Scanning of Thesis Script Similarity with Vector Space Model

Sri Winiarti
Department of Informatics Engineering
Universitas Ahmad Dahlan

Ulaya Ahdiani
Department of English Literature
Universitas Ahmad Dahlan

Romakh Fitriani
Department of Informatics Engineering
Universitas Ahmad Dahlan

ABSTRACT
The rapid growth of online textual data has increased the need for information retrieval (IR) methods that is time efficient. Text classification is the process of finding the category of a document based on its content. However, few discuss text classification using cascading texts. In general, text classification uses the Vector Space Model (VSM) proposed by Salton, Wong, and Yang (1975) as a model for document representation and queries. One of the limitations of VSM is the problem of space, because each document must be represented using all the words in the dictionary (i.e. vocabulary). With the convenience provided by search engines to assist users in searching for information online, the internet is the dominant data and information center. No exception for students, the level of internet use in finding references is very high.

Statistics released by the Indonesian Internet Service Providers Association (APJII) in 2019 stated that Indonesian internet users had reached 171.17 million people or around 64.8%. Young people with an age range of 15-34 years dominate the number of users up to 49.52%. There is possibilities of similarities in publication, due to the large number of scientific publications that are published each year. The highest level of similarity in the thesis text is in the title and theoretical study. In searching for references for theoretical studies, students tend to plagiarize on a scientific work by copying part or even the entire content without mentioning the original source of the scientific work. Therefore, this research aims to create a document similarity detection system using the Vector Space Models (SVM) method. The data sets used to detect the similarities were 443 undergraduate thesis titles and 442 studies on the theory of thesis texts. From the accuracy test carried out on 132 queries from 321 thesis texts, it was obtained a mean average precision of 0.996.

Keywords
Similarity, Vector Space Models, Thesis script, Title, Theoretical Framework

1. INTRODUCTION
Various models that are used for scanning documents or information use the Vector Space Model in searching for information so that the similarity analysis process can be done automatically [1] [2]. Vector Space Model (VSM) is a textual representation method that is widely used in classifying documents. However, it was found that the problem was limited space [3]. SVM-related studies are used to detect documents in common use because of their level of accuracy. Arabic Text Classification (Fawaz S. Al-Anzi and Dia AbuZeina, 2018) [3], a system recommender for predicting similarity, Zang Zoa, et all, 2020) [4], applying SVM in analyzing online social media users (Sarkar, D and Jana, P, 2019) [5].

In their research, Jiang et al. (2015) proposed a method to improve the application of VSM to determine document similarity, which can improve document retrieval performance [6]. It can be seen from the research on the literature above that VSM can convert text information into data that can be recognized by computers, and perform similar feature extraction and text retrieval on text data. To extract feature words, TF-IDF Technique is used to calculate the weight of each feature word, TF (term frequency) represents the frequency of words or phrases in the text, and IDF (Inverse Document Frequency) shows the frequency of the document reversed. By calculating the TF-IDF value Request text, request text can be converted into numeric values, which is conducive to computer recognition [3] [5].

Based on the previous researches, a research was conducted to develop a system that can identify document similarities. The documents used as samples in this study are thesis manuscripts, with variable data in the form of thesis title and theoretical framework. The reason for choosing this variable is because the level of similarity of the two variables is very high, more than 50% is found in the thesis script. By applying the VSM method to represent text and select feature items, and feature words taken from the selected text based on quantified weights to represent text information, this can aid in expression selection and weighting, which can improve the accuracy and classification results of documents. The weighting method in this study also applies the TF-IDF. weighting schemes can be used to analyze and find relationships between objects (such as photos, documents, submissions, etc.) [5]. In this research, several processes were carried out, namely; a process to scan, analyze, and classify the title and theoretical framework of the thesis script to detect and / or predict the level of thesis similarity.

2. LITERARY REVIEW

2.1 Popular Algorithms in Document Classification
Scanning is a process of collecting and using information from the organization's environment to assist management in making decisions within an organization [7]. Aguilar was the first to develop a categorization search method for organizations by scanning the environment [8]. Furthermore, Choo [9] conducted a review by conducting a literature review, in his research concluded several things related to information retrieval, namely: (1) the dimensions of the situation, influenced by indicators: the effect of uncertainty on the surrounding environment, organizational strategy and scanning strategy, managerial characteristics: unanswered questions, (2) information needs: environmental scanning

focus, (3) information retrieval: resource use and preferences, (4) information retrieval: scanning methods, (5) information use: strategic planning and organizational learning improvement. Referring to the research of Auster and Choo [7,9], this paper discusses research that uses information searches carried out in two categories of information retrieval, namely the use of information sources and information retrieval by scanning methods with text documents, namely student thesis texts. Scanning is done is to check the similarity of the thesis script based on the title and theoretical framework contained there in.

Text classification is a method for categorizing individual documents in a document set based on a predefined set of categories. Algorithms that are widely applied and reviewed for classifying text include, Decision Tree [9], Naïve Bayes [10], KNN [11] and Support Vector Model (SVM) [12].

In this paper, a process developed for scanning in search of document similarity uses VSM to represent documents according to the most popular text categorization algorithm [13]. In the text classification process, the first thing to do is determine how to represent the document. One solution to the document representation problem, namely VSM, was introduced by Salton [15]. VSM represents documents and user queries with vectors, performs quantitative calculations to represent documents. The document is converted to a vector by converting each word in the document to a similar word stem using the frequency of each word stem, or other similarity measure in the document as a similar element in the vector used to represent the document.

2.2. Basic Concepts of Vector Space Model

The Vector Space Model is a standard and well-known method for vectorizing information retrieval introduced by Salton, Wong and Yang [15]. The aim of VSM is to convert this textual document into a feature vector [16].

D is defined as a set of specific documents and each document is a set of words. VSM is commonly used in information filtering, System Information Retention, indexing, and relevance ranking. The basic idea of the VSM method is to represent each independent word and each document is represented in a vector so that the complexity of the relationship of words becomes simple and can be calculated. In VSM, each document consists of terms (T1, T2, ..., Tn) and each term Ti has a weight of Wi. Term (T1, T2, ..., Tn) is considered as one of the vector elements in the N-dimensional coordinate system.

The i document with the vector di can be represented as follows:

$$d_i = (w_{1,i}, w_{2,i}, \ldots, w_{n,i}) \qquad (1)$$

where wj, i represents the weight of the word j that appears in the document i and N is the number of words used for vectorization. To calculate wi, j, we can set it to 1 if the word j is in document i and 0 otherwise. The number of times the word j is observed in the document i can also be recorded.

2.3. Weighted Term

The more common scheme is called term frequenct - inverse document frequency (TF-IDF) where wj, i is calculated by equation 2. [3] [5]

$$W_{ij} = TF_{ji} x IDF_t \qquad (2)$$

Term Frequency is a method for calculating the weight of each term in the text. In this method, each term is assumed to have an important value that is proportional to the number of occurrences of the term in the text. The weight of a term t in text d as written in equation 2.

Note:

Wj, i = weight of t (term) in one document

TFj, i = frequency of occurrence of t (term) in document d

IDFt = Inverse document frequency

Term frequency can improve the recall value of information retrieval, but not always correct the precision value. This is due to frequent terms that tend to appear in many texts, so that these terms have little discriminatory / uniqueness power. To correct this problem, terms with high frequency values should be removed from the term set. If term frequency focuses on the appearance of terms in a text, Inverse Document Frequency (IDF) focuses on the appearance of terms in the entire text collection. In the IDF, terms that rarely appear in the entire collection of terms are considered more valuable. The value of importance of each term is assumed to be inversely proportional to the number of texts containing the term. The IDF value of a term t is formulated in equation 3 [3] [5].

$$idf_i = \frac{[D]}{[\{document \in Dj \in document\}]} \qquad (3)$$

Where tfj, i is the frequency of the word j in document i. idfi is the inverse frequency of the word j in all documents, which is the logarithm of the total number of documents divided by the number of documents containing the word j. TF-IDF assigns higher weight to rarer words throughout the document and at the same time has a higher frequency in the documents in which they are used. Also common in all documents is that they are given less weight.

Note:

D = sum of all documents

j = number of documents containing term t

The IDF reflects the spread of terms t in the entire document so that it can show the differences in terms t in each document. TF reflects the spread of terms t in a document. TF-IDF can make exceptions for words that are high frequency but have little in common, so TF-IDF is an effective algorithm for calculating the weight of term t.

After weighting each term, a calculation is needed to perform a ranking to measure the similarity between the query vector and the document vector to be compared. One of the methods commonly used in the similarity calculation is the cosine measurement, which determines the angle between the document vector and the query vector and is defined in equation 4 [17].

$$Similarity = \emptyset(a,b) = cos(\varphi) = \frac{a.b}{\|a\|.\|b\|} = \frac{\sum_{j=1}^{n} a_i.b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}.\sqrt{\sum_{i=1}^{n} b_1^2}} \qquad (4)$$

The normalized point product shows the similarity between two objects, vectors a and b, the denominator in this equation is called the normalization factor which functions to eliminate the effect of document length. This normalization is necessary because long documents tend to have a greater value because they have a large frequency of occurrence of words. [17]. The ranking process of documents can be considered as the process of selecting (vector) documents that are close to the (vector) query, this closeness is indicated by the angle formed.

The cosine value which tends to be large indicates that the document is getting the same as the query. Cosine value equal to 1 indicates that the document is in accordance with query [17].

2.4. Calculating Precision Call Value

Precision is defined as the ratio of true positives (TP) and the number of positives predicted by the model. It is defined with reference to a special case of confusion matrix, with two classes; one designated a positive class, and the other a negative class, as presented in Table 1. To calculate the precision, it can be done in two stages, namely:

1. Calculating Precision

Used to find out how relevant the recommended document is, there are several steps. Precision can then be defined in terms of true positive and false positive (FP) as follows.

$$Precision = \frac{TP}{(TP + FP)} \qquad (5)$$

2. Calculate average precision

Average precision is to find the average value of the jump of relevant precision. The following is the average precision equation shown in equation 6.

$$AP = \frac{1}{[R]} \sum_{i=1}^{n} Precision\ (i) * Relevant\ (i) \qquad (6)$$

Table 1. The outcomes of classification into positive and negative classes

		Assigned Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

After the precision value is obtained, then calculate the mean average precision (MAP) value which is the average value of average precision. Average precision is the value obtained from each value of the resulting relevant precision on item and uses a value of 0 for relevant items not generated by the system. Equation 7 is an equation for calculating the mean average precision in the information retrieval:

$$MAP(Q) = \frac{1}{[Q]} \sum_{j=1}^{[Q]} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \qquad (7)$$

Notes:

Q = Number of test queries
R = relevant items produced by the system
M = The number of relevant items the query returns

3. THE PROPOSED METHOD

3.1 Collecting Data Set

In this study, the data set used two different types of data, namely 443 thesis title data and 442 theoretical framework data from a total of 885 English literature students' thesis texts. The data were taken from the Universitas Ahmad Dahlan digilib libraries.

3.2. Pre-processing

Pre-processing is the process of changing text into basic words. At the text pre-processing stage, there are a number of steps that need to be taken to get the text free of noise or meaningless words. Apart from freeing from noise, text pre-processing also returns words to the root word or root word. The steps in text pre-processing go through several processes, namely; The Tokenizing Process, the Filtering Process, and the Stemming Process, so as to produce terms or words that are ready to be processed to the next stage, namely TF-IDF weighting [18]. Table 2 shows the results of the pre-processing stage.

Table 2. Pre-processing

Title	Text Pre-processing Results
HEGELLIAN PHILOSOPHY OF METAPHYSIC AS REFLECTED IN "THE PHOENIX AND THE TURTLE	hegellian philosophy metaphysic reflected phoenix turtle
BLACK WOMEN'S ROLE AS REFLECTED IN LORRAINE HANSBERRY'S A RAISIN IN THE SUN	black women role reflected lorraine hansberry raisin sun
THE SPEECH ACTS ANALYSIS ON THE SCRIPT NOTTING HILL (A PRAGMATIC STUDY)	speech acts analysis script notting hill pragmatic studi
MORAL VALUES AS REFLECTED IN THE FILM "THE SPONGEBOB SQUARE PANTS MOVIE"	moral values reflected film spongebob square pants movi
AN ANAYSIS ON JARGON LANGUAGE USED IN DAVID WILKERSON'S THE CROSS AND THE SWITCHBLADE: A SOCIOLINGUISTIC STUDY	anaysis jargon language used david wilkerson cross switchblade sociolinguistic studi
A CONVERSATIONAL IMPLICATURE ANALYSIS ON J.K.ROWLING'S HARRY POTTER AND THE CHAMBER OF SECRETS: A PRAGMATIC STUDY	conversational implicature analysis j k rowling harry potter chamber secrets pragmatic studi
LUKE CHANDLER’S MATURITY AS REFLECTED IN GRISHAM’S A PAINTED HOUSE	luke chandler rsquo maturity reflected grisham rsquo painted hous

In text processing, tokenization is the procedure of dividing text into words, phrases or other meaningful parts, namely tokens. In other words, tokenization is a form of text segmentation. Usually, segmentation is done by considering only alphabetic or alphanumeric characters separated by non-alphanumeric characters (e.g., punctuation, spaces). Stop words are words that are commonly encountered in text without dependence on a specific topic (e.g., Conjunctions, prepositions, articles, etc.). Therefore, stop words are usually considered irrelevant in the classification of the study text, and are deleted before classification. The results of the query after processing are shown in Table 3.

3.3. Calculating the TF-IDF Weight

By using equations 2 and 3, weighting is done by finding the weight of each term or word contained in the document.

3.4. Scanning Process with Cosine Similarity

Cosine similarity is the calculation of the similarity between documents and queries using equation 4. Next, checking the similarity results for analysis. The overall process of the Vector Space Model stages in this study is presented in Figure 1.

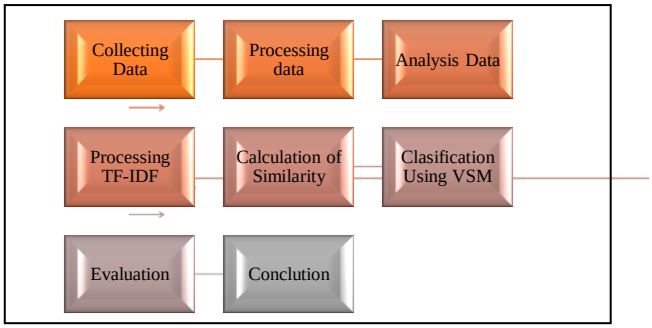

Figure 1: The stages of scanning the similarity of the thesis script with the VSM method

4. RESULT

4.1. TF-IDF Preprocessing and Weighting

In the Python programming language, the application of text pre-processing is carried out using the nltk, literary, and string libraries. The process starts by declaring the stemmer to call the StemmerFactory class and the remover calling the StopWordRemoverFactory class of the literary library. Translator is declared to translate English text. The next process is making stemmerEN and pre-processing functions. The StemmerEN function is used for the text preprocessing process in English, while the process function is for Indonesian. Text must be changed to lowercase, first before doing the process of tokenizing, filtering, and stemming to produce basic words or terms.

Table 3. Preprocessing Query

Query	Text Processing Result
Hegemony in Herman Melville's Bartleby: A Sociology Approach	hegemony herman melville bartleby sociology approach
Feminism values in freeman The Revolt of "Mother" and Steinbeck's The Chrysanthemums	feminism values freeman revolt mother Steinbeck
Politeness Strategies used by the Main Character of Cartoon Movie Shark Tale: Pragmatic Study	politeness strategies used main character cartoon movie shark tale pragmatic studi
Suicide Problem as Reflected by William Loman the Main Character of Arthur Miller's Death of A Salesman	suicide problem reflected william loman main character arthur miller death salesman
The Stylistic Features Found in Andrew Marvell's The Garden	stylistic features found andrew marvell garden
The Portrait of American Dreams as seen in the film The Pursuit of Happiness	portrait american dreams seen film pursuit happi
The Religiosity Aspect of the Poet in His Work based on The World is too much With Us and London, 1802 by William Wordsworth: Hermeneutics Analysis	religiosity aspect poet work based world much us london 1802 william wordsworth hermeneutics analysi

The implementation of text preprocessin in the Python programming language begins with importing the Python library used, namely nltk, literature, and strings. To further declare the stemmer to call the StemmerFactory class in the literary library, declare a remover to call the StopWordFactory class in the literary library, declare a translator to translate the English language test, define a stemmerEN function to perform text preprocessing processes in English text, define a preprocess function to process text preprcessing of Indonesian text. For details, see the following program code:

```
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from
Sastrawi.StopWordRemover.StopWordRemoverFactory
import StopWordRemoverFactory
from     Sastrawi.Stemmer.StemmerFactory     import
StemmerFactory
import string
stemmer = StemmerFactory().create_stemmer()
remover                                          =
StopWordRemoverFactory().create_stop_word_remover()
translator      =      str.maketrans('',      '',
string.punctuation)
def stemmerEN(text):
porter = PorterStemmer()
    stop = set(stopwords.words('english'))
    text = text.lower()
    text = [i for i in text.lower().split() if i
not in stop]
    text = ' '.join(text)
    preprocessed_text = text.translate(translator)
    text_stem = porter.stem(preprocessed_text)
    return text_stem
def preprocess(text):
  text = text.lower()
text_clean = remover.remove(text) #fungsi hapus
stopword
  text_stem = stemmer.stem(text_clean)
    text_stem = stemmerEN(text_stem)
   return text_stem
```

In this piece of the program, the TF-IDF process begins with the import process of the libraries used are numpy and scikit-learn and create an Engine class to process TF-IDF weighting and calculation of cosine similarity. The functions used in the

Implementation of TF-IDF weighting and cosine similarity is carried out using the numpy and scikit-learn libraries. The first process defines training data and testing data. The second process makes a function for TF-IDF weighting and cosine similarity. The cosine similarity value obtained is stored in the form of an array list. The implementation of TF-IDF weighting and cosine similarity is shown in the program code as follows:

```
import numpy as np
import numpy.linalg as LA
from openpyxl import load_workbook
from sklearn.feature_extraction.text
importCountVectorizer
from sklearn.feature_extraction.text import
TfidfTransformer, TfidfVectorizer
Class Engine:
def _init__(self):
     self.cosine_score =[]
     self.train_set =[]  # Documents
self.test_set = [] #Query
def addDocument(self, word):
self.train_set.append(word)
def setQuery(self, word):
self.test_set.append(word)
def process_score(self):
   stopWords = stopwords.words('english')
vectorizer = CountVectorizer()
transformer = TfidfTransformer()
trainVectorizerArray=vectorizer.fit_transform(self.tr
ain_set).toarray()
testVectorizerArray=vectorizer.transform
(self.test_set). toarray()
cx = lamda a, b: round (np.inner (a, b )/LA.norm(a) *
LA.norm (b)),3)
output=[ ]
for i in range (0, len (testVectorizerArray)):
output. Appaned([ ])
for vector in trainVectorizerArray:
u = 0
for testV in testVectorizerArray:
cosine = cx (vector, testV)
output [u].apened (( cosine))
u = u + 1 return output
```

TF-IDF calculation process in Python programming, are namely;

a. to create a function to declare training and testing data

b. to create a function to add a dataset document to the train set list

c. to create a function to add query data to the test set list

d. to create functions for the calculation process

Next, it is necessary to declare a vectorizer to call the Count Vectorizer class in the scikit-learn library, declare a transformer to call the TF-IDF transformer class in the scikit-learn library, perform the document weight calculation process on the dataset then tested and stored in an array form. To perform the query weight calculation process using syntax `testVectorizerArray=vectorizer.transformself.test_set.toarray`, then tested and stored in the form of an array. The process of calculating the value of the cosine in vector form between the document and the query and it is stored and then returned the cosine value to the output variable in the form of a list.

The results of weighting examples from data that have been carried out by the TF-IDF process are presented in Table 4.

Table 4. TF-IDF Weighting

Term	Tf					N/df_t	Idf	Weight (w)				
	Q	D_1	D_2	D_3	D_4			Q	D_1	D_2	D_3	D_4
Code	1	1	1	0	0	4/2=2	0.301	0.301	0.301	0.301	0	0
Mix	1	1	1	0	0	2	0.301	0.301	0.301	0.301	0	0
Analysi	1	1	0	0	1	2	0.301	0.301	0.301	0	0	0.301
sociolinguist	0	1	1	1	0	1.33	0.125	0	0.125	0.125	0.125	0
Use	0	1	1	1	1	1	0	0	0	0	0	0
indonesian	0	1	0	0	0	4	0.602	0	0.602	0	0	0
Vlogger	0	1	0	0	0	1	0.602	0	0.602	0	0	0
approach	0	1	0	0	0	4	0.602	0	0.602	0	0	0
Suhay	0	0	1	0	0	4	0.602	0	0	0.602	0	0
Salim	0	0	1	0	0	4	0.602	0	0	0.602	0	0
Vlog	0	0	1	0	0	1	0.602	0	0	0	0.602	0
Studi	0	0	1	1	0	2	0.301	0	0	0.301	0.301	0
Hip	0	0	0	1	0	4	0.602	0	0	0	0.602	0
Hop	0	0	0	1	0	4	0.602	0	0	0	0.602	0
Slang	0	0	0	1	0	4	0.602	0	0	0	0.602	0
Wiz	0	0	0	1	0	4	0.602	0	0	0	0.602	0
khalifa	0	0	0	1	0	4	0.602	0	0	0	0.602	0
Song	0	0	0	1	0	4	0.602	0	0	0	0.602	0
Lyric	0	0	0	1	0	4	0.602	0	0	0	0.602	0
languag	0	0	0	0	1	4	0.602	0	0	0	0	0.602
deviat	0	0	0	0	1	4	0.602	0	0	0	0	0.602
John	0	0	0	0	1	4	0.602	0	0	0	0	0.602
steinbeck	0	0	0	0	1	4	0.602	0	0	0	0	0.602
Grape	0	0	0	0	1	4	0.602	0	0	0	0	0.602
Stylist	0	0	0	0	1	4	0.602	0	0	0	0	0.602

Note:
Term : basic word
TF : the frequency with which the terms appear in the document
D : document
Q : query
DF : the frequency of documents containing the term
IDF : result (log N/DF)
N : lots of documents
W : Weight

4.2. The scanning process using Cosine Similarity

Cosine similarity is the calculation of the similarity between documents and queries. First, this is done with vector multiplication or scalar multiplication between weight terms in each document. Next, after the vector length product is obtained, is to calculate the square value of each term. The calculation results are shown in Table 5.

Table 5. Results on Documents and Queries

Term	Weight (W)			
	Q	D1	D2	D3
Struggle	1	1	1	1
American	1	1	1	1
Vietnam	0	0.227	0	0
Veteran	0	0.227	0	0
Towards	0	0.227	0	0
Their	0	0.227	0	0
Existences	0	0.227	0	0
Society	0	0.030	0.030	0
Film	0	0	0	0
Main	0	0	0.030	0.030
Character	0	0	0.030	0.030
Tariq	0	0	0.227	0
Mahdi	0	0	0.227	0
Moslem	0	0	0.227	0
Living	0	0	0.227	0
Among	0	0	0.227	0
Modern	0	0	0.227	0
Moozlum	0	0	0.227	0
Josep	0	0	0	0.227
Donelly	0	0	0	0.227
Pursu	0	0	0	0.227
Dream	0	0	0	0.227
Far	0	0	0	0.227
Away	0	0	0	0.227
Sum	**2**	**3.846**	**3.687**	**3.422**

4.3 Precision Measurement and Testing

Measurements are made to test the precision of the return recommendations. Each recommendation determination is checked for the relevance of the recommendation to the query. Each recommendation is also labeled. The label value is determined by the cosine similarity score. Relevant recommendations are labeled with the number 1, while those that are irrelevant are assigned the number 0. Recommendations with a value of 0 to 0.30 will be considered irrelevant, whereas if they are 0.31 to 0.60, they will be caught quite relevant, whereas if they are valued at 0.61 until 1 catch is relevant. These labels are used to calculate or measure the precision value from the recommendation data taken. Calculation of the precision value is performed on each input query. From these results, then calculate the average precision value. Then the average precision score for each query is added and divided by the number of queries to get the mean average precision (MAP) value. Tests are carried out using calculation formulas such as the MAP equation available:

$$MAP(Q) = \frac{1}{[Q]} \sum_{j=1}^{[Q]} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \qquad (7)$$

The tests carried out involved 130 test data and 321 data sets, in the 130-test data from queries 62 to queries 281 which were taken randomly for testing the query. From the test results with 321 data sets, data 130 test, with the result 0.99 which means that the data is relevant, because the relevant value limit is almost close to point 1.

In this study, it is concluded that the documents tested belong to very similar classifications. Table 4 shows the results of weighting using TF-IDF. Term is the root word of the data in the form of a thesis title which has been weighted based on the frequency of occurrence of the term. The test results are presented in Table 8.

After getting the total value of the documents and queries where the data has been squared, then the process of calculating the length of the vector is next. The calculation results are shown in Table 6.

Table 6. The resulting vector length

Vector Length	Q	D1	D2	D3
	1.414	1.961	1.920	1.849

The last step after getting the result of the vector length is calculating the cosine similarity. Calculation of cosine is done with equations Here's the calculation process:

$$\cos(Q, D1) = \frac{2}{1.414 + 1.961} = 0{,}59$$

$$\cos(Q, D2) = \frac{2}{1.414 + 1.920} = 0{,}59$$

$$\cos(Q, D1) = \frac{2}{1.414 + 1.842} = 0{,}61$$

Based on the results of the calculation of cosine similarity, the next step is to make curling, by sorting the values from largest to smallest. The results are shown in Table 7.

Table 7. Cosine Similarity results

D	Similarity Value	Percentage
D1	0,592	59%
D2	0,599	60%
D3	0,614	61%

Measurements are made to test the precision of the return recommendations. Each recommendation determination is checked for the relevance of the recommendation to the query. Each recommendation will also be labeled. The label value is determined by the cosine similarity score. Relevant recommendations will be labeled with the number 1, while

those that are irrelevant will be assigned the number 0. Recommendations with a value of 0 to 0.30 will be considered irrelevant, whereas if they are 0.31 to 0.60 they will be caught quite relevant, whereas if they are valued at 0.61 to 1 caught relevant. The label will be used to calculate or measure the precision value from the recommendation data taken. The calculation of precision values is performed on 130 test data and 321 data sets for each input query. From these results, the average precision value will be calculated. Then the average precision score for each query will be added and divided by the number of queries to get the mean average precision (MAP) value. Tests were carried out using 130 queries and 321 data set documents using equation 7.

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Pecision(R_{jk}) \qquad (7)$$

$$MAP = \frac{1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+\cdots+1+1+1}{321}$$

$$= 0{,}9906$$

In this study it was 0.9906 or 100% when rounded off in percent form. This suggests that the recommendations given to this system fall into a very similar classification. The test sample can be seen in Table 8.

Table 8: Test Result Data Mean Average Precision

Q	No	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Relevant (1)	Irrelevant (0)	AP
queries	1	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	2	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	3	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	4	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	5	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	6	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	7	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	8	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	9	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	10	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	11	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	12	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	13	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	14	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	15	1	1	1	1	1	1	1	1	1	1	10	0	1
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
queries	309	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	310	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	311	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	312	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	313	1	1	1	1	1	1	1	1	1	0	9	1	1
queries	314	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	315	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	316	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	317	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	318	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	319	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	320	1	1	1	1	1	1	1	1	1	1	10	0	1
queries	321	1	1	1	1	1	1	1	1	1	1	10	0	1
Total														318
Mean Average Precision														0,990654

The results of the MAP test are graphically shown in Figure 2 and Figure 3 as a Query Test of 320 thesis title data sets. The explanation from table 8 is depicted in graphic form. The relevant data classification for documents that are stated to be similar is represented in yellow and the classification for not relevant is shown in green.

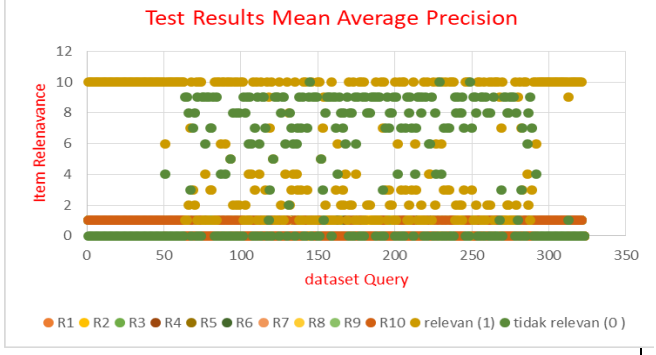

Figure 2: Test Result Mean Average Precision for Query

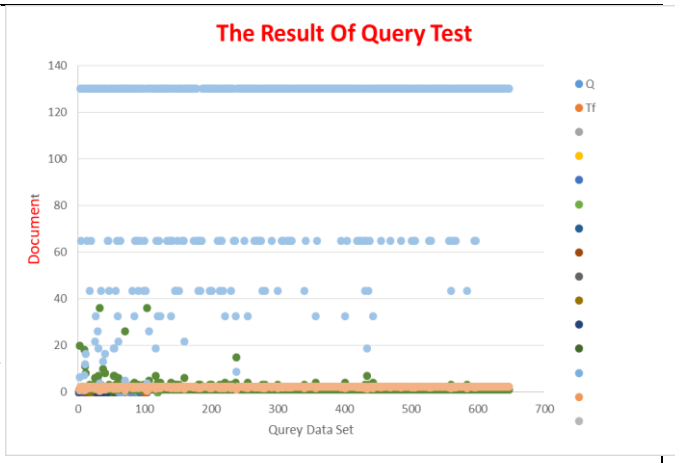

Figure 3: the results of testing the query dataset

5. CONCLUSION

Vector Model has been used to scan student thesis script based on the title and theoretical framework. From the total data set of documents in the form of 432 thesis titles and 431 theoretical frameworks, 131 data queries were tested with relevant precision results of 0.99 or in testing the documents were stated to be very similar. Table 4 shows the TF-IDF value of the 321 thesis title documents and Table 5 is the result of the Query performed. Tabular data can be processed using various data mining algorithms, such as KNN, C.45 Algorithm or Decision Tree to further analyze and visualize to find different accuracy so that the best performance can be obtained.

6. ACKNOWLEDGMENTS

Our sincere appreciation goes to Universitas Ahmad Dahlan for funding this research through the UAD Institute for Research and Community Service. Special gratitude also goes to the English Literature Study Program for providing the availability of data sets in the form of thesis titles and theoretical frameworks for students' undergraduate thesis.

7. REFERENCES

[1] Cheryl Aasheim, C. and Koehler, G.J 2005. Scanning World Wide Web documents with the vector space model. Journal Decision Support Systems 42 (2006) 690– 699, available online at ScienceDirect, Elsevier.com.

[2] Chao Ke, Zhigang Jiang, Hua Zhang, Yan Wang, Shuo Zhu.2020. An intelligent design for remanufacturing method based on vector space model and case-based reasoning. Journal of Cleaner Production 277 (2020) 123269. Available on Elsavier.com.

[3] Al-Anzi,F.S., and AbuZeina, D. 2018. Beyond vector space model for hierarchical Arabic text classification: A Markov chain approach. Journal Information Processing and Management, No 54 2018, pp 105-115. Available on Elsavier.com.

[4] Zhan Su, Xiliang Zheng, Jun Ai, Yuming Shen, Xuanxiong Zhang.2020, Link prediction in recommender systems based on vector similarity, Physica A 560 (2020) 125154, Available on Elsavier.com.

[5] Sarkar, D., Jana, P., 2019. Analyzing User Activities Using Vector Space Model in Online Social Networks arXiv preprint arXiv:1910.05691.

[6] Jiang, R., Kim, S., Banchs, R.E., Li, H., 2015. Towards Improving the Performance of Vector Space Model for Chinese Frequently Asked Question Answering, 2015 International Conference on Asian Language Processing (IALP). IEEE pp. 136e139.

[7] Auster, E and Choo, C.W. 1994. How Senior Managers Acquire and Use Information In Environmental Scanning Journal of Information Processing & Management, Vol. 30, No. 5, pp. 607-618, 1994.

[8] F.J. Aguilar, Scanning the Business Environment, Macmillan, New York, 1967.

[9] C.W. Choo, Information Management for the Intelligent Organization: the Art of Scanning the Environment, Information Today, Inc., Medford, 2002.

[10] Fazayeli,H., Syed-Mohamad, S.S., Md Akhir, , N.S. 2019. Towards Auto-labelling Issue Reports for Pull-Based Software Development using Text Mining Approach, Proceeding: The Fifth Information Systems International Conference 2019, Elsavier.

[11] Rasjid, S.E. and Setiawan, R. 2017. Performance Comparison and Optimization of Text Document Classification using k-NNand Naïve Bayes Classification Techniques, Proceeding 2nd International Conference on Computer Science and Computational Intelligence 2017, ICCSCI 2017, 13- 14 October 2017, Bali. Available on Elsevier.

[12] Sevindik, T. and Cömert, Z. 2010. Using algorithms for evaluation in web based distance education, Procedia Social and Behavioral Sciences 9 (2010) 1777–1780. Available on Elsevier.

[13] H. Robothama, J. Castillo, P. Boscha, J. Perez-Kallens. 2011. A comparison of multi-class support vector machine and classification tree methods for hydroacoustic classification of fish-schools in Chile, Journal of Fisheries Research 111 (2011) pp. 170-176. Available on Elsevier.

[14] Aasheim, C, and Koehler, Gary J.2005. Scanning World Wide Web documents with the vector space model. Journal of Decision Support Systems 42 (2006) 690–699. Available on Elsevier.

[15] G. Salton, Automatic Information Organization and Retrieval, McGraw-Hill Book Company, New York, 1968.

[16] G. Salton, A. Wong and C.S. Yang, A vector space model for automatic indexing, Communications of the ACM 18 (1975), no. 11, 613 – 620.

[17] R. Zafarani, M. A. Abbasi, and H. Liu, Social Media Mining an Introduction, Cambridge University Press, 2014.

[18] F. Trevor Rogers. 2020. Patent text similarity and cross-cultural venture-backed innovation, Journal of Behavioral and Experimental Finance 26 (2020) 100319. Availavle on Elsevier.