Optimizing Feed-Forward Neural Network Topology by Multi-objective Evolutionary Algorithms: A Comparative Study on Biomedical Datasets

Vitoantonio Bevilacqua[1](✉), Fabio Cassano[1], Ernesto Mininno[2], and Giovanni Iacca[2]

[1] Dipartimento di Ingegneria Elettrica e dell'Informazione, Politecnico di Bari, via Orabona 4, 70125 Bari, Italy
vitoantonio.bevilacqua@poliba.it

[2] Cyber Dyne S.r.l., Via Scipione Crisanzio 119, 70123 Bari, Italy

Abstract. The design of robust classifiers, for instance Artificial Neural Networks (ANNs), is a critical aspect in all complex pattern recognition or classification tasks. Poor design choices may undermine the ability of the system to correctly classify the data samples. In this context, evolutionary techniques have proven particularly successful in exploring the complex state-space underlying the design of ANNs. Here, we report an extensive comparative study on the application of several modern Multi-Objective Evolutionary Algorithms to the design and training of an ANN for the classification of samples from two different biomedical datasets. Numerical results show that different algorithms have different strengths and weaknesses, leading to ANNs characterized by different levels of classification accuracy and network complexity.

Keywords: Artificial Neural Networks · Multi-Objective Evolutionary Algorithms · Akaike Information Criterion

1 Introduction

Over the past two decades, the amount of data produced yearly in all human applications has reached an unprecedented level. As the quantity of data generated in the most complex engineering, networking, and financial systems, is for obvious reasons, impossible to analyze manually, the need arises for expert systems capable of analyzing the data automatically, for instance for the purpose of classification, pattern recognition, and feature extraction. This need is bringing Machine Learning (ML) to a new level, and novel approaches are being presented in the literature, specifically tailored for problems of increasing complexity.

Interestingly, many modern ML classification techniques are now based on Artificial Neural Networks (ANNs). In fact, despite being one of the oldest computational tools known in ML, ANNs are still today among the most effective

DOI: 10.1007/978-3-319-32695-5_5

techniques available to solve most kinds of classification problems. On the other hand, while ANNs are powerful learners per se, their performance on specific datasets can be severely affected by several factors. First of all, since neural networks need to learn from examples, typically the training and validation sets must be quite large and should contain balanced class examples. The second problem is feature selection, i.e. the choice of which features should be used as input to the classifier. More features do not lead necessarily to a better classification accuracy, however feature selection can be especially hard in some cases.

One of the most prominent areas of application of ANNs is nowadays health care and health improvement, see e.g. [1,2]. For example, computerized medical imaging systems are constantly improving their ability to extract numerical features from biological data, features that can be used in expert systems (based on ANNs) to assist diagnosis and therapy. Typically, in order to train a robust expert system and obtain a high classification accuracy, one needs a large set of labeled samples. However, in most cases, data acquisition and labeling is expensive (due to the cost of the medical tests, and to the need for a human expert to label the training samples) and the expert system must be trained on a relatively small dataset. Therefore, a third major challenge is to reach a high accuracy with a limited number of labeled samples.

Finally, on top of all the above mentioned problems, there is the choice of the ANN topology, namely the number of layers, the number of nodes per layer, and the activation function used in each node. While simple rules of thumb exist for such a choice, there is no way to predict which configuration of the network is the best to use in each case and one often has to rely on manual trial-and-error. However, the training of each different network configuration is a time consuming process and trial-and-error is obviously prone to sub-optimal results.

Thanks to the ever-increasing availability of computing power, a viable alternative for solving these problems is now the use of automatic techniques that explore the entire space of solutions defined by the ANN topologies, while performing the training of each network on multiple shuffled versions of the dataset at hand. One such example is presented in [3], where Multi-Objective Genetic Algorithm (MOGA) [4] is used to find the optimal ANN topology (i.e., the optimal number of hidden layers and the number of nodes for each hidden layer) which leads to the best classification of the samples from the Wisconsin Breast Cancer Dataset (WBCD) [5].

In this paper, we follow up on [3] by performing an extensive comparison of a whole set of Multi-Objective Evolutionary Algorithms (MOEAs) on two different datasets, namely the aforementioned WBCD and the Hepatitis Dataset (HD)[5]. First, we try to obtain on each dataset the best possible accuracy, by minimizing at the same time the validation and test error. In a second set of experiments, we try to identify the best trade-off between accuracy and network complexity: in this latter case, the optimization criteria are the minimization of (1) the validation error, and (2) a measure of the network complexity, i.e. the Akaike Information Criterion [6] (rather than an explicit minimization of the number of hidden layers and hidden nodes per layer).

The rest of this paper is organized as follows. The next section briefly summarizes the related work on the use of MOEAs for automatic design of classifiers. Section 3 describes the MOEA-based method used in the study, while numerical results are reported in Sect. 4. Finally, Sect. 5 concludes this work.

2 Related Work

MOEAs are bio-inspired multi-objective optimization techniques that have been successfully used in several applications domains, such as engineering design [7,8] and combinatorial optimization [9]. Recently, MOEAs have also been used in real-time applications, as shown in [10], and biomedical applications [11,12].

In the ML domain, there are several examples of application of (either single-objective or multi-objective) Evolutionary Algorithms to the optimization of neural network topology (see e.g. [13]), or for training ANNs [14]. Another example is given by [15], where an improved version of classic Genetic Algorithms (GA) is introduced, specifically designed to optimize the structure of a neural network. In the aforementioned work [4], Multi-Objective Genetic Algorithm (MOGA) has been used to find the best topology in order to improve the neural network accuracy on the WBCD dataset. A similar technique was also proposed in a more recent study [16]. Other biomedical applications of optimized neural networks are also presented in [17–19].

3 Proposed Approach

As mentioned earlier, the main idea of this study is to formulate the problem of the definition of an optimal ANN for a specific dataset in a multi-objective fashion. For example, in order to maximize the accuracy, a Multi-Objective Evolutionary Algorithm can be used to minimize the validation error while minimizing the test error. However, depending on the requirements one could also include different optimization criteria in the problem formulation, such as a measure of complexity of the classifier ANN. We will show this in the next section.

In a nutshell, the proposed multi-objective approach consists of two nested loops, as depicted in Fig. 1: (1) an outer loop, where populations of candidate ANNs are generated and optimized by a MOEA; (2) an inner loop, where each candidate ANN is trained, validated and tested.

More specifically, in the first step, the MOEA defines for each candidate ANN the number of hidden layers, the number of nodes per layer, and (optionally) the activation function that must be used in each layer. This information is then used to create neural networks that are structured as follows: (1) an input layer made of as many nodes as the number of the features of the dataset, with no bias; (2) a variable number of hidden layers, each of which is made of a variable number of nodes (as determined by the MOEA), with bias; (3) an output layer, with a single node (the classification value), with no bias. For every layer, it is possible to select the activation function a priori, or have the MOEA select it: as shown in the next section, in our experiments we tested both options.

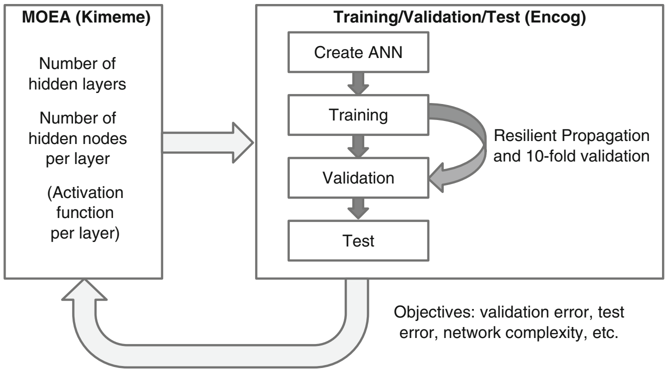
MOEA (Kimeme)
Number of hidden layers
Number of hidden nodes per layer
(Activation function per layer)
Training/Validation/Test (Encog)
Create ANN
Training
Validation
Test
Resilient Propagation and 10-fold validation
Objectives: validation error, test error, network complexity, etc.

Fig. 1. Conceptual scheme of the MOEA-based approach.

In the second step, each ANN so generated, is then trained, validated and tested on the dataset. This procedure is performed as follows. First, the original dataset is shuffled and partitioned into three sets (in our experiments, 60 %, 20 %, and 20 % of the entire dataset, respectively). Then, the first set is used to train and validate the neural network topology, by means of a 10-fold cross validation. The training method we use in our experiments is the Resilient Propagation [20], with stop condition on a training error threshold. The second and third sets are finally used to calculate, respectively, the validation error, the test error, and the confusion matrix. This information (or, if needed, a metric of network complexity) is then fed back to the outer loop, and used to calculate the fitness functions to be optimized by the MOEA.

In all our experiments (see the next section for further details), we used some of the state-of-the-art MOEAs available in Kimeme, a multi-disciplinary optimization platform introduced in [21,22]. The reason for using Kimeme was manifold: first of all, Kimeme provides a rich set of state-of-the-art single and multi-objective optimization algorithms, as well as an extensive post-processing toolkit. Secondly, Kimeme can be easily coupled with external software and pieces of code (such as Java or Python classes, or Matlab scripts). Importantly, Kimeme also integrates a distributed computing framework, which allowed us to easily run massively parallel calculations. As for the ANN implementation, we used the open-source Java library Encog [23], which is characterized by a great flexibility in the definition of neural networks and training algorithms.

4 Numerical Results

In the following, first we describe our experimental setup (datasets and multi-objective algorithms), then we analyze the numerical results obtained in the different experiments with the approach described in the previous section. We finally report a brief analysis of the execution times of our experiments.

4.1 Datasets

In our experimental study, we consider two biomedical datasets:

- The Wisconsin Breast Cancer Dataset (WBCD) [5]. The WBCD is composed of 699 labeled samples, each defined by 9 biomedical features, namely: clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses. The dataset contains 16 samples with one or more missing values, which we omit from our analysis.
- The Hepatitis Dataset (HD) [5]. The dataset is composed of 155 labeled samples, each defined by 19, namely: age, sex, steroid, antivirals, fatigue, malaise, anorexia, liver big, liver firm, spleen palpable, spiders, ascites, varices, bilirubin, alk phosphate, sgot, albumin, protime, histology. 75 samples have one or more missing value, so we discard them in our analysis. Since this dataset is unbalanced, to avoid over-fitting we have added 30 synthetic entries to the least represented class ("die"), each one obtained selecting randomly one of the original samples, and adding (or subtracting), with probability $p = 0.3$, a small uniform random number to each of its features.

On both datasets, we normalize the input features in the range $[0, 1]$. Also, since both datasets refer to a binary classification problem (positive vs negative diagnosis), for the purpose of classification we set a threshold of 0.5 on the output of the single output node (see the previous section), to discriminate between the two sample classes (corresponding to 0/1 classification values).

4.2 Algorithms

Among the open-source algorithms available in Kimeme, we chose for this comparative study four MOEAs together with a version of Multi-Objective Particle Swarm Optimization used here as control experiment[1]. These algorithms were chosen as they are currently considered the state-of-the-art in multi-objective optimization, and our aim here is to show how general-purpose MOEAs can be used for the automatic design of classifiers. A brief description of the selected algorithms follows, with the related parametrization (for a more thorough explanation of the algorithms and their parameters, please refer to the original papers). All algorithms were configured to use a population of 100 individuals, with stop condition on the number of generations (500).

- Multi-Objective Differential Evolution (MODE). This is a custom multi-objective variant (with elitism) of Differential Evolution (DE) [24], that simply combines with the classic DE mutation/crossover operators the non-dominated sorting and crowding distance mechanisms used in NSGA2 (see below). We set crossover rate $Cr = 0.3$ and scale factor $F = 0.5$.

[1] We should note that, technically speaking, MOPSO is not a MOEA, as it is inspired by Swarm Intelligence rather than Evolutionary Algorithms. Nevertheless, for simplicity of notation in the following we will use the wording "MOEAs" to refer generically to all the algorithms tested in this study, including MOPSO.

- Multi-Objective Evolution Strategies (MOES) [25]. This is a multi-objective variant of classic Evolution Strategies (ES), an evolutionary algorithm based on mutation only. Mutation simply adds to each component of the solution a random number drawn from an adaptive distribution. Solutions are then ranked, based on their fitness values, to obtain the Pareto front. We set minimum step size $\mu_{min} = 0.01$, initial step size $\mu_{init} = 0.2$, life span $LS = 30$, scaling factor $\alpha = 0.2$, and learning rate $\tau = 1$.
- Non-Dominated Sorting Genetic Algorithm-2 (NSGA2) [26]. NSGA2 is arguably the most popular algorithm in multi-objective-optimization. It is a variant of Genetic Algorithm that uses a non-dominated sorting mechanism (to rank solutions based on their dominance level) and a crowding distance operator (which preserves a high diversity in the population). We set tournament size $T = 2$, crossover probability $Cr = 0.75$, mutation probability $m = 0.05$, selection percentage $s = 0.35$, and exploration factor $e = 0.8$.
- Strength Pareto Evolutionary Algorithm-2 (SPEA2) [27]. SPEA2 relies on an archive of boundary solutions and a mechanism for pruning such archive along the evolutionary process. Additionally, it incorporates a fine-grained fitness assignment strategy based on a nearest-neighbor density estimation technique which guides the search more efficiently. We set tournament size $T = 2$, crossover probability $Cr = 0.9$, and mutation probability $m = 0.01$.
- Multi-Objective Particle Swarm Optimization (MOPSO) [28]. This is a multi-objective variant of Particle Swarm Optimization, that simply combines with the classic PSO logics the non-dominated sorting used in NSGA2. The parametrization is the one proposed in [28].

4.3 Minimization of Test Error vs Minimization of Validation Error

The first set of experiments has as main objective the minimization of validation and test error. Minimizing the validation error allows one to avoid the overfitting problem, whereas minimizing the test error gives the best performance in terms of accuracy. We repeat the experiments on each dataset in two conditions, i.e. (1) one in which the activation function is fixed, a priori and for the entire network, to one of the following: {Gaussian, Linear, Sigmoid, Sin, Step} and (2) one in which the activation function is free to vary for each layer, and is chosen by the MOEA. In the latter case, the activation function is chosen within the following set of functions: {Bipolar, Competitive, Gaussian, Linear, Log, Ramp, Sigmoid, Sin, SoftMax, Step, Tanh, Elliott, Symmetric Elliott}; moreover, the MOEA is allowed to select the same activation function for more than one layer.

In both cases, the neural network topologies are structured as described in Sect. 3. The number of hidden layers varies in $[1, 3]$, with the first layer having a variable number of nodes in the range $[1, 255]$, while the size of the second and the third layers vary in $[0, 255]$, with zero meaning that the layer is not present. This way, we enforce the condition that the network has at least a hidden layer made of a single node, while the other two hidden layers might not be present.

We execute the five MOEAs 5 independent times, with different random seeds, on both datasets in the two conditions. For each algorithm we then aggregate the Pareto-optimal solutions found at the end of each run, and finally we select the non-dominated solutions among all the optimal solutions found. We report the set of non-dominated solutions obtained by each algorithm on the WBCD in Figs. 2 and 3a, respectively for the case with fixed and variable activation function. As for the HD, due to space limitations we report only the non-dominated solutions obtained with variable activation function, see Fig. 4.

In all figures, the solutions marked with a black square indicate neural networks reaching a full accuracy of 100 %. We should note that, while validation and test error are calculated as Mean Squared Error (MSE) between the expected classification value (0/1) and actual neural network output (ranging in $[0, 1]$, and depending on the output activation function), the accuracy is calculated based on the confusion matrix: (True Positives + True Negatives)/(Total sample size).

The numerical results show that on both datasets and conditions (fixed or variable activation function), the MOEAs obtain several solutions with full accuracy, with no clear superiority of any of the algorithms. Also, the choice of the activation function seems to affect only marginally the performance. Notably, using a variable activation function allowed us to find, on the WBCD, the two full accuracy classifiers with the lowest validation/test error (see the red circle, in Fig. 3a, grey in print). The MOEAs were also successful on the HD, finding numerous full accuracy ANNs, although with a higher validation/test error compared to WBCD (most probably because of the unbalance of the dataset).

To further highlight the potentialities of the MOEA-based method, we report in Tables 1 and 2 a comparison of the best accuracy found in this study against the accuracy obtained in the state-of-art literature, respectively on the WBCD and the HD. We can see that the MOEA-based method tested here is the only one capable of reaching an accuracy value of 100 % on both datasets.

4.4 Minimization of Network Complexity vs Minimization of Validation Error

As an additional experiment, we apply the MOEA-based method to a different formulation of the neural network optimal design, one in which the optimization criteria are the minimization of validation error and network computational complexity. The latter is measured here via the Akaike Information Criterion (AIC) [6], defined as $-2 \cdot \ln(\text{MSE}) + 2k$, where k is the number of weights in the ANNs. This second goal might be important, for instance, in contexts where the classifier ANN must be used in real-time and therefore should be computationally cheap, still guaranteeing robust classification performance.

Due to space limitations, we report only the results on the WBCD (Fig. 3b), but similar considerations apply also to the other dataset. Also, in this case we consider only variable activation functions. Results show that in this case the MOEAs find only one solution with full accuracy, that is associated to the lowest AIC level. On the other hand, ANNs of higher complexity are, unsurprisingly, characterized by a lower validation error but, because of overfitting, none of them is capable of generalizing and obtain full accuracy.

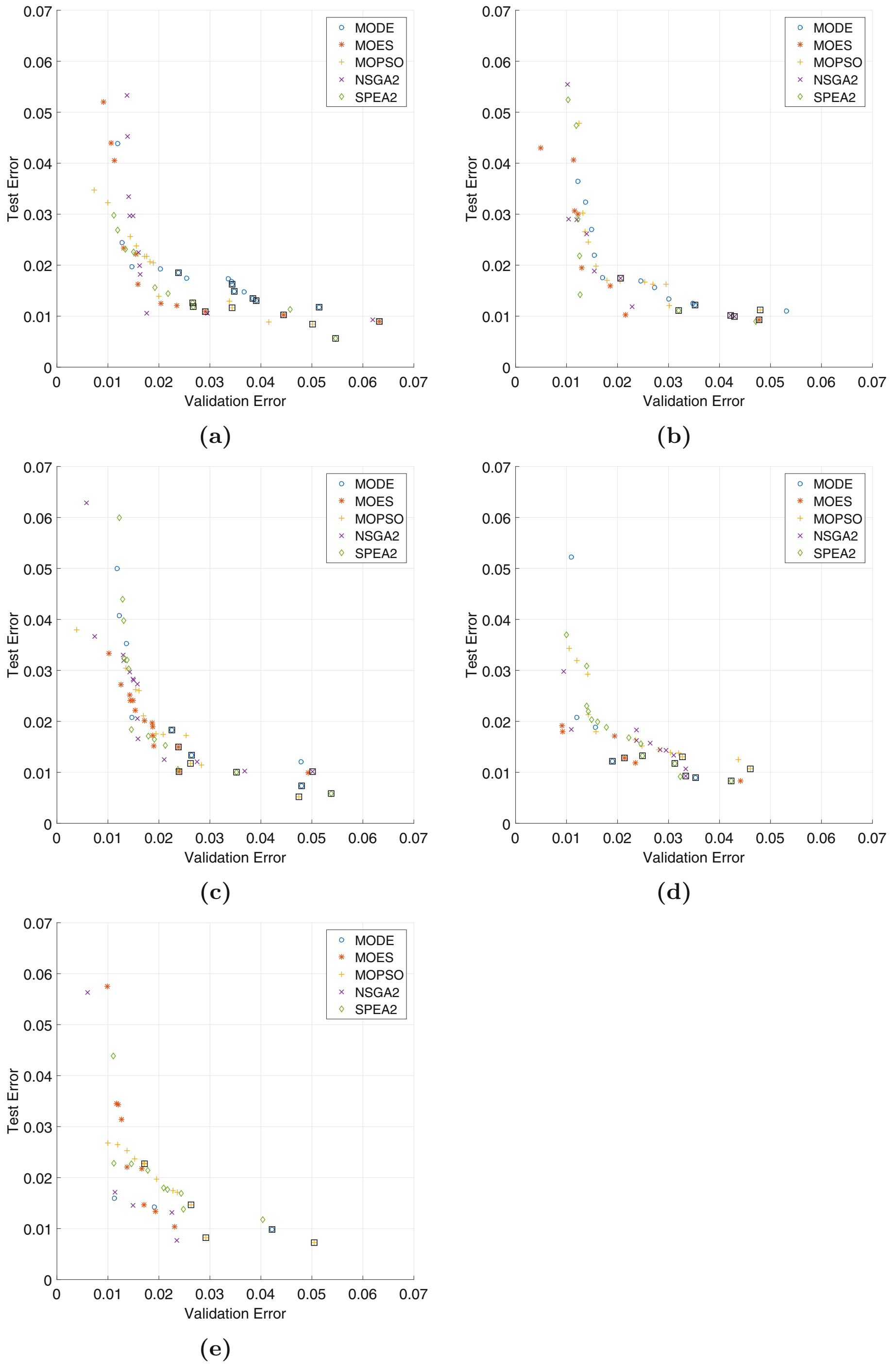

Fig. 2. Non-dominated solutions on the WBCD (with fixed activation function): minimization of validation and test error with Gaussian (a); Linear (b); Sigmoid (c); Sin (d); and Step (e) function.

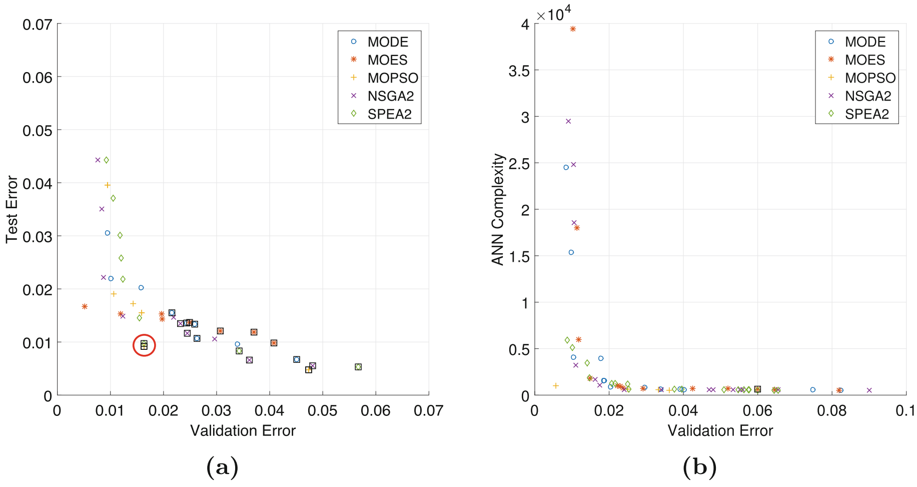

Fig. 3. Non-dominated solutions on the WBCD (with variable activation function): minimization of validation and test error (a); minimization of validation error and Akaike Information Criterion (b).

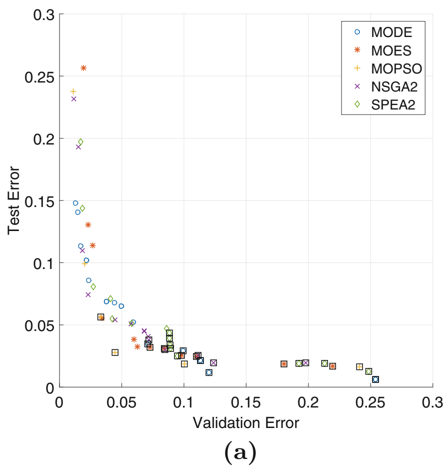

Fig. 4. Non-dominated solutions on the HD (with variable activation function): minimization of validation and test error.

4.5 Execution Times of the Experiments

Finally, we conclude our presentation of the numerical results with a brief analysis of the execution times. On the WBCD, each run of the various MOEAs is executed in approximately 2–6 min. On the HD instead, each run is executed in approximately 2–3 h (except for MOPSO, which takes up to 5 h/run). All tests were executed using 8 threads in parallel on a Linux (Ubuntu 15.04) machine

Table 1. Accuracy on the WBCD

Accuracy	Reference
99.51 %	[29]
99.14 %	[30]
97.8 %	[31]
97.21 %	[32]
100 %	this work

Table 2. Accuracy on the HD

Accuracy	Reference
96.25 %	[33]
94,12 %	[34]
92.9 %	[35]
100 %	this work

with an eight-core i7-5960X CPU and 16 GB RAM. The large difference in runtime between the two datasets can be explained considering that the HD is an unbalanced dataset and the training time on each shuffled version of it takes more time to reach the training error threshold[2].

5 Conclusion

In this paper we have introduced a multi-objective optimization approach for optimally designing and training Artificial Neural Networks used for classification problems. The proposed method leverages several state-of-the-art algorithms provided by Kimeme, an optimization platform available online. We conducted a thorough experimental campaign testing a number of modern multi-objective optimization algorithms, including MOES, MODE, MOPSO, NSGA2 and SPEA2 on two different datasets. Such comparative study was performed on the Breast Cancer Wisconsin Dataset and the Hepatitis dataset from the UCI repository. The aim was to find the non-dominated ANNs minimizing the validation error and the test error, or, alternatively, minimizing at the same time the validation error and the ANN complexity. The latter was measured by means of the Akaike Information Criterion.

All the tested algorithms were able to find, in all conditions on both datasets, several ANNs characterized by 100 % accuracy, improving upon results previously found in the literature. Among the algorithms selected in the study though, we observed a substantial equivalence.

The proposed approach reveals that the automatic design of Artificial Neural Networks by means of multi-objective optimization is a viable solution in the context of complex classification problems. This is especially true when any prior information about the problem at hand is scarce, or not available at all. Furthermore, such an automatic design has a high degree of general purposeness, as it can be easily extended to different classification tasks.

In future studies, we will attempt to test this method on new problems, and we will try to devise novel optimization schemes specifically designed for ML.

[2] We should note though, that while in the case of WBCD we used a training error threshold 0.1, in the case of HD we used a threshold of 0.2, to improve the training time and avoid overfitting.

References

1. Baxt, W.G.: Application of artificial neural networks to clinical medicine. Lancet **346**(8983), 1135–1138 (1995)
2. Floyd, C.E., Lo, J.Y., Yun, A.J., Sullivan, D.C., Kornguth, P.J.: Prediction of breast cancer malignancy using an artificial neural network. Cancer **74**(11), 2944–2948 (1994)
3. Bevilacqua, V., Mastronardi, G., Menolascina, F., Pannarale, P., Pedone, A.: A novel multi-objective genetic algorithm approach to artificial neural network topology optimisation: the breast cancer classification problem. In: International Joint Conference on Neural Networks, pp. 1958–1965. IEEE (2006)
4. Fonseca, C., Fleming, P.: Multiobjective genetic algorithms made easy: selection sharing and mating restriction. In: First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA 1995, pp. 45–52 (1995)
5. Lichman, M.: UCI Machine Learning Repository (2013). http://archive.ics.uci.edu/ml
6. Akaike, H.: A new look at the statistical model identification. IEEE Trans. Autom. Control **19**(6), 716–723 (1974)
7. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms, vol. 16. Wiley, New York (2001)
8. Bevilacqua, V., Costantino, N., Dotoli, M., Falagario, M., Sciancalepore, F.: Strategic design and multi-objective optimisation of distribution networks based on genetic algorithms. Int. J. Comput. Integr. Manufact. **25**(12), 1139–1150 (2012)
9. Ishibuchi, H., Akedo, N., Nojima, Y.: Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. IEEE Trans. Evol. Comput. **19**(2), 264–283 (2015)
10. Coello, C.A.C.: Multi-objective evolutionary algorithms in real-world applications: some recent results and current challenges. In: Greiner, D., Galván, B., Périaux, J., Gauger, N., Giannakoglou, K., Winter, G. (eds.) Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences, pp. 3–18. Springer, New York (2015)
11. Bevilacqua, V., Mastronardi, G., Piscopo, G.: Evolutionary approach to inverse planning in coplanar radiotherapy. Image Vision Comput. **25**(2), 196–203 (2007)
12. Menolascina, F., Bellomo, D., Maiwald, T., Bevilacqua, V., Ciminelli, C., Paradiso, A., Tommasi, S.: Developing optimal input design strategies in cancer systems biology with applications to microfluidic device engineering. BMC Bioinform. **10**(S–12), 1–4 (2009)
13. Whitley, D., Starkweather, T., Bogart, C.: Genetic algorithms and neural networks: optimizing connections and connectivity. Parallel Comput. **14**(3), 347–361 (1990)
14. Ilonen, J., Kamarainen, J.K., Lampinen, J.: Differential evolution training algorithm for feed-forward neural networks. Neural Process. Lett. **17**(1), 93–105 (2003)
15. Leung, F.H., Lam, H.K., Ling, S.H., Tam, P.K.: Tuning of the structure and parameters of a neural network using an improved genetic algorithm. IEEE Trans. Neural Netw. **14**(1), 79–88 (2003)
16. Bhardwaj, A., Tiwari, A.: Breast cancer diagnosis using genetically optimized neural network model. Expert Syst. Appl. **42**(10), 4611–4620 (2015)

17. Bevilacqua, V., Brunetti, A., de Biase, D., Tattoli, G., Santoro, R., Trotta, G.F., Cassano, F., Pantaleo, M., Mastronardi, G., Ivona, F., et al.: A P300 clustering of mild cognitive impairment patients stimulated in an immersive virtual reality scenario. In: Intelligent Computing Theories and Methodologies, pp. 226–236. Springer (2015)
18. Bevilacqua, V., Salatino, A.A., Di Leo, C., Tattoli, G., Buongiorno, D., Signorile, D., Babiloni, C., Del Percio, C., Triggiani, A.I., Gesualdo, L.: Advanced classification of alzheimer's disease and healthy subjects based on EEG markers. In: International Joint Conference On Neural Networks, pp. 1–5. IEEE (2015)
19. Bevilacqua, V., Tattoli, G., Buongiorno, D., Loconsole, C., Leonardis, D., Barsotti, M., Frisoli, A., Bergamasco, M.: A novel BCI-SSVEP based approach for control of walking in virtual environment using a convolutional neural network. In: International Joint Conference On Neural Networks, pp. 4121–4128. IEEE (2014)
20. Riedmiller, M., Braun, H.: RPROP-a Fast Adaptive Learning Algorithm. In: Proceedings of ISCIS VII, Universitat (1992)
21. Cyber Dyne Srl: Kimeme. http://cyberdynesoft.it/
22. Iacca, G., Mininno, E.: Introducing kimeme, a novel platform for multi-disciplinary multi-objective optimization. In: Rossi, F., et al. (eds.) WIVACE 2015. CCIS, vol. 587, pp. 40–52. Springer, Heildelberg (2016). doi:10.1007/978-3-319-32695-5_4
23. Heaton, J.: Programming Neural Networks with Encog 2 in Java (2010)
24. Storn, R., Price, K.: Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**(4), 341–359 (1997)
25. Beyer, H.G., Arnold, D.V.: Theory of evolution strategies - a tutorial. In: Kallel, L., Naudts, B., Rogers, A. (eds.) Theoretical Aspects of Evolutionary Computing, pp. 109–133. Springer, New York (2001)
26. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
27. Zitzler, E., Laumanns, M., Thiele, L., Zitzler, E., Zitzler, E., Thiele, L., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm (2001)
28. Wickramasinghe, U., Li, X.: Choosing leaders for multi-objective PSO algorithms using differential evolution. In: Li, X., et al. (eds.) SEAL 2008. LNCS, vol. 5361, pp. 249–258. Springer, New York (2008)
29. Akay, M.F.: Support vector machines combined with feature selection for breast cancer diagnosis. Expert Syst. Appl. **36**(2), 3240–3247 (2009)
30. Şahan, S., Polat, K., Kodaz, H., Güneş, S.: A new hybrid method based on fuzzy-artificial immune system and k-nn algorithm for breast cancer diagnosis. Comput. Biol. Med. **37**(3), 415–423 (2007)
31. Pena-Reyes, C.A., Sipper, M.: A fuzzy-genetic approach to breast cancer diagnosis. Artif. Intell. Med. **17**(2), 131–155 (1999)
32. Setiono, R., Liu, H.: Symbolic representation of neural networks. Computer **29**(3), 71–77 (1996)
33. Sartakhti, J.S., Zangooei, M.H., Mozafari, K.: Hepatitis disease diagnosis using a novel hybrid method based on support vector machine and simulated annealing (SVM-SA). Comput. Methods Programs Biomed. **108**(2), 570–579 (2012)
34. Polat, K., Güneş, S.: Prediction of hepatitis disease based on principal component analysis and artificial immune recognition system. Appl. Math. Comput. **189**(2), 1282–1291 (2007)
35. Bascil, M.S., Oztekin, H.: A study on hepatitis disease diagnosis using probabilistic neural network. J. Med. Syst. **36**(3), 1603–1606 (2012)