Artificial Neural Networks and Fuzzy Logic for Recognizing Alphabet Characters and Mathematical Symbols

Giuseppe Airò Farulla[1], Tiziana Armano[2], Anna Capietto[2], Nadir Murru[2(✉)], and Rosaria Rossini[3]

[1] Department of Control and Computer Engineering, Politecnico di Torino, Corso Duca Degli Abruzzi 24, Torino 10129, Italy
giuseppe.airof@gmail.com
[2] Department of Mathematics, University of Turin, Via Carlo Alberto 10, Torino 10121, Italy
{tiziana.armano,anna.capietto,nadir.murru}@unito.it
[3] Center for Applied Research on ICT, Istituto Superiore Mario Boella, Via Pier Carlo Boggio 61, Torino 10138, Italy
rosaria.francesca.rossini@gmail.com

Abstract. Optical Character Recognition software (OCR) are important tools for obtaining accessible texts. We propose the use of artificial neural networks (ANN) in order to develop pattern recognition algorithms capable of recognizing both normal texts and formulae. We present an original improvement of the backpropagation algorithm. Moreover, we describe a novel image segmentation algorithm that exploits fuzzy logic for separating touching characters.

Keywords: Artificial neural networks · Fuzzy logic · Kalman filter · Optical character recognition

1 Introduction

Currently, InftyReader is the unique OCR that performs automatic recognition of both normal texts and formulae [20]. The pattern recognition algorithm used by the authors of InftyReader is based on support vector machine learning [20]. In this paper, we propose to base the pattern recognition algorithm on artificial neural networks (ANNs). ANNs can be successfully exploited for developing pattern recognition algorithms [3]. Recently, several studies have been pointed out on ANNs for the automatic recognition of characters in different alphabets, such as Latin [18], Arabic [16] and many others. However, it appears that ANNs have not been exploited for recognizing both alphabet characters and mathematical

R. Rossini—This work has been developed in the framework of an agreement between IRIFOR/UICI (Institute for Research, Education and Rehabilitation/Italian Union for the Blind and Partially Sighted) and Turin University.

K. Miesenberger et al. (Eds.): ICCHP 2016, Part I, LNCS 9758, pp. 7–14, 2016.
DOI: 10.1007/978-3-319-41264-1_1

symbols in printed documents yet, since the large amount of different patterns to recognize does not allow fast convergence of the training algorithms. In this paper, we address this problem by means of an original use of the Kalman filter (mainly based on the work of Murru and Rossini [14]), with the aim of improving the rate of convergence in the training of ANNs even in presence of a large amount of patterns that must be recognized. In particular we study the backpropagation (BP) algorithm. It is well–known that convergence of BP is heavily affected by initial weights [1]. Different initialization techniques have been proposed such as adaptive step size methods [17], partial least squares method [9], interval based method [19]. Other different approaches can be found, e.g., in [2,5]. However, random weight initialization is still the most used method also due to its simplicity. Thus, the study of new initialization methods is an important research field in order to improve application of neural nets and deepen their knowledge.

Within an OCR, pattern segmentation is a required step before pattern analysis and recognition. The most common character segmentation algorithms are based on vertical projection, pitch estimation or character size, contour analysis, or segmentation–recognition coupled techniques [13]. Several methods for separating touching characters have been developed, see, e.g., [7,10,12,15]. In presence of formulae, separation of touching characters is harder since cutting positions can occur vertically, horizontally and diagonally. Several methods perform differently according to the features of the characters and their selection and use is an art rather than a technique. In other words, features selection mainly depends on the experience of the authors. Thus, in this context, fuzzy logic can be very useful, since it is widely used in applications where tuning of features is based on experience and it can be preferred to a deterministic approach. In this paper, we propose a method that combines, by means of a fuzzy logic based approach, some state–of–the–art features usually exploited one at a time.

In Sect. 2, we explain a novel method for the initialization of weights that improves performances of the backpropagation algorithm in order to train neural nets in the field of pattern recognition. In Sect. 3, we present a fuzzy based approach for performig segmentation of touching characters. Finally, Sects. 4 and 5 are devoted to numerical results and conclusions, respectively.

2 An Improvement of the Backpropagation Algorithm

Let us consider a neural network with L layers. Let $N(i)$ be the number of neurons in the layer i, for $i = 1, ..., L$, and $w_{ij}^{(k)}$ be the weight of connection between the i–th neuron in the layer k and the j–th neuron in the layer $k-1$. An ANN is trained over a set of inputs so that it provides a fixed output for a given training input. Let us denote X the set of training inputs. An element $\mathbf{x} \in X$ is a vector (e.g., a string of bits representing a pattern). Let $a_i^{(k,\mathbf{x})}$ be the output of the i–th neuron in layer k when an input $\mathbf{x}$ is processed by the ANN. This output is computed as follows:

$$\begin{cases} a_i^{(1,\mathbf{x})} = f(x_i) \\ a_i^{(k,\mathbf{x})} = f\left(\sum_{j=1}^{N(k-1)} w_{ij}^{(k)} a_j^{(k-1,\mathbf{x})}\right), \quad k = 2, ..., L, \end{cases} \tag{1}$$

where f is the activation function (usually, the sigmoidal or hyperbolic tangent function). Finally, let $\mathbf{y}^{(\mathbf{x})}$ be the desired output of the neural network corresponding to the input $\mathbf{x}$. In other words, we would like that $\mathbf{a}^{(L,\mathbf{x})} = \mathbf{y}^{(\mathbf{x})}$, when neural net processes input $\mathbf{x}$. Clearly, this depends on weights $w_{ij}^{(k)}$ and it is not possible to know their correct values a priori. Thus, it is usual to randomly initialize values of weights and use a training algorithm in order to adjust their values. One of the most common and used training algorithm is the BP. In [14], Murru and Rossini proposed an original approach for the initialization of weights mainly based on a customization of the Kalman filter through a Bayesian approach, in order to improve performances of BP algorithm. The Kalman filter is a well–established technique to estimate the state $\mathbf{w}_t$ of a dynamic process at each time t. Specifically, the Kalman gain matrix balances prior estimations and measurements so that an estimation is provided as follows: $\tilde{\mathbf{w}}_t = \mathbf{w}_t^- + K_t(\mathbf{m}_t - \mathbf{w}_t^-)$, where $\mathbf{m}_t$ is a measurement of the process, $\mathbf{w}_t^-$ a prior estimation of the process, K_t the Kalman gain matrix. As in [14], we customize the Kalman filter modeling measurements and prior estimations by means of multivariate normal random variables $\mathbf{W}_t$ and $\mathbf{M}_t$ such that their density functions satisfy $g(\mathbf{W}_t) = \mathcal{N}(\mathbf{w}_t^-, Q_t)$, $g(\mathbf{M}_t|\mathbf{W}_t) = \mathcal{N}(\mathbf{m}_t^-, R_t)$, where Q_t and R_t are covariance matrices conveniently initialized. In this way, the posterior density is given by $g(\mathbf{W}_t|\mathbf{M}_t) \propto \mathcal{N}(\mathbf{w}_t^-, Q_t)\mathcal{N}(\mathbf{m}_t, R_t) = \mathcal{N}(\tilde{\mathbf{w}}_t, P_t)$, where $\tilde{\mathbf{w}}_t = (Q_t^{-1} + R_t^{-1})^{-1}(Q_t^{-1}\mathbf{w}_t^- + R_t^{-1}\mathbf{m}_t)$, $P_t = (Q_t^{-1} + R_t^{-1})^{-1}$. We can apply this technique to weights initialization considering processes $\mathbf{w}_t(k)$, for $k = 2, ..., L$, as non–time–varying quantities whose components are the unknown values of weights $w^{(k)}$, for $k = 2, ..., L$, of the neural net such that $\mathbf{a}^{(L,\mathbf{x})} = \mathbf{y}^{(\mathbf{x})}$. The goal is to provide an estimation of initial weights to reduce the number of steps that allows convergence of BP neural net. Thus, for each set $w^{(k)}$ we consider initial weights as unknown processes and we optimize randomly generated weights (which we consider as measurements of the processes) with the above approach. In these terms, we derive an optimal initialization of weights by means of the following equations:

$$\begin{cases} \mathbf{m}_t = Rnd(-h, h) \\ (R_t)_{ii} = \dfrac{1}{N(k)N(k-1)} \sum_{\mathbf{x}\in X} \|\mathbf{d}^{(k,\mathbf{x})}\|^2, \quad (R_t)_{lm} = 0.7 \\ \tilde{\mathbf{w}}_t = (Q_t^{-1} + R_t^{-1})^{-1}(Q_t^{-1}\mathbf{w}_t^- + R_t^{-1}\mathbf{m}_t) \\ Q_{t+1} = (Q_t^{-1} + R_t^{-1})^{-1}, \quad \mathbf{w}_{t+1}^- = \tilde{\mathbf{w}}_t \end{cases} \tag{2}$$

where $Rnd(-h, h)$ denotes the function that samples a random real number in the interval $(-h, h)$ and $\mathbf{d}^{(k,\mathbf{x})}$ is the usual error of the k–th layer when an input $\mathbf{x}$ is given.

3 Image Segmentation with Fuzzy Logic

In a binarized image, a pattern can be represented by a matrix whose entries are 0 (white pixels) and 1 (black pixels). Generally, methods for segmenting touching characters define a function based on some features that characterize cut positions. Then, such a function is evaluated for each column (row, or diagonal) of the matrix and the cut position is chosen depending on its values. Classical functions of this kind are the peak–to–valley function g and the function h defined as

$$g(i) = \frac{V(l_i) - 2V(i) + V(r_i)}{V(i) + 1}, \quad h(i) = \frac{V(i-1) - 2V(i) + V(i+1)}{V(i)}, \tag{3}$$

where $V(i)$ denotes the vertical projection function for the i-th column (row or diagonal), l_i and r_i are the peak positions on the left side and right side of i, respectively. Let us denote by $\bar{g}$ and $\bar{h}$ the functions g and h normalized to $[0, 1]$, respectively. In the following, we will consider $\tilde{g} = 1 - \bar{g}$ and $\tilde{h} = 1 - \bar{h}$. Here, we propose a fuzzy routine typically identifying a column, a row, or a diagonal of the matrix that could be a cut point that conveniently separates the touching characters.

Our fuzzy routine combines four state–of–the–art features: distance from the center of the pattern; crossing count, i.e., the number of transitions from white to black pixels, and viceversa; the function $\bar{g}$; the function $\bar{h}$. These features are widely used in literature for determining cut positions in touching characters [12]. However, they are usually managed separately and in a deterministic way. In our approach, these features are combined by means of convenient fuzzy rules in order to exploit the information given by each of these features.

Given a pattern in a binarized image, let A, m, n, and c be the matrix of pixels of the binarized matrix, the number of rows of A, the number of columns of A, and the central column of A, respectively. For the sake of simplicity, in the following we only focus on columns of A and when we refer to a column i of A, we refer to the i–th column of A, i.e., we are considering the vector of length m whose elements are the entries of the i–th column. For each column i of A, we define its normalized distance from the center of the pattern as $d(i) = \frac{|c-i|}{c}$. Moreover, we indicate the crossing count function by f, i.e., the number of transitions between white and black pixels.

To design a suitable fuzzy strategy, some steps are required, in order to introduce the notion of a fuzzy degree qualifying a column i to be a cut position: for short $\rho = \rho(i) \in [0, 1]$. In our model, the low values of ρ locate good cut positions. The strategy can be detailed by means of the fuzzification of the functions d, f, $\tilde{g}$, $\tilde{h}$. Figure 1a, b and c show the fuzzy sets and the related membership functions. Note that we have considered the same fuzzy sets and membership functions for $\tilde{g}$, $\tilde{h}$, and ρ.

For each column i of A, the inference system combines the values $d(i)$, $f(i)$, $\tilde{g}(i)$, $\tilde{h}(i)$ and produces the fuzzy output $\rho(i)$ by means of the following fuzzy rules:

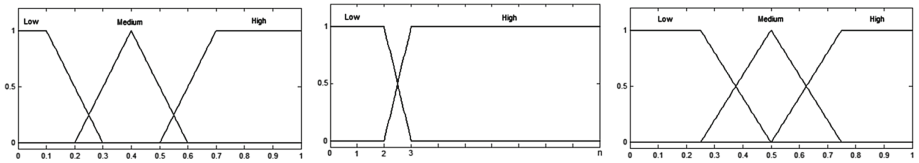


Fig. 1. Membership functions of the fuzzy sets related to, respectively, (a) d, (b) f, (c) $\tilde{g}$, $\tilde{h}$, and ρ.

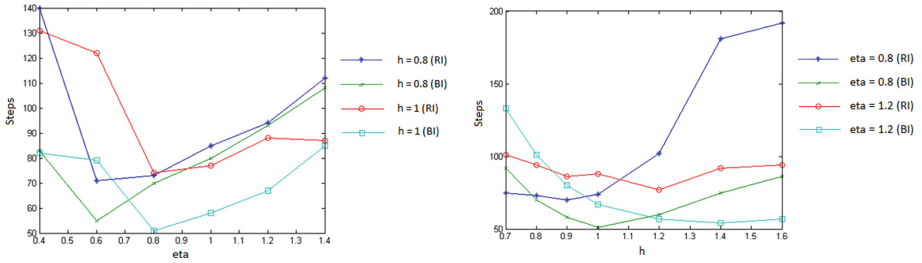
1. if $d(i)$ is Low and $\tilde{g}(i), \tilde{h}(i)$ are not High and $f(i)$ is Low, then $\rho(i)$ is Low;
2. if $\tilde{g}(i), \tilde{h}(i)$ are Low and $d(i)$ is Medium and $f(i)$ is Low, then $\rho(i)$ is Low;
3. if $\tilde{g}(i)$ is Low and $d(i)$ is not High and $\tilde{h}(i)$ is not Low and $f(i)$ is Low, then $\rho(i)$ is Low;
4. if $d(i)$ is Low and $\tilde{g}(i), \tilde{h}(i)$ are not High and $f(i)$ is High, then $\rho(i)$ is Medium;
5. if $\tilde{g}(i), \tilde{h}(i)$ are Low and $d(i)$ is Medium and $f(i)$ is High, then $\rho(i)$ is Medium;
6. if $\tilde{g}(i)$ is Low and $d(i)$ is not High and $\tilde{h}(i)$ is not Low and $f(i)$ is High, then $\rho(i)$ is Medium;
7. if $\tilde{h}(i)$ is Low and $d(i)$ is not High and $g(i)$ is not Low and $f(i)$ is Low, then $\rho(i)$ is Medium;
8. if $d(i), \tilde{g}(i), \tilde{h}(i)$ are Medium and $f(i)$ is Low, then $\rho(i)$ is Medium;
9. otherwise $\rho(i)$ is High.

These fuzzy rules have been tuned using heuristic criteria taking into account that high values of g, h and low values of d, f usually identify cut positions. The inference engine is the basic Mamdani model with if–then rules, minimax set–operations, sum for composition of activated rules and defuzzification based on the centroid method. The Mamdani model is congenial to capture and to code expert-based knowledge.

4 Numerical Results

In this section, we describe the process to train neural networks in order to recognize both characters and mathematical symbols, comparing the rate of convergence of the BP algorithm with the Bayesian initialization (BI) presented in Sect. 2 against classical random initialization (RI).

Firstly, we use a training set composed by 26 Latin printed characters for 7 different fonts (Arial, Cambria, Courier, Georgia, Tahoma, Times New Roman, Verdana), 24 Greek letters, and 35 miscellaneous mathematical symbols, with 12 pt. In Fig. 2a, b and c performances of BP algorithm with BI and RI are compared for different values of h and η, where $(-h, h)$ is the interval where weights are sampled and η is the learning rate of the BP algorithm. The improvement in convergence rate due to BI is noticeable at a glance in these figures. In particular, we can see that BI approach is more resistant than RI with respect to high values of h. In fact, for large values of h, weights can range over a large



(a) (b)

Fig. 2. Comparison between BI and RI for convergence of BP neural net with $L = 3$, $N(1) = 315$, $N(2) = 100$, $N(3) = 85$, and hyperbolic tangent activation function

interval. Consequently, RI produces weights scattered on a large interval causing a slower convergence of BP algorithm. On the other hand, BI seems to set initial weights on regions that allow a faster convergence of BP algorithm, despite the size of h. This could be very useful in complex problems where small values of h do not allow convergence of BP algorithm and large intervals are necessary, as the case of the realization of an OCR for both text and formulae, where the number of different patterns for training the neural net is very high and large values of h are necessary. Indeed, we can observe in our simulations that the best performances are generally obtained by BI with large values of h.

In Table 1, we have trained neural networks on the MNIST training set [11], composed by 60000 handwritten digits. Note that in the case of the MNIST database, if training is accomplished over all the training dataset, then BP algorithm for multilayer neural networks yields a very high accuracy in the recognition on the MNIST validation set composed by 10000 handwritten digits (more than 99 %, see, e.g., [4]).

The computational complexity to implement the classical Kalman filter is polynomial (see, e.g., [6] p. 226). Our customization is faster since it involves

Table 1. Comaprison between BI and RI for convergence of BP neural net trained on the MNIST database

	$L = 5$, $\eta = 1.5$		$L = 3$, $\eta = 3$	
	Random in.	Bayes in.	Random in.	Bayes in.
h	Steps	Steps	Steps	Steps
0.7	832	809	874	868
0.8	823	812	652	631
0.9	748	696	722	706
1	749	671	688	564
1.1	961	929	803	658
1.2	1211	1118	967	872

fewer number of operations (matrix multiplications) than usual Kalman filter and we use circulant matrices, whose inverse can be evaluated in a very fast way. Indeed, these matrices can be diagonalized by using the Discrete Fourier Transform ([8], p. 32); the Discrete Fourier Transform and the inverse of a diagonal matrix are immediate to evaluate. Thus, our initialization algorithm is faster than classical Kalman filter, moreover it is iterated for a low number of steps (usually twice). Surely, this approach has a time complexity greater than random initialization. However, looking at BP Algorithm, we can observe that Eq. 2 involve similar operations (i.e., matrix multiplications or multiplications between matrices and vectors) in a minor quantity; moreover it needs a smaller number of cycles. Furthermore, we have seen that BI generally leads to a noticeable decrease of steps necessary for the convergence of the BP algorithm with respect to random initialization. Thus, using BI we can reach a faster convergence, in terms of time, of the BP algorithm than using random initialization.

Finally, we report the results on the segmentation of 296 touching characters by means of the fuzzy method explained in the previous section. The dataset is composed by 66 touching characters font Verdana and 10 pt, 58 with font Times New Roman and 20 pt, 92 font Lucida and 25 pt, 40 Georgia and 20 pt, 54 Cambria and 20 pt. Our method correctly segments the 93.6 % of these touching characters, improving performances obtained by using functions g and h one at a time that perform a correct segmentation in 76.5 % and 71.1 % of cases, respectively.

5 Conclusion

We propose the development of an OCR able to recognize both alphabet characters and mathematical symbols. The pattern recognition algorithm is based on ANNs trained by means of an improvement of the backpropagation algorithm. The image segmentation algorithm is based on a fuzzy routine that combines some features usually exploited one at a time. Note that we are not proposing an OCR already completed and tuned. The proposed experimental results and simulations show that the approaches presented in this paper introduce improvements and benefits if compared with existing standard achievements.

Future works will deal with an extensive validation of the proposed system with visually impaired and blind subjects. Moreover, we will develop a novel architecture for character recognition, based on an array of ANNs which are applied in parallel to a given input pattern. This choice will enable us to reach better accuracy with respect to more traditional approaches based on a single ANN, with at the same time no appreciable degradation on performances.

References

1. Adam, S., Karras, D.A., Vrahatis, M.N.: Revisiting the problem of weight initialization for multi-layer perceptrons trained with back propagation. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008, Part II. LNCS, vol. 5507, pp. 308–315. Springer, Heidelberg (2009)

2. Adam, S.P., Karras, D.A., Magoulas, G.D., Vrahatis, M.N.: Solving the linear interval tolerance problem for weight initialization of neural networks. Neural Netw. **54**, 17–37 (2014)
3. Bishop, C.M.: Neural Networks for Pattern Recognition. Clarendon Express, Oxford (1995)
4. Cireşan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Deep big multilayer perceptrons for digit recognition. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 7700, 2nd edn, pp. 581–598. Springer, Heidelberg (2012)
5. Erdogmus, D., Romero, O.F., Principe, J.C.: Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response. IEEE Trans. Neural Netw. **16**(2), 325–336 (2005)
6. Hajiyev, C., Caliskan, F.: Fault Diagnosis and Reconfiguration in Flight Control Systems. Springer, Heidelberg (2003)
7. Garain, U., Chaudhuri, B.B.: Segmentation of touching symbols for OCR of printed mathematical expressions: an approach based on multifactorial analysis. In: 8th International Conference in Documents Analysis and Recognition, pp. 177–181 (2005)
8. Gray, R.M.: Toeplitz and circulant matrices: a review. Found. Trends Commun. Inf. Theor. **2**(3), 155–239 (2006)
9. Hsiao, T.C., Lin, C.W., Chiang, H.K.: Partial least squares algorithm for weight initialization of backpropagation network. Neurocomputing **5**, 237–247 (2003)
10. Kumar, A., Yadav, M., Patnaik, T., Kumar, B.: A survey on touching charcter segmentation. Int. J. Eng. Adv. Technol. **2**(3), 569–574 (2013)
11. Lecun, Y., Cortes, C., Burges, C.J.C.: The MNIST database of handwriteen digits. http://yann.lecun.com/exdb/mnist
12. Liang, S., Shridhar, M., Ahmadi, M.: Segmentation of touching characters in printed document recognition. Pattern Recogn. **27**(6), 825–840 (1994)
13. Lu, Y.: Machine printed character segmentation - an overview. Pattern Recogn. **28**(1), 67–80 (1995)
14. Murru, N., Rossini, R.: A Bayesian approach for initialization of weights in back-propagation neural net with application to character recognition. Neurocomputing (2016, to appear)
15. Saba, T., Sulong, G., Rahim, S., Rehman, A.: On the segmentation of multiple touched cursive characters: a heuristic approach. In: Das, V.V., Vijaykumar, R. (eds.) ICT 2010. CCIS, vol. 101, pp. 540–542. Springer, Heidelberg (2010)
16. Sahlol, A.T., Suen, C.Y., Elbasyouni, M.R., Sallam, A.A.: A proposed OCR algorithm for the recognition of handwritten Arabic characters. J. Pattern Recogn. Intell. Syst. **2**(1), 8–22 (2014)
17. Schrusolph, N.N.: Fast curvature matrix-vector products for second order gradient descent. Neural Comput. **14**(7), 1723–1738 (2002)
18. Shrivastava, V.: Artificial neural networks based optical character recognition. Sig. Image Process.: Int. J. **3**(5), 73–80 (2012)
19. Sodhi, S.S., Chandra, P.: Interval based weight initialization method for sigmoidal feedforward artificial neural networks. AASRI Procedia **6**, 19–25 (2014)
20. Suzuki, M., Kanahori, T., Ohtake, N., Yamaguchi, K.: An integrated OCR software for mathematical documents and its output with accessibility. In: Miesenberger, K., Klaus, J., Zagler, W.L., Burger, D. (eds.) ICCHP 2004. LNCS, vol. 3118, pp. 648–655. Springer, Heidelberg (2004)