Article classification using natural language processing and machine learning

Tran Thanh Dien
PhD student of Information Systems
Can Tho University
Can Tho city, Vietnam
thanhdien@ctu.edu.vn

Bui Huu Loc
MobiFone Company
(Can Tho Branch)
Can Tho city, Vietnam
loc.bui.ctv@mobifone.vn

Nguyen Thai-Nghe
College of ICT
Can Tho University
Can Tho city, Vietnam
ntnghe@cit.ctu.edu.vn

***Abstract*- Text classification is an important task which may help human reducing time and effort. This work is aimed to propose an approach for text classification, especially for articles. The proposed method can automatically extract information and categorize articles on suitable topics. The input data were pre-processed, extracted, vectorized and classified using machine learning techniques including Support Vector Machines, Naïve Bayes, and k-Nearest Neighbors. The experiments were carried out on two data sets of articles showed that with the accuracy of over 91%, using natural language processing and support vector machines technique proved its feasibility for developing the automatic classification system of articles.**

Keywords- Article classification, information extraction, natural language processing, support vector machines.

I. INTRODUCTION

With the explosive development of information and the simultaneous development of automatic computing, data classification, especially text classification, has been found to be particularly important [1]. Text classification is a supervised learning technique, (e.g. automatic metadata extraction), widely used in reality [2]. In machine learning and natural language processing (NLP), text classification is a classical text processing problem, aimed at assigning a new text into a group of given texts based on its similarities to the group [3]. According to [23], text classification is the assignment of labels on a new text based on the text's similarities to the labeled texts in the training set. Automated text classification makes storing and searching information faster. In addition, with a large number of texts, sorting each of them takes a lot of time and efforts, not to mention the possibility of inaccurate categorization due to people's subjectivity. There are several applications of text classification such as spam email filtering, news classification by topics in online newspapers, knowledge management and support for search tools on the Internet, etc. [1].

Another important text classification application is article classification. An issue in this area is concerned by both the authors and the editorial board of magazines/journals is how to classify a submitted manuscript into a relevant topic of the magazine/journal. For example, the submission system can automatically classify texts, extract relevant information when the manuscripts are uploaded. For the magazine/journal having large numbers of topics such as Association for Computing Machinery (ACM) with more than 2,000 topics, the author may take a lot of time to determine the topic(s) of the article. Therefore, automatic classification of an article's topic is necessary.

The issue of text classification is concerned by many scientists with different approaches. In which, machine learning is commonly used, with many algorithms such as k-nearest neighbors (kNN), Naïve Bayes, support vector machines (SVM), decision tree, and artificial neural network… that solve text classification problem ([4], [5], [8], [13], [14], [15], [16], [24], [25], [26])

This study proposes an approach of automated classification of articles submitted via an online submission system. When the file of the manuscript (e.g. *.doc(x), *.pdf) is submitted, the system will extract the author's information, title, and abstract, especially determine the article's topic (e.g. information technology, environment, fisheries, etc.).

II. REVIEW OF LITERATURE AND RELATED STUDIES

A. Text classification

In the context of rapid development of digital information, text mining techniques play an important role in information and knowledge management, attracting the attention of researchers [1]. Automated text classification (or categorization) is the division of an input textual dataset into two or more categories, in which each text can belong to one or more categories. Text classification is carried out for the purpose of assigning a predefined label or class to a text. For instance, a new article posted on an online newspaper can be assigned one of the categories such as technology, sports, or entertainment; each article published in a journal can be automatically classified into the topics of information technology, environment, or fisheries, etc.

From a set of documents $D = \{d_1, ..., d_n\}$ called a training set, in which document d_i is labeled under c_j belonging to set of categories $C = \{c_1, ..., c_m\}$, classification model is determined for classifying any document d_k into an appropriate category of set C. In other words, the problem is to find function f:

$$f: D \times C \rightarrow Boolean$$

$$f(d,c) = \begin{cases} true, & if\ d\ belongs\ to\ class\ c \\ false, & if\ d\ does\ not\ belongs\ to\ class\ c \end{cases}$$

B. Text classification algorithms

There are many text classification algorithms. In this study, the algorithms used are kNN, Naïve Bayes, SVM algorithms, of which classification performances are evaluated effective.

1) K-nearest neighbor technique

kNN algorithm is used to classify an object based on the closest distance between the object and its nearest neighbors in the training set [20]. Accordingly, when a new document needs classifying, the algorithm will calculate the distances of all documents in the training set to this new document to develop the kNN set. To classify a class-unknown document x, firstly, the kNN classifier will calculate the distances from text x to all documents in the training set, thereby, determine

978-1-7281-4723-9/19/$31.00 ©2019 IEEE
DOI 10.1109/ACOMP.2019.00019

set *N(x, D, k)* consisting of *k* examples having the closest distance to *x*.

The kNN algorithm is described as follows:

1. Determining the value of parameter *k* (number of nearest neighbors).

2. Calculating the distances between the object with all examples in the training set (often using *Manhattan* or *Euclidean* distance).

3. Arranging the distances in ascending order and determining *k* elements having the smallest distance (*k* nearest neighbors).

4. Statistics on *k* neighbors' frequency of classes: the most common class is the class of the object.

2) Naïve Bayes technique

Naïve Bayes algorithm [17] is a common machine learning algorithm that is evaluated as one of the most effective methods of text classification by [16]. The typical idea of this approach is using conditional probability between words or phrases and topics to predict the topic probability of a classified text.

Naïve Bayes algorithm based on Bayes theorem that is presented as follows:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Using Bayes theorem, we can find the probability of *Y* happening, given that *X* has occurred. Here, *X* is the evidence and *Y* is the hypothesis.

The general idea of Naive Bayes:

1. Represent a document *X* as a set of (*w*, a frequency of *w*) pairs.

2. For each label *y*, build a probabilistic model P(X| Y = y) of documents in class *y*.

3. To classify, select label *y* which is most likely to generate *X*:

$$\hat{y} = \underset{y}{agrmax}\, P(X|y) * P(y)$$

We apply to articles classification:

- Data set vectorized $D = (d_1, d_2, \ldots d_n)$

- Set of classes $C = (C_1, C_2, \ldots, C_m)$

Then,

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

Where $P(x_k|\ C_i)$ is probability of the k^{th} attribute that has the value x_k given that X belongs to class *i*.

3) Support vector machine technique

SVM algorithm was first introduced by [11]. SVM is quite effective for solving problems with multi-dimensional data like text representation vectors. It is also considered the most accurate classifier for text classification problem [19] owing to fast and effective classification speed.

For this method, a given training set is represented in vector space, where each text is considered as a point in this space. This method is to find a hyperplane *h* that makes the best decision to divide the points on this space into two corresponding separate classes, called positive (+) and negative (-) classes. Then, the SVM classifier is a hyperplane that separates the positive class examples from the negative class examples with the largest difference. This difference, also known as the margin distance, is determined by the nearest hyperplane distance between the (+) example and the (-) example (Fig. 1). The larger this distance is, the more clearly divided the examples of the two classes are; this means that a good classification result is achieved. The objective of SVM algorithm is to find the maximum margin distance to obtain good classification results.

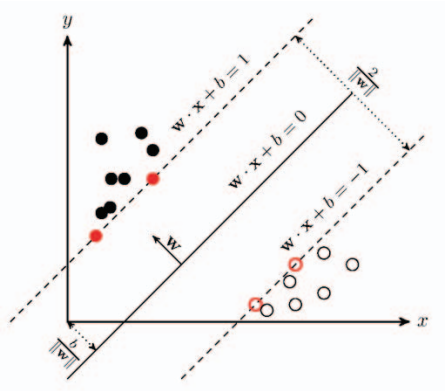

Fig. 1: Maximum hyperplane margin in two-dimensional space [11]

Hyperplane equation contains vector *x* in the object space as follows: $w.x + b = 0$, where, *w* is the weight vector, *b* is the bias. The direction and distance from the origin of the coordinate to the hyperplane change when *w* and *b* are changed.

SVM classifier is defined as follows: $f(x) = sign(w.x + b)$

Where, $\begin{cases} f(x) = +1,\ w.x + b \geq 0 \\ f(x) = -1,\ w.x + b < 0 \end{cases}$

Given y_i with +1 or -1 value. If $y_i = +1$, *x* belongs to class (+), whereas $y_i = -1$, *x* belongs to class (-). The two hyperplanes separate the examples into two parts described by the equations: $w.x + b = 1$ and $w.x + b = -1$

By geometry, it is possible to calculate the distance between these two hyperplanes: $\frac{2}{\|w\|}$

For the maximum margin distance, it is necessary to find the smallest value of $||w||$. At the same time, preventing data points from falling inside the boundary, the following constraints are needed:

$$\begin{cases} w.x_i + b \geq 1, & \text{with example } (+) \\ w.x_i + b \leq -1, & \text{with example } (-) \end{cases}$$

That can be rewritten as: $y_i(w.x_i + b) \geq 1,\ with\ i \in (1, n)$

Then, finding hyperplane *h* is equivalent to solving the problem of finding $Min||w||$ with *w* and *b* satisfying the following conditions: $\forall i \in (1, n)$: $y_i(w.x_i + b) \geq 1$.

From SVM for binary classification above mentioned, we can extend to apply SVM to the multi-class scenario.

4) Algorithm evaluation

Several common indicators are used to evaluate a machine learning algorithm. Supposing to evaluate a classifier temporarily called positive and negative, there are:

- *TP* (True positive) is the number of positive elements classified positive; *FN* (False negative) is the number of positive elements classified negative; *TN* (True negative) is the number of negative elements classified negative; and *FP* (False positive) is the number of negative elements classified positive.

- *Precision* is defined as the ratio of the number of *TP* elements to those classified positive (*TP* + *FP*):

$$Precision = \frac{TP}{TP + FP}$$

- *Recall* is defined as the ratio of the number of TP elements to the elements that are actually positive (TP + FN):

$$Recall = \frac{TP}{TP + FN}$$

High *precision* means that the accuracy of the found elements is high. High *recall* means that the rate of missing actually positive elements is low.

F_1 (*or F-score*) is the balance between *precision* and *recall*. If the *precision* and *recall* are high and balanced, the F_1 is high, whereas precision and recall are low and unbalanced, F_1 is low. Thus, the higher F_1 is, the better the classifier is. When both recall and precision are equal to 1 (best possible), $F_1 = 1$. When both recall and precision are low, for example, 0.1, $F_1 = 0.1$.

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

C. Related studies on text classification

Many studies on text classification have been applied to solve problems in practice. For example, [21] applied SVM and decision tree to solve text classification problem, and compared their effectiveness with that of classical decision tree algorithm. In addition, singular value decomposition technique was applied into SVM algorithm to shorten the dimension of characteristic space and reduce noise, making the classification process more effective. In the pre-processing phase, maximum matching segmentation (MMSEG) algorithm [9] was used to segment words. After segmentation, the text was modeled into a vector form, using the vectorized TF*IDF; then classified using two algorithms of SVM and decision trees in Weka software. With the dataset of 7,842 texts in 10 various topics, 500 texts of each topic were randomly chosen to train, the remaining texts were to verify independence. The results showed that the classification by SVM is really better than that by decision tree. Moreover, using singular value decomposition to analyze and shorten the dimension of characteristic space helped improve effectiveness of SVM classifier.

In the study on classification of Vietnamese documents with Naïve Bayes algorithm, [22] developed a word segmentation module from N-gram model, then modeled the segmented text by TF*IDF vector. After being modeled into vector, the dataset was classified using Naïve Bayes method. A classification software was developed, integrated with the functions of managing, editing and deleting articles to conduct experiments on the dataset of 281 scientific articles in the topic of information technology. The classification result was quite satisfactory; however, the study was still limited in the dataset, and there was no comparison of Naïve Bayes classifier with other classification methods.

Another study related semantic relation extraction and classification in scientific paper abstracts was proposed by [27]. The authors presented the setup and results of semantic relation extraction and classification in scientific papers. The task is divided into three subtasks: classification on clean data, classification on noisy data, and a combined extrac-tion and classification scenario. They also presented the dataset used for the challenge: a subset of abstracts of published papers in the ACL Anthology Reference Corpus, annotated for domain specificentities and semantic relations.

In this work, we propose an approach of automated classification of articles submitted via an online submission system. Accordingly, when an article (e.g. *.doc(x), *.pdf) is submitted, the system will extract the author's information, title, and abstract, especially classify the article's topic. In this approach, first, natural languge processing is used to pre-process the data, then machine learning techniques are used for topic classification.

III. PROPOSED METHOD

A. Classification system model

The proposed overall system of extracting information and classifying article is modeled in Fig. 2:

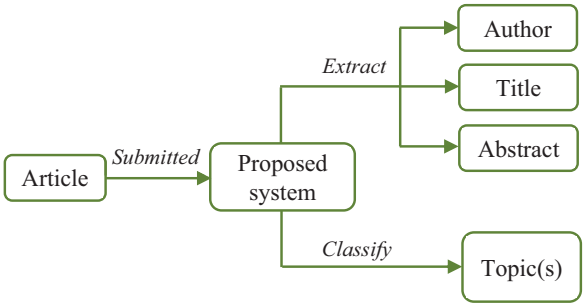

Fig. 2: Model of extraction and classification of articles

With this model, when a new article (.doc(x), .pdf) is submitted to the system, it is automatically extracted information such as author's name, title, abstract, and classified into an appropriate topic based on the trained machine learning model.

Since each article is a pre-formatted template, it is easy to extract the author's information, title, and abstract. Therefore, this study only focuses on solving the problem of topic classification of the articles submitted to the system.

B. Steps of article classification

The automated classification of articles is divided into two phases. At training phase, relying on the collected dataset with machine learning algorithms, classification model generated is described in Fig. 3:

Fig. 3: Training phase

At testing phase, based on the classification model generated at the training step, articles are classified in the testing dataset. This stage is described in Fig. 4.

Fig. 4: Testing phase

1) Data pre-processing

a) File format conversion and word standardization: Because the dataset used is .doc(x) files, it is necessary to convert them to plain text (.txt) for easy use in most algorithms and for libraries serving automated classification. Converting format of an input article is based on Apache POI. Accordingly, Apache POI is used to perform read operations on .doc(x) file, then write the readable content to .txt file. After converting file format, word standardization is proceeded to convert all text characters into lowercase and delete spaces.

For example, the sentence "Xử Lý Ngôn Ngữ Tự nhiên là 1 nhánh của Trí tuệ nhân tạo" is standardized to "xử lý ngôn ngữ tự nhiên là 1 nhánh của trí tuệ nhân tạo".

b) Word segmentation: In Vietnamese, space does not segment words but separate syllables. Therefore, the segmentation phase is quite important in NLP.

Currently, many tools have been successfully developed to segment Vietnamese words with relatively high accuracy. In this study, the *VnTokenizer* segmentation tool by [18] was used. The tool was developed based on the integrated methods of maximum matching, weighted finite-state transducer and regular expression parsing, using the dataset of Vietnamese syllabary and Vietnamese vocabulary dictionary. This automated Vietnamese segmentation tool segments Vietnamese text into vocabulary units (words, names, numbers, dates and other regular expressions) with > 95% accuracy. The process of word segmentation using *VnTokenizer* is described in Fig. 5.

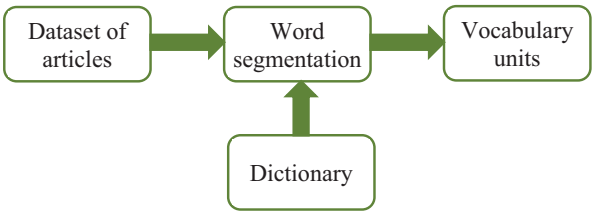

Fig. 5: Process of word segmentation using *VnTokenizer* [18]

For example, the sentence "xử lý ngôn ngữ tự nhiên là 1 nhánh của trí tuệ nhân tạo" is segmented into "xử_lý ngôn_ngữ tự_nhiên là 1 nhánh của trí_tuệ nhân_tạo".

c) Removing stop words: Stop words are the words that commonly appear in all texts of all categories in the dataset, or the words that appear only in one and several texts. It means that stop words do not make sense or do not contain information worth using. In text classification, the appearance of stop words (e.g. thì, là, mà, và, hoặc, bởi, etc.) not only do not help assessing the classification but also make noises and reduce the accuracy of the classification process.

For example, when being removed stop words, the sentence "xử_lý ngôn_ngữ tự_nhiên là 1 nhánh của trí_tuệ nhân_tạo" becomes "xử_lý ngôn_ngữ tự_nhiên nhánh trí_tuệ nhân_tạo".

In this study, after converting the article from .doc(x) to .txt format and segmenting words, the stop word dictionary was used to remove stop words.

2) Text vectorization

There are a number of text representation models, i.e. vector space model based on the frequency weighting method, bag of words model, and graph-based model. In this study, the vector space model [10] was applied. The vector space model can represent an unformatted text document as simple and formulaic notation. Because of the its advantage, lots of researches on vector space model are being actively carried out [28]. According to this model, each article is represented as a vector; each component of the vector is a separate term and is assigned a value that is the weight of this separate term.

The problem of text representation by vector space model is as follows: the input is a set of j documents in application domain D, with $D = \{d_1, d_{2,...} d_j\}$ and m terms (or words) in each document $T = \{t_1, t_{2,...} t_m\}$. In the output, the weight for each term is in turn determined, then the weighting matrix w_{ij}, the weight of term t_i in the document $d_j \in D$, is developed.

For determining the weight of word t_i in text d_j, TF (term frequency) and IDF (inverse document frequency) are commonly used.

TF is used to estimate the occurrence frequency of a term in a certain document. Due to each document has its own length and number of words, appearance frequency of terms is different. To measure the weight of a term, the number of occurrences of the term is divided by the length of the document (the number of words).

$$TF\ (t_i, d_j) = \frac{number\ of\ occurrences\ of\ term\ t_i\ in\ document\ d_j}{total\ number\ of\ terms\ in\ document\ d_j}$$

The values w_{ij} are calculated based on the occurrence frequency (number of times) of term in document. Given f_{ij} is the number of occurrences of term t_i in document d_j, then w_{ij} is calculated by one of three following basic formulas:

$$\begin{cases} w_{ij} = f_{ij} \\ w_{ij} = 1 + \log(f_{ij}) \\ w_{ij} = \sqrt{f_{ij}} \end{cases}$$

Where, f_{ij} the number of occurrences of term t_i in document d_j. If t_i occurs in document d_j, $w_{ij} = 1$, if not, $w_{ij} = 0$.

IDF is used to estimate the importance of a term in a document. When calculating the TF, all the terms have the same importance. However, it is found that not all terms in a dataset are important. Accordingly, the terms that do not have a high degree of importance are connecting terms (such as "nhưng", "bên cạnh đó", "vì thế", etc.), determiners ("kìa", "đó", "ấy", etc.); and prepositions ("trên", "trong", "ngoài", etc.). It is necessary to reduce the importance of those terms by calculating IDF by the formula as follows:

$$IDF(t_i, D) = log \frac{Total\ document\ in\ D}{Number\ of\ document\ including\ t_i}$$

The terms commonly occurring in many documents having low weight. The weights are determined as formula:

$$\begin{cases} w_{ij} = log\left(\frac{m}{df_i}\right) = log(m) - log(df_i), & if\ TF_{ij} \geq 1 \\ w_{ij} = 0, & if\ TF_{ij} = 0 \end{cases}$$

TF*IDF is the integration between TF and IDF. This common method is to calculate the TF*IDF value of a term through its importance in a document belonging to a ducument set.

High IF*IDF terms commonly occurr in a certain document but less occur in other document s. Through this method, it is possible to filter out common words and retain high value words.

$$TF * IDF\ (t_i, d_j, D) = TF\ (t_i, d_j)\ x\ IDF\ (t_i, D)$$

Weight calculation formula:

$$\begin{cases} w_{ij} = \left(1 + \log(f_{ij})\right) \log\left(\frac{N}{df_i}\right), & if\ f_{ij} \geq 1 \\ w_{ij} = 0, & if\ f_{ij} = 0 \end{cases}$$

IV. EXPERIMENTAL RESULTS

This study used two experimental datasets including 680 scientific articles (10 topics) and 10,000 articles of newsletters.

A. Experimental dataset 1: the scientific articles

The experimental dataset consists of 680 scientific articles, being published in Can Tho University Journal of Science from 2016 to 2018, belonging to 10 topics as described in Table 1.

The articles were pre-processed by converting .doc(x) to .txt files, then proceeded word segmentation [18]. The removal of stop words was done using a stop word dictionary-based approach, there remained 4,095 terms (words). In the process of modeling each document was a weighted word vector. Therefore, the modeled dataset was the TF*IDF matrix of terms with a size of 680 * 4,095 elements.

TABLE 1: DISTRIBUTION OF SCIENTIFIC ARTICLES IN 10 TOPICS

No	Topics	Training samples	Testing samples	Total samples (articles)
1	Technology	45	5	50
2	Environment	54	6	60
3	Natural Sciences	54	6	60
4	Animal husbandry	36	4	40
5	Biotechnology	27	3	30
6	Agriculture	90	10	100
7	Fisheries	135	15	150
8	Education	36	4	40
9	Social sciences and Humanities	72	8	80
10	Economics	63	7	70
	Total	612	68	680

After pre-processing and vectorization, the dataset of articles is trained with automated text classification algorithms of SVM, Naïve Bayes, and kNN. The dataset is automatically separated, in which 612 articles (90%) were used as a training set; 68 remaining articles (10%) as a testing set. The three machine learning algorithms of SVM, Naïve Bayes, and kNN were used to do text classification. The evaluation is based on precision, recall and F_1 score shown in Table 2.

It can be seen from Table 2 that the classification performances of the algorithms show their relative effectiveness. In which, SVM algorithm gives the best classification results with the accuracy > 91%, it is feasible for developing automated classification system of scientific articles, contributing to the faster and more accurate process of classifying articles. This result is also consistent with many studies by [6], [7], [12] and [23] that SVM method gives text classification results equivalent or significantly better than those of other classifiers.

TABLE 2: COMPARISON OF CLASSIFICATION RESULTS FOR SCIENTIFIC ARTICLES AMONG ALGORITHMS OF SVM, NAÏVE BAYES, AND KNN

	SVM			Naïve Bayes			kNN		
Topics	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
Technology	0.857	0.857	0.857	0.857	0.857	0.857	0.400	0.571	0.471
Environment	1.000	0.333	0.500	0.400	0.333	0.364	0.667	0.333	0.444
Natural Sciences	0.750	1.000	0.857	0.667	0.667	0.667	0.600	1.000	0.750
Animal husbandry	1.000	1.000	1.000	1.000	0.500	0.667	1.000	0.500	0.667
Biotechnology	1.000	0.500	0.667	1.000	0.500	0.667	1.000	0.500	0.667
Agriculture	0.786	1.000	0.880	0.846	1.000	0.917	0.733	1.000	0.846
Fisheries	0.947	1.000	0.973	0.857	1.000	0.923	0.947	1.000	0.973
Education	1.000	1.000	1.000	1.000	0.500	0.667	1.000	1.000	1.000
Social sciences and Humanities	1.000	1.000	1.000	0.600	0.750	0.667	0.600	0.750	0.667
Economics	1.000	1.000	1.000	0.900	0.818	0.857	1.000	0.545	0.706
Average accuracy rate			**91.2%**			**80.9%**			**76.5%**

However, not all topics are well classified. Even with the best classification algorithm (SVM), there are still some topics of poor classification, such as "Environment" and "Biotechnology" with $F_1 < 67\%$ compared to other topics ($F_1 > 85\%$). The reason is that the recall of these two topics is quite low (<= 0.5), in other words, the rate of the articles accurately predicted to belong to these two topics is not high in reality. This is explained by the overlap of the topics, in other words, an article can belong to both similar topics (e.g. "Agriculture" and "Fisheries") which was called multi-class classification. The remaining topics have relatively high precision and recall, indicating their distinctions with others.

B. Experimental dataset 2: the articles of newsletters

The experimental dataset consists of 10,000 articles from Vnexpress newsletters that was collected in 2018, belonging to 10 topics as described in Table 3.

TABLE 3: DISTRIBUTION OF ARTICLES OF VNEXPRESS NEWSLETTERS IN 10 TOPICS

No	Topics	Training samples	Testing samples	Total samples (articles)
1	Business	800	200	1,000
2	Culture	800	200	1,000
3	Health	800	200	1,000
4	IT	800	200	1,000
5	Law	800	200	1,000
6	Life	800	200	1,000
7	Politics	800	200	1,000
8	Science	800	200	1,000
9	Sports	800	200	1,000
10	World	800	200	1,000
	Total	8,000	2,000	10,000

In experimental dataset 2, the pre-processing steps carried out as work of experimental dataset 1. The dataset was also

automatically separated, in which 8,000 articles (90%) were used as a training set and 2,000 remaining articles (10%) as a testing set.

The stop words were removed using dictionary-based approach. For the training set, there remained 3,485 terms. The process of modeling each document was a weighted word vector. Therefore, the modeled dataset was the TF*IDF matrix of words with a size of 8,000 * 3,485 elements. For the testing set, there remained 3,961 words. In the process of modeling, each document was a weighted word vector. Therefore, the modeled dataset was the TF*IDF matrix of terms with a size of 8,000 * 3,961 elements.

We used k-fold cross validation method with k=3 (one of cross-validation commonly used). The three machine learning algorithms as above were used to classify. The evaluation is based on precision, recall and F_1 score shown in Table 4.

TABLE 4: COMPARISON OF CLASSIFICATION RESULTS FOR ARTICLES OF VNEXPRESS NEWSLETTERS AMONG ALGORITHMS OF SVM, NAÏVE BAYES, AND KNN

	SVM			Naïve Bayes			kNN		
Topics	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
Business	0.775	0.915	0.839	0.807	0.880	0.842	0.513	0.766	0.615
Culture	0.922	0.950	0.936	0.919	0.910	0.915	0.655	0.660	0.658
Health	0.968	0.910	0.938	0.911	0.870	0.890	0.784	0.621	0.693
IT	0.906	0.920	0.913	0.887	0.940	0.913	0.461	0.600	0.521
Law	0.989	0.865	0.923	0.854	0.880	0.867	0.825	0.561	0.668
Life	0.973	0.910	0.941	0.917	0.825	0.868	0.687	0.811	0.744
Politics	0.776	0.850	0.811	0.679	0.815	0.741	0.582	0.453	0.509
Science	0.918	0.845	0.880	0.896	0.815	0.853	0.623	0.435	0.512
Sports	0.985	0.975	0.980	0.989	0.935	0.961	0.938	0.753	0.835
World	0.917	0.935	0.926	0.946	0.870	0.906	0.530	0.670	0.592
Average accuracy rate			**90.1%**			**87.6%**			**46.7%**

SVM algorithm got significantly relative values of precision, recall, thus F_1. The classification results with accuracy was greater than 90%. The second effective algorithm was Naïve Bayes with accuracy was 87.6%. However, kNN algorithm classifier implementation gave low accuracy with 46.7%. Normally, kNN performs better with a lower number of features than a large number of features. When the number of features increases, it leads to increase in dimension, thus leads to the problem of overfitting.

The above results shows that the combination of natural language processing and machine learning algorithm (e.g., SVM) is effective for developing automated classification system of articles in general.

V. CONCLUSIONS

Based on natural language processing and machine learning algorithms, this study proposed a solution of automated classification of articles to help authors/editors save time and efforts when processing articles on the system. Data pre-processing steps are significant to make classification dataset in a standardized format for running the three algorithms of SVM, Naïve Bayes, and kNN. The results showed that SVM algorithm gives better classification performance than the remaining classifiers.

With the proposed model, it is feasible to extract information and automatically classify articles being submitted to a classification system. The experiments on larger datasets should be conducted in the future.

VI. ACKNOWLEDGEMENT

This study is funded in part by the Can Tho University Improvement Project VN14-P6, supported by a Japanese ODA loan.

VII. REFERENCES

[1] K. Thaoroijam, "A Study on Document Classification using Machine Learning Techniques", IJCSI International Journal of Computer Science Issues, vol. 11, no. 2, pp. 1694-0784, 2014.

[2] Y. Li, L. Zhang, Y. Xu, Y. Yao, R. Y. K. Lau and Y. Wu, "Enhancing Binary Classification by Modeling Uncertain Boundary in Three-Way Decisions," in IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 7, pp. 1438-1451, 2017.

[3] F. Sebastiani, "Machine learning in automated text categorization", ACM Computing Surveys, vol. 34, no. 1, pp. 1-47, 2002.

[4] C. Aggarwal and C. Zhai, "A Survey of Text Classification Algorithms", Mining Text Data, pp. 163-222, 2012.

[5] M. Bijaksana, Y. Li and A. Algarni, "A Pattern Based Two-Stage Text Classifier", Machine Learning and Data Mining in Pattern Recognition, pp. 169-182, 2013.

[6] B. Boser, I. Guyon and V. Vapnik, "A training algorithm for optimal margin classifiers", Proceedings of the fifth annual workshop on Computational learning theory - COLT '92, 1992.

[7] C. Burges, Data Mining and Knowledge Discovery, vol. 2, no. 2, pp. 121-167, 1998.

[8] J. Chen, H. Huang, S. Tian and Y. Qu, "Feature selection for text classification with Naïve Bayes", Expert Systems with Applications, vol. 36, no. 3, pp. 5432-5435, 2009.

[9] C.-H, Tsai, "MMSEG: A Word Identification System for Mandarin Chinese Text Based on Two Variants of the Maximum Matching Algorithm", 1996. Available: http://technology.chtsai.org/mmseg/.

[10] S. P. Christian, "Machine Learning: Cosine Similarity for Vector Space Models (Part III)", 2013. Available: http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/.

[11] C. Cortes and V. Vapnik, "Support-vector networks", Machine Learning, vol. 20, no. 3, pp. 273-297, 1995.

[12] S. Dumais, J. Platt, D. Heckerman and M. Sahami, "Inductive learning algorithms and representations for text categorization", Proceedings of the seventh international conference on Information and knowledge management - CIKM '98, 1998.

[13] M. Haddoud, A. Mokhtari, T. Lecroq and S. Abdeddaïm, "Combining supervised term-weighting metrics for SVM text classification with extended term representation", Knowledge and Information Systems, vol. 49, no. 3, pp. 909-931, 2016.

[14] H. J. George and P. Langley, "Estimating continuous distributions in Bayesian classifiers", Proceedings of the Eleventh conference on Uncertainty in artificial intelligence (UAI'95), 1995.

[15] B. Liu, Y. Dai, X. Li, W. S. Lee and P. S. Yu, "Building text classifiers using positive and unlabeled examples," Third IEEE International Conference on Data Mining, Melbourne, FL, USA, pp. 179-186, 2003.

[16] H. KDE Group, "A Comparison of Event Models for Naive Bayes Text Classification", AAAI-98 Workshop on Learning for Text Categorization, pp. 41-48, 1998.

[17] T. Mitchell, "Machine Learning", 1997. Available: https://dl.acm.org/citation.cfm?id=541177

[18] N. T. M. Huyen, V. X. Luong and L. H. Phuong, "VnTokenizer", 2010. Available: http://vntokenizer.sourceforge.net/.

[19] S. Chakrabarti, Mining the Web. Burlington: Morgan Kaufmann, 2003.

[20] P. Tan, M. Steinbach and V. Kumar, "Introduction to Data Mining", 2006. Available: https://www-users.cs.umn.edu/~kumar001/dmbook/index.php.

[21] T. C. De and P. N. Khang, "Text classification with support vector learning and decision tree", Can Tho University Journal of Science, vol. 21a, pp. 52-63, 2012 (in Vietnamese).

[22] T. T. T. Thao and V. T. Chinh, "Development of Vietnamese text classification system", Report of scientific research, 2012 (in Vietnamese).

[23] Y. Yang and O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization", Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97), pp. 412-420, 1997.

[24] L. Zhang, Y. Li, C. Sun and W. Nadee, "Rough Set Based Approach to Text Classification", 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013.

[25] M. Haddoud, A. Mokhtari, T. Lecroq and S. Abdeddaïm, "Combining supervised term-weighting metrics for SVM text classification with extended term representation", *Knowledge and Information Systems*, vol. 49, no. 3, pp. 909-931, 2016.

[26] N. Thai-Nghe and Q. D. Truong, “An Approach for Building A Semi-Automatic Online Consultancy System”. Proceedings of International Conference on Advanced Computing and Applications (ACOMP 2015). pp 51-58, ISBN-13: 978-1-4673-8234-2, IEEE Xplore. 2015.

[27] K. Gábor, D. Buscaldi, A. Schumann, B. QasemiZadeh, H. Zargayouna and T. Charnois, "SemEval-2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers", Proceedings of The 12th International Workshop on Semantic Evaluation, 2018.

[28] J. Chang and I. Kim, "Analysis and Evaluation of Current Graph-Based Text Mining Researches", Advanced Science and Technology Letters, 2013