AUTOMATIC TEKS SUMMARIZATION FOR MULTI DOCUMENTS USING *RELEVANCE VECTOR MACHINE* METHOD

Tendi Arifin[1], Kania Evita Dewi[2]

[1,2] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
Email : tendiarifin0308@gmail.com[1] , kania.evita.dewi@email.unikom.ac.id[2]

ABSTRACT

Auto teks summarization on multi documents is teks summarization many documents with same theme using machine. On the research before, automatic teks summarization on multi documents is using Support Vector Machine methode and the result is recall 0.773, precission 0.771, f-measure 0.771. But on Christopher M. Bishop, Michael E. Tipping research have result Support Vector Machine have higher error than RVM (Relevance Vector Machine) that is SVM 10,6% and RVM 9,3%.

Step on teks summarization is recieve document teks input, preprocessing, weighting proces with fitur ekstraktion and the fiturs is sentence postion, sentence similarity, similarity sentence with title of document, sentence have entity word, sentence have numeric data, long sentence, sentence connectivity, and count of sentence weight connection and next step is result of fiture extraction will be process using RVM, weighting again but on this step using tf-idf method and then the sentence will be stacking with cosine similarity method.

Based on testing result automatix teks summariztion on multi documents using RVM method have performation result quite small the result is recall is 47.83%, precission is 35.48%, f-measure is 40.74%, dan akuration is 67.35%. it happens because the traing date is not balanced between important sentence and not important sentence.

Keyword : *Automatic teks summarization, Automatic teks summarization on multi documents, Relevance Vector Machine, Classification method, Machhi learning RVM.*

1. INTRODUCTION

News webs always have same topic but many different titles. That make reader not easy to understanding the what the main topic and wasting time. To make reader easy for understanding the topic one of manner for summarization is summaring by some peaple. But thats manner is so hard to do and need long time for have result. Because of that need tools/system for summaring [1].

Agus Riyanto reseach is auto teks summarization using K-means methode have result average *precission* 45%, *recall* 57.74%, *f-measure 46,13%* and on this reseach have suggestion to use other algorithm to increase preformation [2]. And on reseach auto teks summarization for multi documents has did because of Deni Firriaman, Masayu Leylia Khodra and Bambang Rianto Trilaksono. This reaseach is auto teks summarization for multi documents using SVM (*Support Vector Machine)*. SVM onthat reseach is using for classification sentence be included on summarization. And then sentence result classification is stacking using MMR (*Maximum Marginal Relevan)* method. This reseach have result average *recall* 0.773, *precission* 0.771, *f-measure* 0.771 [3].

Development SVM method is RVM (*Relevance Vector Machine)*. On Christopher M. Bishop, Michael E. Tipping reseach have result SVM error is bigger than RVM the percentation is SVM 10,6% dan RVM 9,3% [4].

Reseach about fitur extraction did because of Zulkifi, Agung Toto Wibowo, Gia Septiana. On that resach is have result the good fitur extraction is sentence postion, sentence similarity, similarity sentence with title of document, sentence have entity word, sentence have numeric data, long sentence, sentence connectivity, and count of sentence weight connection that fiturs have accuration 53% [5].

Be base on text above the problem formulation can RVM in order to teks summarization for multi documents. The aim this reseach is build auto teks summarization for multi documents and the purpose for measure as much as good the result summarization for multi document using RVM method with find out performation summarization value (*recall, precission, F-measure)* and *accuracy*.

2. CONTENTS OF THE RESEARCH

This section is devide into several parts, the explaination of the system architecture, the preprocessing, weighting fiture extraction, the taining, the calssification, the cosin similarity, 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.

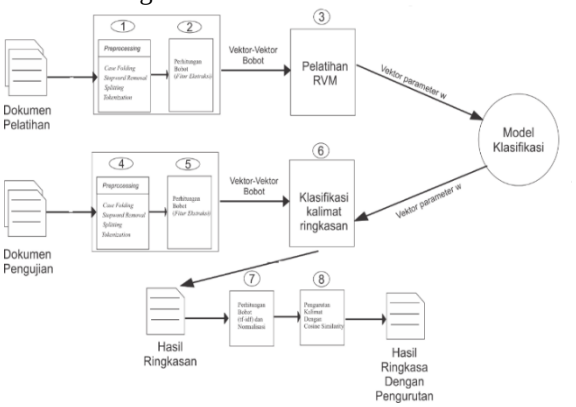


Figure 2.1. System Architecture

The system of the automatic text summarization that to be built has five main processes i.e. the preprocessing, the weighting with *fitue extraction*, 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 and stacking sentence result sing *cosine similarity* method.

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 using fitur extraction where the fitur is sentence postion, sentence similarity, similarity entence with title of document, sentence have entity word, sentence have numeric data, long sentence, sentence connectivity, and count of sentence weight connection.
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 fitur extraktion.
6. Next, the testing process is conducted by the classification model that has been obtained in the training process.
7. And the step is weighting using tf-idf method and will be count using cosine simiarity method the procces has did for stack the result of summary.

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.

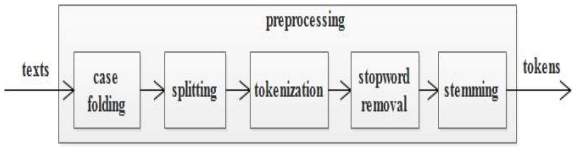


Figure 2. Preprocessing Steps

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”.

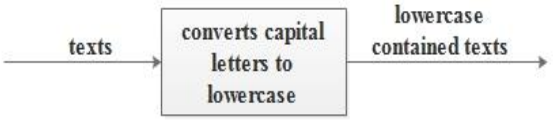


Figure 3. Case Folding

   On this proses have other process, the process is entity union. Result from this process will be use on next proccess.
2. Splitting
   Splitting is breaking a whole text into fragments of sentences. An overview of the process of splitting can be seen in Figure 2.4. The process involves splitting the two sub-processes, namely splitsentences and punctuation.

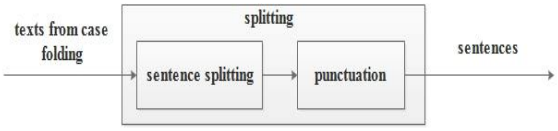


Figure 2. Splitting Steps

   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.

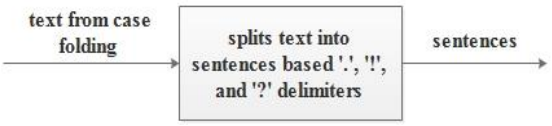


Figure 3. Sentence Splitting

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.

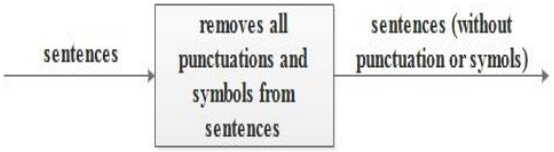


Figure 4. Punctuation

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.

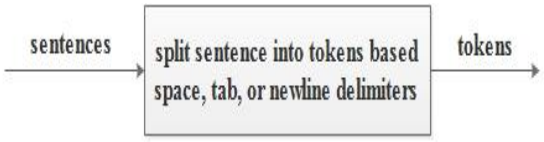


Figure 5. Tokenization

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.

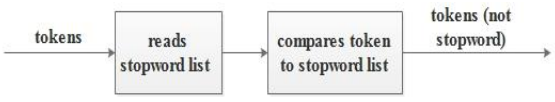


Figure 6. Stopword Removal

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.

2.3 The Fitur Extraction Weighting

Fitur extraction is manner for counting score fromm sentences on documments [5]. For senstens on document, the sentencs score is count be base on fitur extraction where the fitur is can be normalization until the score on range [0,1]. The normalization in order to every fitur not have big gap or difference.

Fitur extraction has used o this reseac is:

1. Sentence Position (f1)

Sentence position is sentence position on the paragraph The first sentence on paragraph is most importand [5]. His fitur can be count with formula 2.1:

$$Score_{f1}(s_i) = \frac{N-i}{N} \qquad (2.1)$$

Where :

N is total of sentence and in is sentence position (start from 0).

2. Sentences Similarity (f2)

Sentence similarity is list of word can tally between one sentence and othhe sentence on deocument [5]. This fitur counting using formula 2.2:

$$Score_{f2}(s) = \left| \frac{keyword\ inside \cap\ keyword\ inside\ between\ sentence}{keyword\ inside\ s \cup keyword\ inside\ between\ sentence} \right| \quad (2.2)$$

3. Similarity Sentence with Title (f3)

Similarity sentence with title of document is words on sentences can tally with words on title of document [5].

This fitur can be count using formula 2.3 :

$$Score_{f3}\ (s) = \left| \frac{keyword\ inside\ s \cap\ keyword\ inside\ title}{keyword\ inside\ s \cup keyword\ iside\ title} \right| \quad (2.3)$$

4. Sentence have entity (f4)

Named entity is list word have sense or compose the istitution. For examle Real madrid is words collection and have sense football club. This fitur can be count using formula 2.4 :

$$Score_{f4}\ (s) = \left| \frac{\#(entity\ inside(s))}{length(s)} \right| \quad (2.4)$$

5. Sentence Have Numeric Data (f5)

On teks summmary reckon numeric data because the sentence have numeric data usually importand and must be on smmary document [5]. This fitur can be count using formula 2.5 :

$$Score_{f5}\ (s) = \left| \frac{\#(numeric\ date\ in\ sentences\ (s))}{length(s)} \right| \quad (2.5)$$

6. Long Sentence (f6)

This fitur as a puspose for dissapear the shorter sentence. This fitur can be count base on word count on sentence alloted with word total on paragraph. Tis fitur can be counted using forula 2.6 :

$$Score_{f6}\ (s) = \left| \frac{\#(words\ total\ in\ sentences\ s)}{\#words\ total\ in\ document} \right| \quad (2.6)$$

7. Sentence connction

Sentence connection is lot of link on sentences connect with other sentences or sentence sum have same word with ote sentence on the one document. This fitur can be count using formula 2.7 :

$$Score_{f7}(s) = \left| \frac{\#total\ connectons\ in\ sentences\ s}{\sum_{i=0}^{n} \#total\ connections\ bettween\ sentences\ s} \right| \quad (2.7)$$

8. Weighting sum sentences connection (f8)

This fitur have function counting word on sentence with ssame word from other sentence on document. This fitur can be count using formula 2.8:

$$Score_{f8}(s) = \left| \frac{\#total\ same\ words\ in\ sentences\ s}{\sum_{i=0}^{n} \#total\ same\ words\ in\ sentences\ s} \right| \quad (2.8)$$

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 9.

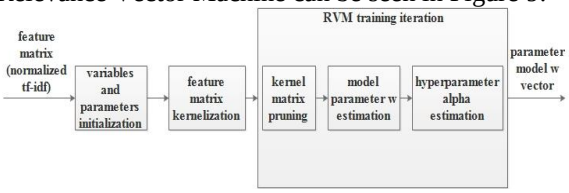


Figure 9. Relevance Vector Machine Training

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:

$$K(x, x') = \exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right) \quad (2.9)$$

   where x and x' is the trained data and σ is a free parameter.
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:

$$\ln p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}) = \sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \quad (2.10)$$

   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.

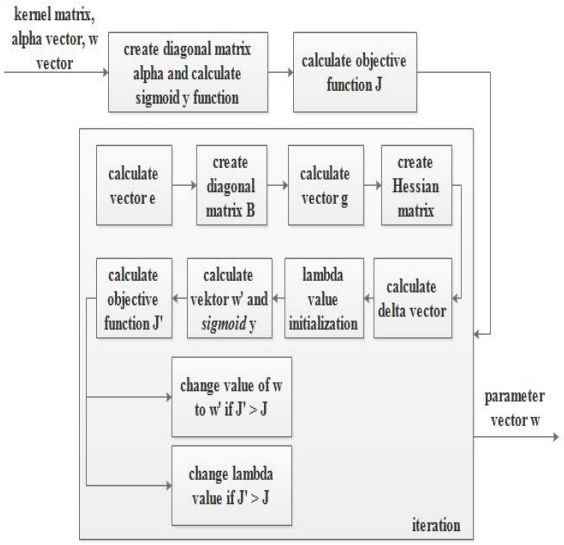


Figure 7. Model Paremeter w Estimation

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:

$$y(\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-(\mathbf{w}.\varphi(\mathbf{x})))} \qquad (2.11)$$

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 (2.10) 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 (2.10). After getting the value of the inverse of the equation (2.10), 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:

$$\mathbf{e} = \mathbf{t} - \mathbf{y} \qquad (2.12)$$

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:

$$\beta_j = y_j\,(1\text{-}y_j) \qquad (2.13)$$

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:

$$\mathbf{g} = \varphi_n^T \mathbf{e} - \mathbf{A}w_n \qquad (2.14)$$

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:

$$\mathbf{H} = \boldsymbol{\varphi}_n^T \mathbf{B} \boldsymbol{\varphi}_n + \mathbf{A} \qquad (2.15)$$

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:

$$\Delta = \mathbf{H}^{-1}.\mathbf{g} \qquad (2.16)$$

where $\mathbf{H}^{-1}$ is inverse of the Hessian matrix, and **g** is the negative gradient vector of the objective function of equation (2.10).

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 (2.10) 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 (2.10) and the weight vector w is as follows.

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

$$\mathbf{w}' = \mathbf{w}_n + \lambda\Delta \qquad (2.17)$$

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:

$$\mathbf{y} = \frac{1}{1 + \exp(-(\mathbf{w}'.\varphi(\mathbf{x})))} \qquad (2.18)$$

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

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

$$J' = -\sum_{n=1}^{N}\{t_n \ln y_n + (1 - t_n)\ln(1 - y_n)\} + \frac{1}{2}\mathbf{w'}^{T}\mathbf{A}\mathbf{w'} \quad (2.19)$$

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 2.12.

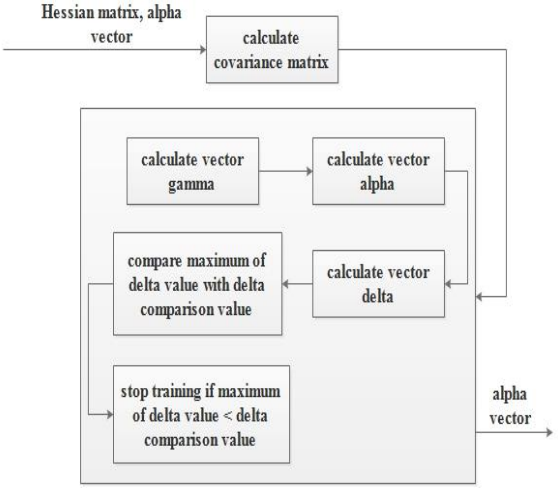


Figure 2.8. Estimation of Hyperparameter Alpha

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

$$\mathbf{\Sigma} = \mathbf{H}^{-1} \quad (2.20)$$

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

$$\gamma_j = 1 - \alpha_j \Sigma_{jj} \quad (2.21)$$

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 (2.21), where γ_i is the vector

$$\alpha_j = \frac{\gamma_j}{w_j^2}; \text{if } i < \frac{i_{max}}{2}$$

$$\alpha_j = \gamma_j \left(\frac{w_j^2}{\gamma_j} - \Sigma_{jj}\right)^{-1}; \text{if } i \geq \frac{i_{max}}{2} \quad (2.22)$$

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

$$\delta_j = \left|\ln(\alpha_j) - \ln(\alpha_j^{lama})\right| \quad (2.23)$$

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:

$$y(x; w) = \sum_{i=1}^{M} \mathbf{w}^{T}\varphi(\mathbf{x}) + w_0 \qquad (2.24)$$

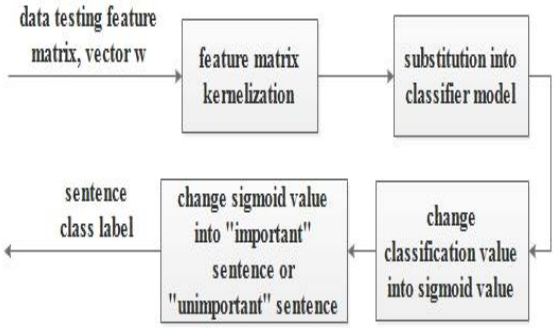


Figure 9. Relevance Vector Machine Classification

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:

$$\sigma(y) = \frac{1}{1 + \exp(-(y))} \qquad (2.25)$$

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 no limit.

2.6 Cosine Similarity

Sentences result summaryzation will be stack using cosine similarity method where before cosine we must do weighting term first. That's do for counting ter weight on sentence where the term is result tokens preprocessing did before. Weghting term is do using tf-idf methode where this method counting frequency term show in document (tf) and counting sum of document have term is search from the document collection / the formula for tf idf is

$$w_{i,j} = \frac{tf_{i,j} \cdot \log_2 \frac{N}{n_j}}{\sqrt{\sum_{s=1}^{k}(tf_{i,s} \cdot \log_2 \frac{N}{n_j})^2}} \qquad (2.26)$$

Where tf is termm frequency, N is total of document and n is sum document where tern is show.\

The nex step is counting using cosine similarity method the formula is :

$sim(K1, Ki) = \frac{\sum(K1.Ki)}{\sqrt{\sum K1^2}\sqrt{\sum Ki^2}}$ (2.27).

this procces will have result the summary has stacked.

2.7 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 test of summary will be do with counting performation recall, precission, f-measure and acuracy. Where the formula is :

$recall = \frac{tp}{tp+fn}$

$precission = \frac{tp}{tp+fp}$

$f\text{-}measure = \frac{2*\text{precission}*\text{recall}}{recall+precission}$

$accuracy = \frac{tp+tn}{tp+fp+fn+tn}$

where :

tp : setences total result by system and have on manual sentences.

fp : sentences total only have on system but no one one manual.

fn : sentences sum only have on manual but no one on system.

tn : sentences sum is nnot include on manual summary minus fp.

on testing proses used data :

Proses	total news article	Total Importa n sentence	Total setence not importa n
training	4	16	28
testing	7	23	75

the result is :

Table 1 result sigma testing (4 training data)

Sigma	tp	fp	fn	tn	*R*	*P*	*F*	*A*
0.1	6	10	17	65	0.26	0.37	0.31	0.72
0.2	9	16	14	59	0.39	0.36	0.38	0.69
0.3	**20**	**12**	**12**	**55**	**0.48**	**0.35**	**0.41**	**0.67**
0.4	10	20	13	55	0.43	0.33	0.37	0.66
0.5	10	22	13	53	0.43	0.31	0.36	0.64
0.6	9	22	14	53	0.39	0.29	0.33	0.63
0.7	9	22	14	52	0.39	0.29	0.33	0.63
0.8	7	22	16	53	0.30	0.24	0.27	0.61
0.9	7	21	16	54	0.30	0.25	0.27	0.62

where :
R = *Recall*
P = *Precission*
F = *F-measure*
A = *Akurasi*

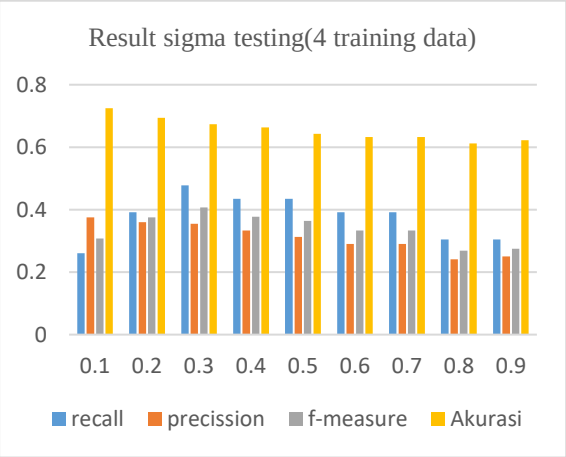


Figure 14 Result sigma training (4 training data)
from that result have best sigma is 0.3 and then the next testing is testing for tesing data:

Tabel 2 Result Testing Tata (4 data latih)

Banyak Data latih	tp	fp	fn	tn	R	P	F	A
1	2	0	21	75	0.09	1	0.16	0.79
2	3	8	20	67	0.13	0.27	0.18	0.71
3	1	1	22	74	0.04	0.5	0.08	0.77
4	**20**	**20**	**12**	**55**	**0.48**	**0.35**	**0.41**	**0.67**
5	8	13	15	62	0.35	0.38	0.36	0.71
6	10	19	13	56	0.43	0.34	0.38	0.67
7	7	13	16	62	0.30	0.35	0.33	0.70

Dimana :
R = *Recall*
P = *Precission*
F = *F-measure*
A = *Akurasi*

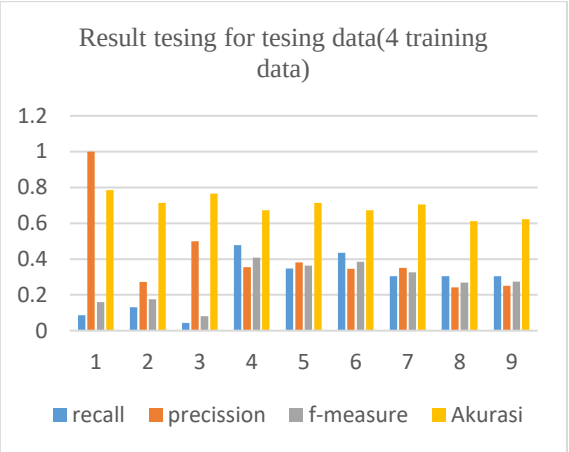


Figure 15 Result tesing for tesing data (4 training data)
the resut is *recall* 0.48, *precission* 0.35, *f-measure* 0.41 and accuracy 0.67.

2 CLOSING

Te conclussion from this reseach is that the Relevance Vector Machine method can be implemented in the case of automatic teks summarization for multi documents. And n the accuracy th summary testing with intrinsik method for counting recall, preccission, f-measure and accuracy.the result from intrinsik method is average recall, precission, f-measure 41% and accuracy 67%.

Kesimpulan yang didapat dari penelitian ini adalah bahwa metode RVM (*Relevance Vector Machine)* dapat digunakan pada peringkasan multi dokumen. Adapun dari sisi akurasinya, sistem peringkasan diui menggunakan metode intrinsik, untuk pencarian nilai *recall, precission, f-measure* dan akurasi-nya. Dari pengujian instrinsik tersebut didapatkan rata-rata *recall, precission dan f-measure* sebesar 41% dengan akurasi sebesar 67%.

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

1. Make auto NER (*Named Entity Recognizer*) for entity detection.
2. base on result f-measure is not enough good. Then need other methode with same fitur extraction..

REFERENCES

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

[2] A. Riyanto, Peringkasan teks menggunakan metode K-means, Bandung: unikom, 2016.

[3] M. L. K. B. R. T. Deni Fitriaman#1, "Peringkasan Teks Otomatis Berita Berbahasa Indonesia Pada Multi-Document Menggunakan Metode Support Vector Machines (SVM)," *Peringkasan Teks otomatis.*

[4] C. M. Bishop, Variational Relevance Vector Machines, 7 J. J. Thompson Avenue, Cambridge CB3 0FB, U.K: http://research.microsoft.com/{~cmbishop,~mtipping}, 2000.

[5] A. T. W. 2. G. S. 3. Zulkifli 1, "Pembobotan Fitur Ekstraksi Pada Peringkasan Teks Bahasa Indonesia Menggunakan Algoritma Genetika," *Pembobotan Fitur Ekstraksi Pada Peringkasan Teks Bahasa Indonesia Menggunakan Algoritma Genetika*, pp. 2-3.

[6] M. E. Tipping, "Sparse Bayesian Learning and the Relevance Vector Machine," *Sparse Bayesian Learning and the Relevance Vector Machine*, 2001.

[7] R. Hidaya, "implementasi Support Vector Machines Untuk Pencarian Informasi Buku Di perpustakaan Daerah Bandug Provinsi Jawa Barat".

[8] M. Mustaqhfiri, Z. Abidin dan R. Kusumawati, "Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance," *Matics Matics*, vol. 4, no. 4, pp. 134-147, 2011.

[9] Andita Dwiyoga Tahitoe - Diana Purwitasari, "IMPLEMENTASI MODIFIKASI ENHANCED CONFIX STRIPPING STEMMER UNTUK BAHASA INDONESIA DENGAN METODE CORPUS BASED STEMMING," *IMPLEMENTASI MODIFIKASI ENHANCED CONFIX STRIPPING STEMMER UNTUK BAHASA INDONESIA DENGAN METODE CORPUS BASED STEMMING* .

[10] A. budiman, Peringkasan Teks Otomatis menggunakan SVM, Bandung: UNIKOM, 2016.

[11] Jogianto, Analisis & Desain, Yogyakarta: Andi.

[12] A.-B. B. LADJAMUDIN, Analisis dan Desain Sistem Informasi, Yogyakarta: GRAHA ILMU, 2005.

[13] Z. A. d. R. K. Muchammad Mustaqhfiri1, "Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance," *Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance* .

[14] A. D. T. -. D. Purwitasari, "IMPLEMENTASI MODIFIKASI ENHANCED CONFIX STRIPPING STEMMER UNTUK BAHASA INDONESIA DENGAN METODE CORPUS BASED STEMMING," *IMPLEMENTASI MODIFIKASI ENHANCED CONFIX STRIPPING STEMMER UNTUK BAHASA INDONESIA DENGAN METODE CORPUS BASED STEMMING* .

[15] P. Roger S. Pressman, Software Engineering A Practitioner' s Approach, Buston: Mc Graw Hill, 2001.

[16] I. Sommerville, Sofware engineriing edisi 6 Jilid 1(Rekayasa Perangkat Lunak).

[17] .. P. D. Sugiyono, Metode Penelitian Kombinasi, 2014.

[18] F. Z. Tala, "A Study of Steming Effect on Information Retrieval in Bahasa Indonesia".