Regularization of Evolving Polynomial Models

Pavel Kordík

Dept. of Computer Science and Engineering, Karlovo nám. 13, 121 35 Praha 2, Czech Republic

kordikp@fel.cvut.cz

Abstract. *Black box models such as neural networks are popular because they can deliver reasonably accurate model almost instantly. Sometimes, it is more convenient to use a math model instead of black box model. Math models can be either designed by experts or automatically generated from data describing modelled systems. The disadvantage of generated math models is that they are often too complex to be understood by experts. In this contribution we experiment with regularization of generated models to enable automatic evolution of models that are both enough accurate and understandable. We limit our experiments to models consisting of polynomial transfer units.*

Keywords

Inductive modelling, regularization, polynomial networks.

1. Introduction

The proper regularization [1] is of crucial importance in the GMDH theory.

Which units are performing best and therefore should survive in a layer is decided using an external criterion [3] of regularity (CR).

There are several possible criteria applicable. The most popular is the criterion of regularity based on the validation using an external data set:

$$AB = \frac{1}{N_B}\sum_{i=1}^{N_B}(y_i(A) - d_i)^2 \rightarrow \min \tag{1}$$

where the $y_i(A)$ is the output of a GMDH model trained on the A data set.

Additional criterion to discriminate units that will be deleted is the variation accuracy criterion (VAC) [2, 4]

$$\delta^2 = \frac{\sum_{i=1}^{N}(y_i - d_i)^2}{\sum_{i=1}^{N}(d_i - \bar{d})^2} \rightarrow \min \tag{2}$$

where the y_i is the output of a GMDH model, d_i is the target variable and $\bar{d}$ is the mean of the target variable. With $\delta^2 < 0.5$ the model is good and when $\delta^2 > 1$, the modeling failed (the output unit should be deleted).

In the GAME engine [5] we use the validation set to prevent the overfiting. However further text shows that this type of regularization is not enough - it does not itself guarantee, that the model of optimal complexity will be evolved.

2. Evolving units (active neurons)

Input connections of units are evolved by means of the niching genetic algorithm described above. It is possible to evolve at the same time also transfer functions of units and their configuration. In the actual version of the GAME engine, only the CombiNeuron unit supports the evolution of its transfer function. We are working on extending the support of transfer function evolution also to other GAME units.

2.1. CombiNeuron - evolving polynomial unit

The GAME engine in the configuration generating homogeneous models with PolySimpleNeuron or CombiNeuron units only, can be classified as Multiplicative-additive (generalized) GMDH algorithm [6]. The transfer function of a polynomial unit can be either pseudo-randomly generated (as implemented in the PolySimpleNeuron unit[1]) or evolved as implemented in the CombiNeuron unit. To be able to evolve the transfer function we have to encode it into chromosome first. The encoding designed for the CombiNeuron unit is displayed in the Figure 1. The advantage of the encoding is that it keeps

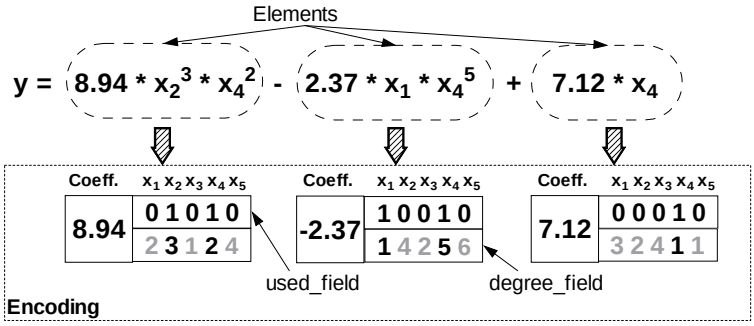

Fig. 1. Encoding of the transfer function for the CombiNeuron unit.

track of degrees of input features (degree_field although for some units particular features are disabled. When the transfer function was added into the chromosome[2], it was necessary to define evolutionary operators. The mutation operator can add/delete one element in the transfer function. It can also mutate

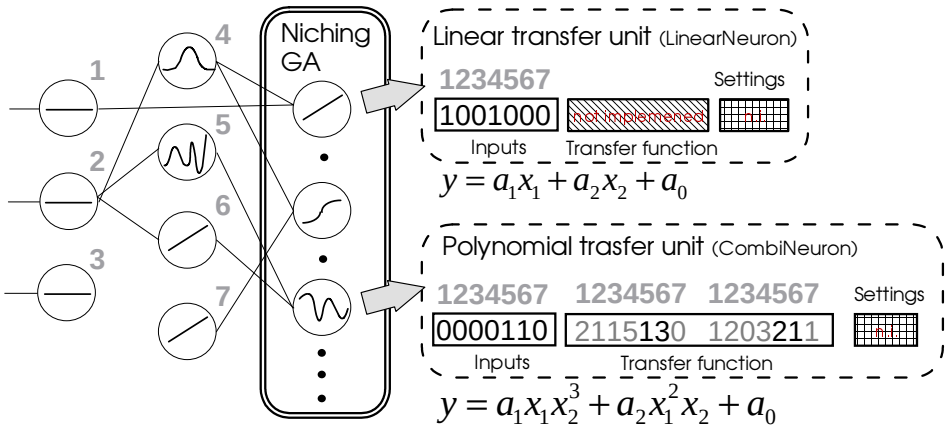

Fig. 2. Encoding of the transfer function for the CombiNeuron unit.

[1] In PolySimpleNeuron units, multiplicative-additive polynomials of complexity increasing with the number of layer in the network are generated pseudo-randomly.

[2] The Genome class was overridden by the CombiGenome class encoding both the inputs and the transfer function.

the degree of arbitrary input feature in arbitrary element of the transfer function.

The crossover operator simply combines elements of transfer functions of both parents. We plan to experiment with more sophisticated crossover techniques utilizing e.g. "historical marking" [7].

In case of noisy data, the CombiNeuron unit should be penalized for complexity to avoid the training data overfitting phenomenon (see Section 3).

When more types of units are enabled and the number of epochs of the niching GA together with the size of population is small, CombiNeuron units have not opportunity to evolve their transfer functions resulting in their poor performance and absence in final models. In this case we advise to reduce the number of unit types and change the inheritance configuration to $p0\%$ (all offsprings inherit their type from parents).

3. Regularization in GAME

The regularization is a methodology allowing to create models that are not too simple, not too complex for an appropriate task. Without any form of regularization, models often overfit a training data loosing generalization abilities and their performance on a new unseen data becomes extremely bad. The GMDH methods usually regularize models using an external data set called a validation set[3]. The criterion of regularity (in this case the error on the validation set) should be minimized:

$$CR_{RMS-val} = \frac{1}{N_B} \sum_{i=1}^{N_B} \left(y_i \left(A \right) - d_i \right)^2 \tag{3}$$

In some cases (e.g. few data records), it is possible to validate also on the training set:

$$CR_{RMS-tr\&val} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i \left(A \right) - d_i \right)^2 \tag{4}$$

Our experiments [5] showed, that the $CR_{RMS-tr\&val}$ regularization was unable to prevent overtraining for noisy data.

Other and very straightforward form of regularization is the penalization for complexity. There exist several criteria (AIC, BIC, etc.) developed in the information theory that can be applied to our problem.

We experimented with a regularization that can be written as:

$$CR_{RMS-R-val} = \left(\frac{1}{N_B} \sum_{i=1}^{N_B} \left(y_i \left(A \right) - d_i \right)^2 \right) * \left(1 + \frac{1}{R} * unit.PenaltyForComplexity() \right), \tag{5}$$

and with the version validating also on the training data $CR_{RMS-R-tr\&val}$. The value of the R coefficient is very important. When you look on the Figure 3, you can see the minimum of the criterion is changing its position with the value of the R parameter. For noisy data, it is better to have a stronger regularization ($R = 12$) and for no noise in data, no regularization is needed $R \to \infty$. To validate our assumptions, we have designed an experiment with an artificial data set, where we adjusted the level of noise from 0% up to 200%. We also generated 30 models for each noise level and different strength of the regularization (from R=12 to 3000).

Theoretically (Figure 4 left), the lowest error on the testing data should be achieved when the strength of the regularization match the level of noise in the data. The experimental result (Figure 4 right) matched our expectations except that for a low regularization ($R = 3000$) the testing error was low also for data with the medium level of noise. This deviation from the theoretical expectations can be caused by the fact, we used just the error of the simple ensemble of 30 models for all configurations (instead of the mean and standard deviation of their error). The ensemble techniques reduced overfitting of models for medium noise, but were unable to correct extremely overfitted models trained on highly noisy data.

[3]The validation set can be created by splitting the training data set into a part used for optimization of model and a part used for the validation.

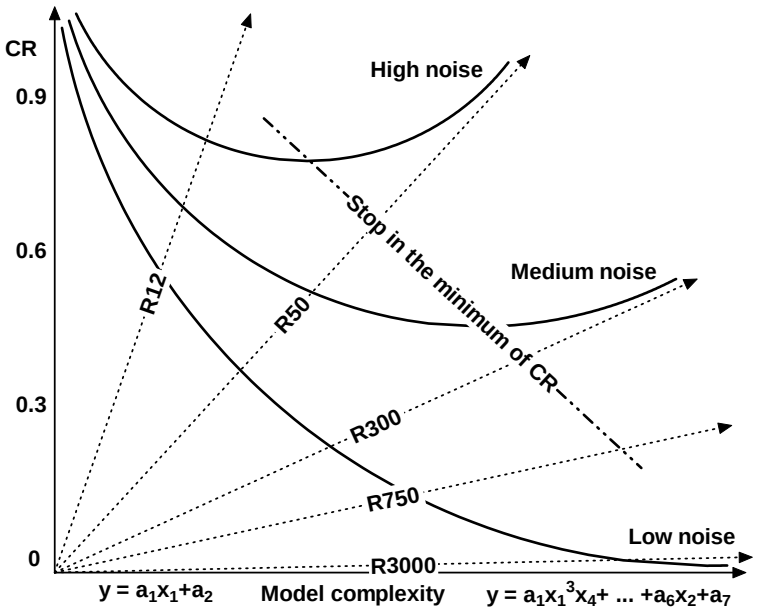

Fig. 3. As described in [4], the regularization criterion (CR) should be sensitive to the presence of noise in data. The complexity of models is increasing during training as new layers are added. Training should be stopped in the minimum of the CR. The penalization for complexity (R???) can be part of the CR, but the value should be adjusted according to the noise level in data.

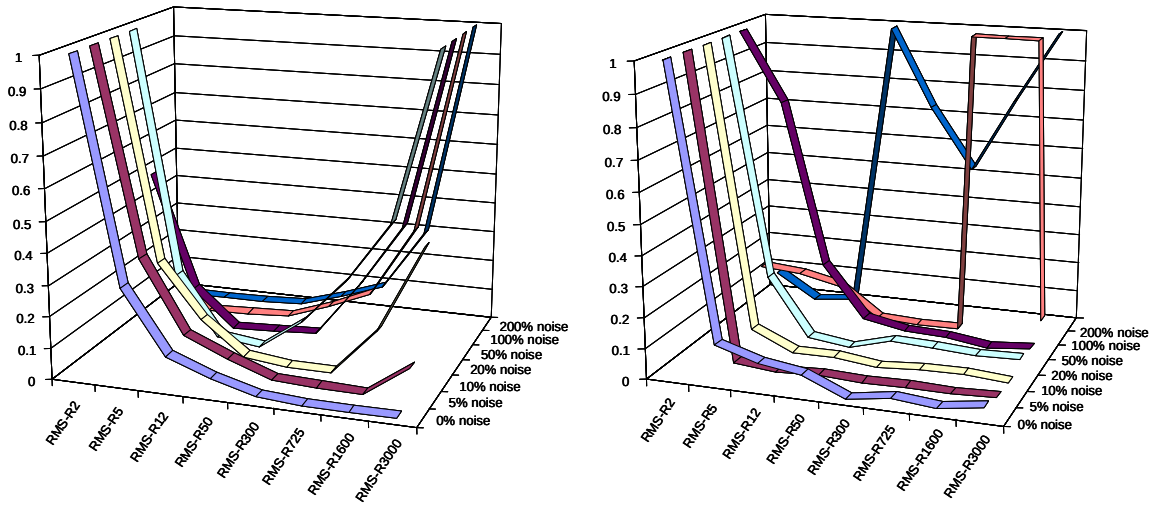

Fig. 4. The stronger penalization for complexity we use, the worse performance we can expect on complex problems with low noise. On the other hand, stronger penalization should perform better for a noisy data (left graph). Experimental measurements (right graph) are in accordance with the theoretical assumptions, except for low regularization and medium noise.

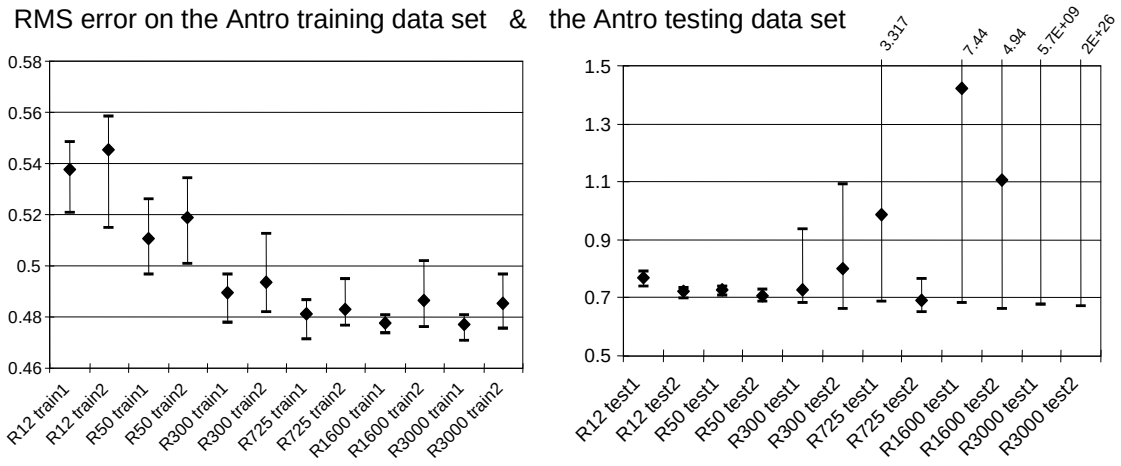

Fig. 5. Regularization of the Combi units by various strengths of a penalization for complexity. The error on Antro training data decreases and best result is achieved for almost no penalization (R3000). The optimal regularization on the testing data set is R300, where errors of individual models are the lowest and their variance is still reasonable.

3.1. Regularization of Combi units on real world data

We we also interested how the regularization affects the performance of the GAME models on real world data sets. We chose the Antro data set (description can be found in [5]) because our previous experiments showed that this data set is considerably noisy. We used two different splits into the training and testing sets (training1, testing1, training2, testing2) to reduce the bias of selecting unrepresentative data. For our experiments we enabled just the CombiNeuron units (see Section 2 for detailed description of this unit). We generated 30 models for each strength of penalization on both training sets. In the Figure 6 you can see the minimum, maximum and the mean error of these models. Results signify that on the Antro data set the optimal value of the R parameter in the Equation 5 is around 300.

The same experiment with the Building data set turned out absolutely differently for two variables with low level of noise (Cold and Hot water consumption). The best accuracy was achieved for the lowest regularization ($R3000$). For the third output variable (Energy consumption) that is considerably noisy, the error also decreased with lower penalty for complexity. In the configuration $R1600$ two from 30 models significantly overfitted the data and also the output of the simple ensemble demonstrated large error on the testing data. Therefore stronger penalization ($R725$) is optimal for the Energy consumption variable.

Both experiments with real world data sets showed that each output variable require different degree of regularization depending on amount of noise present in data vectors.

3.2. Evaluation of regularization criteria

In [4] it was proposed that the criterion of regularity should be changing its minima with a changing level of noise in a data set.

Such ”clever” criterion involves taking into account a noise level of an output variable. However the level of noise can be hardly estimated. The problem is that we cannot say if the variance is caused by noise or by a complex relationship.

Fig. 6. The error of the GAME ensemble on Building training data decreases with the declining penalization. The results on the testing data set show that no regularization is needed for the WBHW and WBCW variables. For the WBE variable that is much noisier than the other two and for low penalization levels models are overfitted.

We can assume that a noise level is correlated with the variance of the output variable in a data set[4]

$$\sigma^2 = \sum_{i=1}^{N} \left(d_i - \bar{d}\right)^2, \tag{6}$$

where d_i are target values of the output variable and $\bar{d}$ is the mean of these values.

Then the regularity criterion can be written as

$$CR_{RMS-p-n-val} = \left(\frac{1}{N_B} \sum_{i=1}^{N_B} \left(y_i\left(A\right) - d_i\right)^2 \right) * \left(1 + \sigma^2 * unit.PenaltyForComplexity()\right). \tag{7}$$

The penalty for complexity is then stronger for higher variance in a data set.

We have been experimenting with above proposed criteria on the artificial data set with various levels of noise. The results in the Figure 7 left indicate that for low levels of noise in data, it is better to validate on both the training and the validation set. For highly noisy data, this regularization (Equation 4) fails and it is better to validate on the validation set only (Equation 3), or use other criteria with the penalization for complexity (e.g. $CR_{RMS-p-n-tr\&val}$). The difference between the criterion with and without the variance considered is not significant (Figure 7 right). The overview of the criteria performance relative to the best results found during the experiment in Figure 4 can be found in the Figure 8.

Results showed that the regularization taking into account the variance of the output variable (Equation 7) is not better than the medium penalization for complexity (Equation 5 with $R = 300$). The problem is that we cannot say if the variance of the output variable is caused by noise or by a complex relationship. Additional research is needed to improve the results of the "adaptive" regularization. In this state of research, we recommend to use the medium penalization for complexity. Even better option would be when a noise level in the data set is supplied by a domain expert as an external information. Then we can adjust the coefficient R from the Equation 5 to the appropriate value. The Table 3.2 shows various strengths of regularization (penalization for complexity) on polynomial units. Results are in accordance with previous findings. The optimal value of the regularization constant R (Equation 5)is

[4]This assumption is not true for several regression data sets and for almost all classification problems.

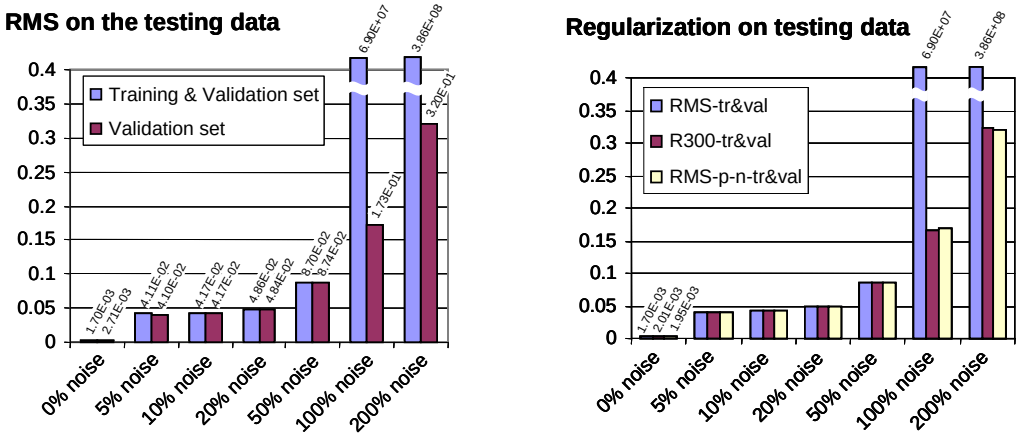

Fig. 7. For low noise levels in data, it is better to validate on both the training and the validation set. For noisy data, just the validation set should be used to prevent the overtraining.

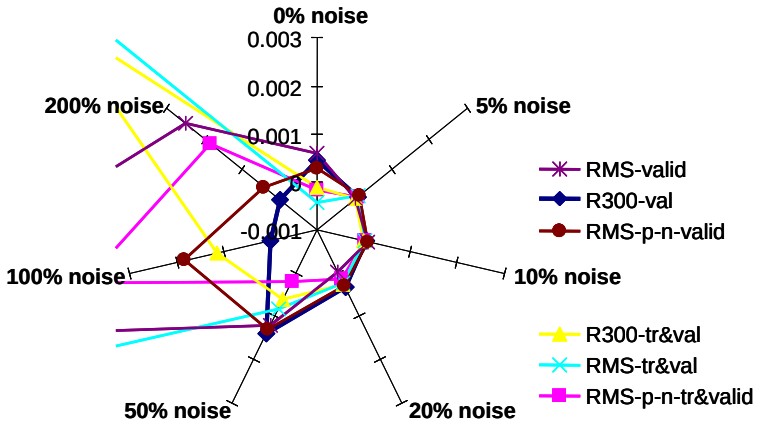

Fig. 8. We can observe similar results as in the previous figure. Regularization methods using both training and validation sets are better with no noise but fail with high noise levels in the data. The R300 regularization criterion proved to perform surprisingly well for all levels of noise. The RMS-p-n criterion should be further adjusted to penalize less on a low noise and more on a high noise levels.

Tab. 1. Results of regularized polynomials on the Antro data (10fold cross validation was used).

Cfg	Fold0	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	**Avg**
R0.1	18.36	17.59	19.46	16.77	16.95	16.63	16.22	17.44	18.07	18.36	**17.58**
R1	13.77	14.50	13.35	12.99	13.20	12.69	13.53	14.06	14.69	14.91	**13.77**
R10	12.75	14.38	13.08	12.09	12.74	12.63	12.84	11.96	13.36	14.20	**13.00**
R100	13.37	12.89	13.39	12.40	12.49	11.21	12.83	12.97	13.52	13.43	**12.85**
R1000	12.87	12.45	12.46	12.61	12.72	12.72	13.22	12.75	12.83	14.22	**12.88**
Linear	12.51	12.56	11.72	11.33	12.61	11.29	12.45	11.68	12.20	12.71	**12.11**

somewhere in between 100 and 1000. However, the best results of models with regularized polynomial units are still worse than the results of models with linear transfer function units (on the Antro data set).

4. Conclusion

In this paper, we described several possible criteria for regularization of polynomial inductive models. We recommend the medium penalization for complexity because we were unable so far to develop adaptive regularization based on the properties of the data set. In our future work we investigate why the regularized polynomial models perform worse than linear units on noisy data sets.

Acknowledgements

This research is partially supported by the grant Automated Knowledge Extraction (KJB201210701) of the Grant Agency of the Academy of Science of the Czech Republic and the research program "Transdisciplinary Research in the Area of Biomedical Engineering II" (MSM6840770012) sponsored by the Ministry of Education, Youth and Sports of the Czech Republic.

References

[1] P. Bearse and H. Bozdogan. Subset selection in vector autoregressive models using the genetic algorithm with informational complexity as the fitness function. *Systems Analysis, Modelling, and Simulation (SAMS)*, 31:61–91, 1998.

[2] V. P. Belogurov. A criterion of model suitabilityfor forcasting quantitative processes. *Soviet Journal of Automation and Information Sciences*, 23(3):21–25, 1990.

[3] A. Ivakhnenko, E. Savchenko, and G. Ivakhnenko. Gmdh algorithm for optimal model choice by the external error criterion with the extension of definition by model bias and its applications to the committees and neural networks. *Pattern Recognition and Image Analysis*, 12(4):347353, 2002.

[4] A. G. Ivakhnenko, G. Ivakhnenko, and J. Muller. Self-organization of neural networks with active neurons. *Pattern Recognition and Image Analysis*, 4(2):185–196, 1997.

[5] P. Kordík. *Fully Automated Knowledge Extraction using Group of Adaptive Models Evolution*. PhD thesis, Czech Technical University in Prague, FEE, Dep. of Comp. Sci. and Computers, FEE, CTU Prague, Czech Republic, September 2006.

[6] H. Madala and A. Ivakhnenko. *Inductive Learning Algorithm for Complex System Modelling*. CRC Press, 1994. Boca Raton.

[7] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002. Massachusetts Institute of Technology.