 3 runs with the maximum budget. We use 25 initial configurations. In contrast to plain HB, the multi-fidelity facade replaces the "random selection of configurations at the beginning of each HB iteration by a model-based search." [10] For further details, we refer to the original paper [10] and the references therein. The objective function we optimise with SMAC3 is the area under the curve of our adversarial attack when plotting the l_2-norm against the number of queries. An

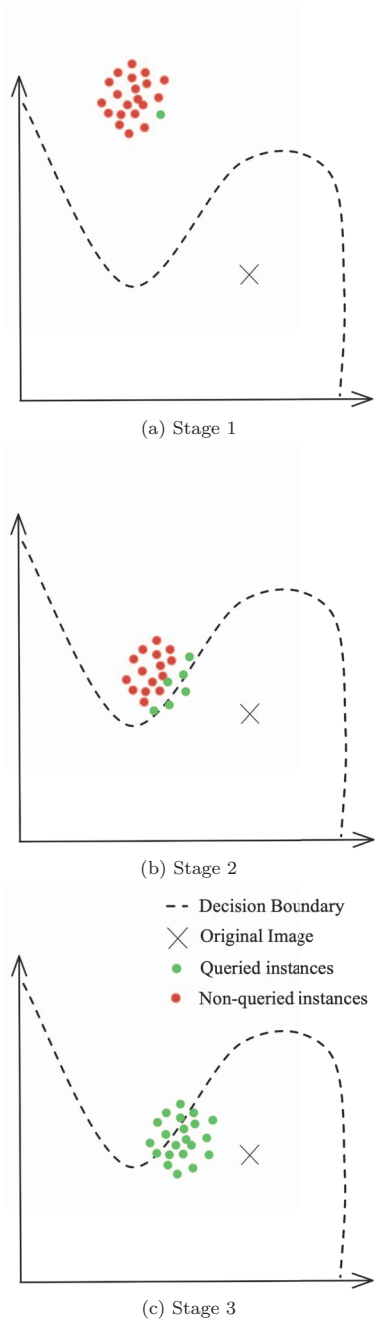
example of these plots can be seen in Figure 2. This, in contrast to optimising the final l_2-norm of the attack, incentivises a good performance across all query budgets.

However, optimising the configuration for decision-based adversarial attacks is costly concerning running time. This stems from the fact that one has to consider multiple seeds due to the stochastic nature of CMA-ES, multiple neural networks (four in this research), and a sufficient number of instances (200 in this work). Running the AAC naively with ten seeds would thus lead to $10 \times 4 \times 200 = 800$ runs per configuration. To make the AAC less costly, we use several approximations for the performance of one configuration over multiple seeds, networks and instances.

First, we associate each seed with a fraction of the set of instances. To be precise, each seed will be evaluated with 20 images only. In total, we use ten seeds, leading to a total number of used instances of 200. This way, we get a sufficient number of instances and seeds without exploding running time costs. The approximation is valid since every network will compared to every other network on the same seed with the same instance. Second, we dump a configuration when the configuration fails to find an adversarial example on too many instances, as we aim to have a 100% success rate on all networks. The fact that an instance is failing is indicated by the AUC. We dump a configuration when its AUC is on average above $600 * 1000$, where 600 is the minimum number of queries a configuration is evaluated on and 1000 is the punishment for not being adversarial. This value is multiplied by a factor of 5 for Subspace Activation (SA) and grid downsizing, as these will not always find adversarial examples.

Appendix C. Query Strategy

Figure 3 illustrates the query strategy with its three stages. In stage 1, when no single instance has ever crossed the decision boundary, only one instance is queried. After that, in stage 2, the effective population size μ_{eff} is queried, which covers around 30% of the instances. If more than 75% of μ_{eff} have crossed the decision boundary, the entire population is queried (stage 3).



(a) Stage 1

(b) Stage 2

(c) Stage 3

Figure 3. Illustration of the query strategy.

Appendix D. Other Downsizing techniques

While BI, NNI, SA and grid downsizing have in common that they search for adversarial perturbations in a lower dimensional space, they are very different in terms of the number of pixels they attack. BI and NNI search for perturbations in a low-dimensional subspace, but project them back to the original image size. On the other hand, the SA technique finds a sensitive region in the form of a rectangle in the image which is

perturbed and added back to the original image. Similarly, grid downsizing selects a grid of pixels modifies them, and applies the perturbation to the selected pixels. In other words: Instead of attacking all pixels, they select a number of pixels which they attack. Thus, SA and grid downsizing are much more constrained and therefore less powerful. This difference is visualised in Figure 4. To get an idea how well our algorithm works in this even more restricted setting, we still analysed SA and grid downsizing as supplementary material. We start by introducing the techniques themselves and then show additional results for them.

D.1. Subspace Activations

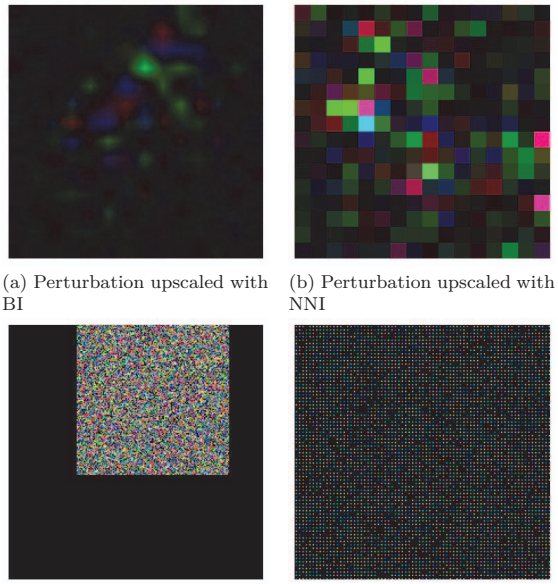
For their CMA-ES-based attack, Li et al. [21] proposed a so-called SA technique. Essentially, they use the red channel of a smoothed image and binarise its values. Their method then finds the largest patch of 1s in this binarised red channel, while enforcing a minimum patch size of 8×8. While this method often captures the object of interest, it is obvious that it will regularly fail to do so, namely when the object of interest has low values in its red channel. An example of the resulting perturbation is shown in Figure 4(c) in Appendix D. As the authors do not provide any code for this technique, we implemented it with the information available in the paper. The authors stated that their approach reduces the dimensionality to less than 57%. We managed to nearly reproduce this result, as our version of this algorithm scales images down to 54% on average.

D.2. Grid Downsizing

Based on the named limitation of SA, and the fact that neighbouring pixels are likely to be of similar importance to the prediction [32], we propose a simple baseline against which SA can be compared. This downscaling technique involves dividing the image into grids of size $n \times n$. We then take the midpoints of these grids for our attack. For example, for $n = 3$, you would divide the image into patches of size 3×3, and by using only the centre point, we effectively reduce the size of the problem by a factor of 9. This will effectively give less focus to the object as it will always cover the whole image, but it will also never miss the object of interest. A perturbation created using a grid is illustrated in Figure 4(d).

D.3. Results Using SA and Grid on ImageNet with ResNet-50

For this comparison, we refer to DACES-BI, DACES-NNI, DACES-SA and DACES-Grid to stress which dimensionality reduction technique was used. While



(a) Perturbation upscaled with BI

(b) Perturbation upscaled with NNI

(c) Perturbation of active subspace

(d) Perturbation with 3×3-grid

Figure 4. Visualisation of the three downscaling techniques. All perturbations were amplified by a factor of 100 to make them visible.

DACES-BI and -NNI find adversarial examples 100% of the time, the SA and Grid variants fail in 12.5% and 4% of the cases, respectively. This finding highlights the difficulty of finding adversarial examples in the decision-based setting as opposed to the score-based setting where one has access to the output probabilities. In their study of score-based attacks with CMA-ES [21], Li et al. were able to achieve a success rate of over 95% on ImageNet with SAs. DACES-Grid, which was intended as a baseline for DACES-SA, performs also poorly.

Not only the success rates suffer from restricting the number of pixels that can be attacked, but also the l_2-norms. On average, DACES-Grid, -SA, -BI and -NNI achieved a l_2-norm after 30 000 queries of 12.8, 28.8, 1.2 and 1.0, which also results in visually perceptible perturbations. This leaves much room for improvement for future work on decision-based attacks with restricted number of attackable pixels.

D.4. Details for NNI

Here, we detail the nearest-neighbour interpolation. Formally, given a $d \times d$ image $\mathbf{x}_{\text{orig}}$ and a scaling factor

Table 6. The left side displays the number of images each attack performed best at per network (after 30 000 queries). The value shown in boldface indicates the SBA. (1+1) denotes the (1+1)-CMA-ES. The rightmost three columns displays the l_2-norms achieved by the VBA, the VBA without our method DACES (VBA-D) and the SBA.

		DACES	HSJA	CGBA	TA	(1+1)	VBA	VBA-D	SBA
ImageNet	ResNet-50	**106**	42	2	0	50	1.39	1.75	1.92
	ResNet-101	**116**	40	3	1	40	1.31	1.72	1.89
	VGG-16	3	15	**149**	0	33	0.90	0.94	1.03
	ViT	19	5	**154**	0	22	1.29	1.40	1.48
	Robust RN-50	**86**	3	83	0	28	1.94	2.78	2.51
	Robust RN-101	**147**	0	0	0	53	1.92	5.81	2.19
	Robust VGG-16	23	6	**139**	0	32	1.31	1.43	1.56
	Total	500	111	530	1	258	1.44	2.26	6.13
CIFAR	ResNet-34	13	**168**	0	0	19	0.12	0.13	0.13
	ResNet-50	16	**168**	1	0	15	0.12	0.12	0.12
	VGG-16	27	**163**	0	0	10	0.14	0.15	0.15
	ViT	3	3	**194**	0	0	0.78	0.79	0.80
	Total	59	502	195	0	44	0.29	0.3	0.53

$s \geq 1$, the upscaled image $\mathbf{x}'_{\text{orig}}$ is [26]:

$$\mathbf{x}'_{\text{orig}}(j \cdot s + k, i \cdot s + l) = \mathbf{x}_{\text{orig}}(j, i) \quad \forall i, j \in \{1, \ldots, d\} \text{ and } k, l \in \{0, \ldots, s-1\} \quad (6)$$

Appendix E. Marginal Contribution of DACES

To investigate marginal contributions, we identified the VBA at the maximum query budget for each network. The VBA is a hypothetical attack that optimally combines a set of attacks by picking the best one for each image; in other words, it is the minimum perturbation achieved by all attacks on an image. In addition, we computed the minimum perturbation of a VBA without using DACES (VBA-D). The difference between these values indicates the marginal contribution of DACES to the state of the art. Furthermore, we identified the Single Best Attack (SBA) for each network. The SBA is the attack that achieves the best median l_2-norm for most instances on a network (or a group of networks). These terms were coined with reference to the terms "Single Best Solver" and "Virtual Best Solver" from the algorithm selection literature [14].

The results of our analysis of the VBA and SBA are shown in Table 6. In addition to the number of images that were solved best by each attack, one can see on the right side the l_2-norm of the VBA, the VBA without using DACES (VBA-D) and the SBA. As explained previously, the difference between the VBA and VBA-D indicates the marginal contribution of DACES. For example, a VBA constructed for ResNet-50 on ImageNet without using DACES merely achieves a l_2-norm of 1.75, while by adding DACES to the portfolio, this value improves by 0.36. This shows that DACES provides significant value beyond existing attacks. For this particular case, using DACES alone (the SBA) would give better performance than using the portfolio of all other algorithms, yielding a perturbation of 1.67 compared to 1.75.

We can see that DACES is the SBA for ResNet-50, ResNet-101 and Robust RN-50 with 116, 127 and 108 instances, respectively, while CGBA is the SBA for VGG-16 and ViT, with 147 and 150 images, respectively. HSJA was observed to perform much better on the smaller CIFAR-100 dataset, dominating across most networks, while CGBA performed best on the ViT on CIFAR-100. This is most likely due to the fact that the ViT uses an upscaled version of CIFAR-100 with the dimensions of $3 \times 224 \times 224$. Still, DACES contributes to the state of the art on CIFAR-100 by solving 45 instances best.

Appendix F. Further Results on ImageNet

Here, we show additional figures and tables on the experiments concerning ImageNet. Table 7 shows the median norm and area under the curve for different query budgets over 200 images from the ImageNet testing set. We also analysed the success rates attainable under varying norm constraints. Figure 5 presents these results for 5 000 and 30 000 queries. We observed that with low norm constraints, a significantly higher number of queries is required to achieve high success rates, whereas with higher norm constraints, the success rates achieved with 5 000 queries frequently approach those obtained with 30 000 queries.

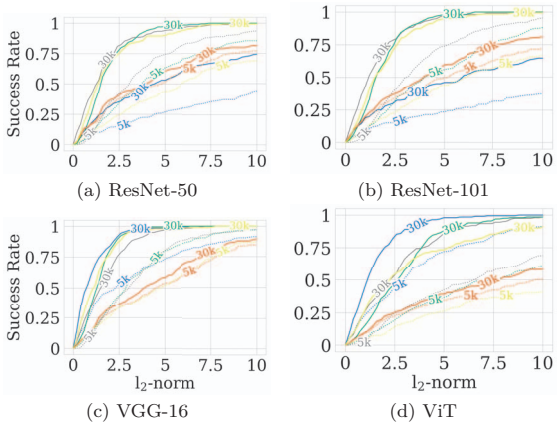
Finally, Table 8 details the average running time over the 200 instances per network.

Appendix G. Further Results on CIFAR-100

For the experiments on CIFAR-100, we use the same parameters as for ImageNet, except for the step-size that we set to a tenth of the value for ImageNet. This is because the CIFAR networks are expected to have lower robustness, which means the algorithm should also search for lower values. On CIFAR-100 the methods perform quite differently despite the similar networks. On ResNet-34, ResNet-50, and VGG-16 HSJA performs best across all query budgets. However, it should also be noted that it fails as the only attack on one instance on ViT. For low query budgets up to 7 000, CGBA and for higher query budgets (1+1)-CMA-ES comes second, followed by TA. DACES performs mediocre for low query budgets, but finishes second on every network. On Vit, CGBA comes first followed by DACES, HSJA and TA. The corresponding

Table 7. Median norm and AUC for different query budgets over 200 images from the ImageNet testing set (lower is better). The best value for each query budget and network is indicated in boldface.

Queries	Attack	ResNet-50	ResNet-101	VGG-16	ViT	R. RN-50	R. RN-101	R. VGG-16	Average
1k	DACES-BI	13.19	15.39	11.55	28.28	28.32	29.79	21.81	21.19
	DACES-NNI	16.06	16.43	12.23	32.58	22.72	22.97	17.58	20.08
	HSJA	15.96	18.98	11.49	32.74	28.28	31.01	15.19	21.95
	CGBA	19.80	24.40	8.25	**10.30**	25.23	35.35	14.38	19.67
	TA	**6.10**	**6.96**	**5.88**	14.52	**10.94**	**10.61**	**7.37**	**8.91**
	(1+1)-CMA-ES	34.19	37.92	26.94	68.88	47.55	46.43	31.98	41.98
10k	DACES-BI	1.80	1.98	2.23	3.39	4.17	4.04	2.95	2.94
	DACES-NNI	**1.63**	**1.62**	1.84	3.18	**3.09**	**2.90**	2.37	**2.38**
	HSJA	2.49	2.61	2.08	4.63	5.88	9.58	3.19	4.35
	CGBA	8.84	12.56	**1.33**	**1.85**	5.63	19.60	**1.87**	7.38
	TA	4.40	4.71	4.17	8.60	8.18	7.49	5.30	6.12
	(1+1)-CMA-ES	2.81	2.95	2.24	6.01	5.41	4.32	3.13	3.84
30k	DACES-BI	1.38	1.35	1.92	2.28	1.92	1.74	1.73	1.76
	DACES-NNI	**1.25**	**1.18**	1.45	2.25	**1.73**	**1.66**	1.54	**1.58**
	HSJA	1.48	1.56	1.29	2.63	3.42	6.38	1.95	2.67
	CGBA	4.36	5.86	**0.70**	**1.10**	2.29	11.15	**0.95**	3.77
	TA	3.96	3.94	3.85	7.78	7.03	6.87	4.77	5.46
	(1+1)-CMA-ES	1.36	1.48	1.07	2.30	2.49	2.13	1.47	1.76
AUC	DACES-BI	122 402	128 560	118 100	222 011	221 207	215 883	173 201	171 623
	DACES-NNI	**107 729**	**114 174**	90 308	214 425	**170 432**	**185 084**	142 120	**146 325**
	HSJA	122 970	129 996	85 002	218 035	244 118	363 201	139 026	186 050
	CGBA	377 780	435 364	**78 872**	**103 177**	286 838	714 723	**128 073**	303 547
	TA	191 133	203 052	161 763	380 992	305 901	334 938	222 726	257 215
	(1+1)-CMA-ES	213 668	217 533	161 167	379 982	331 156	292 567	190 376	255 207



(a) ResNet-50 (b) ResNet-101 (c) VGG-16 (d) ViT

Figure 5. Success rates on 200 ImageNet instances using 5 000 and 30 000 queries. The success rate is the % of instances that are below the l_2-threshold on the x-axis given the query budget. For better readability, we focus on the better-performing DACES-NNI and neglect DACES-BI.

values can be found in Table 10, the figure showing the success rate in Figure 8.

Table 8. Average running time of each attack in seconds per network on ImageNet.

	DACES	CGBA	TA	(1+1)-CMA-ES
ResNet-50	**17.1**	50.2	114.6	276.0
ResNet-101	**25.0**	64.7	170.7	360.2
VGG-16	**17.2**	44.3	71.9	213.6
ViT	**39.4**	68.5	150.1	296.8
Robust ResNet-50	**22.0**	96.8	115.2	418.5
Robust ResNet-101	**31.3**	112.9	174.7	326.7
Robust VGG-16	**24.1**	85.7	81.3	209.8

Table 9. Average running time of each attack per network in seconds on CIFAR-100.

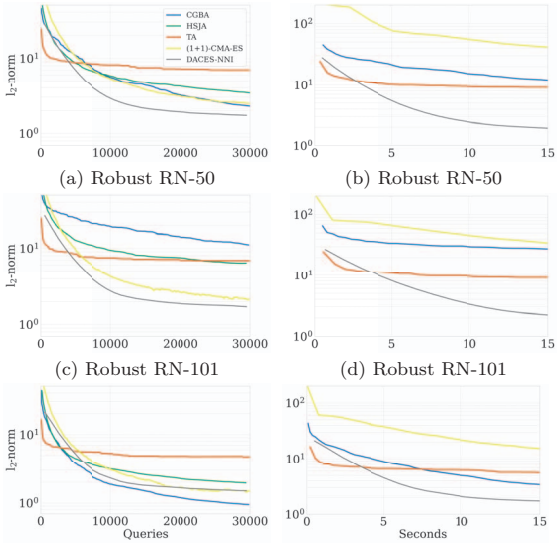
	DACES	CGBA	TA	(1+1)-CMA-ES
ResNet-34	18.2	**8.7**	96.2	82.7
ResNet-50	22.8	**12.8**	116.5	105.9
VGG-16	13.3	**4.8**	68.6	46.4
ViT	**40.0**	69.0	149.2	441.6

Appendix H. Analysis of the Strengths and Weaknesses of DACES

In Table 7, it can be seen that DACES-NNI has the best average performance over all networks as indicated by the lowest average AUC. One also sees that DACES performs consistently well on different networks, which

Table 10. Median norm and AUC for different query budgets on CIFAR-100.

Queries	Attack	ResNet-34	ResNet-50	VGG-16	ViT	Average
1k	DACES	2.28	2.09	2.12	15.15	5.41
	HSJA	**0.44**	**0.36**	**0.45**	14.97	4.06
	CGBA	1.16	1.11	1.16	**4.84**	**2.07**
	TA	2.16	1.60	1.90	5.79	2.86
	(1+1)-CMA-ES	2.60	2.29	2.49	46.96	13.59
10k	DACES	0.54	0.45	0.47	1.70	0.79
	HSJA	**0.13**	**0.11**	**0.14**	2.63	0.75
	CGBA	0.64	0.63	0.60	**0.98**	**0.71**
	TA	2.06	1.59	1.86	3.88	2.35
	(1+1)-CMA-ES	0.18	0.17	0.19	5.69	1.56
30k	DACES	0.13	0.12	0.14	1.40	**0.45**
	HSJA	**0.11**	**0.09**	**0.12**	1.32	0.41
	CGBA	0.59	0.58	0.52	**0.66**	0.59
	TA	2.06	1.59	1.84	3.54	2.26
	(1+1)-CMA-ES	0.14	0.13	0.15	2.70	0.78
AUC	DACES	21 241	20 484	20 212	101 338	40 819
	HSJA	**5 535**	**5 470**	**6 338**	127 058	36 100
	CGBA	25 140	26 766	24 065	**52 114**	**32 021**
	TA	67 023	56 819	60 979	156 501	85 331
	(1+1)-CMA-ES	17 672	17 950	19 719	307 931	90 818

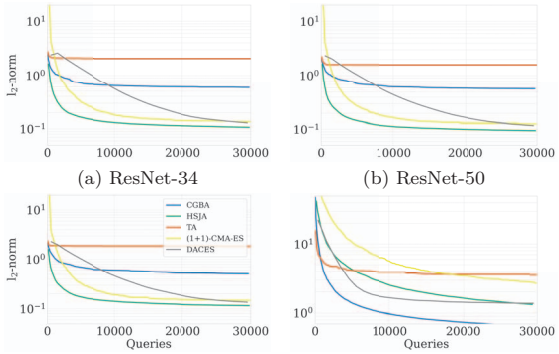


(a) Robust RN-50 (b) Robust RN-50

(c) Robust RN-101 (d) Robust RN-101

(e) VGG-16 (f) VGG-16

Figure 6. Median distance versus number of model queries and time over 200 images from the ImageNet testing set on three adversarially trained networks.

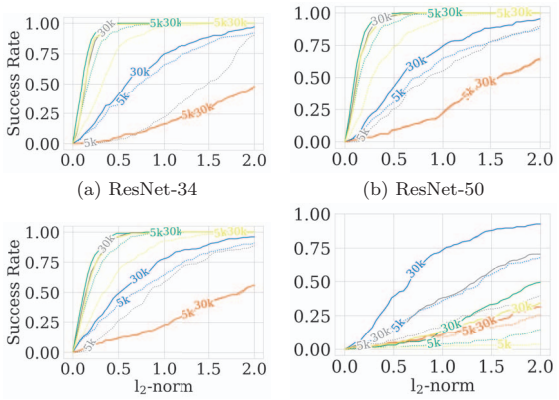


(a) ResNet-34 (b) ResNet-50

(c) VGG-16 (d) ViT

Figure 7. Median distance versus number of model queries over 200 images from the CIFAR-100 testing set (lower is better). The attacks used are listed in the legend of panel (c). We can see that both DACES attacks show a consistently good performance over the ResNets, the VGG-16 and ViT architectures.

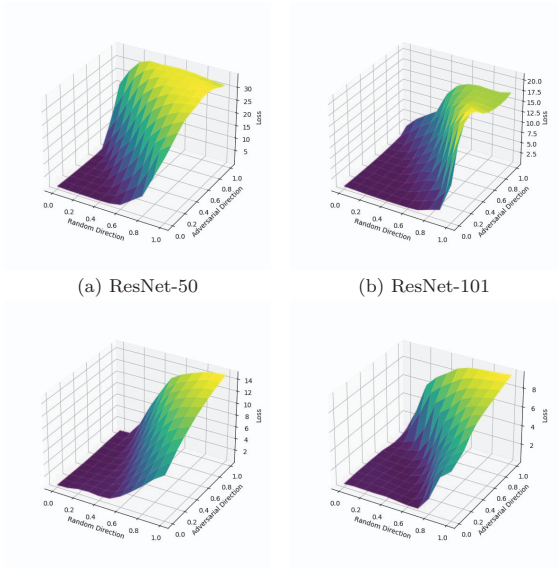
was one of the goals when designing this attack. Still, one can note performance complementarities between DACES and other attacks. In this section, we investigate possible reasons for this complementarity. To be precise, Table 7 shows that the lowest l_2-norm



(a) ResNet-34 (b) ResNet-50

(c) VGG-16 (d) ViT

Figure 8. Success rates on 200 CIFAR-100 instances using 5 000 and 30 000 queries. The success rate is the % of instances that are below the l_2-threshold on the x-axis given the query budget.



(a) ResNet-50 (b) ResNet-101

(c) VGG-16 (d) ViT

Figure 9. Example loss landscape for one image for the four networks.

found for ResNet-50, ResNet-101, VGG-16 and ViT after 30 000 queries are 1.25 (DACES), 1.18 (DACES), 0.70 (CGBA) and 1.10 (CGBA), while the mean l_2-norms for these networks over all attacks are 2.25, 2.55, 1.75 and 3.15, respectively. In particular, DACES-NNI dominates for the ResNet-50 and ResNet-101, while CGBA performs best for VGG-16 and ViT. However, it should be noted that while DACES-NNI performs third and second best on VGG-16 and ViT, CGBA performs worst (rank 6) on ResNet-50 and ResNet-101. Thus, it seems like that DACES-NNI is not bad on these networks, but that CGBA is particularly good on these, which has most likely to do with the specific curvature of these networks, which seems to be a decisive factor for the success of this attack [28]. Still, in this section, we try to analyse which factors lead to DACES performing better or worse. To investigate this, we first investigate the loss landscapes of the different networks and then conduct a statistical analysis.

H.1. Loss Landscapes

A first intuitive hypothesis for why we see a better performance on the ResNets is that the performance of the attacks is related to the neural network architecture. Residual Neural Networks have skip connections that are known to smooth the 'parameter loss landscape' of the neural network [17]. With 'parameter loss landscape', we refer to the landscapes concerning the model parameters that researchers are concerned with when training machine learning models. For our analysis, however, we are concerned with what we call the 'input loss landscape', *i.e.*, how the loss behaves when perturbing the input.

We investigate these input loss landscapes following Zhang et al. [38]: for 100 randomly chosen images from the ImageNet test set, we obtain the vector pointing towards the nearest found adversary d_1 by subtracting the nearest adversarial example from the original image x. We then pick a random vector orthogonal to this vector d_2 and normalise these vectors to obtain the final vectors that span the loss landscape in the form $x + \alpha \cdot d_1 + \beta \cdot d_2$. We normalise the vectors so that only the direction and not the magnitude of the vector is decisive for the perturbation. One example for this is shown in Figure 9. Manually inspecting this and other examples did not reveal any coherent similarities between the networks DACES dominated at (ResNet-50 and ResNet-101) and those it was third and second best (VGG-16 and ViT). One can see that the loss seems to be consistently lower for the latter two networks, which indicates again that the labels for these are harder to flip for DACES.

H.2. Statistical Analysis

We start the statistical analysis by investigating the per-instance performance, as deviations between these get lost when reporting the median. In fact, each attack has large differences in the minimum l_2-norm that can be found for each individual image. In other words: While for some images different attacks find success-

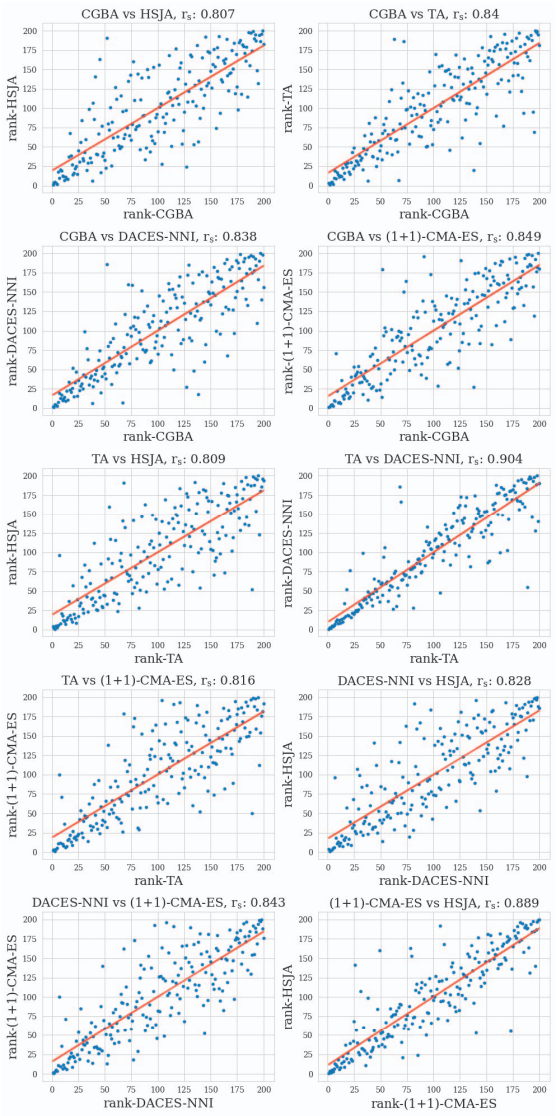


Figure 10. Per-Instance comparison of minimum norm on ImageNet with ResNet-50. r_s denotes the Spearman Rank Coefficient, the red line is the linear regression line.

ful perturbations with l_2-norms of below 0.1, for other images these can be orders of magnitude higher. This has also been found in previous research [3]. Intuitively, one would assume that each attack has difficulties with the same images (when these are far away from the decision boundary). We investigated this assumption by plotting the minimum norm found for each of the 200 images for ResNet-50, always comparing two attacks at a time (Figure 10). For that, we rank each of the 200 images used for evaluation based on the l_2-norm. We further use this ranked data to compute the Spearman Rank Coefficient r_s. We decided to use the rank of the images instead of the l_2-norm values to investigate whether the attacks struggle with the same instances, irrespective of the actual l_2-value. We then plot these ranks, comparing two attacks at a time. It should be noted that we conducted this analysis for all four networks, but we omit the other three as the analysis and conclusions are very similar.

Generally, a high level of correlation is observed between the instances solved best by each method, with Spearman Rank Coefficients ranging from 0.807 to 0.904. This suggests that the different images are generally equally challenging for all attacks. As noted earlier, we are particularly concerned with DACES and CGBA, as these attacks dominate for ResNet-50/ResNet-101 and VGG-16/ViT, respectively. Figure 10 reveals a high correlation between these attacks with a Spearman rank coefficient of 0.838.

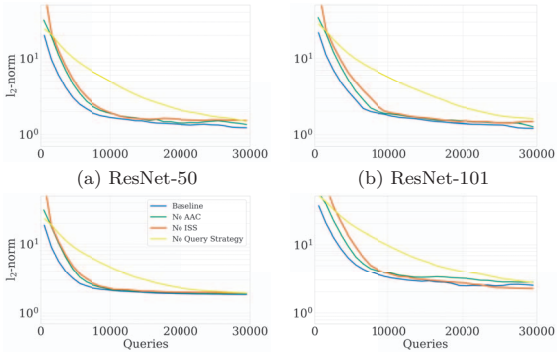
Last, we investigate whether the performance on single instances differs significantly between DACES and CGBA. In particular, we raise the question whether the performance of the attacks (*i.e.* the obtained l_2-norm) is within 10% for the different instances. If this was often the case, one could argue that the performance difference for the multiple networks is not that significant. But from the 800 instances that were investigated across the four networks, the performance of DACES and CGBA was within 10% only for 14 instances, once again stressing that they perform well on different networks.

Appendix I. Ablation

I.1. ISS, AAC and Query Strategy

We performed ablation studies to investigate the effectiveness of the ISS, AAC and our query strategy. For this, we compared DACES-BI with a variant with no ISS, whose parameters were set by the EvoTorch [33] library and a last one that always queried the whole population. For the experiments in our ablation studies, we used one random seed.

In Figure 11, we see that the baseline (DACES-BI) performed significantly better up to 10 000 queries, after which the variants without ISS and AAC show similar behaviour. One can see the advantage of warm-starting the algorithms and using the ISS, as this reduces the initial l_2-norm. Moreover, the variant without this feature failed to find an adversarial example for up to three of the 200 images on which we tested our attack. This shows that this feature helps the al-



(a) ResNet-50 (b) ResNet-101

(c) VGG-16 (d) ViT

Figure 11. Ablation results for ImageNet using bilinear interpolation. 'No AAC' indicates that the default parameters were used, 'No ISS' that we did not use our initial search strategy.

gorithm to always achieve a success rate of 100%, even for difficult instances. While the AAC improves performance for low query budgets, we can also see that our algorithm can be used without prior configuration, especially for higher query budgets. The query strategy has the biggest impact; the variant that queries the entire population performed worse over nearly all query budgets.

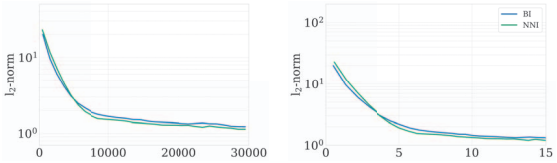
We also examined the impact of the three stages of our query strategy in detail. Across all seeds, images and networks (for ImageNet), the first stage (querying the closest instance) was used in 0.6% of generations, the second stage (querying μ_{eff}) in 61.6%, and the last stage (querying the whole population) in 37.8% of all generations. This resulted in using on average 323 and 23 785 fewer queries per image for stages 1 and 2, respectively. While the effect of stage 1 is negligible on average, it significantly benefits performance on specific instances, being used for over 5 000 generations for some images. As the AAC set the population size to $\lambda = 21$ for DACES-BI, querying 21 times each generation would have allow us to run the optimisation for $30\,000/21 = 1\,428$ generations. Our query strategy, however, enabled CMA-ES to run for 2 575 generations per image on average, which corresponds to 80% more generations.

I.2. Ablation on the Dimensionality Reduction

Despite their difference in performance, the two dimensionality reduction techniques have similar running times, as one can see in Table 11. BI is marginally faster on (Robust) ResNet-50 and VGG-16 and NNI on (Robust) ResNet-101 and ViT. In summary, no clear trend or trade-off can be seen with respect to running time.

Table 11. Average running time of DACES-NNI and DACES-BI in seconds per network on ImageNet.

	DACES-NNI	DACES-BI
ResNet-50	17.1	**16.6**
ResNet-101	**25.0**	25.1
VGG-16	17.2	**16.6**
ViT	**39.4**	41.4
Robust ResNet-50	22.0	**21.9**
Robust ResNet-101	**31.3**	31.6
Robust VGG-16	**24.1**	**24.1**



(a) l_2-norm per queries (b) l_2-norm per time

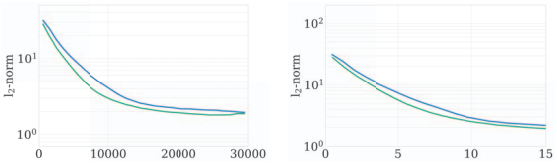
Figure 12. Median distance versus number of model queries and time over 200 images from the ImageNet testing set on a ResNet-50.

While the running time is similar, the situation is more nuanced concerning performance, as one can see in Figure 12, which shows the performance of DACES using NNI and BI. We depict the performance on ResNet-50, but the analysis for other networks with standard training (ResNet-101, VGG-16 and ViT) is similar. One can see that even though NNI generally dominates, BI has a slight advantage for a number of queries up to 5 000 or up to 4 seconds. The advantage with respect to time seems to be the same as with respect to queries due to the similar running time of the algorithms. For adversarially trained networks, such as the Robust RN-50, Robust RN-101 and Robust VGG-16, NNI dominates across all time and query budgets (see Figure 13). The outlined behaviour is confirmed in Table 7.

In conclusion, for normally trained networks it can be beneficial to use BI instead of the default NNI for small query or time budgets. These advantages, however, are smaller than the disadvantages on adversarially trained networks; even for the query budget of only 1k, NNI has a lower average l_2-norm of 20.08 compared to 21.19 for BI. This confirms the decision of using NNI as the default dimensionality reduction technique.

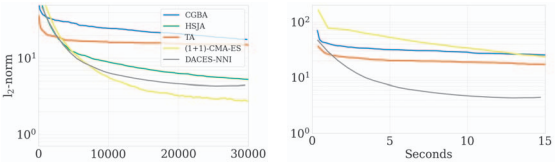
Appendix J. Example of Perturbed Image

Figure 14 shows one example of an original image with the final perturbation found after 30 000 queries and the resulting perturbed image.



(a) l_2-norm per queries (b) l_2-norm per time

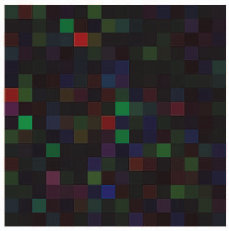
Figure 13. Median distance versus number of model queries and time over 200 images from the ImageNet testing set on an adversarially trained ResNet-50.



(a) Performance over queries (b) Performance over time

Figure 15. Median distance versus number of model queries/time over 200 images from the ImageNet testing set (lower is better).

(a) Original Image (label: goldfish)



(b) Perturbation (amplified by factor 300)

(c) Perturbed Image (label: sea slug)

Figure 14. Example perturbation

Appendix K. Analysis of the Sample Size

In this section, we analyse whether evaluating on more images significantly changes the results of DACES due to its stochastic nature. The authors of TA, CGBA, HSJA and the (1+1)-CMA-ES use 1 000 [35], 200 [28], 100 [3] and 100 [7] ImageNet images for evaluation. In the main paper, we take stochasticity into account by evaluating DACES over 10 independent seeds, *i.e.*, each of the 200 images of all networks was evaluated 10 times, leading to 2 000 runs per network.

We extended the evaluation on ResNet-50, its robust version, Robust RN-101 and Robust VGG-16 to 1 000 images (again, run 10 times with independent seeds) to verify, whether the results change significantly. We then compared the average performance on the original 200 images with the remaining 800 images using two-sided permutation test with 10 000 resamples using the mean statistic. We obtain p-values of 0.77, 0.24, 0.79 and 0.69 for ResNet-50, Robust RN-50, Robust RN-101 and Robust VGG-16, respectively. With a significance threshold of $\alpha = 0.05$, the null-hypothesis is rejected for none of the networks, meaning that there is no statistical significant difference between the two sample sizes.

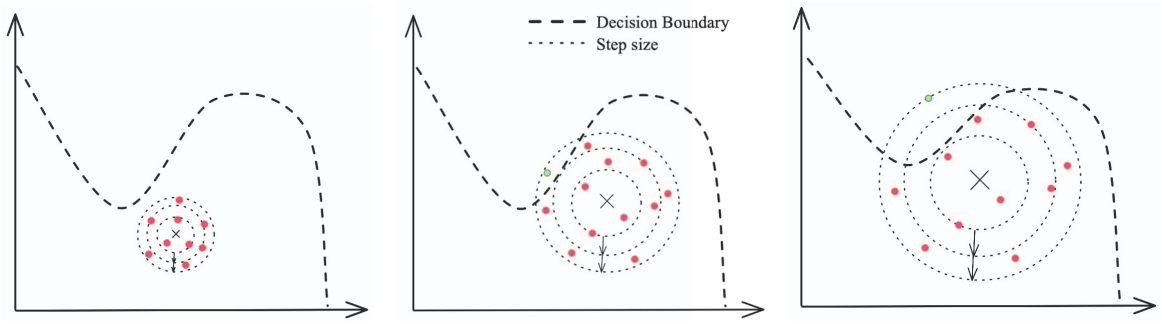
We still compared the average AUC DACES yielded on the two subsets to demonstrate this even further. On average, the performance on the subset of 800 images was only 0.3% better (with respect to AUC) than on the original subset.

Appendix L. Evaluation on Caltech101

Figure 15 shows the performance of all attacks on the Caltech101 dataset [16] using a ResNet-50. Caltech101 contains a total of 9 146 images, with 101 classes. The images are scaled to $224 \times 224 \times 3$. We again randomly sample 200 images for evaluation and follow the evaluation procedures from the main paper.

Figure 15(a) shows the performance over queries. DACES-NNI performs best up until 9 000 queries, from where (1+1)-CMA-ES performs best. This is notable, since the (1+1)-CMA-ES performs, on average, worst over all ImageNet and CIFAR-100 networks, stressing the high performance complementarity within decision-based attacks.

From the front-runner attacks, *i.e.* DACES-NNI, HSJA and CGBA, DACES performs best, followed by HSJA and CGBA. However, HSJA failed to find an adversarial example for one image. Since the (1+1)-CMA-ES performed subpar on ImageNet and CIFAR-100, this result further establishes DACES-NNI as the most consistent attack over all networks and datasets. Figure 15(b) shows the performance over time. Here, DACES-NNI clearly dominates from 1.5 seconds on.



(a) Step-size too small (b) Step-size just right (c) Step-size too big

Figure 16. Illustration of the Initial Search Strategy for the step-size σ. The cross represents the input image $\mathbf{x}_{\text{orig}}$, the red dots samples with label $f_\theta(\mathbf{x}) = y$, and the green dot a sample with label $f_\theta(\mathbf{x}) \neq y$. σ is increased quickly to find an adversarial example within the first generations (b). If σ is increased too slowly, the algorithm might fail to find an adversarial example at all (a) and if it is increased too big, the initialisation might be sub-optimal (c).

Appendix M. Considerations regarding the ISS

As established in Appendix I, the ISS clearly improves the performance of DACES. This section is concerned with algorithmic considerations regarding the ISS, which is mainly controlled by the step-size enlarger e_σ.

Recall that our approach aims to make the warm start of initialising with $\mathbf{m}^{(0)} = 0$ usable for decision-based attacks. We achieve this by multiplying the step-size (or standard deviation) σ each generation with a configurable parameter $\exp(e_\sigma)$, while fixing $\mathbf{m}^{(g)} = 0$, as illustrated in Figure 16. Only after the algorithm finds adversarial examples, it continues with the optimisation process.

If, however, e_σ is set too small, the ISS progresses too slowly in the search space, which costs time and queries. Even worse, with a significantly too small e_σ, DACES could possibly be failing to find an adversarial example at all, since it does not reach the decision boundary (Figure 16(a)). If it is set too high, however, the first found adversarial example might be far away, which puts an additional burden on the subsequent optimisation (Figure 16(c)). It becomes apparent that setting e_σ too small comes with greater risks than setting it too large.

This is why we treat e_σ as a hyperparameter and determine its optimal value among other hyperparameters with automated algorithm configuration. Further, we punish DACES during the configuration when it fails to find an adversarial example at all, see Appendix B.

Our experiments show that with the found e_σ, DACES has a 100% success rate on every dataset and model, indicating a proper setting of e_σ. As an example, DACES always finds the first adversarial example within the first 3 iterations for ResNet-50.