Adversarial robustness with an upper-bound on the norm of the Jacobian matrix

1 Introduction

For the last few years, the machine learning community has started to study the robustness problems of machine learning models, and of neural networks in particular. This study was motivated by the high sensitivity of neural networks to adversarial attacks, i.e., small perturbations that are able to fool a network. Adversarial attacks has been shown to be both ubiquitous and transferable. Beyond the obvious security threat, adversarial attacks are the symptom of the dramatic lack of robustness of neural networks. We believe that understanding and mitigating adversarial attacks will solve a large portion of the robustness issue of neural networks. Nonetheless, it must be pointed out that adversarial attacks, which belongs to the larger category of evasion attacks, are not the only robustness vulnerability of neural networks. Let us mention sensitivity to noise, ability to detect out-of-distribution data, quantification of the uncertainty in order to avoid model overconfidence, data poisoning, byzantine attacks, and model extraction, to name just a few.

In this document, a regularization method for adversarial robustness is presented. This method consists in constraining the information ball to contain the Euclidean ball such that a perturbation bounded in l_2 norm is also bounded in information distance, and this bound is controlled. This condition is enforced by bounding the spectral norm of the Jacobian matrix with respect to the input. Thus, the focus of this method is l_2 white-box attacks against multiclass classification tasks but it may be extended to more general settings (unrestricted attacks, black-box attacks etc.). In section 6, we discuss how this method can be extended to other supervised learning tasks. The method is evaluated on several image datasets, and using several state-of-the-art attacks. We pay a special attention to the computational efficiency of the method since we hope that it could be used in real-world applications.

The goal of this approach is not only to develop a new adversarial defense, but also to better understand the adversarial vulnerability phenomenon in deep learning models. Two questions are of particular interest: is there really a trade-off between accuracy and robustness? And is it possible to provide a certified defense in terms of both accuracy and robustness? This is a required step for the deployment of deep learning models in critical domains where certification is mandatory.

The remaining of this document is divided into five parts. Section 2 introduces the notations and describes the information geometric framework for the study of adversarial robustness. Section 3 presents an optimal condition for adversarial robustness at a point and then derives the Jacobian regularization method that approximates this condition. Section 4 presents several experiments to evaluate the proposed method. Section 5 discusses the results in the lights of related work on adversarial defense. We suggests some directions to answer the questions asked in introduction and potential extensions of this work. Finally, section 6 concludes the document.

2 Information geometric framework

2.1 Multiclass classification and family of categorical distributions

Let $\mathcal{Y} = \{1, \dots, m+1\} \subset \mathbb{N}$ be the set of labels for the classification task (hence there are $m+1$ different classes). Let $\mathcal{X} \subseteq \mathbb{R}^n$ be the **input domain**. We assume than $n > m$. For example, in MNIST we have $\mathcal{X} = [0,1]^n$ (with $n = 784$) since an admissible image must have its pixel's values between 0 and 1, and $m = 9$. In the following, "smooth" means C^∞. We assume that $\mathcal{X}$ is an n-dimensional embedded smooth connected submanifold of $\mathbb{R}^n$. For simplicity, we can think that $\mathcal{X} = \mathbb{R}^n$. A machine learning model (e.g., a neural network) is often seen as assigning a label $y \in \mathcal{Y}$ to a given input $x \in \mathcal{X}$. Instead, in this document, we see a model as assigning the *parameters* of a random variable Y to a given input $x \in \mathcal{X}$. Let us formalize this. We ask that the random variable Y verifies the following assumptions:

1. Let us write $Y : (\Omega, \mathcal{F}, \mathbb{P}) \to ([0, m+1], \mathcal{B}([0, m+1]))$ with $(\Omega, \mathcal{F}, \mathbb{P})$ some probability space and $\mathcal{B}([0, m+1])$ the Borel σ-algebra of $[0, m+1] \subset \mathbb{R}$. Then, we assume that $\mathbb{P}(Y^{-1}(\mathcal{Y})) = 1$.

2. Y is absolutely continuous with respect to the counting measure on $[0, m+1]$. In particular, Y has a probability density function (pdf) with respect to the counting measure.

3. The pdf of Y belongs to a **parameterized family of pdf** $\mathcal{S} = \{p_\theta : \theta \in \Delta^m\}$ (more details below).

We will denote the components of a vector v as v^i with a superscript. The set Δ^m is defined as $\Delta^m = \{\theta \in \mathbb{R}^{m+1} : \sum_{i=1}^{m+1} \theta^i = 1, 0 < \theta^i < 1\}$. It is called the **probability m-simplex**. The inequality $0 < \theta^i < 1$ is strict in order to discard the boundary of the simplex, because the Fisher information metric introduced in the next subsection is not defined on the boundary of Δ^m.

Let $\theta \in \Delta^m$. The pdf $p_\theta : [0, m+1] \to \mathbb{R}^+$ is defined as:

$$p_\theta(y) = \prod_{i=1}^{m+1} (\theta^i)^{\delta_i(y)} = \exp\left(\sum_{i=1}^{m+1} \delta_i(y) \ln \theta^i\right),$$

where $\delta_i(y) = 1$ if $y = i$ and 0 otherwise. Thus, the family $\mathcal{S}$ is the **family of $(m+1)$-dimensional categorical distributions**. $\mathcal{S}$ can be endowed with a differentiable structure by using $p_\theta \in \mathcal{S} \mapsto (\theta^1, \ldots, \theta^m) \in \mathbb{R}^m$ as a global coordinate system. Hence, $\mathcal{S}$ becomes a smooth manifold of dimension m (more details on this construction can be found in Amari 1985 [1] Chapter 2). We can identify p_θ with $(\theta^1, \ldots, \theta^m)$.

In the following, we will note $p_\theta(y) = p(y, \theta)$ where $p : [0, m+1] \times \Delta^m \to \mathbb{R}^+$ is seen as a function of both y and θ. Note that for any $y \in \mathbb{R}$, the function $p(y, \cdot) : \Delta^m \to \mathbb{R}$ is smooth. We will often use the abuse of notation that consists in writing "the distribution $p(y, \theta)$" instead of "the distribution of Y characterized by the pdf $y \mapsto p(y, \theta)$".

Finally, we can define what we mean by "model". We call "**model**" any smooth map $F : \mathcal{X} \to \Delta^m$, that assigns to an input $x \in X$ the parameters $\theta = F(x) \in \Delta^m$ of a $(m+1)$-dimensional categorical distribution $p(y, \theta) \in \mathcal{S}$. In practice, a neural network produces a vector of *logits* $s(x)$. Then, these logits are transformed into the parameters θ with the softmax function: $\theta = \mathrm{softmax}(s(x))$.

2.2 Riemannian metrics

Let F be a model. In order to study the sensitivity of the predicted $F(x) \in \Delta^m$ with respect to the input $x \in \mathcal{X}$, we need to be able to measure distances both in $\mathcal{X}$ and in Δ^m. In order to measure distances on smooth manifolds, we need to equip each manifold with a Riemannian metric. Formally, a Riemannian metric is a covariant tensor field of rank 2 which is positive-definite. Intuitively, a Riemannian metric defines an inner product of on each tangent space, and this inner product changes smoothly when we move smoothly from one tangent space to another. With this inner product, we can compute angles and norms of tangent vectors, which can be seen as "infinitesimal" vectors. By integrating these "infinitesimal" lengths along a curve in the manifold, we can compute the length of the curve. The distance between two points can thus be defined as the infimum of the length of all the curves joining the two points.

2.2.1 The Euclidean metric

Let us begin with $\mathcal{X}$. Since we are studying adversarial robustness, we need a metric that formalizes the idea that two close data points must be "indistinguishable" from a human perspective (or any other relevant perspective). A natural choice is the **Euclidean metric**. Using the standard coordinate of $\mathbb{R}^n$ as a global coordinate system for $\mathcal{X}$, the Euclidean metric is defined as

$$\overline{g}_x = \sum_{i,j=1}^{n} \delta_{ij} dx^i dx^j,$$

where $\delta_{ij} = 1$ if $i = j$ and 0 otherwise. The Euclidean metric corresponds to the l_2 norm. Please note that there are other "dissimilarity" measures that can be used to study adversarial robustness, including all other l_p norms. In particular, the l_∞ norm is often considered to be the most natural choice when working on images. However, the l_∞ norm is not induced by any inner product, and hence, there is no Riemannian metric that induces the l_∞ norm. However, given a l_∞ budget ϵ_∞, we can choose a l_2 budget $\epsilon_2 = \sqrt{n}\epsilon_\infty$ such that any attack in the ϵ_∞ budget will also respect the ϵ_2 budget. When working on images, other dissimilarity measures are rotations, deformations, or color changes of the original image. Contrary to the l_2 or l_∞, these measures are not based on a pixel-based coordinate system. It may be possible to find a coordinate change on $\mathcal{X}$ such that these measures correspond to the l_2 norm in the new coordinate system, or to use another metric. Anyway, in the rest of this document, we assume that $(\mathcal{X}, \overline{g})$ is a Riemannian manifold equipped with the Euclidean metric.

2.2.2 The Fisher information metric

Now, we consider Δ^m. The manifold Δ^m can also be equipped with the Euclidean metric induced from $\mathbb{R}^{m+1}$. However, as described above, we see Δ^m as the family of categorical distributions. In order to find a more natural Riemannian metric for Δ^m, i.e., a metric that reflects the statistical properties of Δ^m, let us consider the loss function used to train a multiclass classification model: the **relative entropy**, also known as Kullback-Leibler divergence. Let θ and ϕ be two points of Δ^m. Then, the relative entropy is:

$$\begin{aligned} H\left(\phi || \theta\right) &= -\mathbb{E}_\phi \left[\ln \frac{\theta}{\phi}\right], \\ &= \sum_{i=1}^{m+1} \phi^i \ln \frac{\phi^i}{\theta^i}. \end{aligned}$$

When training a model, $\theta = F(x)$ is the distribution predicted by the model, while ϕ is the true distribution. The relative entropy can be rewritten $H(\phi||\theta) = -\sum_{i=1}^{m+1} \phi^i \ln \theta^i + \sum_{i=1}^{m+1} \phi^i \ln \phi^i = H(\phi, \theta) - H(\phi)$ where $H(\phi, \theta)$ is the cross entropy and $H(\phi)$ is the entropy of ϕ. In practice, only the cross entropy term is used as a loss function since $H(\phi)$ does not depend on θ.

More generally, we can define a large family of divergences by:

$$H_{r,f}\left(\phi||\theta\right) = \frac{1}{r-1} f\left(\sum_{i=1}^{m+1} (\phi^i)^r (\theta^i)^{1-r}\right),$$

where $r \geq 0$, $r \neq 1$, and f is a C^1 function such that $f(1) = 0$ and $f'(1) = 1$. If we choose $f = \ln$, we obtain the Rényi divergences. Then, it can be shown that:

$$H_{r,\ln}\left(\phi||\theta\right) \xrightarrow[r\to 1]{} H\left(\phi||\theta\right).$$

Now, consider the second-order Taylor approximation of $H_{r,f}\left(\theta||\theta + \delta\theta\right)$ with respect to $\delta\theta$, where $\theta + \delta\theta \in \Delta^m$. For a given $\theta \in \Delta^m$, let g be defined by:

$$g(\delta\theta) = H_{r,f}\left(\theta||\theta + \delta\theta\right) = \frac{1}{r-1} f\left(\sum_{i=1}^{m+1} (\theta^i)^r (\theta^i + \delta\theta^i)^{1-r}\right).$$

The second-order Taylor approximation of g around 0 is:

$$g(\delta\theta) = g(0) + (\nabla g(0))^T \delta\theta + \frac{1}{2} (\delta\theta)^T \nabla^2 g(0) \delta\theta + o\left(\|\delta\theta\|^2\right). \tag{1}$$

The kth component of the gradient ∇g verifies:

$$\nabla g(\delta\theta)_k = -f'\left(\sum_{i=1}^{m+1} (\theta^i)^r (\theta^i + \delta\theta^i)^{1-r}\right) \left(\frac{\theta^k}{\theta^k + \delta\theta^k}\right)^r,$$

such that $\nabla g(0)_k = -f'(1) = 1$ for all $1 \leq k \leq m+1$. Thus, $(\nabla g(0))^T \delta\theta = -\sum_{i=1}^{m+1} \delta\theta^i = 0$ since $\theta \in \Delta^m$ and $\theta + \delta\theta \in \Delta^m$. The k, l entry of the Hessian matrix $\nabla^2 g$ verifies:

$$\begin{aligned}\nabla^2 g(\delta\theta)_{k,l} =& (r-1) f'' \left(\sum_{i=1}^{m+1} (\theta^i)^r (\theta^i + \delta\theta^i)^{1-r} \right) \left(\frac{\theta^k \theta^l}{(\theta^k + \delta\theta^k)(\theta^l + \delta\theta^l)} \right)^r \\ &+ r f' \left(\sum_{i=1}^{m+1} (\theta^i)^r (\theta^i + \delta\theta^i)^{1-r} \right) \left(\frac{\theta^k}{\theta^k + \delta\theta^k} \right)^r \frac{\delta_{kl}}{\theta^k + \delta\theta^k},\end{aligned}$$

such that $\nabla^2 g(0)_{k,l} = (r-1) f''(1) + \frac{r}{\theta^k} \delta_{kl}$. Thus,

$$(\delta\theta)^T \nabla^2 g(0) \delta\theta = (r-1) f''(1) \left(\sum_{i=1}^{m+1} \delta\theta^i \right)^2 + \sum_{i=1}^{m+1} \frac{r}{\theta^i} \delta_{ij} = \sum_{i=1}^{m+1} \frac{r}{\theta^i} \delta_{ij},$$

since $\sum_{i=1}^{m+1} \delta\theta^i = 0$. Finally, by injecting these results in equation 1, we obtain:

$$H_{r,f} \left(\theta || \theta + \delta\theta \right) = \frac{r}{2} \sum_{i=1}^{m+1} \frac{1}{\theta^i} \left(\delta\theta^i \right)^2 + o \left(\|\delta\theta\|^2 \right). \tag{2}$$

Surprisingly, equation 2 does not depend on f. When $\delta\theta$ is small, $H_{r,f} \left(\theta || \theta + \delta\theta \right)$ looks almost like a *squared* Euclidean distance but with an additional coefficient $\frac{r}{2\phi^i}$. Let $T_\theta \Delta^m$ be the tangent space of Δ^m at θ. Let $X, Y \in T_\theta \Delta^m$ be two tangent vectors at θ. Equation 2 suggests that $H_{r,f}$ is an infinitesimal squared distance. If we discard the multiplicative constant $\frac{r}{2}$, the corresponding norm is:

$$\|X\|^2 = \sum_{i=1}^{m+1} \frac{1}{\theta^i} X^i,$$

and the inner product is:

$$\langle X, Y \rangle = \sum_{i=1}^{m+1} \frac{1}{\theta^i} X^i Y^i.$$

We have thus defined a Riemannian metric on Δ^m:

$$g_\theta^{FIM} = \sum_{i,j=1}^{m+1} \frac{1}{\theta^i} \delta_{ij} d\theta^i d\theta^j.$$

The metric g is called the **Fisher information metric** (FIM). Indeed, if we consider the Fisher information defined from the Cramér-Rao bound:

$$\mathcal{I}(\theta) = \mathbb{E}_{y \sim p_\theta} [\nabla_\theta \ln p_\theta(y) \nabla_\theta \ln p_\theta(y)^T],$$

then an easy calculation shows that $\mathcal{I}(\theta) = g_\theta^{FIM}$.

We have already seen that the FIM is the infinitesimal metric associated to a large family of divergences. Another remarkable property of the FIM is Chentsov's theorem that claims that the FIM is the *unique* Riemannian metric on Δ^m that is invariant under sufficient statistics (up to a multiplicative constant). To summarize, the FIM is invariant when we deform the relative entropy, and it is invariant when we deform $\mathcal{Y}$ with a sufficient statistic. Moreover, the FIM tells us how our loss function changes when we move in Δ^m. For all those reasons, we endow Δ^m with the FIM g^{FIM}, making it a Riemannian manifold (Δ^m, g^{FIM}).

The reader may have noticed that we have defined the FIM using the parameter $\theta \in \mathbb{R}^{m+1}$. However, we also stated that Δ^m is a m-dimensional smooth manifold. This means that $\theta \in \mathbb{R}^{m+1}$ is not a coordinate system on Δ^m, while $(\theta^1, \ldots, \theta^m) \in \mathbb{R}^m$ is such a coordinate system. Hence, we have defined the FIM on $\mathbb{R}^{m+1}$ by seeing Δ^m as an embedded submanifold of $\mathbb{R}^{m+1}$. In order to see Δ^m as a manifold in its own right, we must defined the FIM with respect to a coordinate system. Let g be the FIM expressed in the coordinate system $(\theta^1, \ldots, \theta^m)$. In these coordinates, define the inclusion function $i : \Delta^m \to \mathbb{R}^{m+1}$ by

$i(\theta^1, \dots, \theta^m) = (\theta^1, \dots, \theta^m, 1 - \sum_{k=1}^m \theta^k)$. Then, g is the pullback of g^{FIM} by the inclusion function i, i.e., $g = i^* g^{FIM}$. It can easily be shown that:

$$g_\theta = \sum_{i,j=1}^{m} \left(\frac{1}{\theta^i} \delta_{ij} + \frac{1}{1 - \sum_{k=1}^m \theta^k} \right) d\theta^i d\theta^j.$$

Let $\theta^{m+1} = 1 - \sum_{k=1}^m \theta^k$. Then:

$$g_\theta = \sum_{i,j=1}^{m} \left(\frac{1}{\theta^i} \delta_{ij} + \frac{1}{\theta^{m+1}} \right) d\theta^i d\theta^j. \tag{3}$$

In the rest of this document, we use the FIM on Δ^m as defined in equation 3.

2.2.3 The pullback of the FIM

Now, let $F : \mathcal{X} \to \Delta^m$ be a model. Consider the pullback metric $\tilde{g}$ of the FIM by F, i.e. $\tilde{g} = F^* g$. Using the standard coordinates on $\mathcal{X}$ and the coordinates $(\theta^1, \dots, \theta^m)$ on Δ^m, we have for any $x \in \mathcal{X}$:

$$\tilde{g}_x = J_x^T g_{F(x)} J_x, \tag{4}$$

where J_x is the Jacobian matrix of F at x.

3 Jacobian regularization

In this section, we consider a model $F : \mathcal{X} \to \Delta^m$, a point $x \in \mathcal{X}$ and we denote $\theta = F(x) \in \Delta^m$.

3.1 Distance to a decision boundary in the probability simplex

Consider the diffeomorphism $T_1 : \Delta^m \to \Delta^m$ defined by:

$$\mu = T_1(\theta) = 2\sqrt{\theta}, \tag{5}$$

where the square root is applied element-wise. Let $\mu^{m+1} = 2\sqrt{1 - \frac{1}{4}\sum_{k=1}^m (\mu^i)^2}$. Then, we can compute g is these new coordinates:

$$g_\mu = \sum_{i,j=1}^{m} \left(\delta_{ij} + \frac{\mu^i \mu^j}{(\mu^{m+1})^2} \right) d\mu^i d\mu^j.$$

This expression of g is the same as the expression of the Euclidean metric induced on the sphere of radius 2 in the standard coordinates. In other words, Δ^m can be isometrically embedded into a m-sphere of radius 2. The angle β between two distributions of coordinates θ_1 and θ_2, with $\mu_1 = T_1(\theta_1)$ and $\mu_2 = T_2(\theta_2)$ is:

$$\cos(\beta) = \frac{1}{4} \sum_{i=1}^{m+1} \mu_1^i \mu_2^i = \sum_{i=1}^{m+1} \sqrt{\theta_1^i \theta_2^i}.$$

The Riemannian distance between these two points is the arc length on the sphere:

$$d(\theta_1, \theta_2) = 2 \arccos \sum_{i=1}^{m+1} \sqrt{\theta_1^i \theta_2^i}.$$

Now, consider the decision boundary in Δ^m defined as the set $\mathcal{B}$ such that $\theta \in \mathcal{B}$ if the maximum of θ is achieved at two different components, i.e., $\max_{1 \leq k \leq m+1} \theta^k = \theta^i = \theta^j$ for some $i \neq j$. We are interested in the distance $d(F(x), \mathcal{B})$ between $F(x)$ and the decision boundary $\mathcal{B}$. Instead of deriving an exact formula, we use the following upper bound:

$$d(F(x), \mathcal{B}) \leq d(F(x), c),$$

where $c = \frac{1}{m+1}(1, \dots, 1)$ is the center of the simplex Δ^m. Let us denote $\delta(x) = d(F(x), c)$. Thus:

$$\delta(x) = 2 \arccos \sum_{i=1}^{m+1} \sqrt{\frac{F(x)^i}{m+1}}. \tag{6}$$

3.2 Adversarial robustness in the neighborhood of a training point

Consider the Euclidean open ball[1] $\mathcal{B}(x,\epsilon) = \{z \in \mathcal{X} : \|z - x\| < \epsilon\} \subset \mathcal{X}$ where $\|.\|$ is the Euclidean norm (i.e., the l_2 norm). The radius $\epsilon > 0$ is chosen such that all points in $\mathcal{B}(x,\epsilon)$ are considered to be indistinguishable from x (according to some relevant perspective), while every point outside $\mathcal{B}(x,\epsilon)$ can be distinguished from x.

Let $\exp_x : T_x\mathcal{X} \to \mathcal{X}$ be the exponential map[2] induced by the pullback of the FIM $\tilde{g}$. For simplicity, we assume that $\exp_x$ is defined[3] on the entire tangent space $T_x\mathcal{X}$. Let $X \in T_x\mathcal{X}$ be a tangent vector. The distance between x and $\exp_x(X)$ according to $\tilde{g}$ is the same as the distance between $F(x)$ and $F(\exp_x(X))$ in Δ^m. This distance is equal to $\sqrt{\tilde{g}_x(X,X)}$. Let $\widetilde{B}(0,\delta) = \{X \in T_x\mathcal{X} : \sqrt{\tilde{g}_x(X,X)} < \delta\} \subset T_x\mathcal{X}$. The **geodesic ball** $\widetilde{\mathcal{B}}(x,\delta)$ is defined[4] as:

$$\widetilde{\mathcal{B}}(x,\delta) = \{\exp_x(X) \in \mathcal{X} : X \in \widetilde{B}(0,\delta)\} \subset \mathcal{X}.$$

We will say that the model F is **adversarially robust** at x if every point in $\mathcal{B}(x,\epsilon)$ has the same class, i.e., if $\mathcal{B}(x,\epsilon)$ does not intersect any decision boundary. Hence, F **is adversarially robust at** x **if and only if:**

$$\mathcal{B}(x,\epsilon) \subseteq \widetilde{\mathcal{B}}(x, d(F(x),\mathcal{B})), \tag{7}$$

In this document, we focus on the following weaker condition:

$$\mathcal{B}(x,\epsilon) \subseteq \widetilde{\mathcal{B}}(x,\delta(x)), \tag{8}$$

where $\delta(x)$ was introduced in equation 6.

We need to better understand what the assumption of zero curvature implies.

The exact shape of the geodesic ball $\widetilde{\mathcal{B}}(x,\delta)$ depends on the curvature of $\tilde{g}$. In this document, we neglect the curvature of $\tilde{g}$ in $\widetilde{\mathcal{B}}(x,\delta(x))$. In other words, we assume that $(\mathcal{X},\tilde{g})$ is flat in $\widetilde{\mathcal{B}}(x,\delta(x))$. Let $B(0,\epsilon) = \{X \in T_x\mathcal{X} : \sqrt{\overline{g}(X,X)} < \epsilon\}$ be the Euclidean open ball in $T_x\mathcal{X}$. Under the assumption that the curvature is zero in $\widetilde{\mathcal{B}}(x,\delta(x))$, the condition expressed in equation 8 is equivalent to:

$$B(0,\epsilon) \subseteq \widetilde{B}(0,\delta(x)). \tag{9}$$

Equation 9 is implied by the following stronger condition on the metrics:

$$\tilde{g}_x(X,X) \le \frac{\delta(x)^2}{\epsilon^2}\overline{g}_x(X,X), \text{ for all } X \in T_x\mathcal{X}. \tag{10}$$

To summarize, there are three simplifications between the optimal robustness condition (equation 7) and equation 10:

1. $\delta(x)$ is used instead of $d(F(x),\mathcal{B})$.
2. The curvature of $\tilde{g}$ is assumed to be zero in $\widetilde{\mathcal{B}}(x,\delta)$.
3. Equation 10 is stronger than equation 9.

If we assume that equation 10 holds at x, then any successful adversarial attack is either due to simplification 1 or simplification 2. On the other hand, simplification 3 may harm the accuracy of F more than what was necessary to ensure the robustness of F: the trade-off between robustness and accuracy is not optimal.

[1] If $\mathcal{B}(x,\epsilon) \subset \mathbb{R}^n$ does not lie entirely in $\mathcal{X}$, consider the intersection between $\mathcal{B}(x,\epsilon)$ and $\mathcal{X}$.

[2] For $X \in T_x\mathcal{X}$, the point $\exp_x(X) \in \mathcal{X}$ is defined as the point reached by following the geodesic starting at x with initial velocity X during a time interval of 1.

[3] If $\exp_x$ were defined on a subset $U \subset T_x\mathcal{X}$, we assume that U is large enough for the following arguments to hold.

[4] For the geodesic ball to be well defined, $\exp_x$ must be a diffeomorphism on $\widetilde{B}(0,\delta)$. In the following, we assume that this is true.

3.3 Condition on the spectral norm of the Jacobian matrix

Let $\widetilde{G}_x$ be the matrix of $\tilde{g}$ in the standard coordinates of $\mathcal{X} \subseteq \mathbb{R}^n$. In these coordinates, the matrix of $\overline{g}$ is the identity matrix I_n. Equation 10 becomes:

$$X^T \widetilde{G}_x X \leq \frac{\delta(x)^2}{\epsilon^2} X^T X, \text{ for all } X \in \mathbb{R}^n \approx T_x\mathcal{X},$$

which can be rewritten using equation 4:

$$\frac{\delta(x)^2}{\epsilon^2} I_n - \widetilde{G}_x \succeq 0, \tag{11}$$

$$\frac{\delta(x)^2}{\epsilon^2} I_n - J_x^T G_{F(x)} J_x \succeq 0, \tag{12}$$

where the symbol "$\succeq$" means "positive semi-definite" and $G_{F(x)}$ is the matrix of $g_{F(x)}$ in the coordinates $(\theta^1, \dots, \theta^m)$.

3.4 Coordinate change to obtain a diagonal matrix

Consider a first coordinate change $\mu = T_1(\theta)$ where T_1 was defined in equation 5. As mentioned in paragraph 3.1, Δ^m can be isometrically embedded into the m-sphere of radius 2 using the coordinates $(\mu^1, \dots, \mu^m)$.

Now, consider the stereographic projection $T_2 : \Delta^m \to \Delta^m$ defined by:

$$t = T_2(\mu) = \frac{2\mu}{2 - \mu^{m+1}},$$

where the expression is applied element-wise. Let $\|t\|^2 = \sum_{k=1}^m (t^k)^2$. Then, the expression of g is these new coordinates is:

$$g_t = \sum_{i,j=1}^m \frac{4\delta_{ij}}{\left(\|\frac{t}{2}\|^2 + 1\right)^2} dt^i dt^j,$$

which is diagonal for all $x \in \mathcal{X}$. Let $\tilde{J}$ be the Jacobian matrix of $T_2 \circ T_1 \circ F$. Equation 12 translates into:

$$\frac{\delta(x)^2}{\epsilon^2} I_n - \rho(x)^2 \tilde{J}_x^T \tilde{J}_x \succeq 0, \tag{13}$$

where $\rho(x) = \frac{2}{\|\frac{t}{2}\|^2 + 1}$ which can be rewritten as:

$$\rho(x) = \frac{2\left(1 - \sqrt{F^{m+1}(x)}\right) - \|F(x)\|_1}{1 - \sqrt{F^{m+1}(x)}},$$

with $\|F(x)\|_1 = \sum_{i=1}^m F^i(x)$, and F^i is the i-th component of F in the coordinates $(\theta^1, \dots, \theta^m)$. Equation 13 is equivalent to $\lambda_{max} \leq \left(\frac{\delta(x)}{\rho(x)\epsilon}\right)^2$ where λ_{max} is the largest eigenvalue of $\tilde{J}_x^T \tilde{J}_x$. Finally, we can rewrite this condition as:

$$\|\tilde{J}_x\|_2 \leq \frac{\delta(x)}{\rho(x)\epsilon}, \tag{14}$$

where $\|\tilde{J}_x\|_2$ is the spectral norm of the Jacobian matrix $\tilde{J}_x$ i.e., the largest singular value of $\tilde{J}_x$.

3.5 A regularization term to enforce the Jacobian norm condition

We need to chose between Lanczos algorithm (or another algorithm that approximate the largest singular value) and the Frobenius norm.

It is not possible to efficiently compute the largest singular value of $\tilde{J}_x$. However, it can be approximated with an iterative algorithm such as the Lanczos algorithm (applied on $\tilde{J}_x^T \tilde{J}_x$). Another approach consists in using an upper-bound of the spectral norm that is easier to compute. Here, we propose to use the Frobenius norm $\|\cdot\|_F$ since $\|\tilde{J}_x\|_2 \leq \|\tilde{J}_x\|_F$.

Now, we can define a regularization term:

$$\alpha(x, F) = \exp\left(\|\tilde{J}_x\|_F - \frac{\delta(x)}{\rho(x)\epsilon}\right). \tag{15}$$

We use the exponential function in order to have the following properties: $\alpha(x, F) \approx 0$ when equation 14 holds and $\alpha(x, F) \to +\infty$ when equation 14 is violated. Moreover, since the exponential has nonzero derivative, it will inform the model about how close is it to violate equation 14. To compute $\alpha(x, F)$, we only need to compute the Jacobian matrix $\tilde{J}_x$ which can be efficiently achieved with backpropagation. The regularized loss function is:

$$\mathcal{L}(\hat{\theta}, x, F) = (1-\eta)H(\hat{\theta}||F(x)) + \eta\alpha(x, F), \tag{16}$$

where η is a hyperparameter to control the trade-off between the relative entropy and the regularization term. In practice, using the loss $\mathcal{L}$ to train a model will approximately enforce the robustness condition 9 at each training point in expectation. If we come back to the list of simplifications presented at the end of paragraph 3.2, three new simplifications have been added. We reproduce here the six simplifications that prevent the proposed method to be optimally robust:

1. $\delta(x)$ is used instead of $d(F(x), \mathcal{B})$.
2. The curvature of $\tilde{g}$ is assumed to be zero in $\widetilde{\mathcal{B}}(x, \delta)$.
3. Equation 10 is stronger than equation 9. This will harm the accuracy but not the robustness.
4. The Frobenius norm induces a stronger condition than the spectral norm. This will harm the accuracy but not the robustness.
5. If the point $x \in \mathcal{X}$ is not a training point, there is no guarantee that the robustness condition will be enforced. This is a generalization issue.
6. Since we are using a regularization term, we cannot ensure that the constraint will not be violated at some training points.

Simplification 2 may be solved using Jacobi field. Simplification 4 may be partially solved with Lanczos algorithm. Simplification 5 may be solved with constraints on the global Lipschitz constant. Simplification 6 may be solved using techniques from constrained optimization.

4 Experiments

4.1 Toy model

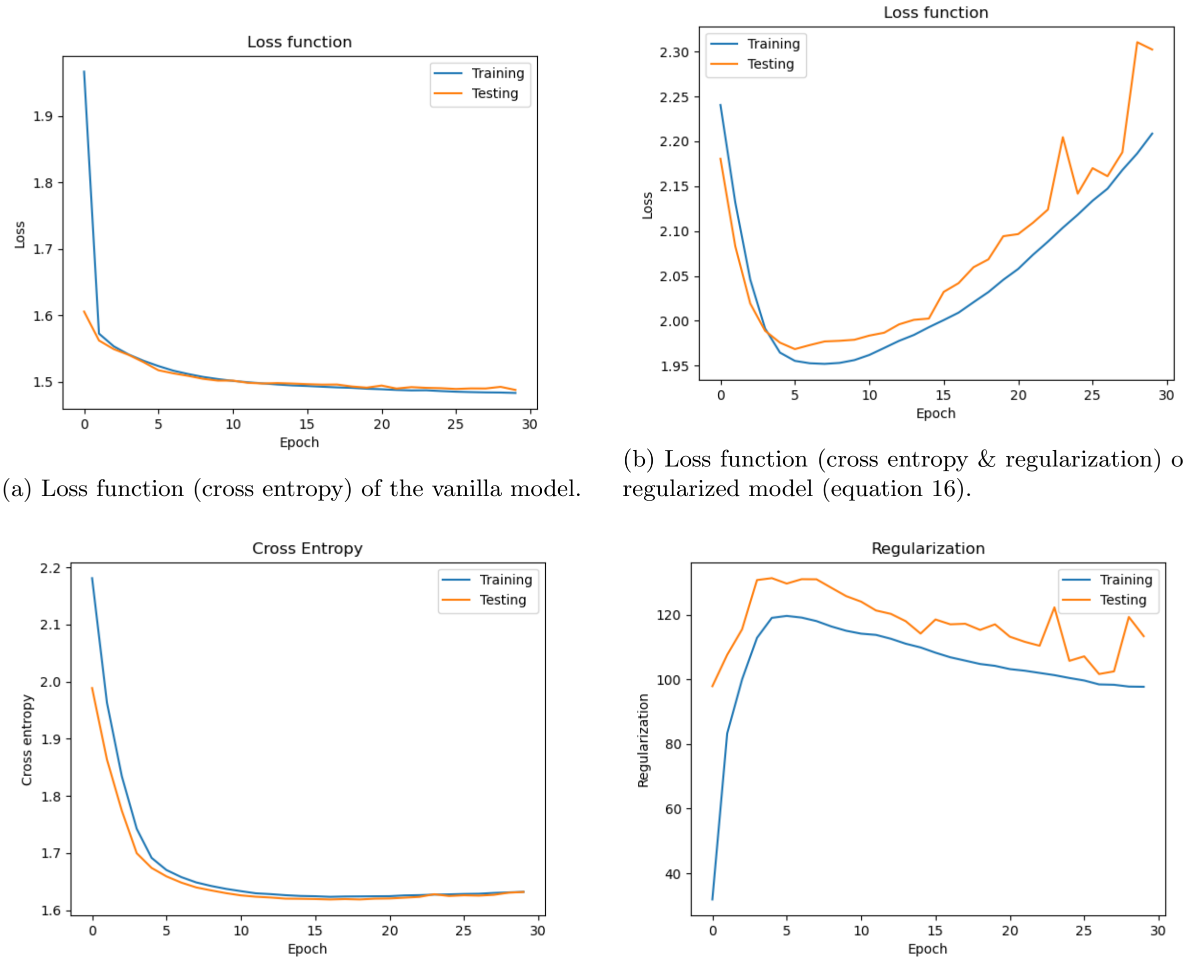
4.2 First results on MNIST

To be done

We present preliminary results obtained on MNIST for the regularization method introduced in section 3. We implement a simple LeNet model with two convolutional layers of 32 and 64 channels respectively, followed by one hidden layer with 128 neurons. We train two models, one with the regularization, and one without. Both models are trained with vanilla SGD on 30 epochs, with batch size 32 and learning rate 0.01. For the regularization, we choose ϵ such that $\frac{\delta_m}{\epsilon} = 10^{-3}$. Instead of choosing a fixed η, we increase η at each epoch of the training according to the following rule:

$$\eta_i = \eta_{min}\left(\frac{\eta_{max}}{\eta_{min}}\right)^{(i-1)/(N_{epoch}-1)},$$

where N_{epoch} is the total number of epochs and $1 \leq i \leq N_{epoch}$ is the current epoch. We chose $\eta_{min} = 0.002$ and $\eta_{max} = 0.006$.



(a) Loss function (cross entropy) of the vanilla model.

(b) Loss function (cross entropy & regularization) of the regularized model (equation 16).

(c) Cross entropy of the regularized model alone.

(d) Regularization of the regularized model alone.

Figure 1: Loss functions of the vanilla model and the regularized model (equation 15).

The models are trained on the 60000 images of MNIST's training set, then tested on the 10000 images of the test set. **The vanilla model achieves an accuracy of 97% (9740/10000) while the regularized model achieves an accuracy of 88% (8829/10000)**. It must be mentioned that, with the current implementation, the regularized model is still almost **12 times slower** to train than the vanilla model. However, it may be possible to accelerate the training using for example the technique proposed by Shafahi et al. [2].

Figure 1 presents the loss functions of both models as well as the cross entropy and the regularization term (equation 15) of the regularized model alone. We can see that the total loss function of the regularized model (figure 1b) decreases until epoch 7 and then increases. This is due to the increase of η. Indeed, we see in figure 1d that the regularization of the initial (random) model is very low and then increases dramatically when the model starts to learn. When η increases, the model is forced to reduce the regularization term but seems to struggle at achieve a large decrease. It may achieve a lower regularization with longer training. We also tried to train the model with a fixed $\eta = 0.005$. With this setting, the model immediately minimize the regularization but is unable to reduce the cross entropy and thus achieves low accuracy.

To measure the adversarial robustness of both models, we use the PGD attack with the l_∞ norm, 40 iterations, and a step size of 0.01. We chose to use the l_∞ norm instead of the l_2 in order to use the hardest possible attack for our method, and because the l_∞ norm corresponds more to the human notion of "indistinguishable images" than the l_2 norm. The attacks are performed on the test set, and only on images

Budget Model	0.1	0.2	0.3
Vanilla (9740)	4138 (42%)	428 (4%)	103 (1%)
Regularized (8829)	8040 (91%)	3708 (42%)	211 (2%)

Table 1: Adversarial robustness of the vanilla and regularized models for various attack budgets. The reported values are the number of perturbed images that were **correctly classified** (i.e., same class as the original image). Next to each model's name, we report the total number of correctly classified original images.

that were correctly classified by each model. The results are reported in table 1. These preliminary results seems to indicate that the regularization proposed in equation 15 improves the adversarial robustness, except for high attack budget. It must be mentioned that a l_∞ budget of 0.3 on MNIST can clearly be noticed with the naked eye.

5 Related work and discussion

6 Conclusion

References

[1] S.-i. Amari, *Differential-Geometrical Methods in Statistics*, vol. 28 of *Lecture Notes in Statistics*. Springer New York, 1985.

[2] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.