Web Search Result Summarization: Title Selection Algorithms and User Satisfaction

Tapas Kanungo*
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
tkanungo@microsoft.com

Nadia Ghamrawi
Yahoo! Labs
701 First Ave
Sunnyvale, CA 94089
nadiag@yahoo-inc.com

Ki Yeun Kim
Yahoo!
701 First Ave
Sunnyvale, CA 94089
kykim@yahoo-inc.com

Lawrence Wai
Yahoo!
701 First Ave
Sunnyvale, CA 94089
wai@yahoo-inc.com

ABSTRACT

Eye tracking experiments have shown that titles of Web search results play a crucial role in guiding a user's search process. We present a machine-learned algorithm that trains a boosted tree to pick the most relevant title for a Web search result. We compare two modeling approaches: i) using absolute editorial judgments and ii) using pairwise preference judgments. We find that the pairwise modeling approach gives better results in terms of three offline metrics. We present results of our models in four regions. We also describe a hybrid user satisfaction evaluation process — search success — that combines page relevance and user click behavior, and show that our machine-learned algorithm improves in search success.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation, Theory

Keywords

Web summarization, machine learning, user satisfaction

1. INTRODUCTION

While the "ten blue links" that search engines return in response to a query are important in the user's search, the titles and summaries associated with the links can greatly influence a user's perception of the link's relevance and the efficiency of the search. Furthermore, badly formed abstracts can lead to "click inversions" where documents ranked lower get more clicks [1].

Titles on the search result page convey the first impression of relevance of the pages to the user. Most major search engines use a variety of sources to pick the title for a search result. Common sources include: HTML title, anchor text, internal anchor text, open directory page title, various HTML headline titles on the page, Yahoo! directory page title, etc. Thus it is important to have a systematic way of picking the best title from the candidate set.

Web page summaries can be query-independent [6] or query dependent [9, 10]. A query-independent summary conveys general information about the document, and can be computed offline and cached for fast access. The main problem with query-independent summaries is that they do not convey to the user *why* the Web page is relevant to the query. Query-dependent summarization attempts to address this by biasing the summaries towards the query. These summaries are typically constructed at query time.

In this paper, we present a machine-learned algorithm for selecting titles. Given a query and a Web page, a query-dependent title is generated as follows. First, all candidate titles are identified, and their features are extracted. Next for each title, we get an editorial (human) judgment regarding their relevance to the query and document on an absolute scale. Then we learn a regression model using features as the independent variables and the judgment as the target. We also show how to induce pairwise preference judgments and train a regression model using the preference judgments.

Machine learning approaches have been recently proposed for query-dependent sentence selection. Wang *et al.* [10] showed that ranking support vector machines (SVMs) outperform SVM classifiers and BM25 on a test collection that only consisted of 10 queries. Metzler and Kanungo [8] use TREC data for sentence selection and propose a machine learning framework. However, they do not apply it to the Web domain and do not provide any click-based evaluation. The above approaches are similar to those proposed in the document ranking literature [7, 11].

Numerous evaluation approaches [2, 4, 5] have been used in the past to to model and measure user satisfaction. Our summarization work, in contrast is based on large samples of Web data and we use a hybrid evaluation approach based on clicks and editorial judgments to judge user satisfaction, in addition to the standard IR metrics like mean reciprocal rank and discounted cumulative gain.

The key contributions of our work are: i) we present a machine-learned algorithm for selecting titles of Web pages in Web search results, ii) we compare absolute regression models for title selec-

*This was done while Kanungo was at Yahoo! Labs.

CIKM'09, November 2–6, 2009, Hong Kong, China.

tion with pairwise preference based models, iii) we propose a hybrid user satisfaction evaluation method that uses clicks and editorial judgments to quantify user satisfaction, and iv) we show results on the US, Taiwan (TW), Korea (KR) and Japan (JP) regions.

2. STATISTICAL MODELING

In this section, we describe two machine learning algorithms for title selection. We use Gradient Boosted Decision Trees (GBDT) to learn models from absolute judgments and Gradient Boosted Ranking (GBRank) to learn from pairwise preference judgments.

2.1 Gradient Boosted Decision Trees

GBDT is a technique that can be used for estimating a regression model. We use the stochastic variant of GBDTs [3]. GBDTs compute a function approximation by performing a numerical optimization in the function space instead of the parameter space. We provide an overview of the the GBDT algorithm.

A basic regression tree $f(x), x \in \mathcal{R}^K$, partitions the space of explanatory variable values into disjoint regions $R_j, j = 1, 2, \ldots, J$ associated with the terminal nodes of the tree. Each region is assigned a value ϕ_j such that $f(x) = \phi_j$ if $x \in R_j$. Thus the complete tree is represented as:

$$T(x; \Theta) = \sum_{j=1}^{J} \phi_j I(x \in R_j),$$

where $\Theta = \{R_j, \phi_j\}_1^J$, and I is the indicator function. Let (x_i, y_i), $i = 1, \ldots, N$ be the given set of points. For a loss function $L(y_i, \phi_j)$, parameters are estimated by minimizing the total loss:

$$\hat{\Theta} = \arg\min_{\Theta} \sum_{j=1}^{J} \sum_{x_i \in R_j} L(y_i, \phi_j).$$

A boosted tree is an aggregate of such trees, each of which is computed in a sequence of stages. That is,

$$f_M(x) = \sum_{m=1}^{M} T(x; \Theta_m),$$

where at each stage m, Θ_m is estimated to fit the *residuals* from the $m-1$th stage:

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} L(y_i, \eta f_{m-1}(x_i) + \phi_{j_m}).$$

where η is a the *learning rate*. In the *stochastic* version of GBDT, instead of using the entire data set to compute the loss function, one sub-samples the data and finds the values ϕ_j that minimize the loss on the test set. The stochastic variant minimizes over-fitting.

In practice, one has to empirically set (by cross-validation) the parameters: the number of trees M, the number of nodes per tree P, learning rate η, and sampling rate ρ, (in the stochastic version).

2.2 Pairwise Preference Models

During the process of collecting absolute judgments, an editor first sees a query and a set of possible titles and then assigns absolute grades to the titles. However, since the editor sees all the titles simultaneously, we suspect the grades are not independent. That is, the grades assigned can easily influence the grade of the next title. This facilitates the use of preference judgments for modeling.

Let $S = \{(x_i, y_i) | g(x_i) \geq g(y_i), i = 1, \ldots, N\}$ be the set of preference judgments such that $x_i \in \mathcal{R}^K$ is the feature vector for query $q(x_i)$ and title $t(x_i)$, $y_i \in \mathcal{R}^K$ is the feature vector for $q(y_i)$ and title $t(y_i)$, and the editorial grades are represented by $g(x_i)$ and $g(y_i)$ respectively.

The learning algorithm needs to learn a function h such that $h(x_i) \geq h(y_i)$ for $x_i, y_i \in S$, or at least try to minimize the number of disagreements with the editorial judgments. GBRank [11] tries to achieve this and a sketch of the algorithm is given below.

1. Guess an initial h_k for $k = 0$.
2. For $k = 1, \ldots, M$.
   (a) Use h_{k-1} as an approximation of h and compute:
   $$S^+ = \{(x_i, y_i) \in S | h_{k-1}(x_i) \geq h_{k-1}(y_i) + \tau\}$$
   $$S^- = \{(x_i, y_i) \in S | h_{k-1}(x_i) < h_{k-1}(y_i) + \tau\}$$
   where $\tau = \alpha(g(x_i) - g(y_i))$
   (b) Fit any regression function g_k (e.g. using GBDT), to correct the incorrectly classified examples:
   $$\{(x_i, h_{k-1}(y_i)+\tau), (y_i, h_{k-1}(x_i)-\tau) | (x_i, y_i) \in S^-\}$$
   (c) Form the current approximate function:
   $$h_k(x) = \frac{k h_{k-1}(x) + \eta g_k(x)}{k+1},$$
   where η is the learning rate.

As in the case of GBDT, GBRank parameters M, P, α, η, and ρ have to be experimentally set (by cross-validation). Both GBDT and GBRank also provide *feature importance* [3], which is computed by keeping track of the reduction in the loss function at each feature variable split and then computing the total reduction of loss function along each explanatory feature variable. The importance is useful for analyzing which features contribute most to the model.

3. FEATURES

In this section we describe a subset of the features we used in our experiments. Some features are query-dependent and others are query-independent.

3.1 Query-Dependent Features

Query-dependent features capture relevance at different levels of granularity and expressiveness. They include:

UniqueQueryUnitHits: Unique query term hits

DuplicateQueryUnitHits: Repeated query term hits

QueryTermHitsFrac: Fraction of query terms hit

FirstHitOffset: The position of first query term hit

HitsCompactness: Number of hits over the hit offset range

URLMatchQ: Number of title terms found in the URL that are not query hits.

3.2 Query-Independent Features

Query-independent features attempt to express *prior* knowledge about titles. They represent the degree to which the title captures the document nature and genre. In fact, they can be used, in part, to pick the best query-independent title. They include:

ClickTextMatch: Fraction of terms that are present in the URL's ClickText. ClickText of a URL is the set of queries for which the URL is clicked, weighted and pruned by a function of clicks and impressions.

URLMatch: Fraction of terms in URL that also occur in the title

URLMatch: Fraction of title terms that also occur in the URL

ScriptLFrac: extent of foreign language characters or words

TitleSourceX: Binary features indicating title source

Other query-independent features capture structural attributes of titles, thus addressing readability at more coarse granularity. They consider fraction of capitalized letters and words, title length in words and characters, word length, and punctuation, for example.

4. MODELING: PROTOCOL

4.1 Data Sampling and Characteristics

Our train and test data was generated as follows. We randomly sampled 425 queries from a two-week query stream in the US region. Each query was issued to the search engine and top 10 URLs were collected. Then a random subset of 2,169 query-URL pairs was selected for editorial judgment. On average, a URL has three title candidates, and URLs for popular queries have up to seven titles. The final train-test sample consists of 7,456 query-URL-title triples. Table 1 has an example of titles for a query-URL pair.

For each query-URL pair, editors assigned a grade (1-5, 5 being the best possible grade) for each title. In addition, they chose a single best title among candidate titles. Data set generation in JP, TW, and KR was similar. Characteristics of the data are in Table 2.

Table 1: An example query-URL pair and associated titles.

URL	http://www.theknot.com/	
QUERY	the knot	
Title Source	**Title Text**	**Judgment**
anchortext	**the knot**	4
headline1	featured content	2
Ydirectory	**The Knot**	5
HTML	Wedding Dresses | Wedding Cakes | Wedding Planning | Unique Wedding Ideas	3

Table 2: Data Characteristics: "avg # t" is average number of title candidates for all query URL pairs. "avg Grd" represents average grade of all titles while "avg B" and "avg NB" are average grade of best titles and of non-best titles, respectively.

lng	**#** q	**#** (q,u)	**#** (q,u,t)	**avg #t**	**avg Grd**	**avg B**	**avg NB**
US	425	2,169	7,456	3.4	3.6	4.5	3.2
JP	379	3,346	13,541	4.0	3.4	4.4	3.1
TW	725	6,190	16,980	2.7	2.9	3.6	2.7
KR	507	2,218	3,612	1.6	3.3	3.6	3.0

4.2 Training and Testing

Experiments consisted of multiple trials. In each trial we trained on a random 70% of data (without replacement) and tested on the remaining 30%. We report averages over trials and t-tests to determine whether the optimal model is better than the basic model.

Query popularity is used to influence the tuple weight in training and testing. We used a scaled, discretized, and smoothed log query frequency as a weight.

We ran experiments varying GBDT/GBRank parameters (different values for M, P, η, and ρ, for example). We chose the optimal parameter setting via cross-validation.[1]

4.3 Evaluation Metrics

Several offline measures are used to estimate the quality of a model. Suppose the test data consists of query-URL pairs $D = (q_1, u_1)...(q_n, u_N)$. The metrics are as follows:

- Accuracy (*ACC*): $\frac{1}{N}\sum_{i=1}^{N} b(r_{f_M}(q_i, u_i, 1))$
- Mean Reciprocal Rank (*MRR*): $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}\frac{b(r_{f_M}(q_i,u_i,j))}{j}$, where pair i has k titles
- Average Grade (*GRD*): $\frac{1}{N}\sum_{i=1}^{N}\frac{g(r_{f_M}(q_i,u_i,1))}{max_{j=1}^{k} g(r_{f_M}(q_i,u_i,j))}$ (note that this is the same as scaled DCG1.)

where $g(t)$ is the grade of title t and r_{f_M} is the ranking function for the model f_M. Thus $r_{f_M}(q_i, u_i, j) = t_{i,j}$ is the *j*th best title, according to f_M, for the *i*th query-URL pair. $b(t_j) = 1$ if t_j is the *best* title picked by the editor, and 0 otherwise.

5. MODELING: RESULTS

We present results for the GBDT and GBRank title selection algorithms for US, JP, TW and KR, and compare the two modeling approaches. Performance of GBRank in the pairwise setting is significantly better than GBDT. We present the timing performance of these algorithms.There are four types of models:

- *gr.basic:* A model trained using GBDT and a *basic* set of features, consisting of all features except ClickTextMatch or URLMatch features (for US, also lacking ScriptLFrac).
- *pw.basic:* Similar to *gr.basic*, trained using pairwise GBRank.
- *gr.optimal:* trained using GBDT and a complete set of features, including clicktext, URL-match and language features.
- *pw.optimal: optimal* model trained using pairwise GBRank.

On average over 30 random trials in US, *pw.basic* had a 0.56% improvement in ACC over *gr.basic*. However, adding features and optimizing to produce *gr.optimal* yielded a 0.6% improvement over *gr.basic*. The model *pw.optimal* trumps this improvement, resulting in a 2.76% improvement in performance over *gr.basic* (see Table 3). Model *pw.optimal* is better than *pw.basic* by 2.17% for US, 3.07% for JP, 4.09% for TW and 4.35% for KR.

Table 3: Average performance over 30 trials for the three offline metrics. t-tests show that the optimal model is better than the basic model for the three regions. In each case we find that the difference is statistically significant at the 0.05 level.

language	**experiment**	**ACC**	**MRR**	**GRD**
US	gr.basic	0.8408	0.9187	0.9732
	pw.basic	0.8455	0.9208	0.9743
	gr.optimal	0.8460	0.9210	0.9737
	pw.optimal	0.8677	0.9318	0.9756
JP	gr.optimal	0.8345	0.9074	0.9909
	pw.optimal	0.8601	0.9218	0.9924
TW	gr.optimal	0.8859	0.9405	0.9825
	pw.optimal	0.9221	0.9592	0.9903
KR	gr.optimal	0.8739	0.9361	0.9816
	pw.optimal	0.9119	0.9556	0.9908

Additionally we use metrics that evaluate the *coverage*, or fraction of titles that changed, over a set of 3000 random query-url pairs. Model *pw.optimal* had 11% coverage with respect to *gr.basic*.

Parameter α, described in section 2, allows judgment grades to influence the score ($\tau = \alpha(g(x_i) - g(y_i))$). Value $\alpha = 0$ uses only pairwise preferences. Figure 1 suggests that graded judgments decrease performance when used in this way (for $0 < \alpha < 0.1$ the same pattern occurred). One possible explanation is that pairwise preferences are more reliable than graded judgments in a pairwise setting, and the grades do not add much to the data point.

Table 3 suggests that with the addition of a richer feature vocabulary, GBRank performance improves, while GBDT performance may saturate. GBDT modeling may be more sensitive to sparse training datasets than pairwise models, since it requires sufficient data for each grade. Additionally, while GBDT has N training examples for each query-URL pair, GBRank has $C(N, 2)$ pairs of feature vectors (43% more datapoints than GBDT).

Both approaches select titles in real-time. Based on 96,545 randomly selected query-URL pairs, on average, *gr.optimal* spent 0.12

[1]For US $pw.optimal$: $M = 750$, $P = 20$, $\eta = 0.05$, $\rho = 0.3$. For US $gb.optimal$: $M = 500$, $P = 6$, $\eta = 0.05$ and $\rho = 0.7$. For JP/TW/KR $gr.optimal$: $M = 1000$, $\eta = 0.15$ and $\rho = 0.7$. For JP/TW $P = 10$ and for KR $P = 8$.

ms to select the best title, while *pw.optimal* spent 0.24 ms. The difference can be explained partly by the fact that *pw.optimal* uses 750 trees while *pw.basic* uses 500 trees. The performance numbers were collected on a 1.8GHz Intel Xenon machine running Linux.

According to feature importance, for GBDT, the top two features were structural. QueryTermHitsFrac, URLMatch, ClickTextMatch and TitleHTML were also important. GBRank was similar, but ranked TitleHTML, FirstHitOffset, ClickTextMatch and URLMatch relatively higher than GBDT.

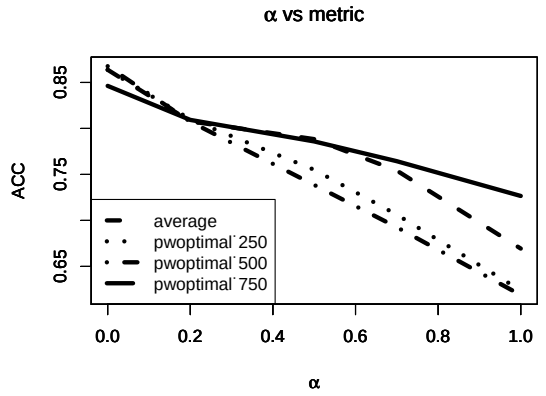

Figure 1: α vs performance for the average over all parameter configurations as well as for the top three performing configurations. *pw.optimal* uses $M = 750$ trees; other models use the same parameters but $M = 500$ or $M = 250$.

6. USER SATISFACTION

6.1 "Search Success" Metric

We measured user satisfaction with the "search success" metric (S), which is defined per query as follows:

- S=1 if the user clicks through to at least one document of good or better quality for non-navigational queries
- S=1 if the user clicks through to at least one document of perfect or excellent quality for navigational queries
- S=0 otherwise

6.2 Experimental Methodology

We randomly sampled a few thousand queries from the control ($N_{control}$) and test (N_{test}) populations over the same week. For each clicked result, the query document relevance was measured by editorial staff, and whether the query was navigational or not. The search success S was measured for every query and the mean search success for test $\bar{S}_{test}$ and control $\bar{S}_{control}$ was computed. A Gaussian approximation to the binomial distribution was used to estimate the error: $\sigma = \sqrt{\frac{\bar{S}\cdot(1-\bar{S})}{N}}$.

6.3 Results

Overall search success rates for the control sample are shown in Table 4, along with click through rate (CTR) for the query results returned at ranks 1 (CTR_1) and 2 (CTR_2). We have divided the queries into navigational vs non-navigational, as well as the cases where the query result document relevance at rank 1 (g_1) is better or worse than the query result document relevance at rank 2 (g_2).

The test sample showed a decrease in CTR_1 by 2%, an increase in CTR_2 by 1%, and a statistically significant increase in mean search success $\bar{S}$ by 0.7% ($> 0.1\%$ at 95% confidence). We found that the queries which had $g_1 < g_2$ contributed to the decrease in CTR_1; in particular, we observed a decrease of clicks on bad and fair documents with the improved summaries. The increase in CTR_2 (and lower) was due to more clicks on good or better documents. The combination of these effects resulted in an increase in $\bar{S}$, which we interpret to be an improvement in user satisfaction. We obtained similar results to those in Table 4 for TW and JP.

Table 4: CTR vs Search Success for US

query	ranking	%	CTR@1	CTR@2	S
non-nav	$g_1 \geq g_2$	59.5%	33.4%	11.6%	25.4%
non-nav	$g_1 < g_2$	9.4%	27.7%	18.1%	18.1%
nav	$g_1 \geq g_2$	30.0%	65.3%	7.7%	68.3%
nav	$g_1 < g_2$	1.1%	36.4%	18.2%	36.4%

7. CONCLUSIONS

Titles and abstracts shown on a search result page influence user perception of a link's relevance. In this paper we presented a fast query-dependent machine-learned algorithm for selecting titles for links to Web pages on a Web search result page. We presented a model trained using absolute human judgments, and pairwise preference judgments. We find that the pairwise training algorithm performs better than the absolute judgment model. We also presented a hybrid metric, *search success*, that uses clicks and editorial judgments to quantify user satisfaction, and show that our algorithm performs better than the baseline.

8. REFERENCES

[1] C. Clarke, E. Agchtein, S. Dumais, and R. White. The influence of caption features on clickthrough patterns in web search. In *Proc. of SIGIR*, 2007.

[2] S. Dumais, T. Joachims, K. Bharat, and A. Weigend. SIGIR 2003 workshop report: Implicit measures of user interests and preferences. *ACM SIGIR Forum*, 37:50–54, 2003.

[3] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38:367–378, 2001.

[4] M. A. Hearst. Models of information seeking. In *Search User Interfaces*. 2009.

[5] R. Khan, D. Mease, and R. Patel. The impact of result abstracts on task completion time. In *Proc. of WWW Workshop on Web Search Result Summarization and Presentation*, 2009.

[6] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. In *Proc. 18th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 68–73, 1995.

[7] P. Li, C. J. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Proc. 21st Proc. of Advances in Neural Information Processing Systems*, 2007.

[8] D. Metzler and T. Kanungo. Machine learned sentence selection strategies for query-biased summarization. In *Proceedings of SIGIR Workshop on Learning to Rank for Information Retrieval*, 2008.

[9] A. Tombros and M. Sanderson. Advantages of query biased summaries in information retrieval. In *Proc. 21st Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 2–10, 1998.

[10] C. Wang, F. Jing, L. Zhang, and H.-J. Zhang. Learning query-biased web page summarization. In *Proc. 16th Intl. Conf. on Information and Knowledge Management*, pages 555–562, 2007.

[11] Z. Zheng, H. Zha, , K. Chen, and G. Sun. A regression framework for learning ranking functions using relative judgments. In *Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 287–294, 2007.