Ranking Social Bookmarks Using Topic Models

Morgan Harvey and Ian Ruthven
University of Strathclyde
Computer and Information Sciences Department
Glasgow, United Kingdom
{morgan,ir}@cis.strath.ac.uk

Mark J. Carman
University of Lugano
Faculty of Informatics
Lugano, Switzerland
mark.carman@lu.unisi.ch

ABSTRACT

Ranking of resources in social tagging systems is a difficult problem due to the inherent sparsity of the data and the vocabulary problems introduced by having a completely unrestricted lexicon. In this paper we propose to use hidden topic models as a principled way of reducing the dimensionality of this data to provide more accurate resource rankings with higher recall. We first describe Latent Dirichlet Allocation (LDA) and then show how it can be used to rank resources in a social bookmarking system. We test the LDA tagging model and compare it with 3 non-topic model baselines on a large data sample obtained from the Delicious social bookmarking site. Our evaluations show that our LDA-based method significantly outperforms all of the baselines.

Categories and Subject Descriptors

H.3.3 [**Information Storage & Retrieval**]: Information Search & Retrieval

General Terms

Algorithms, Experimentation

Keywords

Topic Models, Collaborative Tagging, Social Bookmarking

1. INTRODUCTION

Social tagging systems provide a new way for Internet users to organise and share their own digital content and content from other users. Users are able to annotate each resource with any number of free-form tags of their own choosing without having to adhere to an *a-priori* set of keywords.Their simple nature and unrestricted vocabulary is a boon for annotators, however searching for resources of interest in social tagging systems tends to be a frustrating process. Analyses of tagging systems [2] have shown that term use tends be very inconsistent between different users resulting in a large number of polysemous and synonymous tags. This has a highly detrimental effect on search performance unless the system deals with this inherent variation in some way. Several studies have shown that obtaining high consistency among different taggers is very difficult to achieve and can be affected by many factors including vocabulary use, personal understanding of the resource and language [8, 5].

In current social tagging systems, search algorithms tend to be rather simplistic in nature, often relying on simple term matching algorithms in order to rank resources given a query and seek to exploit the aggregated annotations across all users, the so called "wisdom of the crowds". This simple approach to the problem fails to deal with the vocabulary problems noted above and can result in quite poor rankings, particularly when users make use of very specific or unusual tags.

In this paper we investigate utilising techniques based on topic modelling to rank resources from a social bookmarking system given simple search queries. We first investigate related work in both social tagging and general information retrieval fields. Next we introduce topic modelling by describing the well-known Latent Dirichlet Allocation model [1, 3] and then go on to explain how it can be used to model tagging data. We describe algorithms for ranking resources using this model and evaluate its performance based on a large sample of data from the social bookmarking site delicious and compare this with a number of competitive non-topic model baselines. Finally we conclude with a discussion of the results of the research and some suggestions for future work.

We now describe topic models, explain why we have chosen them as a manner of factorising social tagging data and show how they can be used to rank resources.

2. TOPIC MODELS

Topic models attempt to probabilistically uncover the underlying semantic structure of a collection of resources based on analysis of only the vocabulary words present in each resource, this latent structure is modelled over a number of topics which are assumed to be present in the collection.

In order to use these techniques we need to construct representations of documents made up of terms from a shared vocabulary. In this case our "documents" are the URLs (or in social tagging parlance *resources*) users have chosen to bookmark. We must therefore construct the documents representations by conflating the tags used by all users to annotate each bookmarked URL. Each URL is now represented

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

by the complete set of tags used to describe it by users of the social tagging system. Ideally this approach should allow us to: (1) generalise vocabulary terms to deal with synonymy and polysemy and (2) generalise the resource representations based on the similarity to other resources in the data set. These models operate using Bayesian inference which is useful when reasoning from noisy data, this is particularly appealing in this context as we expect the distributions of tags over resources to be both sparse and noisy.

LDA represents documents as random mixtures over latent topics which are random mixtures over observed words in the vocabulary. The model possesses a number of advantageous attributes; it is fully generative meaning that it is easy to make inferences on new documents or terms and overcomes the overfitting problem present in models such as Probabilistic Latent Semantic Indexing (pLSI) [4]. Also since in LDA each document is a mixture over latent topics it is far more flexible than models that assume each document is only drawn from a single topic.

The parameters estimated in LDA are two matrices Φ and Θ containing estimates for the probability of a word given a topic $P(w|z)$ and a topic given a document $P(z|d)$. Thus each column of the respective matrices contains (estimates for) a probability distribution over words for a particular topic and over topics for a particular document, denoted ϕ_z and θ_d respectively. In order to prevent overfitting the data, LDA places a symmetric Dirichlet prior on both these distributions, resulting in the following expectations for the parameter values under the respective posterior distributions $P(\phi_z|\mathbf{w}, \mathbf{z})$ and $P(\theta_d|\mathbf{z}, \mathbf{d})$, where $\mathbf{w}$ is the vector of words occurrences w_i in the corpus, $\mathbf{z}$ is an assignment of topics to each word position z_i and $\mathbf{d}$ is the vector of documents d_i associated with each word position:

$$\hat{\phi}_{w|z} = \frac{N_{w,z} + \beta\frac{1}{W}}{N_z + \beta}$$

$$\hat{\theta}_{z|d} = \frac{N_{z,d} + \alpha\frac{1}{Z}}{N_d + \alpha}$$

Here $N_{w,z}$, $N_{z,d}$ and N_z are counts denoting the number of times the topic z appears (in $\mathbf{z}$) together with the word w, with the document d and in total. W is the vocabulary size and Z is the number of topics. Symmetric Dirichlet priors with hyperparameters α and β are placed over the distributions θ_d and ϕ_w and essentially act as a pseudo count indicating a relation to smoothing in language models. This allows the model to fall back on the priors in the event of sparse data.

Exact inference of the LDA model is intractable, however a number of methods of approximating the posterior distribution have been proposed including mean field variational inference [1] and Gibbs sampling [3]. Gibbs sampling is a Markov chain Monte Carlo method where a Markov chain is constructed that slowly converges to the target distribution of interest over a number of iterations. Each state of the Markov chain is (in this case) an assignment of a discrete topic (from 1 to Z) to each z_i, i.e. to each observed word in the corpus. In Gibbs sampling the next state in the chain is reached by sampling all variables from their distribution when conditioned on the current values of all the other variables. For LDA the Gibbs sampling assumes the following:

$$P(z_i|w_i, d_i) = \frac{P(z_i, w_i|d_i)}{P(w_i|d_i)} \propto P(w_i|z_i)P(z_i|d_i)$$

After sufficient iterations of the sampler, the Markov chain converges and the parameters of the LDA model can then be estimated from $\mathbf{z}$. We can assume that the chain has converged when we observe minimal change in the model likelihood over successive samples, in the case of LDA the likelihood is:

$$P(\mathbf{w}, \mathbf{z}|\Phi, \Theta) = \prod_i \sum_z \hat{\phi}_{w_i|z}\hat{\theta}_{z|d_i}$$

For increased accuracy, we average parameter estimates over consecutive samples from the Markov chain. We can now use our estimated parameters Φ and Θ to compute a variety of useful distributions such as which documents are similar to each other, which words are similar to each other and by sampling over new data we can easily incorporate new documents into our model without having to re-run the entire algorithm.

3. RANKING RESOURCES

We now describe formulas for ranking resources using the parameters that we have estimated in the topic models described above. Given a query q we wish to return to the user a ranked set of resources ($d \in D$) according to their likelihood given the query under the model, which in the case of LDA can be estimated as follows:

$$\begin{aligned} P(d|q) \propto P(d)P(q|d) &= P(d)\prod_{w\in q} P(w|d) \\ &= P(d)\prod_{w\in q}\sum_z P(w|z)P(z|d) \\ \textit{where} \qquad & P(d) = \lambda(\frac{N_d}{N}) + (1-\lambda)\frac{1}{D} \end{aligned}$$

Notice that the ranking formula consists of the product of 2 distinct parts; a prior on the probability of the resource, $P(d)$ and the probability of the query given the resource, $P(q|d)$. Note that we smooth the document prior by linearly interpolating with a uniform distribution, this was deemed necessary due to the fact that the length of a "document" in this case does not have the same meaning as it would in a standard IR system. In the next section we use a large sample of data obtained from the popular social bookmarking site delicious to evaluate the performance of these models.

4. EXPERIMENTS

In order to evaluate the relative performance of our models on real-world data we performed a crawl of the popular social bookmarking site delicious. To ensure a random sample of recent data we began by downloading the 100 most recent URLs submitted to delicious and recorded the usernames of the users who bookmarked them, continuing until we had collected a sample of 60,663 unique usernames. For each of these usernames we downloaded the respective user's 100 most recent bookmarks (as this is the largest number of bookmarks the API will allow access to). This resulted in

not all users having the maximum number of bookmarks with 31% of the users having less than 100.

Each "document" (URL) is uniquely identified by computing a 32 bit MD5 hash of the complete URL, each URL and user in the data set was assigned a unique and anonymous ID number. To clean the resulting data set, we selected only the URLs which had been bookmarked by more than 2 unique users to ensure that all resources will always exist at least once in the training data. In order to give our systems reasonably complete user profiles to work from we selected only the users who had bookmarked more than 60 unique URLs from the remaining data after the first pass. Each remaining bookmark is a triple consisting of a URL identifier, a user identifier and a set of tags. We parsed the set of tags for each bookmark and finally removed all tags that appeared less than 2 times in the data set.

The final dataset includes 111,232 unique resources (URLs) with a total of 569,117 individual bookmarks by 9,587 users. The vocabulary size was 14,023 and the total number of tag assignments was 2,473,738.

4.1 Evaluation methodology

We separated the dataset into training and testing subsets by retaining the last 10% of bookmarks by each user for testing. In doing so we ensure that the test data is distributed over users in the same way as the training data. In order to generate queries to input into our ranking algorithms we use the set of tags from each test set bookmark as a pseudo query. We now need some form of relevance judgement for each pseudo-query and since we know what resource was chosen for each bookmark we can classify a ranked resource as being relevant if it is the same resource the user actually bookmarked.

We have chosen to use this method as we are interested in personalised results, therefore only the user(s) who originally tagged the resource can really say whether it is truly relevant to them or not. We believe this will accurately reflect the performance of a live system and is likely to actually give a slight under-estimate of the true performance.

In order to evaluate ranking performance we calculated the success at rank k (S@k)[1] and the mean reciprocal rank (MRR). These 2 measures are briefly described below:

S@k - "success at rank k" the ratio of times where there was at least 1 relevant document (resource) in the first k returned.
$S@k = \frac{1}{|q|}\sum_{i}^{|q|} I(rank(d_i, q_i) \leq k)$

MRR = "mean reciprocal rank" the multiplicative inverse of the rank of the first relevant suggested resource, averaged over test resources.
$MRR = \frac{1}{|q|}\sum_{i}^{|q|} \frac{1}{rank(d_i, q_i)}$

Since we are primarily interested in how well these models rank URLs we report the S@k and MMR up to rank 10 as they are the most commonly reported in other literature since people tend to only pay attention to the first page of results in a ranked list.

[1]We note that since we only have one bookmarked URL per set of tags, precision at rank k (P@k) is equal to S@k/k and thus we do not report it separately.

4.2 Parameter settings and sampling

We experimented with a large range of parameter settings for both the number of topics in each model, (discussed further below), and the hyperparameter settings for each of the prior distributions. We set the concentration parameters α and β to be 25.0 and $0.1W$ respectively, which means the α setting is slightly lower than is common in the literature [3]. We found that a slightly smaller value provided better results, perhaps because the average length of a "document" (resource) in these systems is much less than in a more standard IR corpus. We set λ, the smoothing parameter for the probability of a resource to 0.5.

For sampling we use the Rao-Blackwellised Gibbs sampler [3]. For all models we sampled the chain for 300 iterations in total, as this appeared to consistently give good convergence in terms of model likelihood, and discarded the first 200 samples as chain "burn-in". The remaining 100 samples from the end of the chain were averaged over to obtain the final parameter values.

4.3 Baselines

In order to usefully evaluate the performance of the topic models we chose 3 different baselines; SMatch - which emulates the kind of simple matching formulas currently used when searching social tagging sites, Okapi BM25 - a popular and quite robust probabilistic retrieval framework and BayesLM - a competitive baseline Language Model with Bayesian smoothing. For each of the baselines we optimised any free parameters to ensure a fair and unbiased comparison with the topic models. Here we briefly describe the formulas for these models:

SMatch $score(d, q) = \sum_{w \in q} N_{w,d}$

BM25 $score(d, q) = \sum_{w \in q} IDF(w).\frac{N_{w,d}(k_1+1)}{N_{w,d}+k1(1-b+b\frac{|d|}{avgdl})}$
where $IDF(w) = \frac{N-N_w+0.5}{N_w+0.5}$, $|d|$ is the length of resource d and $avgdl$ is the average length of a resource over the whole training corpus. k_1 and b are free parameters which we optimised to 2.0 and 0.1 respectively.

BayesLM $P(d|q) = P(d)\prod_{w \in q} \frac{N_{w,d}+\mu(N_d/N)}{N_d+\mu}$
where μ is the Bayesian smoothing parameter which we optimised to 0.75.

Note that BayesLM is the same as the non-personalised model used by Wang et. al. [6] except that we have adapted it to deal with queries of lengths greater than one. We tried using their full personalised model as a baseline, but found that it performed extremely poorly. This is perhaps because we are using a much larger data set with a vocabulary 14 times larger than theirs. In this case their choice to use raw tags as user profiles (rather than reduced dimensionality features as in this paper) may have resulted in significant overfitting and poor generalisation. We therefore do not report results from their personalised model.

5. RESULTS

Table 1 shows the results of the ranking experiments for all of the models, for the topic model we set the number of topics at 250. Between the more "conventional" ranking methods we see that the language model with Bayesian

	S@1	S@5	S@10	MRR@10
SMatch	0.0555	0.1372	0.1860	0.0900
BM25	0.1701	0.2975	0.3376	0.2238
BayesLM	0.1819	0.3299	0.3772	0.2440
LDA	0.1994*	0.3397	0.3936*	0.2579*

Table 1: Ranking performance of all models on the test data set. $*$ indicates the result is significantly better than all baselines ($p < 0.05$).

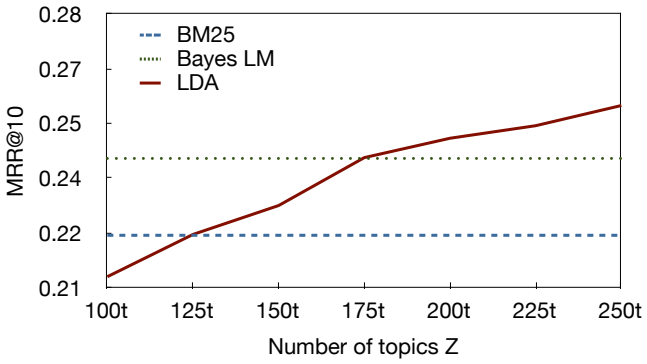

Figure 1: MRR@10 over varying numbers of topics.

smoothing has the best overall performance and considering its relative simplicity, it performs very well. BM25 is clearly less suited to this kind of data than it is to more normal documents and the SMatch algorithm - unsurprisingly - returns particularly poor results.

Comparing the "conventional" models with the topic model results show that over all metrics the topic model performs significantly better than the baselines. This is in contrast to results from previous work into ranking using topic models [7] and perhaps highlights the difference between the "documents" constructed from social tagging data and much longer real-world documents more commonly discussed in IR literature. In the case of social tagging data, the topic model's generalisation of the data and ability to deal with some of the vocabulary problems noted earlier are much more beneficial than perhaps they are with more normal corpora.

5.1 Varying the number of topics

When using hidden topic models an important consideration is how complex a model we should use in terms of the number of latent topics. Therefore we estimated parameters for the topic model over different numbers of topics to see how retrieval performance was effected. Figure 1 shows the results for the metric MRR@10 over the range of topics from 100 to 250 with increments of 25. We also show the results from the 2 most competitive non-topic model baselines to allow direct comparison.

One can see quite clearly from the figure that as we increase the number of topics, the performance also increases. It is apparent that we could achieve even better ranking performance if we were to increase the number of topics even further, however we would expect that at some point performance would peak and we would then be in danger of overfitting the model. Furthermore when using such systems a balance should be made between model complexity in terms of topics and ranking performance, since the amount of time required to rank resources using the models is linear in the number of topics.

6. CONCLUSIONS

In this work we have discussed the problems facing ranking algorithms when dealing with social tagging data and proposed the use of hidden topic models to deal with its inherent sparsity and vocabulary ambiguity. We highlighted the 2 most prominent issues resulting from this kind of data and indicated how such models might be able to at least partially overcome these obstacles.

In order to test the relative performance of the models we proposed an evaluation framework utilising real data obtained by crawling the popular social bookmarking website Delicious and briefly described 3 non-topic model baselines including one previously used to research ranking in a social annotation setting. Finally we described and analysed the results of our experiments on the social tagging data and showed that our intuition of using topic modelling to overcome the vocabulary problems in tagging systems was appropriate.

The results showed that for social tagging data, the topic modelling approaches provided better resource rankings than even the most competitive baselines and outperformed them all by a statistically significant margin. In future work we would like to explore more complex models, perhaps including information about the users into the models. Or where we incorporate more information including perhaps the actual content of the resources or by including temporal information in the model. We also wish to explore sampling and ranking methods that do not assume all queries are the same and instead adapt the rankings algorithms to better suit each individual query.

7. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, (3):993–1022, 2003.

[2] S. Golder and B. A. Huberman. The structure of collaborative tagging systems. In *Journal of Information Science*, volume 32, pages 198–208, 2005.

[3] T. Griffiths and M. Steyvers. Finding scientific topics. *National Academy of Science*, 2004.

[4] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1/2):177–196, 2001.

[5] R. Hooper. Indexer consistency tests—origin, measurements, results and utilization. Technical report, IBM, Bethesda, 1965.

[6] J. Wang, M. Clements, J. Yang, A. P. de Vries, and M. J. T. Reinders. Personalization of tagging systems. In *Information Processing and Management: an International Journal*, volume 460-1, pages 58–70, 2010.

[7] X. Wei and W. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR 2006)*, 2006.

[8] P. Zunde and M. E. Dexter. Indexing consistency and quality. *American Documentation*, 20(3):259–267, April 1969.