Decepticon

Frank-Wolfe and Attention against vision Transformers

Nicola Cassetta
nicola.cassetta@studenti.unipd.it

Andrea Di Trani
andrea.ditrani@studenti.unipd.it

Abstract

Deep nets provide state-of-the-art results for a plethora of tasks. However, most of them are vulnerable to "adversarial examples" i.e: modified inputs, able to mislead a trained model. Depending on how much information one can access to, adversarial attacks can be classified as white-box and black-box: in the first case the adversary has complete access to the target model, while in the second case, the attacker has minimum information about it. In this work we considered the image classification task and the de-facto SoTA architecture: Vision Transformers (ViT). We analyze several technique of adversarial attacks based on the Frank-Wolfe optimization algorithm and its variants, in both white and black box scenario, as well as in the untargeted and targeted case. Moreover we propose also novel, yet simple, step size - named 'Cumulative' - able to achieve a competitive AsR (Attack success rate), together with an attack based on the combination of the FW and the attention maps generated by the ViT.

1 Introduction

Szegedy et al. [8] first pointed out the existence of adversarial examples in the image classification domain: given an input x which belong to a class t, it is possible to find a new, maliciously perturbed, input x' that is similar to x but classified as t'. Based on the target class we can consider "targeted" and "untargeted" (i.e. any class that is not t is valid) attacks. This problem can be formulated as a constraint optimization problem: in the targeted scenario, we can consider the function $f(x) = \ell(x, y_{tar})$ as the attacker loss function, thus, the corresponding problem will result in:

$$min_x f(x)$$
$$subject\,to\, \|x' - x_o\|_p \leq \epsilon$$

In the white-box scenario, adversarial perturbation are generated via an iterative optimization algorithm and the already mentioned attacker loss function. This is the simplest scenario, since the attacker has access to the gradient and to the model's architecture. In a real case scenario, i.e: black-box, the attacker has minimum information about the model, and he can just query the output of the classifier or access to the confidence scores of all classes, thus, an estimation of the gradient is required.

2 Vision Transformers

In the 1980s, the world saw its first CNN (Convolutional neural network) developed by postdoctoral computer science researcher Yann LeCun [9]. Since then, convolution has played a key role in computer vision. In the 2021, a paper named "An image is worth 16x16 words" [7] demonstrated that this reliance on CNNs was not necessary: by modifying the input in a clever way, the Google 'Brain' team, managed to achieve the SoTA results in the image classification task with an architecture developed for NLP tasks, which does not rely on convolution, but rather on self-attention and positional encoding:

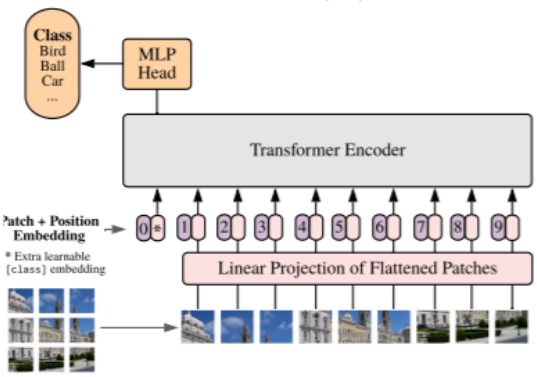


Figure 1: Vision Transformer architecture

The ViT model represents an input image as a se-

ries of image patches, like the series of word embeddings used when using transformers to text, and directly predicts class labels for the input image.

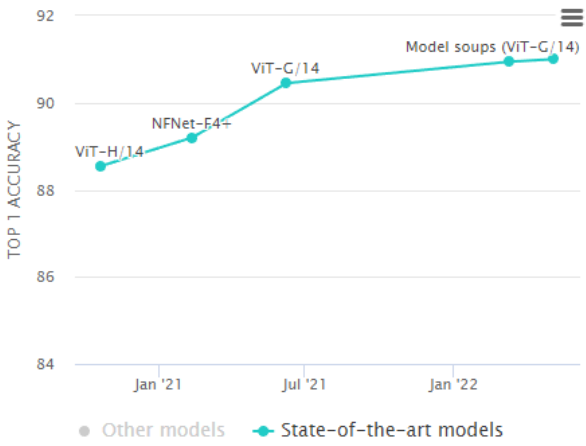


Figure 2: State of the art on Image classification on Imagenet1k

For the purpose of this project, we used a publicly available ViT pre-trained on ImageNet-21k at resolution 224x224 (16 patches), and fine-tuned on ImageNet 2012 (1,000 classes) at the same resolution, named 'ViT-B-16'.

3 Frank-Wolfe

In literature there are many works related to the behaviour of various optimization-based methods in the context of adversarial attacks, such as FGSM [10], PGD [11] and C&W [5] to cite the most commons. However there are very few works that analyze the behaviour of the Frank-Wolfe (FW) [3]. The FW is one of the oldest methods for constrained convex optimization and, recently, it has seen an impressive revival due to the proof of some nice proprieties even in a non-convex scenario:

- (Yu, Zhang, and Schuurmans 2017) Proved the first convergence rate for Frank-Wolfe type algorithm in the non-convex setting. [3]
- (Balasubramanian and Ghadimi 2018) Proved the convergence rate for zeroth-order non-convex Frank-Wolfe algorithm. [3]

Moreover, the PGD, compared to the FW, has a more "aggressive" approach: it first takes a step towards the negative gradient direction while ignoring the constraint to get a new point, often outside the constraint set, and then correct the new point by projecting it back into the constraint set. The FW, instead, is a projection-free method as it calls a Linear Minimization Oracle (LMO) over the constraint set at each iteration.

Algorithm 1 Frank-Wolfe method

Choose a point $x_1 \in C$
for $k = 1, \ldots.$ **do**
  Set $\hat{x}_k = \underset{x \in C}{Argmin}\ \nabla f(x_k)^T (x - x_k)$ ▷ LMO
  if $\hat{x}_k$ satisfies some conditions **then**
    STOP
  else
    Choose α via a suitable stepsize:
    $x_{k+1} = x_k + \alpha_k(\hat{x}_k - x_k)$
  end if
end for

Although the remarkable results of this method, its convergence rate is known to be slow (sublinear) when:

- the solution lies at the boundary: FW directions, when close to this type of optimum points, becomes more and more orthogonal to the gradient leading to a "zig-zag" behaviour as shown in the picture below.

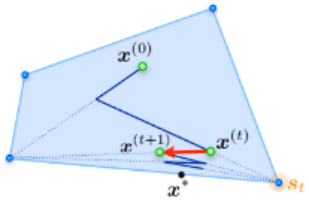


Figure 3: zig-zag behavior, picture taken from [1]

- the objective function doesn't have LCG or is not σ-strongly convex

We based our standard FW algorithm (in both white and black-box scenario) on the work of [3] which includes the momentum mechanism and a ensures in both cases a convergence rate of $\mathcal{O}(1/\sqrt{T})$.

The popularity of the FW, in fact, is mostly derived from its variants, which can achieve a linear convergence rate even if the optimal solution lies on the boundaries of the set. Among these variants it is mandatory mentioning: the Pairwise FW and the Fully-corrective FW. For the implementation of these two variants we took inspiration from the algorithms presented in the work of [1].

3.1 FW Pairwise

The main idea of this variant is to move the weight from the away vertex to the Frank-Wolfe vertex. In practice at each iteration we use as a search direction the combination of the standard FW direction and the so called 'away-direction' of the FW Away-Step version (that is: the direction pointing away from the worst vertex). Even if the linear rate is more loose compared to the Away-Step one, this method is much more efficient in practice:

Algorithm 2 Pairwise FW

Let $x^(0) \in A$ and $S^{(0)} := x^{(0)}$
for $t = 0, ..., T$ **do**
  Set $s_t := LMO_A\ (\nabla f(x^t))$
  Set $v_t := \underset{v \in S^t}{Argmax}\ \langle \nabla f(x^t), v \rangle$
  Set $d_t^{FW} := s_t - x^(t)$ ▷ FW direction
  Set $d_t^A := x^{(t)} - v_t$ ▷ Away direction
  if $g_t^{FW} := \langle -\nabla f(x^t), d_t^{FW} \rangle \leq \epsilon$ **then**
    RETURN x^t ▷ FW gap small enough
  end if
  $d_t = d_t^{PWF} := s_t - v_t$
  $\gamma_{max} := \alpha_{vt}$
  Line-search: $\gamma_t \in [0, \gamma_{max}]$
  Update $x^{(t+1)} := x^{(t)} + \gamma_t d_t$
  Update $S^{(t+1)} := \{v \in A \text{ s.t. } \alpha_v^{(t+1)} > 0\}$
end for

3.2 FW Fully-corrective

This method uses an iterative inner approximation of the feasible set. The set is approximated with the convex hull of an ever expanding - finite - set made up of the atoms (extreme points) of the starting set:

Algorithm 3 Fully-corrective FW

Input: Set of atoms A, active set $S^{(0)}$,
start $x^{(0)} = \sum_{v \in S^{(0)}} \alpha_v^{(0)} v$, stopping criterion ϵ.
Let $A^{(0)} := S^{(0)}$
for $t = 0, ..., T$ **do**
  Set $s_t := LMO_A\ (\nabla f(x^t))$ ▷ FW atom
  Set $d_t^{FW} := s_t - x^{(t)}$
  Set $g_t^{FW} := \langle -\nabla f(x^t), d_t^{FW} \rangle$ ▷ FW gap
  if then $g_t^{FW} \leq \epsilon$
    RETURN x^t
  end if
  $(x^{(t+1)}, A^{(t+1)}) :=$ **Correction** $(x^{(t)}, A^{(t)}, s_t, \epsilon)$
end for

Algorithm 4 Correction

Input: $(x^{(t)}, A^{(t)}, s_t, \epsilon)$
RETURN $(x^{(t+1)}, A^{(t+1)})$ s.t. :
- $S^{(t+1)}$ active set for $x^{(t+1)}$ and $A^{(t+1)} \supseteq S^{(t+1)}$
- $f(x^{(t+1)} \leq \underset{\gamma \in [0,1]}{min} f(x^{(t)} + \gamma(s_t - x^{(t)}))$
- $g_{t+1}^A := \underset{v \in S^{(t+1)}}{max} \langle -\nabla f(x^{(t+1)}, x^{(t+1)} - v) \rangle$

3.3 FW Black-box

As already mentioned in the previous sections, in the black-box setting we cannot perform backprop to calculate the gradient of the loss function, thus, we need to add to the algorithm an estimation step:

Algorithm 5 Gradient estimation step

Input: x, b, δ
Fix: $q = 0$
for $i = 1, ..., b$ **do**
  $q = q + \frac{1}{2\delta b}(f(x + \delta u_i) - f(x - \delta u_i)) * u_i$
end for
Return q

where u_i is sampled from the standard Gaussian distribution $N(0, 1)$.

4 Integrating attention

A Transformer block consists of multiple heads. Each head projects the input data to different sub-spaces, and this helps each individual head to attend to different parts of the image.
In order to speed up the gradient estimation step, we started wondering if a proper use of these maps could reduce the time needed to perform a black-box attack (the estimation of the gradient requires requires several sampling of a tensor of size [256,256,3], which is an extremely costly operation). We merged the different attention heatmaps (one for each head) into a single map, then we used a threshold in order to keep only the most relevant parts. At this point we used the heatmap as a sequence of coordinates, and used these coordinates as a 'mask' in order to avoid the sampling of the full image. This approach comes from the intuition that, maybe, to fool a network, it is sufficient to attack the main subject of the image.

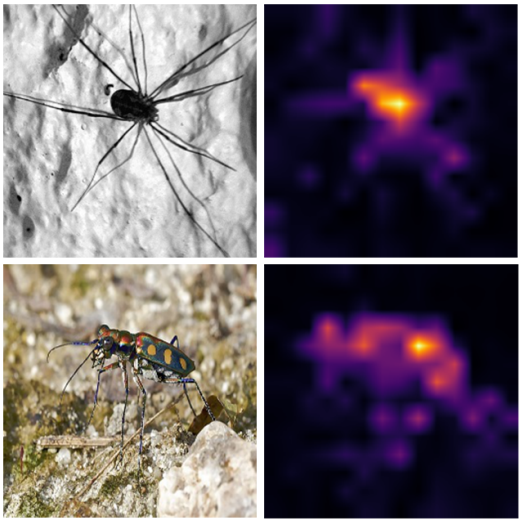


Figure 4: Images from Imagenet and their merged attention map

5 Distance metrics

Recalling the problem formulation mentioned in Section 1, we used several distance metrics (norms) to quantify the similarity between the initial image and the perturbed one. In detail:

- L1: sum of abs value of the vector's component
- L2: standard Euclidean distance
- Infinity: measures the maximum change to any of the coordinates
- Nuclear: following the work of Ehsan Kazemi and Thomas Kerdreux [2] we used the nuclear norm which is the sum of the matrix singular value, a.k.a. the trace norm or the 1-Schatten norm.

Note that for all the distortion sets which we considered in this work, LMO presents a closed form solution: thanks to the work of [3] we have a derivation of LMO for general $p \geq 1$, that is:

$$x_i = -\epsilon \cdot \frac{\operatorname{sign}\left((m_t)_i\right) \cdot |(m_t)_i|^{\frac{1}{p-1}}}{\left(\sum_{i=1}^{d} |(m_t)_i|^{\frac{p}{p-1}}\right)^{\frac{1}{p}}} + (x_{\text{ori}})_i$$

Instead, for the Nuclear norm, we used the solution of [2]

$$\text{LMO}_{||\cdot||_{S1} \leq \rho}(M) \triangleq \rho\, U_1 V_1^T$$

where for $\rho = 1$ we have that U_1, V_1 are the first columns of the matrices U and V of the SVD decomposition of the matrix M given by USV^T.

Recalling the results of [4], we have that the complexity of one FW iteration with these norms is:

Norm	Complexity
ℓ_p-Sphere	$\mathcal{O}(n)$
Nuclear	SVD

6 Line search

The update step:

$$x_{k+1} = x_k + \alpha_k(\hat{x}_k - x_k)$$

requires the calculation of α_k. This operation represent the so-called 'line search'. There are several ways of solving this problem, and in the following subsections we report all the techniques that we used

6.1 Fixed

A fixed step size is given at each iteration: $\alpha_k = s$ with s > 0

6.2 Armijo

In this case we look for a condition to be satisfied in order to guarantee a decrease of the objective function:

Algorithm 6 Armijo Line search

Fix: $\delta \in (0,1)$, $\gamma \in (0,1/2)$, Δ_k = starting step
for $m = 0, \ldots$ **do**
  $\alpha = \delta^m \Delta_k$
  if $f(x_k + \alpha d_k) \leq f(x_k) + \gamma\alpha\nabla f(x_k)^T d_k$ **then**
    RETURN $\alpha_k = \alpha$ ▷ sufficient decrease of f
  end if
end for

6.3 Adaptive

Following the work of [2] we have:

Algorithm 7 Adaptive step

$\gamma_t = clip_{[0,1]}(\langle -\nabla f(x_t, s_t - x_t\rangle / 2(\epsilon)^2)$

6.4 Cumulative

Inspired by the dimishing step size and by the parallelism between the gradient descent and the ball rolling down from a cliff, we tried to formulate a (novel) algorithm for the step size calculation, specifically designed for the adversarial attack task:

Algorithm 8 Cumulative step

Fix: $counter_{lab} = 0$, $pred_{t-1} = 0$
if $pred_{t-1} = pred_t$ **then**
  $counter_{lab} += 1$
  $\alpha_k = \alpha + \underset{[0,1]}{clip}((\alpha * counter_{lab})/2)$
else
  reset $counter_{lab}$
end if

Each time the model predict the modified image as the correct one, we increase the step size by a certain amount (clipped in [0,1]).

7 Test suite

Due to the large amount of tests, and due to the limited resources (both time and computational power), we decided to schedule an extensive explorative test suite on the standard Frank-Wolfe algorithm in the white-box scenario, as a first step, and then, we used the best set of hyperparameters and technique for all the other tests. We evaluated our attacks on a subset of the ImageNet 2012 validation set: we randomly sampled 100 correctly classified images, and used this sample over all the tests. We allowed for a maximum of 100 iteration over a single image.
We highlight the fact that the choice of hyperparameters and test methodologies, as well as evaluation metrics, has been made in such a way as to compare the results obtained with those already present in the literature. All the test have been carried out in Colab with Python 3.8, using a Tesla T4 (12 GB RAM)

	Max distortion		
Norms	ϵ**1**	ϵ**2**	ϵ**3**
L1	3	5	10
L2	1	3	5
Infinity	0.001	0.01	0.1
Nuclear	1	3	5

Table 1: In this table we show the distortion values used for each norm. Since they are quite different, when presenting the results, we will use just ϵ_n, with n in range [1,3] i.e: min distortion, avg distortion and max distortion

7.1 Explorative test results

Here we present the results for the the vanilla FW in the white-box context. Starting from the error rate, we have:

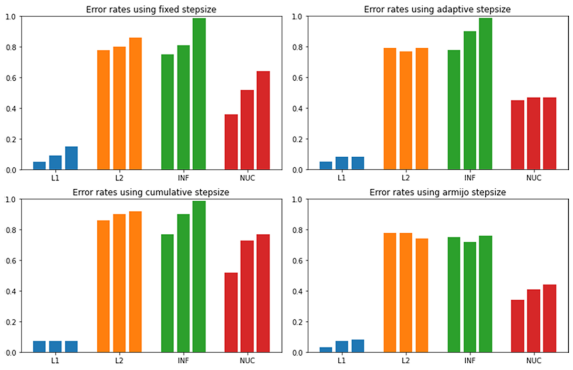


Figure 5: Error rate in 100 epochs. Starting from the top-left corner: Fixed, Adaptive, Cumulative, Armijo

	L1			L2			INF			NUC		
	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3
Fix	0.05	**0.09**	0.15	0.78	0.8	0.86	0.75	0.81	**0.99**	0.36	0.52	0.64
Cumu	0.07	0.07	0.07	0.86	0.9	**0.92**	0.77	0.9	**0.99**	0.52	0.73	**0.77**
Armj	0.03	0.07	0.08	0.78	0.78	0.74	0.75	0.72	0.76	0.34	0.41	0.44
Adapt	0.05	0.08	0.08	0.79	0.77	0.79	0.78	0.9	**0.99**	0.45	0.47	0.47
Armj*	0.05	0.05	0.06	0.84	0.9	0.9	0.77	0.89	**0.99**	0.54	0.65	0.69

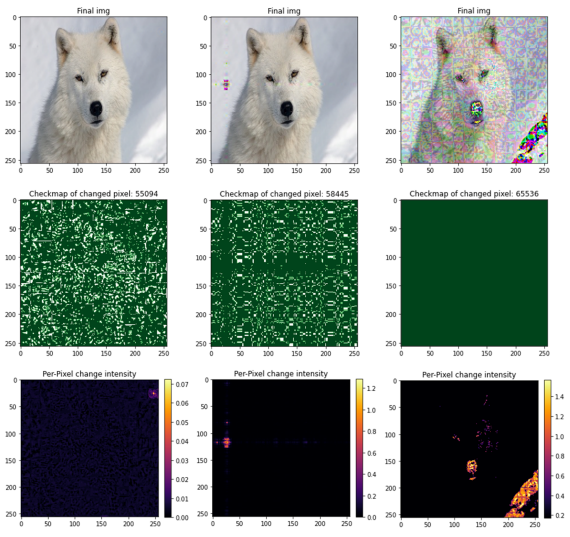
Figure 6: Error rate results for FW in WB. Armj* has a stepsize x50 times larger than the the others (0.5)

as we can see, the L1 norm has the lowest error rate, i.e: the images modified via with this norm have a very little probability to mislead the network, regardless of the maximum distortion allowed. The norms with the highest error rate are the L2 and the Infinity, followed by the Nuclear. Moreover is interesting to see how the Armijo with the increased stepsize has a much higher error rate (+20% on average) with respect to the standard one (this test was carried out due to the shrinking structure of the Armijo: starting with a much larger step size, allows the full potential of this method). However, we must emphasize that the error rate alone is not a good indicator to quantify the goodness of an attack: we're not looking for the attack with the highest error rate, but we're looking for the attack with the highest error rate together with the least perceivable modification to the input image, and the absence of a metric that allows for a clear measure of human perceptual similarity, raises the need to support these results with an analysis of the intensity of the modified pixels, plus a visual inspection. We can look further and focus our attention on the average percentage of pixels changed:

	L1			L2			Inf			Nuc		
	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3	ϵ1	ϵ2	ϵ3
Fix	**0.14**	0.13	0.13	84.7	85.9	86.7	84.5	88.3	91.1	98.7	98.9	99
Cumu	0.05	0.05	0.05	85.1	86.9	87.7	84.4	89.7	91.4	99.1	99.1	99.2
Armj*	**0.14**	0.13	0.13	84.2	85.0	85.2	84.5	84.8	90.2	98.6	98.8	98.9
Adapt	0.12	0.12	0.12	85.5	87.4	**87.9**	84.4	**100**	**100**	99	99	**100**

Figure 7: Changed pixels FW in WB

as we'd expect, the L1 has the lowest percentage, while the Infinity and the Nuclear, regardless of the line search method, modify almost every pixel of the image. We can now continue with a visual inspection of the resulting images:



The above figure is divided in three rows: the first row shows the attacked images, the second row reports the coordinates of modified pixels (green: changed pixel) and the third row indicate the intensity of change of each pixel. In this case the results are related to the 'highest error-rate parameter setup' (Fig.5), starting from left:

- L2 + $\epsilon 3$ + Cumulative step size
- Nuclear + $\epsilon 3$ + Cumulative step size
- Infinity + $\epsilon 3$ + Adaptive step size

as we can see, even if the Infinity norm, with $\epsilon 3$, achieves almost a 100% attack success rate, the resulting image is completely corrupted, while the Nuclear norm performs quite well, with just a single visible artifact on the left of the image. As mentioned at the beginning of this section, it is clear that the use of the 'error-rate' alone is not enough to state that an attack is 'good'.

On the basis of the results obtained during this first phase of testing, we will test all the other methods with the the norms: L2, Infinity and Nuclear, with a maximum distortion value equal to the $\epsilon 2$ value, with Fixed, Cumulative and Armijo* as step sizes.

In order to avoid to flood this work with images, in the following subsections we will only report the results for each method, while we will use the last section 'Visual inspection' to show some of the resulting images (together with the loss and the norm difference) of each test.

7.2 FW Pairwise results

	ϵ **2**		
	L2	**Inf**	**Nuc**
Fix	0.8	0.86	0.58
Cumu	0.98	**1**	0.85
Armj*	0.93	0.93	0.78

Table 2: error rate FW PW

In this scenario almost all the pixel (99% +) where modified

7.3 FC Fully Corrective results

	ϵ **2**		
	L2	**Inf**	**Nuc**
Fix	0.77	0.78	0.37
Cumu	0.77	0.78	0.37
Armj*	0.8	**0.81**	0.39

7.4 Standard FW targeted attack results

As mentioned in the first section, with the targeted attack, we want our input image x to be classified as a specific y_{tar} rather than a generic y' s.t. $y' \neq y_{true}$. This result in the attacker loss function to become $f(x) = \ell(x, y_{tar})$. For these tests we chose a fixed target label: *"white wolf, Arctic wolf, Canis lupus tundrarum"*.
These results are related to a different set of hyperparameter: 150 iteration per image, instead of 100, and $\epsilon 3$ instead of $\epsilon 2$. This choice has been made due to the poor results of the standard set of hyperparameter in this scenario: in this dataset there are several labels that are really different if compared to our target, thus, we assumed, that in this case we could allow a greater distortion (and a higher number of iteration):

	$\epsilon 3$		
	L2	**Inf**	**Nuc**
Fix	0.34	0.42	0.01
Cumu	0.48	**0.6**	0.01
Armj*	0.42	0.49	0.01

Table 3: Error rate for standard FW white-box targeted attack

It seems that the Nuclear norm is not a good norm in this scenario, even if we allowed for the maximum distortion. The best result is a 60% success rate for the Infinity - Cumulative.

7.5 FW & variants time analysis

In this section we show the time needed on average for each method to attack an image

	Avg Time per img (sec)			
	L1	**L2**	**Inf**	**Nuc**
Fix	5	1,4	1,08	10,24
Cumu	4,85	**1**	0,87	8,07
Armj*	10,73	2,47	1,74	13,1
Adapt	4,9	1,39	0,87	10,23

Table 4: Time table FW vanilla

	Avg Time per img (sec)		
	L2	**Inf**	**Nuc**
Fix	2,13	1,91	11,7
Cumu	0,86	**0,58**	7,71
Armj*	2,53	2,21	12,2

Table 5: Time table FW PW

	Avg Time per img (sec)		
	L2	**Inf**	**Nuc**
Fix	74,9	79,6	193,9
Cumu	74,9	79,7	200,6
Armj*	65,2	**60,1**	201,4

Table 6: Time table FW FC

As we can see, the FW-FC requires an enormous amount of time if compared to all the other methods: we expected this result since its structure involve several costly operations. This method, despite the nice theoretical results, performs poorly in our context, on both the error rate and the time needed. Instead the FW-PW presents some really interesting results: we have an overall time reduction if compared to the standard FW, with an astonishing 0,58s with the Infinity norm and the Cumulative step size (we recall that this combination achieves the 100% attack success rate, however is the most 'brutal' attack setup since - as shown in the previous sections - the Infinity norm tends to corrupt the images, while the Cumulative step size is the most aggressive among all the other presented). Also, it is possible to see that the Nuclear norm, despite its interesting results, is the norm that requires the largest amount of time, and this is due to the fact that with this approach we need to compute the SVD at each iteration.
With these results we can say that the best attack (Error rate + low perception + time needed) could be the one carried out with the FW-PW and the L2 - $\epsilon 2$ norm.

8 FW black-box

In the following subsections we show the results of the standard FW in the black box scenario. We must point out that, due to the structure of this attack, the time needed to perform a test was extremely high (4h vs 1h if compared to the white-box). For this reason we were forced to further reduce the tests, leaving only the best setup among all the previous ones.
All the following results have been carried out with 50 iteration, and 5 sample size as sampling parameter (gradient estimation step). The maximum distortion allowed is the $\epsilon 3$.

8.1 FW Black-box results

	ϵ **3**		
	L2	**Inf**	**Iter**
Fix		0.08	12.5
Cumu	0.03		11.3
Armj*	0.03	**0.11**	1 - 4.4

Table 7: Error rate FW vanilla in black-box. Iter column: average number of iteration per image required before mislead the net

	Avg Time per img	
	L2	**Inf**
Fix		175
Cumu	164	
Armj*	182	156

8.2 FW Black-box + Attention results

With some prelimiar tests we have seen how the use of the attention map used to reduce the sample space allowed for a great reduction of the time needed to complete an attack. Thus in this scenario we used 80 iteration and 50 as sample size for the gradient estimation, and we still have a reduction on the overall time elapsed if compared to the standard black-box!

	ϵ **3**		
	L2	**Inf**	**Iter**
Armj*	0.0	**0.3**	2.3

Table 8: Error rate with standard FW black-box + attentiont

	Avg Time per img	
	L2	**Inf**
Armj*	137	131

If we want to analyze the speed up gained with the attention, we have that this approach allows for a $> 2\times$ speed up

WHITE-BOX					
	AsR	**Norm**	**Step size**	**Time**	**Iter**
FW	0.9	L2	Cumu	1	10.2
			Armijo*	2,47	9.1
	0.9	INF	Cumu	0,87	8.8
			Adapt	0,87	4.2
FW-PW	1	INF	Cumu	0,58	10.3
	0.98	L2		0,86	12
	0.93	INF	Armijo*	2,21	9.7
FW-FC	0.81	INF	Armijo*	60,1	8.9

Table 9: Iter column: average number of iteration per image required before mislead the net

We can clearly see that the best norms in this scenario are the L2 and the Infinity, while the best step size choice is one between the Armijo* and the Cumulative. The FW-PW presents the best results among all the versions of the FW, with a 100% attack succes rate (AsR) and an avg time per image equal to 0,58s.

In the targeted scenario, the combination of hyperparameters that allows for a 50%+ AsR is the Infinity norm + Cumulative step size and $\epsilon 3$. We assume however that this result may vary depending on the target label chosen.

In conclusion, for the white-box scenario, our results turns out to be much better than the ones obtained with other algorithms (as FSGD, C&W and PGD) in [6], using a much lower distortion for ($\epsilon = 0.01$ instead of $\epsilon = 0.062$)

L_∞				
Model	**FW**	**FGSM**	**PGD**	**C&W**
ViT-B-16	**90%**	23.1%	0.0%	0.0%

9 Final conclusions

In this section we will discuss the results of this work. Starting from the white-box context, here we present a small table that summarize the best results among all the tests performed with $\epsilon 2$:

while being in line with the ones obtained on different models using the FW method [3]. Considering the black-box scenario, we have shown how our results are incredibly lower if compared to the ones of [3].

L_∞	
Model	**FW Black**
ViT-B-16	11%
InceptionV3	**98.4%**

Nevertheless our results are related to a different model (ViT) which seems extremely robust to this type of black-box attack, plus, our results are based on a sample of 100 images instead that 250 as for the work of Chen. We also allowed for a smaller max distortion ($\epsilon = 0.01$ instead of $\epsilon = 0.05$)

Unfortunately, the use of the attention heatmaps does not resulted in a higher AsR, but only in a much faster (2x) and localized attack. We hypothesized that, in order to speed up and improve a black-box attack, was necessary to focus the attack on the main subject of the image, according to the attention heatmap, however this approach did not improved the AsR in any hyperparameter setup.

As a last point for this section, we want to summarize some other results of this project:

- To the best of our knowledge there are no other works publicly available on the Nuclear-norm with the ImageNet dataset
- To the best of our knowledge there are no other works publicly available that analyze the performance of Chen's zeroth-order Frank-Wolfe based algorithm with a ViT
- To the best of our knowledge there are no other works publicly available that analyze the performance of FW variants in the advarsarial attack contest
- Our idea of using the attention map to increase the performance of the black-box attack didn't work
- The Cumulative step size performs really well in our scenario

10 Future work

With this work we only saw the tip of the iceberg of the possibilities that this domain offers, and we have already detected some points that could be improved in the future:

- Include the intensity information of the attention map to the computation of the step size.
- Implement an approximation of the SVD to speed up the Nuclear norm.
- Analyze how different labels affect the AsR of the targeted attack.
- Search for different Black-box attack.

11 Visual inspections

In this section we show two types of results: one related to the loss function and norm difference behaviour and one related to the resulting images. Since in the white box scenario often we do not have the possibility to appreciate the trends of the two functions (1 or 2 iterations are enough to fool the model), we let the algorithm proceed even after an incorrect prediction. In the second part of this section, we just show some of the resulting images that have been missclassified.

11.1 Loss and Norm diff analysis

11.1.1 FW white L2 Armijo*

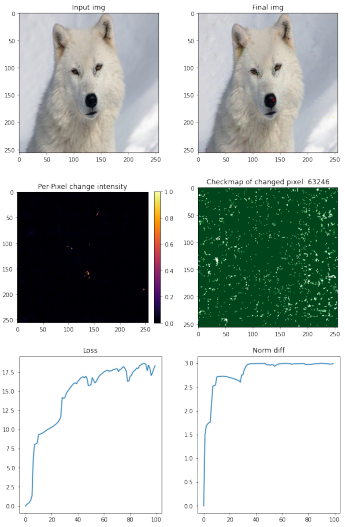


Figure 8: FW white L2 Armijo*

11.1.2 FW white L2 Cumulative

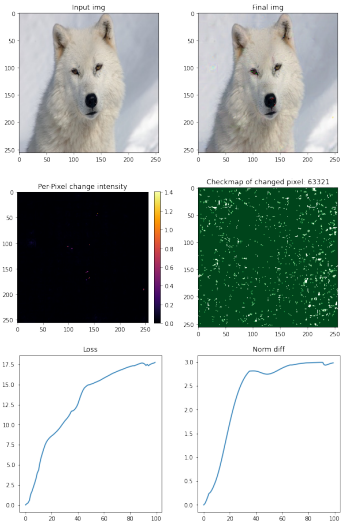


Figure 9: FW white L2 Cumulative

11.1.3 FW white L2 Fixed

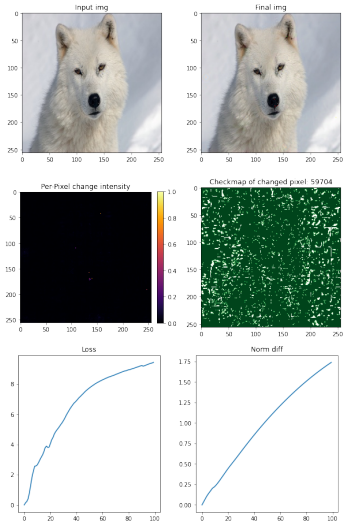


Figure 10: FW white L2 Fixed

11.1.4 FW white Nuclear Armijo*

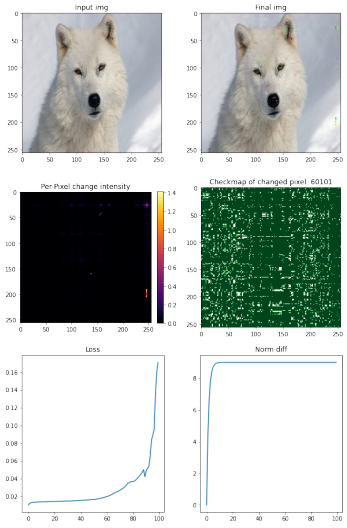


Figure 11: FW white Nuclear Armijo*

11.1.5 FW white Nuclear Cumulative

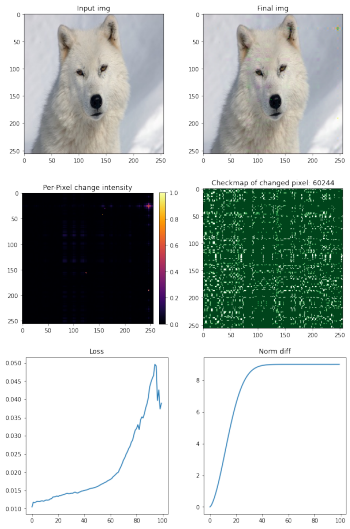


Figure 12: FW white Nuclear Cumulative

11.1.6 FW white TARGET L2 Armijo*

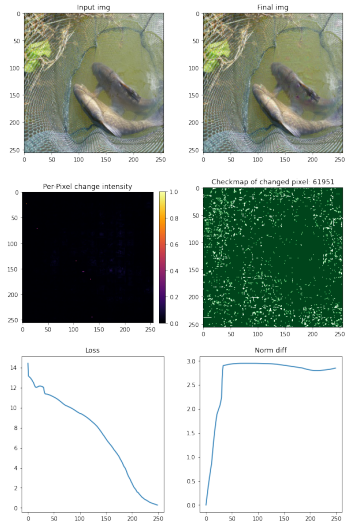


Figure 13: FW white TARGET L2 Armijo*. Target: (270) white wolf, Arctic wolf, Canis lupus tundrarum

11.1.7 FW BLACK L2 Armijo*

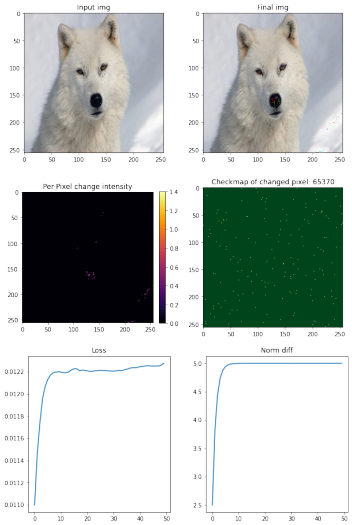


Figure 14: FW BLACK L2 Armijo*

11.1.8 FW BLACK + Attention L2 Armijo*

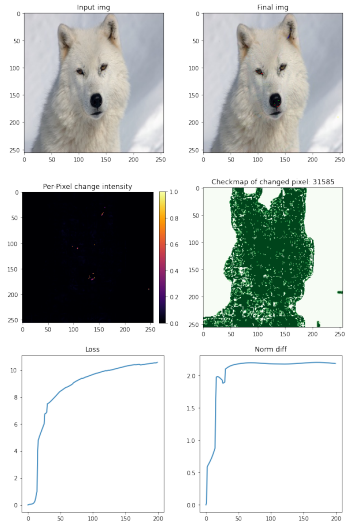


Figure 15: FW BLACK L2 Armijo*

11.2 Examples of misleading images

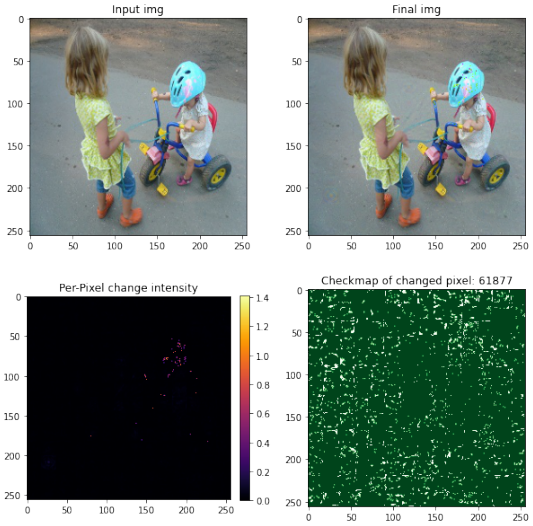


Figure 16: FW white L2 Armijo*, targeted attack: starting label "tricycle, trike, velocipede", confused with "White wolf"

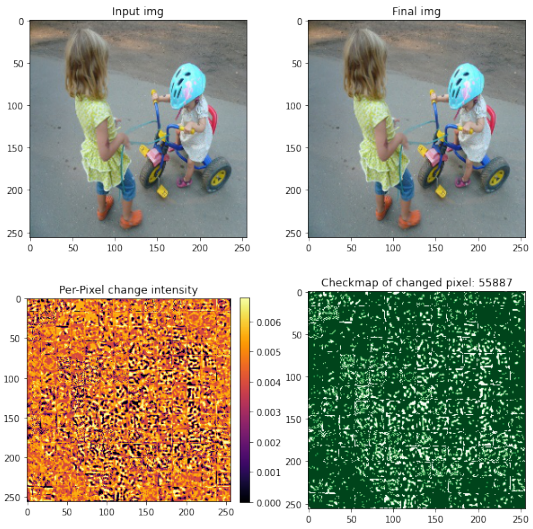


Figure 17: FW-PW white L2 Armijo*: starting label "tricycle, trike, velocipede", confused with "tank, army tank, armored combat vehicle"

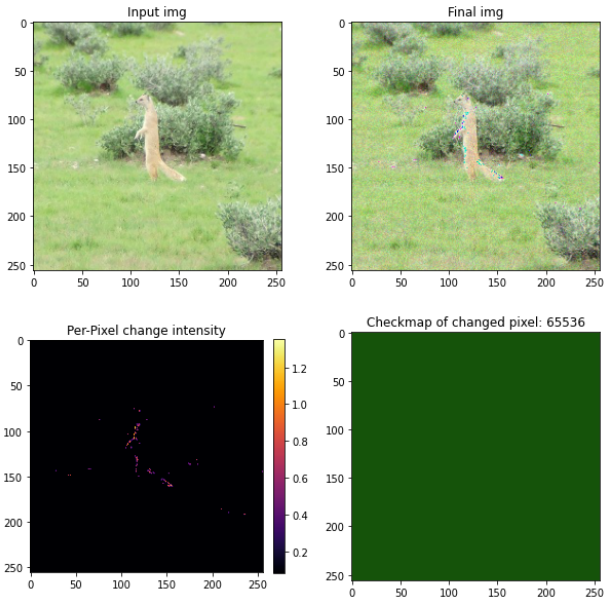


Figure 18: FW Black L2 Armijo*: starting label "mongoose", confused with "weasel"

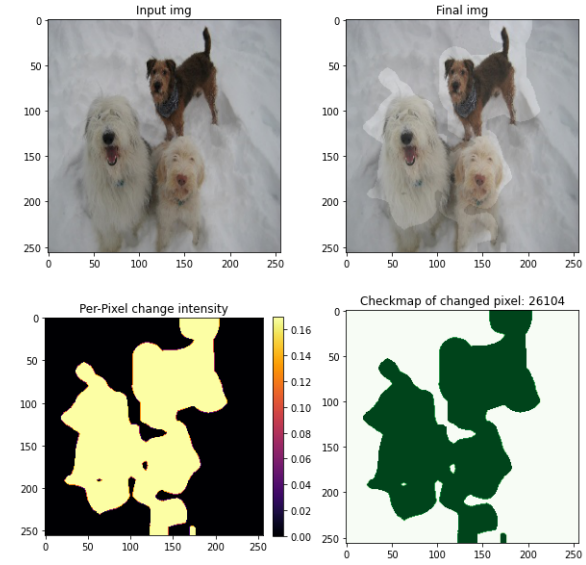


Figure 19: FW Black + Attention L2 Armijo*: starting label "Irish wolfhound", confused with "Norfolk terrier"

References

[1] On the Global Linear Convergence of Frank-Wolfe Optimization Variants - Simon Lacoste-Julien, Martin Jaggi.

[2] Generating Structured Adversarial Attacks Using Frank-Wolfe Method - Ehsan Kazemi, Thomas Kerdreux.

[3] A Frank-Wolfe Framework for Efficient and Effective Adversarial Attacks - Jinghui Chen, Dongruo Zhou, Jinfeng Yi, Quanquan Gu1.

[4] Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization - Martin Jaggi.

[5] Towards Evaluating the Robustness of Neural Networks - Nicholas Carlini, David Wagner.

[6] On the Robustness of Vision Transformers to Adversarial Examples - Kaleel Mahmood, Rigel Mahmood, Marten Van Dijk.

[7] An image is worth 16x16 words: Transformers for image recognition at scale - Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby.

[8] Intriguing properties of neural networks - ICLR (2013). Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R.

[9] GradientBased Learning Applied to Document Recognition - Yann LeCun Leon Bottou Yoshua Bengio and Patrick Haffner.

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

[11] Towards Deep Learning Models Resistant to Adversarial - Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, Adrian Vladu.