The Use of Probabillistic Neural Network and ID3 Algorithm for Java Character Recognition

by Gregorius Satia Budhi

The Use of Probabilistic Neural Network and ID3 Algorithm for Java Character Recognition

Gregorius Satia Budhi
Informatics Department
Petra Christian University
Siwalankerto 121-131
Surabaya, Indonesia
greg@petra.ac.id

Rudy Adipranata
Informatics Department
Petra Christian University
Siwalankerto 121-131
Surabaya, Indonesia
rudya@petra.ac.id

Bondan Sebastian
Informatics Department
Petra Christian University
Siwalankerto 121-131
Surabaya, Indonesia
bondansebastian@gmail.com

Liliana
Informatics Department
Petra Christian University
Siwalankerto 121-131
Surabaya, Indonesia
lilian@petra.ac.id

***Abstract*— Java character is the character that is used especially in Java island, Indonesia. Java character has a special form of writing consist of basis characters, vowels, complementary, etc. In this research, the authors conducted Java character recognition using 2 methods, namely probabilistic neural network (PNN) and ID3 algorithm. PNN is a method of artificial neural network that can be trained supervised and unsupervised. PNN is built based on the theory of probability which is realized as an artificial neural network. This method is used because PNN has a high accuracy in the classification of data, also has high speed when performing the process. ID3 (Iterative Dichotomiser) is a decision tree algorithm. Basic ID3 algorithm using tree induction that gives attribute to the node in the tree based on how much information increases from the node. From experimental results, PNN method can achieve an accuracy up to 92.35% for data that has been trained previously, and up to 61.08 % for data hasn't been trained before. While ID3 can achieve recognition rate of 100% for data has been trained before but only 15.57% for data hasn't been trained before.**

***Keywords*— Java character recognition, probabilistic neural network, ID3.**

I. Introduction

Java character is the character that is used especially in Java island, Indonesia. Java character has a special form of writing consist of basis characters, vowels, complementary, etc. In this research, the authors conducted Java character recognition using 2 methods, namely probabilistic neural network (PNN) [1] and ID3 algorithm [2].

The process of Java character recognition can be divided into three stages: segmenting the image of the Java character document, extracting features of each Java character, and doing recognition of Java character from the features that have been extracted [3]. The image segmentation is the process of cutting Java character document image into pieces, each of which contains only one Java character.

Features extraction is the process of identifying the features of each Java character, where the features can be the outline or shape of the character. Features of Java character can be curve, straight lines, or loop [4]. Features extraction process will identify the edge of curvature, a straight line, a loop, or a number of pixels in small area which is owned by a character. After the features of Java character obtained, then the features will be processed, so that the computer can recognize the character. For recognition process, we use and compare between PNN and ID3 algorithm.

II. Java Character

Java character consist of basis character called *Carakan*, vowel and complimentary called *Sandhangan*, number called *Wilangan*, consonant called *Pasangan*, etc. Basis character of Java character consist of 20 characters, can be seen in Fig. 1 [5].

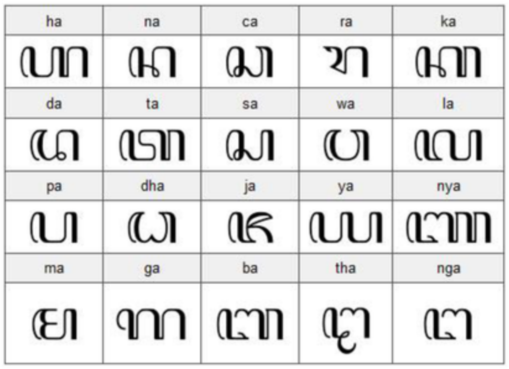
ha
na
ca
ra
ka
da
ta
sa
wa
la
pa
dha
ja
ya
nya
ma
ga
ba
tha
nga

Fig. 1. Basis Character

III. Probabilistic Neural Network

PNN is a method of artificial neural network that can be trained supervised and unsupervised. PNN is built based on the theory of probability which is realized as an artificial neural network [1, 6]. This method is used because PNN has a high accuracy in the classification of data, also has high speed when performing the process. PNN Architecture can be seen in Fig. 2 [1].

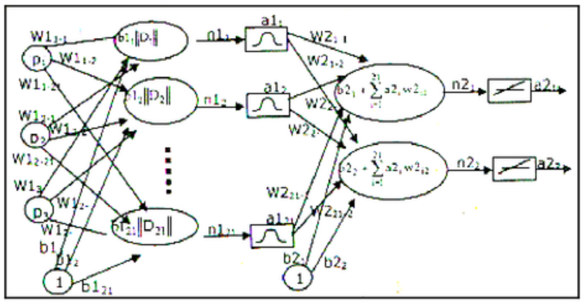

Fig. 2. PNN Architecture

Here is the training algorithm (points 1-3) and classification algorithm (points 4-7) of PNN [1, 6]:

1. Perform initialization of the initial weight *W* of the matrix *Q x R* where *R* is the dimension of input and *Q* is the amount of training data.
2. Perform initialization of bias value (*b*) of the spread value entered using (1).

$$b = \sqrt{(-\ln 0.5)}\,/s \tag{1}$$

3. Perform initialization of final weight matrix *M* which is a *K x Q* where *K* is the number of groups of the classification result and *Q* is the amount of training data. *i-th* row of the matrix *M* representing the the *i-th* training data and *j-th* column matrix values will be valued 1 if the training data entry into the group, otherwise it will be valued 0.
4. Calculate the distance between the input data vector *P* by vectors in each row on the initial weight *W* (euclidean distance between the vector *P* with the vector W_i) resulting in vector distance || *W* - *P* || in *R* dimension.
5. Calculate the activation value of the distance between the initial weight and the input data (vector *a*), using *radbas* function () in (2), (3), (4).

$$\text{radbas(n)} = e^{-n^2} \tag{2}$$

$$n = \|W - P\| \,.\!*\, b \tag{3}$$

$$b = \frac{\sqrt{-\ln 0.5}}{s} \tag{4}$$

6. Multiplying vector *a* by matrix *M* so as to produce the output vector *d*.
7. Looking output of PNN with competitive function *C* where this competitive function will seek the greatest value in the vector *d*. The index of the largest value will indicates the classification result of input data *P*.

IV. DECISION TREE

Decision tree induction is a decision tree learning from class-labeled training tuples. Decision tree shaped like a flowchart in which each internal node represents a test on an attribute, each branch illustrates the results of the tests, and each leaf node indicates the class label. Top node is the root node [2]. Example of a decision tree can be seen in Fig. 3 [2].

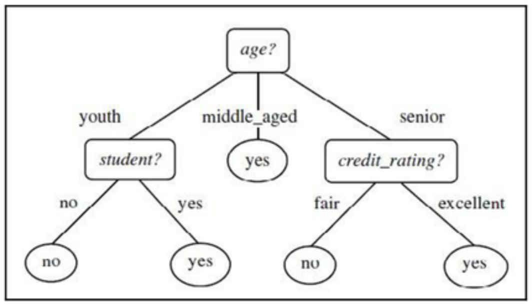

Fig. 3. Decision Tree

Fig. 3 shows the prediction of whether a customer in AllElectronics will buy a computer or not. Internal node described by the box, and the leaf node depicted with oval.

During the making tree, should choose the most excellent attributes in differentiating classes. The algorithms used for decision tree is ID3.

V. ID3

ID3 (Iterative Dichotomiser) is a decision tree algorithm created by J. Ross Quinlan [2]. Basic ID3 algorithm using tree induction that gives attribute to the node in the tree based on how much information increases from the node. ID3 method allows an attribute to have two or more values in a node or split point [7]. Selected attributes for each node is an attribute that maximize information gain. ID3 can classify large amounts of data in a relatively fast, depending on how large the data set used [2].

ID3 use the Information gain for the selection of attributes. Information attribute that has the highest gain was elected splitting attribute for a node. Information that is expected to be required to classify a tuple in D is given by (5) [2]:

$$\text{Info(D)} = -\sum_{i=1}^{m} p_i \log 2\,(p_1) \tag{5}$$

where p_i is the possibility of an arbitrary tuple in *D* belongs to the class C_i and is estimated by $|Ci_{,D}|/|D|$. Log function with base 2 is used as information calculated in bits. *Info(D)* is the average amount of information used to identify the class label of a tuple in *D*. *Info(D)* is known as entropy of *D*. The amount of information is still needed to achieve the desired partition measured by (6) [2]:

$$info_A(D) = a_0 + \sum_{j-l}^{v} \frac{|D_j|}{|D|} Info(D_j) \tag{6}$$

Term $\frac{|Dj|}{|D|}$ functions as the weight of the *j* partition. *InfoA(D)* is estimate of the information needed to classify a tuple of *D* based partitioning by A. The smaller expected information needed, the greater degree of partition purity. Information gain is defined as the difference between the information needs with the needs of the new origin, as calculated using (7) [2].

$$\text{Gain(A)} = \text{Info(D)} - \text{InfoA(D)} \tag{7}$$

In other words, *Gain(A)* tell how much will grow by branching in A. Attribute A with the greatest information gain (*Gain (A)*) will be selected as the splitting attribute at the node.

VI. SYSTEM DESIGN

Java character recognition system can be divided into two main processes, namely training and classification. Process flow is described using flowchart which can be seen in Fig. 4.

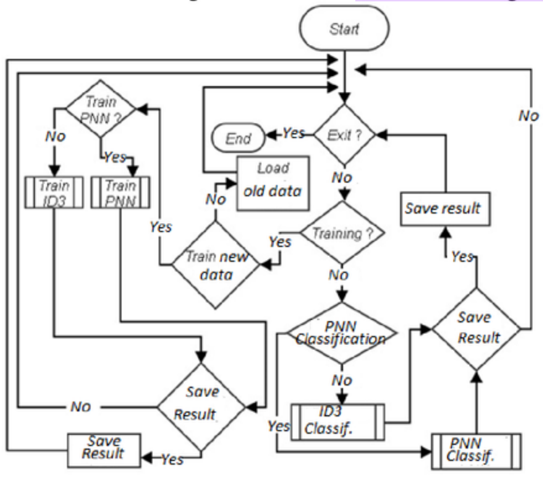

Fig. 4. System Flowchart

Training of PNN method is just the weight 0 initialization process, last weight, and the bias value. Flowchart of PNN training can be seen in Fig. 5.

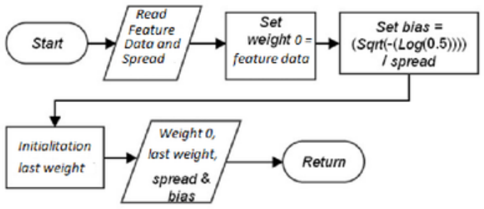
Start
Read Feature Data and Spread
Set weight 0 = feature data
Set bias = (Sqrt(-(Log(0.5)))) / spread
Initialitation last weight
Weight 0, last weight, spread & bias
Return

Fig. 5. PNN Training Flowchart

In Fig. 5, it can be seen that before initialization of variables in PNN done, first read the data features and the value of the spread. Both feature data and the spread value obtained from user input. After the input is read, the data features incorporated into the initial weight (weight 0). Then the value of bias will be calculated based on spread value. The last weight is initialized and then the resulting data sets are ready to be saved to an external file. The data set consists of the initial weight (weight 0), last weight, spread values and bias that have been initialized earlier.

The process of training the decision tree is basically a rule-making tree with specific algorithms to obtain an effective rule for classifying data. In this system, the algorithm used is Iterative Dichotomiser 3 (ID3). In ID3, rule-making is based on information gain contained in an instance/node tree, which in Javanese character recognition system, each node is the type/class of a Java character. Flowchart of the rule-making process tree using ID3 algorithm can be seen in Fig. 6.

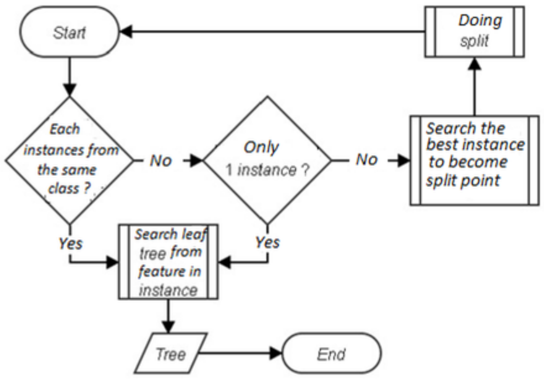
Start
Doing split
Each instances from the same class ?
No
Only 1 instance ?
No
Search the best instance to become split point
Yes
Search leaf tree from feature in instance
Yes
Tree
End

Fig. 6. ID3 Training

The process of establishing rule tree or decision tree occur recursively. This function will stop when the leaf of a tree has only one instance alone, or when the leaf of the tree has multiple instances of the same class. If the condition for stopping has not been met, then the best instance of the search function will be executed. Best instance found will be used as the split point or a branching point in the tree.

The overview process of PNN classification method can be described as follows: PNN that have been trained before will receive input in the form of matrix X. Matrix X is a collection of input vector x, where x is the set of features of a unknown Java character. Furthermore, every vector x will be calculated the distance with each vector in the matrix W (initial weight). This process will produce a distance vector, which then will go through the activation process (radial basis function). The activation process will produce activated distance vector, where the vector is to be incorporated into the competitive function to generate a vector representation index Java character. This vector will be used to search for Java character that matches the index of the pattern unit. After all vectors x in the matrix X are processed, the results matrix will be prepared to be sent to the next process.

ID3 classification overview are the nodes searching in the tree which class is in accordance with the input. Input is in the form of matrix M that every line is a vector m that stores the features of a unknown types/class Java character. Each vector in the matrix M will be read, and then each feature information in it will be matched with each node in the tree until a leaf of the tree is found. Leaf is what would be considered a class of a set of features in the vector m.

VII. EXPERIMENTAL RESULT

In experiment, we used 10 images of Java character documents. Each image has from 25 until 283 Java characters. Experiment is done by using the data that has been previously trained and with the data has never been trained before. The results of recognition will be compared between the use of the PNN method and ID3 method.

Experiment result of PNN method using data that has been trained before can be seen in Table 1. Spread value used in this experiment is 1.

TABLE I RECOGNITION RESULT OF PNN USING DATA TRAINED BEFORE

No.	Document	Number of Java Character	Accuracy (%)
1	Document 1	136	99.26
2	Document 2	55	92.73
3	Document 3	177	89.27
4	Document 4	212	94.81
5	Document 5	283	89.75
6	Document 6	198	93.94
7	Document 7	196	94.90
8	Document 8	25	100
9	Document 9	130	86.15
10	Document 10	156	89.74
Average			**92.35**

Based on experimental results in Table 1, it can be seen that the PNN can achieve an average accuracy rate of 92.35% in classifying the data that has been trained before. In next experiment, PNN will be used to classify the data that has never been trained before. Documents used in the experiment remains the same with the previous experiment, but not all of the data in an image is used for training. Only a portion of the data will be used for training, and the rest tried to be classified. Of each characters contained in the image, limited to a maximum two types of the same typeface that may be used as training data, the rest will be used as a data classification. The experimental results are shown in Table 2.

TABLE II RECOGNITION RESULT OF PNN USING DATA NEVER BEEN TRAINED BEFORE

No.	Document	Number of Java Character for Training	Number of Java Character for Classification	Accuracy (%)
1	Document 1	57	79	54.43
2	Document 2	28	27	55.56
3	Document 3	51	126	64.29
4	Document 4	59	153	66.01
5	Document 5	61	222	56.76
6	Document 6	56	142	62.68
7	Document 7	60	136	63.24
8	Document 8	15	10	60.00
9	Document 9	43	87	75.86
10	Document 10	46	110	49.09
Average				**61.08**

From the experiment result in Table 2, the average accuracy rate is 61.08% for data has never been trained before.

The next experiment is done using ID3 method with data that has been trained before. Recognition accuracy results can be seen in Table 3.

TABLE III EXPERIMENTAL RESULT OF ID3 USING DATA TRAINED BEFORE

No.	Document	Number of Java Character	Accuracy (%)
1	Document 1	136	100
2	Document 2	55	100
3	Document 3	177	100
4	Document 4	212	100
5	Document 5	283	100
6	Document 6	198	100
7	Document 7	196	100
8	Document 8	25	100
9	Document 9	130	100
10	Document 10	156	100
Average			**100**

For the results of the experiment using data that has never been trained before can be seen in Table 4

TABLE IV EXPERIMENTAL RESULT OF ID3 USING DATA NEVER BEEN TRAINED BEFORE

No.	Document	Number of Java Character for Training	Number of Java Character for Classification	Accuracy (%)
1	Document 1	57	79	8.86
2	Document 2	28	27	29.63
3	Document 3	51	126	19.84
4	Document 4	59	153	20.26
5	Document 5	61	222	16.67
6	Document 6	56	142	8.45
7	Document 7	60	136	12.50
8	Document 8	15	10	20.00
9	Document 9	43	87	27.59
10	Document 10	46	110	6.36
Average				**15.57**

The recognition results of ID3 by using the data that has been trained before reached 100% as shown in Table 3. It is better than PNN accuracy rate, but for data that has not been trained, the results of the recognition only 15.57% as shown in Table 4.

VIII. CONCLUSIONS

From experimental results, PNN method can achieve an accuracy rate up to 92.35% for data that has been trained previously, and up to 61.08 % for data hasn't been trained before. While ID3 can achieve recognition rate of 100% for data has been trained before but only 15.57% for data hasn't been trained before. So we can conclude that PNN method is more suitable for use in Java character recognition than ID3 algorithm.

ACKNOWLEDGMENT

We thank to PT Coordination of Private Higher Education Region VII, East Java, Indonesia for funding this research by Research Competitive Grant DIPA-PT Coordination of

Private Higher Education Region VII, East Java, Indonesia fiscal year 2015 no. 20/SP2H/PDSTRL_PEN/LPPM-UKP/IV/2015 entitled (in Bahasa Indonesia) "*Aplikasi Pengenalan Aksara Jawa Guna Digitalisasi Dokumen Beraksara Jawa Untuk Mendukung Pelestarian Budaya Nasional*".

REFERENCES

[1] Specht, D. F. Probabilistic Neural Networks. Neural Networks, vol. 3, pp. 109-118, UK: Elsevier, 1990

[2] Han, J., Kamber, M., Pei, J. Data mining: Concepts and techniques (3nd ed.). San Fransisco, CA: Morgan Kaufmann, 2012

[3] Budhi, Gregorius Satia, Rudy Adipranata. Handwritten Javanese Character Recognition Using Several Artificial Neural Network Methods. Journal of ICT Research and Applications, Vol. 8, No. 3, pp. 195-212, 2015

[4] Adipranata, Rudy, Liliana, Meiliana I, Gregorius S.B. Feature Extraction for Java Recognition. Proc. Of 4th International Conference on Soft Computing, Intelligent System and Information Technology. Bali, Indonesia, 2015

[5] Nusantara Ranger. (2014) Aksara Jawa Hanacaraka. Elang Book chapter 4. [Online]. Available: http://nusantaranger.com/referensi/buku-elang/chapter-4merah/aksara-jawa-hanacaraka/

[6] Budhi, Gregorius S., Tok Fenny H. dan Rudy Adipranata. Leaf Recognition Application for Plant Classfication Using Probabilistic Neural Network Method. National Seminar KOMMIT, Gunadarma, Jakarta, 2008

[7] Berry, Michael W., & Browney, Murray. Lectures note in data mining. USA, World Scientific Publishing Co. Pte. Ltd, 2006.

The Use of Probabillistic Neural Network and ID3 Algorithm for Java Character Recognition

ORIGINALITY REPORT

17%	11%	9%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.itmsoc.org Internet Source	5%
2	kursor.trunojoyo.ac.id Internet Source	4%
3	Communications in Computer and Information Science, 2015. Publication	2%
4	E. Sivasankar, R.S. Rajesh. "Design and development of a clinical decision support system for diagnosing appendicitis", 2012 Computing, Communications and Applications Conference, 2012 Publication	2%
5	Imas Sukaesih Sitanggang. "Classification model for hotspot occurrences using a decision tree method", Geomatics Natural Hazards and Risk, 06/2011 Publication	2%
	fportfolio.petra.ac.id	

6	Internet Source	2%
7	www.scribd.com Internet Source	1%
8	Dash, Ch. Sanjeev Kumar, Satchidananda Dehuri, Sung-Bae Cho, and Gi-Nam Wang. "Towards Crafting a Smooth and Accurate Functional Link Artificial Neural Networks Based on Differential Evolution and Feature Selection for Noisy Database", International Journal of Computational Intelligence Systems, 2015. Publication	1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		