ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

Deep networks with ReLU activation functions can be smooth statistical models

Joseph Rynkiewicz[1]

Universté de Paris 1 - SAMM
90 rue de Tolbiac - France

Abstract. Most Deep neural networks use ReLU activation functions. Since these functions are not differentiable in 0, we may believe that such models may have irregular behavior. In this paper, we will show that the issue is more in the data than in the model, and if the data are "smooth", the model will be differentiable in a suitable sense. We give a striking illustration of this fact with the example of adversarial attacks.

1 Introduction

Deep neural networks are parametric models estimated mainly by minimizing a cost function equivalent to the opposite of a log-likelihood. Namely, a quadratic cost function is the opposite of Gaussian log-likelihood, and cross-entropy is the opposite of categorical (or multinomial) log-likelihood. Such a choice is suitable since it is well known that if the model is "smooth", the maximum likelihood estimator has optimal asymptotic properties (see van der Vaart [2], chapter 8). In the following section, we will give the conditions such that Deep Neural Networks are differentiable in quadratic mean and therefore are smooth statistical models. To be concise, we will restrict our study to the classification case, but the generalization to a regression framework is straightforward. The conditions show the importance of the data for the smoothness of the model. In the last section, to illustrate our paper, we give an example where smoothing the data prevents adversarial attacks.

2 Deep networks for classification

First, we will give a classical statistical framework for the classification problem.

2.1 Classification models

An observation is represented by a p-dimensional vector $x \in \mathbb{R}^p$ and a class y. The class y is a categorical variable, y may be coded as a set of integers $\{1, \cdots, K\}$. Let $\begin{pmatrix} X \\ Y \end{pmatrix}$ be a random vector with probability density $\{P_\theta, \theta \in \Theta\}$, where θ is the parameter vector (the weights) of the model and Θ the set of possible parameters. Let us denote Q the marginal law of X. For any parameter vector $\theta \in \Theta$, the density will be $P_\theta(x, y) = \prod_{k=1}^{K} f_{\theta(x),k}^{\mathbf{1}_k(y)} Q(x)$, where $f_{\theta,k}(x)$ is k-th the output of the Deep network. Note that the marginal law Q of X is not a parameter of interest for most classification models.

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

We will try to estimate the probabilities of the classes, conditionally to the observation x: $f_\theta(x) = (P_\theta(y = k|x))_{1\leq k\leq K}$. For an observation $x \in \mathbb{R}^p$, the deep network function from $\mathbb{R}^p$ into $\mathbb{R}^K$, with L layers can be written as a composition: $f_\theta(x) = f_{out} \circ \phi_L \circ f_L \circ \cdots \circ \phi_1 \circ f_1(x)$, where f_l is a linear preactivation function: $x_l = f_l(x_{l-1}) = \mathbf{W}_l x_{l-1} + \mathbf{b}_l$. The parameter θ is composed of input weight matrices $\mathbf{W}_l \in \mathbb{R}^{d_l \times d_{l-1}}$ and bias vectors $\mathbf{b}_l \in \mathbb{R}^{d_l}$. The activation of the i-th unit in the l-th layer is given by $x_{l,i} = \phi_l\left(f_{l,i}(x_{l-1})\right)$. ϕ_l is, in general, the ReLU activation function: $\phi_l(x) = \max(0, x)$ or continuous pooling functions like averages or maxima. The output of the l-th layer is a vector $x_l = (x_{l,1}, \cdots, x_{l,d_l})^T$. f_{out} is a function to compute the probabilities of classes $\{1, \cdots, K\}$: $((f_{out}(x_L))_k)_{1\leq k\leq K} = softmax\left(\mathbf{W}_{out}^T x_{l-1} + \mathbf{b}_{out}\right)$, where $softmax(z_1, \cdots, z_K) = \left(\frac{\exp(z_k)}{\sum_{l=1}^K \exp(z_l)}\right)_{1\leq k\leq K}$.

2.2 Maximum likelihood estimator

Let $\left(\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \cdots, \begin{pmatrix} x_n \\ y_n \end{pmatrix}\right)$ be a realization of the ramdom sample $\left(\begin{pmatrix} X_1 \\ Y_1 \end{pmatrix}, \cdots, \begin{pmatrix} X_n \\ Y_n \end{pmatrix}\right)$. We consider the opposite of the log-likelihood (conditional to the explicative data $x_1, \cdots, x_n$):

$$-l_\theta\left(\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \cdots, \begin{pmatrix} x_n \\ y_n \end{pmatrix}\right) = -\sum_{t=1}^{n}\sum_{k=1}^{K} \mathbf{1}_k(y_t) \log\left(f_{\theta,k}(x_t)\right)$$

This function is often called the cross-entropy. The estimated parameter $\hat{\theta}_n$ is the maximum likelihood estimator (MLE):

$$\hat{\theta}_n = \arg\min_{\theta\in\Theta} -l_\theta\left(\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \cdots, \begin{pmatrix} x_n \\ y_n \end{pmatrix}\right)$$

3 Differentiability in quadratic mean

Let us denote $\xi_\theta(x, y) = \sqrt{P_\theta(x, y)}$ and μ a dominating measure for $\{P_\theta(x, y), \theta \in \Theta\}$. We denote now by $\|.\|$, the $\mathcal{L}^2$-norm with respect with μ. A family of probabilities is said differentiable in quadratic mean in θ_0 (see Pollard [3]), if for a d-vector $\Delta_{\theta_0}(x, y)$ of functions in $\mathcal{L}^2(\mu)$,

$$\xi_\theta(x, y) = \xi_{\theta_0}(x, y) + (\theta - \theta_0)^T \Delta_{\theta_0}(x, y) + r(x, y, \theta - \theta_0),$$

where $\int \|r(x, y, \theta - \theta_0)\| \, d\mu(x, y) = o(\|\theta - \theta_0\|)$ when $\theta \to \theta_0$. This entails the existence of a vector of a measurable function $\dot{l}_\theta = \left(\dot{l}_{\theta,1}, \cdots, \dot{l}_{\theta,d}\right)$ such that

$$\int \left(\xi_{\theta+h} - \xi_\theta - \frac{1}{2} h \dot{l}_\theta \xi_\theta\right)^2 d\mu = o(\|h\|), \text{ when } h \to 0. \tag{1}$$

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

This property is fundamental for the smoothness of the model and optimal properties of the MLE. To show the importance of this property, we will give an example of what happens when it is not verified.

3.1 A minimal counter-example

Let $x \in \{-1, 1\}$, $f_\theta(x) = \alpha \max(\beta x + \gamma, 0)$, where $\theta = (\alpha, \beta, \gamma)$. Then the function is not differentiable on the sub-manifold $\mathcal{S} = \{\beta = \gamma\} \cup \{\beta = -\gamma\}$. We get the derivatives:

$$\begin{cases} \frac{\partial}{\partial\beta} f_\theta(x) = \frac{\partial}{\partial\gamma} f_\theta(x) = 0 \text{ if } x = 1, \text{ and } \beta < -\gamma \\ \frac{\partial}{\partial\beta} f_\theta(x) = \frac{\partial}{\partial\gamma} f_\theta(x) = \alpha \text{ if } x = 1, \text{ and } \beta > -\gamma \\ \frac{\partial}{\partial\beta} f_\theta(x) = \frac{\partial}{\partial\gamma} f_\theta(x) = 0 \text{ if } x = -1, \text{ and } \beta > \gamma \\ -\frac{\partial}{\partial\beta} f_\theta(x) = \frac{\partial}{\partial\gamma} f_\theta(x) = \alpha \text{ if } x = -1, \text{ and } \beta < \gamma \end{cases}$$

The derivative will not be continuous as soon as $\alpha \neq 0$, and there is no hope to find a function Δ such that

$$\sqrt{P_{(\alpha,\beta+h,-\beta)}(x,y)} - \sqrt{P_{(\alpha,\beta,-\beta)}(x,y)} - \Delta_{(\alpha,\beta,-\beta)}(x,y) = o(h)$$

or

$$\sqrt{P_{(\alpha,\beta+h,\beta)}(x,y)} - \sqrt{P_{(\alpha,\beta,\beta)}(x,y)} - \Delta_{(\alpha,\beta,\beta)}(x,y) = o(h)$$

Hence, a slight variation of the inputs or the parameters may result in a significant variation in the model's output. However, we will prove in the next section that this example can not occur if some components of the explicative variable x have a density with respect to the Lebesque measure.

3.2 Differentiability in quadratic mean for Deep network models

To establish the differentiability in quadratic mean, we have to show the differentiability of the map $\theta \mapsto P_\theta(x, y)$ for almost all (x, y). We must pay attention that for Deep networks with ReLU transfer functions, the set where the map $\theta \mapsto P_\theta(x, y)$ is not differentiable is a function of the variable x. The following theorem is well suited for Deep networks:

Theorem 1 *Let us split the variable x into $x = (x_1, x_2)$, where x_1 is discrete and belongs to a countable set $\mathcal{C}$, and x_2 is continuous and has a density with respect to the Lebesgue measure. Let us write $\mathring{\Theta}$ the interior of the set Θ. For all $\theta \in \mathring{\Theta}$, let $P_\theta(x, y)$ be a density of probability such that for all (x, y), $\theta \mapsto P_\theta(x, y)$ is continuous.*

1. *Assume that for all $\theta \in \mathring{\Theta}$, the set $\mathcal{N}(\theta)$ of x_2 such that $x_2 \mapsto \xi_\theta(x, y)$ is not differentiable is of Lebesgue measure null: $\lambda(N(\theta)) = 0$.*
2. *A square integrable function $\dot{l}(x, y)$ exists such that $\forall\theta \in \mathring{\Theta}$,*

$$\left\| \frac{\partial \xi_\theta(x, y)}{\partial\theta} \right\| \leq \dot{l}(x, y).$$

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1.
Available from http://www.i6doc.com/en/.

Then the map $\theta \mapsto \xi_\theta(x,y)$ is differentiable in quadratic mean, i.e. for all $\theta \in \mathring{\Theta}$, a mesurable function $\dot{l}_\theta(x,y)$ exists, such that:

$$\sum_{x_1 \in \mathcal{C}, y} P(x_1, y)$$
$$\int \left(\frac{\xi_{\theta+h}(x,y) - \xi_\theta(x,y)}{\|h\|} - \frac{1}{2} \frac{h}{\|h\|}^T \dot{l}_\theta(x,y)\, \xi_\theta(x,y) \right)^2 d\lambda(x_2) \overset{h \to 0}{\longrightarrow} 0.$$

proof By the chain rule, for all fixed x, y, for $\theta \in \mathring{\Theta}$ we get:

$$\frac{\partial f_\theta(x,y)}{\partial \theta} = \frac{\partial \xi_\theta^2(x,y)}{\partial \theta} = 2\xi_\theta(x,y) \frac{\partial \xi_\theta(x,y)}{\partial \theta}.$$

Hence, we can write

$$\frac{\partial \xi_\theta(x,y)}{\partial \theta} = \frac{1}{2} \frac{\frac{\partial f_\theta(x,y)}{\partial \theta}}{f_\theta(x,y)} \xi_\theta(x,y) := \frac{1}{2} \dot{l}_\theta(x,y) \xi_\theta(x,y),$$

where

$$\frac{\frac{\partial f_\theta(x,y)}{\partial \theta}}{f_\theta(x,y)} := 0 \text{ on } \{(x,\theta),\ f_\theta(x,y) = 0\}.$$

Since, for all (x,y), $\theta \mapsto \xi_\theta(x,y)$ is continuous, for fixed $h \in \mathbb{R}^k$,

$$\xi_{\theta+h}(x,y) - \xi_\theta(x,y) = \int_0^1 \mathbf{1}_{\{\mathring{\Theta}\}}(\theta + uh)\, h^T \frac{\partial \xi_{\theta+uh}(x,y)}{\partial \theta} du.$$

For fixed (x_1, y), Fubini theorem implies:

$$\int \left(\frac{\xi_{\theta+h}(x,y) - \xi_\theta(x,y)}{\|h\|} \right)^2 d\lambda(x_2) =$$
$$\int \left(\int_0^1 \mathbf{1}_{\{\mathring{\Theta}\}}(\theta + uh)\, h^T \frac{\partial \xi_{\theta+uh}(x,y)}{\partial \theta} du \right)^2 d\lambda(x_2) \leq$$
$$\int_0^1 \mathbf{1}_{\{\mathring{\Theta}\}}(\theta + uth) \frac{h^T}{\|h\|}$$
$$\int \left(\frac{\partial \xi(\theta + uth, x)}{\partial \theta} \left(\frac{\partial \xi(\theta + uth, x)}{\partial \theta} \right)^T d\lambda(x_2) \right) \frac{h}{\|h\|} du =$$
$$\int_0^1 \mathbf{1}_{\{\mathring{\Theta}\}}(\theta + uth) \frac{h^T}{\|h\|}$$
$$\frac{1}{4} \left(\int \mathbf{1}_{\{\mathcal{X}/\mathcal{N}(\theta+uth)\}}(x)\, \dot{l}_{\theta+uth}(x) \left(\dot{l}_{\theta+uth}(x) \right)^T f_\theta(x,y) d\lambda(x_2) \right) \frac{h}{\|h\|} du.$$

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

For $t \to 0$:

$$\int_0^1 \mathbf{1}_{\{\mathring{\Theta}\}}(\theta + uth)\frac{h^T}{\|h\|}$$
$$\frac{1}{4}\left(\int \mathbf{1}_{\{\mathcal{X}/\mathcal{N}(\theta+uth)\}}(x)\,\dot{l}_{\theta+uth}(x)\left(\dot{l}_{\theta+uth}(x)\right)^T f_\theta(x,y)d\lambda(x_2)\right)\frac{h}{\|h\|}du$$
$$\overset{t\to 0}{\longrightarrow} \frac{1}{4}\frac{h^T}{\|h\|}\int \mathbf{1}_{\{\mathcal{X}/\mathcal{N}(\theta)\}}(x)\,\dot{l}_\theta(x)\left(\dot{l}_\theta(x)\right)^T f_\theta(x,y)d\lambda(x_2)\frac{h}{\|h\|}.$$

So, for all sequences $(t_n)_{n\in\mathbb{N}}$, with $t_n \overset{n\to\infty}{\longrightarrow} 0$:

$$\lim_{n\to\infty}\sup_{k\geq n}\int\left(\frac{\xi(\theta+t_k h,x)-\xi(\theta,x)}{\|t_k h\|}\right)^2 d\lambda(x_2) =$$
$$\lim_{n\to\infty}\int\left(\frac{\xi(\theta+t_n h,x)-\xi(\theta,x)}{\|t_n h\|}\right)^2 d\lambda(x_2) \leq$$
$$\frac{1}{4}\frac{h^T}{\|h\|}\int \mathbf{1}_{\{\mathcal{X}/\mathcal{N}(\theta)\}}(x)\,\dot{l}_\theta(x)\left(\dot{l}_\theta(x)\right)^T f_\theta(x,y)d\lambda(x_2)\frac{h}{\|h\|}.$$

Finally, for $x \notin \cup_{n\in\mathbb{N}}\mathcal{N}(\theta+t_n h)\cup\mathcal{N}(\theta)$:

$$\frac{\xi(\theta+t_n h,x)-\xi(\theta,x)}{\|t_n h\|} - \frac{1}{2}\frac{h^T}{\|h\|}\frac{\partial\xi(\theta,x)}{\partial\theta} \overset{n\to\infty}{\longrightarrow} 0,$$

and we conclude with proposition 2.29 of van der Vaart [2]. ■

Remark To check the previous assumptions for Deep networks with ReLU transfer function, we remark that these models may be written as continuous piecewise linear functions. Let us write $\mathbf{I}_{\mathcal{P}}$ the indicator function of the region $\mathcal{P}$, and denote by N the total number of hidden units of the Deep network. Then, according to Rynkiewicz [4], an integer $q \leq 2^N$ exists such that, for any $\theta \in \Theta$, f_θ can be written:

$$f_\theta(x) = \sum_{i=1}^{q}\left({\beta_i}^T x + \alpha_i\right) \times \mathbf{I}_{\mathcal{P}_{\mu(i)}}(x),$$

with $(\beta_i, \alpha_i) \in \mathbb{R}^{p+1}$, $\mu(i)$ a set of parameters and $\left(\mathbf{I}_{\mathcal{P}_{\mu(1)}}(.), \cdots, \mathbf{I}_{\mathcal{P}_{\mu(q)}}(.)\right)$ are linearly independent indicator functions of regions of hyperplane arrangement. Hence, if some components of x have a density with respect to the Lebesgue measure and the weights of the network are bounded, the assumptions of the previous theorem are checked.

4 Example of adversarial attack

A practical consequence of differentiability in quadratic mean is that a small perturbation of the parameter, or the input, will result in a small variation of

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

the model's prediction. This seems in contradiction with an adversarial attack where a well-chosen small perturbation of the input can drastically change the true class's probability. An example is given in the tutorial [6]. However, if this attack works, it is essentially because the pixel of an image are discrete inputs (256 values for each color channel). If we add Gaussian noise to the pixels, the pixels will have a density with respect to the Lebesgue measure, and the model will be smooth. This idea is not new and has been explored, for example, in [5]. However, our interpretation of the efficiency of this technics is new. For example, in the tutorial [6], a hog photo is classified as a hog with a probability 0.996, and the authors compute a small perturbation δ to lower this probability. So, the probability of the hog class of $x + \delta$ will be lesser than 10^{-5}. Moreover, the probability of the wombat class will be more than 0.999. Now, if we add a Gaussian noise ε on the modified input: $x + \delta + \varepsilon = x + \varepsilon + \delta$ we get back a probability to be a hog of more than 0.99. Hence the model with noised input is smooth, and its prediction is no more perturbated by small noise. The experiment can be found in the Colab Notebook [7].

5 Conclusion

If the data are discrete, a Deep network with ReLU transfer functions is not a smooth statistical model and is vulnerable to perturbations. The main example is the adversarial attack see [1]. However, we have shown in this paper that a simple transformation of the input data can regularize the model, and avoid adversarial attack. Note that we do not modify the weights of the network. Hence, we strongly advocate systematically adding noise to input data when possible, especially for the inference, to robustify the prediction of the model.

References

[1] B. Biggio, F. Roli, Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317-331, 2018.

[2] A.W. van der Vaart. *Asymptotic statistics*, Cambridge: Cambridge University Press, Cambridge, 1998.

[3] D. Pollard, Another Look at Differentiability in Quadratic Mean. *In D. Pollard, E. Torgersen, and G.L. Yang (ed.), Festschrift for Lucien Le Cam*, 305–314. New York, Springer, 1997.

[4] J. Rynkiewicz, Asymptotic statistics for multilayer perceptron with ReLU hidden units. *Neurocomputing*, 342:C:16-23, Elsevier, 2019.

[5] Z. You, J. Ye, K. Li, Z. Xu and P. Wang, "Adversarial Noise Layer: Regularize Neural Network by Adding Noise. *IEEE International Conference on Image Processing (ICIP)*, pp. 909-913, 2019.

[6] Adversarial Robustness: Theory and Practice, Z. Kolter and A. Madry, 2018 https://adversarial-ml-tutorial.org/

[7] Notebook for the experiment: https://colab.research.google.com/drive/1XUMMWDlpztqVOUVQkzmhQTfA65CwVISe?usp=sharing