IRIS AperTO

UNIVERSITÀ DEGLI STUDI DI TORINO

AperTO - Archivio Istituzionale Open Access dell'Università di Torino

Matrix pseudoinversion for image neural processing

This is the author's manuscript

Original Citation:

Availability:

This version is available http://hdl.handle.net/2318/118624 since 2017-05-23T18:29:35Z

Publisher:

SPRINGER-VERLAG BERLIN, HEIDELBERGER PLATZ 3, D-14197 BERLIN, GERMANY

Published version:

DOI:10.1007/978-3-642-34500-5_15

Terms of use:

Open Access

Anyone can freely access the full text of works made available as "Open Access". Works made available under a Creative Commons license can be used according to the terms and conditions of said license. Use of all other works requires consent of the right holder (author or publisher) if not exempted from copyright protection by the applicable law.

(Article begins on next page)

21 May 2024

Matrix Pseudoinversion for Image Neural Processing

Rossella Cancelliere[1], Mario Gai[2], Thierry Artières[3], and Patrick Gallinari[3]

[1] Universitá di Torino, Turin, Italy
rossella.cancelliere@unito.it
[2] National Institute of Astrophysics, Turin, Italy
[3] LIP6, Université Pierre et Marie Curie, Paris, France

Abstract. Recently some novel strategies have been proposed for training of Single Hidden Layer Feedforward Networks, that set randomly the weights from input to hidden layer, while weights from hidden to output layer are analytically determined by Moore-Penrose generalised inverse. Such non-iterative strategies are appealing since they allow fast learning, but some care may be required to achieve good results, mainly concerning the procedure used for matrix pseudoinversion. This paper proposes a novel approach based on original determination of the initialization interval for input weights, a careful choice of hidden layer activation functions and on critical use of generalised inverse to determine output weights. We show that this key step suffers from numerical problems related to matrix invertibility, and we propose a heuristic procedure for bringing more robustness to the method. We report results on a difficult astronomical image analysis problem of chromaticity diagnosis to illustrate the various points under study.

Keywords: Pseudoinverse matrix, Weights initialization, Supervised learning.

1 Introduction

In the past two decades, single hidden layer feedforward neural networks (SLFNs) have been one of the most important subject of study and discussion among neural researchers. Methods based on gradient descent have mainly been used, although defining different learning algorithms; among them there is the large family of techniques based on backpropagation, widely studied in its variations [1]. The start-up of these techniques assigns random values to the weights connecting input, hidden and output nodes, these weights are then iteratively adjusted.

In any case, gradient descent-based learning methods are typically slow, frequently require small learning steps, and are subject to convergence to local minima. Therefore, many iterations may be required by such algorithms in order to achieve an adequate learning performance, and many trials are required to avoid poor local minima.

Some non iterative procedures based on the evaluation of generalized pseudoinverse matrices have been recently proposed as novel learning algorithms for

SLFNs: among them the method to improve performance of multilayer perceptron by Halawa [2], the extreme learning machine (elm) [3] and some studies more application oriented [4,5,6]. Usually, input weights (linking input and hidden layers) are randomly chosen, and output weights (linking hidden and output layers) are analytically determined by the Moore-Penrose (MP) generalized inverse (or pseudoinverse). These theoretically appealing methods have many interesting features among which is their non iterative nature, that makes them very fast, but some care may be required because of the known numerical instability of the process of pseudoinverse determination (see [7]).

This paper presents a deep investigation on a few major issues of pseudoinversion-based learning of SLFN's and on related numerical problems; we put in light that this technique can not be used without a careful analysis, because of the eventual presence of almost singular matricies whose pseudoinversion can cause instability and consequent error growth. We also suggest a method based on threshold tuning to give greater stability to solutions.

We first recall the main ideas on SLFN learning through matrix pseudoinversion in section 2. In section 3 we present the main methods to compute output weights and their possible issues. In section 4 we describe the astronomical problem of chromaticity diagnosis through image analysis. Finally in section 5 we propose a new initialization criterion for input weights, we compare various ways of evaluating pseudoinverse and discuss results and performance trends.

2 How to Train Weights by Pseudoinversion

A standard SLFN with P inputs, M hidden neurons, Q output neurons, nonlinear activation functions ϕ on the hidden layer and linear activation functions otherwise, computes an output vector $o = (o_1, ..., o_Q)$ from an input vector $x = (x_1, ..., x_P)$ according to:

$$o_k = b_k^O + \sum_{i=1}^{M} w_{k,i}\phi(\mathbf{c}_i \cdot \mathbf{x} + b_i^H) \qquad k = 1, \cdots Q \tag{1}$$

where the vector $\mathbf{c}_i$ denotes the weights connecting input units to hidden neuron i, $w_{k,i}$ denotes the weight linking the hidden neuron i to the output neurons k, b_k^O and b_i^H denote biases for output and hidden neurons.

Considering a dataset of N distinct training samples of (input, output) pairs $(\mathbf{x}_j, \mathbf{t}_j)$, where $\mathbf{x}_j \in \Re^P$ and $\mathbf{t}_j \in \Re^Q$, learning a SLFN aims at producing the desired output $\mathbf{t}_j$ when $\mathbf{x}_j$ is presented as input, i.e. at determining weights w, c, and biases b such that:

$$t_j^k = b_k^O + \sum_{i=1}^{M} w_{ki}\phi(\mathbf{c}_i \cdot \mathbf{x}_j + b_i^H) = \sum_{i=1}^{M+1} w_{ki}\phi(\mathbf{c}_i \cdot \mathbf{x}_j + b_i^H) \quad k{=}1, \cdots Q \; j{=}1, \cdots N \tag{2}$$

We made the assumptions $\phi(\mathbf{c}_{M+1} \cdot \mathbf{x}_j + b_{M+1}^H) = 1$ and $w_{k,M+1} = b_k^O$ to include the bias terms in the sum; in the following we will deal with a generic number M

of hidden neurons to simplify notation. The above equations can thus be written compactly in matrix form as

$$\mathbf{T} = \mathbf{H}\mathbf{w}, \tag{3}$$

where:

$$\mathbf{w} = \begin{vmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_M^T \end{vmatrix}_{M\times Q} \quad \mathbf{T} = \begin{vmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{vmatrix}_{N\times Q} \quad \mathbf{H} = \begin{vmatrix} \phi(\mathbf{c}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & \phi(\mathbf{c}_M \cdot \mathbf{x}_1 + b_M) \\ \vdots & \ddots & \vdots \\ \phi(\mathbf{c}_1 \cdot \mathbf{x}_N + b_1) & \cdots & \phi(\mathbf{c}_M \cdot \mathbf{x}_N + b_M) \end{vmatrix}_{N\times M} \tag{4}$$

$\mathbf{H}$ is the hidden layer output matrix of the neural network; the i-th column of $\mathbf{H}$ is the i-th hidden node output with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N$. In most cases of interest, the number of hidden nodes is much lower than the number of distinct training samples, i.e. $M << N$, so that $\mathbf{H}$ is a non-square matrix; in this case a *least-squares solution* is searched, determining $\mathbf{c}, b, w$ such that the following cost functional is minimized:

$$E_D = \sum_{j=1}^{N}\sum_{k=1}^{Q}\left(t_j^k - \sum_{i=1}^{M} w_{ki}\phi(\mathbf{c}_i \cdot \mathbf{x}_j + b_i)\right)^2 = ||\mathbf{H}\mathbf{w} - \mathbf{T}||^2. \tag{5}$$

As stated above, gradient-based iterative learning algorithms, that require to adjust input weights and hidden layer biases, are generally used to search the minimum of $||\mathbf{H}\mathbf{w} - \mathbf{T}||^2$.

The least-squares solution w^* of the linear system is then (see [8])

$$\mathbf{w}^* = \mathbf{H}^+\mathbf{T}, \tag{6}$$

where $\mathbf{H}^+$ is the Moore-Penrose pseudoinverse of matrix $\mathbf{H}$.

The solution $\mathbf{w}^*$ has some important properties, in fact it is one of the least-squares solutions of the general linear system (3), hence it reaches the smallest training error. Besides, it has the smallest norm among all least-squares solutions and it is unique.

Huang et al. [3] proved that, contrarily to what happens in traditional training algorithms, input weights and hidden layer biases of a SLFN do not need to be adjusted, but they can be randomly assigned. In doing so, SLFN output weights can be simply considered as the solution of a linear system and analytically determined through generalized inversion (or pseudoinversion) of the hidden layer output matrix, assuming that hidden neurons activation functions are infinitely differentiable. These ideas gave rise to an algorithm, which is extremely fast because it works in a single pass. Besides, pseudoinversion provides the smallest weights norm solution: this point is quite relevant, since, according to Bartlett's theory [9] the smaller is the norm of weights, the better generalization performance is usually achieved by the network.

3 Pseudoinverse Computation

Hereafter we address the main critical issue which concerns learning of output layer weights by pseudoinversion.

Several methods are available, (see e.g. [10]), to evaluate the Moore-Penrose pseudoinverse matrix: in particular, the orthogonal projection (OP) method can be used when $\mathbf{H}^T\mathbf{H}$ is nonsingular, so that

$$\mathbf{H}^+ = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T \tag{7}$$

but it is known that this method is affected by severe limitations when the matrix $\mathbf{H}^T\mathbf{H}$ is almost singular. In this case, the computation of the pseudoinverse is highly unstable, and consequently the product $\mathbf{H}^+\mathbf{H}$ is potentially much different from the unit matrix. A possible solution consists in the addition of a regularization term to the cost functional (5)

$$E = E_D + \lambda E_W \tag{8}$$

where λ is the regularization coefficient that controls the relative strength of the data-dependent error E_D and the regularization term E_W. For L_2 regularization the term E_W usually takes the form $E_W = \frac{1}{2}\mathbf{w}^T\mathbf{w}$, so that the cost functional (5) becomes:

$$E = E_D + \lambda E_W = \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{Q}\left(\left(t_j^k - \sum_{i=1}^{M} w_{ki}\phi(\mathbf{c}_i \cdot \mathbf{x}_j + b_i)\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{M}|w_{ki}|^2\right) \tag{9}$$

With this approach, i.e. regularised OP (ROP), the solution (6) becomes:

$$\mathbf{w}^* = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{T}. \tag{10}$$

A different approach consists in evaluating the singular value decomposition (SVD) of $\mathbf{H}$, $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ (see for instance [11]). $\mathbf{U}^{N\times N}, \mathbf{V}^{M\times M}$ are unitary matrices and $\mathbf{S}^{N\times M}$ has elements $\sigma_{ij} = 0$ for $i \neq j$ and $\sigma_{ii} = \sigma_i$ for $i = j$, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$, $p = \min\{N, M\}$; the elements σ_i are the singular values of the decomposition.

We can also define the pseudoinverse matrix $\mathbf{H}^+$ through the SVD of $\mathbf{H}$ (see [7] for more details) as:

$$\mathbf{H}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\mathbf{T} \tag{11}$$

where $\mathbf{\Sigma}^{+\mathbf{M}\times\mathbf{N}}$ has elements $\sigma_{ij}^+ = 0$ for $i \neq j$ and $\sigma_{ii}^+ = 1/\sigma_i$ for $i = j$. If $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > \sigma_{k+1} = \cdots = \sigma_p = 0$, the rank of matrix $\mathbf{H}$ is k and the inverses of the $p - k$ zero elements are replaced by zeros.

Also, this method may exhibits some problems in the presence of almost singular matrices because of the presence of very small singular values σ_i, whose numerical inversion may cause instability in the algorithm.

In section 5 we will therefore propose a solution through the introduction of a cut-off threshold, acting as a sort of 'regularisation' that stabilises the algorithm.

4 The Astronomical Problem

Astrometry, i.e. the precise determination of stellar positions, distance and motion, is largely based on imaging instrumentation fed by telescopes operating

in the visible range. The measured image profile of a star however depends on its spectral type, i.e. the emitted light distribution as a function of frequency, so that its measured position appears affected by an error called chromaticity. Chromaticity was identified for the first time [12] in the data analysis of the space mission *Hipparcos* of the European Space Agency (ESA).

The chromaticity issue becomes even more important for the current European Space Agency mission Gaia [13] for global astrometry, aiming at much higher precision and approved for launch in 2013.

The detection and correction of chromaticity in different conditions has been addressed in recent years [14,15]; to this purpose, a single-hidden layer feedforward neural network (SLFN), trained by a classical BP algorithm, was used to solve this diagnosis task.

Each image is first reduced to a one-dimensional signal $s(x)$ along the single measurement direction of Gaia, by integration on the orthogonal direction, also to reduce the data volume and telemetry rate.

With respect to our previous studies, we also adopt as input to the neural network processing a more convenient set of statistical moments M_k for image encoding:

$$M_k = \sum_n \left(x_n - x_{COG}\right)^k \cdot s\left(x_n\right) \cdot s_A\left(x_n\right) , \tag{12}$$

where $s\left(x_n\right)$ is the above mentioned signal, $s_A\left(x_n\right)$ is the signal from an ideal instrument, and x_{COG} is the signal barycenter.

Chromaticity can be more conveniently defined using the concept of blue (effective temperature $T = 30,000\,K$) and red ($T = 3,000\,K$) stars, in particular it is the barycenter displacement between them, and this is the neural network target. Correct diagnostics allows effective correction of this kind of error, therefore a good approximation by the neural network results in a small *residual chromaticity* after neural processing.

The 11 neural network inputs are the red barycenter and the moments of red and blue stars from Eq. 12 ($k = 1, \cdots, 5$), computed for each instance.

We have a total of 13000 instances, built according to the above prescriptions, and split in a training set of 10000 instances and a test set with 3000 instances. The data set was provided by the Astrophysical Observatory of Turin of the Italian National Institute for Astrophysics.

5 Experimental Results on Chromaticity Diagnosis

We investigate in this section on the one hand the influence of activation functions choice and on the other hand the importance of procedures used for evaluating the pseudoinverse. As an important result, we further put in evidence a numerical problem encountered dealing with singular value decomposition, that may lead to poor performance and for which we propose a solution that increases the robustness of the method, allowing it also to reach better performance with respect to classical backpropagation.

We investigate neural networks with 11 input neurons corresponding to the 11 statistical moments describing each image, 1 output neuron and a number of hidden neurons varying from 50 to 600. For each initial setup, the number of hidden nodes has been gradually increased adding one node each time and average results are computed over 10 simulation trials for each selected size of SLFN. All the simulations are carried out in the Matlab 7.3 environment.

The use of sigmoidal activation functions has recently been subject to debate because they seem to be more easily driven towards saturation due to their non-zero mean value [16]; hyperbolic tangent seems to be less sensitive to this problem, therefore we utilize this one. Besides, in the perspective of mitigating saturation issues, we propose a uniform random choice of input weights in the range $(-1/\sqrt{M}, 1/\sqrt{M})$, where M is the number of hidden nodes. This links the size of input weights, and therefore of input values to hidden neurons, to the network architecture, thus forcing the use of the almost linear central part of the hyperbolic activation function when exploring the performance as a function of an increasing number of nodes. Because for the hyperbolic tangent this part is centred on zero the presence of this factor contributes also to decrease hidden neurons outputs.

The technique we propose, named Hidden Space Related Pseudoinversion (HSR-Pinv), combines together this choice of initialization intervals and activation functions with the pseudoinverse evaluation method described by eq. (11). It is compared with the commonly used technique that initialize weights uniformly in the interval $(-1, 1)$, and employs sigmoidal activation functions (named σ-SVD).

For the sake of comparison, we evaluate the pseudoinverse matrix using the different methods of evaluation presented in section 3, i.e. OP method, described by eq.(7), and ROP method described by eq.(10).

The performance reached by a standard SLFN neural network, with hyperbolic tangent as hidden neuron activation function trained using the backpropagation algorithm is the reference result. We initialized NN weights randomly according to a uniform distribution, looking for the minimum value of RMSE when the number of hidden nodes is gradually increased from 10 to 200. This range was selected according to the following three main reasons. On one side, previous investigations [3] show that the limiting performance of SLFN trained by backpropagation is achieved with a lower number of hidden neurons with respect to pseudoinversion methods. Besides, we trained some different models by varying the learning parameter η in the range (0.1, 0.9), as shown for a few cases in Fig. 1 (left), and the resulting RMSE level appears to reach an asymptotic value for all values of η quickly, using only a few ten hidden neurons. Finally, the iterative nature of backpropagation results in a heavy computational load, which also depends on the SLFN size.

Since we used a sufficiently large dataset for training and no overfitting arose, best results are achieved without the need for a regularization term, and learning consists then in minimizing the cost functional defined by eq. (5). With such a setting the best RMSE obtained is 3.81, with 90 hidden neurons.

Results are reported in Table 1. We record in the first row the best performance i.e. the best mean value over 10 simulation trials for all methods. For each tabulated value, the corresponding number of hidden neurons M_* used in the simulation is reported on the nearby column. The corresponding standard deviations are recorded in the second row and in the third row there is the optimal performance, i.e. the minimum value over 10 simulation trials.

Table 1. Comparison of performance on the test set

Method:	*HSR*-Pinv	M_*	σ-SVD	M_*	ROP	M_*	OP	M_*
Mean RMSE	1.77	573	8.52	88	7.69	90	9.43	60
St. Dev.	0.73	573	2.54	88	2.28	90	3.91	60
Min. RMSE	0.81	348	1.51	448	4.83	90	6.45	60

We remark that the best performance, in terms of both mean and minimum RMSE, is achieved by our proposed *HSR*-Pinv technique (first column). We also verified that the difference between *HSR*-Pinv mean RMSE value and the value obtained with backpropagation is significant within the 99% confidence interval and that the difference between *HSR*-Pinv min. RMSE value and σ-SVD min. RMSE value is significant within the 80% confidence interval.

Besides, our proposed *HSR*-Pinv technique achieves smaller variance, meaning a better behaviour with respect to over-fitting and it appears to be more able to take advantage of a larger hidden layer.

Fig. 1 (right) gives more insights on the evolution of the performance with the number of hidden units, showing the mean RMSE trend vs. number of hidden nodes (solid line); it is interesting to note that there is an error peak for a number of hidden neurons approximately equal to 250.

This error peak is related to the presence of singular values close to zero in matrix **S**, as discussed in section 3. This correlation is put in evidence by plotting also the ratio of the minimum singular value and the Matlab default

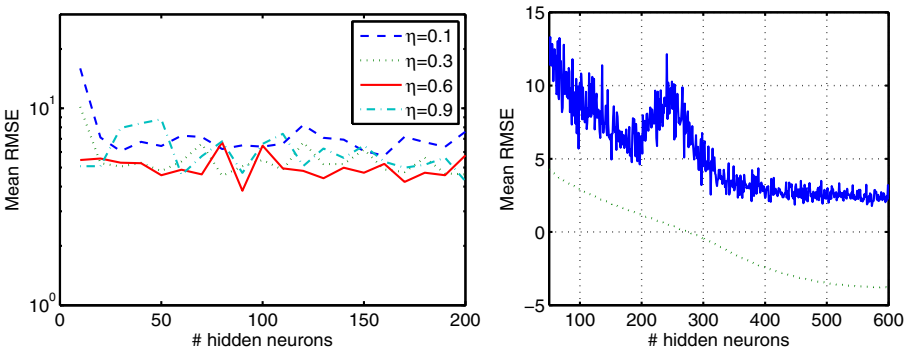

Fig. 1. Left: RMSE vs. number of hidden neurons for backpropagation training. Right: Mean RMSE vs. number of hidden neurons, and threshold ratio (logarithmic units).

threshold, below which singular values are considered too small (dotted line, using logarithmic units). The ratio approaches unity in the peak region.

This fact has potentially dramatic effects on performance, and must therefore be taken into account when SVD is used to evaluate $\mathbf{H}^{+}$, mainly because, as it happens in this case, the best performance can be reached with a number of hidden neurons larger than this critical dimension; the existence of this phenomenon has therefore to be known to avoid to early stop training in presence of error growth.

As a novel strategy to treat this problem we suggest a careful tuning of the threshold. We selected as a new threshold the mean value of the smallest singular values obtained over the 10 trials when using a hidden layer with 180 hidden neurons, because this is approximately the dimension of hidden neuron space when RMSE starts to increase.

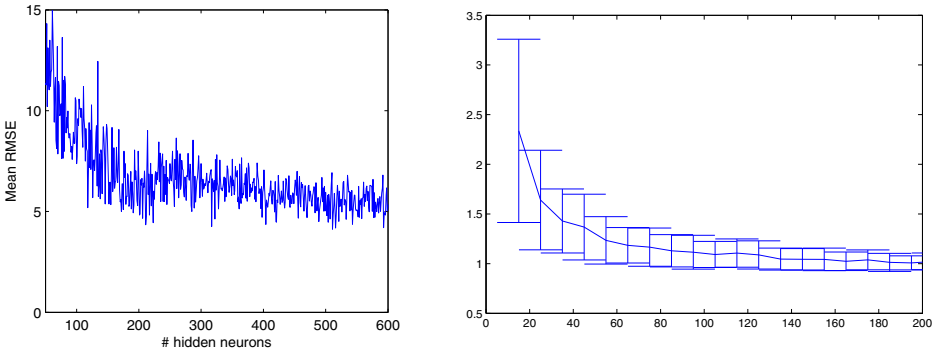

Fig. 2. Left: Mean RMSE vs. number of hidden neurons with threshold tuning. Right: Statistical analysis of parameter space

The effect of such a cut-off in fact is a sort of regularisation that provides a significant improvement in the method robustness, as can be seen from the relevant reduction of error peak, at the expense of a slight increase of RMSE values, as shown in Fig. 2 (left). Since the performance (mean RMSE = 4.12, St. Dev. = 0.83, min RMSE = 1.15) remains quite attractive, threshold tuning appears to be an interesting option. The above results show a clear advantage of the proposed strategy over standard training for SLFN. We provide some hints for a better understanding of such a phenomenon.

The actual model learning includes many random initialisations, at fixed network size, followed by the determination of corresponding output weights; a possible viewpoint is to keep the one that yields the best results, i.e. the optimal value recorded in the last row of Table 1.

Such a training phase, based on multiple random choices of weights, can be supposed to explore more extensively the parameter space, with respect to the trajectories followed by backpropagation algorithm, that develop continuously from a single random starting point. Yet this approach is reasonable provided

that the number of trials, i.e. the number of different random extractions and output weights evaluations, remains limited.

We verified the presence in the parameter space of many *good solutions* that can be reached also with a relatively small number of initial random choices. We first performed a set of 1000 different training trials. Then for a given subset size s, ranging from 10 to 200, we randomly built 100 subsets each containing s trials among the initial 1000 cases, in order to achieve statistical information. We then selected in each subset the minimum value of test error, and evaluated mean values and standard deviations of the distributions of such minima over the 100 subset instances. The test results are summarised in Fig.2 (right).

As expected, the mean value of the error distribution decreases with increasing number of trials; also, the range of variability decreases quickly, meaning that a limited number of trials is indeed required to attain good results, i.e. close to the best case. In our experiments, 100 trials are enough to reach an almost optimal performance, while 10 experiments already provide very good average results. This clearly demonstrates the method reliability, since good results are obtained even with a limited number of trials.

6 Conclusions

Starting from emerging strategies for training SLFN's using non iterative learning schemes, this paper proposes a novel approach based on original determination of the initialization interval for input weights, a careful choice of hidden layer activation functions and on prudent use of SVD to determine output weights.

We test this approach on a difficult astronomical regression problem, showing that we reach the best performance with respect to pseudoinversion by orthogonal projection, with or without regularization, and classical backpropagation.

We also put in evidence the presence of error peaks with increasing number of hidden units. We validate the hypothesis that this trend is due to numerical instability and we propose the adoption of threshold tuning in the SVD context in order to reduce the effects of singular values related peaks, to obtain a more robust learning scheme while still reaching very good results.

Acknowledgements. The activity was supported by Fondazione CRT, Torino, and Università degli Studi di Torino in the context of the World Wide Style Project.

References

1. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.-R.: Efficient BackProp. In: Orr, G.B., Müller, K.-R. (eds.) NIPS-WS 1996. LNCS, vol. 1524, pp. 9–50. Springer, Heidelberg (1998)
2. Halawa, K.: A method to improve the performance of multilayer perceptron by utilizing various activation functions in the last hidden layer and the least squares method. Neural Processing Letters 34, 293–303 (2011)

3. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Extreme Learning Machine: Theory and applications. Neurocomputing 70, 489–501 (2006)
4. Nguyen, T.D., Pham, H.T.B., Dang, V.H.: An efficient Pseudo Inverse matrix-based solution for secure auditing. In: IEEE International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (2010)
5. Kohno, K., Kawamoto, M., Inouye, Y.: A Matrix Pseudoinversion Lemma and its Application to Block-Based Adaptive Blind Deconvolution for MIMO Systems. IEEE Transactions on Circuits and Systems I 57(7), 1449–1462 (2010)
6. Ajorloo, H., Manzuri-Shalmani, M.T., Lakdashti, A.: Restoration of Damaged Slices in Images Using Matrix Pseudo Inversion. In: 22th International Symposium on Computer and Information Sciences, Ankara, pp. 98–104 (2007)
7. Bini, D., Capovani, M., Menchi, O.: Metodi numerici per l'algebra lineare. Ed. Zanichelli, Bologna (1988)
8. Bishop, C. M.: Pattern Recognition and Machine Learning. Ed. Springer, Berlin (2006)
9. Bartlett, P.L.: The sample complexity of pattern classification with neural networks: the size of the weights is more important that the size of the network. IEEE Trans. Inf. Theory 44(2), 525–536 (1998)
10. Ortega, J.M.: Matrix Theory. Plenum Press, New York (1987)
11. Golub, G., van Loan, C.: Matrix computations. The Johns Hopkins University Press, London (1996)
12. Le Gall, J.Y., Saisse, M.: Chromatic Aberration of an All-Reflective Telescope. In: Instrumentation in Astronomy V, London. SPIE, vol. 445, pp. 497–504 (1983)
13. Perryman, M.A.C., et al.: GAIA - Composition, formation and evolution of the galaxy. Concept and technology study, Rep. and Exec. Summary. In: European Space Agency, ESA-SCI, Munich, Germany, vol. 4 (2000)
14. Gai, M., Cancelliere, R.: Neural network correction of astrometric chromaticity. MNRAS 362(4), 1483–1488 (2005)
15. Cancelliere, R., Gai, M.: Efficient computation and Neural Processing of Astrometric Images. Computing and Informatics 28, 711–727 (2009)
16. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: 13th International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia, Italy (2010)