Master M2A: Neural Networks and Numerical Analysis

Bruno Després
Sorbonne Université

Introduction

> Artificial Intelligence, deep learning, machine learning-whatever you're doing if you don't understand it-learn it. Because otherwise you're going to be a dinosaur within 3 years.
>
> Mark Cuban

> Shall I refuse my dinner because I do not fully understand the process of digestion?
>
> Oliver Heaviside

Neural Networks, Machine Learning and Deep Learning are names in the air showing promises. Quoting Chollet [14][page 317]: *The technological revolution that's currently unfolding didn't start with any single breakthrough invention. Rather, like any other revolution, it's the product of a vast accumulation of enabling factors-slowly at first, and then suddenly-* ... The understanding of the factors which are in action in this revolution is an interesting challenge for applied mathematicians and research engineers.

The objective of these notes is to present in a unified framework some basic mathematical and algorithmic principles at the foundations of the **numerical analysis of neural networks**, to introduce some modern softwares and techniques in relation with these principles and to explain some connections with the numerical discretization of simple partial differential equations. There will be a particular emphasis on the following selected topics.

• **In chapter 1.** We will present the basic principles for the approximation of a given objective function $f^{\text{obj}} : \mathbb{R}^m \to \mathbb{R}^n$ with methods implementable in Neural Networks. The methodology will be introduced, and the specific language will be explained. What is the difference between the width and the depth of a NN, between the parameters and the hyper-parameters? What is a CNN (convolutive neural network)? These notions will be explained and illustrated, together with similar ones. The existence and/or uniqueness of the parameters which minimize the cost function will be detailed for neural networks with no hidden layer, both for the mean square cost function and for the cross-entropy-based cost function.

• **In chapter 2.** This part is dedicated to the derivation of best error estimates

mostly with the rectified linear unit function (ReLU) function as activation function. The Cybenko Theorem, the Yarotsky's Theorem and construction of the Takagi function will be detailed. The Yarotsky Theorem was published in 2017, it is the basis of most of the modern research in numerical analysis around Neural Networks. Some functions that can be reproduced in Neural Networks with these structures will serve as numerical illustration. The connection with the theory of Finite Element discretization is made clear when the activation function is the ReLU.

• **In chapter 3.** A recent elaboration on Neural Networks applied to the construction of monovariate polynomials will be presented. This is natural in order to simplify a key part of the Yarotsky Theorem for the construction of general polynomials. It also answers to a question which is natural in the Neural Network community (see Figures 4.7 to 4.10 in the recent book by Le Cun [54]). The answer is that monovariate polynomials correspond the expansion in Neumann series of the solution of a simple contractive functional equation. The expansion can be implemented in Neural Networks. Three different implementations will be discussed.

• **In chapter 4.** Contrary to the two previous chapters which are mostly theoretical, the rest of the notes consider only issues which arise when coming to the computer with a practical problem to implement. This chapter is dedicated to the construction datasets when the underlying space dimension is high. We will focus on numerical inverse problems which is a convenient framework for many problems in applied sciences and engineering. The Latin hypercube method will be detailed as it offers a valuable intermediate between uniform sampling and the Monte-Carlo method. Also a comprehensive convergence theory of the Latin hypercube method can be developed only with numerical analysis tools (more precisely we will not use probability theory). A generalization of the Latin hypercube method with Lattice rules will be detailed.

• **In chapter 5.** We address the numerical analysis by means of ordinary differential equations (ODEs) of some stochastic gradient descent algorithms used to calculate a good value of the parameters. In ML community, this phase is called the training of the parameters. In other communities, it is called numerical optimization. Two important ingredients are momentum methods and batches. A focus will be made on the stability properties of ADAM algorithm. The ADAM algorithm published recently in [48] is becoming a de-facto standard for training. We will briefly discuss (in Section 5.2) why popular algorithms in numerical analysis such as the Newton-Raphson method and the conjugate gradient method cannot be used for training. Other aspects will be discussed, such as the problem of initialization with zero initial guess which is very counterintuitive in numerical analysis where most algorithms can be initialized with a zero initial guess. The regularity problem for the minimization of cost functions assembled with the ReLU activation will

discussed also.

• **In chapter 6.** The final chapter is dedicated to the application of the techniques studied in the the previous chapters to a variety of problems of interest. For pedagogical purposes, the complexity of these problems of interest will be kept to the minimum. We will described the use of NN and ML to the numerical approximation of PDEs such as the advection equation $\partial_t \alpha + \mathbf{a} \cdot \nabla \alpha = 0$ and other problems. The chapter finishes with a review of selected ongoing researches at the intersection of ML and numerical analysis of PDEs.

Thanks

The author is grateful to his colleagues at Sorbonne university for the opportunity to participate to the joint courses of the Master M2A[1] between the Master Mathematics and Applications [2] [3] and the Master in computer sciences[4].

This monograph is oriented towards applied scientists who desire a mathematical acquaintance with the matter in parallel to embarking on their own Machine Leaning and Neural Networks project. It tries to follow a progression adapted to students in mathematics at the Master level. It is not a comprehensive treatise and does not pretend to present the totality of the ongoing research. All approximations or errors of presentations are my responsibility.

Bruno Després

1 (Master Apprentissage et Algorithmes) https://m2a.lip6.fr

2 http://mmath.ent.upmc.fr/fr/m2_parcours_apprentissage_et_algorithmes.html

3 https://www.ljll.math.upmc.fr/MathModel/index_en.html

4 https://sciences.sorbonne-universite.fr/formation-sciences/masters/master-informatique

Contents

1 Objective functions, neural networks and linear algebra

> We share a philosophy about linear algebra: we think basis-free, we write basis-free, but when the chips are down we close the office door and compute with matrices like fury.
>
> Irving Kaplansky

The basic problem is formulated in the context of interpolation of functions. Let f^{obj} be an objective function which models some problem of interest

$$f^{\text{obj}} : \mathbb{R}^m \longrightarrow \mathbb{R}^n. \tag{1.1}$$

Let x_i be an interpolation point in $\mathbb{R}^m$. Let $\varepsilon_i \in \mathbb{R}^n$ which denotes a noise (small enough in a sense which needs to be specified). Noise is an important notion for real data, because real measurements or real data are never perfect. Set

$$y_i = f^{\text{obj}}(x_i) + \varepsilon_i \in \mathbb{R}^n. \tag{1.2}$$

The pair $(x_i, y_i) \in \mathbb{R}^m \times \mathbb{R}^n$ is called an interpolated data with noise $\varepsilon_i \in \mathbb{R}^n$. If $\varepsilon_i = 0$, the interpolation data is noiseless.
The finite collection of interpolation data is the dataset

$$\mathcal{D} = \{(x_i, y_i),\ i = 1, \ldots, N\} \subset \mathbb{R}^m \times \mathbb{R}^n. \tag{1.3}$$

Starting from a given dataset $\mathcal{D}$, a general question is to implement in a computer an approximation denoted as f of the objective function f^{obj}. Some evident difficulties for the design of f are that the level of the noise, the curse of dimension (that is m and/or n take high values), the construction of good datasets (good datasets sample the objective function in meaningful regions), the fact that the implementation must be efficient for large $N = \#(\mathcal{D})$, the quality of the reconstructed function. Other aspects will be developed in the notes.

At the end of the chapter, the structure of basic NNs will be defined.

1.1 Notations

Standard linear algebra notations are used. For an arbitrary vector $z \in \mathbb{R}^p$ in a space of arbitrary dimension p, we note its square $|z|^2 = \sum_{i=1}^p |z_i|^2 \geq 0$. One has