Prediction of Dental Milling Time-error by Flexible Neural Trees and Fuzzy Rules

Pavel Krömer[1,2], Tomáš Novosád[1], Václav Snášel[1,2], Vicente Vera[4], Beatriz Hernando[4], Laura García-Hernandez[5], Héctor Quintián[3], Emilio Corchado[2,3], Raquel Redondo[6], Javier Sedano[5], and Alvaro E. García[4]

[1] Dept. of Computer Science, VŠB-Technical University of Ostrava, Czech Republic
[2] IT4Innovations, Ostrava, Czech Republic
{pavel.kromer,tomas.novosad,vaclav.snasel}@vsb.cz
[3] Departamento de Informática y Automática, Universidad de Salamanca, Spain.
escorchado@usal.es
[4] Facultad de Odontología, UCM, Madrid, Spain. {vicentevera, aegarcia}@odon.ucm.es
[5] Department of Civil Engineering, University of Burgos, Burgos, Spain.
rredondo@ubu.es
[6] Dept. of AI & Applied Electronics, Castilla y León Technological Institute, Burgos, Spain. javier.sedano@itcl.es
[7] Area of Project Engineering, University of Cordoba, Spain
ir1gahel@uco.es

Abstract. This multidisciplinary study presents the application of two soft computing methods utilizing the artificial evolution of symbolic structures – evolutionary fuzzy rules and flexible neural trees – for the prediction of dental milling time-error, i.e. the error between real dental milling time and forecast given by the dental milling machine. In this study a real data set obtained by a dynamic machining center with five axes simultaneously is analyzed to empirically test the novel system in order to optimize the time error.

Keywords: dental milling, prediction, evolutionary algorithms, flexible neural trees, fuzzy rules, industrial applications

1 Introduction

Accurate scheduling and planning becomes increasingly important part of modern industrial processes. To optimize the manufacturing of products and schedule the utilization of devices, the product manufacturing time has to be known in advance. However, the predictions given by traditional methods and tools are often less accurate. Precise prediction of product manufacturing time is important for industrial production planning in order to meet, industrial, technological, and economical objectives [2, 14]. The goal of a production process is to deliver products on time and utilize the resources at maximum during production cycles. Production time estimates provided either by production models (i.e. by

auxiliary software) or human experts are often less accurate than desirable [2]. Soft computing techniques can be used for flexible and detailed modelling of production processes [5]. The area of soft computing represents a set of various technologies involving non-linear dynamics, computational intelligence, ideas drawn from physics, physicology and several other computational frameworks. It investigates, simulates and analyzes very complex issues and phenomena in order to solve real-world problems: such as the failures detection in dental milling process, which requires a multidisciplinary approach [12].

In this study, a real data set obtained by a dynamic machining center with five axes simultaneously is analyzed by means of two soft computing techniques to empirically test the system in order to optimize the time error.

2 Flexible neural tree

Flexible neural tree (FNT) [3] is a hierarchical neural network, which is automatically created in order to solve given problem. Its structure is usually determined using some adaptive mechanism and it is intended to adapt to the problem and data under investigation [11, 10, 4]. Due to this property of the FNTs, it is not necessary to setup some generic static network structure not related to the problem domain beforehand.

A general and enhanced FNT model can be used for problem solving. Based on the predefined instruction/operator sets, a FNT model can be created and evolved. In this approach, over-layer connections, different activation functions for different nodes and input variables selection are allowed. The hierarchical structure could be evolved by using genetic programming. The fine tuning of the parameters encoded in the structure could be accomplished by using parameter optimization algorithms. The FNT evolution used in this study combines both approaches. Starting with random structures and corresponding parameters, it first tries to improve the structure and then as soon as an improved structure is found, it fine tunes its parameters. It then goes back to improving the structure again and, provided it finds a better structure, it again fine tunes the rules' parameters. This loop continues until a satisfactory solution is found or a time limit is reached. A tree-structural based encoding method with specific instruction set is selected for representing a FNT model in this research. The reason for choosing the representation is that the tree can be created and evolved using the existing or modified tree-structure-based approaches.

The fitness function maps the FNT to a scalar, real-valued fitness values that reflect the FNT's performances on a given task. Firstly the fitness functions should be seen as error measures, i.e. mean square error (MSE) or root mean square error ($RMSE$). A secondary non-user-defined objective for which algorithm always optimizes FNTs is FNT size as measured by number of nodes. Among FNTs with equal fitness smaller ones are always preferred. MSE and

$RMSE$ are given by:

$$MSE(i) = \frac{1}{P}\sum_{j=1}^{P}(y_1^j - y_2^j)^2, \qquad RMSE(i) = \sqrt{MSE(i)} \tag{1}$$

where P is the total number of samples, y_1^j and y_2^j are the actual time-series and the FNT model output of j-th sample. $MSE(i)$ and $RMSE(i)$ denotes the fitness value of i-th individual.

2.1 Tree structure and parameter learning

Finding an optimal or near-optimal flexible neural tree can be accomplished by various evolutionary and bio-inspired algorithms [11, 10, 4]. The general learning procedure for constructing the FNT model can be described in high level as follows [3]:

1. Set the initial values of parameters used in the GA algorithms. Set the elitist program as NULL and its fitness value as a biggest positive real number of the computer at hand. Create a random initial population (flexible neural trees and their corresponding parameters)
2. Structure optimization by genetic algorithm, in which the fitness function is calculated by MSE or RMSE
3. If a better structure is found and no better structure is found for certain number of generations, then go to step (4), otherwise go to step (2)
4. Parameter optimization by genetic algorithms. In this stage, the tree structure or architecture of flexible neural tree model is fixed, and it is the best tree taken from the sorted population of trees. All of the parameters used in the best tree formulated a parameter vector to be optimized by local search
5. If the maximum number of local search is reached, or no better parameter vector is found for a significantly long time then go to step (6); otherwise go to step (4);
6. If satisfactory solution is found, then the algorithm is stopped; otherwise go to step (2).

Evolutionary methods [1] are in this study used for FNT structure optimization as well as for activation function parameters and tree nodes weights optimization. The selection, crossover and mutation operators used are same as those of standard genetic programming [1]. A genetic algorithm starts with selection of two parents from current population. The product of crossover operator can be one or more offspring - two in this study. The mutation of offspring is performed at the last step of genetic algorithm. After these three steps we have new offspring which is placed into a newly created population. The process is repeated until desired new population is built. As soon as the new population is built, the new population is evaluated and sorted according to the fitness function. Selection is in the FNT evolution implemented using the weighted roulette wheel algorithm and the tree structure crossover is implemented as an exchange

of randomly selected subtrees of parent chromosomes. The crossover of node weights and activation function parameters is done in a similar way as in previous studies applying genetic algorithms to neural network training [6]. A variety of FNT mutation types were used:

1. Changing one terminal node: randomly select one terminal node in the neural tree and replace it with another terminal node.
2. Changing one function node: randomly select one function node and replace it with a newly generated subtree.
3. Growing: select a random function node in hidden layer of the neural tree and add newly generated subtree as a new child.
4. Pruning: randomly select a node in the neural tree and delete it in the case the parent node has more than two child nodes.

The mutation of tree weights and activation function parameters is the same as in the genetic algorithms for artificial neural networks [6].

3 Fuzzy rules evolved by genetic programming

Fuzzy rules (FR) [7, 8, 13] inspired by the area of fuzzy information retrieval (IR) [9] and evolved by genetic programming have been shown to achieve interesting results in the area of data mining and pattern analysis.

The fuzzy rules use similar data structures, basic concepts, and operations as the fuzzy information retrieval but they can be used for the analysis (i.e. classification, prediction) of general data. A fuzzy rule has the form of a weighted symbolic expression roughly corresponding to an extended Boolean query in the fuzzy IR analogy. The rule consists of weighted feature (attribute) names and weighted aggregation operators. The evaluation of such an expression assigns a real value from the range $[0, 1]$ to each data record. Such a valuation can be interpreted as an ordering or a fuzzy set over the data records. The fuzzy rule is a symbolic expression that can be parsed into a tree structure. The tree structure consists of nodes and leaves (i.e. terminal nodes). An example of fuzzy rule is give below:

feature1:0.5 and:0.4 (feature2[1]:0.3 or:0.1 ([1]:0.1 and:0.2 [2]:0.3))

In the fuzzy rule syntax can be seen three types of nodes: the feature node is defined by feature name and its weight (*feature1:0.5*) and represents a requirement on current value of a feature, past feature node is defined by feature name, index of previous record, and weight (*feature2[1]:0.3*) and it is requirement on previous value of a feature. Finally, the past output node is defined by the index of previous output and weight (*[1]:0.5*) and represents a requirement on previous value of the predicted output variable. Clearly, such a fuzzy rule can be used for the analysis of both, data sets consisting of independent records and time series.

The fuzzy rules are evaluated using the formulas and equations from the area of fuzzy IR and fuzzy sets (see e.g. [7, 8, 13]). The terminal node weights are interpreted as threshold for data feature values and operator nodes are mapped

to fuzzy set operators. The fuzzy rule predicting certain value for a given data set is found using standard genetic programming that evolves a population of tree representations of the rules in a supervised manner. The whole procedure is very similar to the evolution of the FNT structure described in section 2.1 but it differs in the choice of the fitness function which is taken from the area of fuzzy IR. The correctness of search results in IR can be evaluated using the measures precision P and recall R. Precision corresponds to the probability of retrieved document to be relevant and recall can be seen as the probability of retrieving a relevant document. Precision and recall in the extended Boolean IR model can be defined using the $\Sigma-$count $\|A\|$ [15]:

$$\rho(X|Y) = \begin{cases} \frac{\|X \cap Y\|}{\|Y\|} & \|Y\| \neq 0 \\ 1 & \|Y\| = 0 \end{cases}, \; P = \rho(REL|RET), \; R = \rho(RET|REL) \quad (2)$$

where REL stands for the fuzzy set of all relevant documents, RET for the fuzzy set of all retrieved documents, and $\|A\|$ is the $\Sigma-$count, i.e. the sum of the values of characteristic function μ_A for all members of the fuzzy set $\|A\| = \sum_{x \in A} \mu_A(x)$ [15]. The F-score F is among the most used scalar combinations of P and R:

$$F = \frac{(1 + \beta^2)PR}{\beta^2 P + R} \quad (3)$$

For the evolution of fuzzy rules [7, 8, 13] we map the prediction given for training data set by the fuzzy rule to RET and the desired values to REL. F corresponds to the similarity of two fuzzy sets and a fuzzy rule with high F provides good approximation of the output value.

4 Industrial case study and experiments

FNTs and FRs were evolved for the estimation of time-error in a real dental milling process. The data was gathered by means of a Machining Milling Center of HERMLE type-C 20 U (iTNC 530), with swivelling rotary (280 mm), with a control system using high precision drills and bits (see fig. 1) and it was used for the evolution of FNTs and FRs by the genetic programming. The models were trained using an initial data set of 98 samples obtained by the dental scanner in the manufacturing of dental pieces with different tool types (plane, toric, spherical and drill). The data set contained records consisting of 8 input variables (Tool, Radius, Revolutions, Feed rate X, Y and Z, Thickness, Initial Temperature) and 1 output variable (Time Error for manufacturing) as shown in table 1. Time error for manufacturing is the difference between the time estimated by the machine itself and real production time. Negative values indicate that real time exceeds estimated time. The goal of this study was to evaluate the ability of evolutionary evolved FNT and FR to predict the dental milling time-error from the data. The parameters used for the evolution of the FNT and FR are shown in table 2. They were selected on the basis of initial experiments and past

(a) Machining/ Milling Center of HERMLE type-C 20 U (iTNC 530)

(b) Manufacturing of metal pieces

Fig. 1: Machining/ Milling Center of HERMLE type-C 20 U (iTNC 530)

Table 1: Description of variables in the data set.

Variable (Units)	Range of values
Type of tool	Plane, toric, spherical and drill
Radius (mm.)	0.25 to 1.5
Revolutions per minute (RPM)	7,500 to 38,000
Feed rate X (mm. by minute)	0 to 3,000
Feed rate Y (mm. by minute)	0 to 3,000
Feed rate Z (mm. by minute)	50 to 2,000
Thickness (mm.)	10 to 18
Temperature (°C)	24.1 to 31
Real time of work (s)	6 to 1,794
Time errors for manufacturing (s)	-28 to -255

experience with the methods. Because the data set was rather small, all of it was used to train the predictors. The average prediction error ($RMSE$) for both methods on the full dental milling data set is shown in table 3. The table shows that both methods were able to learn the trends in the data set with a similar accuracy of 97.55 % and 98.0%. Visual illustration of the time-error prediction by FNT and FR is shown in fig. 2 and fig. 3 respectively.

5 Conclusions

This study presents initial results on the prediction of dental milling time-error by flexible neural trees and fuzzy rules evolved by artificial evolution. Both soft computing models were trained on a real-world data set describing the production of a dental milling machine and their ability to adapt to the data was compared. Both methods have shown very similar ability to predict the dental milling time-error, the former reaching 0.45 % higher prediction accuracy.

In the future, the prediction of dental milling time-error by both methods will be evaluated in a statistically sounder manner (e.g. by n-fold cross validation) to obtain better comparison of both methods. Moreover, precise production models can be used for further optimization of the industrial processes and device parameters.

Table 2: FNT and FR evolution parameters.

Method	Parameters
FNT	pop. size 100, crossover probability P_C 0.8, mutation probability P_M 0.2, limiting number of 10 generations, fitness function $RMSE$, Gaussian activation function with a, b, and weights from the range $[0, 1]$
FR	pop. size 100, crossover probability P_C 0.8, mutation probability P_M 0.2, limiting number of 1000 generations, no past feature nodes and no past output nodes allowed, fitness function F-Score with $\beta = 1$

Table 3: Dental milling time-error prediction accuracy.

Method	Prediction error (sec)	Prediction accuracy (pct)
FNT	4.52	98.00
FR	5.55	97.55

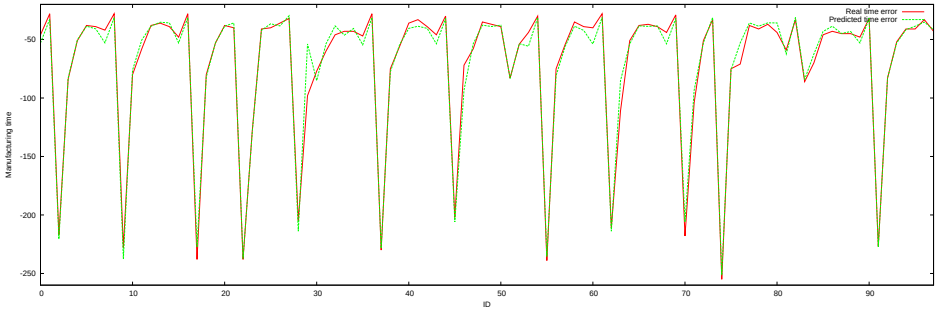

Fig. 2: Visual results of the prediction by FNT.

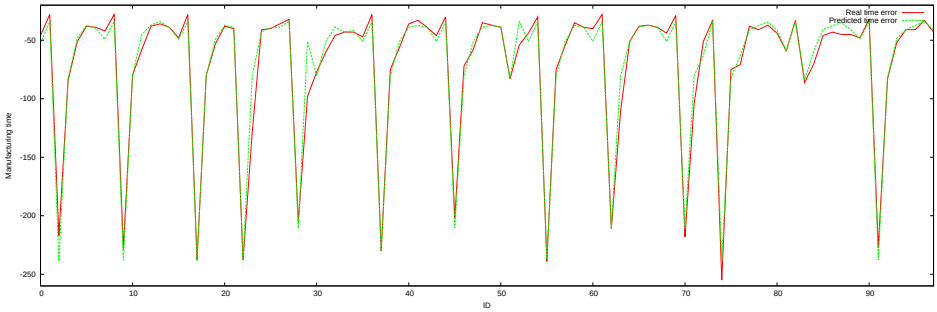

Fig. 3: Visual results of the prediction by FR.

Acknowledgements

This research is partially supported through a projects of the Spanish Ministry of Economy and Competitiveness [ref: TIN2010-21272-C02-01] (funded by the European Regional Development Fund). This work was also supported in the framework of the IT4Innovations Centre of Excellence project, reg. no.

CZ.1.05/1.1.00/02.0070 by operational programme “Research and Development for Innovations“ funded by the Structural Funds of the European Union and state budget of the Czech Republic.

References

1. Affenzeller, M., Winkler, S., Wagner, S., Beham, A.: Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications. Chapman & Hall/CRC (2009)
2. Chang, P., Liao, T.: Combining som and fuzzy rule base for flow time prediction in semiconductor manufacturing factory. Applied Soft Computing 6(2), 198 – 206 (2006)
3. Chen, Y., Abraham, A.: Flexible neural tree: Foundations and applications. In: Tree-Structure based Hybrid Computational Intelligence, Intelligent Systems Reference Library, vol. 2, pp. 39–96. Springer Berlin Heidelberg (2010)
4. Chen, Y., Yang, B., Meng, Q.: Small-time scale network traffic prediction based on flexible neural tree. Appl. Soft Comput. 12(1), 274–279 (Jan 2012)
5. Custodio, L.M.M., Sentieiro, J.J.S., Bispo, C.F.G.: Production planning and scheduling using a fuzzy decision system. IEEE Trans. on Robotics and Automation 10(2), 160–168 (1994)
6. Ding, S., Li, H., Su, C., Yu, J., Jin, F.: Evolutionary artificial neural networks: a review. Artificial Intelligence Review pp. 1–10, (2011)
7. Krömer, P., Platoš, J., Snášel, V., Abraham, A.: Fuzzy classification by evolutionary algorithms. In: IEEE Int. Conf. on Systems, Man, and Cybernetics. pp. 313 – 318. IEEE SMC Society (2011)
8. Krömer, P., Platoš, J., Snášel, V., Abraham, A., Prokop, L., Mišák, S.: Genetically evolved fuzzy predictor for photovoltaic power output estimation. In: INCoS 2011. pp. 41 – 46. IEEE (2011)
9. Pasi, G.: Fuzzy sets in information retrieval: State of the art and research trends. In: Bustince, H., Herrera, F., Montero, J. (eds.) Fuzzy Sets and Their Extensions: Representation, Aggregation and Models, Studies in Fuzziness and Soft Computing, vol. 220, pp. 517–535. Springer Berlin / Heidelberg (2008)
10. Peng, L., Yang, B., Zhang, L., Chen, Y.: A parallel evolving algorithm for flexible neural tree. Parallel Computing 37(1011), 653 – 666 (2011)
11. Qi, F., Liu, X., Ma, Y.: Synthesis of neural tree models by improved breeder genetic programming. Neural Computing & Applications 21, 515–521 (2012)
12. Sedano, J., Curiel, L., Corchado, E., de la Cal, E., Villar, J.R.: A soft computing method for detecting lifetime building thermal insulation failures. Integr. Comput.-Aided Eng. 17(2), 103–115 (Apr 2010)
13. Snásel, V., Krömer, P., Platos, J., Abraham, A.: The evolution of fuzzy classifier for data mining with applications. In: SEAL 2010. Lecture Notes in Computer Science, vol. 6457, pp. 349–358. Springer (2010)
14. Vera, V., Corchado, E., Redondo, R., Sedano, J., Garcia, A.: Applying soft computing techniques to optimize a dental milling process. Neurocomputing Submitted
15. Zadeh, L.A.: Empirical Semantics, Quantitative Semantics, Vol. 12, vol. 1, chap. Test-score semantics dor natural languages and meaning representation via Pruf, pp. 281–349. Studienverlag Brockmeyer, Bochum (1981)