Automated Discovery of Polynomials by Inductive Genetic Programming

Nikolay Nikolaev[1] and Hitoshi Iba[2]

[1] Department of Computer Science, American University in Bulgaria, Blagoevgrad 2700, Bulgaria, nikolaev@nws.aubg.bg

[2] Department of Information and Communication Engineering, School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan, iba@miv.t.u-tokyo.ac.jp

Abstract. This paper presents an approach to automated discovery of high-order multivariate polynomials by inductive Genetic Programming (iGP). Evolutionary search is used for learning polynomials represented as non-linear multivariate trees. Optimal search performance is pursued with balancing the statistical bias and the variance of iGP. We reduce the bias by extending the set of basis polynomials for better agreement with the examples. Possible overfitting due to the reduced bias is conteracted by a variance component, implemented as a regularizing factor of the error in an MDL fitness function. Experimental results demonstrate that regularized iGP discovers accurate, parsimonious, and predictive polynomials when trained on practical data mining tasks.

1 Introduction

Inductive Genetic Programming (iGP) is considered a specialization of the Genetic programming (GP) paradigm [7] for automated knowledge discovery from data. The reasons for this specialization are [9]: 1) *inductive* knowledge discovery is a search problem and GP is a versatile framework for exploration of large search spaces; 2) GP provides *genetic* operators that can be tailored to the particular data mining task; and 3) GP flexibly reformulates *program* solutions. An advantage of iGP is that it automatically discovers the size of the solutions.

Previous research showed that iGP is successful for various data mining applications like: financial engineering [7], classification [2], time-series prediction [11], [5], etc.. A commonality in these evolutionary systems is that they discover non-linear model descriptions, and construct non-linear discriminant boundaries among the example data. This observation inspires us to consider Kolmogorov-Gabor polynomial models represented as non-linear multivariate trees.

An iGP system for evolutionary discovery of multivariate high-order polynomials, STROGANOFF [5], is enhanced. The intention is to achieve optimal search performance which acquires solutions that are not only parsimonious and accurate but also highly predictive. Our strategy to improving the performance trades-off between the statistical bias and the statistical variance of iGP. Statistical bias is the set of basis polynomials with which iGP constructs the target polynomials. Statistical variance is the deviation of the learning efficacy from one sample of examples to another sample that suggest the same target polynomial.

J.M. Żytkow and J. Rauch (Eds.): PKDD'99, LNAI 1704, pp. 456–461, 1999.

The iGP control balances the statistical bias and variance in the following way: 1) an extended set of basis polynomials is used to reduce the statistical bias, thus to fit flexibly the examples; 2) a regularization technique is applied to the fitness function to diminish the variance, in the sense of tendency to overfit the particularities and noise in the examples. The effect of balancing the statistical bias with a variance factor is increasing the degree of generalization and improving the global search performance.

We implement an iGP system with the regularized MDL function, proportional selection, and application of the crossover and mutation operators dependent on the tree size [9]. Mutation and crossover points are chosen with recombinative guidance by the largest error in the tree nodes [5]. Empirical evidence for the efficiency of this regularized iGP on data mining tasks is provided.

The next section of the paper gives the representation of multivariate high-order polynomials as trees and explains how an extended basis set impacts the search performance. Section three defines the regularized MDL-based fitness function. The iGP performance with this fitness formula is investigated in section four. Finally, a discussion is made and conclusions are derived.

2 Knowledge Discovery and Regression

The problem of inductive knowledge discovery can be formulated as a nonparametric regression problem. Given examples $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of instantiated vectors of independent variables $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{il}) \in \mathbb{R}^l$, and the dependent variable $y_i \in \mathbb{R}$, the goal is to find models $y = f(\mathbf{x})$. Due to noise perturbations, the goal becomes to find the *best approximation* $f(\mathbf{x})$ of the *regression function* $\tilde{f}(\mathbf{x}) = E[y|\mathbf{x}]$ by minimizing the empirical error on the examples. When the normal distribution is considered, the least squares fitting criterion is used to search for function $f(\mathbf{x})$ that minimizes the *average squared residual ASR*:

$$ASR = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 \tag{1}$$

where y_i is the true outcome of the i-th example, $f(\mathbf{x}_i)$ is the estimated outcome with this example $\mathbf{x}_i$, and N is the sample size.

2.1 Polynomials as Trees

We develop a knowledge discovery system that deals with high-order multivariate polynomials represented by *non-linear multivariate trees* [5]. They are interpreted as *Kolmogorov-Gabor polynomials*:

$$f(\mathbf{x}) = a_0 + \sum_i a_i x_i + \sum_i \sum_j a_{ij} x_i x_j + \sum_i \sum_j \sum_k a_{ijk} x_i x_j x_k + ... \tag{2}$$

There are three main *issues* in the automated discovery of polynomial models: 1) how to find the polynomial coefficients; 2) which terms with which variables to select; and 3) how to avoid overfitting with the example data.

2.2 Set of Basis Polynomials

The polynomials are modeled according to the Group Method of Data Handling (GMDH) [6]. A polynomial is a composition of *bivariate basis polynomials* in the nodes and independent variables in the leaves. We extend the basis Φ into a set with all *complete* and *incomplete*, first and second order polynomials in order to increase the flexibility of iGP to fit the data. The complete basis polynomial is:

$$f_j(\mathbf{x}) = \mathbf{a}^T\mathbf{h}(\mathbf{x}) \tag{3}$$

which if all coefficients in $\mathbf{a} = (a_0, a_1, ..., a_5)$ are non-zero, i.e. each $a_i \neq 0$, $0 \leq i \leq 5$, expands to:

$$f_j(\mathbf{h}(\mathbf{x})) = a_0h_0(\mathbf{x}) + a_1h_1(\mathbf{x}) + ... + a_5h_5(\mathbf{x}) \tag{4}$$

The vector $\mathbf{h} =(h_0(\mathbf{x}),\ h_1(\mathbf{x}),\ h_2(\mathbf{x}),\ h_3(\mathbf{x}),\ h_4(\mathbf{x}),\ h_5(\mathbf{x}))$ consists of simple functions h_i that produce the polynomial terms: $h_0(\mathbf{x}) = 1$, $h_1(\mathbf{x}) = x_1$, $h_2(\mathbf{x}) = x_2$, $h_3(\mathbf{x}) = x_1x_2$, $h_4(\mathbf{x}) = x_1^2$, and $h_5(\mathbf{x}) = x_2^2$.

The basis set is derived from the complete second-order polynomial (4). The total number of the incomplete polynomials is 25 from all 2^5 combinations of monomials $a_ih_i(\mathbf{x})$,$1 \leq i \leq 5$, and the leading constant term a_0, that contain both the variables x_1 and x_2. We use a subset $\{\Phi\} = 17$ of them after elimination of the symmetric polynomials. The target polynomial $f(\mathbf{x})$ is built by bottom-up tree traversal, and composing the basis polynomials in the nodes.

The benefit of this cascaded tree-like representation of the polynomials is that it allows tractable evaluation of complex, high-order models due to the composition of simple functions with parameters that are computed fast. The coefficients at each tree node are calculated with the matrix formula:

$$\mathbf{a} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y} \tag{5}$$

where $\mathbf{H}$ is a $N \times 6$ matrix of vectors $\mathbf{h}_i = (h_{i0}, h_{i1}, ..., h_{i5})$, $i = 1..N$, and $\mathbf{y}$ is a $N \times 1$ output vector. This is a solution of the *ordinary least-squares* (OLS) fitting problem by the method of normal equations.

2.3 Genetic Operators

The second issue in discovering high-order multivariate polynomials, raised in subsection 2.1, concerns the organization of the iGP search for polynomial terms. We use specific mutation and crossover operators.

The *mutation* operator is context-preserving [9]. It modestly transforms a tree with three elementary submutations: 1) substitution of an arbitrary node by another one; 2) insertion of a node as a parent of a subtree so that the subtree becomes leftmost child of the new node; and 3) deletion of a node only when no subtree below is to be cut. This mutation is applied with probability $p_m = m|g|^2$, where m is a free parameter and $|g|$ is size of tree g.

The iGP *crossover* splices two trees with probability $p_c = c/\sqrt{|g|}$, where c is a free parameter, or swaps them. This crossover operator produces offsprings with larger size than their parents if the parents are of very small size.

The convergence of the evolutionary iGP search process is accelerated by *recombinative guidance* [5]. The tree nodes in which the basis polynomials have largest ASR error are deterministically chosen for mutation and crossover.

3 Regularization Approach to iGP

The third issue in discovering polynomials, raised in subsection 2.1, concerns overfitting avoidance. The problem is to determine the polynomial terms and coefficients with which it optimally approximates the data without overfitting. Criteria for learning parsimonious and accurate models are given by the *Minimum Description Length* (MDL) principle. The *regularization theory* provides heuristics for increasing the predictability.

3.1 The MDL-Based Fitness Function

Adapted for the purpose of polynomial discovery, the MDL principle can be stated as follows: given a set of examples and an effective enumeration of their polynomial models, prefer with greatest confidence the polynomial which has together high *learning accuracy*, and low *structural complexity*. We adopt the following MDL-based *fitness function* [1]:

$$MDL = ASR + \frac{A}{N}\sigma^2 \log(N) \tag{6}$$

where ASR is the average squared residual, A is the number of coefficients, N are the examples, σ^2 is a rough estimate of the error variance.

3.2 Statistical Bias and Variance

When trying to find polynomials from a fixed and finite example set the ASR error may be low but this is not enough to anticipate a high generalization. The reason is that often the examples are noisy. This may be combat by decomposing of the error into a statistical bias and a variance component [4]. The *statistical bias,* proportional to $(E_D[f(\mathbf{x})] - E[y|\mathbf{x}])^2$, accounts only for the degree of fitting the examples, but not for the level of extrapolation. This is the *variance,* proportional to $E_D[(f(\mathbf{x}) - E_D[f(\mathbf{x})])^2]$, that accounts for the generalization.

The risk of overfitting the examples could be minimized if a variance factor is added to the error component of the fitness function. We introduce a *correcting complexity* that penalizes large coefficients in a *regularized average error* RAE:

$$RAE = \frac{1}{N}(\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 + k\sum_{j=1}^{A} a_j^2) \tag{7}$$

where k is a regularization parameter. Motivation for this definition is that large coefficients imply a fluctuating polynomial with large amplitudes that overfits the examples, while small coefficients imply more "regular" approximation. We use RAE in the MDL function instead ASR, which is called from now on MDL_R.

The manner for calculating the polynomial coefficients can be derived from the minimum of the function $\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 + k\sum_{j=1}^{A} a_j^2$ with respect to the coefficients a_j, $1 \leq j \leq A$, assuming the least squares fitting criterion:

$$\mathbf{a} = (\mathbf{H}^T\mathbf{H} + k\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y} \tag{8}$$

where $\mathbf{I}$ is the identity matrix. We select values for k relying on a proof that as long as $0 < k < 2\sigma^2/\mathbf{a}'\mathbf{a}$ the mean squared error of the identified polynomial is smaller than this of the best estimator without correction.

4 Search Performance

We present experimental results trying to answer the questions: whether the *regularization approach* to iGP can find polynomials with better predictive capacities than the *ordinary* iGP, assuming the old system STROGANOFF? How does the regularization affects the generalization quality?

The iGP performance is studied with four data mining tasks from the machine learning repository [8]: *Iris*, *Ionosphere*, *Glass*, *Credit*, and *Vehicle*. Because of the character of the system, experiments were conducted with iGP to find one polynomial for each particular class from each of these tasks.

In Table 1 we give results, measured by the percentage of correctly recognized training examples. The two iGP systems are related to a knowledge discovery system *Ltree* [3] that uses oblique decision trees, since it also finds non-linear approximations of the data, and to the decision tree learning system *C4.5* [10]. An oblique or a decision tree classifies the data into all classes, and the comparison with the polynomials is not straightforward. That is why, we computed the average iGP variances from each group of polynomials that together learn all classes of a task. The variances are evaluated by 10-fold cross-validation.

Data/class	Best	*Ordinary* iGP	*Regularized* iGP	*Ltree*	*C4.5*
Iris	variance(%)	99.74±0.16	99.95±0.21	97.15±2.85	95.15±4.85
	accuracy(%)	99.965	99.982	100	100
Ionosphere	variance(%)	89.23±0.75	90.14±0.26	90.6±4.0	90.9±5.0
	accuracy(%)	92.44	93.37	94.6	95.9
Glass	variance(%)	72.55±5.04	78.10±3.62	65.5±8.0	67.7±12.0
	accuracy(%)	80.05	82.18	73.7	79.7
Credit	variance(%)	77.15±3.48	81.06±2.52	73.6±5.0	70.9±4.0
	accuracy(%)	81.53	83.75	78.6	74.9
Vehicle	variance(%)	75.11±5.12	79.47±2.85	77.5±5.0	71.2±4.0
	accuracy(%)	80.22	83.44	82.5	75.2

Table 1. Variance of the discovered tree-models by *Regularized* iGP using *RAE* with $k = 0.01$, *Ordinary* iGP, *Ltree* and *C4.5*, and accuracy of the best tree-models.

Table 1 shows that the regularized iGP converges to more accurate polynomials than the ordinary STROGANOFF, and also the variances of the regularized polynomials are smaller, therefore they feature higher generalization.

One may observe that iGP discovers better solutions of complex tasks, like *Glass*, *Credit*, and *Vehicle*. The reason is that iGP may evolve very high-order polynomials that closely approximate the data. The non-evolutionary systems *Ltree* and *C4.5* are slightly better on the simpler data sets *Iris* and *Ionosphere*.

On the complex tasks iGP discovers shorter trees, for example of size 8 with 24 coefficients on the *Glass* data while *Ltree* produces trees of size approximately 34 and *C4.5* acquires trees of size 44. Although the iGP system induces trees of almost equal size to these learned by *Ltree* and *C4.5* on the simple tasks, 5 on the *Iris* data set and respectively 12, 15, 19 on the *Ionosphere* data set, the evolved polynomials include more terms.

5 Discussion

An essential advantage of this iGP is that the polynomial coefficients are rapidly computed as least-squares solutions by the method of normal equations. The regularization improves the previous ordinary iGP STROGANOFF providing a reliable scheme for computing the coefficients and, thus, avoiding problems of ill-posedness of the example matrix.

The iGP approaches from the STROGANOFF family as well as GMDH [10] and MAPS [2] have the ability to find automatically the structure of the polynomials. iGP performs search in the space of whole polynomials, while the other iteratively grow a single polynomial. When GMDH and MAPS learn one polynomial layer by layer, they constrain the feeding of higher tree layers since they restrict the search considering subsets of some plausible basis polynomials.

6 Conclusion

This paper contributes to the research into discovery of high-order multivariate polynomials by iGP. It demonstrated an iGP system that can be used for applied nonparametric approximation due to the following advantages: 1) it discovers automatically the model structure; 2) it generates explicit analytical representations in the form of polynomials; and 3) it makes the polynomials well-conditioned suitable for practical purposes.

References

1. Barron, A.R., and Xiao, X. (1991) Discussion on MARS. *Annals of Statistics*, 19: 67–82.
2. Freitas, A. A. (1997) A Genetic Programming Framework for two Data Mining Tasks: Classification and Generalized Rule Regression. In *Genetic Programming 1997: Proc. of the Second Annual Conference*, 96–101, Morgan Kaufmann, CA.
3. Gama, J. (1997) Oblique Linear Tree, In X.Liu, P.Cohen, and M.Berthold (Eds.), *Advances in Intelligent Data Analysis IDA-97*, 187–198, Springer, Berlin.
4. Geman, S., Bienenstock, E., Doursat, R. (1992) Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, **4**(1): 1-58.
5. Iba, H., and de Garis, H. (1996) Extending Genetic Programming with Recombinative Guidance. In *Advances in Genetic Programming 2*, The MIT Press, 69–88.
6. Ivakhnenko, A. G. (1971) Polynomial Theory of Complex Systems. *IEEE Trans. on Systems, Man, and Cybernetics*. **1**(4): 364–378.
7. Koza, J. R. (1992) *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge.
8. Merz, C. J., and Murphy, P. M. (1998) UCI Repository of machine learning databases, Irvine, CA: University of California, Dept. of Inf. and Computer Science `[www.ics.uci.edu/ mlearn/MLRepository.html]`.
9. Nikolaev, N., and Slavov, V. (1998) Concepts of Inductive Genetic Programming. In *EuroGP'98: First European Workshop on Genetic Programming*, LNCS-1391, 49–59, Springer, Berlin.
10. Quinlan, R. (1993) *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.
11. Zhang, B.-T., and Mühlenbein, H. (1995). Balancing Accuracy and Parsimony in Genetic Programming. *Evolutionary Computation* **3**(1):17-38.