Building User Profiles from Topic Models for Personalised Search

Morgan Harvey and Fabio Crestani
University of Lugano
Faculty of Informatics
Lugano, Switzerland
{morgan.harvey, fabio.crestani}@usi.ch

Mark J. Carman
Faculty of IT
Monash University
Melbourne, Australia
mark.carman@monash.edu

ABSTRACT

Personalisation is an important area in the field of IR that attempts to adapt ranking algorithms so that the results returned are tuned towards the searcher's interests. In this work we use query logs to build personalised ranking models in which user profiles are constructed based on the representation of clicked documents over a topic space. However, instead of employing a human-generated ontology, we use novel latent topic models to determine these topics. This means that the topic space used is determined based purely on the query log itself and therefore does not require human involvement to define the topics.

Our experiments show that by subtly introducing user profiles as part of the ranking algorithm, rather than by re-ranking an existing list, we can provide personalised ranked lists of documents which improve significantly over a non-personalised baseline. Further examination shows that the performance of the personalised system is particularly good in cases where prior knowledge of the search query is limited. This is especially useful as these are the cases where we are unable to rely on prior clicks to determine a good ranked list and must instead rely on the ranking model.

Categories and Subject Descriptors

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

General Terms

Machine Learning, Modelling, Experimentation

Keywords

Personalised Search, Topic Models, User Profiles, Query Logs

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...$10.00.

1. INTRODUCTION

The vocabulary problem, where people use the same terms to describe different needs, is a well-known issue affecting Information Retrieval systems and was identified early on in the field's development. Despite this, most IR systems treat all users equally and, given a specific query, will attempt to return an optimal ranked link for the "average user". Recent years have seen a gradual increase in the use of personalisation to improve search results, corresponding with an equally gradual increase in our understanding of how to tackle the problem. Personalisation can be seen as a special case of augmentation to search systems where additional context, beyond merely the search query issued, is used to enhance rankings. The key idea is that by understanding something about the user issuing a query, we can tailor the ranked list presented such that the likelihood of highly ranked documents being relevant is increased. Much early work into such techniques was unsuccessful and many studies have subsequently shown that great care must be taken when applying personalisation so as to avoid damaging an already near-optimal ranked list [17, 8].

In order to construct a personalised ranking for a given user's query, it is first necessary to have some knowledge about that user's previous search behaviour. This prior evidence is often referred to as a user's "profile" and should, in some sense, represent the topical interests of the user. This profile can be built by considering searches made by the user prior to the current one and, for each of these searches, the query terms used and the documents clicked. In early work these profiles were constructed using the raw terms of prior queries or the content of the clicked documents, usually in the form of language models, however this often proves to be ineffective, perhaps because such a representation of interests is too fine-grained given the limited amount of data available. An approach for dealing with this sparsity is to instead base the profile on the main topics discussed in each document, where the topical description of the document is obtained from a human-generated online ontology, such as the Open Directory Project [6, 9, 16]. This approach introduces the problem, however, that many documents may not be present in the online categorisation scheme and requires that people are involved in determining the correct categories for each document, a process that is both expensive and error-prone.

Click-through data, in the form of query logs, is a potentially abundant - since any search engine could generate them - and important source of information regarding search behaviour and can therefore be utilised for personalised search tasks [20]. Query logs generally take the form of triples, consisting of a user ID, a search query and a clicked document. Each clicked document for a particular query is assumed to be either a vote confirming its relevance or a preference for that document over other documents presented higher in the ranked list that were not clicked on. In this work, we follow the intuition that each click on a URL represents an implicit vote for the relevance of the document to the query and that the query words used to search for a document represent that document's content. This framework allows us to construct representations of documents, to build personalised search models and to fairly evaluate the performance of these models, since the query logs represent user-specific relevance judgements in context.

In this work we use query logs to build personalised ranking models in which user profiles are constructed based on the representation of clicked documents over a topic space. However, instead of employing a human-generated ontology, we use latent topic models to determine these topics. This means that the topic space is extracted directly from the query log itself and there is no need for human intervention to define the topics. Our experiments show that by subtly introducing user profiles as part of the ranking algorithm, rather than by re-ranking an existing list, we can provide personalised ranked lists of documents which improve significantly over a non-personalised baseline. Further examination shows that the performance of the personalised system is particularly good in cases where prior knowledge of the search query is limited. This is especially useful as these are the cases where we are unable to rely on prior clicks to determine a good ranked list and must instead rely on the ranking model.

2. RELATED WORK

The idea of using previous interactions of a user with a search system to construct a user profile has been around for some time, and there is significant variation in ways that the problem has been tackled. The approaches differ based on what length of profile data is used and how the data chosen is then turned into a suitable user profile. In some cases researchers have considered only the information from the current search session in order to build short-term profiles [7, 22], whereas other work has attempted to identify longer-term user interests [13, 15]. Recent work by Bennett et al. [1] has even shown how these short and long-term profiles can be combined. In general short-term data is often too sparse to allow for robust personalisation performance and only delivers solid improvements late in long search sessions, which are relatively rare. In this work we focus on long-term click data to build user profiles as it provides a richer source of information about the user's interests and preferences.

Once prior interaction data has been chosen, it must then be converted into a user profile which should form a representation of the user's interests, be they long-term or simply with regard to the current session. These profiles can be generated in a number of different ways. Some approaches use vectors of the original terms [7, 13], often weighted in some fashion. Others attempt to map the user's interests onto a set of topics, which are either defined by the users themselves [14] or extracted from large online ontologies of web sites, such as the Open Directory Project (ODP) [6, 16, 9].

Dou et al. [8] investigated a number of methods for creating user profiles and generating personalised rankings using query logs. Their approach was to use a set of pre-defined interest categories and a K-nearest neighbour approach for clustering similar users. In this work we take a similar view that by reducing the dimensionality of the data we can get better results, however we use more principled techniques that do not rely on predefined categories but derive these from the data as part of the estimation process. The authors found that personalisation is not appropriate for all users and/or queries and may even harm performance. For example, in the case of highly unambiguous queries (e.g. navigational queries such as "google"), where the unpersonalised ranking is close to optimal for all users. In fact, for queries which are both unambiguous and common, optimal results can be obtained by simply ranking documents in order of their prior probability of being clicked for that query. However, this approach is clearly not feasible for the large number of queries where either scant or no prior click data is available.

Teevan et al. [18] confirmed these results and investigated for what kinds of queries personalisation techniques most improved ranking performance. They found that the level of ambiguity of the query provides a good indication of how much benefit will be gained from personalisation. For queries of low ambiguity (where all users tend to find the same results relevant) the personalisation can have a negative impact on performance. This work indicates that we must be careful when designing such systems to ensure that too much weight is not given to prior user preferences in deference to the unpersonalised document score. In later work Teevan et al. [17] demonstrate that the potential that each user/query pair holds for effective personalisation can in some cases be predicted a-priori, allowing the system to select between personalised and unpersonalised rankings.

More similar to the work presented in this paper is that of Harvey et al. [12] and Carman et al [4] where the authors introduce new models based on LDA for the problem of personalised search. These include a user-topic distribution directly in the model, thereby considering the user as part of the generative process. When evaluating these models using query log data it was found that they had an overall negative effect on the ranked lists produced and were therefore unable to improve upon the unpersonalised LDA baseline. The authors note that this is perhaps due to the user becoming too influential in the model and overpowering the perhaps generally more useful information from the documents themselves.

We now present an approach to query log-based search personalisation using sets of latent topics derived directly from the log data itself where the user is not specifically included as part of the generative process but rather is subtly introduced as part of the ranking formula. By means of a large-scale experiment we are able to demonstrate performance improvements over an unpersonalised baseline and show that this new model is particularly effective in cases of sparse prior data where click frequencies cannot be utilised to generate good ranked lists.

3. BUILDING A PERSONALISED MODEL

As stated in the introduction, a basic tenet of personalisation is the idea that information regarding a user's interests and preferences can be garnered from their previous interactions with a search system. More concretely, the idea is to use the terms from the query the user submitted and the specific document(s) (or, in the case of web search, URLs) that they clicked on in the results list for the query, to build a topic level description of the user. The clicked documents should then represent solutions to the actual information need that the user expressed via the query. For example, given a potentially ambiguous query such as “java”, a user interested in computer science is likely to click very different documents from a user who is interested in coffee.

3.1 Query Difficulty

Of course, if we have observed a given user/query pair before, a very simple way to exploit this information about prior clicks is simply to assume that the user will again click on the same document(s) as before. In the vast majority of cases, however, the query/user pair will be novel and we will not have such specific prior information to work with. If the query has been observed many times before, but always by other users, then we may still be able to use this information to provide a good ranking. Consider an unambiguous query such as “facebook” where almost all users will want to click on the same URL. In cases such as this a sensible option is to simply rank the documents in descending order of prior click frequency [8].

Such unambiguous queries can be identified by a measure known as *click entropy.* The click entropy of a observed query q is defined as follows:

$$\mathbb{H}_q = \sum_{d \in D(q)} -P(d|q)\ \log_2 P(d|q)$$

where $D(q)$ is the set of clicked documents for query q and $P(d|q)$ is the (relative) frequency with which document d was clicked amongst all the clicked documents for query q. Entropy values vary in the range zero to the logarithm of the number of distinct documents clicked on for a query. Thus the range of values depends on the query, making it problematic to compare click entropy values across queries. One way to deal with this issue is to report normalized entropy values instead, where we limit the range of values to [0,1] as follows:

$$\tilde{\mathbb{H}}_q = \mathbb{H}_q / \log_2 |D(q)|$$

We will be reporting this measure in our experiments below.

Queries with low click entropy are good candidates for the simple “collaborative” ranking method mentioned above, since in the vast majority of cases we can do no better than this ranking. Conversely, queries with high click entropy are more complicated to deal with and thus the ideal ranking will likely depend on the user who submitted the query.

A second indicator of query “difficulty” is the length of the query since longer queries by their very nature contain more (discerning) information and are therefore less likely to be ambiguous. It has been shown that the difficult queries (those which are short and/or have high click entropy) are the ones for which personalisation can potentially deliver significant ranking improvements [17].

3.2 Representing documents

If we do not have sufficient prior click data to base our ranked list on then we must still find a way to rank documents in decreasing order of probability of being relevant to the query. This is where more traditional information retrieval methods can be used. An unpersonalised ranking can be generated by ranking documents in descending order of similarity to the query, or equivalently in descending order of likelihood that the query was generated by the document. In order to do this we must first have some representation of each document in the collection. In this work we use query logs as a source of data for both training and testing our models and therefore wish to construct document representations from these logs. To do so for a document d, we consider all of the terms of the queries in the log which resulted in the user clicking on d, conflating these terms over all users and queries. This follows the theory that queries should be random draws from the Language Models of the documents for which they are relevant and it has been shown that queries and URL content are strongly correlated [3]. Therefore, provided we ensure enough queries exist to represent each URL, they should describe them well.

Once we have these document-specific language models we must determine how best to represent them. Due to the relatively sparse nature of these language models and the success of using such methods on short documents in the literature [11, 21], we investigate the use of topic models to represent the documents over a reduced-dimensionality latent topic space. In doing so we take a similar approach to many personalisation models in the literature [6, 16, 9], namely that lower-dimensional categories are a better representation of a document’s topical coverage than its raw terms. However, instead of obtaining topic allocations from an online ontology, which may have poor coverage, low levels of granularity and a lack of novel vocabulary, we attempt to derive topics from the data itself.

Topic models attempt to probabilistically uncover the underlying semantic structure of a collection of documents 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. 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 query terms over URLs to be both sparse and noisy.

3.3 Topic models for personalising retrieval

In this section we briefly discuss a latent topic model that extends on Latent Dirichlet Allocation (LDA) [2, 10] and was first applied to query-log based personalised retrieval, without success, by [4]. We use the model to derive topic allocations for each of the documents and to determine each user’s topical interest profile. Finally, we discuss variations on the basic model that allow for successful personalisation.

Figure 1 shows a graphical model diagram for a personalisation topic model. The model involves an observed document d, a latent topic variable z, an observed word w and an observed user u. This structure is repeated for all words in a user’s query[1], all queries by the user and all users in the log. Here we make the modelling assumption that the

[1]We also experimented with a model in which a single topic variable was associated with each query as opposed to each keyword within the query (effectively holding the topic constant over the terms in the query) but observed poorer performance with respect to the model presented.

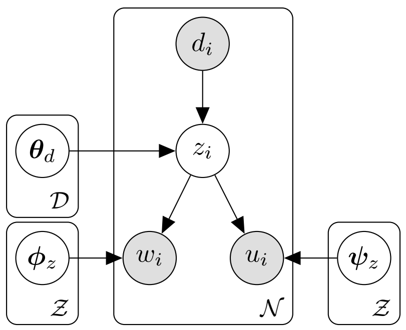

Figure 1: Simple topic model for personalised retrieval (used for ranking).

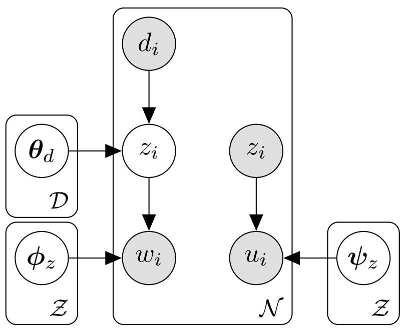

Figure 2: Actual model used for parameter estimation.

user, as well as the word, is dependant on the topic. That is, given the topic distribution of the document, there will be an number of words chosen at random from those topics to describe that document and there will be a number of users who chose to click on that document. These users will be "chosen" according to the topics covered by the document, encoding the idea that users probabilistically choose documents based on their own topical interests and how well these match to the document's topical coverage.

The parameters of this model are a probability vector over topics for each document θ_d, a probability vector over words for each topic ϕ_z and a probability vector over users for each topic ψ_z. Symmetric Dirichlet priors with hyperparameters α, β and γ are placed over the three distributions in order to prevent them from overfitting the data. The hyperparameters essentially act as pseudo counts allowing the model to fall back on uniform distributions in the event of sparse data. Given the prior distributions, expected values for the parameters under their respective posterior distributions are simply:

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

$$\hat{\theta}_{z|d} = \frac{N_{z,d} + \alpha\frac{1}{Z}}{N_d + \alpha} \tag{2}$$

$$\hat{\psi}_{u|z} = \frac{N_{u,z} + \gamma\frac{1}{U}}{N_z + \gamma} \tag{3}$$

Here $N_{w,z}$, $N_{z,d}$ and $N_{u,z}$ are counts denoting the number of times the topic z appears together with the word w, document d and user u respectively. N_z and N_d are the number of times topic z and the document d occur in total. W is the vocabulary size, Z is the number of topics and U is the number of users.

Exact inference for topic models is intractable, however a number of methods of approximating the posterior distribution have been proposed including mean field variational inference [2] and Gibbs sampling [10]. 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. In our case, each state of the Markov chain is a complete assignment of topics to words in the queries. In Gibbs sampling the next state in the chain is reached by resampling all variables from their distribution when conditioned on the current values of all the other variables. After sufficient iterations of the sampler, the Markov chain converges and the parameters of the model can then be estimated. We assume that the chain has converged when we observe minimal change in the model likelihood over successive samples. For increased accuracy, we average parameter estimates over consecutive samples from the Markov chain.

Using the distributions obtained from this model we should be able to construct a ranking formula which, given a query, will consider the probability of each document given both the words in the query and the interests of the user who submitted it. However, as outlined earlier in the paper, in order for personalisation to work it must be applied very subtly. By directly including the user in the topic model we are saying that his/her topical interests are equally important when describing a document he/she has clicked as the words assigned to that document to describe it. The work of Carman et al. [4] demonstrated that this assumption is clearly far too strong as they were unable to obtain successful results from similar models. Instead we consider a different model which does not explicitly include the user in the Markov chain topic sampling but instead calculates an interest distribution for each user after the sampler has converged.

This alternative approach is depicted in Figure 2 where we see that the user does not play a part in the sampling. After the Markov chain has converged, samples from the chain are used (as per normal) to calculate the 3 posterior means (using Equations 1-3 above). The estimates for each user's interests over the topic space (or more precisely the distribution over users for each topic ψ_z) are still obtained, however the sampler does not use these estimates to calculate the conditional distribution over topics when sampling the topic to assign to each word position.

The intuition behind this model (i.e. calculating the probability of a user given a topic and not vice-versa) is that we wish to capture the idea that a user clicks on a document given a specific query due, in part, to his/her interests which are expressed over the topic space. We know from our estimates for θ_d which topics are covered by a document and therefore by multiplying this with $P(u|z)$ we can express (a quantity proportional to) the probability that the user u would have clicked on this document, given the user's interests. This means that if the model is confronted with a new query in which none of the constituent terms have been used by the user previously, it should still be able to map the query onto the user's topic-based profile. This would clearly not be the case if were to instead use the raw (unigram) terms to build the user profiles.

Now, given a query, we wish to construct ranking formulae to order the documents in the collection based on the distributions obtained from the latent topic models. In the case of the personalised model, the ranking should in some sense "perturb" the non-personalised ranking to give higher weight to documents which more closely correspond to the user profile.

4. RANKING DOCUMENTS

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

$$\begin{aligned} P(d|q) \propto P(d)P(q|d) &= P(d) \prod_{w \in q} P(w|d) \\ &= P(d) \prod_{w \in q} \sum_z P(w|z)\, P(z|d) \end{aligned}$$

Notice that the ranking formula consists of the product of 2 distinct parts; a prior on the probability of the document $P(d)$, and the probability of the query given the document $P(q|d)$, with the latter being estimated using parameters from the topic model. In our experiments we use the available click information to set the document prior $P(d)$ to be a Dirichlet smoothed estimate base on the relative frequency of clicks on that particular url in the query log:

$$\hat{\pi}_d = \frac{\#click(d) + \delta \frac{1}{|\mathcal{D}|}}{\sum_d \#click(d) + \delta}$$

So in terms of the parameters from the topic model we can write the ranking formula as:

$$score(d, q) = \hat{\pi}_d \prod_{w \in q} \sum_z \hat{\phi}_{w|z}\, \hat{\theta}_{z|d}$$

For the personalised ranking model, we also know which user issued the query and can therefore include that user's preferences into the ranking formula. We do that by simply ranking documents according to their likelihood given both the query *and the user* as follows:

$$\begin{aligned} P(d|q, u) &\propto P(d) \prod_{w \in q} P(w, u|d) \\ & P(d) \prod_{w \in q} \sum_z P(w|z)\, P(u|z)\, P(z|d) \end{aligned}$$

Now the estimate of the probability of a document includes the probability of the user clicking it, given its similarity to the user's interests over the topic space.

We extend this basic personalisation model by introducing an additional parameter λ in the range zero to one, which we use to weight the probability of a user given a particular topic $P(u|z)$ as follows:

$$\tilde{P}(d|q, u) \propto P(d) \prod_{w \in q} \sum_z P(w|z)\, P(u|z)^{\lambda}\, P(z|d)$$

Dataset	Data set
users	6,581
URLs	15,996
vocabulary size	53,132
queries	2,236,156
word occurrences	6,289,262
average queries/user	340
average queries/url	140
average words/query	2.9
queries/vocab word	56.3

Table 1: Counts and statistics for the AOL dataset used for experimentation.

Thus we now rank documents according to:

$$score(d, q, u) = \hat{\pi}_d \prod_{w \in q} \sum_z \hat{\phi}_{w|z}\, \hat{\psi}^{\lambda}_{u|z}\, \hat{\theta}_{z|d}$$

This new parameter is of critical importance since it allows us to control, in a coherent and discriminative fashion, the amount of influence that the user's topical interests have on the overall ranking. The intuition behind the introduction of this parameter is that documents likely tell us more about their own topic distribution than the users who click on them do.

5. EXPERIMENTS

We now discuss the experiments we performed in personalised retrieval, comparing and anlaysing the performance of our topic model based personalisation approach with a unpersonalised baseline model.

5.1 Preparing the datasets

To evaluate our models on real-world data where each query was made in context we made use of the AOL Query Log dataset. The log contains the queries of 657,426 anonymous users over a 3 month period from March to May, 2006. It is, as far as we know, the only publicly available dataset of sufficient size to perform our analysis. We protected user privacy by analysing results only over aggregate data.

To clean the data we first selected those queries which resulted in a click on a URL. We then selected only those URLs for which more than 100 users had clicked on at least once. We then selected only those users with more than 100 remaining queries. This ensures that all users in the dataset have a reasonably large number of queries from which to build the personalisation models and that the documents constructed for each URL from the queries are of a reasonable size. In order to parse the queries we first separated the words according to whitespace. All punctuation was removed and Porter's algorithm was used for stemming. We did not remove any stopwords but did remove any singleton terms as it is not possible that such a term would exist in both training and testing sets and therefore they would be useless for ranking. The resulting reduced data set is described in more detail in Table 1.

5.2 Evaluation methodology

We separated the dataset into training and testing subsets by retaining the *last 5%* of query log entries for each user for testing, rather than a random split. In doing so we ensure that the test data is distributed over users in the same way as the training data. This also ensures the data are in the correct chronological order: i.e. all of the log entries used for testing were submitted by their respective users after the last training data point. In order to generate queries to input into our ranking algorithms we use the set of terms from each test set query. We also need some form of relevance judgement for each query and since we know which document (URL) was chosen for each query in the log, we can classify a ranked resource as being relevant if it is the same URL the user actually clicked.

We have chosen to use this method as we are interested in personalised results, therefore only the user(s) who originally submitted the queries can really say whether a given URL is truly relevant to them or not. This is in contrast to the more standard approach where relevance judgements are generated by evaluators "after the event" and not by the original searchers themselves. 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)[2] and the mean reciprocal rank (MRR). 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.

Since we are primarily interested in whether or not the personalisation is improving the ranking performance or not versus the baseline LDA model we also report a 3rd metric that we refer to as Personalisation Gain (P-gain). This metric simply compares the number of times the personalisation algorithm improves the ranking with the number of times it worsens it.

$$\text{P-gain} = \frac{\sum_i^Q \mathbf{1}_{\Delta r(d_i,q_i)<0} - \mathbf{1}_{\Delta r(d_i,q_i)>0}}{\sum_i^Q \mathbf{1}_{\Delta r(d_i,q_i)<0} + \mathbf{1}_{\Delta r(d_i,q_i)>0}}$$

Here Q denotes the number of queries, $\mathbf{1}_A$ is an indicator function that equals one whenever A is true and zero otherwise, and $\Delta r(d, q)$ denotes the change in the rank position of document d for query q resulting from personalisation. This can be more simply expressed as the following:

$$\text{P-gain} = \frac{\#better\ -\ \#worse}{\#better\ +\ \#worse}$$

Using this metric a value of 0 indicates no overall change in the rankings due to personalisation, a positive value indicates an improvement in performance and a negative value indicates a degradation in performance.

5.3 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. The choice of hyper parameters was found to have little impact on performance, suggesting that the models are not sensitive to these parameters. We therefore set the concentration parameters α and β to be 50.0 and $0.1W$ respectively, which is common in the literature [10]. For the personalised model we also set γ to 50.

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

For sampling we use the collapsed Gibbs sampler [10]. For both models we rank the Markov chain for 400 iterations in total, as this appeared to consistently give good convergence in terms of model likelihood. We discarded the first 300 samples of the chain as "burn-in" and averaged parameter estimates over the last 100 samples from the end of the chain.

When using hidden topic models an important consideration is how complex a model we should use in terms of the number of latent topics. We can in fact view each of the topic models as being a class of an infinite number of different models, where the complexity in number of topics is in the range $\{1, \ldots, \infty\}$. There has been a considerable amount of work published on so called non-parametric processes where the best model is inferred automatically based on the training data, the most appropriate for this work being Dirichlet Processes [19]. However these processes add significant further complexity and as such it is generally acceptable to use empirical methods to choose the most optimal parameterisation.

In this work we are not trying to optimise in terms of held-out likelihood but in terms of retrieval performance where these techniques may not be as appropriate. We would expect improvements in the held-out likelihood to taper off before improvements in retrieval performance do. Therefore we estimated parameters for topic models over different numbers of topics to see how retrieval performance was effected and found that performance improvements began to level off after around 125 topics. As a result we make use of models consisting of 150 topics for all of the following analysis.

	S@1	**S@10**	**MRR@10**	**P-gain**
LDA	0.2122	0.4283	0.2765	–
PTM	0.2146	0.4316	0.2791	0.0466∗

Table 2: Ranking performance of models on the test data set over all queries. $\lambda = 0.175$

6. RESULTS

Table 2 shows the results of the ranking experiments for the 2 models. A cursory glance suggests that the improvements delivered by the personalised model are not particularly large. The P-gain statistic shows that, on average, the personalised model is improving upon the baseline in 4.66% of cases. However, upon looking at the results more closely we found that in a large number of cases there was no difference in the rank of the relevant document returned by the 2 models. In terms of raw numbers there are 91,280 queries where the 2 models perform the same, 4,626 where LDA performs best and 6,884 where the personalised model performs best. Since these queries are quite common and the performance metrics are based on taking averages over all queries, this strongly dilutes the impact of the personalised model. We will refer to such queries as *same-rank queries*, queries where the rank position of the relevant document was different will be referred to as *different-rank queries*.

Further inspection showed that the click entropy of the same-rank queries was significantly lower than the click entropy for queries where this was not the case. Furthermore, the queries where the ranking was different were significantly more likely to be novel (i.e. not observed in the training data); overall 45.8% of same-rank queries were present in the training data, however this was only the case for 25.7% of different-rank queries. These observations have 2 important outcomes: 1) for a large number of the same-rank queries we can rely on the prior click data to deliver accurate ranking results and 2) a much larger proportion of the same-rank queries have low click entropy, meaning that they are poor candidates for personalisation.

6.1 Different-rank queries

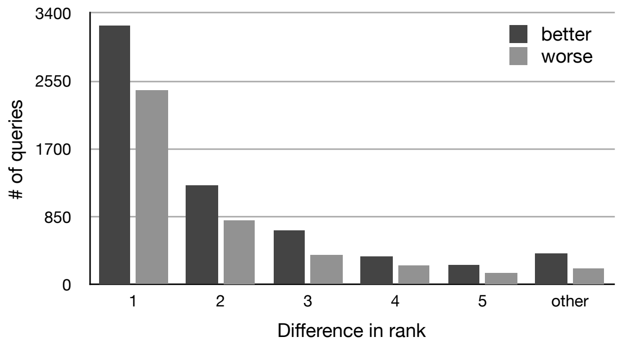

Figure 3: Difference in rank position between the 2 models.

	S@1	S@10	MRR@10	P-gain
LDA	0.2341	0.4766	0.1403	–
PTM	0.2646∗	0.4991∗	0.1599∗	0.1962∗
% improv.	11.5%	4.5%	12.3%	–

Table 3: Ranking performance of models on the test data set over different-rank queries. $\lambda = 0.175$

As a result of these observations, we now focus on queries for which the rankings returned by the models were different. These "harder" queries constitute 14.25% of the total queries. Table 3 shows the same performance metrics, but this time only for these different-rank queries. We can see that for these queries the personalised model is able to deliver much better results in comparison to the non-personalised baseline, registering an improvement in rank in 19.62% of cases. In fact the difference in performance over all metrics is significant [3] (p-value $\ll$ 0.01). The improvements are particularly noticeable in the lower ranks, resulting in a considerable increase in S@1 and MRR.

The difference in ranking performance between the 2 models can be better understood by considering the difference in the ranks of the relevant document. Figure 3 shows the distribution of the difference in the ranking of the relevant document for each query between the 2 models. The darker bars show the number of queries where the ranking was improved, the lighter bars show where the ranking deteriorated, "other" refers to all rank changes greater than 5. The distribution shows, importantly, that the ratio between improved and deteriorated queries increases with the change in rank position. At a rank change of 1 the ratio is only 1.33:1, however it becomes as high as 1.91:1 when we look at queries where the change in rank was greater than 5. This indicates that for a number of queries the personalisation is able to move the relevant document much higher in the rankings, however the opposite case occurs very infrequently.

[3]As determined by 2 sample proportion z-test.

6.2 Impact of query difficulty

Query length	1	2	3	4	> 4
# better	615	1,893	1,685	1,242	1,449
# worse	203	1,082	1,145	953	1,243
P-gain	0.504	0.273	0.191	0.132	0.077

Table 4: Counts of improved and deteriorated ranks and P-gain values for queries of different lengths.

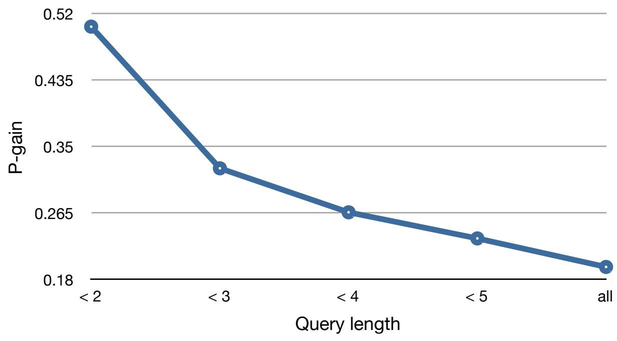

Figure 4: Personalisation performance is query length dependent. Short queries are often more ambiguous and results in better personalisation performance.

As mentioned earlier, we would expect a personalised model to be most beneficial in the case of short and ambiguous queries and perhaps less so for longer queries where the information need has been more thoroughly described. Queries can be described in terms of their "difficulty", with short ambiguous queries being more "difficult" than longer less ambiguous ones. There are a number of measures of query difficulty [5, 18], however 2 common approaches are to look at the length of the query and (when click data is available) the click entropy for that query.

Table 4 details how the performance of our model changes as the length of the queries change. The performance gain of the personalised model is clearly much better for shorter queries, particularly for queries of length 1 or 2, however as the query length increases, the performance of the personalised model - relative to the unpersonalised one - decreases. Regardless of query length, the personalised model is still able to outperform the LDA baseline, however the number of queries for which it is able to produce a better ranking decrease as query length increases. This ties in nicely with the idea that the personalisation is much more effective for ambiguous queries where there is likely to be much more variation between different users. In the case of longer queries, the extra information included in the query reduces the uncertainty and renders the user profile information much less useful. As one would expect the general performance of both models decreases as the query length increases (i.e. as the queries become increasingly less ambiguous). This change in performance as measured by p-gain as query length changes is shown cumulatively in figure 4. This shows, for example, that by focusing purely on queries of length 3 or less we can achieve a p-gain of 0.265. This is an important observation since queries tend to be short and therefore the better performance is obtained for the most common query lengths. In our testing data set queries of length 3 or less account for 70.69% of all queries (72666/102790).

Entropy	**0-0.2**	**0.2-0.4**	**0.4-0.6**	**0.6-0.8**	**0.8-1.0**
# better	398	429	669	605	630
# worse	236	262	319	343	267
P-gain	0.256	0.242	0.354	0.276	0.405

Table 5: Counts of improved and deteriorated ranks and P-gain values for queries of varying click entropy. In general queries with higher click entropy result in better personalisation performance.

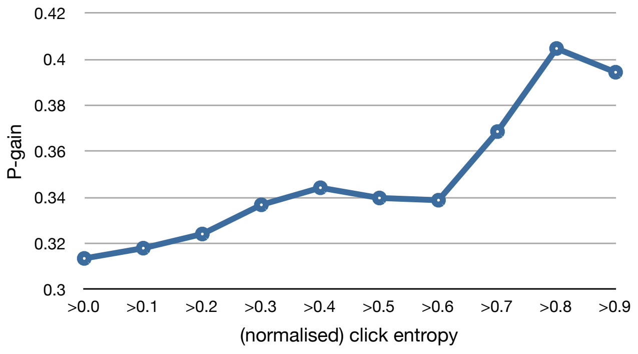

Figure 5: Personalisation performance depends on query ambiguity as this plot demonstrates: For high values of normalised click-entropy the personalisation performance is much higher.

For the more common queries we can measure the query difficulty (ambiguity) more directly in terms of the click-entropy as described in section 3.1.

When initially looking at the click entropy, we did not observe the same relationship. For this metric we actually observed the opposite relationship, although it is not very clear: in general, as the click entropy of the query increases, the relative performance of the personalised model appears to decrease. However, upon further investigation it became clear that quite a large proportion of these entropies were being calculated based on very small numbers of data points, in fact in over 10% of cases the entropy was calculated based on fewer than 5 data points. Clearly when the entropy calculation is based on such a small sample it is highly unlikely to approximate the "true" value over the greater user population. To account for this we considered only the queries for which 20 or more data points were present in the training data set. Table 5 shows how the performance of the models changed as the (normalised) click entropy of the queries increased. By restricting our analysis to only queries that were well represented in the training set we of course reduce the number of data points quite significantly, however the numbers are still large enough to identify general trends. Although the trend is not nearly as clear as it was for the query lengths, we can see that as the click entropy of the query increases, so too does the relative performance of the personalised model (correlation = 0.71). This relationship is more obvious as depicted in figure 5 which shows in finer granularity how performance changes as the click-entropy of the queries increase.

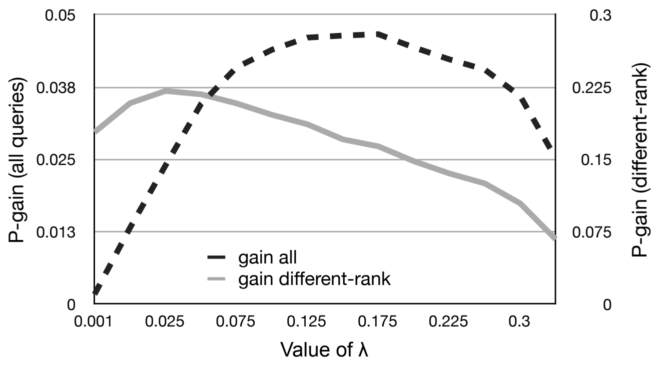

Figure 6: The effect of varying the λ parameter in the personalised ranking algorithm.

6.3 The effect of λ

We introduced a parameter λ into the ranking formula for the personalised model to allow control over the amount of influence the user profile has on the document scores. We tested the effect of this parameter within the range of 0...0.5, where the extreme setting $\lambda = 0$ should collapse the model back to the same estimates as LDA. The effect on performance, in terms of P-gain, over all queries (dashed line) and over just the different-rank queries (solid line) is shown in figure 6. Looking at the different-rank queries we can see that as the parameter value is decreased, the performance seems to increase. However as λ decreases the total number of different-rank queries also decreases, since the differences between the 2 models are becoming increasingly smaller. That being the case, we do not necessarily want to optimise this parameter based purely on performance over this set of queries as we also want to ensure that the positive impact of the personalisation is affecting as many queries as possible. For example in setting λ to 0.025, which appears to yield the best performance, the number of different-rank queries is reduced to just 5,331 (5.2% of the total). If λ is instead optimised for performance over all queries ($\lambda = 0.175$) then the improvement over the subset of different-rank queries is still very high, however the size of this set is increased to 14,656 (14.25%). Note that we have not included points in the plot for $\lambda = 0$ because in this case the algorithm simply collapses back the unpersonalised model and all p-gains are 0.

Figures 7 and 8 further illustrate the effect of varying λ and its relationship with the length of the queries. When comparing these plots to figure 6 we observe approximately the same trends, with the performance over all queries peaking at around 0.175. The performance over only different-rank queries slowly decreases as λ increases, except if the value is set too low, in which case the performance is generally poor. Note that for queries of length one, the perfor-

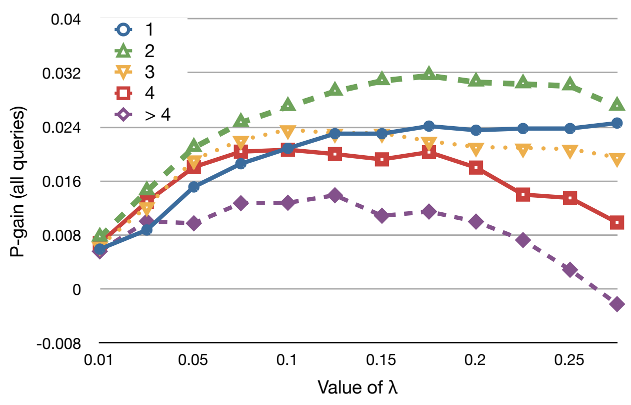

Figure 7: Performance effects of different values of λ for queries of different lengths, showing all queries (not just different-rank queries). Note that short and likely more ambiguous queries of length 1 have a much larger optimal value for λ (greater than 0.25) as compared to queries of length 3 or more, indicating that the importance of the user profile is higher for the shorter queries.

mance over different-rank queries does not appear to have reached its peak at 0.175 and continues to slowly rise after this point. However for longer queries we notice that the performance peaks much earlier and has already degraded by the time λ has reached 0.3, so much so that for queries of length greater than 4 the p-gain is actually slightly negative by this point.

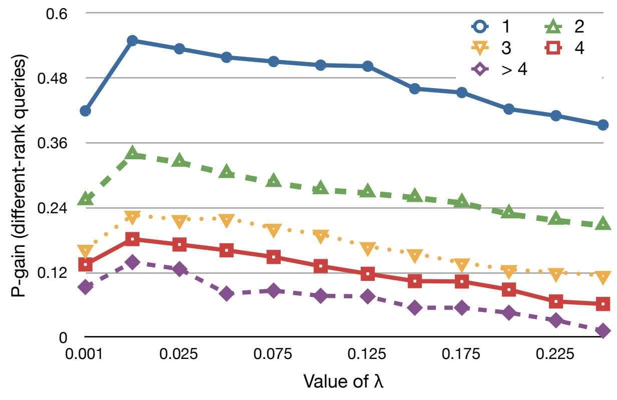

Figure 8: Performance effects of different values of λ for queries of different lengths, showing only different-rank queries.

7. CONCLUSIONS

The results of our analysis indicate that it is possible to improve performance through personalisation by making use of topic-model based user profiles. While in theory, personalisation can offer a path to achieving substantial gains in retrieval performance, in practice performance improvements over all queries will be quite small with respect to the performance of the un-personalised retrieval system. Thus personalisation needs to be introduced with great care in order to obtain gains without adversely affecting average performance.

We have shown that the performance gains for our model are significant for a smaller subset of queries, which can be identified by using query difficulty metrics such as click entropy and query length. Query length was shown to be an excellent indicator of performance and it was shown that there is some correlation between click entropy and performance, although this was not perhaps quite as clear as we might have expected. A better quantity for predicting personalisation performance may take both the click entropy and the query length into account and perhaps even the interaction between these metrics. For example it may be possible to normalise the click entropy by the expected entropy at that query length or it may be useful to consider some linear combination of the metrics to identify when to personalise and when not to.

We believe that there are further gains to be achieved by taking into account to what extent the user profile differs from that of the “average user”, whereby the more particular the interests of the user, the more likely personalisation is to have a positive effect. More generally, we would like to estimate the extent to which the user profile reduces the ambiguity of the query and use that to decide for which query-user pairs to personalise the results. If we can determine that the profile is useful for disambiguating the query, then it makes sense to personalise, otherwise it doesn’t. Such a metric would give a sense of how “contrary” a given user is. For example, if (for most queries) the user regularly clicks on the same URL as the majority of other users then we can say that they are not contrary and should reduce the influence of the user profile accordingly (via λ). If, on the other hand, the user very often chooses URLs contrary to other users then it is likely that an increase in the value of λ will yield better results. We leave these investigations to future work.

In conclusion, this work has presented a new approach to query log-based personalisation which uses latent topic modelling to describe both the clicked URLs and the interests of the users over the same topic space. We note that getting topic-modelling based personalisation to work successfully required not-insignificant alterations to the basic topic model (LDA). Firstly the different parameters of the model needed to be estimated sequentially (rather than contemporaneously), and secondly an additional parameter controlling the influence of the user profile on the ranking needed to be introduced and carefully tuned. By testing this new approach on real click log data we have shown that it is applicable in a “real world” scenario and have shown that it is able to outperform the unpersonalised model in all cases, particularly in the case of difficult queries. These difficult queries - where personalisation performance is best - can be quite easily identified, for example by using simply the query length. Since queries are normally short, they account for a large percentage of the total.

8. REFERENCES

[1] P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short- and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 185–194, New York, NY, USA, 2012. ACM.

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

[3] M. J. Carman, M. Baillie, R. Gwadera, and F. Crestani. A statistical comparison of tag and query logs. pages 123–130, 2009.

[4] M. J. Carman, F. Crestani, M. Harvey, and M. Baillie. Towards query log based personalization using topic models. *19th ACM Conference on Information and Knowledge Management (CIKM)*, pages 1849–1852, 2010.

[5] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 390–397, New York, NY, USA, 2006. ACM.

[6] P. A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschütter. Using odp metadata to personalize search. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 178–185, New York, NY, USA, 2005. ACM.

[7] M. Daoud, L. Tamine-Lechani, M. Boughanem, and B. Chebaro. A session based personalized search using an ontological user profile. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 1732–1736, New York, NY, USA, 2009. ACM.

[8] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. *Proceedings of the 16th international conference on World Wide Web*, pages 581–590, 2007.

[9] S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based user profiles for search and browsing. *WIAS*, pages 219–234, 2003.

[10] T. Griffiths and M. Steyvers. Finding scientific topics. *National Academy of Science*, 101:5228–5235, 2004.

[11] M. Harvey, M. Carman, and I. Ruthven. Improving social bookmark search using personalised latent variable language models. *Forth International Conference on Web Search and Web Data Mining (WSDM)*, 2011.

[12] M. Harvey, I. Ruthven, and M. J. Carman. Improving social bookmark search using personalised latent variable language models. *Forth International Conference on Web Search and Web Data Mining (WSDM)*, pages 485–494, 2011.

[13] N. Matthijs and F. Radlinski. Personalizing web search using long term browsing history. In *WSDM*, pages 25–34, 2011.

[14] A. Pretschner and S. Gauch. Ontology based personalized search. In *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI '99, pages 391–, Washington, DC, USA, 1999. IEEE Computer Society.

[15] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 727–736, New York, NY, USA, 2006. ACM.

[16] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 525–534, New York, NY, USA, 2007. ACM.

[17] J. Teevan, S. T. Dumais, and E. Horvitz. Potential for personalization. *ACM Trans. Comput.-Hum. Interact.*, 17(1):4:1–4:31, Apr. 2010.

[18] J. Teevan, S. T. Dumais, and D. J. Liebling. To personalize or not to personalize: Modeling queries with variation in user intent. *Proceedings of the 30th international ACM SIGIR conference on Research and development in information retrieval*, 2008.

[19] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[20] S. Wedig. A large-scale analysis of query logs for assessing personalization opportunities. In *In Proceedings of KDD '06*, pages 742–747, 2006.

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

[22] R. W. White, P. Bailey, and L. Chen. Predicting user interests from contextual information. In *SIGIR*, pages 363–370, 2009.