Automatic Prediction of Poisonous Mushrooms by Connectionist Systems

María Navarro Cáceres and María Angélica González Arrieta

University of Salamanca, Spain

Abstract. The research offers a quite simple view of methods to classify edible and poisonous mushrooms. In fact, we are looking for not only classification methods but also for an application which supports experts' decisions. To achieve our aim, we will study different structures of neural nets and learning algorithms, and select the best one, according to the test results.

1 Introduction

Fungus are well-known for centuries, at least in their most spectacular manifestations, the mushrooms. Old city Micenas might be called like this due to a fungus: *mykes* in Greek language. Micology, the science which studies fungus, derives from this term *mykes* as well [1].

Humanity interest in fungus is rather old. Our ancestors observed and used the fermentation process. Without this, we would have no bread, wine or beer, among others. About knowledge and use of mushrooms, we have very old evidence, like Pompeya where milk caps are shown in pictures, or the antique tradition of using hallucinogenic mushrooms in some ritual practices [2].

Nowadays, the support for mushroom and fungus is becoming more and more widespread due to the increasing interest in nature. A large amount of mycologic societies have appeared during the last years[3], as well as activities related to them (such as exhibitions or excursion to identify mushrooms). On the other hand, its utility in Biotechnology is also well-known, above all because they are the origin of antibiotics [4].

In this research we propose a way to automate mushrooms classification process. There are more and more people who wants to collect them, but lots of them have to consult an expert to check how many mushrooms are edible. Sometimes, this expert might be wrong, what might endanger health or even the human life [5]. In order to improve this identification system and reduce this error, we will use neural nets, MultiLayer Perceptron (MLP) and Base Radial Function Net(BRF), to classify mushrooms.

In related works about mushroom classification, the authors use data mining techniques due to the large amount of data [6, 7]. In other cases a new algorithm is

S. Omatu et al. (Eds.): *Distrib. Computing & Artificial Intelligence,* AISC 217, pp. 341–349.
DOI: 10.1007/978-3-319-00551-5_42

developed so that the classification can be done [8]. There is an interesting work about artificial vision to recognize different features of a mushroom although uses as well a data mining classification techniques [9]. The main problem of these ones is that the error is not always 0%. Then, we considered whether a neural net solution may give better results in classification problem, at least to classify them only in poisonous or edible ones.

1.1 Theoric Concepts

As we discuss, in this project we used two types of neural nets: supervised ones and hybrid ones. Next, we will describe both briefly.

1.1.1 Perceptron (MLP)

Multilayer Perceptron is a neural net with multiple layers. It can resolve non-linear separable problems (problem XOR), what is the main problem of simple perceptron. MLP can be totally or partially connected (among neurons)[10].

Backpropagation is an algorithm used to train these nets. Due to its properties, this net is one of the most used neural nets to solve problems.[11] Its output is function of the connection weights, a threshold value and input vectors.

Nonetheless, the problem of local minimums in error functions [12] makes difficult the training process, as once a minimum is reached, the algorithm always stops. A possible solution is change the topology, the initials weights or the order in which data training input was presented[10].

1.1.2 Base Radial Function Net (BRF)

BRF are a hybrid model which uses supervised and non-supervised learning. [10]It allows to make no lineal systems modelings. They always have three layers: Input (to transfer data), output (it uses a linear function) and hidden layer (uses Gaussian function to give an output.) The output of hidden layer depends on the distance between centroid and input vector [13, 14]. In this case, the response is not global, because neurons only give an output if distance between centroid and input vector is small enough.

However, topology of these nets depends on size and shape of input data. The number of neurons can be assigned randomly or using an algorithm to add neurons till a threshold error was achieved. Besides, we usually train first hidden layer to determine centroid values, using k-means algorithm [15], and after output layer, with a backpropagation algorithm.

2 Materials and Methods

2.1 BRF versus MLP

BRF and MLP are feedfowarding nets organised in layers, and both can aproximate any continuous function. As regards differences between BRF and MLP, we can note

that BRF have one hidden layer, whereas MLP can have several of them. Neurons in BRF have no weights assigned in connections between input layer and hidden layer. Besides, the activation function of the output layer is always a linear function. However, the main difference is the function of activation in the hidden layer. In BRF we use a base radial function (like Gaussian function), which provokes that each neuron is activated in one zone of the input space data. On the other hand, MLP uses a sigmoid function, which permits each neuron was activated with each input. Due to this features, we can conclude that:

1. MLP gets global relationships between input and output. Because of this, learning is slower, [15] as the change of only one weight provokes changes in the output for all the input data.
2. BRF gets local relationships. Each element of the hidden layer specializes in a certain region of the outputs. So, this provokes a more quick learning, because the change of one weight only affects to the processing element associated to this weight [16], and also to a certain group of inputs, belonging to the class which the neural element affected is representing.

2.2 Data Processing

Data have been obtained from `http://archive.ics.uci.edu/ml/datasets/Mushroom` It is a database of 8124 registries, with 22 attributes of nominal type. The attributes are:

Table 1 Studied attributes

cap-shape	cap-surface
cap-color	bruises
odor	gill-attachment
gill-spacing	gill-size
gill-color	stalk-shape
stalk-root	stalk-surface-below-ring
stalk-surface-above-ring	stalk-color-above-ring
stalk-color-below-ring	veil-type
veil-color	ring-number
ring-type	spore-print-color
population	habitat

All the attributes were nominal type, so we have to convert it into numerical type so as the neural net could process them. The solution is to turn nominal data into binary data as we show in the table with the example: odor-attribute.

We have the next values for odor: *almond, anise, none, fishy, foul, spicy, musty, creosote* and *pungent*. If we have odor = *almond*, we turn it into a numerical value, as shown in Table 2.2.

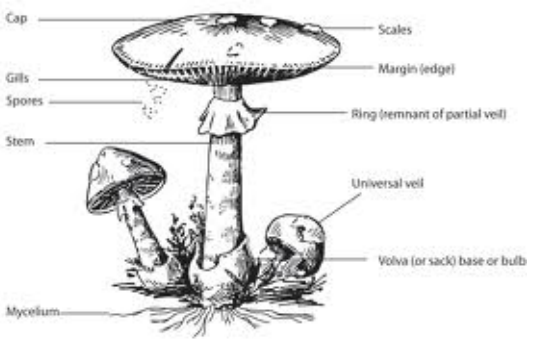

Fig. 1 Parts of a general mushroom

Table 2 Example of nominal data transformation

Odor-p	Odor-a	Odor-l	Odor-y	Odor-f	Odor-n	Odor-c	Odor-m	Odor-s
0	1	0	0	0	0	0	0	0

With this method, we got more than 50 values for each input registry, too much to process. Thats why we began to make tests to remove unnecessary attributes. After some inquiries and tests in MATLAB, we decided to use only 5 attributes: odor, stalk-surface, spore-color, bruises and gill-size.

2.3 Process Modelling

We used MATLAB to carry out this research. The simple is stored in an Excel file retrieved by MATLAB [17]and saved in two matrix: one for inputs and one for targets. We used rows from 1 to 6000 as training data, and from 6001 to 8124 as test data. We also implement a function which chose a certain number of random values from this matrix, in order to make different studies about the data and the results. Our MLP has one hidden layer with 60 neurons. On the other hand, BRF is automatically created by adding neurons in the hidden layer until threshold error was achieved. So, it is important not to have many training data to avoid the non-convergence of the threshold error established by the user.

3 Results

In order to study the different neural structures behavior, we put into practice them to predict edible or poisonous mushroom. We got different results, that we will discuss now. In both MLP and RBF cases, we carried out two experiments.

3.1 MLP Results

3.1.1 Training Data

Training sample size influences notably in the neural network precision. With a small sample, neurons are not capable of classifying test data right. As in Base Radial Network as in MLP, we used 100, 1000 and 6000 training registries. In MLP, 100 registries are not enough to achieve a minimal error. In fact, mean error obtained is 0.9135. The Figure 2 shows the classification results of the test sample. It is not an acceptable result at all.

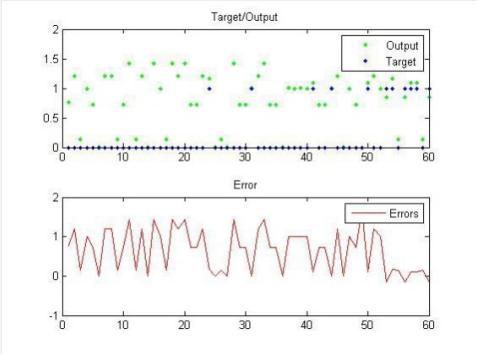

Fig. 2 MLP results with 100 training input registries

Lets see with 1000 data. Results shown in Figure 3 In this case, error is 0.7532,

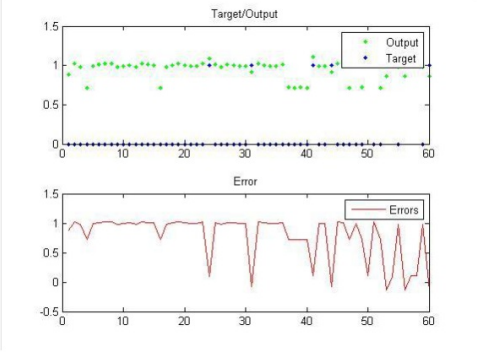

Fig. 3 MLP results with 1000 training input registries

not yet aceptable. However, with 6000 data, MLP network has a mean error of 0.0092. By inserting random data test, the results are quite positive, as is shown in the Figure 4.

In conclusion, the larger training simple, the better to train the neural net.

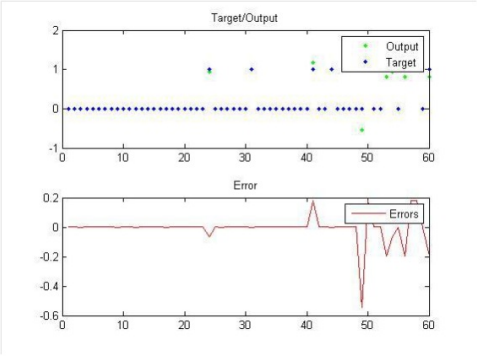

Fig. 4 MLP results with 6000 training input data and 60 neurons

3.1.2 Number of Neurons

We verify that by increasing the number of neurons, the error was reduced to some extent in which error can even increase again. With 10 neurons in hidden layer, error was about 0.7281. With 60 neurons, the error was 0.0094. (See Figure 4) On the other hand, with 200 neurons in the hidden layer, training time was increased and mean error was quite high, 0.7712.

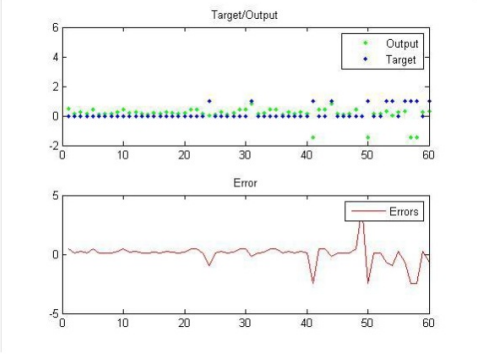

Fig. 5 MLP results with 10 neurons

3.2 RBF

3.2.1 Training Data

Threshold error was established in 10^{-3} to create the net. By inserting 100 training data, the network created has 99 neurons. The output and target data is shown in Figure 6.

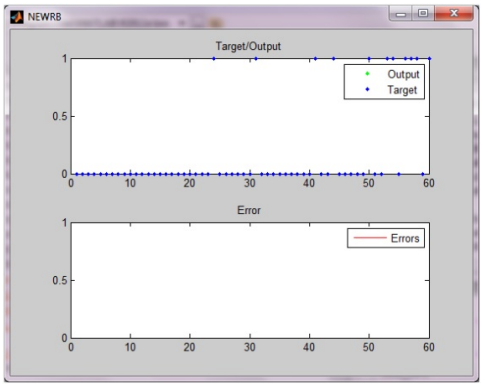

Fig. 6 RBF results with 100 training data (Threshold error is 10^{-3})

If we insert 500 values, the net never converge, because there are too data for the net to generalize, as we can see in Figure 7.

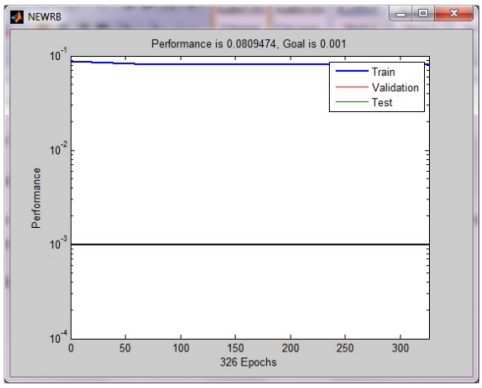

Fig. 7 RBF training process with 500 training data (threshold error is 10^{-3}) Convergence is not possible, at least in a reasonable period of time

With a simple of 100 values, this net predict the output very well. In fact, the error obtained in this case, was about 0.

3.2.2 Threshold Error

First of all, we put an error of 0.1. The number of neurons has been reduced (only 21). However, the difference between data output and target output was increased. The mean error is about 0.2472. The lower error we want, the more number of neurons we need. Perhaps, if we search a minimal error, the net may not ever converge, like in the excess of data situation.

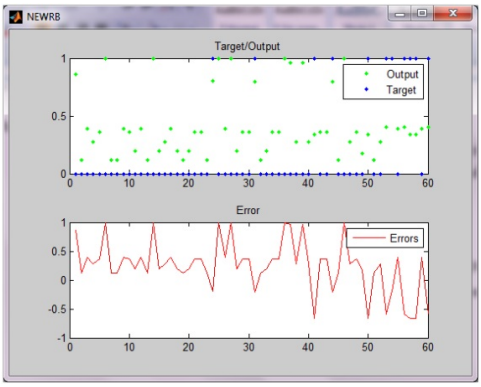

Fig. 8 RBF results. Threshold error set 0.1

4 Conclusions

In this research, two methods were studied in order to achieve the main purpose: recognize an edible mushroom. The first, based on supervised learning, gave acceptable results, but only if the training sample was quite large. On the other hand, Base Radial Network works properly only with a short training sample, due to its k-means training algorithm [15], and in general, because of its hybrid learning.

In view of the results obtained, and bearing in mind the error minimization in BRF with a reasonable number of neurons, we can conclude that BRF is a better model to achieve the best result in our studied case. Nevertheless, MLP will be much more appropriate in case we have a training sample quite large, although in our research the error function was not minimal.

Yet, the application has to be tested with another data. We can also consider to improve the MLP structure to achieve better results with an error of almost 0. Its functionality can be extended by classifying not only in edible or poisonous mushrooms, but also in its own species. In this hypothetical case, we may use another training algorithm, or even another learning mechanism which provokes a change in the neural structure. This new changes have to be chosen carefully, as the error has to be 0% in the majority of the situations in order to guarantee the reliability of this application. Also as a future work, it is important to visualize results in a properly way for the final user.

References

1. Alexopoulos, C.J.Y., Mins, C.W.: Introducción a la Micología. Omega, Barcelona (1985)
2. Ruiz Herrera, J.: El asombroso reino de los hongos. Avance y Perspectiva. Colección Micológica, Herbario de la Facultad de Biología (2001)
3. Micology experts and amateurs, `http://www.micologia.net/`
4. Palazó, F.: Setas para todos. Pirineo, Castellano (2001)

5. García Rollán, M.: Setas venenosas: intoxicaciones y prevención. Ministerio de Sanidad y Consumo, Madrid, Castellano (1990)
6. Chai, X., Deng, L., Yang, Q., Ling, C.X.: Test-Cost Sensitive Naive Bayes Classification (2003)
7. Bay, S.D., Pazzani, M.J.: Detecting Group Differences: Mining Contrast Sets. Data Min. Knowl. Discov. 5 (2001)
8. Li, J., Dong, G., Ramamohanarao, K., Wong, L.: DeEPs: A New Instance-based Discovery and Classification System. In: Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases (2001)
9. Matti, D., Vajda, P., Ebrahimi, T.: Mushroom Recognition (2010)
10. Hayking, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall (1998)
11. Bishop, M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
12. Jaynes, E.T.: Probability Theory: The Logic of Science. Cambridge University Press (2003)
13. Tao, K.K.: A closer look at the radial basis function (RBF) networks. In: Singh, A. (ed.) Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems, and Computers. IEEE Comput. Soc. Press, Los Alamitos (1993)
14. Fine, T.L.: Feedforward Neural Network Methodology, 3rd edn. Springer, Nueva York (1999)
15. MacKay, D.J.C.: Information Theory, Inference and Learning Algorithms. Cambridge University Press (2004)
16. Alonso, G., Becerril, J.L.: Introducción a la inteligencia artificial. Multimedia Ediciones, S.A. Barcelona (1993)
17. Matlab Official Site, `http://www.mathworks.es`