INFORMATION EXTRACTION OF SCIENTIFIC PAPERS WITH LEARNING VECTOR QUANTIZATION ALGORITHM

Firdamdam Sasmita[1], Ken Kinanti Purnamasari[2]

[1,2] Universitas Komputer Indonesia
Jl. Dipati Ukur No.112-116, Lebakgede, Coblong, Kota Bandung, Jawa Barat 40132
E-mail : firdamdamsasmita2211@gmail.com[1], ken.kinanti@email.unikom.ac.id[2]

ABSTRACT

Detecting every component of a scientific paperwork document is difficult to do in its various formats. Previous research related to the introduction of scientific writing documents has been carried out for Indonesian scientific thesis writing documents. Based on testing in 50 documents in 2013 and 2017, the average accuracy was 57%. These problems can be overcome if using machine learning. The machine learning algorithm used in this study is LVQ. In previous studies in the field of text classification and image processing, LVQ had superior accuracy compared to SVM, KNN, FFBP, RBF, Naive Bayes, and C4.5. The LVQ algorithm has the possibility of obtaining better accuracy when done in the field of information extraction. Based on testing of 40 thesis scientific papers from 2011 to 2018, the average token-class accuracy was 78%, class-token accuracy was 19%. The acquisition of token-class accuracy is caused by the use of the weighting feature. Meanwhile, the acquisition of class-token accuracy is caused by the influence of algorithms, misspellings, and the appearance of irregular symbols.

Keywords: Information Extraction, Learning Vector Quantization, LVQ, Scientific paperwork, Thesis.

1. INTRODUCTION

Information extraction is a process to change the unstructured information contained in the text into structured data [1]. Information extraction is done by detecting the components in each document, one of which is a scientific writing document. Detecting each component in a scientific paperwork document is difficult for documents with a variety of formats because it requires an extraction rule that is appropriate for each document.

Previous studies related to the introduction of scientific writing documents have been conducted for Indonesian language thesis documents [2]. The research extracts information using a rule-based system to get the identity of the thesis document, including a cover, abstract, and abstract. Based on the testing of 3 documents in 2017, an accuracy of 100% was obtained. Whereas, if the tests were carried out on 50 documents included with 2013, the average accuracy was 57%. The decrease in accuracy that occurs is caused by a rule-based system that is unable to handle documents with various formats in 2013. Therefore, these problems can be overcome if using machine learning.

In previous studies, the use of machine learning in the field of information extraction has been carried out in scientific papers [3]. Likewise in other fields, such as text classification [4] with the comparison of SVM (Support Vector Machine) algorithm, LVQ (Learning Vector Quantization), and KNN (K-Nearest Neighbour). The study states that the algorithm that is superior is the LVQ algorithm with an accuracy of 80%. Similar to the research conducted by Novizon and Zulkarnain [5] in image processing by comparing the FFBP (Feed-Forward Back-Propagation) algorithm, RBF (Radial Basis Function), and LVQ, the LVQ algorithm produces an accuracy of 100%. Whereas in the research conducted by Umer and Khiyal [6] in the field of text classification with the comparison of LVQ, RBF, KNN, Naive Bayes, and C4.5 algorithms, the LVQ algorithm produces an accuracy of 84%. Therefore, the LVQ algorithm has the possibility to obtain better accuracy when done in the information extraction field.

Thus, this study will build an information extraction system using the LVQ algorithm for Indonesian-language scientific writing documents, with the limitations of the documents that will be used in this research is the thesis scientific documents of Informatics Engineering Study Program, Faculty of Engineering and Computer Science, Indonesian Computer University.

2. RESEARCH CONTENT

In this research, describe the research method, thesis scientific paperwork, system architecture, tokenization, data segmentation, feature extraction, LVQ algorithm, plotting per class, and test results.

2.1 Research Method

In this research, there are four stages of research workflow, including literature study, data collection, software development, and testing. The following blocks of research workflow diagram.

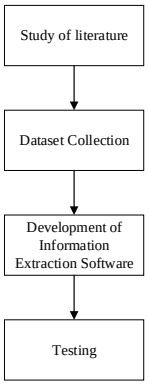


Figure 1. Research Workflow Block Diagram

2.2 Thesis Scientific Writing Document

In this research, the scientific paper thesis document of Informatics Engineering Study Program, Faculty of Engineering and Informatics, Indonesian Computer University is used as a dataset that has several categories on the cover and abstract sheets. The number of datasets used is 80 data, 40 documents from 2011 to 2018 and 40 documents from 2011 to 2018. The following sections of the categories on the cover sheet and abstract can be seen in Table 1.

Table 1. Categories of Cover Sheets and Abstracts

Sheet			
No.	**Cover**	**No.**	**Abstract**
1	Judul Penelitian (Sampul)	10	Judul Halaman Abstrak
2	Jenis Penelitian	11	Judul Penelitian (Abstrak)
3	Kalimat Pengajuan	12	Other
4	Penulis (Sampul)	13	Penulis (Abstrak)
5	NIM (Sampul)	14	NIM (Abstrak)
6	Program Studi	15	Isi Abstrak
7	Fakultas	16	Kata kunci
8	Universitas		
9	Tahun		

Several categories in Table 1 are represented as numeric classes for LVQ algorithm learning and classification processes. The total class that will be used in this study is 16 classes, calculated from the number on the cover sheet to the abstract sheet.

2.3 System Architecture

The construction of information extraction systems consists of several processes. The main process contained in the information extraction system using the LVQ algorithm consists of input training and testing data, preprocessing training and testing data, LVQ training, LVQ testing, and Plotting Per Class. For some other processes, it can be seen in the following block diagram system architecture.

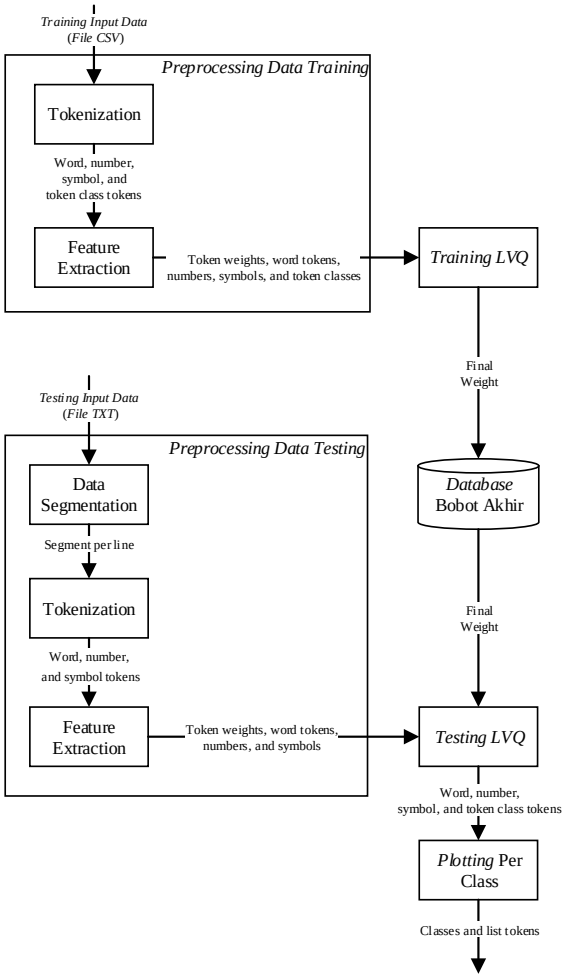


Figure 2. System Architecture Block Diagram

In the preprocessing data training phase, the input data to be used is a CSV file. The file contains a set of data from the labeling (plotting) of several categories contained in the PDF file the results of scanning the cover sheet and thesis abstract that is done outside the system, for example, the Title category. The CSV file will then go through the tokenization and feature extraction process. Tokenization will produce a word, number, and symbol tokens, and token classes from the process of moving a set of rows of data in a CSV file to an array. The resulting token will be given weight through the feature extraction process. Each token weight, word token, number, symbol and token class will go through the LVQ training process, resulting in the final weight stored in the final weight database, which is in the form of a CSV file.

2.4 Tokenization

A process for separating each word from a series of sentences is called the tokenization process. According to Amin, tokens are often referred to as

terms or words. A token is a sequence of characters from a particular document grouped as a semantic unit that is useful for processing [23]. The following is a block diagram tokenization.

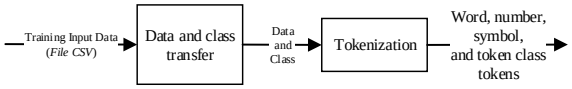


Figure 3. Tokenization Block Diagram

In this research, tokenization is not done through filtering stages to eliminate a number of unimportant terms because the terms that are not important or do not have meaning are very important for punctuation feature extraction.

2.5 Data Segmentation

Data segmentation is the process used to delete blank lines and generate segment data per line. This process is only carried out at the preprocessing data testing stage, the input data is a text file. The following is a block diagram of data segmentation.

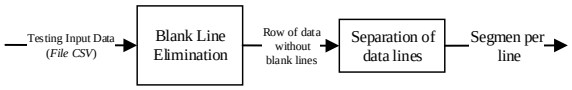


Figure 4. Data Segmentation Block Diagram

In Figure 4, a few blank lines on the input testing data go through the process of deleting blank lines. Then, the data row without blank lines is separated to become a number of segment lines.

2.6 Ekstraksi Fitur

According to Prihatini, feature extraction is a process to look for feature values contained in the document [7]. Features can be interpreted as a characteristic of every data recognized by the system so as to produce feature values. Feature extraction is an important topic in classification because good features will be able to increase the level of accuracy, while features that are not good tend to exacerbate the level of accuracy [8]. The following block diagram feature extraction.

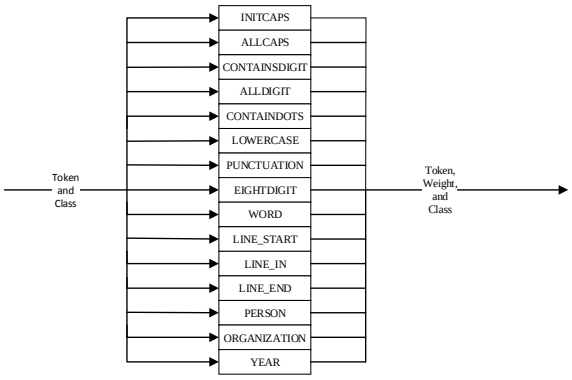


Figure 5. Feature Extraction Block Diagram

In this research, feature extraction is used to give a value/weight for each token based on a predetermined feature. 15 feature features are used. For 13 features refer to the research conducted by Peng [9], while for LOWERCASE features, and EIGHTDIGIT is features created by the author. The following explanation of the fifteen features used in this research can be seen in Table 2.

Table 2. Feature Extraction

No.	Feature Extraction	Information
1	INITCAPS	Identify each token whose letter begins with a capital.
2	ALLCAPS	Identify each token that is all capital letters.
3	CONTAINSDIGIT	Identify each token that contains digits.
4	ALLDIGIT	Identify each token that is all digits.
5	CONTAINSDOTS	Identify each token that contains a dots.
6	LOWERCASE	Identify each token that is all lowercase.
7	PUNCTUATION	Identify each token that contains certain marks such as dots, commas, colons, semicolons, parentheses, and exclamation points.
8	EIGHTDIGIT	Additional features in this research, this feature is specifically for recognizing tokens that have 8 digits in length.
9	WORD	Additional features in this study, this feature is devoted to giving weight to tokens for JENIS_PENELITIAN and KALIMAT_PENGAJUAN classes.
10	LINE_START	Identify the token position in the initial array index.
11	LINE_IN	Identify the token position in the middle array index.
12	LINE_END	Identify the token position on the final array index.

13	PERSON	Identify someone's name token.
14	ORGANIZATION	Identify an organization's token.
15	YEAR	Recognize the features of year tokens.

Each feature in Table 2 will give a weight of 1 or 0. Weight 1 is given by each feature if the token is included in the feature. Likewise, tokens that are not included in the feature will be given a weight of 0.

2.4 *Learning Vector Quantization*

Learning Vector Quantization (LVQ) is one part of the algorithm of Artificial Neural Networks (ANN) and is commonly called the LVQ Network. The LVQ network is created by the Self-Organizing Maps (SOM) algorithm [10]. Supervised or active learning is a process that requires teachers, something that has knowledge of the environment [16]. The LVQ network belongs to the competitive layer, consists of two layers and can work to perform multi-class classifications [11], which means that if 2 input vectors are close together, the competitive layer will put the two inputs vectors into the same class. The LVQ algorithm can be referred to as Single Layer Feedforward which means the neural network with feed-forward single-layer architecture [15]. The LVQ network is referred to as the Single Layer Feedforward because the "single-layer" question is referenced to the output node layer directly [17]. Following is the LVQ network structure.

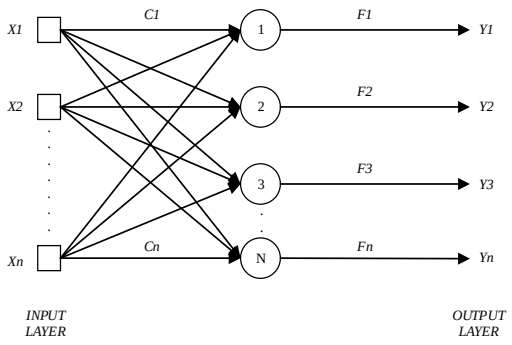


Figure 6. LVQ Network Structure

In

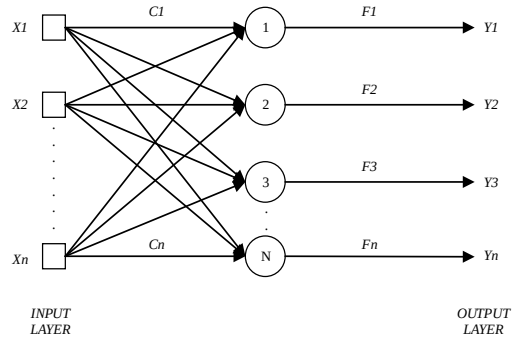


Figure **6**. the working principle of the LVQ algorithm is to automatically learn to classify input vectors into certain classes. following parameters and steps for conducting LVQ training and testing [12] [10].

Table 3. Descriptive Algorithms of LVQ Training and Testing

Training LVQ

1. Initialize Codebook Vectors, Epoch, Max Epoch, Initial Learning Rate, next Learning Rate, New Learning Rate, Reduction of Learning Rate, and expected minimum Error.
2. Set initial conditions for $Epoch = 0$.
3. Do Epoch until $MaxEpoch$ if $(Epoch < MaxEpoch)$ or $(\alpha > Eps)$.
4. Determine the minimum distance value with *Euclidean Distance* [11] :

$$d_{r,b} = \sqrt{\sum_{j=1}^{J} \left(x_{r,j} - c_{b,j}\right)^2} \quad (1)$$

5. Update c_w under the condition [11] :

- If $class\ x_r\ same\ with\ class\ c_w$ then :
  $c_w = c_w + \alpha(t) \cdot (x_r - c_w)$ (2)
  or can read with [4]
  $c_{b,j(new)} = c_{b,j(old)} + \alpha(t) \cdot \left(x_{r,j} - c_{b,j(old)}\right)$ (3)

- If $class\ x_r not\ same\ with\ class\ c_w$ then :
  $c_w = c_w - \alpha(t) \cdot (x_{rj} - c_w)$ (4)
  or can read with [4]
  $c_{b,j(new)} = c_{b,j(old)} - \alpha(t) \cdot (x_{rj} - c_{b,j(old)})$ (5)

For all other codebook vectors do not change :

$$c_b = c_b\ for\ b\ not\ same\ with\ w \quad (6)$$

6. Reduction *Learning Rate* (α) with formula [13] :

$$\alpha(t+1) = \alpha(t) - \frac{\alpha(0) - u}{K} \quad (7)$$

7. Addition *epoch* with formula [12] :

$$Epoch = Epoch + 1 \quad (8)$$

Testing LVQ

1. Inizialitation *Sample Vector* as $x_{r,j}$.
2. Prepare the final weight that has been obtained in the training phase as $c_{b,j}$.
3. Determine the minimum distance value with *Euclidean Distance* [10] :

$$d_{r,b} = \sqrt{\sum_{j=1}^{J} \left(x_{r,j} - c_{b,j}\right)^2} \quad (9)$$

4. Classify value of $d_{r,b}$ as class from $x_{r,j}$.

In the number formula (1) and (9), J is the total of jth neurons, $x_{r,j}$ is *sample vector* x_r, $c_{b,j}$ is *codebook vectors* c_b, and $d_{r,b}$ is the minimum distance between $x_{r,j}$ and $c_{b,j}$. Meanwhile, in the number formula (7), t in the form of time (shows the change in weight at one time in accordance with the number of inputs entered [14] or it can be seen as an iteration [13]), $\alpha(0)$ is the initial learning rate that is determined constantly, $\alpha(t+1)$ is the next learning rate from the previous time, $\alpha(t)$ is a new learning rate for a certain time, u is the value of reducing the learning rate, and K is the number of classes.

Need to know that if there is the same minimum value for each weight, then for the first minimum value obtained is the one who wins. Given that the LVQ algorithm has the concept of competitive learning [27]. LVQ training process requires codebook vectors and its parameters, namely epoch, maxepoch, learningrate, and others. Here are some parameters used in the LVQ algorithm can be seen in Table 4.

Table 4. LVQ Algorithm Parameters

No.	Parameter	Symbol
1	*Codebook vectors*	C
2	*Epoch*	$epoch$
3	*Maxepoch*	$maxepoch$
4	Learning rate at epoch 0	$\alpha(0)$
5	Next learning rate	$\alpha(t+1)$
6	Learning rate at a certain time	$\alpha(t)$
7	Number of Classes	K
8	Reducing Learningrate	$Pengurangan\ LearningRate$
9	*Eps*	Eps

Codebook vectors can be said as initial weight [12] or reference vectors [13]. Codebook vectors represent each class chosen as a reference for the training process.

2.5 Plotting Per Class

Each token that has each class will be plotted per class. Plotting per class is carried out to produce a series of data based on token classes, such as the research title category (cover) which is represented as class 0. Before plotting, the system goes through a process to improve the token class that does not match the original class. Following are block plotting diagrams per class.

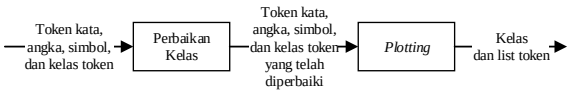


Figure 7. Block Diagram Plotting Per Class

In Figure 7, the word token, number, a symbol will be fixed in the class. Class repairs are carried out based on the results of the classification of each token that does not match the original class.

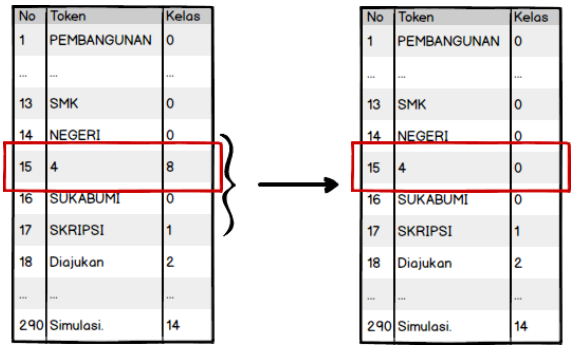
2.5.1 Class Repairment

Class repairment is a process to correct the token class mismatch results from LVQ algorithm classifications with the original class. Improvements are made to obtain better accuracy for the plotting process. The rules applied to repair the token class can be seen in Table 5.

Table 5. Rule Class Repairment

No.	Condition	Action
1	($T_i \neq T_{i-1}$) AND ($T_i = T_{i+1}$) AND ($T_i > T_{i-1}$)	Check the token class based on the order.
2	($T_i \neq T_{i-1}$) AND ($T_i \neq T_{i+1}$) AND ($T_i < T_{i-1}$)	Check the token class based on the majority of token classes.
3	($T_i \neq T_{i-1}$) AND ($T_i \neq T_{i+1}$) AND ($T_i > T_{i-1}$) AND ($T_i \neq T_{i-1+1}$)	Check the token class based on the order.
4	($T_i \neq T_{i-1}$) AND ($T_i \neq T_{i+1}$) AND ($T_i > T_{i-1}$) AND ($T_i < T_{i+1}$)	Check the token class based on the token class specified for 1 particular token.
5	($T_i \neq T_{i-1}$) AND ($T_i \neq T_{i+1}$) AND ($T_i < T_{i-1}$) AND ($T_i \neq T_{i-1+1}$)	Check the token class based on the order.
6	($T_i = T_{i-1}$) AND ($T_i = T_{i+1}$)	Check the token class based on the order.
7	($T_i = T_{i-1}$) AND ($T_i \neq T_{i+1}$)	Check the token class based on the majority of token classes.

In Table 5, symbol T shows the class variable i with several variations of parameters, such as T_{i-1}, T_{i+1}, and T_{i-1+1}. Here is one example of class improvement for rule number 3.



No	Token	Kelas
1	PEMBANGUNAN	0
...	...	...
13	SMK	0
14	NEGERI	0
15	4	8
16	SUKABUMI	0
17	SKRIPSI	1
18	Diajukan	2
...	...	...
290	Simulasi.	14

No	Token	Kelas
1	PEMBANGUNAN	0
...	...	...
13	SMK	0
14	NEGERI	0
15	4	0
16	SUKABUMI	0
17	SKRIPSI	1
18	Diajukan	2
...	...	...
290	Simulasi.	14

Figure 8. Example of Class Repairment

In Figure 8, the '4' number token has class 8 that does not match the original class. The original class of the number '4' token is class 0.

2.5.2 Plotting

Plotting is the process of forming a token list based on token classes. Each token that has a class token that results from a class upgrade will be combined based on the token class. The following is an example of forming a token list.

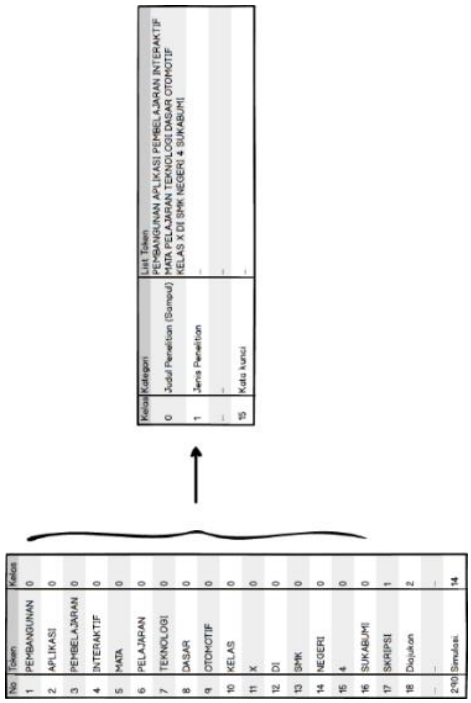


Figure 9. Example of Forming a List Token

In Figure 9, the system will detect each class of tokens that have the same token class 0, then combine them based on the similarity of the class. Token 0 class is a token class that has gone through the class repair process as shown in Figure 8. Each token that has class 0 is combined, thus forming a token list with the Research Title (Cover) category.

2.7 Test Result

Testing is done with the form of testing token-class and class-token. All documents that have an accuracy value will be calculated on the average value of accuracy and error. Following the results of testing the value of accuracy and error.

2.7.1 Form of Testing Token-Class

Every word token, number, and symbol that has a validated class is correct, then the level of truth and error is calculated, resulting in accuracy and error values. The following classification results with the form of testing token-class can be seen in Table 6.

Table 6. Form of Testing Token-Class Test Results

No.	Document Name	Accuracy	Error
1	1.txt	80%	20%
2	2.txt	87%	13%
3	3.txt	76%	24%
4	4.txt	82%	18%
5	5.txt	82%	18%
6	6.txt	75%	25%
7	7.txt	76%	24%
8	8.txt	72%	28%
9	9.txt	71%	29%
10	10.txt	70%	30%
11	11.txt	71%	29%
12	12.txt	71%	29%
13	13.txt	77%	23%
14	14.txt	77%	23%
15	15.txt	81%	19%
16	16.txt	82%	18%
17	17.txt	86%	14%
18	18.txt	84%	16%
19	19.txt	81%	19%
20	20.txt	83%	17%
21	21.txt	81%	19%
22	22.txt	79%	21%
23	23.txt	80%	20%
24	24.txt	78%	22%
25	25.txt	82%	18%
26	26.txt	78%	22%
27	27.txt	81%	19%
28	28.txt	86%	14%
29	29.txt	76%	24%
30	30.txt	81%	19%
31	31.txt	78%	22%
32	32.txt	73%	27%
33	33.txt	71%	29%
34	34.txt	67%	33%
35	35.txt	69%	31%
36	36.txt	83%	17%
37	37.txt	81%	19%

38	38.txt	77%	23%
39	39.txt	81%	19%
40	40.txt	82%	18%
Average		78%	22%

2.7.2 Form of Testing Class-Token

Tests are carried out with the form of testing class-tokens based on the results of plotting per class that has been done before. Some token lists based on their class consist of tokens that have gone through the class repair process. The following classification results with the form of testing token-class can be seen in Table 7.

Table 7. Form of Testing Class-Token Test Results

No.	Document Name	Accuracy	Error
1	1.txt	0%	100%
2	2.txt	13%	88%
3	3.txt	0%	100%
4	4.txt	0%	100%
5	5.txt	13%	88%
6	6.txt	6%	94%
7	7.txt	19%	81%
8	8.txt	0%	100%
9	9.txt	19%	81%
10	10.txt	6%	94%
11	11.txt	6%	94%
12	12.txt	0%	100%
13	13.txt	0%	100%
14	14.txt	6%	94%
15	15.txt	6%	94%
16	16.txt	13%	88%
17	17.txt	0%	100%
18	18.txt	6%	94%
19	19.txt	6%	94%
20	20.txt	13%	88%
21	21.txt	0%	100%
22	22.txt	0%	100%
23	23.txt	0%	100%
24	24.txt	0%	100%
25	25.txt	6%	94%
26	26.txt	0%	100%
27	27.txt	0%	100%
28	28.txt	6%	94%
29	29.txt	13%	88%
30	30.txt	13%	88%
31	31.txt	6%	94%
32	32.txt	6%	94%
33	33.txt	6%	94%
34	34.txt	6%	94%
35	35.txt	6%	94%
36	36.txt	0%	100%
37	37.txt	13%	88%
38	38.txt	13%	88%
39	39.txt	13%	88%
40	40.txt	13%	88%
Average		6%	94%

2.8 Analysis of Test Results

High and low accuracy and errors obtained have several causes. The analysis carried out on the test results with the token-class testing form concluded that the first is that because the input data used by the system is a text file, there are limitations to the features used. Second, low accuracy values are obtained because in the testing document there are several irregular symbols produced by the previous conversion process. Third, based on observations made, the effects that affect the decline in the value of accuracy are always caused by the ORGANIZATION feature which is devoted to giving weighting to the categories of Study Programs, Faculties, and Universities. Because, the LVQ algorithm has a competitive network, so if the Study Program is assumed to win with the same weight between Faculties and Universities, Faculties and Universities will be classified into the same class as the Study Program. Fourth, the lack of use of feature extraction can identify several tokens specifically to the number of classes, namely 16 classes, so to give weight to each token is not significant.

Whereas, based on observations, the value of the accuracy obtained never exceeds 50%, because the first is the existence of spelling/typo errors in word tokens, such as Teknologi Dasar into Teknologi Dasr. Second, the emergence of a space mark for word token and number token, such as 10111654 to 1011 (space) 1654. Third, the appearance of irregular symbols caused by the document conversion process, so that irregular symbols become a barrier to data, such as ___ ... WW. _ _N ...- yy n-r.

3 CLOSING

Based on the system functionality testing and accurate measurement that has been done, the information extraction system using Learning Vector Quantization (LVQ) algorithm has been successfully built with the acquisition of token-class accuracy of 78% and class-token accuracy of 6%. The acquisition of token-class accuracy is caused by the use of insignificant weighting features, so the LVQ algorithm cannot correctly classify. Meanwhile, the acquisition of class-token accuracy is caused by the influence of token-class LVQ algorithm classification, the existence of misspellings/typos, and the appearance of irregular symbols resulting from conversion from PDF files.

Suggestions for further development use input data with .docx or .html format to be able to use some additional features such as detecting bold, italic, underline, as well as several other font styles, a significant additional feature is needed for weighting the categories of Faculties, Study Programs, Universities, and requires checking and correcting spelling or typo errors on the conversion of PDF files.

BIBLIOGRAPHY

[1] D. Jurafsky dan J. H. Martin, "*Chapter* 21 - *Information Extraction*," dalam *Speech and Language Processing*, 2017, pp. 1 - 31.

[2] D. Mustaqwa, "Implementasi Ekstraksi Informasi Pada Dokumen Teks Skripsi Menggunakan Metode Rule Based," Skripsi, Universitas Komputer Indonesia, Bandung, Jawa Barat, 2018.

[3] A. I. Riaddy, Y. S. S.Si, M.T. dan A. A. S.Si, M.Si, "Ekstraksi Informasi pada Makalah Ilmiah dengan Pendekatan *Supervised Learning*," *e-Proceeding of Engineering*, vol. 3, no. No. 1, pp. 1184 - 1190, 2016.

[4] M. T. Pilevar, H. F. dan M. S. , "*Classification of Persian textual documents using Learning Vector Quantization*".

[5] N. dan Z. Abdul-Malek, "*Neural Networks for Fault Classification: Comparison between Feed-Forward Back-Propagation, RBF, and LVQ Neural Network*," *Applied Mechanics and Materials*, vol. 818, pp. 96 - 100, 2016.

[6] M. F. Umer dan M. S. H. Khiyal, "*Classification of Textual Documents Using Learning Vector Quantization*," *Information Technology Journal*, vol. 6, no. 1, pp. 154-159, 2007.

[7] P. M. Prihatini, "Implementasi Ekstraksi Fitur Pada Pengolahan Dokumen Berbahasa Indonesia," Jurnal Matrix, vol. 6, no. 3, pp. 174-178, 2016.

[8] M. Indrawijaya, L. dan R. Adipranata, "Aplikasi Ekstraksi Fitur Citra Hufur Jawa Berdasarkan Morfologinya".

[9] F. Peng dan A. MCCallum, "*Accurate Information Extraction from Research Papers using Conditional Random Fields*," pp. 1-8, 2004.

[10] I. N. D. Silva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni dan S. F. d. Reis Alves, "*Artificial Neural Networks A Practical Course*," dalam 9.3 *LVQ Networks (Learning Vector Quantization)*, Brazil, *Springer International Publishing Switzerland*, 2017, pp. 176 - 181.

[11] G. R. Lloyd, R. G. Brereton, R. Faria dan J. C. Duncan, "*Learning Vector Quantization for Multiclass Classification: Application to Characterization of Plastics*," J. Chem. Inf. Model., vol. 47, pp. 1553-1563, 2007.

[12] E. Sabrina dan I. G. P. A. Buditjahjanto, "Klasifikasi Penyakit Diabetic Retinopathy menggunakan Metode *Learning Vector Quantization* (LVQ)," Jurnal Teknik Elektro, vol. 6, no. Nomor 02, pp. 97 - 104, 2017.

[13] M. T. Martin-Valdivia, M. Garcia-Vega dan L. A. Urena-Lopez, "*LVQ for text categorization using a multilingual linguistic resource*," *Neurocomputing*, vol. 55, pp. 665 - 679, 2002.

[14] F. R. Hariri, E. Utami dan A. Amborowati, "*Learning Vector Quantization* untuk Klasifikasi Abstrak Tesis," Citec Journal, vol. 2, no. No. 2, pp. 128 - 143, 2015.

[15] M. W. Dessy and I. Afrianto, "Perbandingan Metode Jaringan Syaraf Tiruan *Backpropagation* dan *Learning Vector Quantization* Pada Pengenalan Wajah," Jurnal Komputer dan Informatika (KOMPUTA), pp. 45-51, 2012.

[16] Suyanto, "*Artificial Intelligence*," in *Searching, Reasoning, Planning*, dan *Learning* Revisi Kedua, Bandung, Informatika, 2014, p. 179.

[17] S. Haykin, "*Neural Networks and Learning Machines Third Edition*," *in Neural Networks and Learning Machines*, New Jersey, Pearson Education, 2008, p. 21.