Computational experience with pseudoinversion-based training of neural networks using random projection matrices

Luca Rubini[1] Rossella Cancelliere[1] Patrick Gallinari[2]
Andrea Grosso[1] Antonino Raiti[1]

[1]University of Torino - Department of Computer Science, Torino, Italy

[2]Laboratory of Computer Sciences, LIP6, Université Pierre et Marie Curie
Paris, France

September 12, 2014

Summary

1. Introduction
2. Training by Pseudoinversion
3. Random Projections
4. Experimental Investigation
5. Conclusions and future works

Introduction

- Our work moves forward from some new ideas concerning the use of matrix pseudoinversion to **train** a Single Hidden Layer Feedforward Networks (SLFN): the Extreme Learning Machine (Huang et al., 2006).
- The main feature of this method is that **input weights** are randomly chosen, and never modified, while **output weights** are anatically determined by pseudoinversion.
- This **non iterative** procedure makes training very fast, but some care is required because of the known numerical instability of pseudoinversion.

Introduction

- We have introduced a **regularisation task** to increas the numerical stability of the problem.
- The aim of our work is to find out if there is a possible **better inizialisation for the input weights**. We look at the **Random Projection** for their recent results in other fields *(Achliopts, 2001 and Bingham and Mannila, 2001)*.

Summary

1 Introduction

2 Training by Pseudoinversion

3 Random Projections

4 Experimental Investigation

5 Conclusions and future works

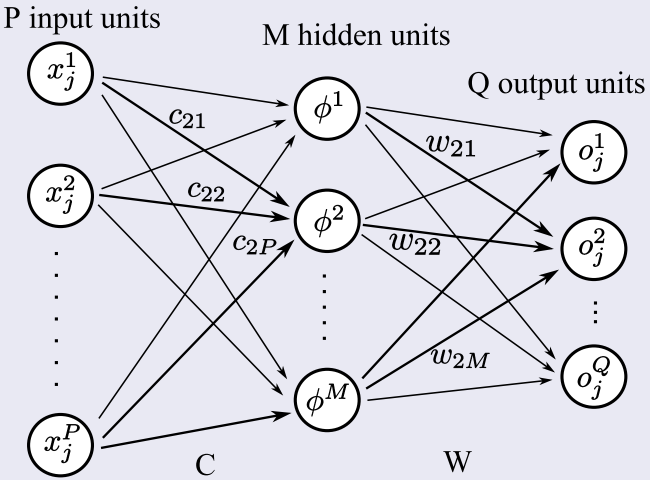
Single Layer FeedForward Network

SLFN Structure

- P input neurons
- M hidden neurons
- Q output neurons
- ϕ non-linear activation function *(sigmoid, hyperbolic tangent...)*

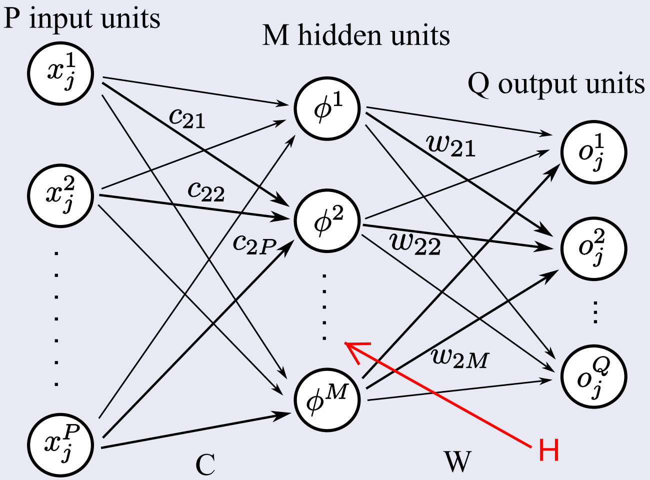
Single Layer FeedForward Network

SLFN Scheme



Single Layer FeedForward Network

SLFN Scheme



Training by Pseudoinversion

Dataset

N training samples:

$$(\mathbf{x}_i, \mathbf{t}_j) \in \mathbb{R}^P \times \mathbb{R}^Q \implies \begin{array}{l} X \in \mathbb{R}^{N \times P} (\text{input}) \\ \\ T \in \mathbb{R}^{N \times Q} (\text{desired output}) \end{array}$$

$C = (c_{ij})$ input weights

$W = (w_{ij})$ output weights

We get the linear system

$$HW = T$$

where

$$H = \phi(XC) \in \mathbb{R}^{N \times M}$$

Training by Pseudoinversion

To find the solution we consider the cost functional

$$E_D = ||HW - T||_2^2$$

The least square solution W^* is

$$W^* = H^+ T$$

H^+: Moore-Penrose pseudoinverse

Tikhonov regularisation

The problem is ill-posed $\Rightarrow$ We want to transform in a well-posed.

$$E \equiv E_D + E_R = ||HW - T||_2^2 + \lambda ||W||_2^2.$$

Training by Pseudoinversion

Compute the regularised solution $\hat{W}$

SVD decomposition of H (remeber that $H = H(C)$)

$$H = U\Sigma V^t$$

regularized solution is

$$\boxed{\hat{W} = VDU^t T,}$$

where

$$D_i = \frac{\sigma_i}{\sigma_i^2 + \lambda}.$$

Summary

1. Introduction
2. Training by Pseudoinversion
3. Random Projections
4. Experimental Investigation
5. Conclusions and future works

Basic Ideas on Random Projections

$$X^{RP}_{N \times K} = X_{N \times P} C_{P \times K}$$

Columns of $C_{P \times K}$ are **orthogonal versors**.

Lemma (Johnson-Lindenstrauss, 1984)

If a set of points in a vector space is randomly projected onto a selected space of suitable dimension, then the original distances between the points are approximately preserved in the new space, with only minimal distortions.

- Preserve the topological structure of the initial input space.
- Creation of a new optimal data representation in the hidden layer space, able to easy the classification/diagnosis task and to increase performance.
- Computational advantages.

Basic Ideas on Random Projections

It's difficult to obtain this matrix without a process of orthonormalisation, but these two results help us.

Theorem (Hecht-Nielsen, 2011)

In a high-dimensional space, there exists a much larger number of almost orthogonal than strictly orthogonal directions.

Proposition (Bingham and Mannila, 2001)

Vectors with random directions might be sufficiently close to orthogonality,

$$C^T C \approx I.$$

Random Projection Initialization

RP Gaussian initialization

$$c_{ij} = \mathcal{N}(0, 1)$$

RP Sparse initialization (Achiloptas, 2001)

$$c_{ij} = \sqrt{3} \cdot \begin{cases} +1 & \text{prob. } 1/6 \\ 0 & \text{prob. } 2/3 \\ -1 & \text{prob. } 1/6 \end{cases}$$

Uniform initialization (not RP)

$$c_{ij} = \mathcal{U}(-1, 1)$$

Summary

1. Introduction
2. Training by Pseudoinversion
3. Random Projections
4. Experimental Investigation
5. Conclusions and future works

UCI datasets characteristics

Dataset	Type	N. Instances	N. Attributes	N. Classes
Abalone	Regr.	4177	8	-
Cpu	Regr.	209	6	-
Delta Ailerons	Regr.	7129	5	-
Housing	Regr.	506	13	-
Iris	Class.	150	4	3
Diabetes	Class.	768	8	2
Wine	Class.	178	13	3
Landsat	Class.	4435	36	7

Calibration phase

Free parameters:

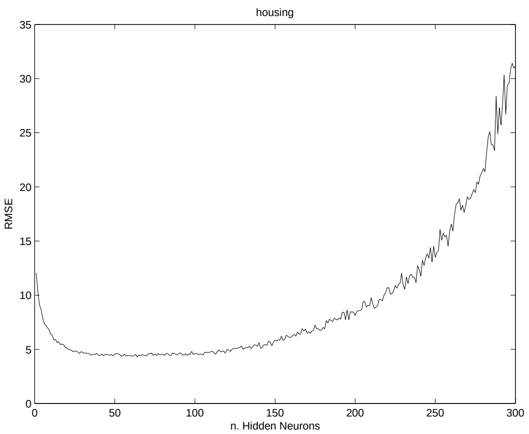
- nH: Number of Hidden neurons.
- λ: regularisation value.

Calibration

1. 100 trials of unregularised state varying number of **hidden neurons** on the validation set.
2. Fixed the size of hidden layer find the λ value of the **best performance** on the validation set.

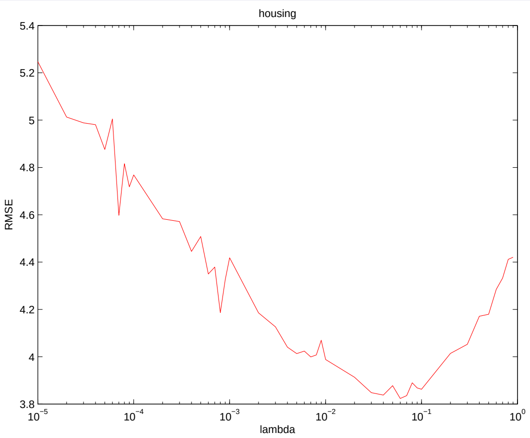
Calibration phase

Unregularized performance (validation set error)



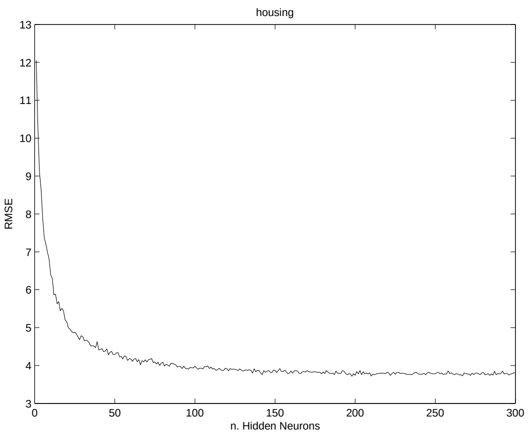
Calibration phase

Finding λ (validation set error)



Calibration phase

Regularised performance (validation set error)



Calibration phase

Calibrated parameters

	Unif.		Gauss.		Sparse	
	nH	λ	nH	λ	nH	λ
Abalone	128	3e-2	129	3e-1	118	3e-2
Mach. Cpu	98	4e-2	61	8e-1	89	5e-1
Delta Ail.	244	3e-3	272	3e-2	225	3e-3
Housing	130	8e-3	200	5e-2	180	7e-2
Iris	102	3e-4	120	3e-2	266	3e-3
Diabetes	266	3e-3	173	3e-2	192	3e-3
Wine	60	3e-2	70	2e-1	80	8e-2
Landsat	579	3e-3	600	3e-2	600	3e-3

Experimentations

Methodology

- 100 trials for each fixed size (nH, λ).
- The error is the average test set error value of 100 trials.
- In **bold** results statistically significat with confidence level of 95% for a Student's test.

Experimentations

Results

	Unif.		Gauss.		Sparse	
	Avg	StD	Avg	StD	Avg	StD
Abalone	2.165	0.004	2.169	0.009	**2.162**	0.006
Mach. Cpu	57.35	1.7	**56.85**	2.8	57.86	1.6
Delta Ail.	1.636e-4	2e-3	**1.630e-4**	4e-3	1.636e-4	2e-3
Housing	3.61	0.21	3.58	0.19	3.64	0.18
Iris	1.00	1.1	1.88	1.1	1.08	1.0
Diabetes	20.312	0.8	20.430	1.0	**20.086**	1.0
Wine	2.2542	1.5246	2.0847	1.6313	2.5593	1.5704
Landsat	10.438	0.32	**9.848**	0.30	10.394	0.32

Experimentations

Random projections based training (pseudoinversion) vs. backpropagation (tunedit.org)

Dataset	PINV		Backprop.	
	Avg.	stdDev	Avg.	stdDev
Abalone	**2.162**	0.006	2.3044	0.1908
Mach. Cpu	**56.85**	2.8	28.6673	27.3535
Delta Ail.	**1.630e-4**	4e-7	2e-3	0.0
Housing	**3.58**	0.19	4.5492	0.9517
Iris	**1.00**	1.1	1.73	0.85
Diabetes	**20.086**	1.0	26.52	2.38
Wine	**2.0847**	1.6313	3.77	0
Landsat	**9.848**	0.30	13.03	0.63

Summary

1. Introduction
2. Training by Pseudoinversion
3. Random Projections
4. Experimental Investigation
5. Conclusions and future works

Conclusions

- RP give **better performance** (excluse Iris) than classic uniform initialization.
- There is a **stastically significance** of the goodness of results related to RP initializations.
- RP initialization matrices is a useful tool for improving **computationally and complexity** performances.
- Direct method regularised (PINV) is **faster** then classic iterative (Backprop.).

Future works

- Obtain better performances **varying** the input weights matrix C.
- Improvement not made by a random *blind* search, but starting from an initialization $C^{(1)}$ and **moving around** it getting new matrices $C^{(k)}$.
- **Strategy**: swapping two rows, so a RP matrix with swapped rows is still a RP matrix (Local Search).

Thanks for your attention. . .