Comparison of Frank-Wolfe Varients for White-Box Adversarial Attacks

Tanner Aaron Graves - 2073559 Alessandro Pala - 2107800

June 2024

1 Abstract

With deep neural networks becoming ubiquitous in application, adversarial attacks have recieved much attention, as it has proved remarkably easy to create adversarial examples- genuine data that undergoes a minimal and unobtrusive corruption process inorder to maximally harm the performance of a model. Access to a models architecture can enable white-box attacks, where gradients of loss with respect to input examples are exploided to create adversarial examples. The requirement that examples be minimally preturbed is a constraint on the optimization. The ability of Frank-wolf and vairents have gathered much attention for their ability to be efficiently create adversarial examples while staying in a constraint set. In the paper we introduce discuss the application of Frank-Wolfe and two varients to this non-convex constrained optimization problem. Furthermore we discuss popular optimzations and their effect convergence and attack efficacy, comparing performance on attacks on models trained on MNSIT, FashionMNIST, and CFAIR-10 datasets. Finally, there is a discussion of the theoretical underpinnings of each algorithim.

2 Introduction

This is typically stated as the following constrained optimization problem:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & ||x||_p \leq \epsilon \end{aligned} \tag{1}$$

In the case of untargeted attacks on a classifier, we perturb an example with the aim it be incorrectly predicted as any other class. Here $f(x)$ is the loss function of the attacked model $-\ell(x, \hat{y})$. In the case of targeted attacks, we aim to maximize the liklihood of another class $y \neq \hat{y}$. The cost then is $f(x) = \ell(x, y)$. We have implemented both targeted and untargeted attacks, and come to focus on targeted attacks as the algorithioms are seen to require more iterations

to converge. The L_p constraint $||x||_p \leq \epsilon$ directly restricts the size of perturbations made the to the example. An inherient problem of DNNs is often attacks can be consistently sucessful even with very small, even imperceptable ϵ. Different choices of p may be made giving $||x||_p = (\sum_i x_i^p)^{1/p}$. Or commonly, as we use here $L_\infty(x) = \max_i |x_i|$. We define the constraint set $\mathcal{M} = \{x : L_p(x) \leq \epsilon\}$. Of pirticular note is when $\mathcal{M}$ is a polytope, as is the case when $p \in \{1, \infty\}$. This corresponds to models making perturbations that are either sparse or have a maximal distubance along each element. In these cases, the constraintset can be expressed as the convex combination of a finite set of verticies $\mathcal{M} = \text{Conv}\{\mathcal{A}\}$. This will be of pirticular relevance in the discussion of Away-step and pairwise varients. Otherwise, $\mathcal{A}$ is taken to be the boundry of $\mathcal{M}$. The constrained nature of this problem limits the applicabiliy of a method like gradient descent, and requires integrating knowledge of the constraint space for effective optimization. Methods like Fast Signed Gradient attacks and projected gradient descent are popular choices, but either create unsifisticated adversarial examples or require wasteful prejection onto the constraint space.

We explore Frank-Wolfe variants which are well suited to this problem by ensuring feasibility within the constraint set at each iteration with the efficient solving of a Linear Minimization Oracle (LMO).

$$LMO_{\mathcal{A}}(\nabla f(x_t)) \in \arg\min_{x \in \mathcal{A}} \langle \nabla f(x_t), x \rangle$$

The LMO is responsible for solving what is called the "Frank-Wolfe Subproblem" and at each iteration provides and optimal s_t such that updating x_{t+1} to move in the direction of s_t will remain in $\mathcal{M}$ Given it moves no more than some maximum stepsize. By solving the LMO efficiently, the algorithm ensures that each iteration makes significant progress towards the optimal solution while respecting the constraints, thereby maintaining feasibility and accelerating convergence. Efficient solving of the LMO requires exploiting the structure of the constraint set $\mathcal{M}$. In the case of the L_∞ norm it has closed form solution:

$$LMO_{\mathcal{A}} = s_t = -\epsilon \text{sign}(\nabla f(x_t)) + x_0$$

This is clearly of $O(n)$ complexity where n is the number of elements in the gradient. This can be intrepreted as defining an attack direction where each element is the maximum allowable perturbation $\pm\epsilon$ according to the gradient. With one iteration, this is exactly the outcome of a Fast Signed Gradient Attack. At each iteration, we can see that the LMO will give a vertex on the boundry of the constraint set $\mathcal{M}$ which was optimally chosen to be as close to the true gradient as possible while permitting subsequent iterations staty within the feasible set.

Algorithm 1 An algorithm with caption

Require: maximum iterations T, stepsizes $\{\gamma_t\}$, convergence tolerence δ
Ensure: $y = x^n$

1: $x_0 = x_{\text{ori}}$
2: **for** $t = 1, ..., T$ **do**
3: $\quad s_t = \arg\min_{x \in \mathcal{M}} \langle x, \nabla f(x_t) \rangle$ ▷ LMO step
4: $\quad d_t = s_t - x_t$
5: $\quad x_{t+1} = x_t + \gamma_t d_t$
6: $\quad$ **if** $\langle d_t, -\nabla f(x) \rangle < \delta$ **then** return ▷ FW gap convergence criterion
7: $\quad$ **end if**
8: **end for**

3 Algorithims

3.1 Frank-Wolfe

Observing oscilation in Frank Wolfe convergence is common and consequence of optimal points lying on a face of $\mathcal{M}$. Since at each iteration the method is moving twords a vetex of polytope $\mathcal{M}$, in $\mathcal{S}$, the method "zigzags", moving twords different points in effort to gradually approach the face on which the optimum lies. in the convex case, Frank Wolfe is seen to have linear complexity when optimal point x^* lies in the interior of $\mathcal{M}$, the osicilation causes sub-linear convergence when x^* on the boundry. We implement varients that aim to address this problem to provide better convergence. The simplest of which is adding momentum to standerd frank wolf which replaces the gradient in the LMO calulation in line (4) with a momentum term $m_t = \beta m_{t-1} + (1-\beta)\nabla f(x_t)$ and initialize $m_0 = \nabla f(x_0)$. By considering this exponentially weighted average of gradient information, momentum varients are emperically observed to have nicer convergence.

$$s_t = -\epsilon \text{ sign}(\nabla f(x)) + x_{\text{ori}}$$

The FW algorthim has a sueful intrepretation that serves as basis for the following varients: Away-Step FW and Pairwise FW. Namely that at each iteration x_t is perturbed in some direction s_t, making x_{t+1} a convex combination of x_t and s_t. We record these directions in an active set $S_{t+1} = S_t \bigcup \{s_t\}$ and observe that initially x_0 is a convex combination of of active set $S_0 = \{x_0\}$. Then by induction, x_t is a convex combination of directions in S_t, admitting coefficients *alpha* such that $\sum \alpha_{s_i} s_i = x_t$. Each iteration of the FW algorithim is seen to increase or introduce the contribution of s_t in the convex combination while shrink the α coefficients of all other verticies, or atoms uniformly. The innovation of the Away-Step and Pairwise varient is to recognize contribution of "bad atoms" can prevent convergence to an optimum on the boundry. These varients more directly diminish such atoms contributions by either taking steps away from selected atoms, or transfering mass between two selected atoms at each iteration as the case with the pairwise varient.

3.2 Away-Step Frank-Wolfe

Algorithm 2 Away-Step FW for Adversarial Attacks

Require: maximum iterations T, stepsizes $\{\gamma_t\}$, convergence tolerence δ, $x_0 \in \mathcal{M}$

Define $S_0 := \{x_0\}$ with $\alpha_{x_0} = 1$
for $t = 1, ..., T$ **do**
  $s_t := \arg\min_{x \in \mathcal{M}} \langle x, \nabla f(x_t) \rangle$ ▷ LMO step
  $d_t^{\mathrm{FW}} := s_t - x_t$
  $v_t := \arg\max_{v \in S_t} \langle v, \nabla f(x_t) \rangle$
  $d_t^{\mathrm{A}} := x_t - v_t$
  if $\langle d_t^{\mathrm{FW}}, -\nabla f(x) \rangle < \delta$ **then** return x_t ▷ FW gap convergence criterion
  end if
  if then$\langle d_t^{\mathrm{FW}}, -\nabla f(x) \rangle < \langle d_t^{\mathrm{A}}, -\nabla f(x) \rangle$
    $d_t = d_t^{\mathrm{FW}}$, $\gamma_{\max} := 1$
  else
    $d_t := d_t^{\mathrm{A}}$, $\gamma_{\max} := \frac{\alpha_{v_t}}{1 - \alpha_{v_t}}$
  end if
  $x_{t+1} = x_t + \gamma_t d_t$
  Update α, S_{t+1} s.t. $\langle \alpha, S_{t+1} \rangle = x_{t+1}$ (See below)
end for

3.3 Pairwise Frank-Wolfe

4 Results

Introduce Datasets

4.1 Momentum

4.2 Early-Stopping (Convergece Criterion)

It is worth noting that Convergence Criterion For Frank-Wolfe is a somewhat imprecise surrogate for success in the context of adversarial attacks. For many examples, we find that Frank-Wolfe methods create successful attacks several iterations before convergence. We attribute this to an incorrect class probability being grater than the correct class being sufficient for success where convergence is reached when the new output class probability is maximized. We observe the convergence of the Frank-Wolfe gap

4.3 Stepsize

The methods for stepsize were implemented as follows: Lipschitz constant-based (fixed) stepsize, where the stepsize is determined using the Lipschitz

constant L with $\gamma_t = \frac{1}{L}$; exact inesearching, which solves the optimization problem $\arg\min_\gamma f(x + \gamma d_t)$ where $\gamma \in (0, 1]$; Decaying stepsize, where the stepsize decreases over time according to $\gamma_t = \frac{2}{t+2}$; and Armijo-rule search, which chooses γ_t to satisfy the Armijo rule $f(x + \gamma_t d_t) \leq f(x) + \delta\gamma_t \nabla f(x)^T d_t$, where $delta \in (0, 1)$ controls the sufficient decrease condition.

4.4 ϵ Choice

Create plot showing how accurate attacks are with different ϵ constraints.

5 Convergence Analysis

The constrained nature of the Adversarial Attack problem means that the norm of the gradient $||\nabla_x f(x)||$ is not a suitable convergence criterion as boundary points need not have 0 gradient. The Frank-Wolfe gap provides provides measure of both optimality and point feasibility. It is a measure of the maximum improvement over the current iteration x_t within the constraints C and defined in terms of the FW direction:

$$g(x_t) = \max_{x \in C} \langle x - x_t, -\nabla f(x_t) \rangle$$

We always have $g(x_t) \geq 0$ and its usefulness as a convergence criterion comes from $g(x_t) = 0$ iff x_t is a stationary point. For convex problems, we would have that the linear approximation $f(x_t) + \langle x_t - x, -\nabla f(x_t) \rangle \geq f(x)$. However, the loss of DNNs as commonly the subject of adversarial attacks, are highly non-convex, making this only true locally. This complicate the convergence of Frank-Wolfe in this application, as we are gaurenteed not a global optimum or a sucessful attack, but convergence to a stationary point.

For the following proof we assume that f has L-Lipschitz continuous gradient on M. This is often the case for DNN Lipshitz continuious gradient gives us bound on curvature constant C_f

5.1 Frank-Wolfe

5.2 Pairwise Frank-Wolfe

5.3 Away-Step Frank-Wolfe