Tatami Grammatical Evolution for Feature Extraction with Multi Fitness Evaluation

Go Frendi Gunawan 1st Affiliation
Dept of Informatics, Faculty of Information Technology,
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
gofrendiasgard@gmail.com

Joko Lianto Buliali
Dept of Informatics, Faculty of Information Technology,
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
joko@its-sby.edu

Abstract

Feature Extraction is a significant topic in classification problem solving. Until now, there is no such a standard way to find the best features of a data. In this research, grammatical evolution with multiple fitness evaluation approach (named as *GE Tatami*) has been developed to extract best features of the data. The method generate *n*-1 features which are able to separate data hierarchically, with n is number of classes.

Some methods have been evaluated in this research, including genetics algorithm, grammatical evolution with global fitness measurement, grammatical evolution with multi fitness measurement, grammatical evolution with Tatami fitness measurement, and Gavrilis's grammatical evolution.

The experiment shows that Tatami method produces a better result compared to the four other methods for synthesis data using decision tree classifier. The synthesis data is hierarchically separable. However Tatami method shows a bad result when it failed to find ideal features or using SVM classifier.

***Index Terms* — Keywords: feature extraction, grammatical evolution, classification, multi-fitness.**

I. INTRODUCTION

Feature Extraction is a process to find transformation mapping of original features set into new features set that shall produce better class separation [5]. Feature extraction is a significant topic in classification problem, since good feature set may boost classification accuracy, while bad feature set tend to make classification accuracy worse.

Some genetics algorithm based method has been conducted to make good feature sets. In [3] and [4], grammatical evolution has been conducted for feature extraction purpose. Gavrilis and other researchers created a method that produces a set of new feature with classifier's accuracy as fitness function. In [5] and [6], similar method has also been used for different cases. However sometime, several irrelevant features are generated and attached to the feature set.

In [1], a new approach by measure per-feature accuracy as a goodness measurement has been created. However, this approach tends to produce bad accuracy since it only use one feature to separate all classes.

In this paper, we propose a multi-fitness evaluation to hierarchically separate the data. This multi-fitness evaluation is named as Tatami Grammatical Evolution (GE Tatami) since it shows a figure looks like Japanese traditional *tatami*. Several earlier methods also used as comparison.

II. DATA AND FEATURE SPACE

In classification problem, a data usually consists of several records, columns (features) and classes. The data in table 1.1, cosists of two features, (x and y) and 3 classes (A, B and C).

Since the data consists of 2 original features, it is possible to plot it in *Cartesian Diagram* to produce better visualization. Figure 1.1 shows how the data plotted.

As shown in the figure 1.1, it is impossible to separate the data by using just one of the original features. The two features should be used altogether, producing a feature space to make data separation possible.

Table 2.1 Example of Numeric Data

Original Features		Class
x	y	
0.3	0.5	A
0.4	0.9	B
0.6	0.2	A
0.9	1.0	C
1.0	0.3	C
0.8	0.2	B
...	...	...

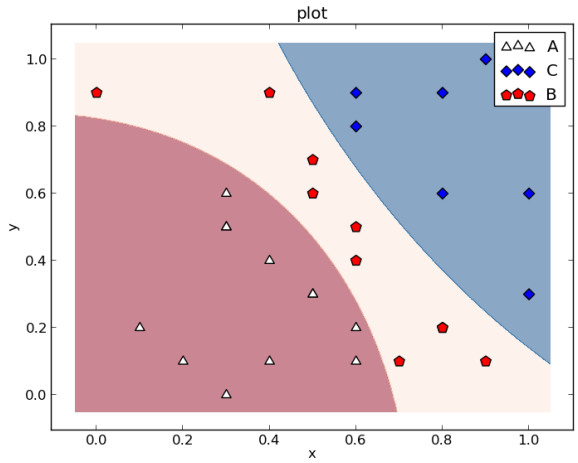

Figure 2.1 Feature Space Produced by Original Features and Classification Process by Using RBF SVM

A commonly used classifier such a SVM or neural network can use the feature space to separate the data very well. However the separation cannot be done linearly. The separator lines are a bit curvy. Those curvy lines are mathematically more complex than the linear ones.

III. Feature Extraction

Feature Extraction is basically a mapping from original feature space to another one. The mapping can produce higher dimension, or lower dimension.

In classification problem, it is intuitive to map any inseparable (or difficultly separated) data into higher dimension. Higher dimension tend to produce more possibility to separate the data. However, this will also lead to more complex calculation as well.

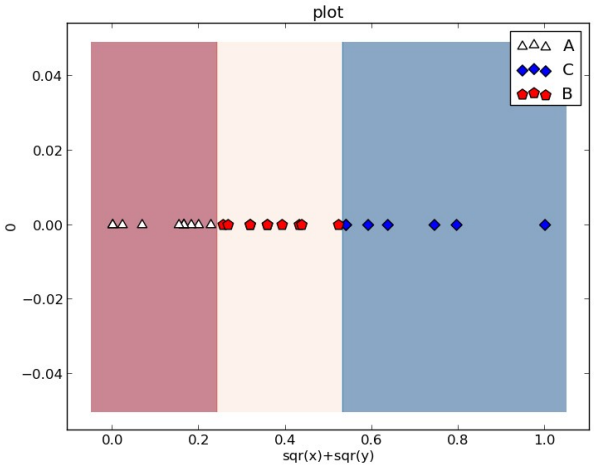

Figure 3.1 Feature Space Produced by New Feature (sqr(x)+sqr(y))

The goal of feature extraction is to produce a feature set that consist of as few features as possible. T he generated feature set should also able to produce good data separation in relatively simple way (i.e.: such as using linear lines).

In order to explain the purpose of feature extraction, one can transform the data on table 2.1 which consists of 2 into features (x and y) into a new feature space containing just 1 feature, sqr(x)+sqr(y). The feature space is shown as in figure 3.1

This transformation allows us to separate data by using a linear line, which mathematically is simpler than the previous one.

IV. Grammatical Evolution

Grammatical Evolution is an evolutionary algorithm based on genetics algorithm. This method takes empowered by a predefined context-free grammar to transform an individual into anything allowed by the grammar.

Each individual consists of a binary (or integer) string called genotype. The genotype is then transformed by using the grammar into a phenotype. The phenotype can be a mathematics operation, a function, or even a whole computer program, depend on the grammar used.

For example, we use a grammar as defined in table 4.1.
Let's say an individual's genotype consists of an integer string 220, 203, 51, 123, 2, 45. The process started by taking <*expr*> as start symbol. <*expr*> has 4 possible evolution rule, therefore, we take the first segment of the integer string (in this case 220), and do modulo operation. Since 220 mod 4 produce 0, we then take the 0^{th} rule. This make <*expr*> evolved into <expr><op><expr>. Next, we go with the second segment of integer string (which is 203) and choose the first non-terminal of ***<expr>***<*op*><*expr*> (which is <*expr*>). Again, we do a module operation. Since <expr> has 4 possible evolution rule, and 203 mod 4 produce 3, we take the 3^{rd} rule (which is <*var*>). Now we have <*var*><*op*><*expr*>. The process continued until we get x*x as phenotype. Figure 4.2 shows the complete transformation process.

After getting the phenotype, the process then continued as those in genetics algorithm. The phenotype's fitness measured by certain fitness function, producing a fitness value. The fitness value of each individual then used to determine individual's survival rate.

For feature extraction problem, classifier accuracy usually used as fitness value. Thus, each individual's fitness value shall ranged between 0 and 1.

As in genetics algorithm, after several generation, the best individual will be found. This best individual is then used as problem solution.

Table 4.1 Grammar Example

Node	Production Rule	Index
<expr>	<expr><op><expr>	0
	(<expr><op><expr>)	1
	<pre-op><expr>	2
	<var>	3
<op>	+	0
	-	1
	*	2
	/	3
<pre-op>	Sin	0
	Cos	1
	Tan	2
var	x	0

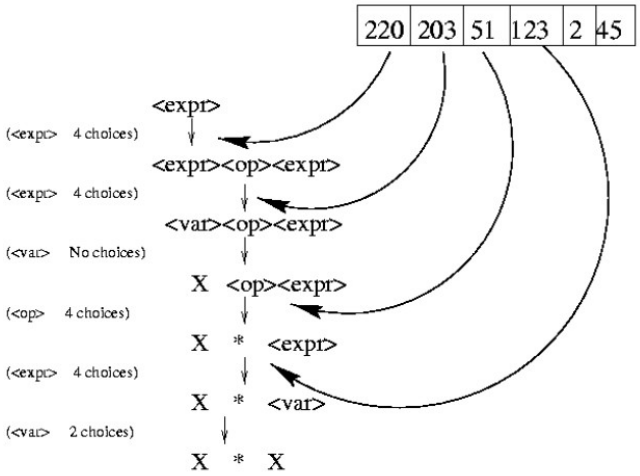

Figure 4.1 Transformation Process

V. GE MULTI

Suppose there are *n* classes in the data, it is logical to think that one can has *n* features to separate data based on it's classes. Each features should be able to separate a certain class from others.

To achieve this, a simple grammatical evolution with single fitness value for every individual will not work. Therefore, we have develop a modified version of grammatical evolution which produce several fitness value for every individual. This modified version is named as GE Multi.

Let's say one has a data with *n* classes: $\{c_1, c_2, c_3, \ldots, c_n\}$. In GE Multi, each individual has *n* fitness values: $\{f_1, f_2, f_3, \ldots, f_n\}$. The first fitness value f_1 represent the goodness of individual's phenotype to separate c_1 and the others ($\{c_2, c_3, \ldots, c_n\}$). The second fitness value f_2 represent the goodness of individual's phenotype to separate c_2 and the others ($\{c_1, c_3, \ldots, c_n\}$). The *n*th fitness value f_n represent the goodness of individual's phenotype to separate c_n and the others ($\{c_1, c_2, c_3, \ldots, c_{n-1}\}$).

To calculate each fitness value of each individual, the data should be transformed so that it only has 2 classes. To measure the first fitness value, $\{c_2, c_3, \ldots, c_n\}$ should be merged into one class, let's say $c_{\sim 1}$. The classifier then work to separate c_1 and $c_{\sim 1}$. The classifier's accuracy then used as fitness value f_1.

Using this approach, there will be *n* or less best individual. Each represent best fitness value for each classes (i.e : There will be individual with best f_1 value, There will be individual with best f_2 value, and so on). There is also possibility, an individual has best value for two or more fitness value. Therefore, using GE Multi as feature extractor, one will get numerous features ranged from 0 to *n*, where *n* is number of classes in the data.

VI. GE TATAMI

GE Tatami is an improvement of GE Multi which is designed for hierarchically separable data. For hierarchically separable data, the separation can also be done hierarchically. For example, if a data has *n* classes, $\{c_1, c_2, c_3, \ldots, c_n\}$. Once c_1 separated from the rest, c_1 can be ignored, so that in the next iteration, only $\{c_2, c_3, \ldots, c_n\}$ involved.

GE Tatami consists of *n*-1 iteration where *n*=number of classes. At each iteration GE Multi is used to determine best individuals. Maximum fitness values of all best individuals is then selected to determine which class is best separated from the others. The individual corresponded to the best separated class is then selected and added to the new features list. The process then repeated with the best separated class omitted. On the last iteration, there will be only 2 classes left. The process ended after best individual to separate the 2 classes added to the new features list. Figure 6.1 shows the global flowchart of GE Tatami.

Using this approach, there will be maximum of *n*-1 features. Figure 6.2 shows how 2 features produced by GE Tatami can separate 3 classes very well.

GE Tatami is sparingly suitable for decision tree since both are designed to separate hierarchically separable data. The decision tree produced by using features generated by GE Tatami is shown in figure 6.3.

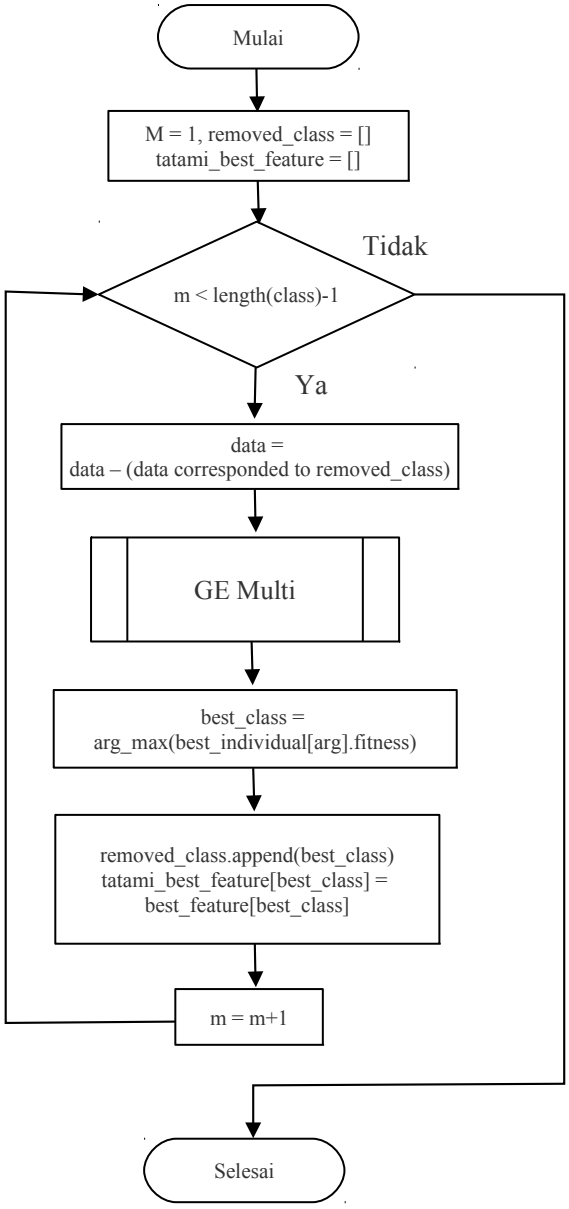

Figure 6.1 Flowchart of GE Tatami

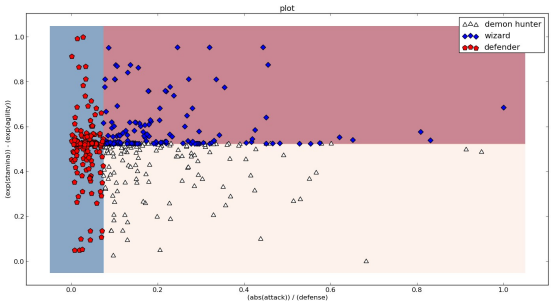

Figure 6.2 Feature Space Produced by GE Tatami

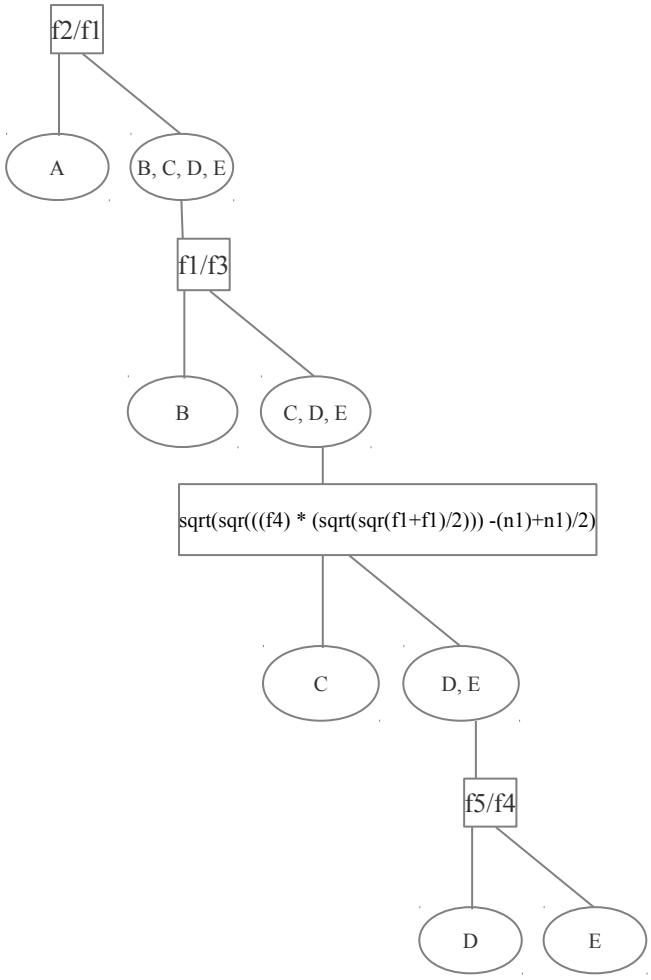

Figure 6.3 Decision Tree Using GE Tatami's Generated Feature

VII. Result And Analysis

To compare effectiveness of GE Multi and GE Tatami with earlier methods, we have conduct several experiments involving several datasets. The datasets consist of 3 synthesis data which is hierarchically separable and 3 commonly used datasets taken from UCI Machine Learning website (iris, balanced scale and E-Coli). In synthesis 03, the original features is hidden. The hidden features can be retrieved by mathematically fiddling with shown features.

For each dataset, we perform 5 fold cross-validation and another test which all data used as training set and testing set. The complete result of experiments is available freely at https://github.com/goFrendiAsgard/feature-extractor/

The results are then resumed by calculating average of each case accuracy, producing table 7.1. By looking at the table, one can say that GE Tatami shows a very good result on hierarchically separable data (synthesis 01, synthesis 02, and synthesis 03). However, GE Multi generally shows better

accuracy in not only synthesis data, but also on other commonly used ones.

The results also show that GE Tatami prevail on synthesis 02 and synthesis 03. This is happened since the classes on those datasets can be hierarchically separated, and GE Tatami succeed to provide ideal feature space (as shown in figure 6.2 and figure 6.3). On the other hand, GE Tatami didn't show such a prevailing result on synthesis 01 due to it only has 3 classes. The hierarchical structure of 3 classes seems to not being significant in the process.

Figure 7.1 Experiment Result Average

Experiment		GA Select Feature		GE Global		GE Multi		GE Tatami		GE Gavrilis	
		Accuracy (%)	**Features**	**Accuracy (%)**	**Features**	**Accuracy (%)**	**Features**	**Accuracy (%)**	**Features**	**Accuracy (%)**	**Features**
Iris	**Train**	96.2	2	98.9	1	98.8	3	98.9	2	**99.0**	1
	Test	93.7		92.5		94.7		92.5		**97.5**	
	Total	95.7		97.6		98.0		97.6		**98.7**	
B. Scale	**Train**	71.3	3	88.6	1	**96.2**	1	93.3	2	82.7	25
	Test	69.8		79.0		**81.6**		81.5		75.4	
	Total	71.0		86.7		**93.3**		91.0		81.2	
E. Coli	**Train**	97.0	5	86.9	1	**98.2**	8	97.2	7	98.0	15
	Test	**78.2**		65.1		63.5		50.6		67.2	
	Total	**93.4**		82.7		91.5		88.2		92.0	
Synthesis 01	**Train**	73.9	3	78.8	1	99.8	3	**100**	2	87.9	20
	Test	71.8		42.4		**84.8**		82.2		80.7	
	Total	73.5		71.7		**96.9**		96.5		86.5	
Synthesis 02	**Train**	78.3	4	71.4	1	99.8	3	**100**	3	89.4	12
	Test	74.4		39.7		73.0		**79.3**		78.0	
	Total	77.5		65.2		94.5		**95.9**		87.2	
Synthesis 03	**Train**	72.7	5	67.8	1	99.1	5	**100**	4	81.9	19
	Test	40.5		36.9		63.0		**77.6**		50.8	
	Total	66.3		61.7		92.0		**95.5**		75.8	
AVG	**Train**	81.6	3	82.1	1	**98.6**	3	98.2	3	89.8	15
	Test	71.4		59.3		76.8		**77.3**		74.9	
	Total	79.6		77.6		**94.4**		94.1		86.9	

VIII. CONCLUSION

The experiment conducted shows that using decision tree as classifier, GE Tatami shows a very good performance on hierarchically separated synthesis data. However, using the same classifier GE Multi shows generally good performance.

This also led us to conclude that combining features generated by GE Tatami and GE Multi is possibly able to boost accuracy.

In the experiment conducted, classifier's accuracy is used to determine fitness value of each individual. Finding a better mechanism to measure fitness value will also probably make the calculation faster.

ACKNOWLEDGMENT

Few has help the main author to finish this research. Not to mention, Mr. Joko Lianto, as supervisor. His suggestions inspire the main author to solve several problems and finish this research on time.

Some friends, such a Mr. Mukhlis Amien and Mr. Hendra Suprayogi also help the main authors by providing several out-of-the-box suggestions that sometimes useful and never thought before.

REFERENCES

[1] Gunawan G. F., Gosaria S, Arifin A. Z. (2012). “Grammatical Evolution For Feature Extraction In Local Thresholding Problem”, Jurnal Ilmu Komputer dan Informasi, Vol 5, No 2 (2012)

[2] Harper R., Blair A. (2006). “Dynamically Define Functions in Grammatical Evolution”, IEEE Congress of Evolutionary Computation, July 16-21, 2006

[3] Gavrilis D., Tsoulous I. G., Georgoulas G., Glavas E. (2005). "Classification of Fetal Heart Rate Using Grammatical Evolution", IEEE Workshop on Signal Processing Systems Design and Implementation, 2005.

[4] Gavrilis D., Tsoulous I. G., Dermatas E. (2008). "Selecting and Constructing Features Using Grammatical Evolution", Journal Pattern Recognition Letters Volume 29 Issue 9, July, 2008 Pages 1358-1365 .

[5] Guo L., Rivero D., Dorado J., Munteanu C. R., Pazos A. (2011). "Automatic feature extraction using genetic programming: An application to epileptic EEG classification ", Expert Systems with Applications 38 Pages 10425-10436

[6] Li B., Zhang P.Y., Tian H., Mi S.S., Liu D.S., Ruo G.Q. (2011). "A new feature extraction and selection scheme for hybrid fault diagnosis of gearbox", Expert Systems with Applications 38 Pages 10000-10009

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V. , Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P. , Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research Vol. 12 Pages 2825-2830 Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 1955. (*references*)