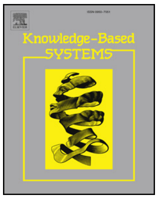
Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



A content-based recommender system for computer science publications

Donghui Wang[a], Yanchun Liang[a,b,c], Dong Xu[a,c], Xiaoyue Feng[a], Renchu Guan[*,a,b]

[a] *Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, College of Computer Science and Technology, Jilin University, Changchun 130012, China*

[b] *Zhuhai Laboratory of Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Zhuhai College of Jilin University, Zhuhai 519041, China*

[c] *Department of Computer Science, Informatics Institute, and Christopher S. Bond Life Sciences Center, University of Missouri, Columbia, MO 65211, USA*

ARTICLE INFO

Keywords:
Recommender system
Softmax regression
Chi-square feature selection
Computer science publications

ABSTRACT

As computer science and information technology are making broad and deep impacts on our daily lives, more and more papers are being submitted to computer science journals and conferences. To help authors decide where they should submit their manuscripts, we present the Content-based Journals & Conferences Recommender System on computer science, as well as its web service at http://www.keaml.cn/prs/. This system recommends suitable journals or conferences with a priority order based on the abstract of a manuscript. To follow the fast development of computer science and technology, a web crawler is employed to continuously update the training set and the learning model. To achieve interactive online response, we propose an efficient hybrid model based on chi-square feature selection and softmax regression. Our test results show that, the system can achieve an accuracy of 61.37% and suggest the best journals or conferences in about 5 s on average.

1. Introduction

With the flourishing of e-commerce, recommender system (RS) is undergoing rapid transformation in almost all aspects. These systems have been applied to many areas, such as movie recommendations [1–3], music recommendations [4,5], news recommendations [6,7], webpage and document recommendations [8]. Many companies have employed and benefited from recommender systems, such as the book recommendation of Amazon, music recommendation of Apple Music, and product recommendation of TaoBao.

While recommender systems for many areas have been in various stages of development, to the best our knowledge, a customized recommender system using abstract for authors of computer science publications has not been proposed until now. Meanwhile, with the fast development of artificial intelligence and cloud computing, more and more computer science conferences and journals are available. The number of computer science conferences exceeds 9585 (http://dblp.uni-trier.de/db/conf/) and the number of journals is more than 4152 (http://dblp.uni-trier.de/db/journals/). Facing so many publication venues, it is often hard for authors to select the most suitable journal or conference to publish their papers. Submitting a paper to a wrong journal often causes rejection, delay or less readership of the publication.

To help the authors find the most suitable venue and accelerate the submission process, we propose a recommender system on computer science publications referred to as the Publication Recommender System (PRS). This system is based on a new content-based filtering (CBF) recommendation model using chi-square and softmax regression, which are combined to construct a real-time online system. The contributions of the proposed recommendation system are as follows: (1) PRS is a non-profit driven recommender system covering 66 top computer science publication venues across more than five digital libraries, such as Springer, IEEE, ACM, AAAI and SIAM. It can simultaneously recommend top journals and conferences using the input of abstract or the whole manuscript. (2) Considering the continually expanding field of computer science and daily updated corpus of the ever-changing, the recommendation ability for cutting edge research areas and topics is essential. Our recommender model can automatically update once a new training set is formed. It can ensure that those cutting edge research topics can be recommended effectively. (3) PRS could respond to user in real time and easily be deployed on a web server. In order to make PRS more practical, all methods used in PRS are designed to be less computational complexity.

The rest of this paper is organized as follows. In Section 2, development process of recommender system and ways to implement recommender system are introduced. In Section 3, we briefly introduce

[*] Corresponding author at: Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, College of Computer Science and Technology, Jilin University, Changchun 130012, China.

E-mail addresses: xiaozhen1124@sina.com (D. Wang), ycliang@jlu.edu.cn (Y. Liang), xudong@missouri.edu (D. Xu), fengxy@jlu.edu.cn (X. Feng), guanrenchu@jlu.edu.cn (R. Guan).

https://doi.org/10.1016/j.knosys.2018.05.001
Received 8 February 2017; Received in revised form 21 February 2018; Accepted 1 May 2018
Available online 17 May 2018

the feature selection and classification methods related to our work, and the methods to select features and classify features used in the publication recommender system are proposed. Experimental results and analysis are in Section 4. The conclusion and future work are in the last section.

2. Background

In the late 20th century, Armstrong et al. [9] proposed a personalized navigation system called "Web Watcher" at the AAAI (American Association for Artificial Intelligence). Meanwhile, Balabanovic proposed a personalized recommender system called "LIRA" [10]. In August 1995, Henry Lieberman, from the Massachusetts Institute of Technology, presented a personalized navigation agent "Letizia" at the International Joint Conference on Artificial Intelligence (IJCAI). In 1997, AT&T Labs presented personalized recommendation systems based on collaborative filtering, called "PHOAKS" [11] and "Referral Web" [12]. In 1999, Tanja Joerding of Dresden University of Technology in Germany implemented a personalized e-commerce prototype system, "TELLIM" [13]. In 2001, IBM added personalized features to its e-commerce platform, "Websphere" [14], to enable businesses to develop personalized e-commerce sites. In 2003, Google made "AdWords" profitable which provided relevant ads through keywords that users searched for. In 2009, Overstock (US famous online retailers) began to use ChoiceStream company's personalized banner advertising program, to run product ads in some high-traffic sites.

Most of these recommendations are mainly implemented in two ways: collaborative or content-based filtering [15]. Collaborative filtering approaches learn a model from a user's past behavior as well as similar decisions made by other users, and predict items (or ratings for items) that users may be interested in [16]. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties [17]. The two approaches can also be combined as hybrid recommender systems. Other novel techniques can be introduced into recommendation system, such as social network and semantic information[18,19].

In terms of content-based filtering approaches, it tries to recommend items to the active user similar to those rated positively in the past. It is based on the concept that items with similar attributes will be rated similarly. The information source that content-based filtering systems are mostly used are text documents. A set of descriptors or terms, typically Term Frequency (TF) and Inverse Document Frequency (IDF), are used to describe the items. A standard approach for term parsing selects single words from documents. The vector space model and latent semantic indexing are two methods that use these terms to represent documents as vectors in a multi-dimensional space. CBF is becoming especially important as RS incorporate information on items from users working in web 2.0 environments, such as tags, posts, opinions and multimedia material [20].

3. Publication recommender system

Publication Recommender System consists of two modules: feature selection module and softmax regression module. Feature vector space is generated in feature selection module and feature vectors are used to train softmax regressor in softmax regression module. In this section, details of the two module will be introduced.

3.1. Feature selection module

In this subsection, the brief descriptions of TF-IDF and three feature selection models are introduced. Then, the feature selection method for Publication Recommender System is proposed.

3.1.1. TF-IDF

Term frequency and inverse document frequency (TF-IDF) can recognize the important words or phrases of articles [21]. It is the most common weighting method used to describe documents in the vector space model [22,23]. If a word is infrequent but its number of occurrence is large in one or a few articles, it probably plays a key role in the article. Moreover, the higher a word's $tf - idf$ is, the more important it is in the article. $tf - idf$ is calculated as a combination of the term frequency and inverse document frequency. tf is the number of times that a word w occurs in the document d. Considering the length of documents, tf should be standardized.

$$tf = T/L. \tag{1}$$

or

$$tf = T/T_i. \tag{2}$$

where, T is the term frequency, L is the count of the unique words in document d, and T_i denotes the frequency of the most frequent word in document d. idf reveals how much information the word provides. It is calculated using D and D_i, where D denotes the number of all documents and D_i is the number of the documents which include the word w.

$$idf = log\frac{D}{D_i + 1} \tag{3}$$

Then,

$$tf\text{-}idf = tf \times idf. \tag{4}$$

It can be seen that $tf - idf$ is proportional to T and inversely proportional to D_i.

3.1.2. Chi-square feature selection

The chi-square statistic (χ^2) measures the dependence between the term t and a category c (such as, in our case, computer journals or conferences), which can be seen as the χ^2 distributions with one degree of freedom to judge extremeness [24]. With a term t and a category c, χ^2 can be achieved using a two-way contingency table [25] (Table 1).

A is the number of documents including term t, which belongs to category c; B is the number of documents including t, which does not belong to c; C is the number of documents in category c, which does not include t; D is the number of documents in other categories and without term t. The χ^2 of t is defined as:

$$\chi^2(t, c) = \frac{N \times (AD - BC)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)} \tag{5}$$

where N is the total number of documents. Obviously, $A + B + C + D = N$. $A + C$ and $B + D$ are equal to a constant for each term t in category c when computing $\chi^2(t, c)$, and then Eq. (5) can be simplified as:

$$\chi^2(t, c) \approx \frac{(AD - BC)^2}{(A + B) \times (C + D)} \tag{6}$$

χ^2 has a natural value of zero if term t and category c are independent, which means term t does not contain any information about category c. In contrast, χ^2 has a high value if term t and category c are dependent.

Two other widely used feature selection models—mutual information (MI) and information gain (IG) can be calculated as follows:

$$MI(t, c) = log\left(\frac{A/(A + C)}{(A + B)/N}\right) \tag{7}$$

Table 1
Two-way contingency table for χ^2 .

$A = \#(t, c)$	$C = \#(\neg t, c)$
$B = \#(t, \neg c)$ $N = A + B + C + D$	$D = \#(\neg t, \neg c)$

$$IG(t, c) = Entropy(S) + \frac{A+B}{N_1+N_2}\left(\frac{A}{A+B}log\left(\frac{A}{A+B}\right) + \frac{B}{A+B}log\left(\frac{B}{A+B}\right)\right) + \frac{C+D}{N_1+N_2}\left(\frac{C}{C+D}log\left(\frac{C}{C+D}\right) + \frac{D}{C+D}log\left(\frac{D}{C+D}\right)\right) \tag{8}$$

$$Entropy(S) = -\frac{N_1}{N_1+N_2}log\left(\frac{N_1}{N_1+N_2}\right) - \frac{N_2}{N_1+N_2}log\left(\frac{N_2}{N_1+N_2}\right) \tag{9}$$

where $N_1 = A + C$, $N_2 = B + D$.

All the three metrics are used in our experiments to evaluate the performance of recommendation results in Section 3.

3.1.3. Proposed method

The feature selection procedure is to evaluate each feature according to a selection metric and pick the rich informative features. The evaluation involves counting the occurrences of a feature in a class with both positive and negative training examples. As mentioned in the previous subsection, χ^2 is used to measure the lack of independence between the feature t and category c. In fact, what we are interested in are those terms having strong relationships with their category. Therefore, a number of highly dependent words from every text category using χ^2 will be selected to generate a feature vector using Eq. (6).

Specifically, suppose N_c is the number of categories in the training set and t_j^i is the *jth* unique term in the *ith* category. The detailed procedure is as follows. Firstly, text preprocessing like tokenization, stopwords removal and stemming are performed on each document. Secondly, we use Eq. (6) to compute $\chi^2(t_j^i, c_i)$ for the *jth* unique term belonging to the *ith* category.

To reconstruct an effective vector space, we sort all $\chi^2(t_j^i, c_i)$ in descending order and select the top M terms as the feature vector space FV^i of the *ith* category. At last, we combine all feature vectors $FV^1FV^2\cdots FV^{N_c}$ together and remove the duplicate terms to generate a new feature vector space.

FV: $[t_1, t_2, \cdots, t_{N_{FV}}]$

where N_{FV} is the number of elements in the final feature vector space.

For a document j belonging to category i, we define its feature vector FV_j^i as:

$$FV_j^i = \left[f_j^i(t_1), f_j^i(t_2), \cdots, f_j^i(t_{N_{FV}})\right] \tag{10}$$

where $f_j^i(t_k)$ is $tf - idf$ of the *kth* feature term t_k in document j, which belongs to category i. Fig. 1 shows the feature selection and vector generation method.

For example, suppose there are two classes of all documents in training set, which are named c_1 and c_2. After text preprocessing, all unique items from c_1 and c_2 are t^1= {"despit", "signific", "invest", "commerci"} and t^2= {"far", "solv", "week", "invest"} respectively. Then, we use Eq. (6) to compute $\chi^2(t_j^i, c_i)$ which are $\chi^2(t^1, c_1) = \{0.1, 0.5, 0.7, 0.4\}$ and $\chi^2(t^2, c_2) = \{0.1, 0.3, 0.2, 0.8\}$ for $i = 1, 2, j = 1, 2, 3, 4$. We select the top $M = 2$ terms from t^i according to the value of χ^2 as the feature vector space of the *ith* category, which are FV^1= {"signific", "invest"} and FV^2= {"solv", "invest"}. Finally, we combine FV^1 and FV^2 together and remove the duplicate terms to generate the feature vector space FV= {"signific", "invest", "solv"}. It should be noted that all data above are not real and for simulation only.

3.2. Softmax regression module

Since features are selected and the feature vector space is generated, feature vectors could be computed easily and used to train a classifier. Softmax regression is chosen to be the classifier because there are many journals and conferences in the recommender system and all journals and conferences need to be ranked for recommending according to the classification scores.

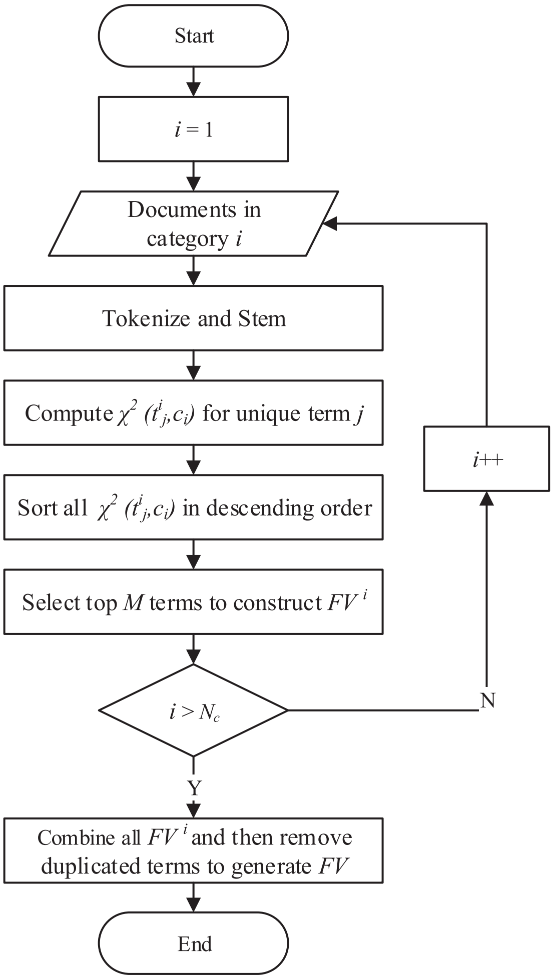

Start
$i = 1$
Documents in category i
Tokenize and Stem
Compute χ^2 (t^i_j, c_i) for unique term j
Sort all χ^2 (t^i_j, c_i) in descending order
Select top M terms to construct FV^i
$i > N_c$
N
Y
i++
Combine all FV^i and then remove duplicated terms to generate FV
End

Fig. 1. Flowchart of feature selection and vector generation.

3.2.1. Softmax regression

Softmax regression is the extension of logistic regression, which can solve multi-class problems (the number of classes $c > 2$). When $c = 2$, softmax regression is degenerated to logistic regression. Given a test input x, our hypothesis is to estimate the probability of $p(y = j|x)$ for each value of $j = 1, \cdots, k$. Thus, the hypothesis leads to output of a k dimensional vector (whose elements' sum is 1), giving us k estimated probabilities:

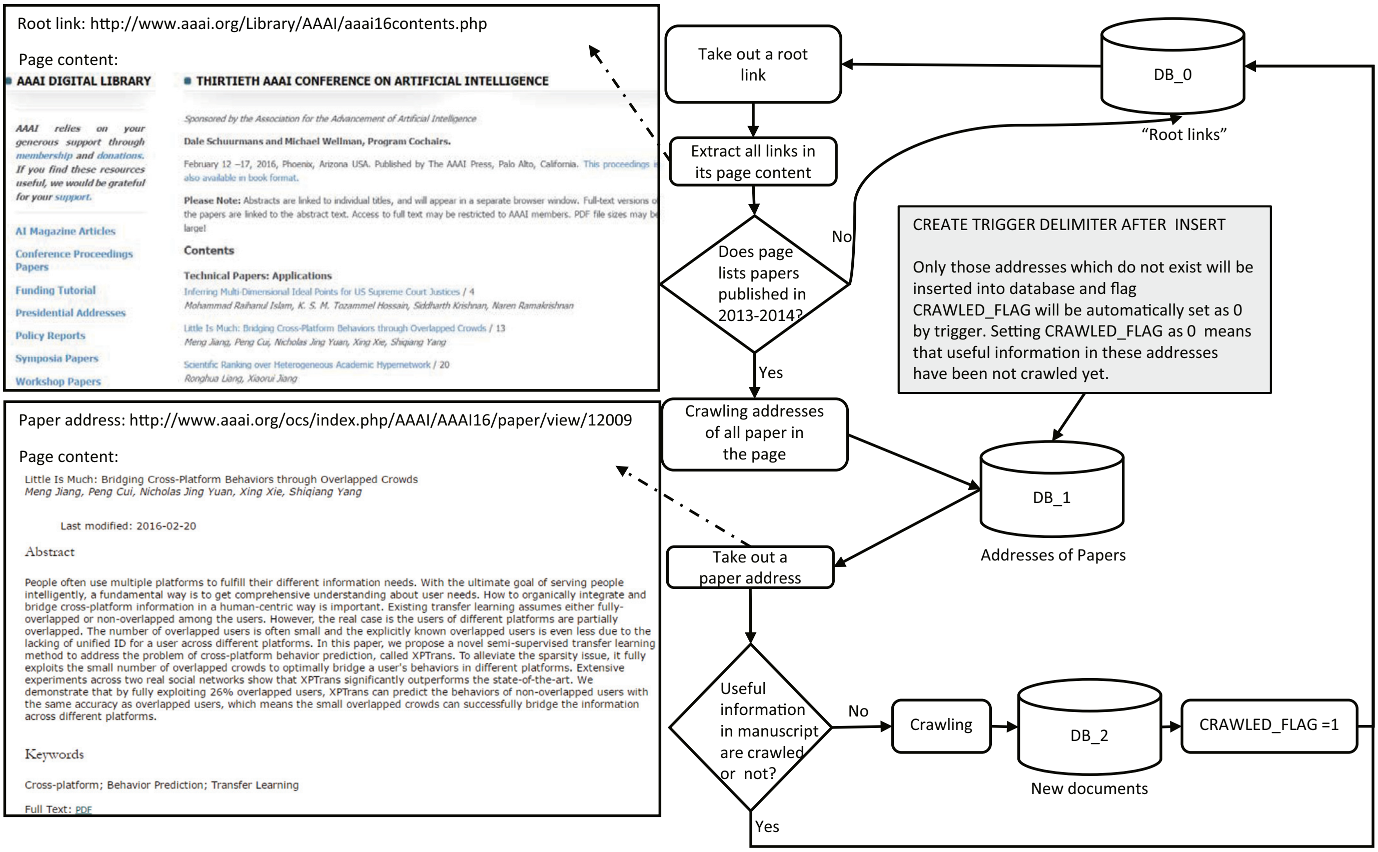
$$h_\theta(x) = \begin{bmatrix} p(y = 1|x; \theta) \\ p(y = 2|x; \theta) \\ \cdots \\ p(y = k|x; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x}} \begin{bmatrix} e^{\theta_1^T x} \\ e^{\theta_2^T x} \\ \cdots \\ e^{\theta_k^T x} \end{bmatrix} \tag{11}$$

where $\theta_1, \theta_2, \cdots, \theta_k \in R^{n+1}$ are parameters of the model. Because it is of low computational complexity and easy to perform, we selected softmax regression to predict recommendation results.

3.2.2. Proposed method

Softmax regression module generalizes logistic regression to classification problems where the class label y can take more than two possible values, which means it can be used for multi-class classification problems.

After feature selection procedure, a document of training data set is represented using $tf - idf$ according to the FV. The softmax regression is used as a classifier. When training Softmax, x and y of a sample (x, y) represent the feature vector and category of that sample respectively. Feature vector x is computed using Eq. (10), which is as the input data



Root link: http://www.aaai.org/Library/AAAI/aaai16contents.php

Page content:

AAAI DIGITAL LIBRARY

AAAI relies on your generous support through membership and donations. If you find these resources useful, we would be grateful for your support.

AI Magazine Articles
Conference Proceedings Papers
Funding Tutorial
Presidential Addresses
Policy Reports
Symposia Papers
Workshop Papers

THIRTIETH AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE

Sponsored by the Association for the Advancement of Artificial Intelligence

Dale Schuurmans and Michael Wellman, Program Cochairs.

February 12 –17, 2016, Phoenix, Arizona USA. Published by The AAAI Press, Palo Alto, California. This proceedings i also available in book format.

Please Note: Abstracts are linked to individual titles, and will appear in a separate browser window. Full-text versions o the papers are linked to the abstract text. Access to full text may be restricted to AAAI members. PDF file sizes may b large!

Contents

Technical Papers: Applications

Inferring Multi-Dimensional Ideal Points for US Supreme Court Justices / 4
Mohammad Raihanul Islam, K. S. M. Tozammel Hossain, Siddharth Krishnan, Naren Ramakrishnan

Little Is Much: Bridging Cross-Platform Behaviors through Overlapped Crowds / 13
Meng Jiang, Peng Cui, Nicholas Jing Yuan, Xing Xie, Shiqiang Yang

Scientific Ranking over Heterogeneous Academic Hypernetwork / 20
Ronghua Liang, Xiaorui Jiang

Paper address: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12009

Page content:

Little Is Much: Bridging Cross-Platform Behaviors through Overlapped Crowds
Meng Jiang, Peng Cui, Nicholas Jing Yuan, Xing Xie, Shiqiang Yang

Last modified: 2016-02-20

Abstract

People often use multiple platforms to fulfill their different information needs. With the ultimate goal of serving people intelligently, a fundamental way is to get comprehensive understanding about user needs. How to organically integrate and bridge cross-platform information in a human-centric way is important. Existing transfer learning assumes either fully-overlapped or non-overlapped among the users. However, the real case is the users of different platforms are partially overlapped. The number of overlapped users is often small and the explicitly known overlapped users is even less due to the lacking of unified ID for a user across different platforms. In this paper, we propose a novel semi-supervised transfer learning method to address the problem of cross-platform behavior prediction, called XPTrans. To alleviate the sparsity issue, it fully exploits the small number of overlapped crowds to optimally bridge a user's behaviors in different platforms. Extensive experiments across two real social networks show that XPTrans significantly outperforms the state-of-the-art. We demonstrate that by fully exploiting 26% overlapped users, XPTrans can predict the behaviors of non-overlapped users with the same accuracy as overlapped users, which means the small overlapped crowds can successfully bridge the information across different platforms.

Keywords

Cross-platform; Behavior Prediction; Transfer Learning

Full Text: PDF

CREATE TRIGGER DELIMITER AFTER INSERT

Only those addresses which do not exist will be inserted into database and flag CRAWLED_FLAG will be automatically set as 0 by trigger. Setting CRAWLED_FLAG as 0 means that useful information in these addresses have been not crawled yet.

Take out a root link
DB_0
"Root links"
Extract all links in its page content
Does page lists papers published in 2013-2014?
No
Yes
Crawling addresses of all paper in the page
DB_1
Addresses of Papers
Take out a paper address
Useful information in manuscript are crawled or not?
No
Crawling
DB_2
New documents
CRAWLED_FLAG =1
Yes

Fig. 2. Flowchart of the crawlers.

of Softmax. When testing the trained model, the way to extract the feature vector x of a document is exactly the same as training. The multi-label classification results are used to recommend for that document, which means that the predicted class from model is the recommended category. In specific, we chose the Top 3 classes according to the probability $p(y|x)$ instead of only one from our model's output as the final classification result. The reason is that the publication scopes of some computer journals and conferences have significant overlaps among them (e.g. International Conference on Computer Vision, IEEE Conference on Computer Vision and Pattern Recognition and IEEE Transactions on Image Processing). Therefore, if a manuscript falls into these publication scopes (such as image classification), its publication choice is often more than one. In addition, it will give the user more choices.

4. Experimental results and analysis

4.1. Dataset

In our experiments, the abstracts of computer science papers from different journals and conferences are collected for training. To get the abstracts and other information, a special automatic web crawler is constructed. We manually collected home page links as "root links" of 28 journals and 38 conferences (listed in supplementary material), which were ranked as A-class by the China Computer Federation (CCF) [26]. Fig. 2 describes how the automatic crawler works. Firstly, a root link is taken out from DB_0 which stores "root links". Then, all links are extracted from page content of the root link using a series of regular expressions. Thirdly, those links whose page content listing papers of specific years (e.g. year 2013–2014) will be kept and each address of papers in the paper list will be inserted into database DB_1. A flag CRAWLED_FLAG of the address will be set as 0 as long as the INSERT operation is successful (implemented by database trigger), which means that the address is a new one. In addition, a paper address is taken out from DB_1. If the address is not crawled before (i.e. its CRAWLED_FLAG = 0), useful information (such as abstracts, authors, etc.) in the manuscript will be crawled and stored into DB_2 as a new document for training. After that, its flag CRAWLED_FLAG will be set as 1, which means the address has been crawled. Finally, another root link is going to be taken out from DB_0 and analyzed accordingly. These procedures in the flowchart ensure that the crawler will work automatically and continuously to update the training dataset.

The distribution of training and testing dataset is shown in Supplementary Material. There are totally 14,012 records containing title, abstract, author and link of papers. To ensure the records in the dataset are correct, 20 percent of abstracts from each journal and conference are checked manually. Two-thirds of all abstracts are used as training samples, and the rest are used for testing. In experiment, papers published in 2014 and 2013 are considered. Specially, for some journals or conferences which did not publish enough papers between 2013 and 2014, papers published in other years are also considered.

To compute $tf - idf$ for each term, instead of searching documents, we use the inverted index to compute $tf - idf$ in the feature vector space. Fig. 3 is the inverted index construction figure.

4.2. Text preprocessing

Preprocessing not only can reduce the computational complexity but also improve the recommender performance. Moreover, text preprocessing is also necessary before generating the inverted index. Fig. 4 shows the flowchart of preprocessing in our system. The first step of preprocessing is tokenization with white space and punctuation. After that, we use stop-words (listed in Supplementary Material) to filter those meaningless tokens such as auxiliary verbs, prepositions, conjunctions, and interjections. There are also some tokens with the same word root but in different forms, e.g. "create": "created" and "creating".

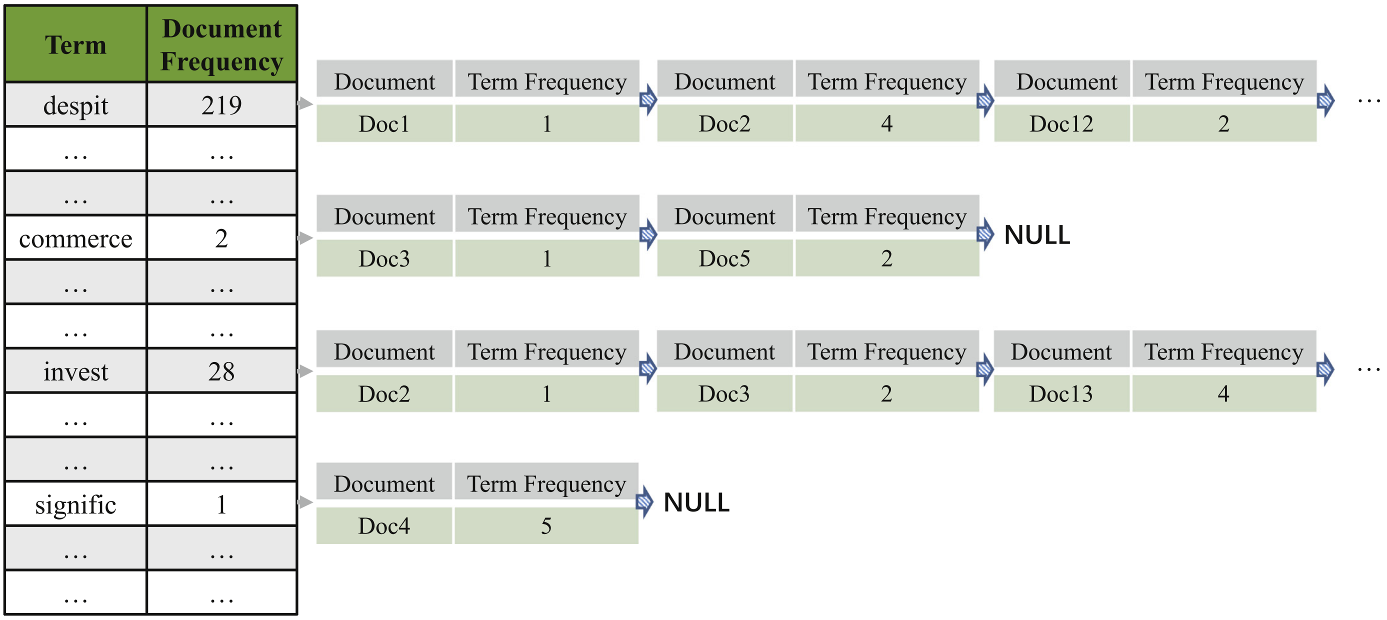


Fig. 3. Inverted index construction.

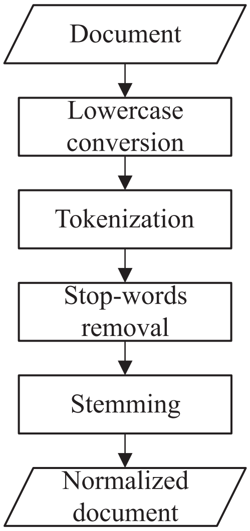


Fig. 4. Flowchart of document preprocessing.

Therefore, stemming is also required to degenerate different grammatical forms of a word to its root form [27]. We use the state-of-the-art stemming algorithm, that is, "Porter Stemmer" in our system [28].

4.3. Experiments and analysis

The new proposed system provides two kinds of recommendation results: one-class and three-class. The one-class (Top1) version recommends only one journal or conference, and it is very strict to evaluate the recommendation result. And the three-class (Top3) version shows three candidate journals or conferences for a manuscript (see Fig. 5), which shows the result of a query in our recommender system. The three-class (Top3) version is evaluated by selecting the top three categories as recommendation results. In other words, if one of the outputs of recommender system matches the publication journal or conference, it can be considered as a successful recommendation. The three-class version could provide more choices for users. In addition, different conferences or journals often share similar publication scopes, such as ICCV, CVPR, TIP, etc. Furthermore, some journal papers are the extended versions of conference papers.

To generate a good feature space, we performed several experiments on selecting the features for each category. Chi-square, MI and IG are implemented to make comparisons for feature selection. The top M terms are selected as the ith category feature vector to construct the FV^i. Considering the tradeoff between accuracy and efficiency, M is set as 200 according to experimental results shown in Fig. 8. Then, we combine $FV^1, FV^2, \cdots, FV^{N_c}$ and remove duplicated terms. For chi-square statistics, we achieve $N_{FV\text{-}Chi\text{-}Square} = 11,521$ feature space dimensions. For MI and IG, we also select the top 200 terms for each category and get $N_{FV\text{-}MI} = 12,696$ and $N_{FV\text{-}IG} = 6101$ dimensional feature spaces by combining and merging strategy, respectively.

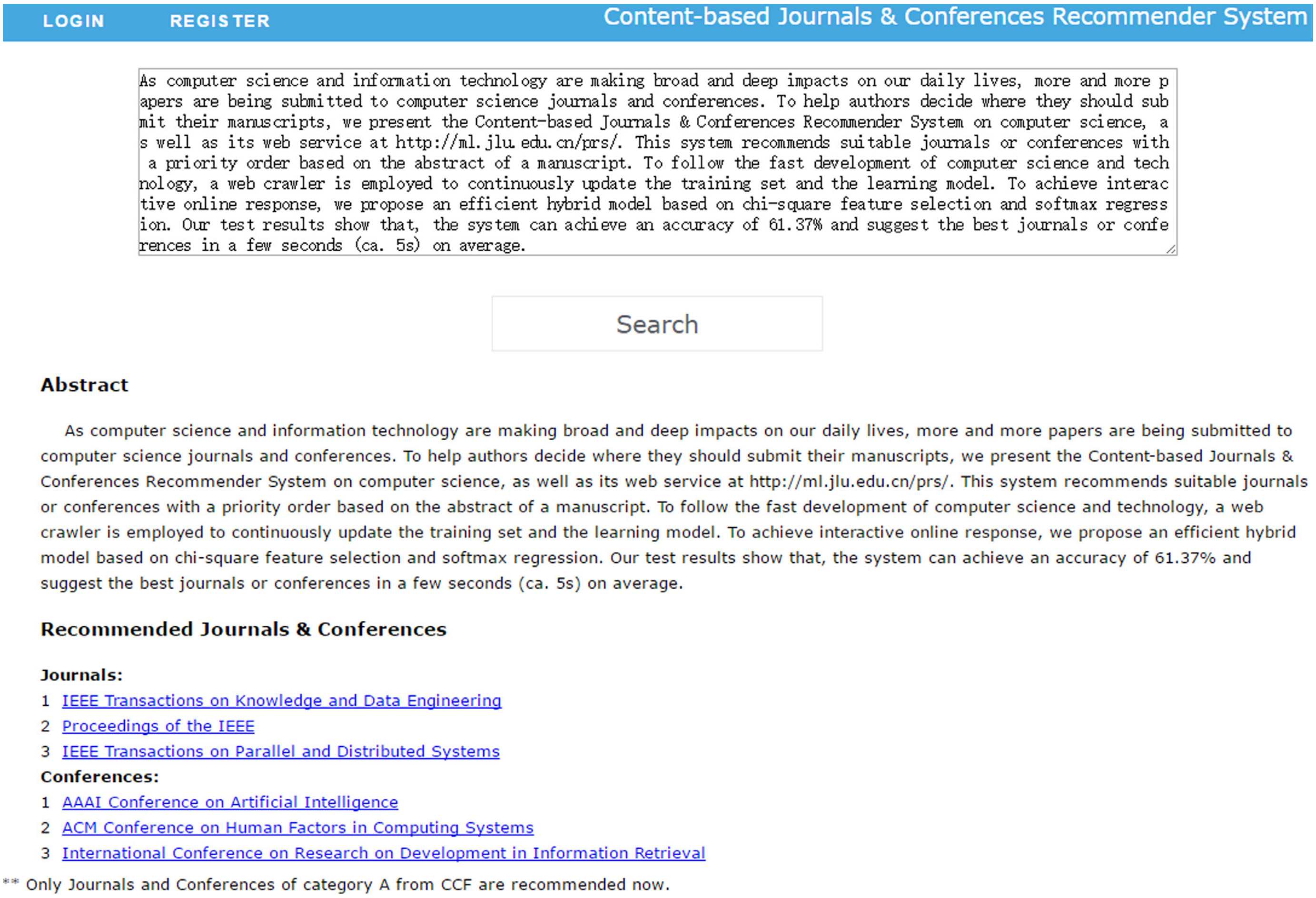
With respect to time complexity in testing phase, it can be calculated as follows. Suppose that N_w is the number of unique words or phases in an abstract, N_{FV} is the number of elements in feature vector and N_c is the number of conferences and journals in the system. In document preprocessing step, time complexity is $O(N_w)$. Because Inverse Index table has been constructed and stored into database ahead, so the calculation quantity of $tf - idf$ is constant $const$. Therefore, in feature vector computing step, time complexity is $O(N_{FV}*const)$. In softmax regression step, time complexity is $O(N_{FV}*N_c + N_c)$. Overall, the time complexity of predicting an abstract is $O(N_w + N_{FV}*const + N_{FV}*N_c + N_c) = O(N_w + N_{FV}*N_c)$.

Accuracy (Eq. (12)), F-measure (Eq. (13)) from [29] and ROC are used to evaluate the recommender system. Because our system uses multi-label classification model, we use macro-averaged ROC, which are defined in Eqs. (14) and (15).

$$Accuracy = \frac{\sum_{i=1}^{N} |P_i \bigcap G_i|}{\sum_{i=1}^{N} |G_i|} \tag{12}$$

$$F\text{-}measure = \frac{1}{N}\sum_{i=1}^{N}\frac{2|P_i \bigcap G_i|}{|P_i| + |G_i|} \tag{13}$$

$$TPR_{macro} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{14}$$



As computer science and information technology are making broad and deep impacts on our daily lives, more and more papers are being submitted to computer science journals and conferences. To help authors decide where they should submit their manuscripts, we present the Content-based Journals & Conferences Recommender System on computer science, as well as its web service at http://ml.jlu.edu.cn/prs/. This system recommends suitable journals or conferences with a priority order based on the abstract of a manuscript. To follow the fast development of computer science and technology, a web crawler is employed to continuously update the training set and the learning model. To achieve interactive online response, we propose an efficient hybrid model based on chi-square feature selection and softmax regression. Our test results show that, the system can achieve an accuracy of 61.37% and suggest the best journals or conferences in a few seconds (ca. 5s) on average.

Abstract

As computer science and information technology are making broad and deep impacts on our daily lives, more and more papers are being submitted to computer science journals and conferences. To help authors decide where they should submit their manuscripts, we present the Content-based Journals & Conferences Recommender System on computer science, as well as its web service at http://ml.jlu.edu.cn/prs/. This system recommends suitable journals or conferences with a priority order based on the abstract of a manuscript. To follow the fast development of computer science and technology, a web crawler is employed to continuously update the training set and the learning model. To achieve interactive online response, we propose an efficient hybrid model based on chi-square feature selection and softmax regression. Our test results show that, the system can achieve an accuracy of 61.37% and suggest the best journals or conferences in a few seconds (ca. 5s) on average.

Recommended Journals & Conferences

Journals:

1 IEEE Transactions on Knowledge and Data Engineering
2 Proceedings of the IEEE
3 IEEE Transactions on Parallel and Distributed Systems

Conferences:

1 AAAI Conference on Artificial Intelligence
2 ACM Conference on Human Factors in Computing Systems
3 International Conference on Research on Development in Information Retrieval

** Only Journals and Conferences of category A from CCF are recommended now.

Fig. 5. Results of three classes recommendations.

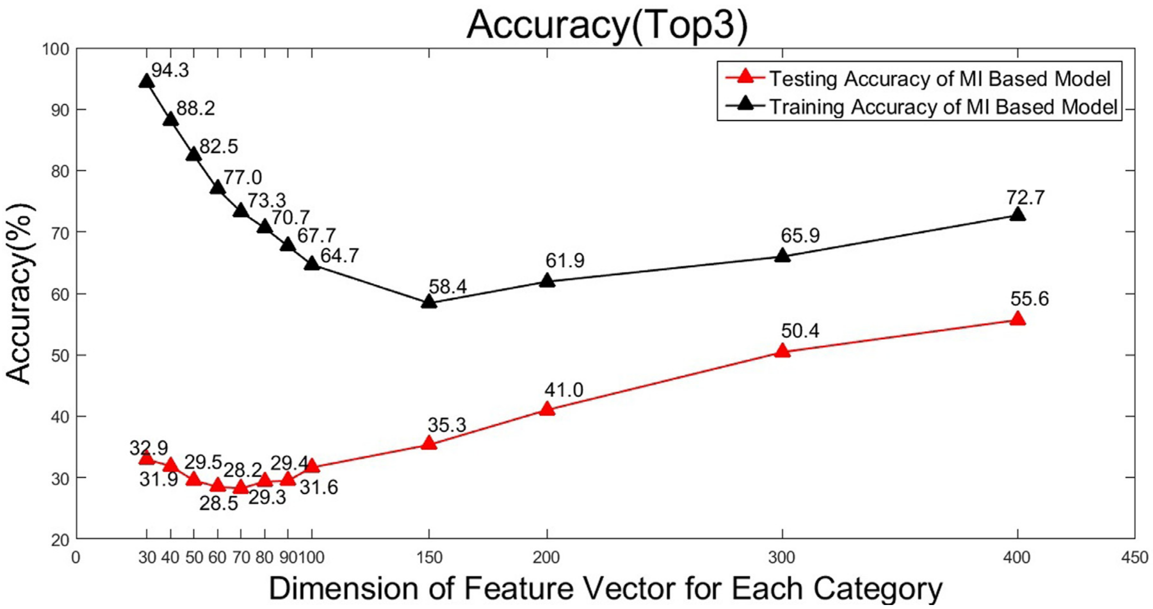


Fig. 6. Accuracy with different number of features for each category using MI.

$$FPR_{macro} = \frac{1}{N}\sum_{i=1}^{N}\frac{FP_i}{FP_i + TN_i} \tag{15}$$

where P_i is the set of test samples predicted as the *ith* category and G_i is the set of test samples labeled as the *ith* category. TP_i, FN_i, FP_i and TN_i are the number of true positives, false negatives, false positives and true negatives for *ith* category respectively.

Figs. 6–8 depict the accuracy of training and testing data with a different number of features for each category using MI, IG and chi-square feature selection models, respectively. Fig. 9 and Table 2 show the comparison results of the three feature-selection models.

In Fig. 6, it is obviously that the MI based model gets high accuracy

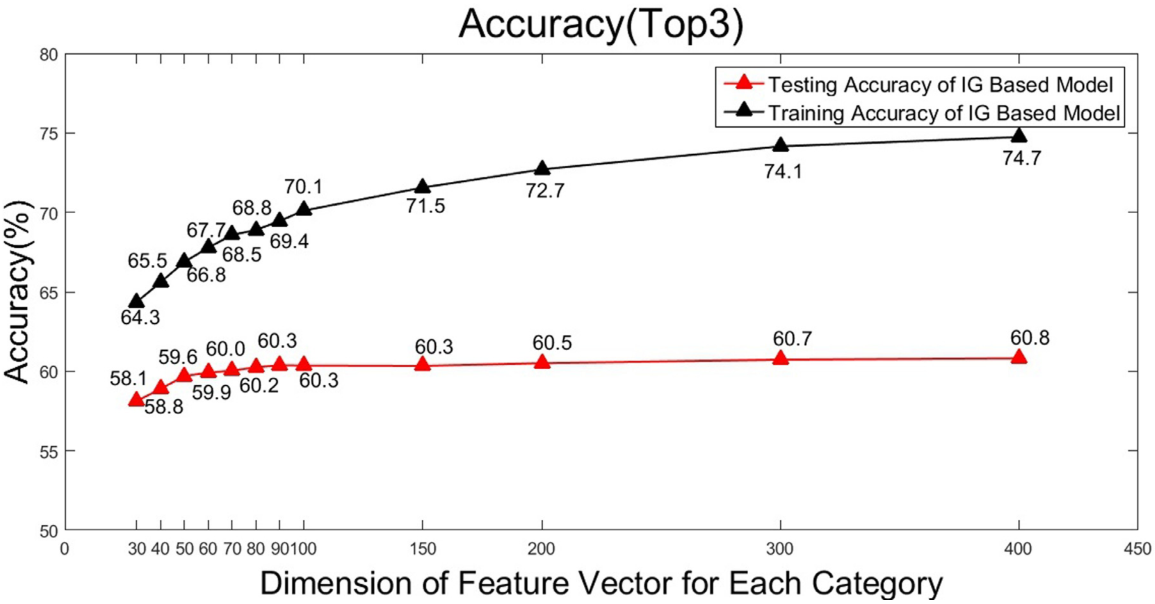


Fig. 7. Accuracy with different number of features for each category using IG.

on training set and low accuracy on testing set when the number of features is around 60, i.e. model with those features are not generalized enough. The reason is that noise features are selected by MI based model. Those noise features are well applicable for training set but not for testing set. It is the noise features that directly lead to the valley in the curve. When the number of elements in feature vector are larger 150, the model performs better gradually because the features selected by MI become discriminative for both training and testing set.

In Fig. 7 and 8, the models perform better with the number of elements in feature vector increasing. The curves of the two models do not show valleys because they do not select the noise features at the beginning. This concludes that features selected by IG and chi-squared are more discriminative than MI.

From Figs. 6–8, it can be seen that:

1) Using IG and chi-square selection method, the accuracy increases with the feature number, going up from 30 without decreasing. However, the accuracy of the MI based model has a 16.60% drop from the starting point to 70 features for each class. After the minimum, it climbs up as the feature number increases. But even with 400 features, the MI based model can only reach the accuracy of 55.7%, which is 7.9% and 9.3% less than the IG and chi-square based model, respectively.

2) The classification accuracy of a chi-square based model is always better than MI and IG. For example, at the beginning of testing accuracy curves, the chi-square based model was 78.7% and 1.2% higher than MI and IG, respectively; at the end, it achieves a higher 12.2% and 2.8% accuracy rating than the other two, respectively.

From Fig. 9, we can see that:

AUCs (Area Under Curve) of chi-squared and IG based models are much higher than MI based model. Chi-square has an AUC of 0.9404, while IG has 0.9415. Both chi-squared and IG have approximately 14.5% higher AUC than MI, which has 0.8273. From the comparison of AUC, it seems that MI is not a good choice for feature selection in publication recommendation. Hence, chi-square and IG are more suitable.

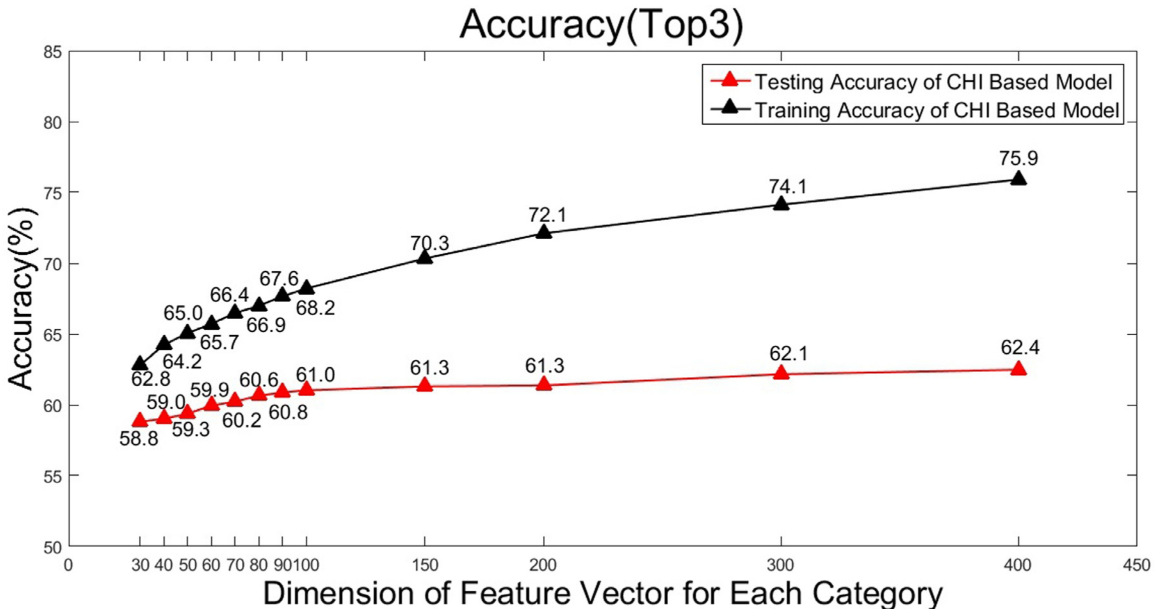


Fig. 8. Accuracy with different number of features for each category using Chi-square.

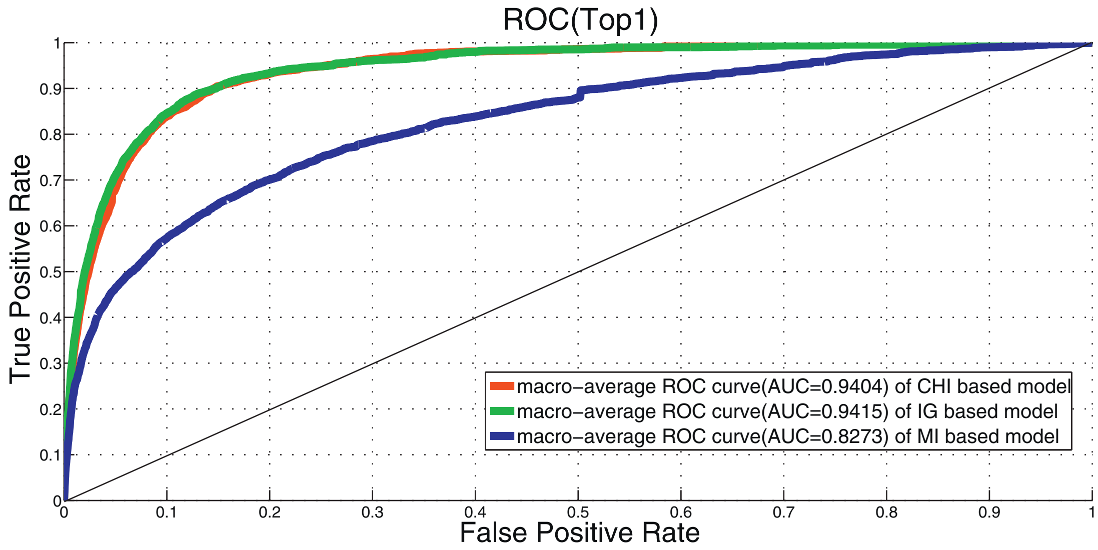


Fig. 9. Macro-averaged ROC of three feature selection models.

Table 2
Accuracy and F-measure of three feature selection models.

Metrics models	MI (Top1)	IG	CHI	MI (Top3)	IG	CHI
Accuracy(%)	20.55	34.06	**35.03**	41.03	60.51	**61.37**
F-measure	0.13	0.16	**0.18**	0.18	0.21	**0.23**

From Table 2, we can see that:

1) Results of Top3 are much higher than that of Top1. For example, the accuracy of chi-square based classification accuracy can reach 61.37% for the Top3 task, which is 75.2% higher than Top1. It is because that Top3 can give a higher solution space. In fact, different conferences or journals often share similar publication scopes, and the results of Top1 are strictly programmed to provide only one candidate.

2) The chi-square model achieves the highest accuracies and F-measures. For example, the accuracy of chi-square for Top3 is 61.37%, which is 49.6% higher than MI and 1.4% higher than IG. On F-measure, Chi-Square gets 0.23, which is 27.7% higher than MI and 9.5% higher than IG. This is because the features selected by chi-square for each category are less independent when compared to other categories. IG gets a 47.5% higher accuracy rating than MI and a 16.7% higher accuracy rating on F-measures. Meanwhile, for the strict Top1 tasks, chi-square achieves 35.03% and 0.18 on accuracy and F-measure, which are a 70.5% and 2.8% higher accuracy rating than MI and IG, respectively. It also performs better on F-measure, with a 38.2% and 13.1% higher than MI and IG. From the comparison of the experiment results, it also seems that chi-square and IG are more suitable than MI for feature selection in publication recommendation.

5. Conclusions

We developed a content-based journal and conference recommender system for computer science and technology. As far as we know, there is no similar recommender system or published method like what we have introduced here. Moreover, there was no dataset to use. Therefore, we first designed a web crawler to collect data and generate training and testing data sets. Then, we used different feature selection methods and performed several experiments to select a good strategy and reconstruct feature space. Finally, using a softmax regression model, the system shows a three class-recommendation solution to help user find out the candidate journal or conference.

In this paper, the proposed recommendation method is used for recommending computer science publications. It is worth mentioning that only abstract of a paper is used to recommend. The method could also be used by other e-library recommender systems[8]. For example, the results of the recommendation could be used to help readers to quickly determine the domain of a paper or to retrieve similar papers. The method could be also used in e-business applications. For example, a business user's preferences can be summarized as a text (like abstract of a paper) and some recommendations, such as the engaged business area, could be generated using our proposed method based on the text. These recommendations facilitate the decision process of a business user (e.g., buyer) in selecting qualified business partners (e.g., sellers) [18].

Although achieving 61.37% accuracy for paper recommendation, we believe that the accuracy and F-measure can be further improved, which will be our future work.

Acknowledgments

This study is supported by the National Natural Science Foundation of China (61602207, 61572228, 61472158), the National Basic Research Program of China (2015CB453000), and the Science Technology Development Project from Jilin Province (20160101247JC and 20140520070JH). Premier-Discipline Enhancement Scheme supported by Zhuhai Government and Premier Key-Discipline Enhancement Scheme supported by Guangdong Government Funds.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.knosys.2018.05.001

References

[1] M.-H. Chen, C.-H. Teng, P.-C. Chang, Applying artificial immune systems to collaborative filtering for movie recommendation, Adv. Eng. Inf. 29 (4) (2015) 830–839.

[2] Q. Diao, M. Qiu, C.-Y. Wu, A.J. Smola, J. Jiang, C. Wang, Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS), Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 193–202.

[3] W. Qu, K.-S. Song, Y.-F. Zhang, S. Feng, D.-L. Wang, G. Yu, A novel approach based on multi-view content analysis and semi-supervised enrichment for movie recommendation, J. Comput. Sci. Technol. 28 (5) (2013) 776–787.

[4] M. Schedl, D. Hauger, Tailoring music recommendations to users by considering diversity, mainstreaminess, and novelty, Proceedings of the 38th International ACM

SIGIR Conference on Research and Development in Information Retrieval, ACM, 2015, pp. 947–950.

[5] M. Kaminskas, F. Ricci, M. Schedl, Location-aware music recommendation using auto-tagging and hybrid matching, Proceedings of the 7th ACM conference on Recommender systems, ACM, 2013, pp. 17–24.

[6] J. Turcotte, C. York, J. Irving, R.M. Scholl, R.J. Pingree, News recommendations from social media opinion leaders: effects on media trust and information seeking, J. Comput. Mediated Commun. 20 (5) (2015) 520–535.

[7] F. Hopfgartner, B. Kille, A. Lommatzsch, T. Plumbaum, T. Brodt, T. Heintz, Benchmarking news recommendations in a living lab, International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2014, pp. 250–267.

[8] J. Lu, D. Wu, M. Mao, W. Wang, G. Zhang, Recommender system application developments: a survey, Decis. Support Syst. 74 (2015) 12–32.

[9] R. Armstrong, D. Freitag, T. Joachims, T. Mitchell, et al., WebWatcher: a learning apprentice for the world wide web, AAAI Spring symposium on Information gathering from Heterogeneous, distributed environments, (1995), pp. 6–12.

[10] M. Balabanovic, Y. Shoham, Learning information retrieval agents: experiments with automated web browsing, On-line Working Notes of the AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments, (1995), pp. 13–18.

[11] L. Terveen, W. Hill, B. Amento, D. McDonald, J. Creter, PHOAKS: a system for sharing recommendations, Commun. ACM 40 (3) (1997) 59–62.

[12] H. Kautz, B. Selman, M. Shah, Referral web: combining social networks and collaborative filtering, Commun. ACM 40 (3) (1997) 63–65.

[13] T. Joerding, A temporary user modeling approach for adaptive shopping on the web, Proceedings of Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, Toronto and Banff, Canada. Computer Science Report, (1999), pp. 99–107.

[14] B. Moore, R. Janker, B. Papez, L. Power, R. Watkins, Migrating weblogic applications to websphere advanced edition, IBM Redbooks, (2001), p. 1.

[15] H. Jafarkarimi, A.T.H. Sim, R. Saadatdoost, A naive recommendation model for large databases, Int. J. Inf. Educ. Technol. 2 (3) (2012) 216.

[16] P. Melville, V. Sindhwani, Recommender systems, Encyclopedia of Machine Learning, Springer, 2011, pp. 829–838.

[17] R.J. Mooney, L. Roy, Content-based book recommending using learning for text categorization, Proceedings of the Fifth ACM Conference on Digital Libraries, ACM, 2000, pp. 195–204.

[18] Q. Shambour, J. Lu, A trust-semantic fusion-based recommendation approach for e-business applications, Decis. Support Syst. 54 (1) (2012) 768–780.

[19] M. Mao, J. Lu, G. Zhang, J. Zhang, Multirelational social recommendations via multigraph ranking, IEEE Trans. Cybern. 47 (12) (2017) 4049–4061.

[20] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl. Based Syst. 46 (2013) 109–132.

[21] B. Chen, H. He, J. Guo, Constructing maximum entropy language models for movie review subjectivity analysis, J. Comput. Sci. Technol. 23 (2) (2008) 231–239.

[22] N. Gupta, P. Saxena, J. Gupta, Document summarisation based on sentence ranking using vector space model, Int. J. Data Min. Model. Manag. 5 (4) (2013) 380–406.

[23] R. Costa, C. Lima, Document clustering using an ontology-based vector space model, Int. J. Inf. Retr. Res. 5 (3) (2015) 39–60.

[24] C. Jin, T. Ma, R. Hou, M. Tang, Y. Tian, A. Al-Dhelaan, M. Al-Rodhaan, Chi-square statistics feature selection based on term frequency and distribution for text categorization, IETE J. Res. 61 (4) (2015) 351–362.

[25] A. Moh'd A Mesleh, Chi square feature extraction based SVMs arabic language text categorization system, J. Comput. Sci. 3 (6) (2007) 430–435.

[26] CCF recommended ranking, 2015.

[27] M.A. Otair, Comparative analysis of arabic stemming algorithms, Int. J. Manag. Inf. Technol. 5 (2) (2013) 1.

[28] M.F. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.

[29] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv. 47 (3) (2015) 52.