Multi-layer Perceptrons for Functional Data Analysis: A Projection Based Approach

Brieuc Conan-Guez[1] and Fabrice Rossi[2]

[1] INRIA, Domaine de Voluceau, Rocquencourt, B.P. 105 78153 Le Chesnay Cedex, France
Brieuc.Conan-Guez@inria.fr
[2] LISE/CEREMADE, UMR CNRS 7534, Université Paris-IX Dauphine, Place du Maréchal de Lattre de Tassigny, 75016 Paris, France
Fabrice.Rossi@dauphine.fr

Abstract. In this paper, we propose a new way to use Functional Multi-Layer Perceptrons (FMLP). In our previous work, we introduced a natural extension of Multi Layer Perceptrons (MLP) to functional inputs based on direct manipulation of input functions. We propose here to rely on a representation of input and weight functions thanks to projection on a truncated base. We show that the proposed model has the universal approximation property. Moreover, parameter estimation for this model is consistent. The new model is compared to the previous one on simulated data: performances are comparable but training time it greatly reduced.

1 Introduction

In many practical applications, multivariate representation of studied objects is not a very satisfactory choice. If we study for instance the growth of a group of children, it is interesting to take into account the fact that the height of each child is a continuous function from a real interval (the time period of the study) to $\mathbb{R}$. Functional Data Analysis (FDA) [5] is an extension of traditional Data Analysis to individuals described by one or several regular real valued functions. The main difference between multivariate analysis and FDA is that the latter manipulates the functional representation of each individual and therefore can take into account smoothness of the considered functions. Moreover, when measurement points differ from one individual to another, multivariate representation is quite difficult because features (or variables) are now different from one individual to another: direct comparison is no more possible.

In [6,7], we have proposed an extension of Multi-Layer Perceptrons (MLP) that works on functional inputs. Given an input x in $\mathbb{R}^n$, a numerical neuron calculates an output given by $T(ax+b)$, where T is an activation function from $\mathbb{R}$ to $\mathbb{R}$, a is a vector in $\mathbb{R}^n$ and b is a real number. The main idea of Functional MLP (FMLP) is to replace the linear form $x \mapsto bx$ by a continuous linear form defined on a functional space. This approach is quite different from what is commonly used in FDA. FDA is indeed mostly based on linear methods.

J.R. Dorronsoro (Ed.): ICANN 2002, LNCS 2415, pp. 667–672, 2002.

Moreover, we proposed a direct manipulation of functions, whereas FDA relies on a pre-processing phase. In this phase, each input function is projected on a set of smooth functions (for instance on a truncated B-Spline base). In the present paper, we show that FMLP can also be applied to pre-processed functions. The main advantage of such a solution is that computation time can be greatly reduced without impairing performances (for low dimensional input spaces). As this approach uses linear models to represent input and weight functions, in high dimensional spaces this model has reduced performances compared to the direct approach, which can represent weight functions thanks to nonlinear models.

The paper is organized as follows. We first introduce the FMLP model and our new projection based approach. Then we give important theoretical results for the new model: universal approximation and consistency of parameter estimation. We conclude with some experiments that illustrate differences between direct and projection base approaches.

2 Functional MLP

2.1 Functional Neurons

As explained in the introduction, the main idea of FMLP is to replace in a neuron a linear form on $\mathbb{R}^n$ by a continuous linear form defined on the input space of the neuron. This idea was proposed and studied on a theoretical point of view in [8]. Basically, a generalized neuron takes input in E, a normed vectorial space. It is parametrized thanks to a real number b and a weight form, i.e., an element w of E^*, the vectorial space of continuous linear forms on E. The output of the neuron is $T(w(x) + b)$.

Of course, it is quite difficult to represent arbitrary continuous linear forms. That's why we proposed in [7] to restrict ourselves to functional inputs. More precisely, we denote μ a finite positive Borel measure on $\mathbb{R}^n$ (the rational for using such a measure will be explained in section 3), and $L^p(\mu)$ the space of measurable functions from $\mathbb{R}^n$ to $\mathbb{R}$ such that $\int |f|^p d\mu < \infty$. Then we can define a neuron that maps elements of $L^p(\mu)$ to $\mathbb{R}$ thanks to an activation function T, a numerical threshold b and a weight function, i.e. a function $w \in L^q(\mu)$ (where q is the conjugate exponent of p). Such a neuron maps an input function g to $T(b + \int wg \; d\mu) \in \mathbb{R}$.

2.2 Functional MLP

As a functional neuron gives a numerical output, we can define a functional MLP by combining numerical neurons with functional neurons. The first hidden layer of the network consists exclusively in functional neurons (defined thanks to weight functions w_i), whereas subsequent layers are constructed exclusively with numerical neurons. For instance, an one hidden layer functional MLP with real output computes the following function:

$$H(g) = \sum_{i=1}^{k} a_i T\left(b_i + \int w_i g \, d\mu\right) \tag{1}$$

3 Practical Implementation

3.1 Two Problems

Unfortunately, even if equation 1 uses simplified linear forms, we still have two practical problems: it is not easy to manipulate functions (i.e., weights) and we have only incomplete data (i.e., input functions are known only thanks to finite sets of input/output pairs). Therefore, computing $\int wg\,d\mu$ is quite difficult.

3.2 Projection Based Solution

We have already proposed a direct solution to both problems in [6,7]. The main drawback of the direct solution is that computation times can be quite long.

In this paper, we propose a new solution based on projected representation commonly used in FDA (see [5]). The main idea is to use a regularized representation of each input function. In this case we assume that the input space is $L^2(\mu)$ and we consider $(\phi_i)_{i\in\mathbb{N}^*}$ a Hilbertian base of $L^2(\mu)$. We define $\Pi_n(g)$ as the projection of $g \in L^2(\mu)$ on the vectorial space spanned by $(\phi_1,\ldots,\phi_n)$ (denoted $span(\phi_1,\ldots,\phi_n)$), i.e. $\Pi_n(g) = \sum_{i=1}^{n}(\int \phi_i g\,d\mu)\phi_i$. Rather than computing $H(g)$, we try to compute $H(\Pi_n(g))$. It is important to note that in the proposed approach we consider $\Pi_n(g)$ to be the effective input function. Therefore, with the projection based approach, we do not focus anymore on computing $H(g)$.

In order to represent the weight functions, we consider another Hilbertian base of $L^2(\mu)$, $(\psi_j)_{j\in\mathbb{N}^*}$ and we choose weight functions in $span(\psi_1,\ldots,\psi_p)$. For the weight function $w = \sum_{j=1}^{p}\alpha_j\psi_j$, we have:

$$\int w\Pi_n(g)\,d\mu = \sum_{i=1}^{n}\sum_{j=1}^{p}(\int \phi_i g\,d\mu)\alpha_j \int \phi_i\psi_j\,d\mu \tag{2}$$

For well chosen bases (e.g., B-splines and Fourier series), $\int \phi_i\psi_j\,d\mu$ can be easily computed exactly (especially if we use the same base for both input and weight functions!). Moreover, the values do not depend on actual weight and/or input functions. Efficiency of the method is deeply linked to this fact which is a direct consequence of using truncated bases.

A FMLP based on this approach uses a finite number of real parameters because each weight function is represented thanks to its p real coordinates in $span(\psi_1,\ldots,\psi_p)$. Weights can be adjusted by gradient descent based algorithms, as the actual calculation done by a functional neuron is now a weighted sum of its inputs if we consider the finite dimensional input vector $(\int \phi_1 g\,d\mu,\ldots,\int \phi_n g\,d\mu)$. Back-propagation is very easily adapted to this model and allows efficient calculation of the gradient.

3.3 Approximation of the Projection

Of course, $\Pi_n(g)$ cannot be computed exactly because our knowledge on g is limited to a finite set of input/output pairs, i.e., $(x_k, g(x_k))$. We assume that the

measurement points x_k are realizations of independent identically distributed random variables $(X_k)_{k\in\mathbb{N}}$ (defined on a probability space $\mathcal{P}$) and we denote P_X the probability measure induced on $\mathbb{R}^n$ by those random variables. This observation measure weights measurement points and it seems therefore natural to consider that $\mu = P_X$.

Then we can define the random element $\widehat{\Pi}_n(g)^m$ as the function $\sum_{i=1}^n \widehat{\beta}_i\phi_i$ that minimizes $\frac{1}{m}\sum_{k=1}^m \left(g(X_k) - \sum_{i=1}^n \beta_i\phi_i(X_k)\right)^2$. This is a straightforward generalized linear problem which can be solved easily and efficiently with standard techniques. Therefore, rather than computing $H(\Pi_n(g))$ we compute a realization of the random variable $H(\widehat{\Pi}_n(g)^m)$. Following [1] we show in [4] that $\widehat{\Pi}_n(g)^m$ converges almost surely to $\Pi_n(g)$ (in $L^2(\mu)$). This proof is based on an application of theorem 1.1.1 of [3].

4 Theoretical Results

4.1 Universal Approximation

Projection based FMLP are universal approximators:

Corollary 1. *We use notations and hypotheses of subsection 3.2. Let F be a continuous function from a compact subset K of $L^2(\mu)$ to $\mathbb{R}$ and let ϵ be an arbitrary strictly positive real number. We use a continuous non polynomial activation function T. Then there is $n > 0$ and $p > 0$ and a FMLP (based on T and using weight functions in span$(\psi_1, \ldots, \psi_p)$) such that $|H(\Pi_n(g)) - F(g)| < \epsilon$ for each $g \in K$.*

Proof. Details can be found in [4]. First we apply corollary 1 of [7] (this result is based on very general theorems from [8]) to find a FMLP H based on T that approximates F with precision $\frac{\epsilon}{2}$. This is possible because $(\psi_j)_{j\in\mathbb{N}^*}$ is a Hilbertian base of $L^2(\mu)$ which is its own dual. p is obtained as the maximum number of base functions used to represent weight functions in the obtained FMLP.

H is continuous (because T is continuous) and therefore uniformly continuous on K. There is $\eta > 0$ such that for each $(f, g) \in K^2$, $\|f - g\|_2 < \eta$ implies $|H(f) - H(g)| < \frac{\epsilon}{2}$. As $(\phi_i)_{i\in\mathbb{N}^*}$ is a basis and K is compact, there is $n > 0$ such that for each $g \in K$, $\|\Pi_n(g) - g\|_2 < \eta$, which allows to conclude.

In the practical case, $\Pi_n(g)$ is replaced by $\widehat{\Pi}_n(g)^m$ which does not introduce any problem thanks to the convergence result given in 3.3.

4.2 Consistency

When we estimate optimal parameters for a Functional MLP, our knowledge of data is limited in two ways. As always in Data Analysis problems, we have a finite set of input/output pairs. Moreover (and this is specific to FDA), we have also a limited knowledge of each input function. Therefore we cannot apply classical

MLP consistency results (e.g., [9]) to our model. Nevertheless, we demonstrate a consistency result in [4], which is summarized here.

We assume that input functions are realizations of a sequence of independent identically distributed random elements $(G^j)_{j\in\mathbb{N}}$ with values in $L^2(\mu)$ and we denote $G = G^1$. In regression or discrimination problems, each studied function G^j is associated to a real value Y^j ($(Y^j)_{j\in\mathbb{N}}$ is a sequence of independent identically distributed random variables and we denote $Y = Y^1$). Our goal is that the FMLP gives a correct mapping from g to y. This is modeled thanks to a cost function l (in fact we embed in l both the cost function and the FMLP calculation). If we call w the vector of all numerical parameters of a FMLP, we have to minimize $\lambda(w) = E(l(G, Y, w))$.

As we have a finite number of functions, the theoretical cost is replaced by the random variable $\widehat{\lambda}_N(w) = \frac{1}{N}\sum_{j=1}^{N} l(G^j, Y^j, w)$. Moreover, $H(\Pi_n(g))$ is replaced by the random variable $H(\widehat{\Pi}_n(g)^m)$. Therefore, we have to replace l by an approximation which gives the random variable $\widehat{\lambda}_N(w)^m = \frac{1}{N}\sum_{j=1}^{N} l(G^j, Y^j, w)^m$.

We call $\widehat{w}_N^m$ a minimizer of $\widehat{\lambda}_N(w)^m$ and W^* the set of minimizers of $\lambda(w)$. We show in [4] that $\lim_{n\to\infty}\lim_{m\to\infty} d(\widehat{w}_n^m, W^*) = 0$.

5 Simulation Result

We have tried our model on a simple discrimination problem: we ask to a functional MLP to classify functions into classes. Examples of the first class are functions of the form $f_d(x) = \sin(2\pi(x-d))$, whereas functions of the second class have the general form $g_d(x) = \sin(4\pi(x-d))$.

For each class we generate example input functions according to the following procedure:

1. d is randomly chosen uniformly in $[0, 1]$
2. 25 measurement points are randomly chosen uniformly in $[0, 1]$
3. we add a centered Gaussian noise with 0.7 standard deviation to the corresponding outputs

Training is done thanks to a conjugate gradient algorithm and uses early stopping: we used 100 training examples (50 of each class), 100 validation examples (50 of each class) and 300 test examples (150 for each class).

We have compared on those data our direct approach ([6]) to the projection based approach. For both approaches we used a FMLP with three hidden functional neurons. Each functional neuron uses 4 cubic B-splines to represent its weight functions. For the projection based approach, input functions are represented thanks to 6 cubic B-splines. Both models use a total of 19 numerical parameters. For the direct approach, we obtain a recognition rate of 94.4 % whereas the projection based approach achieves 97% recognition rate. Moreover, each iteration of the training algorithm takes about 20 times more time for the direct approach than for the projection based approach.

We have conducted other experiments (for instance the circle based one described in [6]) which give similar results. When the dimension of the input space

of each function increases, the generalized linear model used to represent input and weight functions becomes less efficient than a non linear representation ([2]). We have therefore to increase the number of parameters in order to remain competitive with the direct method which can use numerical MLP to represent weights.

6 Conclusion

We have proposed in this paper a new way to work with Functional Multi Layer Perceptrons (FMLP). The projection based approach shares with the direct one very important theoretical properties: projection based FMLP are universal approximators and parameter estimation is consistent for such model.

The main advantage of projection based FMLP is that computation time is greatly reduced compared to FMLP based on direct manipulation of input functions, whereas performances remain comparable. Additional experiments are needed on input functions with higher dimensional input spaces (especially on real world data). For this kind of functions, advantages of projection based FMLP should be reduced by the fact that they cannot use traditional MLP to represent weight and input functions. The direct approach does not have this limitation and should therefore use less parameters. Our goal is to establish practical guidelines for choosing between both approaches.

References

1. Christophe Abraham, Pierre-André Cornillon, Eric Matzner-Lober, and Nicolas Molinari. Unsupervised curve clustering using b-splines. Technical Report 00-04, ENSAM-INRA-UM 11-Montpellier, October 2001.
2. Andrew R. Barron. Universal Approximation Bounds for Superpositions of a Sigmoidal Function. *IEEE Trum. Information Theory*, 39(3):930–945, May 1993.
3. Helga Bunke and Olaf Bunke, editors. *Nonlinear Regression, Functional Relations and Robust Methods*, volume II of Series in Probability and Mathematical Statistics. Wiley, 1989.
4. Brieuc Conan-Guez and Fabrice Rossi. Projection based functional multi layer perceptrons. Technical report, LISE/CEREMADE & INRIA, http://www.ceremade.dauphine.fr/, february 2002.
5. Jim Ramsay and Bernard Silverman. *Functional Data Analysis.* Springer Series in Statistics. Springer Verlag, June 1997.
6. Fabrice Rossi, Brieuc Conan-Guez, and François Fleuret. Functional data analysis with multi layer perceptrons. In *IJCNN 2002/WCCI 2002*, volume 3, pages 2843–2848. IEEE/NNS/INNS, May 2002.
7. Fabrice Rossi, Brieuc Conan-Guez, and Fraqois Fleuret. Theoretical properties of functional multi layer perceptrons. In *ESANN 2002*, April 2002.
8. Maxwell B. Stinchcombe. Neural network approximation of continuous functionals and continuous functions on compactifications. *Neural Networks*, 12(3):467–477, 1999.
9. Halbert White. Learning in Artificial Neural Networks: A Statistical Perspective. *Neural Computation*, 1(4):425–464, 1989.