21st Century Search and Recommendation: Exploiting Personalisation and Social Media

Morgan Harvey and Fabio Crestani

Faculty of Informatics, University of Lugano, Switzerland
{morgan.harvey,fabio.crestani}@usi.ch

Abstract

Using the Internet to find information and interesting content is now one of the most common tasks performed on a computer. Up until recently, search algorithms returned only one-size-fits-all rankings, resulting in very poor performance for ambiguous search queries. Recent work has demonstrated that contextual information - such as the interests of the searcher - can be utilised to provide more accurate results which have been "personalised" and adapted to the user's current information need and situation. Likewise, information about the user can be brought to bear to mitigate the problem of information overload and filter content so that users are only shown items they are likely to be interested in.

In this book chapter we explore new methods for assisting users to find the information they want by reducing the complexity of the search task through *personalisation*. We explore this problem from the perspective of web search and then by considering a very common form of new socially-generated data - microblogs. We first tackle the problem of search result personalisation in the face of extremely sparse and noisy data from a query log. We describe a novel approach which uses query logs to build personalised ranking models in which user profiles are constructed based on the representation of clicked documents over a topic space. Our experiments show that this model 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.

We then turn our attention to the related problem of recommendation (where the user profile is itself the query) and, more specifically, discuss the possibility of learning user interests from social media data (specifically micro blog posts). We present a short introduction to early work focussing on the difficult task of making use of this vast array of ever-changing data. We demonstrate via experiment that our methods are able to predict, with a high level of precision, which posts will be of interest to users and comment on possibilities for future work.

1 Personalised Search and the Vocabulary Problem

As discussed in the MUMIA memorandum of understanding, the sheer volume of data which has become available in the last decade as a result of web becoming more social (the so called *web 2.0*) will require a new generation of search systems to be developed. This huge new resource will only be useful if the raw data can be transformed into high value information which users can easily manage, understand and exploit. One of the ways to help in making sense of all this data is to use context to intelligently narrow down the amount which is presented to individual users, especially when searching.

Search, particularly on the web, is undeniably an important topic in computer science with several very large companies such as Google, Microsoft and Yahoo dedicating large sums of money to research and development in this field. Using a search engine to find information is often listed as one of the most common tasks performed on the Internet [41]. Up until recently, focus was placed primarily on matching short textual queries with documents, resulting in generally excellent performance for a large proportion of search tasks. Modern search systems consider a large number of features when attempting to determine relevant documents and then to optimally rank these documents in a list. Notable breakthroughs have been made by considering features other than simply the content of the documents, such as the Google PageRank algorithm [7] which also considers the hyperlinks between web pages.

Search engines provide us with the means to rapidly access information and to narrow down the overwhelming number of web pages and resources available on the web in order to find a small number of relevant items. This is usually achieved via the submission of a - typically very short - textual query which the search engine must interpret in order to retrieve the most appropriate pages to present to the user [33]. However, understanding a user's information needs given such a small amount of information is far from a trivial task, especially in cases where the user himself only has a very vague idea of what he is looking for. The traditional approach in Information Retrieval (IR) is to simply return the documents which are in some sense most similar to the query terms, with this often being determined by either proximity in a vector space or via probability theory [16].

As search engines have developed, the scope for dramatically improving result quality has narrowed, making it necessary that researchers focus on more niche problems or leverage new sources of data. A large percentage of searches can now be handled optimally by modern search systems, in the sense that the result the user was looking for is returned at the top rank position, by applying standard IR technology. However, for a smaller number of more "difficult" search queries the opportunity still exists to make significant improvements. These queries are often difficult because they are ambiguous and could potentially be referring to more than one concept or information need (related to the "vocabulary problem", where people use the same terms to describe different needs [23]). In this case the query terms are not sufficient to return useful results and it is not possible to create a ranked list which is optimal for all users. This is where more contextual information, particularly about the user's interests, can be utilised to augment the scanty information provided by the query.

Other basic forms of context can be put to use, such as time and location [35], however a developing source of additional information for search is the plethora of modern social media sites. These sites do not present simply static content, but rather allow users to interact with each other to share information and thoughts and to connect with each other, forming complex social networks [25]. The sheer volume of information submitted to these services gives a far greater insight into human language and behaviour than has previously been possible.

This new data represents an additional source of information about users, their interests, their thoughts and feelings. Users often post web sites they have found and want to share with others, information which could be used to learn what is popular on the web right now and which could be fed into search engines and recommendation algorithms to augment their existing data streams. The responses to questions posted to social networks could even be mined to provide intelligent yet instantaneous answers to similar questions posed by search engine users.

Difficult queries (i.e. those which would benefit most from additional context) can often be detected by measures such as *click entropy* - which is simply the entropy of the distribution of prior clicks on different URLs, given the query of interest - or by looking at the length of the query [48]. 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 [19]. Conversely, queries with high click entropy are more complicated to deal with and thus the ideal ranking will likely depend on the person who submitted the query and so a ranking personalised to the interest profile of the user is likely to yield better results.

Such profiles are often built by considering searches made by the user prior to the current one and, for each of these, the query terms used and the documents clicked. Alternative sources of profile data, such as a user's social media history, are often easier to gain access to and may be more abundant, potentially leading to a more granular understanding of the user's interests. There is however at present little work where this new source of data is exploited, however work has shown that the terms used on Twitter to describe a given URL are useful as descriptions of that web page [28]. In early approaches user 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.

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

An idealised IR system should, given all the information available, rank documents in descending order of their expected relevance to an information need, usually expressed as a short keyword-based query [44]. Most early retrieval systems considered the query in isolation and while this is obviously an extremely strong indicator of what information or resources the searcher is looking for, there are other - perhaps less obvious - clues which can be used, particularly in cases where the query itself is ambiguous. In this section we outline work already conducted in the areas of contextual search and user profiles, personalised search and, lastly, topic modelling.

2.1 Context and User Profiles

There are many different sources of context that can potentially be exploited when attempting to improve search engine rankings, often by subtly altering an existing ranked list [39]. This extra contextual information should allow the system a better understanding of the current search "situation," beyond the often meagre but essential information given by the query. For example, the user's location can be used to select which language(s) should be considered and to focus search results so that they are geographically close to the user [2]. Results can be tailored towards the age and/or linguistic ability of the user by considering factors such as reading level [15] (i.e. how complex the use of language within the document is). A perhaps more obvious source of contextual data is the interests and preferences of the searcher which must be expressed in such a way that the system can make use of them to improve search effectiveness. These representations of preference are often referred to as *user profiles* and can be defined by the users themselves [36, 14] or, instead, automatically by learning from the user's prior interactions with the system.

The idea of using previous interactions of a user with a search system to construct a user profile has been around since the mid-1990s [43], and there is significant variation in the ways that the problem has been tackled [1]. The approaches differ based on what length of profile data is used, which 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 [17, 50], whereas other work has attempted to identify longer-term user interests [37, 42]. Recent work by Bennett et al. [3] has even shown how these short and long-term profiles can be effectively 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 true 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. These profiles can be generated in a number of different ways. Some approaches use vectors of the original terms [17, 37] from the queries or the URLs clicked, often weighted in some fashion. Others attempt to map the user's interests onto a set of topics extracted from large online ontologies of web sites, such as the Open Directory Project (ODP) [14, 45, 24]. Some methods do not make use of any terms, but rather rely solely on which URLs were clicked, given different queries. For example Cao et al. [9] modelled the sequence of queries and clicks on URLs in order to create sequential estimates of most probable future clicks. In this work we use topic modelling techniques to map the original query terms onto a lower-dimensional space which itself is derived from the original data.

2.2 Personalising Search

Dou et al. [19] 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 chapter 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. Dou et al. 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. [48] 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. Building on this earlier work, Teevan et al. [47] later demonstrated 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.

Matthijs et al. [37] constructed user profiles using different textual summaries from each previously-clicked URL including summaries derived from a page's title, content and metadata as well as set of "important keywords" as determined by a Term Extraction algorithm. Term candidates form a summarisation of the full text and were found using a number of linguistic patterns and are assigned a weight based on the frequency of the term and its sub-terms. These profiles were then used to subtlety rerank the top 50 results returned by Google. Their method was tested via a user study in which users were asked to download and use a browser plugin, some participants being then shown the reranked results when performing a Google search, the others receiving the unaltered ranked list. This method was reported to deliver good results, however unfortunately the authors did not attempt to apply it to any large-scale data sets such as a query log.

Rather than re-ranking search results by using profiles based on raw terms, many approaches instead attempt to map user interests and documents onto a set of categories or topics. Doing so can potentially alleviate many of the issues resulting from discordant term use since exact matches between terms in the query and in documents are no longer necessary. Most personalised search models which employ topics use sets of pre-defined, human-curated categories, such as those provided by the Open Directory Project (ODP) [14, 45, 24]. While this is a straightforward approach to acquiring sets of topics, it suffers from the fact that the topics are not derived from the data being categorised and therefore may not be an especially good match, necessitating that some documents are "pigeon-holed" into topics to which they do not clearly belong. Furthermore this method either requires humans to manually label each document - an extremely expensive and time-consuming task - or some method of automatically assigning documents to topics must be devised, often replying on similar approaches to classical IR document matching with all of the inherent vocabulary issues. Instead, it is possible to automatically derive sets of topics from the document collection itself, an idea which has been investigated in recent years [27, 11].

2.3 Topic Models

The idea of automatically modelling the topical content of documents has been present in the IR field for some time and developed from early work - termed Latent Semantic Indexing or LSI - which used a Singular Value Decomposition of document-term matrices to represent the documents in a lower-dimensional space [18]. This idea was reformulated as a probabilistic generative model by Hofmann [30] and then further improved upon by Blei and Jordan [4], who developed the Latent Dirichlet Allocation (LDA) model. Both probabilistic models attempt to uncover the underlying semantic structure of a collection of documents based on analysis of their vocabulary. This latent topical structure is modelled over an often pre-defined number of topics, which are assumed to be present in the collection.

LDA has served as the basic building block for a large number of more complex models, dealing with problems such as image categorisation [22], movie recommendation [29] and even to determine the ancestry of people based on their genes [21]. While early methods of inferring topics from documents [4] were somewhat cumbersome and complex, making it quite a difficult task to extend upon them, more recent approaches are more straightforward and easier to develop and build upon [26]. Wei et al. [49] showed that it was possible to use topic models to improve the performance of search systems by matching queries to documents at a semantic level.

Topic models have been considered as a solution to personalisation. Harvey et al. [27] and Carman et al. [11] introduced new models based on LDA for the problem of personalised search which both 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. Both 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 A Topic Model for Personalised Search

As already stated, 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, 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.

For a document (URL) d, we consider all of the query terms 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 [10]. 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 [27, 49], 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 existing personalisation models [14, 45, 24], 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.

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.

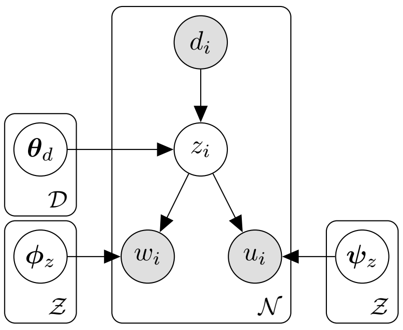

Fig. 1. Simple topic model for personalised retrieval (used for ranking).

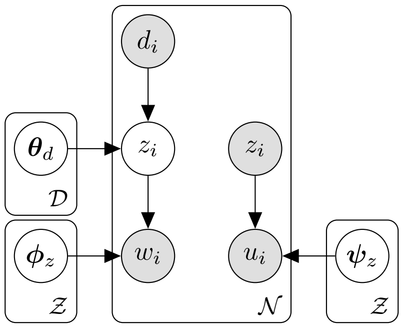

Fig. 2. Actual model used for parameter estimation.

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 user, as well as the word, is dependant on the topic. That is, given the topic distribution of the document, there will be a 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. This modelling assumption expresses 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:

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

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

[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.

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

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 [4] and Gibbs sampling [26]. 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. [11] 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 we 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 based 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:

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

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.

Note that the estimates of relevance to the query as computed here could be combined linearly with standard IR features such as term frequency, as demonstrated by Wei [49]. However, since we are interested in understanding the effect of personalisation on rankings, and not absolute retrieval performance, in this work we experiment purely with the topic model-based ranking algorithms.

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.

5 How Effective is Personalisation?

To evaluate our models on real-world data where each query was made in context we used 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. Users' personalised details were protected by analysing results only over aggregate data. 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. We use each query in the test set as the input query and since we know for each query which document (URL) was clicked, we can classify a ranked resource as being relevant if it is the same URL the user actually clicked. Therefore our relevance assessments for each query are obtained directly from the log and are determined by the author of each query.

In order to clean the data, we first selected those queries which resulted in a click on a URL. Secondly, we selected only those URLs for which more than 100 users had clicked on at least once. Finally, we 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.

We report two standard Information Retrieval evaluation metrics - Success at rank k (S@k) and Mean Reciprocal Rank (MRR). *S@k* is the proportion of instances where the relevant document is ranked at position *k* or higher. For example

S@1 indicates the proportion of queries for which the ranking algorithm is able to return the relevant document at the top rank. MRR is the inverse of the rank of the relevant document averaged over all queries. In addition to these metrics we also report Personalisation Gain (P-gain), which simply compares the number of times the personalisation algorithm improves the ranking with the number of times it worsens it. 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. We compare the new personalised model against a competitive topic-based, but unpersonalised, baseline (Latent Dirichlet Allocation - LDA). Note that we experimented with a variety of different settings for Z (the number of latent topics) and found that for both models the performance peaked at somewhere between 125 and 150 topics. The following analysis is conducted on models with 150 latent topics.

5.1 Results

	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 2. Ranking performance on the test data set

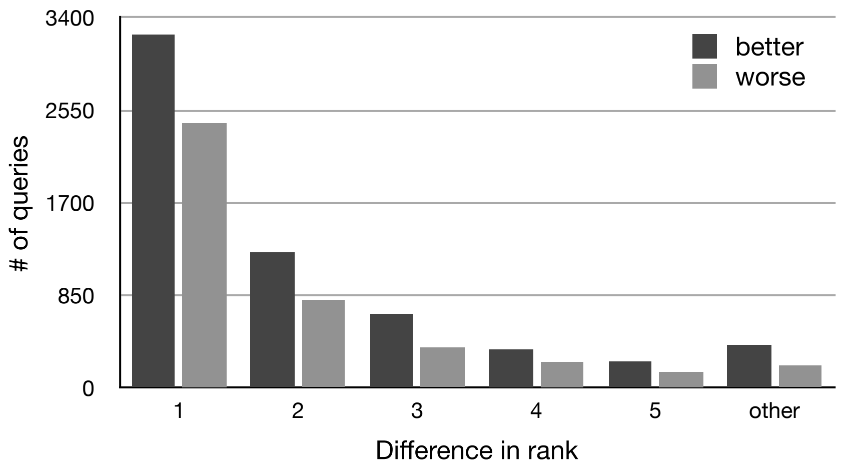

Fig. 3. Difference in rank position between the 2 models.

Table 2 shows the results for the two models. We can see that 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[2] (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 two 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 two 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.

5.2 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

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 3. Performance as query "difficulty" changes.

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 [12, 48], 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 3:top 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

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

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 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). For example, by focusing purely on queries of length 3 or less, we can achieve an overall 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).

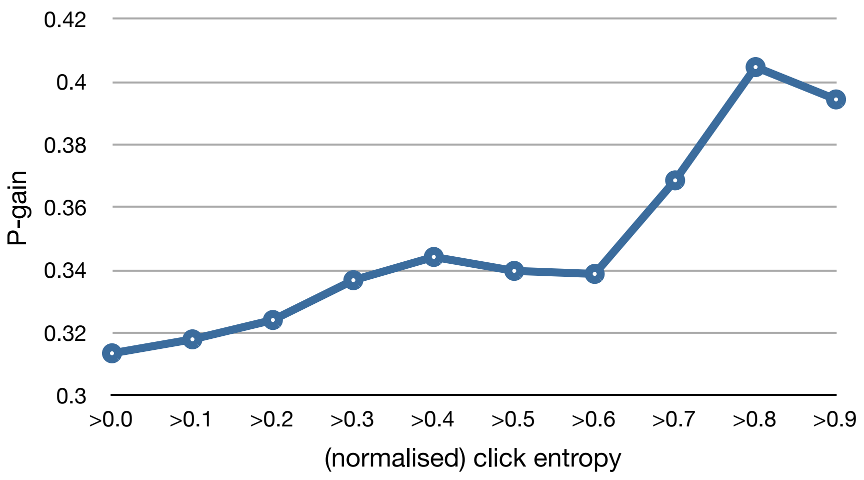

Fig. 4. Personalisation performance depends on query ambiguity as this plot demonstrates: For high values of normalised click-entropy the personalisation performance is much higher.

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

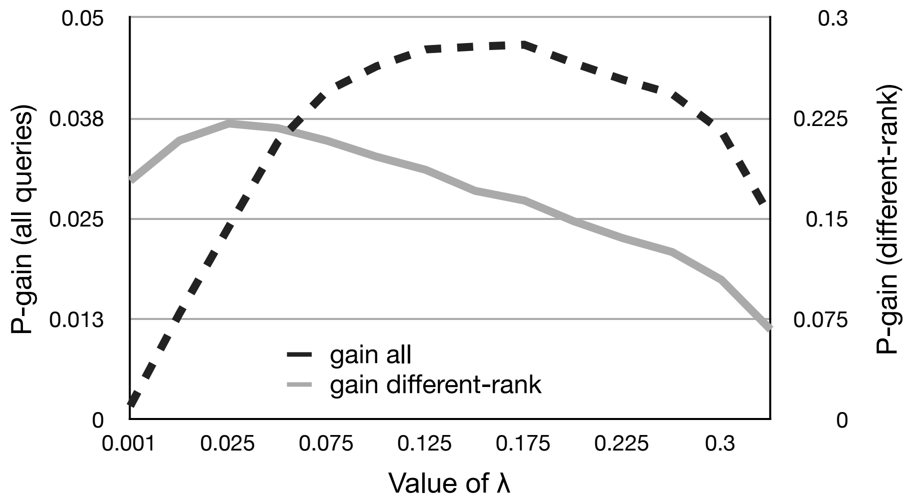

Fig. 5. The effect of varying the λ parameter in the personalised ranking algorithm.

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 3:bottom 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 4 which shows in finer granularity how performance changes as the click-entropy of the queries increase.

5.3 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 5. 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 two 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 6 and 7 further illustrate the effect of varying λ and its relationship with the length of the queries. When comparing these plots to figure 5 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 performance 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.

5.4 Model Performance Summary

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.

6 Recommending Tweets

It can be argued that personalisation and recommendation are in fact two sides of the same coin as they both use knowledge about the user - usually some kind

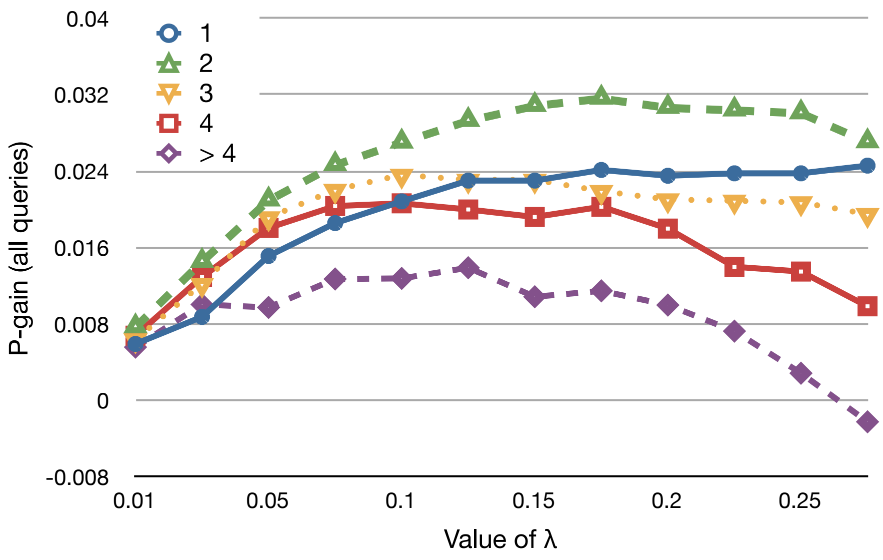

Fig. 6. Performance effects of different values of λ for queries of different lengths, showing all queries (not just different-rank queries). Note that short and likely more ambiguous queries of length 1 have a much larger optimal value for λ (greater than 0.25) as compared to queries of length 3 or more, indicating that the importance of the user profile is higher for the shorter queries.

of profile - to filter down a large list of possible items to present to the user. The key difference being that in the case of personalised search the system is given an input query, whereas in the case of recommendation there is no query, rather the user's interests (as well as other contextual data) alone must be used to narrow the candidate items [1, 40]. Therefore, having shown how personalisation can be used to improve the ranking performance of a search engine, we now turn briefly to the related problem of recommendation and present the preliminary stages of research into personalised suggestion of interesting Tweets.

Search on the web evolves constantly, with social media having an ever-increasing impact and even being included in mainstream search engine results. In the 1990s the main concerns for a search engine were ensuring a good match between query terms and documents returned and to keep spam at bay. However as the web has matured, issues such as freshness, trustworthiness and serendipity have slowly become more important [8]. The rapid growth of social media in in the last decade has provided an unprecedented volume and breadth of data, yet the sheer scale of this new data source can be overwhelming for users [5, 46]. Perhaps the most well-known and popular example of social media is the short messaging service Twitter.

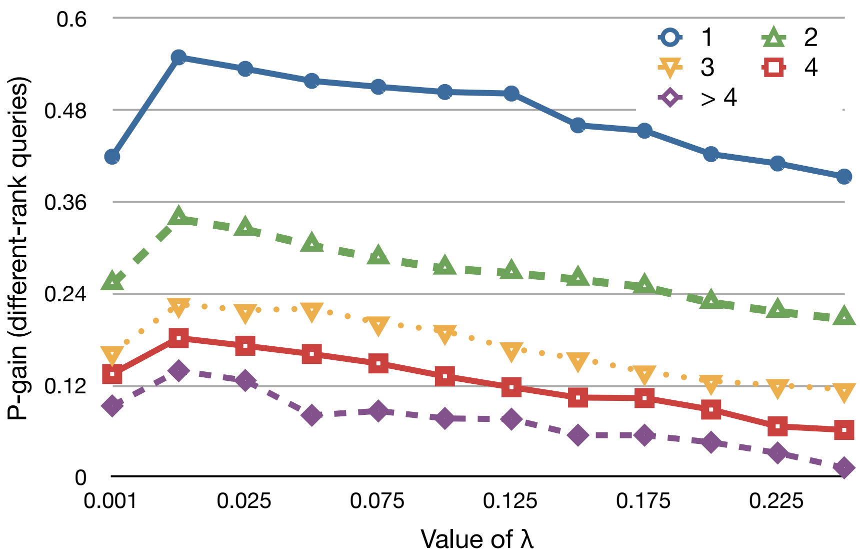

Fig. 7. Performance effects of different values of λ for queries of different lengths, showing only different-rank queries.

Twitter is a socially-focussed short messaging ("micro-blogging") service that allows users to post and read short messages - known as "tweets" - of up to 140 characters in length. In these tweets users post about what they are currently reading, thinking and doing and often post URLs to web sites of interest to them [34, 38]. In addition to being a platform for socially sharing thoughts and opinions, work has shown that Twitter also represents a valuable, user-driven source of information of unprecedented volume [5]. Tweets can provide "specific information, useful links, and insights from personal experiences" [32] and it has been shown that people search for information on Twitter for a wide variety of different reasons and to fulfil very different information needs [20].

Users are encouraged to "follow" others on the service, doing so results in all of the followed user's public tweets (messages) being displayed in the following user's "home timeline;" an up-to-date set of tweets shown to the user after logging in to the service. Analysis of Twitter data has shown [31] that users have an average of 80 friends (people they follow) meaning that their home timelines are being populated with hundreds or even thousands of tweets per day. Clearly this is a case of information overload: there are a huge number of Tweets available to read but this is most likely only a small proportion the user will truly be interested in. Any tweets a user is interested in can be either added to a favourites list (be "favourited") for later retrieval or be "retweeted," meaning that the original Tweet is shared with followers of the retweeting user.

A system able to identify which tweets from a user's home timeline are likely to be favourited or retweeted by that user would therefore be useful in coping with this information overload and would allow Twitter users to narrow down the huge number of posts and focus on the most interesting for them. We can define this as a classification problem by considering the list of tweets appearing on a user's home timeline and considering all those which are favourited or retweeted as belonging to the positive class with all others being negative. Of course in order to make accurate classifications we need to develop a number of features of the tweets which will allow us to distinguish between those which are interesting and those which aren't. A further consideration is that the features which best describe favourited tweets may not be the same as those which best describe retweets, perhaps necessitating that separate models be developed.

6.1 Understanding what Makes a Tweet Interesting

In order to build prediction models we need to have some data on which to train and data which can be used to test the performance of the suggestions made. To do so we collected data from the Twitter API for a sample of 27 users over a period of 2 months. These users were recruited via a combination of direct mails and tweets advertising the study and asking users to take part. Users were simply directed to a page where the Twitter API asked them to confirm that they were willing to allow us access to their data. During this time we polled the API [3] once per hour and collected information about all tweets which appeared on each user's home timeline and in particularly whether the user subsequently favourited or retweeted each one, yielding tweets from around 7 thousand different authors. We also queried the API for information about each of these authors. Having access to each user's full Twitter data made it possible to ensure that the home timelines we were downloading were exactly the same as those really seen by each user.

To learn more about the users' interests, so that personal profiles could be constructed, we queried the API for all retweets and favourites made by each user. This data was then used to construct a pair of profiles based on the terms appearing in all of the user's favourited and retweeted posts. Profiles were constructed by considering all of the terms derived from the posts after they had been lower-cased and a list of Twitter-specific stop words had been removed. In addition, terms only appearing once in any given profile were removed and two bag-of-words models were created from the resulting profiles to represent each user.

With the collected data we computed a large number of different features (n=24) which could potentially hint at how interesting or useful a tweet might

[3] Twitter REST API version 1.1: https://dev.twitter.com/docs/api/1.1

Feature	**Description**	**RT**	**Fav**
RTSim	Similarity between post content and user's retweet profile	×	
FavSim	Similarity between post content and user's favourites profile		×
RTCount	Total number of times post has been retweeted by other users	×	−
MentionCount	Number of users mentioned in post (@username)	×	×
URLCount	Number of URLs included in the post		−
FavCount	Total number of times post has been favourited by other users	×	×
Mention	Whether of not the target user is mentioned in the post	×	×
FollowerCount	The follower count of the author	−	−

Table 4. Useful features for predicting retweets and favourites on Twitter.

be. These included features describing the post itself such as the bag-of-words of terms in the post, the number of photographs, user mentions or URLs embedded in the tweet and the number of times it had been retweeted or favourited by other users. To determine if the content of the tweet might be interesting to the user we calculated the Cosine similarity between the user's profiles and the bag-of-words representation of the tweet. As is standard in Information Retrieval, TD-IDF weighting was applied to the term weights. Features were also constructed describing the post's author including how many followers the author had and how many times they had tweeted since joining the service.

Using a simple logistic regression model we are able to identify which features contribute significantly to the prediction task. A description of the most statistically powerful features is given in Table 4 as well as which of the two models (retweets and favourites) the features pertain to. × indicates that the feature correlated positively with the target class, − indicates a negative correlation. Note that different combinations of features are useful for the two different classification model, hinting that while the problem may be similar, the features which encourage a retweet are somewhat different to those encouraging the user to favourite.

We trained our models on a sample of the data collection comprising a total of 20,454 tweets, only 133 of which were favourited and 356 retweeted. These numbers highlight one of the key difficulties with this task: the sheer imbalance of the classes. If we consider favourites prediction then the class imbalance is 153:1! Although the regression models mentioned earlier were able to identify useful features and could return fairly accurate class predictions, we do not use them as our final classifier due to their inability to cope well in the case of imbalanced data. Instead, to train the final classifier, we used random forests [6], a modern and highly competitive [13] tree-based classification algorithm. In this ensemble learning method a large number of decision trees are computed based on different random samples (subsets) of the training data. When classifying a new data point, all of the decision trees calculated during training vote on the correct output class and a final classification is made based on the mode of the classes output by the individual trees. Given a new tweet (represented by its set of features), this method returns the probability that it belongs to the positive class.

Since the classifier is optimised based on classification accuracy, we cannot simply assign tweets to the positive class if the predicted probability is greater than 0.5, as would normally be the case. To understand why, consider that it is possibly to design a trivial classifier for this task by simply returning the negative class regardless of the input. Such as classifier would achieve an "accuracy" of 99.3% when predicting favourites however it would be of little use as it would never return a single "interesting" tweet. To deal with this issue we also train a threshold value on the outputted probabilities, above which the tweet will be classified as positive. We selected this threshold by choosing the value which returned the best F1 score on the training data, which is an appropriate metric as it is the harmonic mean of precision and recall. This means that a high F1 score indicates that the classifier is able to return a good proportion of the positively-classed tweets (the true positives) whilst making few mistakes (i.e. returning false positives). For both prediction tasks the threshold was optimised to 0.15.

Despite the complexity of the classification problem at hand, our favourites prediction model is able to achieve an F1 score of 0.491 (precision: 0.455, recall: 0.534) and an accuracy of 0.993%, the retweet model achieves 0.394 (precision: 0.354, recall: 0.444) and 0.976%. Both models are able to simultaneously achieve very good F1 scores and overall prediction accuracy. In both cases the recall scores are better than the precision scores indicating that the models are retrieving a larger percentage of the positively-classed tweets at the cost of introducing more false positives.

It appears that the similarity between a post's content and user's profile is a good indicator of interestingness and it also seems that there is a difference between the terms of tweets users retweet and those they favourite (removing the user profile similarity from the favourites model reduces the F1 score significantly to 0.3). Interestingly, the number of times the post has been retweeted by other users is a positive feature for detecting retweets but negative when detecting favourites. Perhaps people feel encouraged to retweet a post which they have observed has been already retweeted by some of the people they follow. For both models the number of users mentioned in the post is a positive feature as is the number of times the post has been favourited. When the target user is mentioned in the post, the probability of it being retweeted or favourited increases dramatically, particularly for some users. This hints that ego plays some part in choosing tweets or that users want to ensure easy refinding of tweets which were specifically directed at them. The number of URLs in the post and the follower count of the author are both negative features, perhaps because posts with large numbers of URLs are often SPAM, advertisements or automatic posts from news sources.

7 Conclusions and Future Work

With the explosive growth of new data available on the web everyday from both traditional static web pages and newer social-generated sources, it is becoming increasingly difficult to filter out what is important and interesting, often leading to "information overload". In this book chapter we have considered two instances where users could be assisted in finding resources of interest to them from the vast array of potential candidate items available. In the first case we illustrated how a web search can be improved by means of personalising the results and in the second case we showed how items can be recommended to the user. We argue that these two scenarios are in fact two sides of the the same coin, the difference being that in the first case the user provides a query as input whereas in the second instance the user profile itself is the query.

We built models to learn about a person's interests and use of terminology based purely on their previous search queries which were then used to develop personalised search algorithms and were tested on real search log data. 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.

To recommend social media posts to users we collected a large sample of data from Twitter and attempted to predict which posts a user would retweet or favourite (both indicators of interest in the post). By defining a number of features describing the post itself, its similarity to the user's interests (as determined by previous posts they showed interest in) and the author we were able to build classification models with high levels of both precision and recall. In future work we could consider more complex features, for example features based on the Twitter social network (i.e. the network of links between users defined by friendships and follower relationships). With this information we could use graph theory techniques to gain a better understanding of how the social links contribute to interestingness. We also acknowledge that our method of constructing user profiles is somewhat rudimentary and could certainly be improved by either implementing some intelligent term selection routines or even using topic models.

In summary we have shown that there is much to be gained from considering contextual information about the user, whether from web logs or social media, when personalising search results and recommending items of interest. However, we have also demonstrated that this information must be incorporated in a subtle manner if it is to be of benefit. The directions indicated for future work have the potential to further increase the existing performance gains and lower further the information load imposed upon users.

References

1. G. Adomavicius and A. Tuzhilin Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions *IEEE Transactions on Knowledge and Data Engineering*, 17(6) : pages 734–749, 2005.
2. P. N. Bennett, F. Radlinski, R. W. White, and E. Yilmaz. Inferring and using location metadata to personalize web search. In *34th ACM Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 135–144. ACM, 2011.
3. 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 *35th ACM Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 185–194. ACM, 2012.
4. D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.
5. D. Boyd, S. Golder, and G. Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *43rd Hawaii Conference on System Sciences*, HICSS '10, pages 1–10. IEEE Computer Society, 2010.
6. L. Breiman. Random forests. *Machine Learning*, 45(1): pages 5–32, Oct. 2001.

7. S. Brin and L. Page The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30: pages 107-117, 1998.
8. B. Cambazoglu, F. Junqueira, V. Plachouras, S. Banachowski, B. Cui, S. Lim and B. Bridge A Refreshing Perspective of Search Engine Caching In 19th International Conference on World Wide Web (WWW), WWW '10, pages 181–190. ACM, 2010.
9. H. Cao, D. H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang. Context-aware query classification. In *32nd ACM Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 3–10. ACM, 2009.
10. M. J. Carman, M. Baillie, R. Gwadera, and F. Crestani. A statistical comparison of tag and query logs. In *32nd ACM Conference on Research and development in Information Retrieval*, SIGIR '09, pages 123–130. ACM, 2009.
11. M. J. Carman, F. Crestani, M. Harvey, and M. Baillie. Towards query log based personalization using topic models. In *19th ACM conference on Information and knowledge management*, CIKM '10, pages 1849–1852. ACM, 2010.
12. D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *29th ACM Conference on Research and development in information retrieval*, SIGIR '06, pages 390–397. ACM, 2006.
13. R. Caruana and A. Niculescu-Mizil An Empirical Comparison of Supervised Learning Algorithms In *23rd International Conference on Machine Learning*, ICML '06, pages 161–168. ACM, 2006.
14. P. A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschütter. Using odp metadata to personalize search. In *28th ACM Conference on Research and development in information retrieval*, SIGIR '05, pages 178–185. ACM, 2005.
15. K. Collins-Thompson, P. N. Bennett, R. W. White, S. de la Chica, and D. Sontag. Personalizing web search results by reading level. In *20th ACM Conference on Information and Knowledge Management*, CIKM '11, pages 403–412. ACM, 2011.
16. B. Croft, D. Metzler and T. Strohman. Search Engines: Information Retrieval in Practice *Addison-Wesley Publishing Company, USA*, 2009.
17. M. Daoud, L. Tamine-Lechani, M. Boughanem, and B. Chebaro. A session based personalized search using an ontological user profile. In *2009 ACM symposium on Applied Computing*, SAC '09, pages 1732–1736. ACM, 2009.
18. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41: pages 391–407, 1990.
19. Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *16th conference on World Wide Web*, WWW '07, pages 581–590. ACM, 2007.
20. D. Elsweiler and M. Harvey. Engaging and maintaing a sense of being informed: Understanding the tasks motivating twitter search. *Journal of the American Society for Information Science and Technology (JASIST)*, 2014.
21. D. Falush, M. Stephens, and J. K. Pritchard. Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies. *Genetics*, 164(4): pages 1567–1587, 2003.
22. L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005*, CVPR 2005, volume 2, pages 524–531. IEEE, 2005.
23. G. Furnas, T. Landauer, L. Gomez and S. Dumais The Vocabulary Problem in Human-System Communication *Communications of the ACM*, 30 (11): pages 964–971, 1987,
24. S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based user profiles for search and browsing. *Web Intelligence and Agent Systems*, 1(3-4): pages 219–234, 2003.
25. S. Golder and B. Huberman The structure of collaborative tagging systems *Journal of Information Science*, 32(2) : pages 198–208, 2005.
26. T. Griffiths and M. Steyvers. Finding scientific topics. *National Academy of Science*, 101: pages 5228–5235, 2004.
27. M. Harvey, M. Carman, and I. Ruthven. Improving social bookmark search using personalised latent variable language models. In *4th ACM Conference on Web Search and Data Mining*, WSDM '11, pages 485–494. ACM, 2011.

28. M. Harvey, M. J. Carman, and D. Elsweiler. Comparing tweets and tags for urls. In *34th European Conference on IR Research*, ECIR 2012, pages 73–84. Springer, 2012.
29. M. Harvey, I. Ruthven, F. Crestani, and M. Carman. Bayesian latent variable models for collaborative item rating prediction. In *20th ACM Conference on Information and Knowledge Management*, CIKM 2011, pages 699–708. ACM, 2011.
30. T. Hofmann. Probabilistic latent semantic indexing. In *22nd ACM Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57. ACM, 1999.
31. B. A. Huberman, D. M. Romero, and F. Wu. Social networks that matter: Twitter under the microscope. *First Monday*, 14(1), 2009.
32. J. Hurlock and M. L. Wilson. Searching twitter: Separating the tweet from the chaff. In *5th AAAI Conference on Weblogs and Social Media*, ICWSM 2011. AAAI, 2011.
33. B.J. Jansen and A. Spink. How are we searching the World Wide Web? A comparison of nine search engine transaction logs *Information Processing and Management (IPM)*, 42 : pages 248–263, 2006.
34. A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. 2007.
35. S. Lawrence Context in Web Search *IEEE Data Engineering Bulletin*, 23 : pages 25–32, 2000.
36. Z. Ma, G. Pant, and O. R. L. Sheng. Interest-based personalized search. *ACM Transactions on Information Systems*, 25(1), February 2007.
37. N. Matthijs and F. Radlinski. Personalizing web search using long term browsing history. In *4th ACM Conference on Web Search and Data Mining*, WSDM '11, pages 25–34. ACM, 2011.
38. P. McFedries. Technically speaking: All a-twitter. *Spectrum, IEEE*, 44(10): pages 84–84, 2007.
39. M. Melucci. A basis for information retrieval in context. *ACM Transactions of Information Systems*, 26(3):14:1–14:41, June 2008.
40. S. Nakamura, S. Konishi, A. Jatowt, H. Ohshima, H. Kondo, T. Tezuka, S. Oyama and K. Tanaka Trustworthiness analysis of web search results *Proceedings of the 11th European conference on Research and Advanced Technology for Digital Libraries (ECDL)*, pages 38–49. Springer, 2007.
41. K. Purcell. Search and email still top the list of most popular online activities *Pew Internet Center*, August 2011.
42. F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *15th conference on World Wide Web*, WWW '06, pages 727–736. ACM, 2006.
43. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl GroupLens: an open architecture for collaborative filtering of netnews *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW)*, pages 175–186, 1994.
44. S. E. Robertson. The Probability Ranking Principle in IR. *Readings in information retrieval*, pages 281–286. Morgan Kaufmann Publishers Inc., 1997.
45. A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *16th ACM conference on Conference on information and knowledge management*, CIKM 2007, pages 525–534. ACM, 2007.
46. I. Soboroff, D. McCullough, C. Macdonald, I. Ounis, and R. McCreadie. Evaluating real-time search over tweets. In *6th AAAI Conference on Weblogs and Social Media*, ICWSM 2012. AAAI, 2012.
47. J. Teevan, S. T. Dumais, and E. Horvitz. Potential for personalization. *ACM Transactions on Computer-Human Interaction*, 17(1):4:1–4:31, Apr. 2010.
48. J. Teevan, S. T. Dumais, and D. J. Liebling. To personalize or not to personalize: Modeling queries with variation in user intent. In *31st ACM Conference on Research and development in information retrieval*, SIGIR '08, pages 163–170. ACM, 2008.
49. X. Wei and W. Croft. Lda-based document models for ad-hoc retrieval. *29th ACM Conference on Research and development in information retrieval*, SIGIR '06, pages 178-185. ACM, 2006.
50. R. W. White, P. Bailey, and L. Chen. Predicting user interests from contextual information. In *32nd ACM Conference on Research and development in information retrieval*, SIGIR '09 pages 363–370. ACM, 2009.