Domain-Specific Term Rankings Using Topic Models

Zhiyuan Liu and Maosong Sun

Department of Computer Science and Technology
State Key Lab on Intelligent Technology and Systems
National Lab for Information Science and Technology
Tsinghua University, Beijing 100084, China
lzy.thu@gmail.com, sms@tsinghua.edu.cn

Abstract. A widely used approach for keyword extraction and content-based tag recommendation is ranking terms according to some statistical criteria. In many cases documents such as news articles and product reviews are in some specific domains. Domain knowledge may be important information for term rankings. In this paper, we present to model domain knowledge using latent topic models, referred to as Domain-Topic Model (DTM). Using DTM we perform domain-specific term rankings according to the relatedness between terms and domains. Experimental results on both keyword extraction and tag recommendation show advantages of DTM for domain-specific term rankings.

Keywords: Domain-Topic Model, term ranking, keyword extraction, social tag recommendation.

1 Introduction

Keyword extraction for documents such as research papers and news articles is widely used in digital library and information retrieval. Content-based tag recommendation for web resources such as blogs and product reviews is an important application in Web 2.0, which can ease the process of social tagging for users. Since both keyword extraction and content-based tag recommendation seek to identify most representative terms from document contents, ranking terms thus plays a crucial role in the both applications.

Previous methods simply rank terms by frequency measures such as $tfidf$ or χ^2 statistics, which are found to lead to poor results. In recent years, there is a surge of studies on graph-based methods for term rankings. These methods apply graph-based ranking algorithms, such as PageRank [4] and HITS [15], to rank terms in given document and select those with the highest ranking values as keywords or tags [20].

Domain knowledge is the knowledge of some specialized disciplines. In practice, there is usually domain information in addition to documents, which indicates the domain knowledge shared by the document and other documents in the same domain. The domain of a document can be either large or small. Here are two examples of domain information of documents. For news websites such as CNN.com, news articles are classified into different categories such as *sports*, *science*, *business* and *politics*. These category-like domains are large. Meanwhile, these news articles are also organized according to various popular themes, such as *financial crisis*, *air pollution* and *world cup*

P.-J. Cheng et al. (Eds.): AIRS 2010, LNCS 6458, pp. 454–465, 2010.

football match. These theme-like domains are small. Similar to news articles, for online bookstores such as `Amazon.com`, book reviews are divided into categories such as *arts*, *literature*, *philosophy* and *history*. The reviews under the same category share a large category-like domain knowledge. Meanwhile, according to some popular themes, various books on the same themes are collected together and their reviews share a small theme-like domain.

Generally, for a document under a specific domain, we are usually more interested in the part that is related to the domain. Domain knowledge of a document may thus play an important role in keyword extraction and tag recommendation, which can help focus on the important part of documents and filter out noise. In order to apply domain knowledge, we have to find a method to represent both terms and domains, and then rank terms by measuring their relatedness with given domains. In this paper, we use latent topic models to represent terms and domains, and then perform domain-specific term rankings.

The key idea of latent topic models is to represent terms and document as a mixture over latent topics. In a pioneer work [6], a topic model, i.e., probabilistic latent semantic analysis (pLSA), is proposed to rank documents in a corpus. Latent Dirichlet allocation (LDA) [3] was further proposed by adding Dirichlet priors on pLSA to avoid overfitting problem. The success of latent topic models for modeling documents motivates us to propose a new latent topic model, referred to as Domain-Topic Model (DTM), for modeling domain knowledge of documents. Given a set of documents where each document has one or more domain labels, either category-like or theme-like, DTM learns topic distributions of words as well as topic distributions of domains. Using a learned DTM, we can rank term w in domain c by comparing their topic distributions. In this paper we investigate various measures of computing relatedness between domains and terms using their topic distributions.

2 Related Work

Existing methods for keyword extraction can be categorized into supervised classification-based approach and unsupervised ranking-based approach. For the former approach, some researchers adopted supervised learning methods to build two-class (is a keyword or not) classifiers to identify keywords, which is first proposed by [27] and the following supervise-based methods mostly considered more linguistic features of terms for developing classification system [12]. Since the supervised approach needs manually annotated training set, it is sometimes not practical under web circumstance. Starting from TextRank [20], graph-based ranking methods are becoming the most widely used ranking approach for keyword extraction. In recent years, many extensions of TextRank have been proposed for keyword extraction [17,29,28,18]. Besides, clustering methods on word graphs are also proposed for keyword extraction [19,10]. Unsupervised ranking terms is a practical approach for keyword extraction.

Some researchers have noticed the importance of domain knowledge [8,13]. In these work, however, human annotations are required to reflect domain knowledge in keyword extraction, either by building separate training set of different domains for classifiers [8] or using human-defined domain specific thesaurus as external knowledge [13].

The method in this paper, in contrast, can extract domain knowledge in an unsupervised manner.

Social tagging, as known as folksonomy, is a popular approach to organize online resources like documents, bookmarks and photos. Tag recommendation systems are usually designed to ease the process of social tagging, which can suggest tags to a new object based on previous tagged objects, user preferences or keyword extraction from the text content of the object. Most tag recommendation methods focus on analyzing the relations between object, tags and users. For some special online resources such as blogs and product reviews, it is may be practical to recommend tags based on contents. Many supervised methods have been proposed for content-based tag recommendation [21,14,26]. None of these methods, however, have taken domain knowledge into account for content-based tag recommendation. Since we focus on investigating the usefulness of domain knowledge for term rankings, our method is thus only compared with TextRank, without considering the supervised methods mentioned above.

3 Domain-Specific Term Rankings Using Domain-Topic Model

3.1 Domain-Topic Model

DTM models documents composed of words and domain labels. Each document d consists of N_d words $\boldsymbol{w}_d = \{w_{d,1}, \ldots, w_{d,N_d}\}$ and L_d domain labels $\boldsymbol{y}_d = \{y_{d,1}, \ldots, y_{d,L_d}\}$. The words are selected from a vocabulary of V unique words, and the domain labels are selected from a set of C unique labels.

Under the assumption that most words in a document describe the domain information of the document, we define DTM as a generative process as follows:

1. For each domain $c \in \{1, \ldots, C\}$, choose $\theta^{(c)} \sim Dir(\alpha)$.
2. For each topic $k \in \{1, \ldots, T\}$, choose $\phi^{(k)} \sim Dir(\beta)$.
3. For each document $d \in \{1, \ldots, D\}$, given the domain labels $\boldsymbol{y}_d$ For each word w_i, out of the N_d words:
   (a) Choose a domain label $u_i \sim Uni(\boldsymbol{y}^d)$
   (b) Choose a topic $z_i \sim Mul(\theta^{(u_i)})$
   (c) Choose a word $w_i \sim Mul(\phi^{(z_i)})$

where $Dir(*)$, $Uni(*)$ and $Mul(*)$ indicate the Dirichlet distribution, Uniform distribution and Multinomial distribution, respectively. In the generative process, θ_c is the topic distributions of domain c, drawn from a symmetric Dirichlet prior α; ϕ_k is the word distributions of topic k, drawn from a symmetric Dirichlet prior β. α and β are predefined as prior knowledge. Empirically we set $\alpha = 50/T$ and $\beta = 0.01$ where T is the topic number [9].

Figure 1 shows the graphical representation of DTM in comparison with LDA. DTM is identical with Author-Topic Model (ATM) [25] in form, while ATM was proposed for modeling documents with authors.

Let $\boldsymbol{w}$, $\boldsymbol{z}$ and $\boldsymbol{u}$ be the vectors of all words and their topic and domain label assignments of the whole document collection D. Given observed words and domain labels of a set of documents, the problem is inferring the hidden topics z. Since estimating posterior is intractable, various approximate methods could be used to estimate DTM.

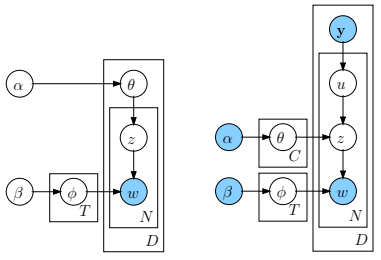

Fig. 1. Graphical model representations of LDA and DTM

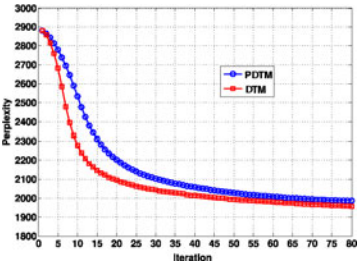

Fig. 2. Comparison between PDTM ($P = 10$) and DTM on perplexity versus Gibbs Sampling iterations

In this paper, we use Gibbs Sampling [9] to learn DTM by sampling u_i and z_i for each word w_i. The topic assignment for a particular word depends on the current topic assignments of all the other words. That is, the domain and topic of a particular word w_i in document d with domain labels $\boldsymbol{y}_d$ is sampled from the following distribution:

$$\mathrm{P}(u_i = c, z_i = k \mid \boldsymbol{w}, \boldsymbol{z}_{-i}, \boldsymbol{u}_{-i}, \boldsymbol{y}_d, \alpha, \beta) = \frac{n_{-i,k}^{(w_i)} + \beta}{\sum_{w'} n_{-i,k}^{(w')} + V\beta} \frac{n_{-i,k}^{(c)} + \alpha}{\sum_{k'} n_{-i,k'}^{(c)} + T\alpha} \quad (1)$$

where $\boldsymbol{w}_{-i}$, $\boldsymbol{z}_{-i}$ and $\boldsymbol{u}_{-i}$ are all other words in $\boldsymbol{w}$ except current word w_i and their corresponding topic and domain label assignments, $n_{-i,k}^{(w_i)}$ is the number of times that topic k is assigned to word w_i, and $n_{-i,k}^{(c)}$ is the number of times that topic k is assigned to domain c. The $-i$ indicates the counts are taken by not including the value of w_i.

When finishing Gibbs Sampling, we obtain the model consisting of both topic distributions of each word w, i.e., $\phi^{(w)}$, and topic distributions of each domain c, i.e., $\theta^{(c)}$. The probabilities of word w and domain c given topic k are computed as follows:

$$\mathrm{P}(w|k) = \phi_k^{(w)} = \frac{n_k^{(w)} + \beta}{\sum_{w'} n_k^{(w')} + V\beta}, \quad \mathrm{P}(c|k) = \theta_k^{(c)} = \frac{n_k^{(c)} + \alpha}{\sum_{k'} n_{k'}^{(c)} + T\alpha} \quad (2)$$

where $n_k^{(w)}$ is the total number of times that topic k is assigned to word w, and $n_k^{(c)}$ is the total number of times that topic k is assigned to domain c. Correspondingly, we can also obtain the probabilities of topic k given word w, i.e., $\mathrm{P}(k|w)$ and given domain c, i.e., $\mathrm{P}(k|c)$ using $n_k^{(w)}$ and $n_k^{(c)}$ for all words, topics and domains. These will be used for domain-specific term rankings, which is introduced in the following section.

3.2 Distributed Gibbs Sampling of DTM

To handle large-scale learning tasks, in this paper we implement a parallel version of DTM using the MapReduce parallel programming model [7], namely PDTM, where training documents are distributed over distinct processors for distributed Gibbs Sampling. The basic idea of PDTM is motivated by [22]. Here is an overview of PDTM

algorithm. Given P processors, we partition documents and corresponding assignments into P disjoint subsets. Then simultaneous Gibbs Sampling is performed independently on each processor, as each processor thinks it is the only processor and updates it own copy of DTM. After each Gibbs Sampling iteration, each processor outputs its update to the model. Then an operation is performed to collect and merge the outputs from all processors, and broadcast the new model to all processors.

Our experiments consistently showed that the convergence rate for the distributed algorithms is as rapid as for the single processor case. As an example, Figure 2 shows test perplexity versus iteration number of Gibbs Sampling (NIPS data, $T = 50$ and $P = 10$). Despite having no guaranteed formal convergence, PDTM works very well empirically.

3.3 Ranking Terms Using DTM

To investigate the performance of DTM for domain-specific term ranking, we focus two important applications, keyword extraction and content-based tag recommendation.

A term for rankings usually contains more than one words. For a given term t, we compute

$$\mathrm{P}(t|k) = \prod_{w \in t} \mathrm{P}(w|k) \tag{3}$$

and

$$\mathrm{P}(k|t) = \frac{\mathrm{P}(t|k)\,\mathrm{P}(k)}{\mathrm{P}(t)} = \frac{n_k}{n_t} \prod_{w \in t} \mathrm{P}(w|k) \tag{4}$$

where the fraction $\frac{\mathrm{P}(k)}{\mathrm{P}(t)}$ is approximated by the number of times that topic k occurs in training corpus, i.e., n_k, divided by the number of words in term t, i.e., n_t.

After obtaining $\mathrm{P}(t|k)$ and $\mathrm{P}(k|t)$, using DTM we can rank term t given domain c by *characteristics* measure [6]:

$$\mathrm{char}(t;c) = \mathrm{P}(t|c)\,\mathrm{P}(c|t) = \left[\sum_{k=1}^{T} \mathrm{P}(t|k)\,\mathrm{P}(k|c)\right]\left[\sum_{k=1}^{T} \mathrm{P}(c|k)\,\mathrm{P}(k|t)\right] \tag{5}$$

This measure can identify what terms are *characteristic* of a domain c. A rarely-used term that is mainly used in domain c is not characteristic of c; neither is a term with high frequency in domain c, if it is also heavily-used by all other domains. The idea is similar to the measure of $tfidf$. That is, the former factor is controlled by $\mathrm{P}(t|c)$ and the latter is by $\mathrm{P}(c|t)$.

Given a domain, we can also rank terms by KL-divergence between the topic distributions of terms and the domain. KL-divergence is a asymmetric measure of the difference between two probability distributions by computing the expected extra information of one distribution compared to another distribution. By representing both terms and domains using topic distributions, we can use KL-divergence to measure the semantic diversity between terms and domains:

$$\mathrm{D_{KL}}(t||c) = \sum_{k=1}^{T} \mathrm{P}(k|t) log \frac{\mathrm{P}(k|t)}{\mathrm{P}(k|c)} \tag{6}$$

The third method to rank terms is predictive likelihood [11]:

$$\mathrm{P}(t|c) = \sum_{k=1}^{T} \mathrm{P}(t|k)\,\mathrm{P}(k|c) \tag{7}$$

This is a part of the formula (Eq. 5) of characteristics, which indicates how likely term t occurs under domain c.

4 Experiment Results

In this section, we evaluate DTM on both keyword extraction and content-based tag recommendation. Since we focus on investigating the usefulness of domain-specific rankings, we only compare our methods with other content-based methods, i.e., TextRank and LDA for evaluation.

4.1 Keyword Extraction

The experiments of keyword extraction using DTM are carried out on a dataset of English news articles. However, we should note that the method is language independent. We first introduce the dataset and the training process of DTM.

Dataset. The dataset of news articles is annotated by Wan and Xiao [29], based on 308 news articles in DUC2001 [23]. The news articles are divided into 30 document sets. Each set is on an event or theme, which is regarded as a specific domain in experiments.

The news articles are too few to learn DTM. We construct a corpus consisting of both the news articles and English Wikipedia to learn DTM. Since the articles in Wikipedia are supposed to compile all human knowledge, high quality topics can thus be learned by taking Wikipedia as background. In experiments, Wikipedia articles come from the March 2008 snapshot. After removing non-article pages and removing articles shorter than 100 words, we collected $2,122,618$ articles. After tokenization, stop word removal and word stemming, we build the vocabulary by selecting $20,000$ words with top document frequency value. We learn DTM on the corpus by taking the set of each news article as its domain label, as well as taking the title of each Wikipedia article as its unique domain label. In this paper, we use the learned DTM with the number of topics $T = 1,000$ selected by cross validation.

Results and Evaluation. Keyword is a term which may be a single word or a multi-word phrase. Not all words or phrases in document are possible to be selected as keywords. In order to filter out some noisy words and phrases in advance, we select candidate keywords using some heuristic rules. This step proceeds as follows. Firstly, the document is tokenized for English or segmented into words for Chinese and other languages without word-separators. As reported in [12], most manually assigned keywords turn out to be noun phrases. Therefore, we annotate the document with POS tags using Stanford Log-Linear Tagger [1]. Then we extract the noun phrases

[1] The package could be accessed from `http://nlp.stanford.edu/software/tagger.shtml`. In this paper, we select the English tagging model "left3words-distsim-wsj-clean".

Table 1. Comparing TextRank, LDA and DTM when $m = 10$

Methods	Correct No.	Precision	Recall	F-measure
DTM+CHAR	821	0.269	0.331	0.297
DTM+KL	862	0.282	0.348	0.311
DTM+PSC	808	0.264	0.326	0.292
LDA+CHAR	772	0.253	0.311	0.279
LDA+KL	802	0.262	0.324	0.290
LDA+PSC	766	0.251	0.309	0.277
TextRank	744	0.243	0.300	0.269

whose pattern is zero or more adjectives followed by one or more nouns, represented as `(adjective)*(noun)+`. These noun phrases are regarded as the candidate keywords of the document.

Using DTM, for each candidate keyword, we compute its relatedness with the domain using characteristics, KL-divergence and predictive likelihood. With the relatedness, we carry out domain-specific rankings for keyword extraction. In contrast with DTM, using LDA, we are only able to measure the relatedness between candidate keywords and the document according to their topic distributions. For TextRank, we follow the implementation in [29].

For evaluation, the keywords extracted by different methods are compared with manually labelled keywords. All words in a keyword are first reduced to their base forms for comparison. In this paper, we use Porter Stemmer [2] to complete the process. The precision, recall and F-measure are used as evaluation metrics, which are widely used in keyword extraction task.

The experiment results of TextRank, LDA and DTM when extracting 10 keywords are shown in Table 1, where "CHAR", "KL" and "PSC" indicate characteristics, KL-divergence and predictive likelihood for measuring relatedness, respectively. It is clear that LDA outperforms TextRank and DTM outperforms both TextRank and LDA, which indicates the effect of domain knowledge. We also note that KL-divergence outperforms other measures in both LDA and DTM. In Figure 3, we demonstrate the precision, recall and F-measure with different number of keywords from 1 to 20 for TextRank, LDA and DTM (using KL-divergence). We can see that DTM always outperforms TextRank and LDA especially on precision when the number of keywords are small.

We further demonstrate an example of extracted keywords from a news article by TextRank, LDA and DTM. The title of this article is *Commodities and Agriculture: Investors sought for Zimbabwe diamond mine*. The domain of the article is on *diamond mines and diamond trading*. As shown in Table 2, top five keywords extracted by the three methods is listed in descending order of ranking values. Compared to standard answers, these keywords are identified with $+$ and $-$ marks.

Because TextRank only uses the cooccurrence information within the article for ranking terms, it can not properly measure the relatedness between terms and the article. TextRank thus can not identify most representative terms of the article. As shown in

[2] The package could be accessed from `http://tartarus.org/~martin/PorterStemmer/`

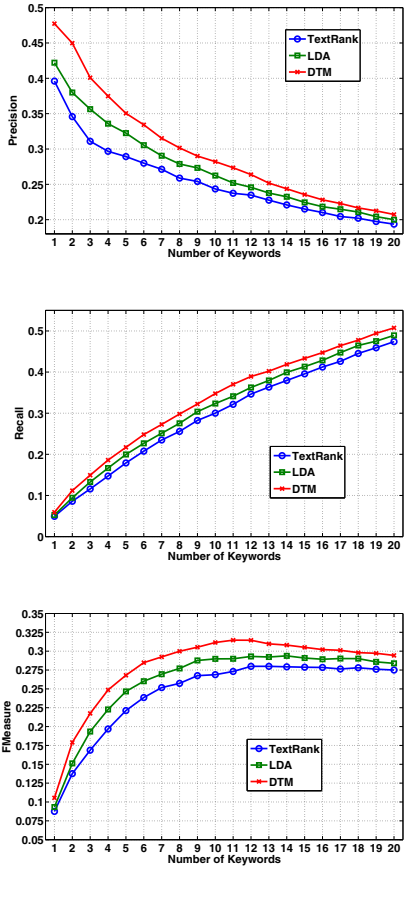

Fig. 3. Comparing TextRank, LDA and DTM for keyword extraction with different number of extracted keywords

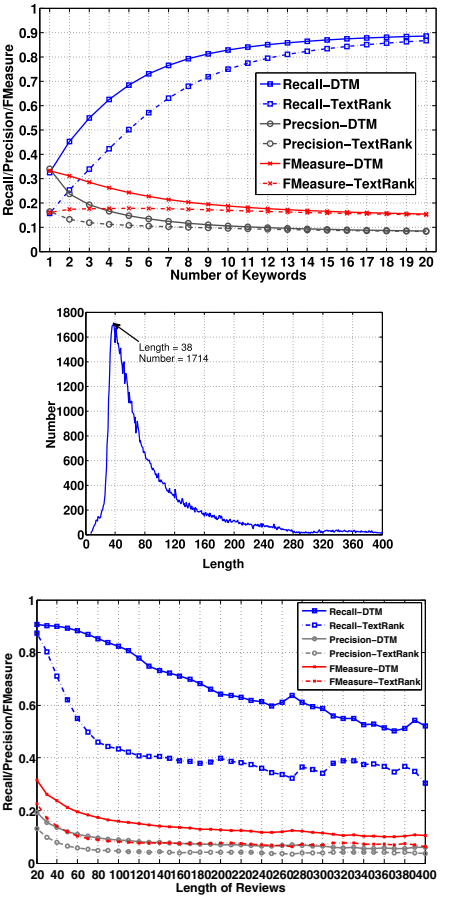

Fig. 4. Comparing DTM and TextRank on tag recommendation for restaurant reviews

Table 2, TextRank selects the heavily-connected *de beers* in word graph as a keyword, which in fact does not match the theme of the article. By representing terms and the article using their topic distributions, LDA easily identifies the most theme-relevant terms of the article. Nevertheless, since there are some "noisy" parts in the article which may have nothing to do with the domain of this article, LDA selects *gem quality* as a keyword by mistake. DTM, however, can avoid the mistake by taking the domain knowledge of the article into consideration and make all of top five extracted keywords correct. We also show top five topics of article FT933-8941 and its domain in Table 3 and Table 4 for comparison. These topics are represented by their top three characteristic words. We find that both the domain and the article focus on Topic-1307 on *diamond*. The domain, however, pays more concentration of Topic-1307 with probability 0.1261 compared to 0.0648 of the article on this topic. This makes sure that DTM is more robust against noises.

Table 2. Top 5 keywords extracted by TextRank, LDA and DTM of article FT933-8941

Methods	Keywords
TextRank	de beers official (-), de beers (-), diamond mine (+), mr robin baxter-brown (-)
LDA	diamond business (+), gem quality (-), diamond deposit (+), diamond trade (+), diamond (-)
DTM	diamond mines (+), diamond trade (+), diamond deposit (+), diamond exploration experience (+), diamond mine (+)

Table 3. Top 5 topics of article FT933-8941 and their characteristic words by LDA

Rank	Topic	$\mathrm{P}(k\|c)$	Characteristic words of k
1	1307	0.0648	diamond, logan, veronica
2	0899	0.0409	company, ceo, acquisition
3	0133	0.0405	fund, money, pay
4	1500	0.0336	zimbabwe, rhodesia, angola
5	0632	0.0302	beer, brewer, brew

Table 4. Top 5 topics of article FT933-8941 and their characteristic words by DTM

Rank	Topic	$\mathrm{P}(k\|c)$	Characteristic words of k
1	1307	0.1261	diamond, logan, veronica
2	1500	0.0620	zimbabwe, rhodesia, angola
3	0632	0.0544	beer, brewer, brew
4	0378	0.0531	price, cost, demand
5	0379	0.0501	sector, export, econom

From the evaluation results and examples, compared to TextRank and LDA, we show the robust advantages of domain-specific term rankings for keyword extraction using DTM in the news dataset. In the following section, we will further investigate the performance of DTM for content-based tag recommendation.

4.2 Content-Based Tag Recommendation

In this section, we try to recommend tags for restaurant reviews based on their contents. The reviews were crawled from Dianping (`http://www.dianping.com`), the largest Web review service website in China.

Dataset. The review dataset contains $108,161$ restaurant reviews in Chinese. The dataset itself is also too small to learn DTM. We thus combine more web articles with the review dataset together as the corpus for learning DTM, where the web article dataset contains more than 2 million articles. In this corpus, all restaurant reviews are labeled with domain label *restaurant*, and each web article takes its ID as its unique domain label. After word segmentation and stop word removal, we select $50,000$ words with top document frequency as the vocabulary. We set the number of topics $T = 800$ to learn DTM.

Results and Evaluation. As described in Section 3.3, given the trained DTM, we rank each term in the review given the domain of *restaurant*. In the experiment using TextRank, we follow the approach described by [20] to build a word graph for each review and rank terms using their PageRank values.

Most Dianping reviews have user-labeled tags. For experiments, we only keep the tags that occur in the corresponding reviews. The $108,161$ reviews used here are all selected with one or more tags that occur in the corresponding reviews. Among the reviews, $103,068$ are labeled with one tag, $4,856$ with two keywords and 237 with three or more tags.

We select top-m words from review using DTM and TextRank respectively. Here we do not list the result obtained by LDA due to its poor performance, whose max F-measure value is only 14.2% for m from 1 to 20. This also indicates the importance of modeling domain information for ranking. Similar to the results in Section 4.1, KL-divergence also outperforms other measures in domain-specific term rankings using DTM. In the following results, we only demonstrate the results of DTM using KL-divergence. Figure 4 (Top) compares DTM and TextRank with respect to various number of keywords. From this figure, we can see that DTM outperforms TextRank in all three measures. When m is small, this advantage is especially salient. We also examine the effectiveness of DTM and TextRank with respect to document length. The distribution of document length in Dianping dataset is shown in Figure 4 (Middle), and most reviews have 20 to 100 words. As shown in Figure 4 (Bottom), when $m = 10$, DTM outperforms TextRank completely in range of document length from 20 to 400.

In Table 5, we show top four topics with the highest $\mathrm{P}(k|c)$ values in reviews and their characteristic words. Note that we translate original Chinese words in English here. Top 10 characteristic words of reviews computed using Equation 5 are translated as follows: delicious, hot pot, tofu, beef, pepper, snack, taste, flavor, Sichuan-food and wine shop. We can see that using DTM to model domain knowledge for content-based tag recommendation is reasonable.

Comparing the performance of LDA on keyword extraction and tag recommendation, we find that LDA performs better than TextRank on news articles and worse on restaurant reviews. This may be because news articles are better-structured than restaurant reviews. On the other hand, restaurant reviews are generated by millions of users and there are thus more noises. Since LDA ranks terms by comparing their topic distributions with the topic distribution of the document, it is sensitive to the noises in the document. Therefore, LDA performs poorly on the documents with many noises. In contrast, by considering domain information, DTM performs the best on both news articles and restaurant reviews. This indicates that domain information can effectively prevent term rankings from the influence of document noises. In other words, the more

Table 5. Topics with the highest probability in Dianping reviews and most characteristic words

Rank	$\mathrm{P}(k\|c)$	Characteristic words of k
1	0.079	delicious, hot pot, snack, pepper
2	0.020	egg, potato, Chinese cabbage, tofu
3	0.019	chocolate, bread, fruit, cola
4	0.007	wine shop, customer, waiter, hotel

noises that a document has, the more necessarily we should take domain knowledge into consideration for term rankings.

5 Conclusions and Future Work

In this paper we present a novel method, referred to as Domain-Topic Model (DTM), to model domain knowledge of documents. Using DTM, we further perform domain-specific term rankings. By introducing the domain information as a crucial ranking factor in keyword extraction and content-based tag recommendation, DTM outperforms TextRank and LDA.

In this paper, DTM only considers the importance of a term in a specific domain, without taking the importance of the term in the corresponding document. We plan to combine both measures together for keyword extraction. Considering that in many cases, we do not have explicit domain information, instead, we may have a query. An interesting future work is to extend DTM to query-focused term rankings. In these years, several supervised topic models have been proposed [5,1,2,16,24]. We plan to try these supervised topic models as possible alternatives to DTM for domain-specific term rankings.

Acknowledgments

We want to thank Dr. Yi Wang for helpful discussions and comments. This work is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 60873174.

References

1. Andrzejewski, D., Zhu, X., Craven, M.: Incorporating domain knowledge into topic modeling via dirichlet forest priors. In: Proceedings of ICML, pp. 25–32 (2009)
2. Blei, D.M., McAuliffe, J.: Supervised topic models. In: Proceedings of NIPS, pp. 121–128 (2007)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30, 1–7 (1998)
5. Chemudugunta, C., Holloway, A., Smyth, P., Steyvers, M.: Modeling documents by combining semantic concepts with unsupervised statistical learning. In: Proceedings of ISWC, pp. 229–244 (2010)
6. Cohn, D., Chang, H.: Learning to probabilistically identify authoritative documents. In: Proceedings of ICML, pp. 167–174 (2000)
7. Dean, J., Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. In: Proceedings of OSDI, pp. 137–150 (2004)
8. Frank, E., Paynter, G., Witten, I., Gutwin, C., Nevill-Manning, C.: Domain-specific keyphrase extraction. In: Proceedings of IJCAI, vol. 16, pp. 668–673 (1999)
9. Griffiths, T.L., Steyvers, M.: Finding scientific topics. PNAS 101, 5228–5235 (2004)
10. Grineva, M., Grinev, M., Lizorkin, D.: Extracting key terms from noisy and multitheme documents. In: Proceedings of WWW, pp. 661–670 (2009)

11. Heinrich, G.: Parameter estimation for text analysis. Tech. rep., Vsonix GmbH and University of Leipzig (2008)
12. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of EMNLP, pp. 216–223 (2003)
13. Hulth, A., Karlgren, J., Jonsson, A., Bostrm, H., Asker, L.: Automatic Keyword Extraction Using Domain Knowledge. In: Gelbukh, A. (ed.) CICLing 2001. LNCS, vol. 2004, pp. 472–482. Springer, Heidelberg (2001)
14. Katakis, I., Tsoumakas, G., Vlahavas, I.: Multilabel text classification for automated tag suggestion. In: ECML/PKDD Discovery Challenge 2008 (2008)
15. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
16. Lacoste-Julien, S., Sha, F., Jordan, M.: Disclda: Discriminative learning for dimensionality reduction and classification. In: NIPS, pp. 897–904 (2008)
17. Litvak, M., Last, M.: Graph-based keyword extraction for single-document summarization. In: Proceedings of the Workshop Multi-source Multilingual Information Extraction and Summarization, pp. 17–24 (2008)
18. Liu, Z., Huang, W., Zheng, Y., Sun, M.: Extracting keyphrases via topic decomposition. In: Proceedings of EMNLP (2010)
19. Liu, Z., Li, P., Zheng, Y., Sun, M.: Clustering to find exemplar terms for keyphrase extraction. In: Proceedings of EMNLP, pp. 257–266 (2009)
20. Mihalcea, R., Tarau, P.: Textrank: Bringing order into texts. In: Proceedings of EMNLP, pp. 404–411 (2004)
21. Mishne, G.: Autotag: a collaborative approach to automated tag assignment for weblog posts. In: Proceedings of WWW, pp. 953–954 (2006)
22. Newman, D., Asuncion, A., Smyth, P., Welling, M.: Distributed inference for latent Dirichlet allocation. In: Proceedings of NIPS, pp. 1081–1088 (2007)
23. Over, P., Liggett, W., Gilbert, H., Sakharov, A., Thatcher, M.: Introduction to duc-2001: An intrinsic evaluation of generic news text summarization systems. In: Proceedings of DUC 2001 (2001)
24. Ramage, D., Hall, D., Nallapati, R., Manning, C.: Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In: Proceedings of EMNLP, pp. 248–256 (2009)
25. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. In: Proceedings of UAI, pp. 487–494 (2004)
26. Tatu, M., Srikanth, M., D'Silva, T.: RSDC 2008: Tag recommendations using bookmark content. ECML/PKDD Discovery Challenge (2008)
27. Turney, P.D.: Learning algorithms for keyphrase extraction. Information Retrieval 2, 303–336 (2000)
28. Wan, X., Xiao, J.: Collabrank: Towards a collaborative approach to single-document keyphrase extraction. In: Proceedings of COLING, pp. 969–976 (2008)
29. Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: Proceedings of AAAI, pp. 855–860 (2008)