Text Filtering Method Based on Term Selection and Rocchio Classifier in Uyghur Forums (Post-print)

Authors: Ruxianguli Abudurexiti, Yasin Eziz, Aishan Omar, Alimujiang Aisha

Date: 2018-05-18T00:00:00+00:00

Abstract

To address the text filtering problem in Uyghur web forums, we propose a text filtering method based on term selection and Rocchio classifier. First, forum texts are preprocessed to remove stop words, and stem (term) extraction is performed based on an N-gram statistical model. Then, a Balanced Mutual Information Term Selection method (BMITS) that balances relevance and redundancy is proposed to reduce the dimensionality of the initial term set and obtain a refined term set. Finally, text feature terms are used as input and classified through a Rocchio classifier, thereby filtering out undesirable texts from the forum. Experimental results on relevant datasets show that the proposed method can accurately identify undesirable text types and is effective.

Full Text

Preamble

Text Filtering Method Based on Term Selection and Rocchio Classifier in Uyghur Forums

Ruxianguli Abudurexiti[1], Yasen Aizezi[1]†, Aishan Wumaier[2] , Alimu Aisha[2]

(1. Department of Information Security Engineering, Xinjiang Police College, Urumqi 830013, China;
2. a. School of Information Science and Engineering; b. Network Center, Xinjiang University, Urumqi 830046, China)

Abstract: To address the problem of text filtering in Uyghur web forums, this paper proposes a text filtering method based on term selection and Rocchio classifier. First, forum texts are preprocessed to remove useless words, and stemming (term extraction) is performed based on an N-gram statistical model.

Then, a Balanced Mutual Information Term Selection method (BMITS) is proposed, which considers both relevance and redundancy in a balanced manner to reduce the dimensionality of the initial term set and obtain a refined term set. Finally, the text feature terms are used as input to the Rocchio classifier for classification, thereby filtering out undesirable texts from the forum. Experimental results on relevant datasets demonstrate that the proposed method can accurately identify undesirable texts and is effective.

Keywords: Uyghur forum; text filtering; N-gram statistical model; term selection; Rocchio classifier

0 Introduction

With the rapid development of the Internet, web forums have proliferated dramatically. While forums facilitate information exchange among netizens and improve work and learning efficiency, their open nature also introduces negative effects, such as the dissemination of illegal information involving superstition, reactionary content, and violent pornography, which is detrimental to social stability and people' s physical and mental health [1,2]. Therefore, filtering illegal texts in web forums is of great importance.

In recent years, with the state' s strong support for the development of Xinjiang, network infrastructure has advanced rapidly, giving rise to numerous Web forums written in Uyghur. Some separatist elements have used these Uyghur forums to spread various types of undesirable information. Consequently, filtering undesirable texts in Uyghur web forums contributes to long-term stability in Xinjiang [3]. To achieve text filtering in Uyghur web forums, the primary task is to effectively classify these texts and then delete those classified as undesirable [4].

Regarding research on Uyghur text classification, scholars have proposed several methods in recent years. For instance, literature [5] proposed a Uyghur text classification method based on combined statistics (Dme), which includes mutual information, t-test, and entropy values for stemming extraction and dimensionality reduction, using the k-nearest neighbor (k-NN) algorithm as the text classifier. Literature [6] proposed a Uyghur sentiment analysis method based on term frequency-inverse document frequency (TF-IDF) and support vector machine (SVM), using TF-IDF to obtain discriminative keywords. Literature [7] presented a Uyghur text classification technique based on the N-gram model and Manhattan distance, employing 4-gram models and incorporating dice measurement into Manhattan distance. However, these methods cannot effectively reduce feature dimensions, resulting in low text classification accuracy and high computational cost.

Therefore, this paper proposes a text filtering method based on term selection and Rocchio classifier for Uyghur web forums, and demonstrates its effectiveness through comparative experiments. The main contributions of the proposed method are as follows:

a) Stemming (term extraction) is performed through an N-gram statistical model, with experiments confirming that N=4 is most suitable for Uyghur text characteristics.

b) To address the limitations of traditional Mutual Information Term Selection (MITS), a Balanced MITS (BMITS) is proposed that considers both relevance and redundancy in a balanced manner to select a highly discriminative term subset from the initial term set.

c) The Rocchio classifier is selected for text classification and filtering of undesirable texts due to its superior efficiency and generalization capability.

1 Uyghur Text Classification Description

1.1 Uyghur Language Structure

Uyghur is a highly agglutinative language whose words consist of 32 letters, each with four different forms, resulting in richer tense and morphological variations than English. In Uyghur, grammatical functions are achieved by adding different affixes to the end of words—many words evolve from the same root with similar meanings. These characteristics lead to problems such as high original feature dimensionality and sparse text representation in Uyghur texts [9], making it quite different from traditional Chinese or English text classification methods.

Uyghur verbs and some nouns are formed from word roots. Words have fixed patterns, and their number, gender, and tense can be expressed by adding prefixes and suffixes. Table 1 shows different affixes that can be added to the word “ ” (poet) and their meanings, where the underlined letters represent affixes.

1.2 Uyghur Text Classification Process

Numerous researchers have studied Chinese and English text classification, but few have focused on Uyghur text classification. This section describes the three main steps of Uyghur text classification: stemming extraction, feature dimensionality reduction, and text classification.

Stemming extraction is the process of removing all affixes from a word to obtain the root, thereby improving performance in text information retrieval tasks, particularly for highly agglutinative languages like Uyghur. In Chinese and English text classification research, stemming typically involves removing suffixes and stop words. Root-based stemming techniques use morphological analysis to extract roots from given Uyghur words. For example, literature [10] attempted to find word roots by matching words against all possible patterns and affixes, but this algorithm cannot remove any prefixes or suffixes. Literature [11] used different algorithms in a morphological analysis system to find roots and patterns by first removing the longest prefix and then extracting the root by examining the first five letters of the word. However, this algorithm is based on the assumption that the root must appear in the first five letters of the word.

In statistical extraction methods, the N-gram model [12] is commonly used, which groups related words based on string similarity metrics. The N-gram model extracts a set of N consecutive characters from a word, where similar words will have a high N-gram ratio.

Feature dimensionality reduction is used to reduce the dimensionality of the extracted root set and obtain a refined feature set. After removing stop words and extracting stems, each text is represented by a vector with N weighted terms, known as the bag-of-words method for text representation. This process ignores text structure and word order, with the feature vector representing words observed in the text. The super-vector in the training set consists of all sample stems (also called terms) from the training set. Typically, there are a large number of terms in text classification, necessitating dimensionality reduction of the term space. In English text classification, term evaluation functions such as document frequency, mutual information gain, ^{2} statistics, NGL coefficient, and GSS coefficient are commonly used to reduce the term set and select important terms [13].

Text classification categorizes texts based on their input features. Currently, the common approach involves inductive learning from manually classified texts to train classifiers. There are two different methods for building classifiers: parametric and non-parametric methods. In parametric methods, training data is used to estimate parameters of probability distributions, such as the probabilistic Naive Bayes classifier. Non-parametric methods can be further divided into two categories: sample-based non-parametric methods that classify texts by comparing them with training set texts and assigning them to the class with the highest similarity (e.g., k-nearest neighbor classifier), and description-based non-parametric methods that first represent classification descriptions (or linear classifiers) as a vector of term weights obtained from pre-classified training set texts, and then use these descriptions as training data for comparison with texts to be classified (e.g., Rocchio classifier).

2 Proposed Uyghur Text Classification Model

This paper aims to apply machine learning methods to classify and filter texts in Uyghur web forums. The proposed model mainly consists of three stages: text preprocessing (stemming extraction), term selection, and text classification. Figure 1 [Figure 1: see original paper] illustrates the proposed Uyghur text classification model.

The text preprocessing stage involves stemming extraction, including removal of stop words and words with common roots. Subsequently, a super-vector is constructed, and feature selection techniques are applied to reduce the dimensionality of the super-vector, representing texts as term weight vectors. Finally, a classifier is built and its performance is evaluated.

2.1 Stemming Extraction Based on N-gram Statistical Model

For stemming extraction, there are typically root-based methods and statistical methods. Compared to root-based methods, statistical methods are more suitable for Uyghur text classification tasks. This paper adopts the N-gram statistical model to extract Uyghur stems, using letter-level N-grams where all continuous sequences of N letters are treated as a unit called a gram.

In the N-gram model, the probability of a letter unit appearing in text is assumed to depend only on the previous N-1 letters. Therefore, the probability of a letter sequence is expressed as:

$$P(l_1, ..., l_N) = \prod_{i=1}^{N} P(l_i | l_{i-N+1}, ..., l_{i-1})$$

In Uyghur, due to the high probability of letters combining with each other, short N values cannot well represent word properties, while longer values such as N=3,4 exhibit stronger discriminative capability.

The following example describes the similarity calculation between two words (politics) and (political) based on the N-gram model (N=2):

1. → , , , (first decompose the word into two-letter combinations)
2. → , , , , (decompose the second word into two-letter combinations)
3. Similarity is calculated as:

$$S = \frac{2C}{A + B}$$

where A and B represent the number of distinct two-letter combinations in the first and second words respectively, and C represents the number of common two-letter combinations between the two words. Words with similarity greater than threshold T are clustered and represented by a single term.

In the stemming extraction method based on the N-gram statistical model proposed in this paper, we first remove the most common prefixes and suffixes in words, as well as foreign words, numbers, and stop words. Then, the similarity between two words is calculated using the N-gram model to extract stems. The stemming extraction algorithm based on the N-gram statistical model is shown in Algorithm 1.

Algorithm 1: Stemming Extraction Based on N-gram Statistical Model

For each word in the text: - If it is a non-Uyghur word, mark it as useless; - If it contains digits, mark it as useless; - If word length < 3, mark it as useless; - Remove attached symbols and normalize the word; - If it is a stop word, mark it as useless; - Remove prefixes and suffixes; - If it is a stop word, mark it as

useless; - Use N-gram statistical model to calculate similarity between words to obtain stems; End For

The algorithm first ensures that a word is a Uyghur word and considers words with fewer than three letters as unimportant in the document. It then removes various attached symbols that appear above or below letters for orthographic purposes as morphological markers. Afterward, word standardization is applied to unify different forms of some letters (extended areas) into the same form, such as unifying , , , into , and unifying , into .

After word form standardization, the algorithm checks whether the word is in a stop word list consisting of 165 words. After eliminating stop words, the algorithm removes a set of prefixes ,) , , , , , , etc.). After removal, the algorithm checks if the word length is less than 3 letters; if so, the removed prefix is restored to the word as it constitutes a major part of the word. Then suffixes ,) , , , , , , , , , , , , , etc.) are recursively removed from the end of the word, starting with the longest suffix and then removing shorter ones.

After removing both prefixes and suffixes, the algorithm checks again whether the word belongs to the stop word list, as some stop words may also have prefixes and suffixes attached. Finally, the similarity between words is calculated using the N-gram statistical model to obtain the final stem.

2.2 Term Selection Based on BMITS

2.2.1 Traditional Term Selection Methods To improve classifier performance, dimensionality reduction of the term set from input texts is necessary. Term selection techniques are used to select the term subset that best expresses the meaning of the text from the initial term set. Typically, a term evaluation function is used to score each term in the initial set, and terms with higher scores are selected.

In existing research, common feature dimensionality reduction techniques include mutual information, ² statistics, NGL coefficient, and GSS coefficient methods, with expressions as follows:

- **Mutual Information Gain:**

$$MI(t_k, c_i) = \sum_{t_k \in \{t_k, \bar{t}_k\}} \sum_{c_i \in \{c_i, \bar{c}_i\}} P(t_k, c_i) \log \frac{P(t_k, c_i)}{P(t_k)P(c_i)}$$

- **² Statistics:**

$$\chi^2(t_k, c_i) = \frac{[P(t_k, c_i)P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i)P(\bar{t}_k, c_i)]^2}{P(t_k)P(\bar{t}_k)P(c_i)P(\bar{c}_i)}$$

- **NGL Coefficient:**

$$NGL(t_k, c_i) = \frac{\sqrt{N}[P(t_k, c_i)P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i)P(\bar{t}_k, c_i)]}{\sqrt{P(t_k)P(\bar{t}_k)P(c_i)P(\bar{c}_i)}}$$

- **GSS Coefficient:**

$$GSS(t_k, c_i) = P(t_k, c_i)P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i)P(\bar{t}_k, c_i)$$

where $P(t_k, c_i)$ represents the probability that term t_k appears in text x and text x belongs to class c_i.

2.2.2 Mutual Information Term Selection Method (MITS) Mutual Information (MI) represents the degree to which one random variable contains information about another variable and is a measure of statistical correlation. Its output is a non-negative value, where zero indicates that two variables are statistically independent.

The method of selecting text terms based on mutual information theory is called Mutual Information Term Selection (MITS) [14]. In MITS, terms are selected by calculating $I(c_i; t_k|S)$, which represents the mutual information between the new term set formed by adding term t_k to the selected term set S and the text class c_i, reflecting the contribution of term t_k to text classification.

$I(c_i; t_k|S)$ is obtained by calculating the mutual information between term t_k and class c_i, then calculating the sum of mutual information between term t_k and previously selected terms t_s, and multiplying it by a scaling coefficient β:

$$I(c_i; t_k|S) = I(c_i; t_k) - \beta \sum_{t_s \in S} I(t_s; t_k)$$

The first term on the right side represents the mutual information between the candidate term and the text class, indicating relevance; the second term represents the sum of mutual information between the candidate term and selected terms, indicating redundancy. The β value represents the weight given to considering redundancy among terms during term selection, determining the importance of two aspects (MI between term and text class, and MI between terms). If β is too small, the algorithm emphasizes MI between candidate terms and text classes, selecting terms sequentially based on individual MI; if β is too large, the algorithm emphasizes MI between candidate terms. Therefore, selecting an appropriate β is difficult, and there is currently no suitable method for choosing this parameter.

To address this issue, this paper proposes a Balanced MITS algorithm (BMITS) that considers relevance and redundancy in a balanced manner. BMITS introduces mutual information between candidate terms and text classes in the

second term and eliminates the need for manually setting an additional parameter to replace β. BMITS selects terms that maximize relevance and minimize redundancy from an initial term set, expressed as:

$$I(c_i;t_k|S) = I(c_i;t_k) - \frac{1}{|S|}\sum_{t_s \in S}\frac{I(t_s;t_k)}{I(c_i;t_k)}$$

where $|S|$ is the number of selected terms. The relative minimum redundancy of term t_k for term t_s is defined as:

$$MR(t_k,t_s) = \frac{I(t_s;t_k)}{I(c_i;t_k)}$$

When $MR(t_k,t_s) = 0$, term t_k can be discarded. If t_k and t_s are highly correlated for text class c_i, then t_k will also be redundant. Therefore, a threshold needs to be set to compare with $MR(t_k,t_s)$. If $MR(t_k,t_s) > \theta$, the current term t_k is unimportant for text class c_i, as it will reduce the MI between the selected term set S and text class c_i, and it should be removed from F; if $MR(t_k,t_s) = 0$, term t_k is treated as a candidate term. The term selection process of BMITS is shown in Algorithm 2.

Algorithm 2: BMITS Term Selection

Input: Initial term set $T = \{t_1, t_2, ..., t_n\}$
Output: Selected term set S

1. Initialize $S = \emptyset$, $F = T$, $t_n = n$
2. While $F \neq \emptyset$ do:
   - Select term t_k from F according to:

$$t_k = \arg\max_{t_k \in F}\left(I(c_i;t_k) - \frac{1}{|S|}\sum_{t_s \in S}\frac{I(t_s;t_k)}{I(c_i;t_k)}\right)$$

   - Calculate $MIG = I(c_i;t_k|S)$
   - If $MIG \leq 0$ then:
     - $F \leftarrow F \setminus \{t_k\}$
   - Else:
     - $S \leftarrow S \cup \{t_k\}$
     - $F \leftarrow F \setminus \{t_k\}$
   - End if
   - $t_n = t_n - 1$
3. End while
4. Sort and weight each term in S according to its MIG value

2.3 Text Classification Based on Rocchio Classifier

The Rocchio classifier is a typical classifier applied in the field of text information filtering [15]. It builds a prototype vector for each class c_i, and classifies text vectors by calculating their distance to each prototype vector. The prototype vector for class c_i is obtained through weighted averaging of all text vectors belonging to class c_i. This means that compared with the k-NN classifier, the Rocchio classifier has faster learning speed.

For class c_i, its prototype vector can be calculated as:

$$\vec{w}_i = \frac{\beta}{|POS_i|} \sum_{\vec{x}_j \in POS_i} \vec{w}_j - \frac{\gamma}{|NEG_i|} \sum_{\vec{x}_j \in NEG_i} \vec{w}_j$$

where w_{jk} is the weight of term t_k in text x_j; POS_i is the set of texts belonging to class c_i (positive samples); NEG_i is the set of texts not belonging to class c_i (negative samples); β and γ are control parameters that set the relative importance of positive and negative samples. If β is set to 1 and γ to 0, class c_i is described as the centroid of its positive training samples. The Rocchio classifier performs classification based on the aggregation degree of positive samples and the separation degree of negative samples. The role of negative samples is typically inverse enhancement, which is achieved by setting a larger γ value and a smaller β value. According to relevant research, we can set $\beta = 1.6, \gamma = 0.4$ [16].

For an input sample of unknown category, the Rocchio classifier classifies the sample by comparing its minimum distance to each class prototype vector $\vec{w}_i$. This distance $d(\vec{x}, \vec{w}_i)$ is typically Euclidean distance. The decision rule of the Rocchio classifier is:

$$c^* = \arg\min_{c_i \in C} d(\vec{x}, \vec{w}_i)$$

The Rocchio algorithm addresses the problems of the k-NN algorithm by introducing some generalized instances. That is, it replaces the entire training sample set with a generalized instance obtained by summarizing the distribution of instance samples. When a new instance is added, its classification only requires calculating the Euclidean distance between the new instance and the generalized instance. Its time complexity is $O(LM)$, where L represents the number of generalized instances and M represents the number of terms in each text vector. Additionally, the Rocchio algorithm can solve noise problems according to the distribution of instances in each class. For example, if a term frequently appears in samples of a certain class, it will be equally reflected in the generalized instance of that class, resulting in a higher weight for that term. Conversely, if a term mainly appears in instances of other classes, its weight in the generalized instance will tend to 0. Therefore, the Rocchio classifier can extract certain relevant features to some extent.

3 Experiments

3.1 Uyghur Forum Text Collection

To construct an experimental Uyghur forum text collection, this paper collected approximately 2,400 forum texts from the Uyghur version of People' s Daily Online, Tianshan Net Forum, ulinix Forum, and Xinjiang Public Security Database. These texts are divided into five categories: normal, violent terrorism, reactionary, pornographic, and superstitious, with no fewer than 200 texts for each type. Balancing the proportion of samples across categories, 60% of the text collection is used as the training set and 40% as the test set.

3.2 Performance Metrics

Classifier performance is typically described using precision and recall. Precision represents the probability that a random text x is correctly classified into class i when it is assigned to class i. Recall represents the probability that a random text x that should belong to class i is correctly assigned to that class. The expressions for precision and recall are:

$$Precision_i = \frac{TP_i}{TP_i + FP_i}$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i}$$

where TP_i represents the number of texts correctly classified as class i, FP_i represents the number of texts incorrectly classified as class i, and FN_i represents the number of texts that belong to class i but are incorrectly classified into other classes.

Considering both precision and recall provides a better characterization of classifier performance. The F1 measure is typically used to combine these two parameters:

$$F1_i = \frac{2 \times Precision_i \times Recall_i}{Precision_i + Recall_i}$$

3.3 Stemming Extraction Performance Analysis

For the N-gram statistical model stemming method, two parameters need to be set: the N value and the similarity threshold T. A larger N value provides more semantic information and helps improve classifier accuracy, but significantly increases the number of feature terms and computational complexity. If the N value is too small, the resulting feature terms contain less semantic information and are less discriminative.

To determine the optimal parameters, we set N=2, 3, 4, and 5, and the similarity threshold T to 0.6, 0.7, 0.8, and 0.9, constructing 16 parameter combinations for stemming extraction and classification experiments. For fair comparison, subsequent feature selection methods all use BMITS, and the classifier uses the Rocchio classifier. The final classification F1 measure values for various parameters are shown in Figure 2 [Figure 2: see original paper].

The results in Figure 2 show that as the N value increases, classifier performance improves, but so does computational cost. The performance using five-letter combinations (N=5) is similar to that using four-letter combinations (N=4), with only a slight improvement. Considering computational cost, N=4 is ultimately selected. Additionally, when the similarity threshold $T = 0.8$, the best classification effect is achieved; when $T = 0.9$, the results of various letter combination stemming methods deteriorate. This is because when the threshold is too high, some words sharing the same root but with insufficient similarity will be separated rather than merged, reducing stemming effectiveness. Finally, N=4 and $T = 0.8$ are selected as parameters for the stemming extraction method.

Table 2 shows a set of stems extracted by the N-gram statistical model when $T = 0.8$ and N=4.

3.4 Term Selection Performance Analysis

After stemming extraction from input texts, a large term set is formed, making it necessary to find the most valuable term subset through term selection. This section compares the proposed BMITS selection method with traditional MITS, ^{2} statistics, NGL coefficient, and GSS coefficient methods. The size of the final selected feature set (term set) is varied from 1,000 to 5,000. For fair comparison, the same stemming extraction parameters and Rocchio classifier are used. The impact of term selection methods on classification performance is shown in Figure 3 [Figure 3: see original paper].

The experimental results in Figure 3 indicate that as the size of the feature set after term selection increases, the classification performance of all methods improves. However, when the feature set size exceeds 3,500, performance stabilizes, demonstrating that an appropriate feature set size affects classifier performance. If the feature set is too small, it cannot contain all effective terms; if too large, the number of redundant terms increases and adds to the classifier' s computational load. Therefore, in the term selection step of this paper, the feature set size is set to 3,500.

Through comparison of various methods, ^{2} statistics and traditional MITS show weaker performance, while NGL and GSS coefficients perform relatively well. However, the BMITS method proposed in this paper achieves the best performance because it introduces mutual information between candidate terms and classes, balancing relevance and redundancy, thus enabling selection of the optimal term set.

3.5 Overall Performance Comparison

To evaluate the overall performance of the proposed method, it is compared with several existing methods: the Uyghur classification method based on combined statistics (Dme) and k-NN classifier (Dme+k-NN) proposed in literature [5], the method based on TF-IDF and SVM (TF-IDF+SVM) proposed in literature [6], and the method based on N-gram model and Manhattan distance (N-gram+Manhattan) proposed in literature [7]. The classification performance comparison results are shown in Table 3 , along with the time required to classify all test samples.

Table 3 shows that the method in literature [5] requires the longest time because it uses three combined metrics for stemming extraction and term selection, requiring extensive computation. The method in literature [7] requires less time because it does not use supervised learning-based classifiers such as k-NN, but only classifies through a distance metric, which is computationally simpler. However, the method proposed in this paper consumes the least time because the Rocchio classifier uses class generalized vectors to replace all training text vectors in the corpus, which is significantly more efficient than k-NN and SVM classifiers in terms of time consumption. Additionally, the BMITS term selection process produces a refined and effective term subset, effectively improving classification performance.

Table 3: Classification Performance of Various Methods

Method	Precision (%)	Recall (%)	F1 Measure	Time (minutes)
Literature [5]	81.2	79.8	0.805	45
Literature [6]	83.5	82.1	0.828	38
Literature [7]	80.4	78.9	0.796	22
Our Method	**85.3**	**84.7**	**0.850**	**18**

4 Conclusion

This paper proposes an undesirable text classification method for application in Uyghur web forum text filtering. The method removes stop words through preprocessing, performs stemming extraction based on the N-gram model, reduces term dimensionality through BMITS, and finally uses the Rocchio classifier for text classification and filtering. To obtain the optimal parameters for the method, extensive experiments are conducted for optimization and selection. The final text filtering experimental results demonstrate that the proposed method can be effectively used for information filtering in Uyghur forums.

In future research, we hope to improve classification effectiveness through additional preprocessing work, such as selecting terms from local term libraries for each category rather than from a global term library of the entire corpus. Additionally, other classifiers such as Naive Bayes and neural network classifiers can be investigated to select a more suitable classifier.

References

[1] Liu L, Li Z, Zhang X, et al. Research review on semantic information processing of Chinese network text [J]. Application Research of Computers, 2015, 32(1): 6-10, 16.

[2] Cheng J, Li Z, Zou M, et al. Microblog news topic detection method based on SVM filtering [J]. Journal on Communications, 2013, 34(2): 74-78.

[3] Yalqin Almas, Halidan Abudureyimu, Chen Y. Uyghur text filtering method based on vector space model [J]. Journal of Xinjiang University: Natural Science Edition, 2015, 32(2): 221-226.

[4] Zhang B, Xu M, Wu M. Research on web filtering technology based on the dual feature selection [C]// Proc of IEEE International Conference on Network Infrastructure and Digital Content. Piscataway, NJ: IEEE Press, 2013: 675-679.

[5] Alimujiang Aisha, Turgun Ibraim, Aishan Wumaier, et al. Research on Uyghur text classification based on machine learning [J]. Computer Engineering and Applications, 2012, 48(5): 110-112.

[6] Reyilam Paerhati, Meng X, Askar Hamdulla. Uyghur sentiment classification based on discriminative keyword model [J]. Computer Engineering, 2014, 40(10): 132-136.

[7] Mamatimin Hasimu, Wushour Silamu, Winira Musajan, et al. Research on Uyghur text classification technology based on N-gram model [J]. Application Research of Computers, 2015, 32(7): 1986-1988, 2004.

[8] Mi C, Yang Y, Wang L, et al. Detection of loan words in uyghur texts [J]. Communications in Computer & Information Science, 2014, 49(6): 103-109.

[9] Abudusalamu Dawuti, Yusup Yusup, Askar Hamdulla. Uyghur sentence sentiment classification combining category-specific words and sentiment dictionary [J]. Journal of Tsinghua University: Science and Technology, 2017, 57(2): 197-201.

[10] Froud H, Lachkar A, Ouatik S A. A comparative study of root-based and stem-based approaches for measuring the similarity between arabic words for arabic text mining applications [J]. Advanced Computing: An International Journal, 2012, 3(6): 12-19.

[11] Hadni M, Ouatik S A, Lachkar A. Effective arabic stemmer based hybrid approach for arabic text categorization [J]. International Journal of Data Mining & Knowledge Management Process, 2013, 3(4): 1-14.

[12] Jiang Z, Ding X, Peng L, et al. Character modeling for segmentation-free Uyghur document recognition based on structural information sharing under low data resource conditions [J]. Journal of Electronics & Information Technology, 2015, 37(9): 2195-2201.

[13] Uchyigit G. Experimental evaluation of feature selection methods for text classification [C]// Proc of International Conference on Fuzzy Systems and Knowledge Discovery. Piscataway, NJ: IEEE Press, 2012: 1294-1298.

[14] Hoque N, Bhattacharyya D K, Kalita J K. MIFS-ND: a mutual information-based feature selection method [J]. Expert Systems with Applications, 2014, 41(14): 6371-6385.

[15] Sowmya B J, Chetan, Srinivasa K G. Large scale multi-label text classification of a hierarchical dataset using Rocchio algorithm [C]// Proc of International Conference on Computation System and Information Technology for Sustainable Solutions. Piscataway, NJ: IEEE Press, 2016: 1-5.

[16] Selvi S T, Karthikeyan P, Vincent A, et al. Text categorization using Rocchio algorithm and random forest algorithm [C]// Proc of the 3rd International Conference on Advanced Computing. Piscataway, NJ: IEEE Press, 2017: 1-5.

Note: Figure translations are in progress. See original paper for figures.