Training Neural Networks Using Input Data Characteristics

Cosmin CERNĂZANU
"Politehnica" University of Timişoara
Vasile Pârvan St., No. 2, Timişoara
cosmin.cernazanu@ac.upt.ro

***Abstract*—Feature selection is often an essential data processing step prior to applying a learning algorithm. The aim of this paper consists in trying to discover whether removal of irrelevant and redundant information improves the performance of neural network training results.**

The present study will describe a new method of training the neural networks, namely, training neural networks using input data features. For selecting the features, we used a filtering technique (borrowed from data mining) which consists in selecting the best features from a training set. The technique is made up of two components: a feature evaluation technique and a search algorithm for selecting the best features.

When applied as a data preprocessing step for one common neural network training algorithms, the best data results obtained from this network are favorably comparable to a classical neural network training algorithms. Nevertheless, the first one requires less computation.

***Index Terms*—neural networks, data mining, correlation-based feature subset selection method, data features extraction, training algorithm**

I. FEATURE SELECTION AND NEURAL NETWORKS

An artificial neural network (ANN) is a model that emulates a biologic neural network. An ANN is made up of thousands of artificial neurons; elements of non-linear processing that operate in parallel. [1]

The main characteristics of the neural networks are the same with those of the human brain, that is:

- capacity of learning
- capacity of generalizing

If trained adequately, the artificial neural networks would be capable of providing correct answers even for the set-entries different from those they have already been used to, as long as they do not differ too much. This generalization is made automatically as a result of their structure and not as a result of human intelligence which is included in a program as in the case of the expert systems. [2]

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. [3] A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections in given new situations of interest and answer "what if" questions.

Multilayered feedforward neural networks possess a number of properties which make them particularly suited to complex pattern classification problems. However, their application to some real world problems has been hampered by the lack of a training algorithm which can reliably find a nearly optimal set of weights in a relatively short time.

On that account, in the reference literature we can find a great number of techniques that help to improve the training speed for a feedforward multilayer neural network. These techniques can be grouped in several categories: techniques dealing with the computation speed (there have been generated techniques that run the algorithm on a network of computers [4]; techniques that run on GPU [5], etc.) and techniques dealing with optimizing the training algorithm (techniques of using the genetic algorithms for the initial generation of weights [6], techniques of initializing the weights by using previous information [7], etc.)

From the multitude of techniques which aim at optimizing the training algorithm, we will select for study only the techniques using the data mining concepts. Data mining is a field generally acknowledged for its notable performances in discovering certain patterns in enormous databases (KDD – Knowledge Discovery in Databases). A KDD procedure can be defined as „the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" and can be represented as in Fig. 1.

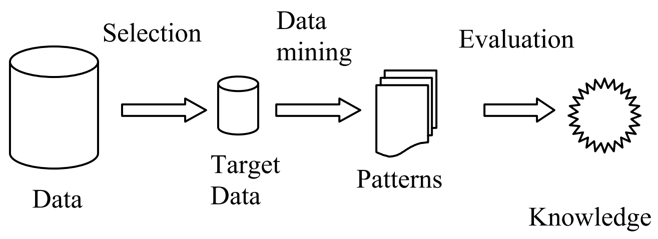

Figure 1. Overview of the steps constituting the KDD process.

The KDD process aims at selecting certain features from the input data so that the selected features become representative for the task to be completed [8]. The task of discovering regularities can be made easier and less time consuming by removing features of the data that are irrelevant or redundant with respect to the task to be learned. This process is called feature selection. Feature selection process is well defined and has the potential to be fully automatic and computationally tractable.

Feature selection algorithms must address 4 basic issues affecting the nature of the search [9]:

1. Starting point. We must start by selecting a point in the feature subset space which can affect the direction of the search. One option is to begin with no features and successively add feature (forward), other option is to begin with all features and successively remove them (backward) and the third option is to begin somewhere in the middle and move outwards from this point.
2. Search organization. Heuristic search strategies are more feasible than exhaustive ones and give good

results, although they do not guarantee finding the optimal subset.

3. Evaluation strategy. Is the biggest differentiating factor among feature selection algorithms and specifies how feature subsets are evaluated. We have two main directions for these strategies: one is algorithms which use heuristics based on general characteristics of the data to evaluate the merit of feature subset and the other argue that the bias of a particular induction algorithm should be taken into account when selecting features.
4. Stopping criterion. A feature selection must decide when to stop searching through the space of feature subset. For this issue we have the following possibilities: we can stop adding or removing features when none of the alternatives improves current feature subset, we can continue revise the current feature subset as long as the merit does not degrade or we can continue generating feature subset until reaching the opposite end of the search space and select the best.

Other authors mentioned as well the use of certain feature selection techniques from a database for the purpose of optimizing the process of solving a problem. We will continue our research by presenting some of the current feature selection techniques developed until now which have been successfully applied for solving certain ANN problems. In the chapter regarding the experiments, we will make a comparison between the results obtained by other authors (by using various feature selection techniques) and the results obtained by using the technique which is the subject of the present study.

In 1997 Jain and Zongker define the problem of feature selection as follows: given a set of candidate features, it must be selected a subset that performs the best under some classification systems. [10] They say that the procedure can reduce not only the cost of recognition by reducing the number of features that need to be collected, but in some cases it can also provide a better classification accuracy due to finite sample size effect. They study a large number of algorithms proposed for feature subset selection and chose the sequential forward floating selection (SFFS) developed by Pudil et all to be the best. [11]

In 2003, Kim and Street advance a new method named ELSA (Evolutionary Local Selection Algorithms) and use it to search the possible combinations of features with ANN to score the probability of buying new services or products using only the selected features by ELSA. [12] The authors conclude that the ELSA/ANN model showed promising results when market managers have clear decision scenario or not.

In 2007 Gigli and all [13] presents a new data mining architecture to integrate a library of feature extraction, data mining and fusion techniques to automatically and optimally configure a classification solution for a given labeled set of training patterns. They describe how feature selection and data mining algorithms are combined through a Genetic Algorithm, using single source data, and how multi source data are combined through several best-suited fusion techniques by employing a genetic algorithm for optimal fusion.

All these studies emphasize the importance of applying a feature selection technique to a database in order to obtain a smaller database than the initial one. This reduced set of data will preserve all the features necessary for solving the given problem and will optimize the time and the resources involved in the process. We should mention that there have been cases when the data selected through a feature selection technique increased the accuracy necessary for solving the given problem.

In conclusion the idea of using input data features for training a neural network may seem simple and very natural. We may also consider that an important part of the input data currently used in training the different types of neural networks are recurrent and display unnecessary information for the training process. Nevertheless, the process of selecting some data features which can successfully represent the initial data in the training process is a challenge that, in present, is not 100% solved. The field that successfully deals with finding the best solutions for such problems is data mining.

II. SELECTING THE TRAINING SET FEATURES BY USING DATA MINING TECHNIQUES

By using the data mining procedure, we will try to find some correlations in a data set in order to select a feature set. For this purpose, a technique capable of selecting the best features from a database must be considered.

The technique used implies two components: a feature evaluation technique and a search algorithm for selecting the best features.

The evaluation technique used is named CFS (Correlation based Feature Selection) and it is an algorithm that combines this evaluation formula with an appropriate correlation measure and a heuristic search strategy. [14]

The technique is based on the hypothesis according to which a good feature set must have characteristics that are closely correlated to the respective set and less correlated (or uncorrelated) to other sets.

If the correlation between each of the components in a test and the outside variable is known, and the inter-correlation between each pair of components is given, then the correlation between a composite test consisting of the summed components and the outside variable can be predicted from:

$$r_{ZC} = \frac{k\,\overline{r_{zi}}}{\sqrt{k + k(k-1)\,\overline{r_{ii}}}} \qquad (1)$$

where r_{zc} is the correlation between the summed components and the outside variable, k is the number of components, $\overline{r_{zi}}$ is the average of the correlation between the components and the outside variable, and $\overline{r_{ii}}$ is the average inter-correlation between components.

CFS is a simple filter algorithm that ranks feature subsets according to a correlation based heuristic evaluation function. The main features from the dataset are highly correlated with a class and uncorrelated with each other.

Irrelevant features should be ignored because they will have low correlation with the class. Redundant features will be eliminated from the list despite being highly correlated with one or more classes because they belong to other features as well.

The acceptance of a feature will depend on its power of predicting certain classes from the datasets that cannot be predicted by other features corresponding to equation (1).

Experiments on artificial datasets showed that CFS quickly identifies and screens irrelevant, redundant, and noisy features, and identifies relevant features as long as their relevance does not strongly depend on other features.

As search algorithm, we used the BestFirst which is a greedy hill climbing algorithm [15] coupled with a backtracking strategy.

The hill climbing algorithm aims at maximizing (minimizing) an f(x) function, where x can take values in a discreet space. If we represent the x values as nods in a graph, then the arch between 2 nods represents the similitude between those 2 nods (states).

To resolve the problem we can use many hill-climbing methods. Forward selection (FS) is one of the simplest and highly used methods existent. It starts with an empty set and greedily adds features, one at a time, until all features have been added. At each step FS adds the feature that, when added to the current set, yields the learned structure that generalizes best. By adding one feature at a time, the features already added to the current dataset cannot be removed anymore.

Backward elimination (BE) is similar to FS, the only difference consisting in the fact that BE starts with all the features in the set and eliminates each feature at a time. Moreover, once a feature is removed from the set, it cannot subsequently be added.

The process of adding or removing a feature through a final action can be a hazardous attempt because the feature may subsequently prove its value in a future stage of the algorithm. Consequently, a safer method would be either add or remove an attribute in each stage. We can start with an empty attribute set, with a complete attribute set or with a random number of elements and subsequently, we can either add or remove any attribute. Forward stepwise selection (FSS) is a greedy hill climbing feature initialized with the empty feature set and backward stepwise elimination (BSE) is a greedy hill climbing feature initialized with a complete set of features.

The algorithm will run through the entire graph, nod by nod, each time trying to maximize (minimize) the f value, until a local maximum (minimum) is reached.

In Fig. 2, we can see the graphic representation for a function that has a single local maximum. We should mention that this representation serves just as example; more frequently, in practice we deal with functions having more maximum/ minimum local points.

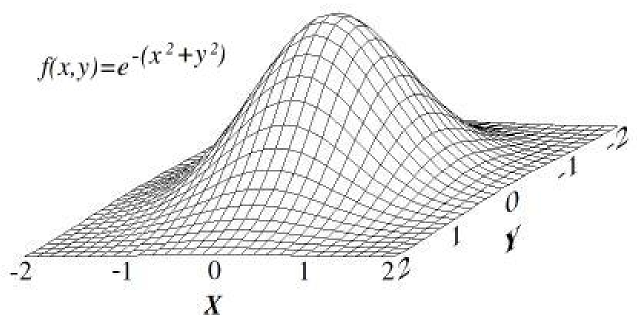

Figure 2. Graphic representation for a function that has a single local maximum.

For this reason, when the algorithm finds a maximum or a minimum point, it must determine whether it is local or general. The final task of the algorithm is to find a local maximum/ minimum point as close as possible to the general one.

By combining the two components, i.e. the feature evaluation technique and the search algorithm for selecting the best features, we will try to select the best features from a dataset using the data mining Weka program. [16-17]

The selected features will subsequently be used for training an ANN.

Further on, we will present the experiments performed and the data obtained through these experiments.

III. EXPERIMENTS

The experiments performed in this chapter were meant to determine two aspects of training an ANN with input data features. The first aspect refers to the accuracy of the features. We will try to determine whether by training an ANN with input data features, we obtain recognition rates comparable to the ones obtained by a network trained with original data. This is one of the most important aspects of training an ANN with input data features because our aim is to maintain the same recognition rates obtained by the original network.

The second aspect we want to emphasize is the training time. In what follows, we will demonstrate that by training an ANN with input data features, the training time will be considerably reduced in comparison with the time obtained by training a network with original data.

For the experiments we used an ANN multilayer perceptron network with one hidden layer.(Fig.3) The weights and biases of the neural networks are initialized randomly between 0.5 and -0.5, and the number of hidden nodes is determined heuristically for each problem.

The first experiment consisted in training a neural network to recognize handwriting. As training and testing data for these networks, we used the NIST 19 international database letter set, that contains over 800.000 entries. [18]

The characters used for this first example are represented on a 1024 pixels (32x32) image, each letter previously being processed through a corresponding scaling and a mass centre translation.

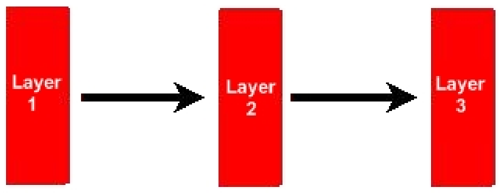

Figure 3. Multilayer perceptron with one hidden layer.

The first neural network has the following configuration:

- Input layer – it is made up of 1024 neurons. The activating function is the linear function f(x)=x. The set entries can take the following values: 0 or 1.
- Hidden layer – it is made up of 713 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))
- Output layer: it is made up of 4 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))

The neural network was trained to recognize 4 letters: a, e, n, r. All of them are present in the training set with 4000 patterns each.

For the second neural network, we firstly filtered the input data and then, we selected only those features that included the characteristics common for all of the 4 letters. The data filtering operation was performed by using the Weka program, to which a supervised 'selection of the best features' filter was used. [19]

After the filtering operation, only 100 of the 1024 features involved in experiment were preserved. The remaining attributes are: 80,84,111,115,120,121, 136,137,138,169,175, 237,238,301,396,408,426,429,434,435,436,440,441,459,460 ,465,466,467,468,471,472,473,488,489,491,492,498,499, 500,503,504,520,523,528,529,530,531,532,536,537,552,553 ,554,555,559,562,563,564,565,566,567,568,569,584,585, 593,595,596,597,598,599,600,616,617,623,626,627,628,631 ,632,633,648,649,650,656,657,658,659,660,661,662,663, 664,680,688,689,690,691,692,693,695,696,712,714,719,720 ,721,722,723,724,725,727,728,729,752,753,754,755,756, 758,760,761,777,784,785,786,787,788,789,790,791,792,816 ,817,818,819,820,821,822,823,824,825,845,847,848,850, 851,852,854,855,856,873,876,877,878,879,880,881,882,883 ,884,885,886,887,888,889,904,905,908,909,910,911,912,91 3,914,915,916,917,918,919,920,921,937,940,944,945,946,9 47,949,950,951,952 and 985.

Each of these features determines a major change in the output of the network. In Fig. 4, we can easily notice that the output of the network is highly influenced by the change of value for 2 inputs of the network.

There were mentioned changes in the structure of the neural network as well, as we can see: the second neural network has the following configuration:

- Input layer – it is made up of 193 neurons. The activating function is the linear function f(x)=x. The set entries can take the following values: 0 or 1.
- Hidden layer – it is made up of 225 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))
- Output layer: it is made up of 4 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))
-

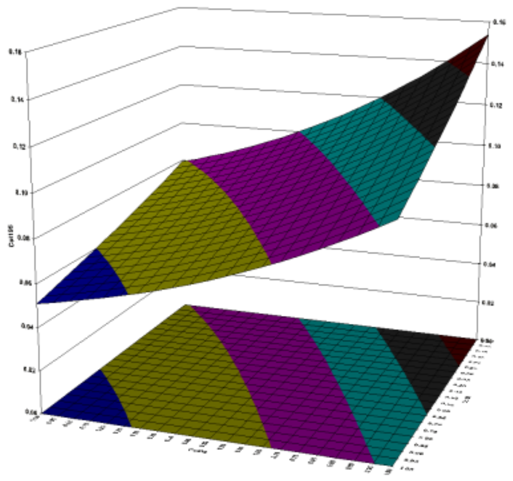

Figure 4. How neural network output 1(character 'a') is influenced by input 529 and 659.

After the training operation, we tested both networks using a set of 4000 letter patterns, each of the 4 letters having 1000 patterns. The results obtained are presented in Table I and Table II and the graphic representation can be seen in Fig. 5 and Fig. 6.

TABLE I. EXPERIMENT 1 RESULTS FOR LETTER RECOGNITION RATE

Letter	ANN 1 recognition rate	ANN 2 recognition rate
A	91.12 %	**91.12 %**
E	93.76 %	**93.84 %**
N	95.6 %	**95.92 %**
R	98.48 %	**98.4 %**

TABLE II. EXPERIMENT 1 RESULTS FOR NEURAL NETWORKS TRAINING TIME

No. of epochs	ANN 1 (sec)	ANN 2 (sec)
1	316	**18**
4	1268	**74**
6	1884	**110**
10	3157	**181**
22	6962	**403**
30	9495	**544**
38	11998	**694**
40	12619	**724**
50	15774	**905**
55	17342	**998**
60	18931	**1085**
65	20499	**1178**
70	22088	**1267**
76	23972	**1377**
80	25236	**1449**
85	26813	**1539**
90	28393	**1629**
95	29970	**1721**
100	31550	**1812**

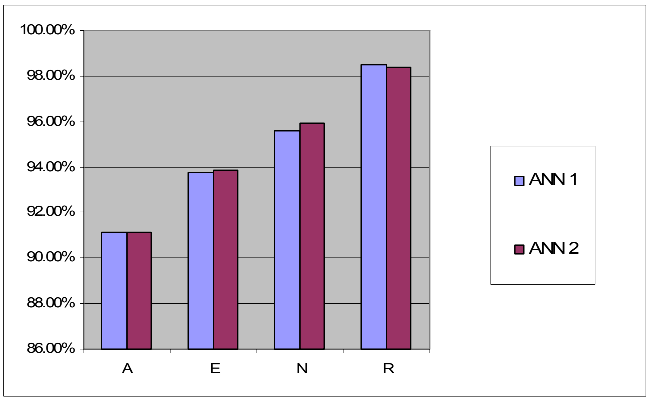

Figure 5. Recognition rate for character recognition.

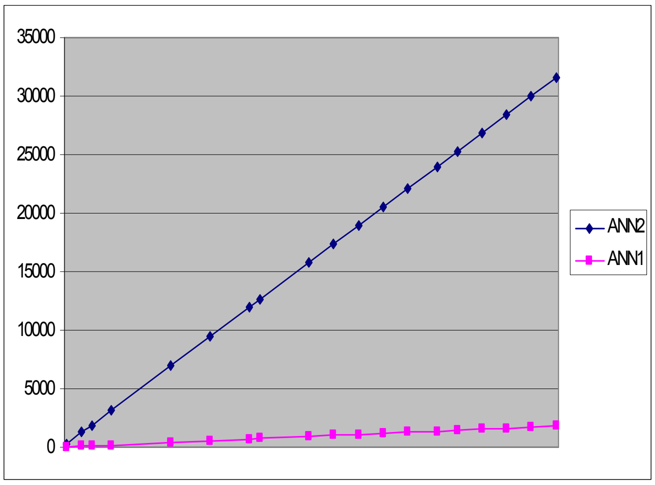

Figure 6. Training time for ANN1 and ANN2 in seconds.

We can easily notice a favorable recognition rate of ANN2 in comparison with ANN1 and the training time being very shorter for the second network.

For the second experiment, we tested the performance of neural networks trained with data features on several data sets that were used in many other benchmarks. [12] We will give a detailed account only of the segment problem, the rest being somehow similar to this.[20] The results obtained for all the problems are briefly displayed in Table V.

Each training pattern is made up of the features from a 3x3 pixels segment of an image. The neural network must analyze these features and classify the respective pattern into one of the 7 possible patterns available: pat, foliage, window, brick face, cement, sky, grass. The same as for the first experiment, we trained two neural networks: one with the initial data, the other with the filtered data.

After filtering the pattern data features, out of 19 features, only 6 were selected as being the most important for selecting the category to which the respective pattern belongs. The remaining attributes are: region-centroid-col, region-centroid-row, hedge-mean, rawred-mean, hue-mean.

The third neural network (ANN3) has the following configuration:

- Input layer – it is made up of 19 neurons. The activating function is the linear function f(x)=x. The set entries can take the following values: 0 or 1.
- Hidden layer – it is made up of 48 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))
- Output layer: it is made up of 7 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))

The fourth neural network (ANN4) has the following configuration:

- Input layer – it is made up of 6 neurons. The activating function is the linear function f(x)=x. The set entries can take the following values: 0 or 1.
- Hidden layer – it is made up of 41 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))
- Output layer: it is made up of 7 neurons. The activating function is a logistic function and has the following configuration: f(x)=1/(1+exp(-x))

After the training operation, we tested both networks using a set of 1500 patterns, each of the 7 letters having 215 patterns. The results obtained are presented in Table III and Table IV and the graphic representation can be seen in Fig. 7 and Fig. 8.

TABLE III. EXPERIMENT 2 RESULTS FOR PATTERN RECOGNITION RATE

Classes	ANN 3 recognition rate	ANN 4 recognition rate
Pat	96%	**96 %**
Foliage	97 %	**97.1 %**
Window	95.8 %	**95.8 %**
Brickface	97.2 %	**97.2 %**
Cement	96.4 %	**96.3 %**
Sky	96.8 %	**97 %**
Grass	96 %	**96 %**

TABLE IV. EXPERIMENT 2 RESULTS FOR NEURAL NETWORKS TRAINING TIME

Epochs	ANN 3 (sec)	ANN 4 (sec)
1	1	**1**
20	9	**2**
40	19	**5**
100	47	**11**
200	94	**24**
400	188	**48**
500	235	**60**
800	375	**97**
1000	470	**121**

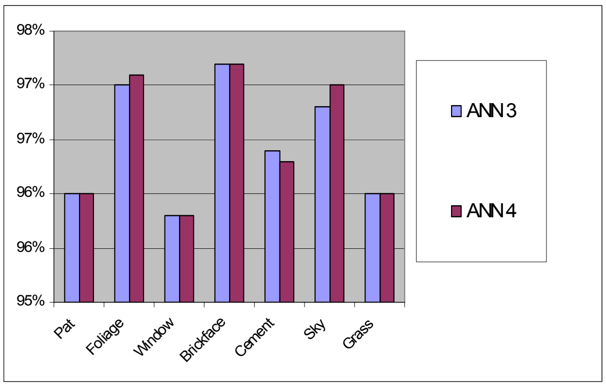

Figure 7. Recognition rate for “segment” problem.

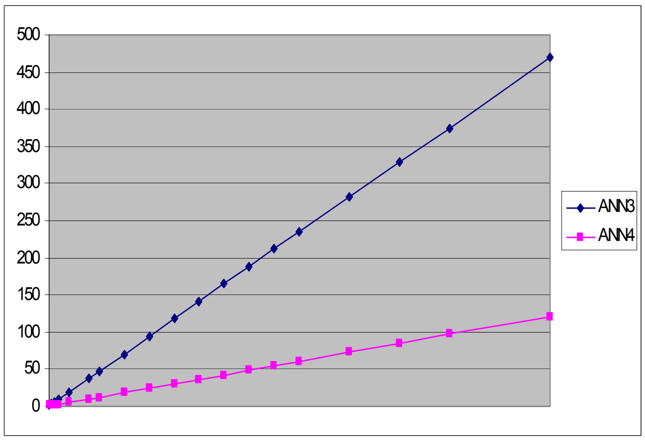

Figure 8. Training time for ANN3 and ANN4 in seconds.

The final results for all the data sets used in the second experiment are displayed in Table V.

TABLE V. EXPERIMENT 2 FOR ALL DATA SETS

Data set	Single net	Bagging	Ada Boost	GEFS	MEE	Our nets
Credita	84.3	86.2	84.3	86.8	86.4	**87.2**
Creditg	71.7	75.8	74.7	75.2	75.6	**78.2**
Diabetes	76.4	77.2	76.7	77.0	76.8	**78.99**
Glass	57.1	66.9	68.9	69.6	61.1	**73.58**
Cleveland	80.7	83.0	78.9	83.9	83.3	**85.2**
Hepatitis	81.5	82.2	80.3	83.3	84.9	**84.9**
Votes-84	95.9	95.9	94.7	95.6	96.1	**96.3**
Hypo	93.8	93.8	93.8	94.1	93.9	**94.8**
Ionosphere	89.3	90.8	91.7	94.6	93.5	**95.2**
Iris	95.9	96.0	96.1	96.7	96.5	**96.6**
Krvskp	98.8	99.2	99.7	99.3	99.3	**99.6**
Labor	91.6	95.8	96.8	96.5	94.4	**97.2**
Segment	92.3	94.6	96.7	96.4	93.2	**96.4**
Sick	95.2	94.3	95.5	96.5	99.3	**99.5**
Sonar	80.5	83.2	87.0	82.2	85.2	**87.7**
Soybean	92.0	93.1	93.7	94.1	93.8	**94.2**
Vehicle	74.7	79.3	80.3	81.0	76.4	**82.5**

We can clearly notice that the recognition rates obtained by training an artificial neural network with input data features are considerably superior to other training methods. Moreover, the training time for the networks that use input data features is evidently superior (sometimes up to 20 times shorter) to the networks trained with the entire data set.

IV. CONCLUSION

The present paper aimed at presenting a new neural network training method. The procedure consists in preprocessing the input data for selecting their main features. The filtering operation was performed by using a data mining algorithm for selecting the best features.

We obtained superior recognition rates for the neural networks to which the input data were previously filtered. Moreover, the training time was visibly shorter (up to 20 times) than a classic training and the complexity of the network trained with input features was evidently superior.

The results have been obtained by training several ANNs with data from different public databases and absolutely in all the cases, the implementations of ANNs trained with input data features are highly superior to the ones obtained by applying other algorithms.

We should also mention that an input data preprocessing by filtering the data main features is more than necessary because it eliminates the data recurrences and the features that are not used in the training procedure.

A preprocessing operation of the input data by selecting the data main features is highly recommendable as it considerably improves the neural network training procedure.

REFERENCES

[1] Negnevitsky, M., “Artificial Intelligence: A Guide to Intelligent Systems”(2nd Edition), Addison Wesley, England,2005.

[2] Luger G., “Artificial Intelligence :Structures and Strategies for Complex Problem Solving” (Fifth Edition) Addison Wesley, 2005.

[3] Stergiou, C., Siganos, D., “Neural networks” , http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html, 1996

[4] Babii, S., Cretu, V., Petriu, E.M.: Performance Evaluation of Two Distributed BackPropagation Implementations, Neural Networks 2007, IJCNN 2007, pp. 1578-1583

[5] Zhongwen, L., Hongzhi, L., Xincai, W.: Artificial neural network computation on graphic process unit, Neural Networks, 2005, IJCNN 2005, pp. 622-626.

[6] Siddique, M.N.H., Tokhi, M.O.: Training neural networks: backpropagation vs. genetic algorithms, Neural Networks, 2001, IJCNN, 2001, pp. 2673-2678

[7] Nguyen, D., Widrow, B.: Improving the learning speed of 2-layer neural networks by choosing initial values of adaptive weights, Neural Networks 1990, IJCNN, 1990, pp. 21-26, Volume. 3

[8] Gorea, D. : Dynamically Integrating Knowledge in Applications. An Online Scoring Engine Architecture, Advances in Electrical and Computer Engineering, Suceava, Romania,Volume 8,2008, pp.44-49

[9] Langley, P. : Selection of relevant features in machine learning, Proceedings of the AAAI Fall Symposium on Relevance, AAAI Press, 1994

[10] Jain, A., Zongker, D. : Feature selection: evaluation, application and small sample performance, Pattern Analysis and Machine Learning Intelligence, IEEE Transactions on, Volume 19, 1997, pp. 153-158

[11] Pudil, P., Novovicova, J., Kittler, J. : Floating search methods in feature selection, Pattern Recognition Letters, Volume 15,November 1994,pp. 1119-1125.

[12] Kim, Y., Street, W.N., Menczer, F. Roussell, G.J.: Feature selection in data mining, J. Wang Editor, Data Mining: Opportunities and Challenges, Idea Group Publishing, 2003, pages 80-105.

[13] Gigli, G., Bosse, I., Lampropoulos, G.A. : A optimized architecture for classification combining data fusion and data mining, Information Fusion, Volume 8, 2007, pp. 366-378

[14] Hall, M. “Correlation-based Feature Selection for Machine Learning”, Ph. D. diss. Hamilton, NZ: Waikato Uiversity, Department of Computer Science, 1998

[15] Boyan, J., Moore, A., “Learning evaluation functions to improve optimization by local search”, Journal of Machine Learning Research, Volume 1, pp. 77-112, 2000

[16] Weka3, “Data mining Software in Java”, The University of Waikato, http://www.cs.waikato.ac.nz/ml/weka, 2008

[17] Witten, I. H., Frank, E., “Data mining: Practical Machine Tools and Techniques (Second Edition), Morgan Kaufmann, 2005.

[18] NIST Handprinted Forms and Characters Database, www.nist.gov/srd/nistsd19.htm, 2007.

[19] http://weka.sourceforge.net/wiki/index.php, Performing attribute selection, 2008

[20] Image Segmentation Data, Vision Group, University of Massachusetts, November, 1990.