# Tagging data as implicit feedback for learning-to-rank

Beate Navarro Bullock
Knowledge and Data Engineering Group (KDE)
University of Kassel, Germany
navarro@cs.uni-kassel.de

Robert Jäschke
Knowledge and Data Engineering Group (KDE)
University of Kassel, Germany
jaeschke@cs.uni-kassel.de

Andreas Hotho
Department for Artificial Intelligence and Applied Computer Science
University of Würzburg
hotho@informatik.uni-wuerzburg.de

## ABSTRACT

Learning-to-rank methods automatically generate ranking functions which can be used for ordering unknown resources according to their relevance for a specific search query. The training data to construct such a model consists of features describing a document-query-pair as well as relevance scores indicating how important the document is for the query. In general, these relevance scores are derived by asking experts to manually assess search results or by exploiting user search behaviour such as click data. The human evaluation of ranking results gives explicit relevance scores, but it is expensive to obtain. Clickdata can be logged from the user interaction with a search engine, but the feedback is noisy. In this paper, we want to explore a novel source of implicit feedback for web search: tagging data. Creating relevance feedback from tagging data leads to a further source of implicit relevance feedback which helps improve the reliability of automatically generated relevance scores and therefore the quality of learning-to-rank models.

## Keywords

learning-to-rank, implicit-feedback, folksonomy

## 1. INTRODUCTION

When people find interesting web pages in the internet, they can store links to those pages in social bookmarking systems and add tags which help to describe the specific web page. If not explicitly denoted, the link, title and tags of the bookmark are public and can be searched and copied by other users. The collaborative storage and annotation of information in social bookmarking systems leads to the creation of a social web index in which documents are filtered by users. This manually created index is an alternative to classical indices created by web crawlers following the internet's hyperlink structure.

The process of storing and annotating web links can be seen as an expression of relevance: by tagging a specific URL, a user judges this resource to be of importance. The resource's tags indicate what it is relevant for. Mostly, tags describe a topic, the resource's context or the user's intention for tagging the resource. For instance, the tag *learning-to-rank* describes the web link http://en.wikipedia.org/wiki/Learning_to_rank, while the tag *lecture* provides the context and the tag *toread* the intention what to do with the specific resource. The popular algorithm PageRank [6] is based on a similar assumption by promoting a web page when it has been linked by many other (important) web pages. In contrast to tagging data, however, the topic or the reason why this page is important can not be revealed as no tags but only titles or text information close to the link are available.

Not only the PageRank algorithm is based on implicit feedback information. Many other search algorithms also need some kind of relevance feedback. A prominent example are learning-to-rank methods [1] where a model for ranking objects is constructed from training data and applied to infer rankings on unknown data. Training data consists of relevance scores assigned to query-document pairs. The scores for the training data are either derived by exploiting user search behaviour such as click data (for example in [4, 5, 2]) or by asking experts to manually assess search results. The human evaluation of ranking results gives explicit relevance scores, but it is expensive to obtain. Clickdata can be logged from the user interaction with a search engine, but the feedback is noisy as a click does not always indicate relevance. Sometimes, people are not satisfied with the clicked result, reformulate their information need or click on another resource.

In contrast, tagging data seems to be more explicit as the process of tagging involves finding a relevant resource and then storing and categorizing it in the bookmarking system. While search queries express a specific information need and there is no evidence as to whether a clicked URL fulfills this need or not, tags serve as a description or categorization for the specific resource. It would therefore be of tremendous value for generating training- and test data for learning-to-rank, if one could use tagging data as a further source of implicit feedback.

In order to explore the practicability of using tagging data as a source of implicit feedback, we compare implicit feedback generated from tagging data to implicit feedback generated from clickdata. Given a search query and a ranking of a search engine, we match the query and URLs with tags and resources of a social bookmarking system. We thereby assume that a URL of a ranking list is important if it has been tagged with the query terms (or similar tags). At the same time, we assume that the URL is relevant if it has been clicked after the submission of the specific query.

In Section 2 different strategies for modeling implicit feedback are presented. To compare the feedback type's performance, ranking models are learned using training data, where the relevance scores are generated from a specific feedback type (for example from tagging data). The models are tested by predicting relevance scores generated from other feedback types (for example click data). The experimental results, described in Section 3, show that ranking models generated from both tagging and click data are comparable.

## 2. IMPLICIT FEEDBACK FOR LEARNING-TO-RANK

This section briefly explains learning-to-rank methods, the matching of click- and folksonomy data to rankings and the inference of preference scores for learning rankings.

### 2.1 Learning-to-rank

Learning-to-rank aims at constructing models from preference data which allows for ordering unknown entities. The goal is to learn a ranking function to present search results. The training data consists of queries and documents matching the query ordered according to their relevance to the query. A query-document pair is usually represented by feature vectors with features such as the frequency of the query in the document's summary or the length of the document.

Different approaches exist for solving the learning-to-rank task: pointwise, pairwise and listwise approaches [1]. In this paper, we focus on the pairwise approach. A binary classifier is learned which classifies one of two documents to be relatively more relevant.

### 2.2 Mapping click and tagging data to rankings

Given a search query and a ranking of a search engine, one can match the query and ranked URLs with tags and resources of the social bookmarking system. Different possibilities exist to derive such a match: a link in the social bookmarking system can be seen as an indicator for relevance if it contains one of the query terms as tag, all query terms as tags or tags that are similar to the query terms. The experiments of this paper consider all social bookmarking links as relevant which contain all of the tags. Table 1 depicts an example for such a matching. The query submitted to the search engine is "social web". The first column shows which URLs in the ranking were clicked by MSN users. The second table shows which URLs in the ranking also appear in the folksonomy with "social web" as tags.

### 2.3 Preference Strategies

By mapping click- or tagging data to query rankings, we know which URLs of a ranking have been clicked or tagged. One can then extract preference pairs from the different lists $l_q$ in the form of tuples $\{(q, r_i, r_j) \mid r_i \succ r_j\}$, which means that document $r_i$ is more relevant than document $r_j$ for the query $q$. Different strategies can be defined to extract the relevance pairs considering the order of resources, the number of times a resources was clicked or the co-occurrence of resources. In the following, strategy names with the suffix *tag* refer to a strategy using tagging data, without the suffix to one using clickdata.

*Binary relevance (binary, binarytag):* A preference pair $(q, r_i, r_j)$ is extracted if an arbitrary user clicked or tagged a resource $r_i$ while no user clicked or tagged the resource $r_j$. The click pairs of Table 1 would be $\{(r_1, r_2), (r_1, r_4), (r_3, r_2), (r_3, r_4)\}$.

*Heuristic rules:* In Joachims et al. [4] different heuristic rules were proposed to infer preference statements from clickdata.

- *Skip above pair extraction (sa, safull, satag, satagfull):* For the ordered ranking list and a URL at position $i$, which was either tagged or clicked, all unclicked/untagged results ranked above $i$ are predicted to be less relevant than the result at position $i$. URLs after $i$ are ignored (*sa*, *satag*) or also considered as non-relevant (*safull*, *satagfull*). Example pairs of Table 1 for *satag*: $\{(r_4, r_2), (r_4, r_3)\}$, for *satagfull*: $\{(r_4, r_2), (r_4, r_3), (r_1, r_2), (r_1, r_3)\}$

**Table 1: Example of a mapping for an MSN search query, user clicks and resources in Delicious**

| query: *social web* | | |
|---|---|---|
| MSN click | Delicious resource | MSN ranking |
| x | x | http://www.socialweb.net/ |
| | | http://de.wikipedia.org/wiki/Web_2.0 |
| x | | http://www.socialweb.siteblob.com |
| | x | http://www.extensions.joomla.org/extensions/social-web |

- *Skip above reverse pair extraction (sar, sarfull, sartag, sartagfull):* Search engine users normally scan a result list from the top to the bottom. After clicking on one document, their information need is either satisfied or they get back to the list and continue scanning. Resources clicked at later positions have therefore been seen after the clicks at earlier positions. They can be considered as more likely to be relevant than the clicked documents before. Example pairs for *sar* are: $\{(r_3, r_1)\}$, *sarfull*: $\{(r_3, r_2), (r_3, r_1)\}$

*Popularity information (popularity, poptag):* When aggregating clickthrough or tagging data over different users, one can get information about the popularity of the resource. The more often a resource was tagged, the more relevant it might be for the tags/queries in question. Hence, we can extract a preference pair if $r_i$ was more often clicked or tagged than $r_j$ counting clicks over all sessions of the query log and all posts of the folksonomy. This strategy does not consider multiple clicks for the same query within one session as they are often caused by spammers. In folksonomies, users can tag resources only once.

*FolkRank (folkrank):* In [3], the well-known PageRank algorithm was adapted to folksonomies. The key idea of the FolkRank algorithm is that a resource which is tagged with important tags by important users becomes important itself. The same holds, symmetrically, for tags and users. One can thus create a graph of resource, user and tag vertices which are mutually reinforcing each other by spreading their weights. Similar to the PageRank algorithm, preference scores can be assigned to tags in the random surfer vector. FolkRank then orders resources according to their relevance regarding those tags. Tags can again be matched with the query terms. Resources which appear in the folksonomy ranking as well as the search engine's ranking list are preferred to those which were not ranked highly by the FolkRank algorithm.

Based on the described strategies, pairwise preferences can be extracted and ordered into (partial) ranking lists. For example, if three URLs of a ranked list have been clicked, those URLs would be ordered according to their popularity by means of the *popularity* strategy, all other URLs in the ranked list would be set to non-relevant (e. g., receive a rank score of 0).

## 3. EXPERIMENTS

This section describes the experiments' datasets and the general, experimental setting.

### 3.1 Datasets

For our experiments, we combine three different kinds of data sources: Ranking data, click data and social bookmarking data.

**MSN ranking data** We obtained a ranking dataset from Microsoft collected in May 2006.[1] The dataset consists of about 1,6 million rankings having up to 50 ranked URLs each.

[1]http://research.microsoft.com/ur/us/fundingopps/RFPs/Search_2006_RFP.aspx

**MSN click data** We also obtained a click data set from Microsoft for the period of May 2006. [1] The dataset consists of about 15 million queries which were tracked from the MSN search engine users in the United States in May 2006. For about 700,000 queries of the MSN click data set we have a set of ranked URLs from the MSN ranking data containing more than one URL and with at least one URL clicked in the MSN click data set.

**Delicious data**: A dataset of the social bookmarking system Delicious [7] is used for infering implicit feedback from tagging data. The dataset contains the public bookmarks of about 950,000 users retrieved from between Dec. 2007 and Apr. 2008. To be comparable to the ranking and click datasets, we only consider posts before end of May 2006. Tags are normalized by splitting all queries into single, lower case terms and all characters expect of letters and numbers are removed. Further, only those posts are filtered, which match a query-doc pair in the click data set. We therefore normalize the queries in the same manner as done with the tags and filter the posts which have the same URL and contain at least all query terms as tags. Overall, we get 36,830 queries with rankings where at least one URL in the ranking has been tagged in Delicious with the corresponding query terms.

## 3.2 Setting

Since no ground truth of the rankings exists, we compare the performance of different strategies against each other. The basic idea is to learn a ranking model using training examples with preference scores generated from one of the strategies (see Section 2.3) and predict new rankings with this model. The predicted rankings can then be compared to preference scores derived from other strategies. If click and tagging data based strategies generate similar results, they can be considered both as valuable sources for implicit feedback. We construct training examples using the information (document title, summary, URL) given by the MSN ranking dataset. Features similar to those proposed in the LeToR 4.0 benchmark dataset [2] are computed to represent a query-document pair. The features include term frequency, the length, tf-idf values, BM25 and different language models of the document fields (body, anchor, title, URL or entire document). As the MSN dataset offers only summary snippets as descriptions of a website, we use those snippets as body text and skip all features based on anchor text. Overall, each query-document pair consists of 46 features. The relevance scores of each query-document pair are inferred from one of the strategies proposed in 2.3. For example, the *binarytag* strategy would set all query-document pairs as relevant which have been tagged with the appropriate query terms.

In [2] the click entropy was proposed to classify queries. Queries with lower click entropies are navigational queries (for example "yahoo" or "facebook"). The result lists of those queries are normally less diverse and easier to predict. The measure itself is defined as $ClickEntropy(q) = \sum_{d \in D(q)} -P(d|q) log_2 P(d|q)$ where $Click-Entropy(q)$ is the click entropy of query $q$. In our settings, we have either clicks or posts including the tagged resources and the query as their tags. $D(q)$ is then the collection of documents clicked or tagged for query $q$. $P(d|q)$ is the percentage of clicks on document $d$ among all clicks on $q$ or the percentage of posts with document $d$ among all posts considering $q$ as tags. In our experiments, we consider queries with a click or tag entropy lower than 0.5. Further, query rankings with less than five clicks or five posts are filtered. Please note that the resulting sizes of the training data are different depending on the strategy and the underlying data (click or tagging data).

We use ranking SVM [4] to learn the different models.

[2]http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR4.0

# 4. RESULTS

We first compare the similarity of ranking lists derived from the different strategies in Section 2.3 by means of a correlation analysis. Second, the performance of models derived from different preference scores are compared in Sectopm 4.2.

## 4.1 Correlations

The similarity of ranking lists derived from the different strategies can be compared by analyzing their correlations. As correlation measure, Kendall tau-b ($\tau_b$) is used, which measures the degree of correlation between rankings by considering the number of concordant and discordant pairs. In contrast to Kendall tau, the measure additionally considers ties [2]. A Kendall tau-b of 1 yields a perfect correlation while a correlation of -1 reveals an inverse ranking. In order to obtain significant results, we filter rankings having less than 10 URLs in common.

Table 2 shows the correlations of each ranking list with all other ranking lists. Each ranking strategy is perfectly correlated with itself. Reverse strategies such as *sa* and *sar* are perfectly anti-correlated. The strategies *sarfull* and *safull* correlate strongly as the full rankings, i. e., also non-clicked or -tagged URLs are considered. The feedback generated from *satas* correlates strongly with *rank* and *sa* as no reordering takes place. The correlation of the *folkrank* and *poptag* strategies with feedback generated from clickdata (for example, *sa*, *sar*) is mostly positive, but low. The highest correlation yields the *popularity* strategy (0.20 / 0.279). Similiar feedback ranking lists seem to be generated from the *folkrank* and *poptag* strategies (0.88). Overall, one can find a positive correlation between feedback lists generated from click and tagging data. However, the correlation is not as high as feedback generated from strategies of the same feedback data type.

## 4.2 Error

As performance measure we consider the training error, defined by the number of missclassified pairs divided by the number of total pairs.

Table 3 depicts the errors made of models derived from training data using different strategies (rows) and tested on test sets of a specific strategy and a corresponding random test set. The random test sets contain the same number of preference pairs as a test set of a specific strategy, but preferences are uniformly sampled from all possible preference pairs. The error of models tested on those random test examples is close to 0.5. The error of models tested on examples with preference scores derived from the corresponding strategy is smaller. For example, the model generated from the *folkrank* strategy (3rd row) has an error of 0.26 when tested on test preference pairs generated from the *binary* strategy, while the random test set (*rand_binary*) reveals an error of 0.5. The error difference between non-random and random models demonstrates that non-random models approximately predict document orders according to the relevance of the documents. Please note that not all possible test sets are shown. However, results using the other test sets were similar.

Table 4 shows the prediction errors obtained on all different test sets (without random counterparts). Again, the rows indicate the strategy used for generating preference scores for the training examples, the columns represent the strategies for the preference scores used for testing. The best performing models for each column are marked in bold.

Though the model learned from a strategy often performs best when tested against the rankings inferred from the specific strategy (for example *binary* or *binarytag*), this is not always the case (for ex-

**Table 2: Correlation of preference scores derived from the different strategies**

| | rank | safull | sa | sar-full | sar | popularity | safull-tag | satas | sarfull-tag | sar-tag | poptag | folk-rank |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rank | 1.0 | 0.984 | 1.0 | 0.982 | -1.0 | 0.263 | 0.971 | 1.0 | 0.965 | -1.0 | 0.150 | 0.130 |
| safull | | 1.0 | 1.0 | 0.998 | -1.0 | 0.263 | 0.952 | 0.886 | 0.947 | -0.886 | 0.182 | 0.140 |
| sa | | | 1.0 | -1.0 | -1.0 | 0.263 | 0.803 | 1.0 | 0.570 | -1.0 | 0.226 | 0.158 |
| sarfull | | | | 1.0 | 1.0 | -0.263 | 0.949 | 0.794 | 0.944 | -0.794 | 0.171 | 0.136 |
| sar | | | | | 1.0 | -0.263 | -0.803 | -1.0 | -0.570 | 1.0 | -0.226 | -0.158 |
| popularity | | | | | | 1.0 | 0.285 | 0.483 | 0.191 | -0.483 | 0.279 | 0.203 |
| safulltag | | | | | | | 1.0 | 1.0 | 0.994 | -1.0 | 0.150 | 0.350 |
| satag | | | | | | | | 1.0 | -1.0 | -1.0 | 0.150 | 0.134 |
| sarfulltas | | | | | | | | | 1.0 | 1.0 | -0.150 | 0.301 |
| sartag | | | | | | | | | | 1.0 | -0.150 | -0.138 |
| popularitytag | | | | | | | | | | | 1.0 | 0.880 |
| folkrank | | | | | | | | | | | | 1.0 |

**Table 4: Prediction errors of different strategies. The lowest errors for each test dataset are marked in bold.**

| | binary | binarytag | folkrank | popularity | poptag | rank | sa | safull | safulltag |
|---|---|---|---|---|---|---|---|---|---|
| binary | **0.26** | 0.37 | 0.33 | **0.27** | 0.37 | 0.43 | **0.27** | 0.42 | 0.45 |
| binarytag | 0.31 | **0.33** | **0.31** | 0.31 | **0.33** | 0.45 | 0.32 | 0.44 | 0.45 |
| folkrank | **0.26** | **0.33** | 0.34 | **0.27** | 0.34 | **0.42** | **0.27** | **0.41** | **0.43** |
| popularity | **0.26** | 0.37 | 0.33 | **0.27** | 0.37 | 0.43 | **0.27** | 0.42 | 0.45 |
| popularitytag | 0.31 | **0.33** | **0.31** | 0.31 | 0.34 | 0.45 | 0.31 | 0.44 | 0.45 |
| rank | 0.29 | 0.41 | 0.36 | 0.29 | 0.41 | **0.42** | 0.29 | **0.41** | 0.45 |
| sa | **0.26** | 0.37 | 0.33 | **0.27** | 0.38 | 0.43 | **0.27** | 0.42 | 0.45 |
| safull | 0.28 | 0.4 | 0.35 | 0.28 | 0.4 | **0.42** | 0.28 | **0.41** | 0.45 |
| safulltag | 0.3 | 0.36 | 0.32 | 0.3 | 0.36 | 0.43 | 0.3 | 0.42 | **0.43** |

**Table 3: Prediction errors on test datasets and their random counterparts**

| training / test | binary | rand_ binary | binarytag | rand_ binarytag | folkrank | rand_ folkrank |
|---|---|---|---|---|---|---|
| binary | **0.26** | 0.48 | 0.37 | 0.49 | 0.33 | 0.49 |
| binarytag | 0.31 | 0.51 | **0.33** | 0.50 | **0.31** | 0.50 |
| folkrank | **0.26** | 0.50 | **0.33** | 0.51 | 0.34 | 0.51 |
| popularity | **0.26** | 0.50 | 0.37 | 0.50 | 0.33 | 0.51 |
| popularitytag | 0.31 | 0.48 | **0.33** | 0.50 | **0.31** | 0.49 |
| rank | 0.29 | 0.51 | 0.41 | 0.50 | 0.36 | 0.51 |
| sa | **0.26** | 0.49 | 0.37 | 0.49 | 0.33 | 0.49 |
| safull | 0.28 | 0.51 | 0.40 | 0.51 | 0.35 | 0.52 |
| safulltag | 0.30 | 0.48 | 0.36 | 0.49 | 0.32 | 0.49 |

ample *poptag*). The *rank* strategy, derived from the original MSN ranking, performs worse than the other strategies in the majority of cases. Comparing strategies derived from the clickdata and those derived from the tagging dataset, one can find that clickdata-derived strategies perform better on the clickdata and tagging-data-derived strategies on tagging data. Only models derived from the *folkrank* strategy perform well on both kinds of data. As the *folkrank* strategy does not match query terms to tags, but computes a ranking using the entire folksonomy, the results can be seen as an indicator to test more elaborate matching approaches than matching folksonomy resources to rankings only when the query terms appear as tags. This can be seen as an indicator that both click and tagging feedback help improve rankings.

# 5. CONCLUSION

In this work, we presented a first comparison of implicit feedback strategies for a learning-to-rank scenario. Analogously to previous works proposing strategies for extracting preference scores from clickdata, we proposed different methods to infer feedback from tagging systems. By learning models with training examples from one of the strategies and predicting the outcome of other strategies, we could analyze similarities and differences between click- and tagging data. While the *folkrank* strategy predicts feedback from both types of data reasonably well, the other strategies perform better when predicting feedback of examples generated from their corresponding dataset.

In future work we would like to develop more sophisticated strategies by considering the time of a post, the activity and specific interests of users and by matching not only posts containing the same tags as query terms, but also similiar tags. As our evaluation method only compares strategies, but can not show a preference for one of them, we need to create a ground truth dataset by manually labeling examples. Furthermore, transfer learning methods can be an interesting future research direction, as the transfer of a model learned on a specific dataset to another dataset allows the evaluation of the strategies on existing datasets such as the LeToR datasets.

# 6. REFERENCES


[1] X. Dong, X. Chen, Y. Guan, Z. Yu, and S. Li. An overview of learning to rank for information retrieval. In M. Burgin, M. H. Chowdhury, C. H. Ham, S. A. Ludwig, W. Su, and S. Yenduri, editors, *CSIE (3)*, pages 600–606. IEEE Computer Society, 2009.

[2] Z. Dou, R. Song, X. Yuan, and J.-R. Wen. Are click-through data adequate for learning web search rankings? In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 73–82, New York, NY, USA, 2008. ACM.

[3] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In Y. Sure and J. Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, pages 411–426, Heidelberg, June 2006. Springer.

[4] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, 2002.

[5] C. Macdonald and I. Ounis. Usefulness of quality click-through data for training. In *WSCD '09: Proceedings of the 2009 workshop on Web Search Click Data*, pages 75–79, New York, NY, USA, 2009. ACM.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, Nov. 1999.

[7] R. Wetzker, C. Zimmermann, and C. Bauckhage. Analyzing social bookmarking systems: A del.icio.us cookbook. In *Mining Social Data (MSoDa) Workshop Proceedings*, pages 26–30. ECAI 2008, July 2008.