///

Information Retrieval and Machine Learning

Massimo Melucci

University of Padua
Department of Information Engineering
massimo.melucci@unipd.it

CIMI School in Machine Learning 2015

///

///

Information Retrieval and Machine Learning

Massimo Melucci

University of Padua
Department of Information Engineering
massimo.melucci@unipd.it

CIMI School in Machine Learning 2015

Summary

From traditional IR to IR based on ML

- Traditional IR is based on query-document similarity, content, weighting schemes (e.g. TFIDF, BM25) and RF.
- Context (e.g. user task, query intent, document quality) and documents (e.g. structure, language, network) are becoming complex.
- Search is complex due to user interaction, collaborative filtering, contextual search, multimedia search, vertical search.
- Since early 2000 evidence sources beyond document text are available:
  - Hypertext properties (e.g. PageRank), anchor texts, URLs.
  - Language properties: named entity extraction (e.g. what is "bank").
  - User behavior data, e.g. single clicks, complex browsing patterns, dwell time, eye tracks, previous queries.
- See for example [23], [27].

Basic problem and idea

- Basic problem:
  - Designing one single model for all tasks (top-down approach) is hard.
  - Combining many different models is also hard.
- Basic idea:
  - To leverage large amounts of data.
  - To learn a model from the data (bottom-up approach).

IR supplies abundant things to ML (and viceversa)

- Classification (e.g. document retrieval, query intent, user categorization, spam detection, entity recognition).
- Unsupervised learning: search engine retrieval page clustering, frequent term set mining.
- Semi-supervised learning: how to deal with unlabeled documents since usually only relevant documents are known.
- Active and online learning: recommender systems, online advertising.

ML in IR *ante litteram*

- Experiments in the probabilistic retrieval of full text documents [4].
- Incorporating models and combining evidence [9, 10].
- Optimum polynomial retrieval functions based on the probability ranking principle [13].
- Relevance feedback in information retrieval [37].

Summary

Essential concepts

- To each query-*documents* pair, d features are associated and one feature vector $\mathrm{x} \in \mathbb{R}^d$ is computed.
- Moreover, each query-documents pair is associated to a pattern y.
- A training data set is thus represented by a set $\{(\mathrm{x}^{(1)}, y^{(1)}), \ldots, (\mathrm{x}^{(m)}, y^{(m)})\}$.
- However, y is implemented depending on the approach to learning to rank (i.e. points, pairs, lists; see later).
- Basically, x comes from the system while y comes from the user (i.e. satisfaction, relevance, etc.). However, there are some issues as regards y.

Issues of learning using explicit relevance assessments

- Judgments of which documents are approved and disapproved gives a rich set of additional information for the search engine to use.
- In practice, users are hardly likely to give explicit feedback about every document in the search results; however, information can be gleaned from the user's subsequent behavior:
  - Which documents does she click on.
  - How long does she dwell on each one.
  - Are her needs satisfied or does she return to search again?

Issues of learning using user interaction data

- It seems that search engines cannot understand the user's behavior.
- However, they could intercept the clicks made on the search results page because clicking on "Massimo Melucci's homepage" is actually clicking on `http://www.google.it/url?sa=t&url=http%3A%2F%2Fwww.dei.unipd.it%2F~melo`.
- It does seem difficult to determine information about your subsequent behavior: how long you spend with each document, or what you do next.
- But, are you using the search engine's browser? Have you downloaded any toolbar or browser add-on?
- If the outcome improves the results of your searches, you might well be prepared to share this information with the search engines; queries are already shared.

Types of feature

- Content supplied by queries and documents.
- Links supplied by other authors (in-links): also anchors (useful for indexing non-text).
- Feedback supplied by users (readers, not writers): (dis)likes, click-through, behaviour, etc.
- See for example [30].

Content-based features

- Document content supplied by authors of the document: keywords, part-of-speech tags, logical structure, etc.
- Query term frequency: frequency of each query word in each document field (e.g. title, anchor, URL, body).
- BM25: scores for each each query word or n-gram in each document field [48].
- Proximity: distance measures between query words in documents [11], [17].

Link-based features

- Number of in-links.
- PageRank.
- Hyper-linked Induced Topic Search (HITS).
- See for example `http://research.microsoft.com/~LETOR/`.

User behaviour-based features

- BrowseRank.
- Number of clicks.
- Dwell or display time.
- (Dis)likes, radio button clicks, forward to users.
- See for example [18], [28], [29], [33], [38], [39], [51].

Summary

Main approaches

- Point-wise: to predict relevance degree (qrel) of an object
- Pair-wise: to predict preference between two objects
- List-wise: to predict order of a list of objects

Point-wise approach

doc
1
2
3
4
5

$\rightarrow$

doc	qrel
1	1
2	0
3	2
4	0
5	1

From points to pairs

- Two problems with points explain why the pair-wise approach may be preferred to the point-wise approach.
- Arbitrariness of relevance degree: for example, each point may be assigned a relevance judgment between 1 and 5 in one ranking, a degree between 0 and 1 in another ranking, or a degree between 1 and 5 by another user but a different meaning.
- Number of relevance assessments: 10 assessments $\times$ 17000 queries is not small but negligible if compared with the number of documents and queries.
- Pairing up examples has the twin effect of greatly multiplying the number of different inputs, and eliminating the arbitrariness of the rating scale (for each query, five assessments produce ten pair assessments).

Pair-wise approach

$$\begin{array}{c} \text{doc} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \quad \rightarrow \quad \begin{array}{cccccc} & \multicolumn{5}{c}{\text{preferences}} \\ & 1 & 2 & 3 & 4 & 5 \\ 1 & & > & < & > & = \\ 2 & & & < & = & < \\ 3 & & & & > & > \\ 4 & & & & & < \end{array}$$

Pair-wise approach and implicit feedback

- The problem is to convert clicks into pairs.
- One method is to use the differences between numbers of clicks on documents to derive preferences (relative relevance) on document pairs.
- Given a query, if the number of clicks on one document is higher than the number of clicks on the other document, the former is preferred to the latter.
- However, *click bias* may occur: given a ranking list of documents, users tend to click documents on the top, even the documents may not be relevant. As a result, documents on the top tend to have more clicks.
- See for example [20, 34, 35].

From pairs to lists

- Pairs only give a partial order of the documents because some preferences might not be available:
  - for example, only $1 > 2$ and $5 > 3$ might be known out of ten documents.
- When documents are preferred with respect to different queries, the ordered pairs may be inconsistent:
  - for example, $1 > 2$ w.r.t. one query and $1 < 2$ w.r.t. the other query.
- The list-wise approach consists of transforming points or pairs into classes of equivalent totally ordered lists:
  - note that it is infeasible that a user provides totally ordered lists.
- See for example [2].

List-wise approach

doc		ranked lists			
1		3	3	3	3
2		1	5	1	5
3	$\rightarrow$	5	1	5	1
4		3	3	4	4
5		4	4	3	3

Approaches and input features

- Point-wise approach:
  - features are document-query points (note it is a point not a pair, i.e. the document is the point)
  - given a query, to predict the relevance degree of each document of a collection
- Pair-wise approach:
  - features are document pair together with a query (documents are paired)
  - given a query, to predict the order of each document pair
- List-wise approach:
  - features are document lists with a query (documents are arranged in ranked lists)
  - given a query, to predict the ranking, i.e. the rank or the relevance degree of each document in a ranked list

Approaches and learning functions

- Regression
- Classification
- Ordinal regression

Regression

- Output is a scalar
  - Point-wise: rank, relevance degree or relevance score (e.g. BM25).
  - Pair-wise: order (e.g. $+1 = i > j, 0 = i = j, -1 = i < j$).
- A function of the input (feature vector) predicts the output scalar (e.g. the releance degree of the document to the query).
- The loss function is the squared difference between the predicted output and true output.
- The risk is the sum of loss function values over training sets.
- See for example [6], [13].

Classification

- Output is a class label
  - Point-wise: rank or relevance degree.
  - Pair-wise: order (e.g. $+1 = i > j, 0 = i = j, -1 = i < j$).
- Learning function predicts the class of the document with respect to the query.
- See for example [5], [7], [14], [25], [31], [36], [41].

Ordinal regression

- It is like classification where classes can be ordered (a.k.a. ordinal classification) or like regression where the dependent variable is discrete (and a metric can be applied)
- See for example [8], [16], [44].

Summary

ML and IR

- Key Concepts
- Features and patterns
- Approaches
- Applications
- Evaluation

IR related tasks

- Ad-hoc retrieval, Relevance Feedback (RF) and search in the World Wide Web (WWW)
- Collaborative filtering
- Definition search
- Keyphrase extraction
- Multimedia retrieval
- Text summarization
- Online advertisement

Ad-hoc retrieval, RF and search in the WWW

- Create a training set with many examples of documents that contain the terms in a query, along with human judgments about how relevant they are.
- The learning algorithm analyzes this training data and comes up with a way to predict the relevance judgment for any document and query – this is used to rank queries in the future.
- This is RF.
- The modern application is WWW search engine.
- A modern search engine exploits dozens of features (see Learning to Rank (LETOR)).

Collaborative filtering

- Items (e.g. products, movies, etc.) have to be ranked and/or suggested.
- Input features are metadata (e.g. price, country, genre, etc.).
- Output data are preferences, labels, judgments, etc.
- Training data are collected by past usage (purchases, rentals, etc.).
- See for example [1], [42], [52].

Definition search

- Definitions (e.g. dictionary-like entries, answers to questions, etc.) have to be ranked and/or suggested.
- Input features are keywords, grammatical structure, function words, word positions, part-of-speech tags, etc.
- Output data are definitions.
- Training data are collected by human expert supported by Natural Language Processing (NLP) tools.
- A variant is keyphrase or index term extraction.
- See for example [45], [43].

Multimedia retrieval

- Non-fully textual documents (e.g. images, video, sound, music) have to be ranked and/or suggested.
- Input features are multimedia or multimodal low-level data or by text around these media.
- Output data are relevance degrees.
- Training data are collected by user behaviour mainly.
- See for example [3], [15], [24], [40], [50], [49].

Text summarization

- Document sentences or terms have to be ranked and/or suggested perhaps after some NLP driven concatenation.
- Input features are keyword weights.
- Output data are goodness scores or labels.
- Training data are collected by expert humans.
- See for example [19], [32]

Online advertisement

- Ads have to be ranked and/or suggested.
- Input features are keyword weights and other contextual features.
- Output data are binary decisions or ranks.
- Training data are mainly collected by user behaviour.
- See for example [22], [26], [53].

Summary

Training and testing

- Training data set: queries, documents, relevance assessments
  - Queries are sampled from the relevance assessors or from logs.
  - Documents are crawled from the WWW, databases, collections, etc.
  - Relevance assessments are collected from humans, explicitly or implicity
  - Features are computed from queries *and* documents

Training and testing

- Training data set: queries, documents, relevance assessments
  - Queries are sampled from the relevance assessors or from logs.
  - Documents are crawled from the WWW, databases, collections, etc.
  - Relevance assessments are collected from humans, explicitly or implicity
  - Features are computed from queries *and* documents
- Testing data set: new queries and documents
  - New queries are read from the input stream.
  - New documents are retrieved.
  - Features are computed from the new queries and documents.
  - The documents are ranked according to the ranking function learned from the training data.

Training data set creation

- Two approaches to create a training data set.
- Explicit feedback: relevance assessments are explicitly obtained by assessors
- Implicit feedback: relevance assessments are implicitly inferred by users' behaviour

Training data set creation using explicit feedback

- In an initial experiment a training set of queries is assembled, with almost 1000 documents for each one:
  - for example, MSN assembled 17,000 queries;
  - TREC utilises the pooling method.
- For each query, the k top-ranked documents may be retrieved (e.g. using BM25 or TFIDF).
- Given a query, dozens or hundrends features are calculated for each query / retrieved document pair.
  - The majority of queries and documents can be used for training and the remainder were set aside for independent testing.
- Human evaluators judge the relevance of the top k documents on a scale of, say, 0 (irrelevant) to 5 (excellent):
  - For example, if the query is "machine learning", and the document is Bishop's book, then the pattern is "excellent", the Wikipedia page about ML is "fair", a page only mentioning ML will be labeled as "poor".
- Patterns representing relevance are then assigned to the query document pairs.

Training data set creation using explicit feedback

- Since human labeling is expensive, it is often the case that some query-document pairs are only judged by one single judge; however, it may be biased.
- To reduce bias, the labeling on query-document pairs can be performed by multiple judges, and then majority voting can be conducted; however, it may be expensive.

Training data set creation using implicit feedback

- The other way of generating relevance assessments is implicitly from click through data.
- Click-through data at a web search engine records clicks on documents by users after they submit queries.
- Click-through data represents implicit feedbacks on relevance from users and thus is useful for relevance judgments.
- See for example [12], [21], [47], [46].

Explicit vs implicit feedback in training data set creation

- In general, it is very hard to maintain the quality of data, when it is created by humans – however relevance cannot be assessed by machines.
- Both implicit feedback and explicit feedback have pros and cons.
- Explicit feedback:
  - Human judges are prone to errors.
  - Their understanding on relevance also has limitations because they are not query owners.
  - Furthermore, manual data labeling is also costly.
- Implicit feedback:
  - Derivation of training data from click-through data is cheap.
  - The data might also represent real users' judgments.
  - However, click-through data is noisy.
  - It is only available for head queries (high frequency queries).
  - It is subject to click bias.

Learning to Rank (LETOR)

- `http://research.microsoft.com/letor`
- It helps design, implement and evaluate learning-to-rank algorithms.
- It consists of document corpora, query sets and relevance assessments
- Documents are also provided in samples.
- Features are available
- Data are folded.

Data

- Document set: GOV2 (crawl of .gov in 2004, 25M documents).
- Query set: Text REtrieval Conference (TREC) million query track:
  - 1,700 queries from TREC 2007,
  - 800 queries from TREC 2008.
- Relevance assessments: three degrees.

Features[1]

- $\sum_{t_i \in q \cap d} TF(t_i, d)$ in body
- $\sum_{t_i \in q \cap d} TF(t_i, d)$ in anchor
- $\sum_{t_i \in q \cap d} TF(t_i, d)$ in title
- $\sum_{t_i \in q \cap d} TF(t_i, d)$ in URL
- $\sum_{t_i \in q \cap d} TF(t_i, d)$ in the whole document
- $\sum_{t_i \in q} IDF(t_i)$ in body
- $\sum_{t_i \in q} IDF(t_i)$ in anchor
- $\sum_{t_i \in q} IDF(t_i)$ in title
- $\sum_{t_i \in q} IDF(t_i)$ in URL
- $\sum_{t_i \in q} IDF(t_i)$ in the whole document
- $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ in body
- $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ in anchor
- $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ in title
- $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ in URL
- $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ in the whole document
- $LEN(d)$ of body
- $LEN(d)$ of anchor
- $LEN(d)$ of title
- $LEN(d)$ of URL
- $LEN(d)$ of the whole document

- BM25 of body
- BM25 of anchor
- BM25 of title
- BM25 of URL
- BM25 of the whole document
- LMIR.ABS of body
- LMIR.ABS of anchor
- LMIR.ABS of title
- LMIR.ABS of URL
- LMIR.ABS of the whole document
- LMIR.DIR of body
- LMIR.DIR of anchor
- LMIR.DIR of title
- LMIR.DIR of URL
- LMIR.DIR of the whole document
- LMIR.JM of body
- LMIR.JM of anchor
- LMIR.JM of title
- LMIR.JM of URL

[1] See [27]

Features

LMIR.JM of the whole document
Sitemap based term propagation
Sitemap based score propagation
Hyperlink based score propagation: weighted in-link
Hyperlink based score propagation: weighted out-link
Hyperlink based score propagation: uniform out-link
Hyperlink based feature propagation: weighted in-link
Hyperlink based feature propagation: weighted out-link
Hyperlink based feature propagation: uniform out-link
HITS authority
HITS hub
PageRank
HostRank
Topical PageRank
Topical HITS authority
Topical HITS hub
Inlink number
Outlink number
Number of slash in URL
Length of URL
Number of child page

BM25 of extracted title
LMIR.ABS of extracted title
LMIR.DIR of extracted title
LMIR.JM of extracted title

[1] S. Balakrishnan and S. Chopra.
Collaborative ranking.
In *Proceedings of WSDM*, pages 143–152, 2012.

[2] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li.
Learning to rank: From pairwise approach to listwise approach.
In *Proceedings of ICML*, pages 129–136, 2007.

[3] S. Chelaru, C. Orellana-Rodriguez, and I. S. Altingovde.
How useful is social feedback for learning to rank youtube videos?
World Wide Web, 17(5):997–1025, 2014.

[4] W. S. Cooper, A. Chen, and F. C. Gey.
Experiments in the probabilistic retrieval of full text documents.
In *Proceedings of TREC*, pages 127–134, 1994.

[5] W. S. Cooper, F. C. Gey, and D. Dabney.
Probabilistic retrieval based on staged logistic regression.
In *Proceedings of SIGIR*, pages 198–210, 1992.

[6] D. Cossock and T. Zhang.
Statistical analysis of Bayes optimal subset ranking.
In *Proceedings of COLT*, pages 605–619, 2006.

[7] K. Crammer and Y. Singer.
Pranking with ranking.
In *Proceedings of NIPS*, pages 641–647, 2001.

[8] K. Crammer and Y. Singer.
A new family of online algorithms for category ranking.
In *Proceedings of SIGIR*, pages 151–158, 2002.

[9] W. Croft.
Combining approaches to Information Retrieval.
In *Advances in Information Retrieval: Recent research from the Center for Intelligent Information Retrieval*, pages 1–36, 2000.

[10] W. B. Croft.
Incorporating different search models into one document retrieval system.
In *Proceedings of SIGIR*, pages 40–45, 1981.

[11] V. Dang, M. Bendersky, and W. B. Croft.
Two-stage learning to rank for information retrieval.
In *Proceedings of ECIR*, pages 423–434, 2013.

[12] Z. Dou, R. Song, X. Yuan, and J.-R. Wen.
Are click-through data adequate for learning web search rankings?

In *Proceedings of CIKM*, pages 73–82, 2008.

[13] N. Fuhr.
Optimum polynomial retrieval functions based on the probability ranking principle.
ACM Transactions on Information Systems, 7(3):183–204, 1989.

[14] F. C. Gey.
Inferring probability of relevance using the method of logistic regression.
In *Proceedings of SIGIR*, Pittsburgh, PA, July 1993.

[15] A. Habibian, T. Mensink, and C. G. Snoek.
Discovering semantic vocabularies for cross-media retrieval.
In *Proceedings of ICMR*, pages 131–138, 2015.

[16] R. Herbrich, T. Graepel, and K. Obermayer.
Large margin rank boundaries for ordinal regression, chapter 7, pages 115–132.
MIT Press, 2000.

[17] S. Jameel, W. Lam, and L. Bing.
Supervised topic models with word order structure for document classification and retrieval learning.
Information Retrieval Journal, 18(4):283–330, 2015.

[18] D. Jiang, J. Pei, and H. Li.
Mining search and browse logs for web search: A survey.
ACM Trans. Intell. Syst. Technol., 4(4):57:1–57:37, Oct. 2013.

[19] F. Jin, M. Huang, and X. Zhu.
A comparative study on ranking and selection strategies for multi-document summarization.
In *Proceedings of COLING*, pages 525–533, 2010.

[20] T. Joachims.
Optimizing search engines using clickthrough data.
In *Proceedings of SIGKDD*, pages 133–142, 2002.

[21] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay.
Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search.
ACM Trans. Inf. Syst., 25(2), Apr. 2007.

[22] C. Li, Y. Lu, Q. Mei, D. Wang, and S. Pandey.
Click-through prediction for advertising in twitter timeline.
In *Proceedings of SIGKDD*, pages 1959–1968, 2015.

[23] H. Li.

Learning to Rank for Information Retrieval and Natural Language Processing.
Morgan and Claypool, 2011.

[24] J. Li, D. Xu, and W. Gao.
Removing label ambiguity in learning-based visual saliency estimation.
IEEE Transactions on Image Processing, 21(4):1513–1525, 2012.

[25] P. Li, C. Burges, , and Q. Wu.
Mcrank: Learning to rank using multiple classification and gradient boosting.
In *Proceedings of NIPS*, pages 897–904, 2008.

[26] N. Liu, J. Yan, D. Shen, D. Chen, Z. Chen, and Y. Li.
Learning to rank audience for behavioral targeting.
In *Proceedings of SIGIR*, pages 719–720, 2010.

[27] T.-Y. Liu.
Learning to Rank for Information Retrieval.
Springer, 2011.

[28] Y. Liu, B. Gao, T.-Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li.
Browserank: Letting web users vote for page importance.
In *Proceedings of SIGIR*, pages 451–458, 2008.

[29] Y. Liu, T.-Y. Liu, B. Gao, Z. Ma, and H. Li.
A framework to compute page importance based on user behaviors.
Inf. Retr., 13(1):22–45, Feb. 2010.

[30] M. Melucci.
Contextual Search: A Computational Framework.
Foundations and Trends in Information Retrieval. Now Publishers, 2012.

[31] R. Nallapati.
Discriminative models for information retrieval.
In *Proceedings of SIGIR*, pages 64–71, 2004.

[32] Y. Ouyang, W. Li, S. Li, and Q. Lu.
Applying regression models to query-focused multi-document summarization.
Information Processing and Management, 47(2):227–237, 2011.

[33] Y. Pan, T. Yao, X. Tian, H. Li, and C.-W. Ngo.
Click-through-based subspace learning for image search.
In *Proceedings of MM*, pages 233–236, 2014.

[34] F. Radlinski and T. Joachims.
Query chains: learning to rank from implicit feedback.

In *Proceedings of SIGKDD*, pages 239–248, 2005.

[35] F. Radlinski and T. Joachims.
Active exploration for learning rankings from clickthrough data.
In *Proceedings of SIGKDD*, pages 570–579, 2007.

[36] S. Robertson and K. Sparck Jones.
Relevance weighting of search terms.
Journal of the American Society for Information Science, 27:129–146, May 1976.

[37] J. Rocchio.
Relevance feedback in information retrieval.
In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter 14, pages 313–323. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[38] Y. Tagami, S. Ono, K. Yamamoto, K. Tsukamoto, and A. Tajima.
Ctr prediction for contextual advertising: Learning-to-rank approach.
In *Proceedings of the Workshop on Data Mining for Online Advertising*, ADKDD '13, pages 1–8, 2013.

[39] J. Tang, N. Liu, J. Yan, Y. Shen, S. Guo, B. Gao, S. Yan, and M. Zhang.

Learning to rank audience for behavioral targeting in display ads.
In *Proceedings of CIKM*, pages 605–610, 2011.

[40] I. Vatolkin, M. Preuß, and G. Rudolph.
Multi-objective feature selection in music genre and style recognition tasks.
In *Proceedings of GECCO*, pages 411–418, 2011.

[41] A. Veloso, H. de Almeida, M. Goncalves, and M. J. Wagner.
Learning to rank at query-time using association rules.
In *Proceedings of SIGIR*, pages 267–274, 2008.

[42] M. N. Volkovs and R. S. Zemel.
New learning methods for supervised and unsupervised preference aggregation.
J. Mach. Learn. Res., 15(1):1135–1176, 2014.

[43] J. Xu, Y. Cao, H. Li, and Y. Huang.
Cost-sensitive learning of svm for ranking.
In *Proceedings of ECML*, pages 833–840, 2006.

[44] J. Xu, Y. Cao, H. Li, and M. Zhao.
Ranking definitions with supervised learning methods.
In *Proceedings of WWW*, pages 811–819, 2005.

[45] J. Xu, Y. Cao, H. Li, and M. Zhao.
Ranking definitions with supervised learning methods.
In *Proceedings of WWW*, pages 811–819, 2005.

[46] J. Xu, C. Chen, G. Xu, H. Li, and E. R. T. Abib.
Improving quality of training data for learning to rank using click-through data.
In *Proceedings of WSDM*, pages 171–180, 2010.

[47] J. Xu and H. Li.
Adarank: a boosting algorithm for information retrieval.
In *Proceedings of SIGIR*, pages 391–398, 2007.

[48] J. Xu, H. Li, and C. Zhong.
Relevance ranking using kernels.
In *Proceedings of AIRS*, pages 1–12, 2010.

[49] Y. Yang and W. Hsu.
Video search reranking via online ordinal reranking.
In *Proceedings of ICME*, pages 285—288, 2008.

[50] Y. Yang, P. Wu, C. Lee, K. Lin, W. Hsu, and H. Chen.

Contextseer: context search and recommendation at query time for shared consumer photos.
In *Proceedings of MM*, pages 199—208, 2008.

[51] X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan.
Beyond clicks: Dwell time for personalization.
In *Proceedings of RecSys*, pages 113–120, 2014.

[52] X. W. Zhao, Y. Guo, Y. He, H. Jiang, Y. Wu, and X. Li.
We know what you want to buy: A demographic-based system for product recommendation on microblogs.
In *Proceedings of SIGKDD*, pages 1935–1944, 2014.

[53] Y. Zhu, G. Wang, J. Yang, D. Wang, J. Yan, J. Hu, and Z. Chen.
Optimizing search engine revenue in sponsored search.
In *Proceedings of SIGIR*, pages 588–595, 2009.