THE IMPLEMENTATION OF THE RELEVANCE VECTOR MACHINE METHOD IN AUTOMATIC TEXT SUMMARIZATION

Arief Budiman Eka Putra
Indonesian Computer University
Jl. Dipatiukur No. 112-116, Bandung 40132
E-mail : ariefbud688@gmail.com

ABSTRACT

Support Vector Machine (SVM) is one of supervised machine learning methods used for regression and classification process. Support Vector Machine has been widely implemented in various real-world problems, one of which is to automatic text summarization. Based on study results conducted, Support Vector Machine has several weaknesses. Therefore, a classification and regression method was made to overcome Support Vector Machine weaknesses, i.e. Relevance Vector Machine (RVM). The results of various studies and comparisons conducted show that the Relevance Vector Machine in regression and classification has a better accuracy rate with faster computation time than Support Vector Machine. This study attempts to implement Relevance Vector Machine method on automatic text summarization on a single document. Relevance Vector Machine method will be used during sentence classification process of sentence preprocessing results in the document. Sentence in the document will be divided into two classes, positive class and negative class. Positive sentence contains a sentence included in the summary, while the negative sentence is not included in the summary. This study result found that the Relevance Vector Machine method can be implemented in automatic text summarization case. From its accuracy, summarization system was tested using the intrinsic summary evaluation method, in which resulting the average value of recall, precision and f-measure by 53%.

Keywords: SVM, RVM, Summarization, Preprocessing, Classification

1. INTRODUCTION

Support Vector Machine (SVM) is a learning method of machine that is supervised. It is used for regression and classification process [1]. From the results of several studies that examine both theoretical and implementation aspects, Support Vector Machine has several advantages, including the high degree of accuracy as well as its implementation that is relatively easy. Support Vector Machine has been widely implemented in various real-world problems, one of which is for automatic text summarization [2].

Based on the results of research conducted [1], Support Vector Machine has several drawbacks. One of the weaknesses is the other processes that are not probabilistic predictions can reduce the accuracy of the classification [1]. Therefore, a classification and a regression method are made to overcome the shortcomings owned by Support Vector Machine. The method is Relevance Vector Machine (RVM).

Relevance Vector Machine is a learning method of machine adapted from Bayesian Framework and has a similar shape to the function model of Support Vector Machine. From the results of various studies and comparisons show that in doing regression and classification, the Relevance Vector Machine has a level of accuracy that is better with a faster computing of Support Vector Machine [1] [3], which is indicated by the number of relevant vector (RV) which produced by Relevance Vector Machine that much less than the amount of support vector (SV) generated by Support Vector Machine.

This research attempts to implement the method of Relevance Vector Machine automatic on text summarization on single document. As with the other learning methods of machine, Relevance Vector Machine method will be used during the classification process results preprocessing sentences in the document. Sentences in the document will be divided into two classes, the positive and negative class. Positive sentence contains a sentence that is included in the summary, while the negative sentence is not included in the summary [2]. The objectives to be achieved in this research is to look at the accuracy of the method of Relevance Vector Machine in making a summary, by using the intrinsic method of evaluation i.e. the accuracy of the summary only measured by the quality of the output of summaries.

The results of the research hopefully can be used as a new approach in developing automatic text summarization that is still being developed and studied [4].

2. CONTENTS OF THE RESEARCH

This section is divided into several parts, the explanation of the system architecture, the preprocessing, the weighting term, the training, the classification, and the summary testing.

2.1 The System Architecture

The system architecture describes the technical components of the system to be built. The architecture of the text summarization system that to be built can be seen in Figure 1.

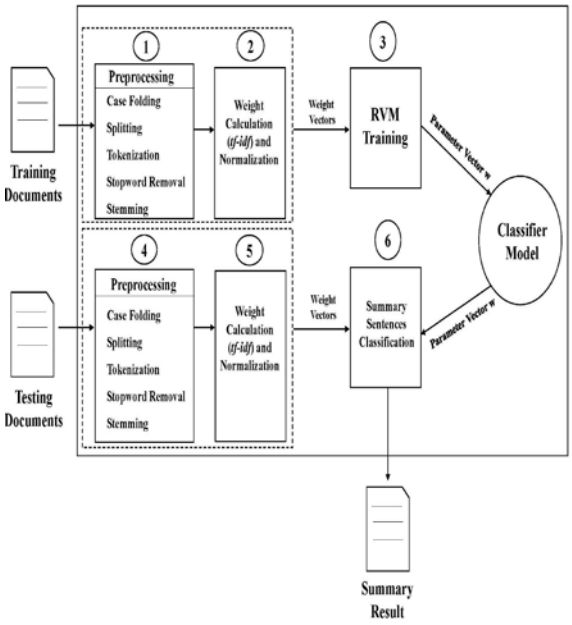


Figure 1. System Architecture

The system of the automatic text summarization that to be built has four main processes i.e. the preprocessing, the weight calculation of the word (term), the training process to generate the classification model, and the testing process in the form of sentences that would be used as a summary using a model that has been obtained in the training process.

For the explanation of each step in the process of the summarization based on Figure 1 can be seen as follows:

1. Trained data conducts the preprocessing which is the stage where the trained data will be purged from the noise or things that are not considered to have an influence in the summary. The preprocessing that is performed in this stage is case folding, splitting, tokenization, stopword removal, and stemming.
2. The next step is calculating the weight of each word (term) in the trained data. It uses tf-idf method to calculate TF (Term Frequency), DF (Document Frequency), and calculates the weight of each term in the trained data. Having obtained the tf-idf value of each sentence, the next step is normalizing the tf-idf.
3. The next step is conducting the training process of Relevance Vector Machine against vectors obtained from the previous process to obtain the classification model.
4. The next stage is the same as the first stage, i.e. perform preprocessing, but it is conducted for the tested data. Preprocessing is performed in this case is the same, namely folding, splitting, tokenization, stopword removal, and stemming.
5. Subsequently, the weighting weights on each term for tested data, which was the same as the third stage i.e. using the tf-idf method and later the tf-idf value will be normalized.
6. Next, the testing process is conducted by the classification model that has been obtained in the training process.

The explanation of each main process is explained in the next couple of sub-chapters.

2.2 Preprocessing

This preprocessing is the processes conducted to obtain the feature of clean words so it will not contain the noise that will disturb the results for further processing. The steps of preprocessing can be seen in Figure 2.

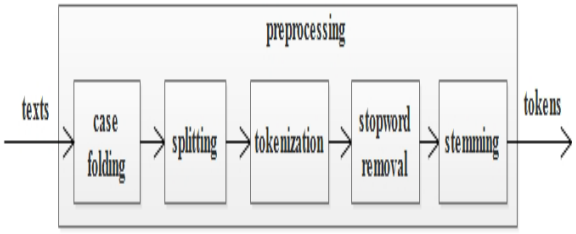


Figure 2. Preprocessing Steps

1. Case Folding
   Case folding is the process that converts all the letters in the document text to lowercase. The process of folding case can be seen in Figure 3. For example, the word “In” after going through the process of folding case turns into the word “in”, where the capital letter “I” was changed to the letter “i”.

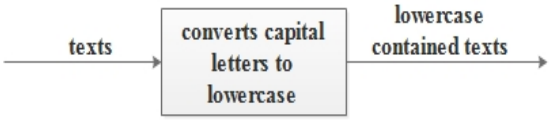


Figure 3. Case Folding

2. Splitting
   Splitting is breaking a whole text into fragments of sentences. An overview of the process of splitting can be seen in Figure 4. The process involves splitting the two sub-processes, namely splitsentences and punctuation.

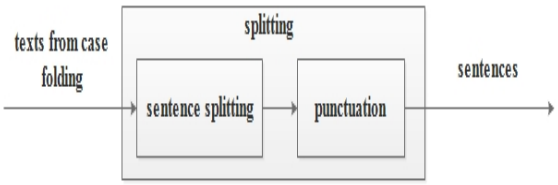


Figure 4. Splitting Steps

Sentence splitting (breaking the sentence) is the process of breaking a long string of document texts into a group of sentences. Breaking sentence in the document into sentences can be done by detecting the full stop (.), the question mark (?), and the exclamation mark (!) as the delimiter for cutting the document string [5].

The steps of sentence splitting can be seen in Figure 5.

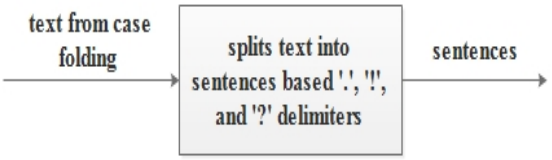


Figure 5. Sentence Splitting

Punctuation is a process of removing punctuations or symbols on the sentence that has been broken down [4]. The process of punctuation can be seen in Figure 6.

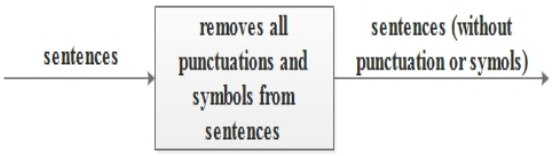


Figure 6. Punctuation

3. Tokenization
   Tokenization is the process of cutting the string into a collection of tokens. Splitting the sentence into single words is done by scanning the sentence with a separator (delimiter) and white space (spaces, tabs, and newline). The steps of tokenization can be seen in Figure 7.

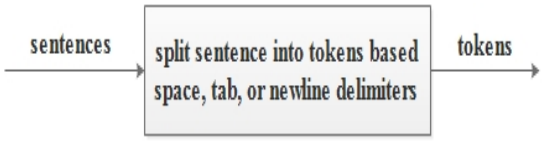


Figure 7. Tokenization

For example if there is a sentence “in matters of technology development”, after passing through the steps of tokenization, the tokens that are formed are “in”, “matters”, “of”, “development” and “technology”.

4. Stopword Removal
   Stopword removal is an elimination process of stopword. Stopword can be a preposition, conjunction, and pronoun. Stopword checking is done by checking whether the word is registered in the dictionary of stopword. If registered in the dictionary then the word will be omitted. Stopword dictionaries are taken from recommendations of Fadillah Z Tala in her research entitled “A Study of Stemming Effects on Information Retrieval in Indonesian”. The process of stopword removal can be seen in Figure 8.

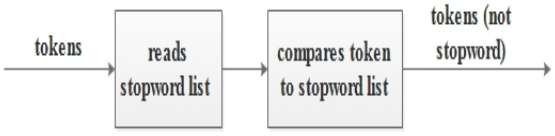


Figure 8. Stopword Removal

For example, the tokens are “in”, “matters”, “of”, “development” and “technology” after passing through the steps of stopword removal, the token is left become “matters”, “development” and “technology”. That is because the token “in” contained in the dictionary of stopword so that the token is removed.

5. Stemming
   Stemming is the process of returning additive word into the root word. The algorithm used in the process of stemming of this research is the algorithm of Nazief-Adriani. The description of the process of stemming can be seen in Figure 9.

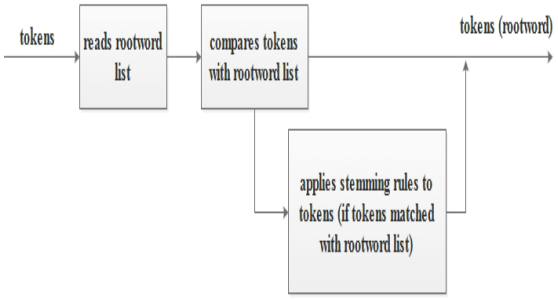


Figure 9. Stemming

For example, the token “development”, after passing through the steps of stemming, it becomes “develop”.

2.3 The Term Weighting

The term weighting is done to calculate the weight of term in the sentence. These terms are tokens resulted from the preprocessing that had previously been done. The term weighting uses the tf-idf method. The method is calculating the number of terms in the document (tf) and this calculation is found from an existing document (idf). The tf-idf equation that will be used is the equation of tf-idf that normalized, the equation is as follows

$$w_{i,j} = \frac{tf_{i,j} \cdot \log_2 \frac{N}{n_j}}{\sqrt{\sum_{s=1}^{k} (tf_{i,s} \cdot \log_2 \frac{N}{n_j})^2}} \quad (1)$$

where tf is the term frequency, N is the number of documents, and n_j is the number of documents in which the term j appears.

2.4 Relevance Vector Machine Training

The training of Relevance Vector Machine method is done to get the parameters of the equations model of classifications that will be used for the tested process of classification. The steps in the training of Relevance Vector Machine can be seen in Figure 10.

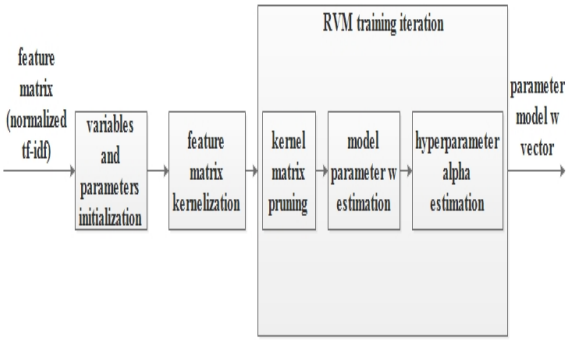


Figure 10. Relevance Vector Machine Training

In training process, the data will be used as trained data that must first be labelled to indicate the class of elements such data. In this case, the sentences in the trained data will be labelled class is a number of 0 or 1. The number 0 represents that the sentence into the negative class (the sentence is not important), while the number 1 represents that the sentence is entered into the positive class (the sentence is not important).

Based on Figure 10, there are five major processes in training, namely initialization of parameters and variables, kernelization of matrix feature, pruning of kernel matrix, estimation of model parameters w, and estimation of hyperparameter models. Detail steps on the fifth process is as follows.

1. Initialization of Parameter
   Initialization of parameters and variables is a process of initial values of the parameters necessary for the training process. The parameters that will be initialized is the parameter model of vector w, vector of hyperparameter α, and the maximum number of iterations performed. In this research, the elements of the model vector w will be initialized with a value of 0, the vector of the hyperparameter α will be initialized with any value, the maximum iteration, consisting of a maximum number of iterations and the maximum number of iterations in the process of the model w of parameter estimation constrained respectively of 250 and 25 iterations.
2. Kernelization of Matrix Features
   Kernelization of matrix feature is used to change the trained data of matrix resulted from a feature extraction into kernel matrix. The function of the kernel used in this research is the function of kernel gauss, with the equation:

$$K(x, x') = \exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right) \qquad (2)$$

   where x and x' is the trained data and σ is a free parameter. The matrix resulted from kernel will also be coupled with bias, where the first column of the matrix results kernel will be filled with the number of 1, resulting in a matrix columns with the size of M+1, where M is the initial number of columns of trained matrix data. Bias values will be used during the classification process.
3. Pruning of Kernel Matrix
   Pruning is the process of cutting element of the kernel matrix which are considered irrelevant. Kernel matrix elements that are not relevant will not be processed in the next iteration. The indicator of a kernel matrix element whether relevant or not is the hyperparameter α value generated by the matrix elements. If the value of hyperparameter α is considered very large to the value of infinity, then the matrix that corresponds to the value of hyperparameter will be omitted. To determine that the value of the hyperparameter included in the relevant category or not needed a value of comparator alpha, which in this research, the value of comparative is determined by score of 10^{12}.
4. Estimation of Parameter Model w
   Estimation of the parameter model w is performed by maximizing the objective function in the equation:

$$\ln p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}) = \sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \qquad (3)$$

   where N is the number of trained data, w is the weight vector, t is a vector of class labels of trained data, and A is a diagonal matrix consisting elements of vector α. In addition to maximizing the objective function of ln p (w|t, α), the estimated parameter w also done by minimizing the value of the function -ln p (w|t, α), The detailed steps of estimation of model parameters w are outlined in the block diagram in Figure 11.

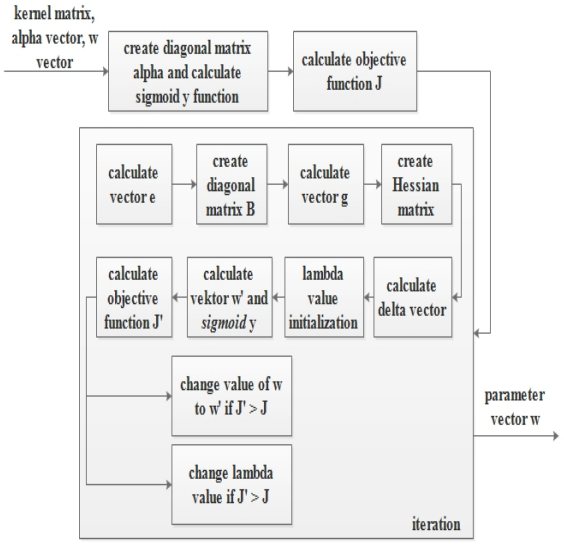


Figure 11. Model Paremeter w Estimation

The first step is to create diagonal matrix **A** that its diagonal element is the value of the vector of hyperparameter **α** which has been initialized previously in the first step. The next step is to find the variable of vector y using equation:

$$y(\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-(\mathbf{w}.\varphi(\mathbf{x})))} \quad (4)$$

where x is the trained data, w is a vector of model parameters that have previously been initialized, φ(x) is a kernel function.

After getting the required parameters, then the equation (3) can be done to maximize the process. The estimation process uses iterative reweighted least squares (IRLS) that first seeks the inverse of the result of the equation (3). After getting the value of the inverse of the equation (3), the next step is the process of iterative reweighted least squares with the following steps.

a. First of all, find the vector of variables e, by the equation:

$$\mathbf{e} = \mathbf{t} - \mathbf{y} \quad (5)$$

where **t** is the vector of label class and **y** is vector y.

b. The next step is to find the vector **β**, which each values of vector element calculated by the equation:

$$\beta_j = y_j\,(1\text{-}y_j) \quad (6)$$

where y is the vector elements of **y**, and j is a vector index.

c. After getting the vector **β**, the next is to create a diagonal matrix **B**, its diagonal elements consist of elements of the vector **β**.

d. The next step is to find for the negative gradient of the variable J by using the equation:

$$\mathbf{g} = \varphi_n^T e - \mathbf{A}w_n \quad (7)$$

where **φ** is the kernel matrix, as well as the **B** and **A** is a diagonal matrix of vector elements **α** and **β**.

e. The next step is to create a matrix of Hessian **H** from the second derivative of the objective function of equation (3). It uses the equation:

$$\mathbf{H} = \boldsymbol{\varphi}_n^T \mathbf{B} \boldsymbol{\varphi}_n + \mathbf{A} \quad (8)$$

where **φ** is the kernel matrix, as well as the **B** and **A** is a diagonal matrix of vector elements **α** and **β**.

f. The next step is to calculate the vector Δ using the equation:

$$\Delta = \mathbf{H}^{-1}.\mathbf{g} \quad (9)$$

where $\mathbf{H}^{-1}$ is inverse of the Hessian matrix, and **g** is the negative gradient vector of the objective function of equation (3).

g. The next is to initialize the value of λ that will be used to process of iteration of update vector y, the value of the objective function of equation (3) and the weight vector **w**. Initialize the value of λ with a value of 1. As for the iteration process of update vector y, the value of the objective function is equation (3) and the weight vector w is as follows.

1) Calculate the vector **w'** using the equation:

$$\mathbf{w}' = \mathbf{w}_n + \lambda\Delta \quad (10)$$

where $\mathbf{w}_n$ is a parameter vector that previously initialized to 0 and Δ that is vector Δ.

2) Next update vector **y** using the equation:

$$\mathbf{y} = \frac{1}{1 + \exp(-(\mathbf{w}'.\varphi(\mathbf{x})))} \quad (11)$$

where **w**' is the weight vector and φ (**x**) is a kernel function.

3) Next, calculate the variable J' using the following equation:

$$J' = -\sum_{n=1}^{N}\{t_n \ln y_n + (1 - t_n)\ln(1 - y_n)\} + \frac{1}{2}\mathbf{w}'^T\mathbf{A}\mathbf{w}' \quad (12)$$

where N is the number of trained data, **w**' is the weight vector, **t** is a vector class labels of trained data, and **A** is a diagonal matrix consisting of vector elements **α**.

4) Then compare the calculation result of J' with the value of objective function of equation (3) before. If the value of J' is more than the previous value, that value is considered a minimum and change the parameter vector w with the new values and give the lambda value to 0 in order to update the iteration variable y, J' and the weight of vector w stopped. Meanwhile, if the value of J' is less than the previous value, then the value of lambda divided by 2, and

the iteration updates of variable y, J' and the weight of vector w resumed. The results obtained from previous calculations that the new value J' is not greater than the previous value, therefore the iteration process continued.

h. After iteration step a) to g) performed and have reached the maximum number of iterations, the new vector **w** has been obtained, and the process of iterative reweighted least squares have been completed.

5. Estimation of Hyperparameter Alpha

After the estimation process of parameter vector w has been done, the next step is to estimate hyperparameter alpha α. The steps in terms of the estimated hyperparameter can be seen in Figure 12.

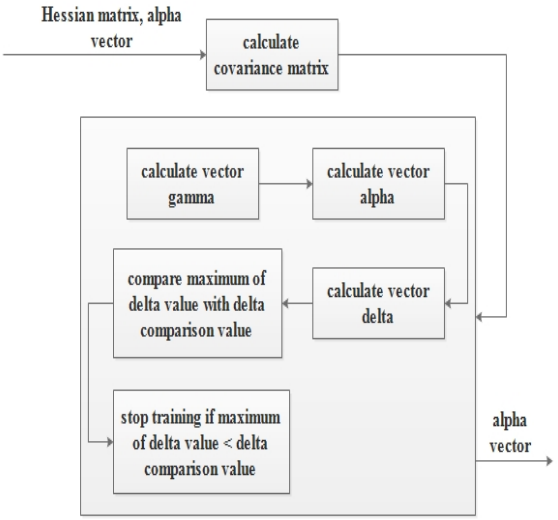


Figure 12. Estimation of Hyperparameter Alpha

The first step is to make a matrix of covariance **Σ**, using the equation:

$$\mathbf{\Sigma} = \mathbf{H}^{-1} \tag{13}$$

where $\mathbf{H^{-1}}$ is the inverse of the Hessian matrix. The next step is to estimate the iteration process of hyperparameter **α** vector. Its processes are as follows.

a. Calculate the variable vector γ using the equation

$$\gamma_j = 1 - \alpha_j \Sigma_{jj} \tag{14}$$

where α is a vector hyperparameter and Σ is the covariance matrix.

b. The next step is to replace the previous hyperparameter vector to the new vector hyperparameter. It is obtained by using the equation (15), where γ_i is the vector

$$\alpha_j = \frac{\gamma_j}{w_j^2}; \text{if } i < \frac{i_{max}}{2}$$
$$\alpha_j = \gamma_j \left(\frac{w_j^2}{\gamma_j} - \Sigma_{jj} \right)^{-1}; \text{if } i \geq \frac{i_{max}}{2} \tag{15}$$

elements of **γ**, w_i is an element of the weight vector **w**, and i_{max} is the maximum number of iterations of the training have been initialized earlier. Next, hyperparamater vector is replaced with a new hyperparameter vector.

c. The next step is to calculate the variable vector **δ**, using the equation

$$\delta_j = \left| \ln(\alpha_j) - \ln(\alpha_j^{lama}) \right| \tag{16}$$

where **α** is a hyperparameter vector.

After iteration steps a through c are done, the next step is to compare the value of the largest element of the vector of delta alpha δ with minimum values for the alpha delta. If the value of the largest element of the vector of delta alpha δ is less than the minimum values for the alpha delta, the training process is completed and the estimated parameters considered optimum. Meanwhile, if the value of the largest element of the vector alpha δ delta is bigger or equal to the minimum values for the alpha delta, the training process is being continued. The comparison minimum value of delta alpha in this research is 10^{-6}.

All of the steps that have been made constantly repeated until it reaches the maximum value of iteration or another stop condition has been reached. The end result of the training process is the form of the vector of model parameter w that have been optimal.

2.5 Relevance Vector Machine Classification

After obtaining the vector of model parameter w in the training process that has been done before, next, do the classification using existing models. The classification process in the case automatic text summarization is the classification of sentences in a text into sentences that are important (the probability the importance score is more than 0.5), and the unimportant sentence (the probability the importance score is less than or equal to 0.5). The steps of the classification can be seen in Figure 13.

Before the classification, firstly, the calculation of the predicted value for each element vectors of the tested data of sentences is conducted by using the equation:

$$y(x; w) = \sum_{i=1}^{M} \mathbf{w}^T \varphi(\mathbf{x}) + w_0 \tag{17}$$

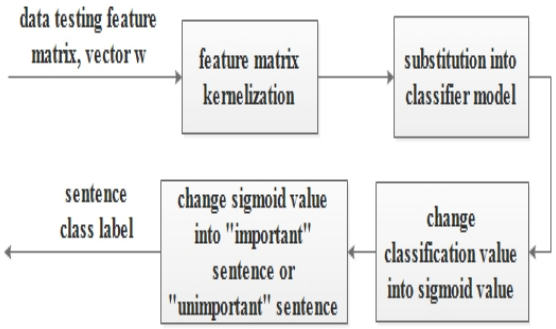


Figure 13. Relevance Vector Machine Classification

where **w** is a weighting parameter vector which has been obtained in the training process, and **x** is the data. The sentence predictive value is then converted into the form of probabilities, using logisticsigmoid function, using the equation:

$$\sigma(y) = \frac{1}{1 + \exp(-(y))} \quad (18)$$

where y is the predicted value previously calculated. The probability value will be converted into class labels (0 or 1), which will determine whether the sentence into the sentence is important or not. If the value of the probability of a sentence is more than 0.5, then the sentence has a numeric value of 1 or into the class label sentence " Important" Meanwhile, if the sentence probability value is less than or equal to 0.5, then the sentence has a numeric value of 0 or categorized a class label of the phrase "Not Important".

After the classification of sentences, the final step is to construct the results of sentence classification into a summary. Summary consists of a group of sentences that have a class label of the words "Important" or of value 1. The amount of the sentence is taken as many as 30% of the overall sentence. If the results of the classification are more than 30% of the number of sentences, then the sentences to be taken are sentences that have the highest probability value.

2.6 The Test of Resulted Summary

The test of results summary is done using an intrinsic evaluation method that is only measured by the quality of the output generated summaries. The evaluators create a manual summary with an extractive model to test the text. Then compare the results of the system-based summary with the manual summary.

The data that will be used to perform the testing is the news of "The Shooting of Russian Ambassador in Ankara". The news is divided into two parts, namely the trained data and tested data. Trained data consists of 20 news articles and the tested data is as much as 6 news articles that is taken from several news portal sites, such as detik.com, kompas.com, viva.com, voa.com, and okezone.com. For the trained data and test data, a summary of the manual will be made by a linguist i.e. an Indonesian teacher. The tested data will be calculated its performance to measure the accuracy of the automatic text summarization system that have been developed. The details of the trained data and tested data that to be used can be seen in Table 1, and the comparison of the resulted summary generated manually by language experts and the system can be seen in Table 2.

Table 1. Data Training and Data Tesing Detail in Summary Testing

Process	Number of Articles	Number of Important Sentences	Number of Unimportant Sentences
Training	20	98	230
Testing	6	35	81

Table 2. Comparison Of The Resulted Summary Generated Manually By Language Experts And The System

Document	Number of Summary Sentences	Summary Index (Manual)	Summary Index (System)
1	5	1,2,4, 5,17	1,2,3, 4,15
2	3	1,2,5	1,2,4
3	5	1,2,3, 8,12	1,2,3, 5,6
4	12	1,2,3, 4,16,19, 20,21,22, 26,29,31	1,2,3, 4,5,6, 9,11,14, 15,16,17
5	5	1,2,4, 6,16	1,2,3, 13,14
6	4	1,2,4, 11	1,2,3, 13

Performance testing is done by calculating the recall, precision and F-measure for each document, with the equation:

$$r = \frac{\sum KR}{\sum KRM} \quad (19)$$

$$p = \frac{\sum KR}{\sum KRS} \quad (20)$$

$$f = \frac{2 * p * r}{p + r} \quad (21)$$

ΣKR is a number of relevant sentences, the number of sentences contained in the summary of the system and also contained in the summary of the manual, ΣKRM is number of summary sentences generated manually,

and ΣKRS is number of summary sentences generated by the system.

The calculation results of the recall, precision and F-measure for each document can be seen in Table 3.

Table 3. Performance Calculation Results

Dok	ΣKRM	ΣKR	ΣKRS	*r*	*p*	*f*
1	5	5	3	0.6	0.6	0.6
2	3	3	2	0.67	0.67	0.67
3	5	5	3	0.6	0.6	0.6
4	12	12	5	0.42	0.42	0.42
5	5	5	2	0.4	0.4	0.4
6	4	4	2	0.5	0.5	0.5

Information:
ΣKRS: Number of Summary Sentences Generated By The System
ΣKRM: Number of Summary Sentences Generated Manually
ΣKR: Number of Relevant Sentences
r: recall
p: precision
f: f-measure

The results of the calculations in Table 3 are then calculated their average score to obtain a value of recall, precision and F-measure, which resulted in a score of 0.53 or 53%.

3. CLOSING

The conclusion from this research is that the Relevance Vector Machine method can be implemented in the case of the automatic text summarization for single document. In the term of accuracy, the summarization system was tested using the intrinsic method of evaluation summaries to find their score of recall, precision and F-measure. From the intrinsic test, the average score of recall, precision and F-measure are obtained i.e. 53%.

There are some suggestions for improvement in this research, which are as follows.

1. For the preprocessing, especially in the stopword removal. The stopword removal methods can be tested not only by using stopword dictionaries but also by using the statistical approach.
2. There are other extractions of other features that can be tested, such as the position of the sentence, lexical bond among sentences, and other features that have been tested by any other machine learning methods [2].

REFERENCES

[1] M. E. Tipping, "Relevance Vector Machine," *Advances in Neural Information Processing Systems,*, vol. NIPS 1999, no. 12, pp. 652-658, 2000.

[2] D. Fitriaman, M. L. Khodra and B. R. Trilaksono, "Peringkasan Teks Otomatis Berita Berbahasa Indonesia Pada Multi-Document Menggunakan Metode Support Vector Machines (SVM)," Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung, Bandung, 2012.

[3] B. Demir, "Hyperspectral Image Classification Using Relevance Vector Machines," *IEEE Geoscience And Remote Sensing Letters*, vol. 4, no. 4, pp. 586-589, 2007.

[4] Juan-Manuel and T.-M. , Automatic Text Summarization, Great Britain: ISTE Ltd and John Wiley & Sons, Inc., 2014.

[5] A. D. Tahitoe and D. Purwitasari, "Implementasi Modifikasi Enchanced Confix Stripping Stemmer Untuk Bahasa Indonesia Dengan Metode Corpus Based Stemming," *Jurnal Informatika*, 2010.