Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Improving Recommendation of Tail Tags for Questions in Community Question Answering

Yu Wu,[†][*] **Wei Wu,**[‡] **Xiang Zhang,**[†] **Zhoujun Li,**[†] **Ming Zhou**[‡]
[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡] Microsoft Research, Beijing, China
{wuyu,lizj,zhangxiang}@buaa.edu.cn {wuwei,mingzhou}@microsoft.com

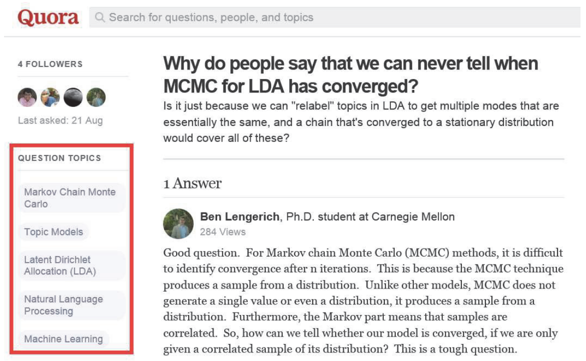
Abstract

We study tag recommendation for questions in community question answering (CQA). Tags represent the semantic summarization of questions are useful for navigation and expert finding in CQA and can facilitate content consumption such as searching and mining in these web sites. The task is challenging, as both questions and tags are short and a large fraction of tags are tail tags which occur very infrequently. To solve these problems, we propose matching questions and tags not only by themselves, but also by similar questions and similar tags. The idea is then formalized as a model in which we calculate question-tag similarity using a linear combination of similarity with similar questions and tags weighted by tag importance. Question similarity, tag similarity, and tag importance are learned in a supervised random walk framework by fusing multiple features. Our model thus can not only accurately identify question-tag similarity for head tags, but also improve the accuracy of recommendation of tail tags. Experimental results show that the proposed method significantly outperforms state-of-the-art methods on tag recommendation for questions. Particularly, it improves tail tag recommendation accuracy by a large margin.

Introduction

Community question answering (CQA) has become a popular platform for people sharing their knowledge and learning from each other. Large CQA portals like Yahoo! Answers have accumulated a large amount of user generated content which has played an important role in knowledge dissemination and information seeking. Recently, new types of CQA service such as Quora, Stack Overflow, and Zhihu have emerged. These web sites bring new features to conventional CQA portals such as Yahoo! Answers, and these features can guide the healthy growth of communities.

One creative feature in the new CQA service is the tagging system in which users can manually annotate multiple tags to their questions without any constraints. The tags represent semantic summarization of questions and reveal users' interests. Figure 1 gives an example from Quora in which *Markov Chain Monte Carlo, Topic Models, LDA, Natural Language Processing* and *Machine Learning* are tags



Why do people say that we can never tell when MCMC for LDA has converged?
Is it just because we can "relabel" topics in LDA to get multiple modes that are essentially the same, and a chain that's converged to a stationary distribution would cover all of these?

Ben Lengerich, Ph.D. student at Carnegie Mellon

Good question. For Markov chain Monte Carlo (MCMC) methods, it is difficult to identify convergence after n iterations. This is because the MCMC technique produces a sample from a distribution. Unlike other models, MCMC does not generate a single value or even a distribution, it produces a sample from a distribution. Furthermore, the Markov part means that samples are correlated. So, how can we tell whether our model is converged, if we are only given a correlated sample of its distribution? This is a tough question.

Figure 1: An example of descriptive tags in Quora

annotated for the question "Why do people say that we can never tell when MCMC for LDA has converged?". The tags have greatly improved user experience in the CQA ecosystem. Tags make questions easier to understand and thus can accelerate the process of the questions being solved. Under the guidance of tags, answerers can quickly locate questions within their expertise and provide answers with higher quality. This high quality content raises the overall quality of the CQA portals and can attract more users. Through tags, browsers can quickly find content they are interested in, easily keep track of the update of their favorite content, and get more engaged. Besides these advantages, an accurate and complete tag schema is also beneficial to consumption of the content in CQA. For example, tags can facilitate indexing, searching, and knowledge mining of the content in CQA.

Despite the importance of tags, existing CQA portals suffer from an incomplete tagging problem. According to (Nie et al. 2014), more than 50% questions in a subset of Zhihu questions have less than 3 tags. The problem is caused by the complexity of the tagging mechanism and incomprehensive question understanding, and severely hurts the performance of the tagging system in CQA. To solve this problem, we consider automatically recommending tags for questions. The recommendation provides tag candidates for users and can simplify the tagging procedure. Although tag recommendation has been studied for a long time (Herlocker et al. 2004; Rendle et al. 2009;

[*]The work was done when the first author was an intern in Microsoft Research Asia.

Table 1: Tag distribution in CQA

	$\geqslant 10^4$	$10^3 \sim 10^4$	$500 \sim 10^3$	$\leqslant 500$
Quora	0.01%	0.41%	0.56%	99.02%
Zhihu	0.04%	0.71%	0.82%	98.43%

Feng and Wang 2012), little attention has been paid to tag recommendation for questions in CQA. The key step for recommending tags for questions is measuring question-tag similarity. Compared with general tag recommendation tasks, the task for questions in CQA has some unique challenges. First, both questions and tags are very short. For example, on randomly sampled 1 million questions from Quora, we found that the average length of questions and tags are 9.48 and 2.27 respectively. In such short texts, useful information that can help bridge the semantic gap between questions and tags is very sparse. Second, we have observed that question tags follow a heavy tailed Zipf distribution in which a large fraction of tags occur very infrequently. Table 1 gives some statistics from 1 million questions randomly sampled. We can see that more than 98% tags appear no more than 500 times (i.e., 0.05% questions) on both data sets. We call these tags "tail tags" and other frequent tags "head tags". Although occurring infrequently, tail tags are specific and fine-grained descriptions of questions which reflect more accurate semantics and thus are more useful for expert finding, searching and knowledge mining, etc. For example, *Markov Chain Monte Carlo* is a tail tag in Figure 1. It is more informative than head tags like *Natural Language Processing*. From this we know that we need to rout the question to experts on sampling algorithms rather than those on syntactic parsing who also have expertise on *Natural Language Processing*. A good tagging system should be accurate on both head tags and tail tags. However, due to the long tail characteristic, it is very difficult to identify the similarity between questions and tail tags. We aim to solve information sparsity and long tail problems. Our idea is that we calculate similarity between a question and a tag not only by themselves, but also by similar questions and similar tags. The idea is formalized as a model in which we calculate question-tag similarity using a linear combination of similarity with similar questions and tags. The underlying assumption is that we can capture the similarity between a question and a tag by taking similar questions and similar tags as bridges. By this means, we can leverage rich tag-tag and question-question relations to compensate the sparse information between questions and tags, and tail tags can be boosted by head tags through tag-tag relations. The question-tag similarity is weighted by tag importance which is calculated by random walk on a tag graph with edges weighted by tag similarity. Question similarity, tag similarity and tag importance are thus learned in a supervised random walk framework by fusing multiple features.

We derive a gradient descent based algorithm to implement our model and test its performance on Quora and Zhihu data. Experimental results show that our method significantly outperforms state-of-the-art tag recommendation methods. Particularly, it can significantly improve the accuracy of recommendations of tail tags.

Our contributions in this paper are three-fold: 1)a proposal for improving accuracy on recommendation of tail tags for questions; 2)a proposal for leveraging similar questions and similar tags for tagging and learning question similarity, tag similarity and tag importance in a unified framework; 3) empirical verification of the effectiveness of the proposed method, especially on the recommendation of tail tags.

Related Work

Tag recommendation has been a hot research topic for a long time. Existing research on tag recommendation can be categorized into two groups: user based methods and content based methods. The former group utilizes relations among users, tags and items recommending personalized tags for items. Representative approaches in this group include collaboration filtering (Herlocker et al. 2004), tensor factorization (Rendle and Schmidt-Thieme 2010; Rendle et al. 2009), and graph based approaches (Feng and Wang 2012; Guan et al. 2009). The other group measures item-tag similarity using content. For example, Song et al. (2008) formalize tag recommendation as a classification problem. Liu et al. (2011) translate items to tags using a statistical machine translation technique. Besides these methods, topic models (Iwata, Yamada, and Ueda 2009; Krestel, Fankhauser, and Nejdl 2009; Godin et al. 2013) are also proposed which represent the relationship between items and tags in a generative process. Recently, tag recommendation in social media has drawn much attention, and there has emerged a lot of work on hashtag recommendation for tweets. For example, Ma et al. (2014) extend PLSA and modeled user, time, tweet content, and hashtag in a unified framework. Ding et al. (2013) learn a topical translation model for hashtag suggestion. Weston et al. (2014) embed both tweets and hashtags to calculate their similarity. In this paper, we study tag recommendation for questions in CQA. Before us, Stanley et al. (2013) propose a Bayesian probabilistic model to predict tags. Nie et al. (2014) leveraged similar questions to suggest tags for new questions. Differing from the method proposed by Nie et al., our method not only exploits question similarity, but also leverages tag similarity and tag importance. Question similarity, tag similarity, and tag importance are automatically learned in a supervised random walk framework (Backstrom and Leskovec 2011). Our method extends the general graph based method for social tagging (Feng and Wang 2012) by considering similar questions and similar tags in the objective. It significantly improves the accuracy of tail tags recommendation, which is a major challenge for tag recommendation in CQA but cannot be solved properly with existing methods.

Learning to Recommend Tags for Questions

We elaborate our method for tagging questions in CQA. The task is essentially measuring similarity between a question and a tag, and is challenging since both questions and tags are short and there is a large proportion of tail tags occurring very infrequently. Our solution is to match question-tag pair not only by themselves, but also by their similar questions and similar tags. Our method can thus leverage rich question-question and tag-tag relations to compensate sparse information between questions and tags, and boost the accuracy of recommendation of tail tags by their similar head tags. We weight the similarity function by tag importance which is the result of random walk on a tag graph. By doing so, we represent question similarity, tag similarity, and tag importance in a unified framework and learn them from human annotations by employing a supervised random walk model (Backstrom and Leskovec 2011).

Learning Method

Suppose that we have a question set $\mathbb{Q} = \{q_1, \ldots, q_N\}$. Each q_i in $\mathbb{Q}$ is associated with a relevant tag set $\mathbb{T}_i^+ = \{t_{i1}^+, \ldots, t_{iM_i^+}^+\}$ and an irrelevant tag set $\mathbb{T}_i^- = \{t_{i1}^-, \ldots, t_{iM_i^-}^-\}$. The relevant tags and irrelevant tags can be obtained from user annotations in CQA and negative sampling. Details will be discussed later. Our goal is to learn a question-tag similarity function $s(q,t)$ using $\mathbb{Q}$, $\{\mathbb{T}_i^+\}_{i=1}^N$, and $\{\mathbb{T}_i^-\}_{i=1}^N$. With $s(q,t)$, we can recommend relevant tags for a new question q from a candidate tag set $\mathbb{T}_q$. Due to the sparse information in questions and tags and the long tail characteristic of the tag distribution, it is difficult to directly calculate $s(q,t)$ from a pair (q,t). We calculate $s(q,t)$ not only by (q,t) themselves, but also by their similar questions and similar tags. The rationale is that similar questions and similar tags can provide extra information and improve learning accuracy. Moreover, $s(q,t)$ should be weighted by tag importance, since tag importance can filter out noise and make the similarity function more robust. To formulate our ideas, for each q (either in $\mathbb{Q}$ or a new question), we assume that there is a similar question set $\mathbb{Q}^s = \{q_1^s, \ldots, q_n^s\}$. $\forall 1 \leqslant j \leqslant n$, q_j^s corresponds to a tag set $\mathbb{T}_{q_j^s}$. We further assume that there are a heuristically designed question-tag similarity function $f(q,t)$, a question-question similarity function $g(q,q')$, and a tag-tag similarity function $h(t,t')$. Each tag t corresponds to a importance score $\pi(t)$. The similarity function $s(q,t)$ then is defined as

$$\pi(t)[f(q,t) + \sum_{q' \in \mathbb{Q}^s} g(q,q')f(q',t) + \sum_{t' \in \mathbb{T}_{\mathbb{Q}^s}} h(t,t')f(q,t')] \tag{1}$$

where $\mathbb{T}_{\mathbb{Q}^s}$ is defined as $\cup_{q_j^s \in \mathbb{Q}^s} \mathbb{T}_{q_j^s}$. Here, we define $f(q,t) = 1$, if t is a tag annotated to q by a user in CQA, otherwise $f(q,t)$ is a language model (Zhai and Lafferty 2004) which measures the generation probability of t from q.[1] $g(q,q')$ is defined through a dot product of a weight vector and a feature vector $\vec{\theta} = (\theta_1, \ldots, \theta_p)$ which can be formulated as

$$g(q,q') = \frac{1}{\sum_{1 \leqslant i \leqslant p} e^{w_i}} \sum_{1 \leqslant i \leqslant p} e^{w_i}\theta_i, \tag{2}$$

[1]Our goal is to verify the effectiveness of similar questions and similar tags, therefore we only employ a heuristically designed question-tag similarity function. One can easily replace it with a more sophisticated model.

where $\vec{w} = (w_1, \ldots, w_p)$ are parameters. We transform w_i to e^{w_i} to make the weight positive and further scale the weight to $(0,1)$. Similarly, we define $h(t,t')$ as

$$h(t_i,t_j) = \frac{1}{\sum_{1 \leqslant i \leqslant l} e^{v_i}} \sum_{1 \leqslant i \leqslant l} e^{v_i}\xi_i, \tag{3}$$

where $\vec{v} = (v_1, \ldots, v_l)$ are parameters and $\vec{\xi} = (\xi_1, \ldots, \xi_l)$ are features. We assume that the importance of tag t is determined by the importance of its similar tags and its own prior probability. To this end, we build an undirected graph $G_t = (V_t, E_t)$ for candidate tags with nodes V_t tags and edges E_t normalized tag similarity. Without loss of generality, we assume that the construction of G_t is dependent on t and thus G_t could be a local graph of t. Then tag importance is defined as a result of random walk with restart on G_t:

$$\vec{\pi} = \alpha A_t^\top \vec{\pi} + (1-\alpha)\vec{z_t}, \tag{4}$$

where $\vec{\pi}$ is an importance vector with each element importance of a node in G_t. Suppose that the number of nodes of G_t is n_t, then $A_t = (a_{ij})_{n_t \times n_t}$ is a transition matrix. $\forall i,j$, a_{ij} is defined as $\frac{h(t_i,t_j)}{\sum_{t \in V_t} h(t_i,t)}$. $\vec{z_t}$ represents restart probabilities of nodes and $\alpha \in (0,1)$ controls the probability of random walk from neighboring nodes and the probability of random jump from any other nodes. The advantage of calculating tag importance in this way is that we can leverage tag similarity to estimate tag importance and learn the similarity and importance in a unified framework, as will be seen later.

Equation (1) can be interpreted as first finding similar questions and similar tags and then linearly combining the similarity of a question-tag pair and their similarity with the similar questions and tags. The more similar a question or a tag is, the more contributions it will make to the final similarity $s(q,t)$. The model enjoys many advantages: first, it can leverage rich question-question and tag-tag relations to compensate sparse information between questions and tags; second, it can boost the accuracy of recommendation of tail tags by leveraging similar tags and similar questions containing these tags; third, as will be seen later, all the parameters in the model can be learned from user annotations by employing a supervised random walk approach which has been verified effective in other tasks like link prediction.

To better illustrate our model, we give an example. For a question "how many people agree that Movie 'Life of Pi' deserves more Oscars?", "Ang Lee", the director of the movie "Life of Pi", is a relevant tag. Directly measuring the similarity between the question and the tag is difficult, since there is a big semantic gap between them and "Ang Lee" is a tail tag with little information indicating its similarity with the question. On the other hand, the tag "Life of Pi" is similar to the question (they have term overlap) and high significance score in $\pi(t)$ (high frequency in similar questions). Moreover, we know that "Ang Lee" and "Life of Pi" are similar tags (they co-occur frequently) , then by taking "Life of Pi" as a bridge, our model can still identify the similarity between "Ang Lee" and the question.

We learn parameters $\vec{w}$ and $\vec{v}$ from $\mathbb{Q}$, $\{\mathbb{T}_i^+\}_{i=1}^N$, and $\{\mathbb{T}_i^-\}_{i=1}^N$ by maximizing the margin between similarity

Algorithm 1 Optimization algorithm for problem (5)

```
input:c $\mathbb{Q}$, $\{\mathbb{T}_i^+\}_{i=1}^N$, $\{\mathbb{T}_i^-\}_{i=1}^N$, $\{\mathbb{Q}^s\}$, $\{\mathbb{T}_{\mathbb{Q}^s}\}$, $\alpha$, $\lambda$, $\vec{z}_t$,
$T$ = maximal iteration count, and $\delta > 0$.
Initialize: $\vec{w} = \vec{w}_0$, $\vec{v} = \vec{v}_0$, and $i = 1$
while $J$ has not converged and $i < T$ do
    Fix: $\vec{w}_{i-1}$ and $\vec{v}_{i-1}$
    Calculate $\vec{\pi}$ by $\vec{\pi} = (1-\alpha)\left(I - \alpha A_t^\top\right)^{-1} \vec{z}_t$.
    Calculate $\frac{\partial \vec{\pi}}{\partial v_k}$ by Equation (13), $k = 1:l$.
    Calculate $\frac{\partial J}{\partial w_k}$ by Equation (6), (7), and (8), $k = 1:p$.
    Calculate $\frac{\partial J}{\partial v_k}$ by Equation (9), (10), and (11), $k = 1:l$.
    Update:
    $w_{ik} = w_{(i-1)k} - \delta \frac{\partial J}{\partial w_k}, k = 1:p$,
    $v_{ik} = v_{(i-1)k} - \delta \frac{\partial J}{\partial v_k}, k = 1:l$.
    $i = i+1$
end while
Output: $\vec{w}_i$, $\vec{v}_i$.
```

functions corresponding to relevant tags and irrelevant tags. Formally, we consider the following problem:

$$\begin{aligned} \arg\min_{\vec{w},\vec{v}} \quad & \sum_{i=1}^{N}\sum_{j=1}^{M_i^+}\sum_{r=1}^{M_i^-}[1-(s(q_i,t_{ij}^+)-s(q_i,t_{ir}^-))]_+ \\ & +\lambda(||\vec{w}||^2+||\vec{v}||^2) \\ \text{s.t.} \quad & \vec{\pi} = \alpha A_t^\top \vec{\pi} + (1-\alpha)\vec{z}_t, \quad \forall t \in \cup_{i=1}^N \mathbb{T}_i^+ \cup \mathbb{T}_i^-, \end{aligned} \tag{5}$$

where $[x]_+$ is defined as $\max(x, 0)$ and $||\cdot||$ is ℓ_2-norm. λ is a trade-off between the loss and regularization of parameters. Our method optimizes an objective function with random walk on tag graphs as a constraint, which turns out to be an application of the supervised random walk approach (Backstrom and Leskovec 2011) to the problem of question tagging. We extends the approach by considering similar questions and similar tags in the objective function. The extension can improve the accuracy of recommendation of tail tags which is a major challenge for question tagging in CQA, as will be seen in our experiments.

Nie et al. (2014) also propose leveraging question similarity and tag importance to recommend tags for questions. Our method is different from theirs in that it not only exploits question similarity but also considers tag similarity. Question similarity, tag similarity, and tag importance are learned from data rather than heuristically designed.

Algorithm

We derive a gradient descent based algorithm to solve the optimization problem (5). Specifically, we denote the objective function in problem (5) as $J(\vec{w}, \vec{v})$ and $s(q_i, t_{ir}^-) - s(q_i, t_{ij}^+)$ as $O(q_i, t_{ij}^+, t_{ir}^-)$. Then $\forall 1 \leqslant k \leqslant p$, $\frac{\partial J}{\partial w_k}$ is given by

$$\sum_{i=1}^{N}\sum_{j=1}^{M_i^+}\sum_{r=1}^{M_i^-}\frac{\partial O(q_i,t_{ij}^+,t_{ir}^-)}{\partial w_k}\mathcal{I}\left(1+O(q_i,t_{ij}^+,t_{ir}^-)\right)+2\lambda w_k, \tag{6}$$

where $\frac{\partial O(q_i,t_{ij}^+,t_{ir}^-)}{\partial w_k} = \frac{\partial s(q_i,t_{ir}^-)}{\partial w_k} - \frac{\partial s(q_i,t_{ij}^+)}{\partial w_k}$, and $\mathcal{I}(x)$ is 1 if $x > 0$, otherwise it is 0. According to Equation (1), $\forall(q,t)$, we have

$$\frac{\partial s(q,t)}{\partial w_k} = \sum_{q' \in \mathbb{Q}^s} \frac{\partial g(q,q')}{\partial w_k}\pi(t) f(q',t), \tag{7}$$

and

$$\frac{\partial g(q,q')}{\partial w_k} = \frac{e^{w_k}\theta_k \sum_{1\leqslant j\leqslant p} e^{w_j} - e^{w_k}\sum_{1\leqslant j\leqslant p} e^{w_j}\theta_j}{(\sum_{1\leqslant j\leqslant p} e^{w_j})^2}. \tag{8}$$

Following the same technique, $\forall 1 \leqslant k \leqslant l$, we can calculate $\frac{\partial J}{\partial v_k}$ as

$$\sum_{i=1}^{N}\sum_{j=1}^{M_i^+}\sum_{r=1}^{M_i^-}\frac{\partial O(q_i,t_{ij}^+,t_{ir}^-)}{\partial v_k}\mathcal{I}\left(1+O(q_i,t_{ij}^+,t_{ir}^-)\right)+2\lambda v_k, \tag{9}$$

and $\frac{\partial O(q_i,t_{ij}^+,t_{ir}^-)}{\partial v_k} = \frac{\partial s(q_i,t_{ir}^-)}{\partial v_k} - \frac{\partial s(q_i,t_{ij}^+)}{\partial v_k}$. $\forall(q,t)$,

$$\frac{\partial s(q,t)}{\partial v_k} = \frac{s(q,t)}{\pi(t)}\frac{\partial \pi(t)}{\partial v_k} + \sum_{t' \in \mathbb{T}_{\mathbb{Q}^s}} \frac{\partial h(t,t')}{\partial v_k}\pi(t) f(q,t), \tag{10}$$

and

$$\frac{\partial h(t,t')}{\partial v_k} = \frac{e^{v_k}\xi_k \sum_{1\leqslant j\leqslant l} e^{v_j} - e^{v_k}\sum_{1\leqslant j\leqslant l} e^{v_j}\xi_j}{(\sum_{1\leqslant j\leqslant l} e^{v_j})^2}. \tag{11}$$

From Equation (10), we know that we need $\frac{\partial \pi(t)}{\partial v_k}$ to calculate $\frac{\partial J}{\partial v_k}$. Since tag importance follows Equation (4), by differentiating Equation (4) on both sides, we have

$$\frac{\partial \vec{\pi}}{\partial v_k} = \alpha A_t^\top \frac{\partial \vec{\pi}}{\partial v_k} + \alpha \frac{\partial A_t^\top}{\partial v_k}\vec{\pi}. \tag{12}$$

Then, we have

$$\frac{\partial \vec{\pi}}{\partial v_k} = \alpha(I - \alpha A_t^\top)^{-1}\frac{\partial A_t^\top}{\partial v_k}\vec{\pi}. \tag{13}$$

$\frac{\partial A_t^\top}{\partial v_k} = \left(\frac{\partial a_{ij}}{\partial v_k}\right)_{n_t \times n_t}^\top$, and

$$\frac{\partial a_{ij}}{\partial v_k} = \frac{\frac{\partial h(t_i,t_j)}{\partial v_k}\sum_{t\in V_t} h(t_i,t) - h(t_i,t_j)\sum_{t\in V_t}\frac{\partial h(t_i,t)}{\partial v_k}}{\left(\sum_{t\in V_t} h(t_i,t)\right)^2}.$$

The equation above can be calculated by Equation (11). For a fixed point $(\vec{w}, \vec{v})$, we obtain $\frac{\partial \vec{\pi}}{\partial v_k}$ by Equation (13). With $\frac{\partial \vec{\pi}}{\partial v_k}$, we obtain both $\frac{\partial J}{\partial w_k}$ and $\frac{\partial J}{\partial v_k}$ by Equation (6), (7), (8), (9), (10), and (11). Then we can rely on $\frac{\partial J}{\partial w_k}$ and $\frac{\partial J}{\partial v_k}$ to update $(\vec{w}, \vec{v})$. The procedure is repeated several times until we reach the stopping criteria. The details are described in Algorithm 1. The complexity of the algorithm is $O\left(plN\bar{M}^2 + lN\bar{M}\bar{n}^3\right)$, where N is the number of questions in training, p and l are the numbers of features of questions and tags respectively, $\bar{M}$ is the upper bound of the numbers of tags in relevant sets and irrelevant sets, and $\bar{n}$ is the upper bound of node numbers of tag graphs. In practice, p, l, $\bar{M}$, and $\bar{n}$ are small. In our experiments, p and l are 3, $\bar{M}$ is no more than 40, and $\bar{n}$ is no more than 220. Therefore, the optimization problem can be efficiently solved. The code of Algorithm 1 is shared at https://github.com/MarkWuNLP/QuestionTagging.

Implementation Details

To implement our method, we need to clarify (1) how to construct the similar question set $\mathbb{Q}^s$ and the associated tag set $\mathbb{T}_{\mathbb{Q}^s}$ for a question q; (2) how to construct the relevant tag set $\mathbb{T}_i^+$ and the irrelevant tag set $\mathbb{T}_i^-$ for a question q_i in the training set $\mathbb{Q}$; (3) how to select the tag candidates $\mathbb{T}_q$ for a new question q; (4) how to build the tag graph G_t and how to define the restart probability $\vec{z}_t$; and (5) what features are used in $g(q, q')$ and $h(t, t')$.

We index a large scale of questions crawled from a CQA web site such as Quora using an open source Lucene.Net. Each question is associated with tags annotated by the asker of the question. Given a question q, we rely on the inline ranking algorithm in Lucene.Net to retrieve the top 50 questions as similar questions $\mathbb{Q}^s$, and then collect all tags that are annotated to questions in $\mathbb{Q}^s$ as $\mathbb{T}_{\mathbb{Q}^s}$. The algorithm finds similar questions for q based on common terms they share and thus can balance efficacy and efficiency. All tags annotated by the asker of a question form the relevant tag set of the question. We adopt a heuristic yet effective method to construct the irrelevant tag set. Specifically, we collect all tags in similar questions, calculate their frequency in the similar question set, and select the top n frequent tags that are not contained by the relevant tag set as the irrelevant tags where n is equal to the number of relevant tags. For any question in the test set, we simply take all tags in similar questions as candidates for recommendation. We build tag graphs with all tags in similar questions. In training, relevant tags of the training questions are also considered as nodes in the tag graphs. We calculate the number of times a tag appears in the similar questions and normalize the number with the total number of similar questions as the restart probability vector $\vec{z}_t$.

To calculate question-question similarity, we consider three features: (1) cosine of tf-idf weighted vectors of two questions; (2) cosine of topic distributions of two questions. The topic distribution is estimated by running a Twitter LDA (Zhao et al. 2011) on the crawled questions; (3) translation probability $p(q'|q)$. The probability is calculated by a translation based language model (Xue, Jeon, and Croft 2008) estimated using all the crawled questions with answers. We normalize the probability by defining $p'(q'|q) = p(q'|q)/p(q|q)$ and further symmetrize the score by $0.5\,(p'(q'|q) + p'(q|q'))$. For tag-tag similarity, we also consider three features: (1) probability of local co-occurrence which is defined as the quotient of co-occurrence count of two tags in the similar questions and the total number of similar questions (i.e., 50); (2) cosine of tag vectors. The tag vector is obtained by training a word embedding model (Mikolov et al. 2013) on all the crawled questions with answers and averaging the vectors of words in the tag; (3) global closeness which is defined as the quotient of co-occurrence count of two tags in all the crawled questions and the minimum occurrence count in the two tags.

Table 2: Statistics about the crawled data

	#Question	#Answer	#Tag	#Dist Tag
Quora	648,996	1,739,222	2,107,204	69,522
Zhihu	1,069,403	1,634,489	3,149,900	59,379

Experiment

Experiment Setup

We crawled a large number of questions associated with tags and answers from English CQA portal Quora (https://www.quora.com/) and Chinese CQA portal Zhihu (http://www.zhihu.com/)[2]. Table 2 reports some statistics about the data we crawl. On average, each question in Quora has 3.25 tags and 1.55 tail tags which appear no more than 500 times in all the questions we crawled. The average numbers of tags and tail tags per question are 2.95 and 1.27 respectively in Zhihu data. All English questions and answers were stemmed and all Chinese questions and answers were segmented. Stop words in both English data and Chinese data were removed. For both of the data sets, we sampled 1000 questions associated with tags to form a validation set and another 3000 questions with tags as a test set. In validation and test, we regarded tags annotated by users as ground truth. Although there is noise in the annotated tags, the way we constructed the ground truth can help us avoid expensive and exhausting human judgments and thus allows us to do large scale evaluations. We followed the implementation details described above to implement our method. We randomly sampled 5000 questions in the training sets of Quora and Zhihu to learn our model. On average, there are 96.9 and 92.35 candidate tags for each question in the validation set for Quora and the validation set for Zhihu respectively, and there are 97.26 and 91.85 candidate tags per question in the test set of Quora and the test set of Zhihu respectively. 76.8% candidate tags in the test set for Quora and 66.3% candidate tags in the test set for Zhihu are tail tags.

To evaluate the performance of different models, we selected $P@K$ ($K = 1, 3, 5$) as a metric. $P@K$ is defined as the ratio of correct tags (i.e., user annotated tags) in the top K recommended tags. In addition to calculating accuracy on all recommended tags, we also compared different models in terms of their performance on the recommendation of tail tags. We calculated $P_t@K$ ($K = 1, 3, 5$) on tail tags as an evaluation metric, where $P_t@K$ is defined as the ratio of correct tail tags in the top K recommended tags. A large $P_t@K$ means a method can accurately identify the similarity of questions and tail tags and rank them in high positions.

We chose Naive Bayes (NB) (Mazzia and Juett 2009) as a baseline method which estimates the posterior probability of a tag given a question. Besides this model, we implemented the model proposed by Liu et al. (2011), and denoted it as translation model (TM). We implemented the topical translation model (TTM) proposed by Ding et al. (2013) which represents the state-of-the-art topic model based method for

[2]Questions without any tags have been filtered out beforehand.

Table 3: Evaluation on Quora data

	$P@1$	$P@3$	$P@5$	$P_t@1$	$P_t@3$	$P_t@5$
NB	0.365	0.235	0.186	0.037	0.018	0.011
TM	0.376	0.279	0.228	0.109	0.086	0.075
TTM	0.467	0.304	0.231	0.085	0.067	0.046
HGM	0.496	0.357	0.273	0.142	**0.123**	**0.097**
Ours	**0.553**	**0.371**	**0.275**	**0.165**	**0.122**	**0.092**

Table 4: Evaluation on Zhihu data

	$P@1$	$P@3$	$P@5$	$P_t@1$	$P_t@3$	$P_t@5$
NB	0.351	0.172	0.123	0.027	0.016	0.011
TM	0.315	0.233	0.194	0.095	0.073	0.064
TTM	0.419	0.257	0.197	0.082	0.061	0.054
HGM	0.427	0.322	0.252	0.156	0.114	**0.098**
Ours	**0.577**	**0.378**	**0.277**	**0.176**	**0.125**	**0.099**

tag suggestion. We also compared our model with the one proposed by Nie et al. (2014). It builds a hypergraph of questions for question tagging in CQA. We denote the model as hypergraph based model (HGM). Since we only consider the relationship between questions and tags, tensor decomposition based methods are not applicable. All baseline methods were trained using the entire body of training data.

Evaluation Results

We followed the settings in existing literatures to implement the baseline methods. For our method, we varied α in Equation (4) from 0 to 1 with a step 0.2, and chose the number of similar questions in $\mathbb{Q}^s$ from $\{10, 30, 50, 80, 100\}$. We relied on $P@K$ on the validation data to select the best parameters. On both of datasets, the best choice of α is 0.5, and the optimal number of similar questions is 50. We set the maximum iteration count as 200 and fixed λ in Equation (5) as 0.001.

Tables 3 and 4 report the evaluation results. In both tables, the first three columns show overall precision and the last three columns present precision on tail tags. The statistically significant (t-test with p value < 0.01) results are written in bold. Note that the numbers of $P_t@K$ are small, because we only focus on correct tail tags when we calculate $P_t@K$, and head tags that are correct recommended were not taken into account. In terms of the overall precision (i.e., $P@K$), our method performs much better than HGM, since HGM heuristically weights each tag by $tf \times idf$ and ignores tag-tag relations, while our method sufficiently leverages tag-tag relations and learns question similarity, tag similarity, and tag importance from user annotations. HGM is particularly bad on $P@1$ as noise is boosted by the heuristically designed weight. Both of the two models significantly outperform other models, indicating that similar questions can help solve the information sparsity problem and effectively bridge the semantic gap between tags and questions. TTM enjoys the advantages of both topic model and translation model, therefore it is better than the simple TM. NB only capture the similarity between questions and very head tags, therefore it has a relatively good $P@1$ but very bad $P@3$ and $P@5$. In terms of precision on tail tags (i.e., $P_t@K$), results on both tables show that our method can significantly improve the accuracy of recommendation of tail tags. The big gap on $P_t@1$ indicates that our method can rank correct tail tags in higher positions and thus can better represent the semantics of questions. We find that tag similarity make great contributions to the improvement of tail tag recommendation, particularly in the comparison of our method and HGM. HGM only relies on similar questions and the heuristically designed tag importance is vulnerable to noise in tags. On the other hand, our method boosts tail tags by both similar head tags and similar questions and learns question similarity, tag similarity and tag importance from data. Therefore, it is more robust to noise. NB, TM and TTM perform poorly on tail tags, since they learn question-tag similarity only using question-tag pairs and thus are overwhelmed by head tags which have enough frequency and rich information to indicate their similarity to questions.

Discussions

Our method leverages both the similar question set $\mathbb{Q}^s$ with $\mathbb{T}_{\mathbb{Q}^s}$ and the learned question similarity, tag similarity and tag importance to conduct tag recommendation. An interesting question is how much the learning contributes to the accuracy of recommendation. To figure this out, we compared our model with two heuristic models. First, we simply removed $\sum_{q' \in \mathbb{Q}^s} g(q, q') f(q', t)$ and $\sum_{t' \in \mathbb{T}^+_{\mathbb{Q}^s}} h(t, t') f(q, t')$ from Equation (1), and ranked tags by the product of their frequency in similar questions and an inverse frequency score which is defined as $1/\log(o(t) + 1)$ where $o(t)$ is the frequency of tag t in the entire crawled data. This is exactly the tag importance used by HGM. We name it $tf \times idf$. Second, we removed the feature fusing process from Equation (1) and only relied on one feature to define $g(q, q')$ and $h(t, t')$. After trying different feature combinations, we found that cosine of tf-idf weighted vectors of two questions and probability of local co-occurrence are the most useful features for $g(q, q')$ and $h(t, t')$ respectively. We refer the model as "one feature". For better illustration of two categories feature contribution, we conducted two experiments, one is only uses question features, the other only employs tag features for tag suggestion, which named "question sim" and "tag sim" respectively.

From the results in Table 5, $tf \times idf$ boosts tail tags by the inverse frequency score but severely hurt the overall precision. The results indicate that we cannot simply rely on $\mathbb{Q}^s$ with $\mathbb{T}_{\mathbb{Q}^s}$ and heuristically designed tag importance to recommend tags. One feature performs much better than $tf \times idf$ on overall precision but the precision on tail tags dropped. Our model improves both overall precision and precision on tail tags. The results demonstrate that question similarity and tag similarity are useful for tag recommendation and our learning approach can improve the recommendation accuracy on both head tags and tail tags.

Conclusion

We explore the problem of question tagging in CQA. Our method can leverage similar questions and similar tags to

Table 5: Learning v.s. no learning on Quora data and Zhihu data

	Quora data						**Zhihu data**					
	$P@1$	$P@3$	$P@5$	$P_t@1$	$P_t@3$	$P_t@5$	$P@1$	$P@3$	$P@5$	$P_t@1$	$P_t@3$	$P_t@5$
$tf \times idf$	0.523	0.350	0.251	0.214	0.139	0.102	0.511	0.308	0.218	0.246	0.130	0.089
One Feature	0.550	0.365	0.276	0.161	0.117	0.089	0.574	0.374	0.278	0.171	0.113	0.097
Tag Sim	0.535	0.355	0.268	0.141	0.109	0.086	0.547	0.372	0.269	0.162	0.119	0.088
Question Sim	0.550	0.375	0.276	0.161	0.117	0.089	0.570	0.374	0.278	0.169	0.119	0.092
Our Model	0.553	0.371	0.275	0.165	0.122	0.092	0.577	0.378	0.277	0.176	0.125	0.099

boost the recommendation of tail tags and learn question similarity, tag similarity, and tag importance in a unified framework. Experimental results show that the method significantly outperforms state-of-the-art methods of tag recommendation for questions. Particularly, it can significantly improve accuracy of recommendation of tail tags.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (Grant Nos. 61170189,61370126, 61202239), National High Technology Research and Development Program of China under grant(No.2015AA016004), Major Projects of the National Social Science Fund of China under grant (No.14 & ZH0036), Science and Technology Innovation Ability Promotion Project of Beijing (PXM2015-014203-000059), the Fund of the State Key Laboratory of Software Development Environment (No.SKLSDE-2015ZX-16), Microsoft Research Asia Fund (No. FY14-RES-OPP-105).

References

Backstrom, L., and Leskovec, J. 2011. Supervised random walks: predicting and recommending links in social networks. In *WSDM'10*, 635–644.

Ding, Z.; Qiu, X.; Zhang, Q.; and Huang, X. 2013. Learning topical translation model for microblog hashtag suggestion. In *IJCAI'13*, 2078–2084.

Feng, W., and Wang, J. 2012. Incorporating heterogeneous information for personalized tag recommendation in social tagging systems. In *KDD'12*, 1276–1284.

Godin, F.; Slavkovikj, V.; De Neve, W.; Schrauwen, B.; and Van de Walle, R. 2013. Using topic models for twitter hashtag recommendation. In *Proceedings of the 22nd international conference on World Wide Web companion*, 593–596. International World Wide Web Conferences Steering Committee.

Guan, Z.; Bu, J.; Mei, Q.; Chen, C.; and Wang, C. 2009. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In *SIGIR'09*, 540–547.

Herlocker, J. L.; Konstan, J. A.; Terveen, L. G.; and Riedl, J. T. 2004. Evaluating collaborative filtering recommender systems. *TOIS* 22(1):5–53.

Iwata, T.; Yamada, T.; and Ueda, N. 2009. Modeling social annotation data with content relevance using a topic model. In *NIPS'09*, 835–843.

Krestel, R.; Fankhauser, P.; and Nejdl, W. 2009. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*, 61–68.

Liu, Z.; Chen, X.; and Sun, M. 2011. A simple word trigger method for social tag suggestion. In *EMNLP'11*, 1577–1588.

Ma, Z.; Sun, A.; Yuan, Q.; and Cong, G. 2014. Tagging your tweets: A probabilistic modeling of hashtag annotation in twitter. In *CIKM'14*, 999–1008.

Mazzia, A., and Juett, J. 2009. Suggesting hashtags on twitter. *EECS 545m, Machine Learning, Computer Science and Engineering, University of Michigan*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS'13*, 3111–3119.

Nie, L.; Zhao, Y.-L.; Wang, X.; Shen, J.; and Chua, T.-S. 2014. Learning to recommend descriptive tags for questions in social forums. *TOIS* 32(1):5.

Rendle, S., and Schmidt-Thieme, L. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM'10*, 81–90.

Rendle, S.; Balby Marinho, L.; Nanopoulos, A.; and Schmidt-Thieme, L. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD'09*, 727–736.

Song, Y.; Zhang, L.; and Giles, C. L. 2008. A sparse gaussian processes classification framework for fast tag suggestions. In *CIKM'08*, 93–102.

Stanley, C., and Byrne, M. D. 2013. Predicting tags for stackoverflow posts. In *Proceedings of ICCM*, volume 2013.

Weston, J.; Chopra, S.; and Adams, K. 2014. tagspace: Semantic embeddings from hashtags. In *EMNLP'14*, 1822–1827.

Xue, X.; Jeon, J.; and Croft, W. B. 2008. Retrieval models for question and answer archives. In *SIGIR'08*, 475–482.

Zhai, C., and Lafferty, J. 2004. A study of smoothing methods for language models applied to information retrieval. *TOIS* 22(2):179–214.

Zhao, W. X.; Jiang, J.; Weng, J.; He, J.; Lim, E.-P.; Yan, H.; and Li, X. 2011. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*. Springer. 338–349.