A SELF-CONFIGURING GENETIC ALGORITHM FOR THE AUTOMATED DESIGN OF SEMI-SUPERVISED ARTIFICIAL NEURAL NETWORKS[1]

Maria Semenkina

Siberian State University of Science and Technology
e-mail: semenkina88@mail.ru
Russia

Abstract: Semi-supervised techniques based on using both labelled and unlabelled examples can be an efficient tool for solving real-world problems with large amounts of unlabelled data. The usual method of solving such problems demands the long work of human experts in its initial stages to prepare the learning data, a process which includes such complex tasks as the labelling of large numbers of examples. In this case, it is not necessary to label all of these many examples, but just a few of them. In this paper, a new self-configuring genetic algorithm (SelfCGA) for the automated design of semi-supervised artificial neural networks (ANN) is presented. Firstly, the performance and behaviour of the proposed semi-supervised ANNs were studied under common experimental settings and their workability was established. Then their efficiency was evaluated on a real-world problem.

Key words: Semi-Supervised Learning, Genetic algorithms, Artificial Neural Networks, Self-configuration, Automated generation.

1. INTRODUCTION

Nowadays in different fields of human activity, we can encounter the following situation: there is a large amount of data without labels. The usual method of "machine" information extraction demands the long work of human experts in its initial stages to prepare the learning data, a process which includes such complex tasks as the labelling of large numbers of examples. Semi-supervised techniques can use both labelled and unlabelled data to construct appropriate models [10]. In this case, it is not necessary to label all of this large number of examples, but just a few of them.

In this study, we use several semi-supervised techniques, such as semi-supervised artificial neural networks trained by evolutionary algorithms

[1] The paper has been presented at the Sixth International Workshop on Mathematical Models and their Applications, 13-15.11.2017, Krasnoyarsk, Russian Federation and is not published in the Conference Proceedings.

2. SEMI-SUPERVISED ANN AUTOMATED DESIGN

The appropriate structure of the ANN must be chosen for the effective solving of the problem. Below, we consider a genetic algorithm (GA) for the choice of the number of layers, the number of neurons in each layer and the type of the activation function of each neuron for the multi-layered perceptron in the case of semi-supervised learning.

2.1. ANN in binary string

First of all, we choose the perceptron with 5 hidden layers and 5 neurons in each hidden layer as the maximum size of the structure for the ANN. Each node is represented by a binary string of length 4. If the string consists of zeros ("0000") then this node does not exist in the ANN. So, the whole structure of the neural network is represented by a binary string of length 100 (25x4); each 20 variables represent one hidden layer. The number of input neurons depends on the problem in hand. The ANN has one output layer.

We use 15 activation functions such as a bipolar sigmoid, a unipolar sigmoid, Gaussian, a threshold function and a linear function. For determining which activation function will be used on a given node, the integer that corresponds to its binary string is calculated.

Thus, we use optimization methods for problems with binary variables for finding the best structure and the optimization method for problems with real-valued variables for the weight coefficient adjustment of each structure.

Although the automated design of the ANN structure by self-adapting optimization techniques improves their efficiency, it can work unsatisfactorily with large real-world problems. Therefore, the automation of the most important input selection can have a significant impact on the efficiency of neural networks. In this paper, we use additional bits in every string for the choice of relevant variables to put them in model. The number of these bits equals the number of input variables. If this bit is equal to '0', then the corresponding input variable is not used in the model and is removed from the sample. During initialization, the probability for a variable to be significant will be equal to 1/3. This idea can help end users to avoid the significant and complicated procedure of choosing the appropriate set of input variables with the necessary impact on the model performance.

For the choice of more flexible models, more sophisticated tools must be used.

2.2. Self-configuring genetic algorithm

If the decision is made to use evolutionary algorithms for solving real world optimization problems, it will be necessary to choose an effective variant of algorithm parameters such as the kind of selection, recombination and mutation operators. Choosing the right EA setting for each problem is a difficult task even for experts in the field of evolutionary computation. It is the main problem in effectively implementing evolutionary algorithms for end users. We can conclude that it is necessary to find the solution for the main problem of evolutionary algorithms before suggesting for end users any EA application for the automated design of tools for solving real world problems.

We propose using the self-configuring evolutionary algorithms (SelfCEA) which do not need any end user effort as the algorithm itself adjusts automatically to the given problem. In these algorithms [9], the dynamic adaptation of operators' probabilistic rates on the level of the population with centralized control techniques is applied.

Instead of adjusting real parameters, setting variants were used, namely the types of selection (fitness proportional, rank-based, and tournament-based with three tournament sizes), crossover (one-point, two-point, as well as equiprobable, fitness proportional, rank-based, and tournament-based uniform crossovers [9]), population control and level of mutation (medium, low, high for two mutation types). Each of these has its own initial probability distribution which is changed as the algorithm executes.

This self-configuring technique can be used both for the genetic algorithm (SelfCGA). In [9] SelfCGA, performance was estimated on 14 test problems from [5]. The statistical significance was estimated with ANOVA.

Analysing the results related to SelfCGA [9], it can be seen that self-configuring evolutionary algorithms demonstrate higher reliability than the average reliability of the corresponding single best algorithm but sometimes worse than the best reliability of this algorithm.

SelfCGA can be used for the automated choice of effective structures and weight tuning of ANN-based predictors. For such purposes, classification accuracy can be used as a fitness function.

2.3. Semi-supervised ANN design by evolutionary algorithms

In the case of solving semi-supervised problems, the additional set $U = \{x_{l+1}, \dots, x_{l+u}\}$ of unlabelled training patterns is given. The artificial neural network has the aim of finding an optimal prediction function for unseen data based on both the labelled and the unlabelled part of the data [7].

In this study, self-training was used to learn from the unlabelled data. More specifically, the idea is to design a model with labelled data and then use the model's own predictions as labels for the unlabelled data to retrain a new model with the original labelled data and the newly labelled data and then iteratively repeat this process.

The problem with this method is that it can suffer from "semantic drift", where considering its own predictions as true labels can cause the model to drift away from the correct model. The model would then continue to mislabel data and use it again and continue to drift further and further away from where it should be. To prevent this problem, in [8] the model's predictions to label the data were used only when there was a high level of confidence about the predictions.

Generally, any supervised techniques contain two stages:

1. extracted attributes or the most relevant of them should be involved in the supervised learning process to adjust a classifier;
2. and then the trained classification model receives an unlabelled feature vector to make a prediction.

The method of genetic algorithm implementation in such a case was described above. However, in the case of semi-supervised techniques, the following basic steps have to be implemented [2]:

1. Train ANN on the labelled set;
2. Use the obtained ANN to classify all unlabelled instances from *U* by checking the confidence criteria;
3. Label instances from the set *U* if this is possible;
4. Repeat from the first step.

In the case of using a genetic algorithm for the ANN design in solving semi-supervised problems, it is necessary to make some decision (see Figure 1).

The first important question is: "Do we have to train just ANN weights or automatically design the ANN structure?". There are two possible answers (Fig.1a):

1. Only the weight coefficients of the ANN will be adjusted (SelfCGA-ANN-w);
2. The complete ANN will be designed, including both the ANN structure design and the adjusting of weights (SelfCGA-ANN).

The main question is: "Which ANN from the population of ANNs will be making the decision about labelling some example?". There are two possible answers (Fig.1b):

1. The best individual in the generation will be used for labelling examples if the confidence criterion is met (SelfCGA-ANN-Elitism);
2. All population members will vote and if the majority of them are confident in one decision, the example will be labelled (SelfCGA-ANN-Ensemble).

And the last question is: "How often should we stop the evaluation process and begin the process of labelling for test (unlabelled) data?".

1. The SelfCGA for the automated ANN structure design has to make a pause every 5 generations, try to label the data and after this continue its work with a new learning set (additional examples that took a label).
2. The SelfCGA for ANN weight training has to make a pause every 10 generations, try to label data and after that continue its work with a new learning set (additional examples that took the label).

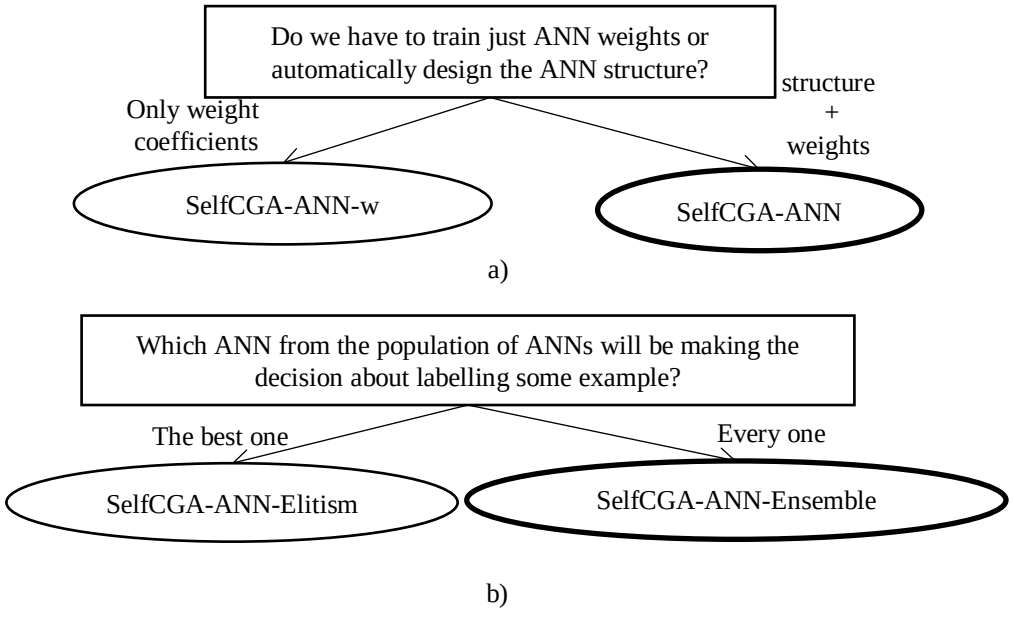

Fig. 1. Semi-supervised genetic algorithm classification.

In Table 1, all algorithm variants are presented.

Table 1. Algorithms names

Algorithm name	*SelfCGA-ANN-w*	*SelfCGA-ANN-Elitism*	*SelfCGA-ANN-Ensemble*
Weights or Structure?	*Only weights*	*Weights + Structure*	*Weights + Structure*
Who make decision?	*The best one*	*The best one*	*Every one*
How often will labelling be performed?	*every 10 generations*	*every 5 generations*	*every 5 generations*

3. EXPERAMENTAL RESULTS

At the first stage of experiments, we tested all algorithm variants on one artificial and two real-world problems that will be described in Table 2. All these data sets are classification problems and for the testing of semi-supervised techniques, each data set instance was randomly split into two parts: one labelled and one unlabelled – and different ratios for the particular settings were used.

Table 2. Data sets

Data Set Name	*Example's number*	*Input number*
Moons	*200*	*2*
Breast Cancer Wisconsin	*699*	*9*
Pima Indians Diabetes	*768*	*8*

First of all, one well-known artificial problem was considered, namely the two-dimensional "Moons" data set [6]. This problem is known to be a complex problem for semi-supervised techniques and a very simple problem for humans. This is why it is often used as a test problem for different machine learning algorithms and became a classical test problem for them. It consists of two groups of moon-like sets of points and it has a separating hyperplane between them. Thus, it has a non-linear structure that makes it difficult for semi-supervised support vector machines. In this experiment, the starting learning set contains only 4 labelled examples, 2 from one class and 2 from another one that were randomly chosen. All other examples must be labelled during the run.

The usual results obtained on the "Moons" problem are shown in Figure 2 (SelfCGA-ANN). As can be seen, the algorithms do not recognize all the points correctly. However, most of the points are in the right class. SelfCGA-ANN-Ensemble builds a quite complex separating hyperplane. The best result was shown by SelfCGA-ANN-Elitism. It usually made mistakes only on 1-2 points. It is probable that SelfCGA-ANN-Ensemble excessively averaged single ANN results.

Then two medical diagnostic problems, namely Breast Cancer Wisconsin and Pima Indian Diabetes [4], were solved. Both problems are binary classification tasks. For these data sets, 10 examples were randomly selected to be used as labelled examples, and the remaining instances were used as unlabelled data. The experiments are repeated 50 times and the average accuracies and standard deviations are recorded. Alternative algorithms (linear SVMs) for comparison are taken from [11]. The results are shown in Table 3.

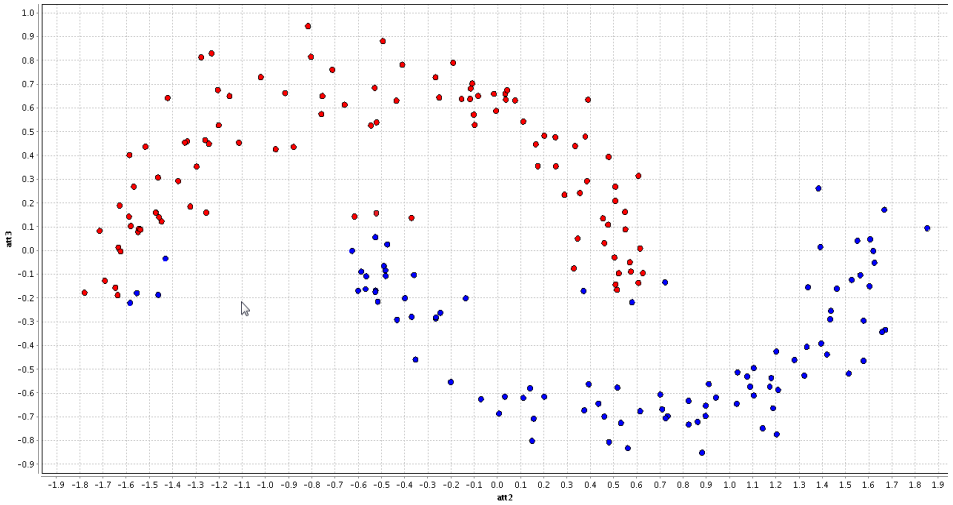

Fig. 2. Semi-supervised classification of "Moons" by SelfCGA-ANN-Ensemble

Table 3. Performance comparison for medical diagnostics problems

Algorithm's Name	*Breast Cancer Wisconsin*	*Pima Indians Diabetes*
TSVM	*89.2±8.6*	*63.4±7.6*
S3VM-c	*94.2±4.9*	*63.2±6.8*
S3VM-p	*93.9±4.9*	*65.6±4.8*
S3VM-us	*93.6±5.4*	*65.2±5.0*
SelfCGA-ANN-w	*94.8±2.1*	*66.7±2.3*
SelfCGA-ANN-Elitism	***96.5±1.9***	***69.4±1.8***
SelfCGA-ANN-Ensemble	*95.6±1.3*	*68.7±1.5*

As can be seen, SelfCGA-ANN is sufficiently effective for solving semi-supervised problems.

At the second stage of experiments, we tested all algorithm variants on speech-based emotion recognition problems. In the cases of both supervised and semi-supervised learning for speech-based nonlinguistic information extraction, some learning data is needed. Generally, any approach applied to this recognition problem contains the step of acoustic characteristic extraction.

An appropriate set of acoustic characteristics representing any speech signal was introduced at the INTERSPEECH 2009 Emotion Challenge. This set of features comprises attributes such as power, mean, root mean square, jitter, shimmer, 12 MFCCs, 5 formants and the mean, minimum, maximum, range and deviation of the pitch, intensity and harmonicity. The number of characteristics is 384. To get the conventional feature set introduced at INTERSPEECH 2009, the Praat [1] or OpenSMILE [3] systems might be used.

In this study, the emotional database was considered. It consists of labelled emotional utterances which were spoken by actors. Each utterance has one of the emotional labels: neutral or strong. The average time of one record is 2.7 seconds. It contains 3210 examples,

426 of them belonging to a neutral class. This problem had 384 features and only 321 randomly selected instances in the initial learning set (stratified sampling) and 2889 instances which were used as unlabelled ones. We used this dataset for the preliminary testing of semi-supervised techniques before the implementation in a real problem with unlabelled data.

Thus, during the algorithm run only 10% of the data set will be used as labelled data (321 examples). The rest will be considered as unlabelled.

The experiments are repeated 50 times and the average accuracies and range of variation are recorded in Table 4. In all experiments, the weighted accuracy was assessed to compare the quality of classification. The statistical robustness of the results obtained was confirmed by ANOVA tests, which were used for processing the received evaluations of our algorithms' performance.

The classification quality is relatively high even with only 10% of labelled examples in the training set. This result gives us the possibility of using a small amount of data labelled by experts with a huge amount of available unlabelled data for nonlinguistic information extraction in the future.

Table 4. Performance comparison for the emotion recognition problem

Algorithm's Name	*Average Accuracy*	*Worth and Best Accuracy*
SelfCGA-ANN-Elitism	***0.8864***	***[0.8794; 0.8901]***
SelfCGA-ANN-Ensemble	*0.8807*	*[0.8775; 0.8849]*
SelfCGA-ANN-w	*0.8582*	*[0.8534; 0.8623]*

4. CONCLUSION

The possibility of using semi-supervised classification for nonlinguistic information extraction is important due to the fact that getting labelled examples is often very expensive and sometimes must be repeated for any new person. Using unlabelled data during classification may be helpful. In this paper, semi-supervised ANNs were automatically designed by SelfCGA for solving semi-supervised classification problems in the field of speech-based emotion recognition. The results show that the proposed approaches are sufficiently effective for solving this kind of problems. The comparison of their results show that models with a more complex structure, for example, ANNs with a more flexible structure, can give better results.

The further development of the system is aimed at the expansion of its functionality by including the other types of IITs such as fuzzy logic systems [12] and multiobjective genetic algorithms [13], or by using additional variants of operators [14].

ACKNOWLEDGEMENT

The work is supported by Grant of the President of the Russian Federation for state support of young Russian scientists (MK- 3378.2017.9).

REFERENCES

[1] Boersma, P. Praat, a system for doing phonetics by computer. *Glot international.,* **9/10** (vol. 5), 2002, pp. 341–345.

[2] Chapelle, O., Zien ,A., Schoelkopf, B. (Eds.) Semi-supervised learning. *MIT Press*, 2006.
[3] Eyben, F., Wllmer, M., Schuller, B. Opensmile: the Munich versatile and fast opensource audio feature extractor. *Proceedings of the International Conference on Multimedia, ACM*, 2010, pp. 1459–1462.
[4] Frank, A. Asuncion, A. UCI Machine Learning Repository. *Irvine, CA: University of California, School of Information and Computer Science*, 2010, Available at: http://archive.ics.uci.edu/ml
[5] Finck, S. et al. Real-parameter black-box optimization benchmarking 2009. *Presentation of the noiseless functions. Technical Report Researh Center PPE*, 2009.
[6] Jain, A., Law, M. Data clustering: A user's dilemma. *LNCS*, **3776**, 2005, pp. 1-10.
[7] Joachims, T. Transductive inference for text classification using support vector machines. *International Conference on Machine Learning*, 1999.
[8] Ravi, S. Semi-supervised Learning in Support Vector Machines. *Project Report COS*, **521**, 2014.
[9] Semenkin, E.S., Semenkina, M.E. Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator. *LNCS*, **7331**, 2012, pp. 414-421.
[10] Zhu, X., Goldberg, A.B. Introduction to Semi-Supervised Learning. *Morgan and Claypool*, 2009.
[11] Li, Y.F., Zhou, Z.H. Improving Semi-Supervised Support Vector Machines Through Unlabeled Instances Selection. *The Twenty Fifth AAAI Conference on Artificial Intelligence*, 2011, pp. 386–391.
[12] Nikolova-Poceva, S., Iliev, A. Hybrid Fuzzy Regression Model for Determining Specific Active Power Generation Characteristic of Hydro Power Plants. *International Journal on Information Technologies & Security*, **1**, 2016, pp. 55-68.
[13] Brester, C., Ryzhikov, I., Semenkin, E. Restart Operator for Multi-Objective Genetic Algorithms: Implementation, Choice of Control Parameters and Ways of Improvement. *International Journal on Information Technologies and Security*, **4** (vol. 9), 2017, pp. 25-36.
[14] Sopov, E., Semenkin, E. Automated Synthesis of Selection Operators in Genetic Algorithms using Genetic Programming. *International Journal on Information Technologies & Security*, **4**, 2016, pp. 13-24.

Information about the author:

Semenkina Maria - PhD, Associate Professor of the Department of Higher Mathematics, Siberian State University of Science and Technology. Areas of scientific research are evolutionary computation, intelligent information techniques, modelling and optimization in complex systems. +7-391-2919119. semenkina88@mail.ru.

Manuscript (final version) received on 02 April 2018