Multiobjectivization for Classifier Parameter Tuning

Martin Pilát
Faculty of Mathematics and Physics,
Charles University in Prague,
Malostranské náměstí 25, Prague, CZ
Martin.Pilat@mff.cuni.cz

Roman Neruda
Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2, Prague 8, CZ
roman@cs.cas.cz

ABSTRACT

We present a multiobjectivization approach to the parameter tuning of RBF networks and multilayer perceptrons. The approach works by adding two new objectives – maximization of kappa statistic and minimization of root mean square error – to the originally single-objective problem of minimizing the classification error of the model. We show the performance of the multiobjectivization approach on five datasets.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*heuristic methods*; I.6.3 [**Simulation and Modeling**]: Applications

Keywords

Multiobjective optimization; classification; machine learning; evolutionary algorithm; multiobjectivization; parameter tuning

1. INTRODUCTION

Most classifiers have at least a few parameters, which more or less affect their performance on a given dataset. The values of these parameters need to be set before the classifier can be used to solve a particular task. The problem is that the best values of the parameters differ for different datasets. Although there are usually some guidelines for the parameter settings, finding the optimal values often requires expert knowledge, intuition, and trial-and-error. An appealing approach is to use an evolutionary algorithm to find the optimal parameters. However, many classifiers provide similar results for lots of different parameter setttings [7], thus making the search space difficult to explore.

In this paper we deal with the problem of tuning the parameters of Radial Basis Function (RBF) Networks [4] with the goal to provide settings which minimize the classification error. To this end we use the concept of multiobjectivization – solving the single-objective problem by means of multiobjective optimization. The key observation is that the additional objectives add more information and thus make the optimization easier. We compare the proposed approach to the performance of a simple single-objective evolutionary algorithm. To our best knowledge, there is no such comparison in the literature. We are also not aware of any application of multiobjectivization for parameter tuning.

2. MULTIOBJECTIVIZATION

Our goal is to provide good parameter settings for a given classifier. The quality of the settings is measured by the classification error of the classifier (the number of incorrectly classified instances). In order to improve the convergence rate of the algorithm, we add two more objectives whose values are not important for our task, but which are correlated to the error rate of the optimizer. Then, a multiobjective evolutionary algorithm is used to solve the multiobjective optimization problem.

The three objectives we optimize are: the classification error (minimization) (the percentage of incorrectly classified instances), kappa statistic [3] (maximization) (the classifier and training set agreement), and the root mean square error (minimization). The error rate and kappa statistic are highly correlated, especially in cases where all classes are represented by the same number of instances in the training set. This is also the reason to add the third objective, which may seem unrelated to classification.

Root mean square error (RMSE) is traditionally used as the objective which is minimized in regression tasks. As it is implemented here (i.e. the model is trained to predict the unary representation of the class label), the number does not tell much about the quality of the classifier itself. However, it is more sensitive to changes in the parameter settings. Moreover, RMSE guides the optimization to an optima of the classification error – if the RMSE is zero the classification error is also zero.

To solve the multiobjective optimization problem defined above, we use the well known NSGA-II multiobjective evolutionary algorithm [2]. All the error rates are computed using 10-fold cross-validation, which should reduce the risk of overfitting the model to the training set and also provide more robust results.

3. EXPERIMENTS

We implemented the algorithms described above to tune the parameters of the RBF network given bellow, together with their parameter settings. We optimized the number of clusters (integer between 2 and 10), the minimal width of the Gaussians (real number between 0.01 - 1.0), the ridge parameter for the logistic regression (real number between 0.000000001 and 10), and the maximum number of iterations for the logistic regression (integer between -1 and 50, -1 meaning "until convergence").

	Simple EA		
	best	average	std. dev.
balance-scale	0.0512	0.0712	0.0163
breast-w	**0.0286**	0.0308	0.0015
car	0.0741	0.0779	0.0039
haberman	0.2386	0.2474	0.0049
iris	0.0200	0.0300	0.0045
	Multiobjective EA		
	best	average	std. dev.
balance-scale	**0.0464**	**0.0498**	0.0015
breast-w	**0.0286**	**0.0296**	0.0011
car	**0.0712**	**0.0737**	0.0021
haberman	**0.2288**	**0.2395**	0.0064
iris	**0.0133**	**0.0200**	0.0060

Table 1: The results of different parameter tuners for RBF Network after 300 evaluations.

The performance of the tuners was tested on the following five datasets which are available from the UCI Machine Learning Repository [1] – balance-scale, breast-w, car, haberman, and iris.

The tuners were given the computational budget of 300 objective function evaluations. One evaluation is a 10-fold crossvalidation with the parameters given by the tuner on the respective training set. All evaluated individuals are saved in an archive and if the same individual is generated multiple times in the same run it is evaluated only once.

The single-objective evolutionary algorithm uses population of 10 individuals, one point crossover and Gaussian mutation with standard deviation equal to 30% of the range of the particular parameter. Moreover, the evolutinary algorithm uses tournament selection and 10% elitism (i.e. one individual). The multiobjective evolutionary algorithm uses again the same parameters, the only difference being in the selection phase, where it uses the NSGA-II selection based on dominance and crowding distance.

The results of the experiments are presented in Table 1, and they show that the multiobjective optimizer provides the best results in all the runs (when the minimum classification error is considered).

More specifically, on the balance-scale dataset we can observe the largest difference between the single-objective and the multiobjective optimizers. For this dataset, the multiobjective optimizer clearly wins and, moreover, after approximately 50 evaluations provides results, which are not provided by the other models even after 300 evaluations. We can see similar behaviour also on the car dataset.

On the breast-w dataset the performance of the two optimizers is similar, with the multiobjective optimizer being only slightly better on average.

On the iris dataset, the multiobjective optimizer was able to find a setting which was better than all settings we were able to obtain previously (using different optimizers like simulated annealing, grid search and random search). In the later phases, the average performance of the multiobjective optimizer in this case is the same as the best performance of the single objective optimizers.

4. CONCLUSION AND FUTURE WORK

We have shown that using multiobjectivization for the parameter tuning improves the performance of the hyperparameter optimizers significantly, mainly for the specific objective function with lots of plateuas. The additional objectives, which are not in fact directly important for the optimization task at hand, improve the results by providing direction in these plateau regions.

Some of the ranges of the parameters we used were quite limited. This corresponds to our future usage of parameter tuning – we would like to use meta-learning which would provide us with the model type, together with the region of interest of its parameters, and then we will use the parameter tuning to find the best possible settings of its parameters.

Another interesting future direction of research may be the use of surrogate multiobjective algorithms in this setting. We are somehow skeptical to the performance of the multiobjective algorithms which model each of the objectives separately (as the same problem as with one objective would also appear in the case of more objectives) but multiobjective optimizers with aggregate surrogate models (e.g. [5, 6] may be able to avoid or reduce this problem.

Acknowledgement

Roman Neruda was supported by Czech Science Foundation project no. P202/11/1368. Martin Pilát was supported by the Charles University Grant Agency project no. 345511.

5. REFERENCES

[1] D. N. A. Asuncion. UCI machine learning repository, 2007.

[2] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. M. Guervós, and H.-P. Schwefel, editors, *PPSN*, volume 1917 of *Lecture Notes in Computer Science*, pages 849–858. Springer, 2000.

[3] B. Di Eugenio and M. Glass. The kappa statistic: a second look. *Comput. Linguist.*, 30(1):95–101, Mar. 2004.

[4] S. Haykin. *Neural Networks: A Comprehensive Foundation (2nd Edition)*. Prentice Hall, 2 edition, July 1998.

[5] I. Loshchilov, M. Schoenauer, and M. Sebag. A mono surrogate for multiobjective optimization. In M. Pelikan and J. Branke, editors, *GECCO*, pages 471–478. ACM, 2010.

[6] M. Pilát and R. Neruda. ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2011*, pages 1202–1208. IEEE, 2011.

[7] M. Reif, F. Shafait, and A. Dengel. Meta-learning for evolutionary parameter optimization of classifiers. *Mach. Learn.*, 87(3):357–380, June 2012.