MICROSOFT RESEARCH ASIA AT WEB TRACK AND TERABYTE TRACK OF TREC 2004

Ruihua Song, Ji-Rong Wen, Shuming Shi, Guomao Xin, Tie-Yan Liu, Tao Qin, Xin Zheng, Jiyu Zhang, Guirong Xue and Wei-Ying Ma

Microsoft Research Asia
5f, Beijing Sigma Center
No.49, Zhichun Road Haidian District
Beijing 100080, P. R. China

INTRODUCTION

Here we report Microsoft Research Asia (MSRA)'s experiments on the mixed query task of Web track and Terabyte track at TREC 2004.

For Web track, we mainly test a set of new technologies. One of our efforts is to test some new features of Web pages to see if they are helpful to retrieval performance. Title extraction, sitemap based feature propagation, and URL scoring are of this kind. Another effort is to propose new function or algorithm to improve relevance or importance ranking. For example, we found that a new link analysis algorithm named HostRank that can outweigh PageRank [4] for topic distillation queries based on our experimental results. Eventually, linear combination of multiple scores with normalizations is tried to achieve stable performance improvement with mixed queries. We highlight several of the main technologies below:

- Title extraction: While title has been proved to be a high-quality information source in past experiments, we found that titles of about 1/3 pages in the .GOV dataset are meaningless. For example, some pages miss the title fields, some have titles of "untitled", and sometimes too many pages in one web site share the same title. We use a method to automatically extract proper titles for these pages. Experimental results showed about 6% improvement for named/home page finding and a small improvement for topic distillation.
- Sitemap based feature propagation: In the topic distillation task, a key resource is not necessarily highly relevant to the query by itself since it may be an entry that points to other good children pages. Only using words in the page may lead to bad ranking of this key resource. To avoid this difficulty, we proposed an approach to propagate keywords from children pages to parent pages following the site tree.
- URL scoring: Compared to last year's experiments, we pay more attention to utilizing information in URLs this year. Maximum string matching is used in our experiments and both the length and degree of matched substring are taken account of in URL relevance scoring. On the other side, our URL priors are functions of the URL and the query and are classified based on the difference between the depth of URL and the depth of the last query term occurrence. Finally five classes are defined (zero, one, two, no less than three and non-occurrence) in our experiments.

- HostRank: We continue to test the effectiveness of link analysis algorithms this year. A new HostRank algorithm is used for the first time. While previous PageRank algorithms compute an importance value for a web page, HostRank compute an importance value for a web site or a host. And the hierarchy structure of a host is then used to distribute the host's importance to web pages within the host.
- Query classification: We have tried to tune two sets of separate parameters for ranking functions of topic distillation and named/home page finding respectively. But we find that query classification is not an easy task and experimental results showed little improvement when using separate parameters in comparison with turning a set of parameters for mixed queries. Therefore, we did not use query classification in our submissions for mixed query task, but we submitted 5 runs for query classification task anyway. Details on our query classification are described in the section of "Query Classification".

For terabyte track, we build a distributed search engine to parse, index and search the 25M Web pages. Every 4 millions of pages are indexed at one machine and local idf is used in ranking functions instead of global idf. The speed is about 1.5 second per query. Basically, the ranking methods we used in the terabyte track are similar to those we used in the Web track.

We submitted 5 runs for the mixed query task, 5 runs for the query classification task and 5 runs for Terabyte track. The evaluation results and additional experimental results are shown in the section of "Experimental Results" section.

TITLE EXTRACTION

Title, as a general or descriptive heading of a document, has been proved to be a high-quality information source in past experiments. Generally, title is easy to extract since it is marked by <TITLE> and </TITLE> tags in HTML. However, we found that about 1/3 pages in .GOV have no meaningful titles. There are three different categories of meaningless titles:

1) The title field is absent or blank (5.76%);
2) A default title such as "untitled" or automatically generated title is used (0.81%);
3) The same title is used for many pages in a Web site (26.90%). In our study, a title would be considered as failing into this category when it reoccurs more than N times in a Web site. The minimum value of N is set to 5 and it will increases with the scale of a Web site. We treat this kind of title as meaningless tiltles since they are a negative factor to discriminate different pages. For example, the website "student.gov" contains 10108 pages and 5395 of these pages use the same *"Students.Gov - Student Gateway to the U.S. Government"* title. Figure 1 shows two pages from this website. Obviously, *"national parks"* and *"health care"* are more suitable titles for them respectively.

MSRA has used machine learning approaches to train a title extractor[1]. The extractor extracts titles for 71% pages in the .GOV dataset. For each page, title and extracted title are merged to a ComboTitle in our experiments.

We compare the retrieval performance between using original title and ComboTitle in this way: First, the relevance scores of only using Title and ComboTitle are calculated respectively. Then the scores are linearly combined with the scores of using the document body respectively. All the scores are calculated by BM2500. The results on the query set of TREC2003 are shown in Table 1 and Table 2. It is shown that ComboTitle

[1] We appreciate Hang Li, Yunhua Hu and Yunbo Cao for sharing title extractor with us. They are working on extracting titles from doc, ppt and html files by machine learning approaches. And their papers are planned to submit to related conferences, such as SIGIR.

gained 6% more improvement than Title for the named/home page finding task and a tiny improvement for the topic distillation task.

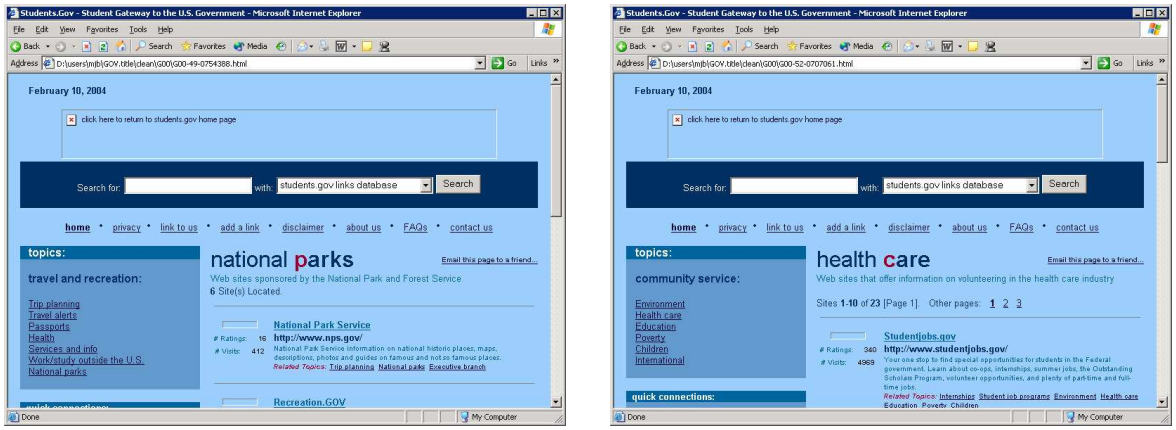

Figure 1. Two Pages from Student.gov

Table 1. Combination Results on 2003.NP/HP

	Average Precision	Improvement on Base
Base	0.574384	0
Base+Title	0.618306	+7.65%
Base+ComboTitle	0.652409	+13.58%

Table 2. Combination Results on 2003.TD

	Average Precision	Improvement on Base
Base	0.121941	0
Base+Title	0.146246	+19.93%
Base+ComboTitle	0.147089	+20.62%

SITEMAP BASED FEATURE PROPAGATION

According to the task description, for topic distillation, the answer should be "a good entry point to a website principally devoted to the topic". However, a good entry point is not necessarily highly relevant to the query by itself since it may be an entry point to other good children pages. Only using words in the page may lead to bad ranking of this key resource. For example, for the query TD4 "wireless communication" in TREC2003, totally seven pages in the same site (cio.doe.gov) appears in the top 1000 retrieval results by BM25 [5]. The relevance score of the entry page (with ID G07-78-3824915) is significantly smaller than its child pages G35-97-1056561, G36-35-1278614 and G35-01-1537522. If the query related features can be propagated from children pages to their parents according to the sitemap, the rank of an entry point can be boosted up. This approach is called sitemap based feature propagation (SBFP).

Table 3. An Example of Search Results without Feature Propagation

Rank	Document ID	Relevance	URL
70	G35-97-1056561	9.857887	cio.doe.gov/wireless/3g/3g_index.htm
470	G07-38-3990160	9.507885	cio.doe.gov/spectrum/groups.htm
477	G35-75-1119753	9.480914	cio.doe.gov/spectrum/philo.htm
518	G36-35-1278614	9.319749	cio.doe.gov/wireless/background.htm
571	G07-10-2999356	9.093273	cio.doe.gov/spectrum/background.htm
648	G35-01-1537522	8.816766	cio.doe.gov/wireless/wwg/wwg_index.htm
649	**G07-78-3824915**	**8.815378**	**cio.doe.gov/wireless/**

SBFP propagates the term frequency from children pages to its parent based on the following formula:

$$F'(p) = (1+\alpha) * F(p) + (1-\alpha) * \frac{\sum_{q \in Child(p)} F(q)}{|Child(p)|}$$

Where $F(p)$ is the feature of page p before propagation, $F'(p)$ is the feature of page p after propagation, q is the child page of p, and α is the weight which controls the contribution of the child pages. And correspondingly, SBSP approach modifies the document length of a page as the following:

$$L'(p) = (1+\alpha) * L(p)$$

Where $L(p)$ is the document length of p before propagation, and $L'(p)$ is the document length after propagation.

URL SCORING

Relevant URLs often contain query terms or their abbreviations (e.g. *bibliography* is shortened as *biblio*). And sometimes terms or abbreviations are glued together as one term in URLs (e.g. *video cast* are glued as *videocast,* and *electoral college* are shortened and glued as *electcoll*). An obvious way of exploring URL information is to match query terms and URL terms, but the naive method could not handle the abbreviation and glue problems, which are quite common on the Web. [3] tackles these problems by treating both the URL and a query term as a character sequence, and then computes a character-based trigram generative probability. In our experiments, we use maximum prefix string matching. For each query term, we found the maximum substring in URL terms that matches the prefix of the query term. For example, given the query of *electoral college* and the URL term of *electcoll*, the maximum matched substring is *elect* for *electoral* and *coll* for *college*. The preliminary URL relevance score is proposed as:

$$S_1 = \sum_i idf_i \cdot \frac{l_{matched}}{l_i}$$

Where,

idf_i is the term weight of the query term.

$l_{matched}$ is the count of the prefix characters matched

l_i is the length of the query term

The factor of $\frac{l_{matched}}{l_i}$ indicates how likely that the matched substring is a short of the query term.

However, S_1 cannot discriminate the following two URLs, though the first is relevant whereas the second is irrelevant.

http://.../electcoll/...

http://.../electronic/.../collection/...

We found that parts of *electronic* or *collection* are matched, while the whole of *eletcoll* is also matched. So we introduce another concept called layer-match degree. As we know, slashes split a URL into several layers, and each layer is the name of a domain, a directory or a file. Given a layer, we count the characters of the substring that matched prefix of any query term. And layer-matched degree is defined as the layer-matched count divided by the depth of the layer. Thus partial layer-match is punished. So the relevance score function of URL is modified as:

$$S_{rel} = \sum_i idf_i \cdot \frac{l_{matched}}{l_i} \cdot \frac{l_{layer-matched}}{l_{layer}}$$

Where,

idf_i , $l_{matched}$, l_i are same as those in S_1

$l_{layer-matched}$ is the total matched length of the layer

l_{layer} is the length of the layer

Previous works found that entry page URLs tend to be higher in a site tree than other web pages [2][3]. URLs are classed as 4 types: ROOT, SUBROOT, FILE and PATH. Such structure information has been proved useful in the home page finding task. This year, for the mixed query task, we further found that the layer of matched terms in URLs is a strong cue of target pages. For example, in Table 3, the URL *cio.doe.gov/wireless/* not only matches a query term *wireless,* but the matched layer is the last one.

We also assign each URL a prior probability of being a named page or homepage or key resource given the URL type. The URL type is query-dependent. Based on the difference between the depth of URL and the depth of the last query term occurrence, URLs are classes into 5 types: 0, 1, 2, >=3, and no match. Given a type *t*, this prior is consistently used for HP, NP and TD queries. Since NP, HP and TD are disjoint,

$P(TD \vee HP \vee NP \mid t) = P(TD \mid t) + P(HP \mid t) + P(NP \mid t)$

We rewrite it by Bayes' formula.

$$P(TD \vee HP \vee NP \mid t) = \frac{P(t \mid TD)P(TD) + P(t \mid HP)P(HP) + P(t \mid NP)P(NP)}{P(t)}$$

And we assume $P(NP) = P(HP) = P(TD) = 1/3$ when there is no knowledge about the proportions of NP, HP and TD queries. Thus the denominator is estimated by

$$c(t,TD)/3c(TD) + c(t,HP)/3c(HP) + c(t,NP)/3c(NP)$$

where $c(t,TD)$ denotes the number of type *t* relevant pages for topic distillation queries in the training data and $c(TD)$ denotes the number of all relevant pages for topic distillation queries in the training data. Here, $c(TD)$ is not equal to the number of queries because a TD query usually has more than one relevant document. In [11], such difference does not exist because the number of queries is equal to the number of relevant documents for NP and HP tasks.

$P(t)$ is estimated based on the training data. Since the URL hit type is a function of both a URL and a query, the collection of documents is aggregated by sampling using queries of TREC2002 (NP only) and TREC2003

(NP, HP and TD). For each query i, $c_i(t)$ is counted among top N documents returned by content retrieval. Given the number of queries for TD, HP and NP are n_{td}, n_{hp} and n_{np} respectively, $P(t)$ is estimated as the following formula does because we assumed that TD, HP and NP are equally likely.

$$P(t) = \frac{1}{3}\frac{\sum_{i=1}^{n_{td}} c_i(t)}{n_{td} \cdot N} + \frac{1}{3}\frac{\sum_{j=1}^{n_{hp}} c_j(t)}{n_{hp} \cdot N} + \frac{1}{3}\frac{\sum_{k=1}^{n_{np}} c_k(t)}{n_{np} \cdot N}$$

With a little rewriting and discarding the constants 3 and N, we get our formula for estimation of $P(t)$.

$$P(t) \propto \sum_{i=1}^{n_{td}} c_i(t) + \frac{n_{td}}{n_{hp}} \sum_{j=1}^{n_{hp}} c_j(t) + \frac{n_{td}}{n_{np}} \sum_{k=1}^{n_{np}} c_k(t)$$

Consequently, the estimated posteriors and priors for different URL types are shown in Table 4.

Using the prior estimated, our final URL score formula is:

$$S_{url} = S_{rel} \cdot P(TD \vee NP \vee HP \mid type(URL, Q))$$

We use the formula above to do the 2002 NP and 2003 NP task. For 2002 NP task the average precision is 0.1823, for 2003 NP task the average precision is 0.1324. On TREC2004 mixed query task, the performance of URL scoring is shown in Table 5. Results show that priors give an improvement for the unseen queries.

Table 4 Estimation of prior probabilities

Type(D,Q)	NP	HP	TD	NP at 194	TD at 194	NP+HP+TD at 194	Posterior	Top2000 result	Prior
0	160	70	325	96.1	32.7	198.8	0.3416	51693	0.00383
1	36	29	174	21.6	17.5	68.1	0.117	44014	0.00155
2	10	11	53	6.0	5.3	22.3	0.0383	57628	0.000387
>=3	19	3	18	11.4	1.8	16.2	0.0278	72555	0.000223
No Match	98	81	1359	58.9	136.7	276.6	0.4753	860078	0.00032
Total	323	194	1929	194	194	582	1	1085968	

Table 5 Average Precision of URL scoring on TREC2004

	TD	HP/NP
S_{rel} (no priors)	0.0468	0.1116
S_{url} (with priors)	0.0650	0.1487

HOSTRANK

We continue to test the effectiveness of link analysis algorithms this year. A new algorithm called HostRank is used for the first time. While previous PageRank algorithms compute an importance value for a web page, HostRank compute an importance value for a web site or a host. And the hierarchy structure of a host is then

used to distribute the host's importance to web pages within the host. Correspondingly, the computation of HostRank could be separated to 2 phrases:

1) Host's importance computation

At a coarse level of granularity, we can think of the Web as a collection of the hosts that grow more or less independently of each other. Thus, the Web graph is aggregated into the host graph by following steps:

Step 1: the host of the Web graph is detected, then the Web graph is partitioned into m hosts: $H_1, \ldots, H_m$.

And H_i contains all the node of the host *i*.

Step 2: all the hyperlinks of a host H_i that link to other hosts are aggregated to the host H_i. So the hyperlinks within the host are not taken into account.

Thus, the host graph is modeled as a weighted directed graph $G = (V, E)$, where V represent the hosts and V encompasses the links between hosts. Furthermore, each link $l_{i,j} \in E$ is associated with a parameter $W_{i,j}$ denoting the weight of the host H_j to host H_i, where the weight is calculated according to the number of hyperlinks between the hosts. After obtaining the host graph, we apply the PageRank algorithm to calculate the importance of the hosts.

2) Propagation of the host's importance

After getting the importance of the hosts, each page' importance score is calculated according to the host's importance and the hierarchical structure of the host. There are several factors should be taken into account: the depth of the URL, whether the page is an index page or a content page, links pointing to the page from outside hosts.

We conduct experiments to compare the performance between the PageRank and HostRank algorithms on the dataset of TD at TREC2003. We re-rank the results by linear combining the page's importance and its relevance score. Results are shown in Figure 2.

SCORE COMBINATION

After obtaining multiple scores from various sources, a challenging task is to combine them into an overall one to get a final ranking. In this year, we mainly explored linear score combination because of its simplicity. We believe that score normalization, which makes different kinds of scores to be comparable, is crucial to combination. So we adopt a two-step process to do combination here. In the first step, we do different kinds of normalization for different types of scores. And in the second step, the normalized scores are combined by adopting a hill-climbing method.

1) Score normalization

Scores we have computed for each page include: the base score (a BM25 score over all fields of the page), title BM25 score, anchor BM25 score, body text proximity score, title proximity score, anchor proximity score, URL score, PageRank score, HostRank score, etc. All these scores can be categorized as four types: BM25 score, proximity score, URL score, and PageRank/HostRank score. The four types of scores are computed by different mechanisms, and so it is reasonable that they are normalized in different ways. For each score type, we normalize the values to be in [0, 1], and hope that the *possible ideal* best scores will be 1 and worst scores be zero.

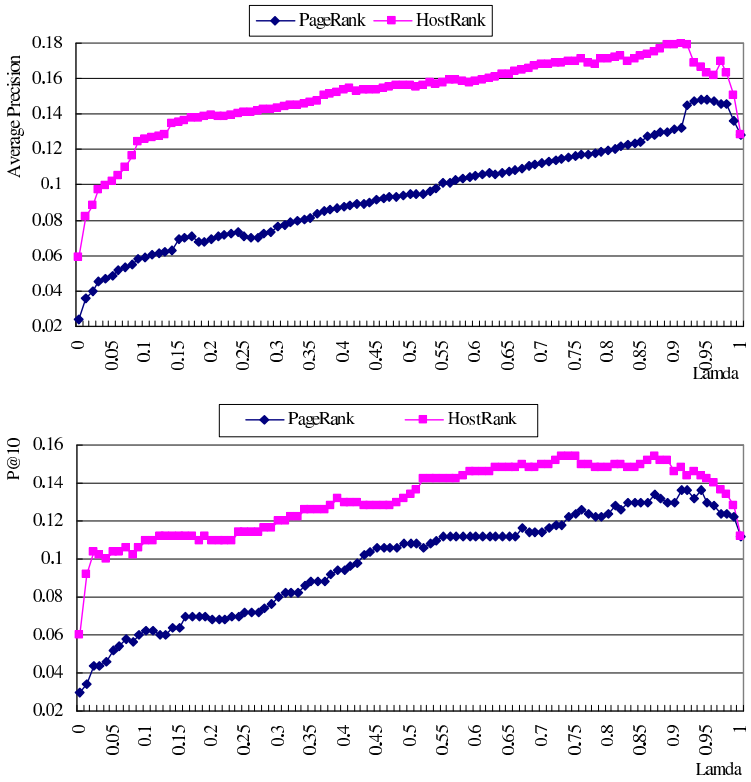

Figure 2. Comparison between PageRank and HostRank score combined with relevance score

All BM25 scores are obtained by using a form of BM25 formula [5][6] with parameters k_2 set to be zero and k_3 be large enough (e.g. 1000), that is,

$$S_{bm25} = \sum_{T \in Q} w^{(1)} \frac{(k_1 + 1)tf}{K + tf}$$

As $(k_1 + 1)tf / (K + tf)$ approaches to $k_1 + 1$ when tf is large enough, so the following formula will guarantee that the normalized scores are in [0, 1]:

$$S_{bm25_normalized} = \frac{S_{bm25}}{\sum_{T \in Q} w^{(1)} (k_1 + 1)}$$

PageRank scores have a zipf-like distribution where most documents have moderate or small scores, while the scores of a few documents are extremely high. Similar to most existing normalization methods for PageRank, we compute *logarithm* for scores as follows,

$$S_{pagerank_normalized} = \frac{\log(1 + S_{pagerank})}{\log(1 + S_{\max}))}$$

where S_{pagerank} is the original PageRank score, and $S_{\max}$ denotes the maximal scores of them.

The distribution of HostRank scores is quite different from that of PageRank scores. And computing logarithm for them seems can not produce good results in the combination step. Here we simply compute normalization scores independently for the first N results of each query by using the following formula.

$$S_{hostrank_normalized} = \frac{S_{hostrank} - S_{\min}}{S_{\max} - S_{\min}}$$

where S_{max}, S_{min} are the maximal and minimal HostRank score of the N results respectively.

The normalization of URL scores is similar to that of BM25 scores, i.e., normalized scores are computed by dividing the possible maximal URL score.

2) Combination

Given the normalized scores, we adopt a hill-climbing method to combine them and compute the final score. The official evaluation results for 2002/2003 np/hp/td are used for training the coefficients. It is really hard to get a global optimized coefficients using hill-climbing. The hill-climbing may converge to multiple local optimization values given different initial coefficients. We simply take two set of local optimized coefficient values for the final runs submitted.

QUERY CLASSIFICATION

For query classification, we first extract some features for each query, and then adopt machine learning methods to classify these queries. We totally extract eight kinds of features for query classification.

1) Information in the query itself, e.g. the length of the query (since the query length of TD is often short than that of NP and HP), the number of acronym in the query, the number of stop word and so on.
2) Information related to inverted document frequency.
3) BM25 scores in the top 2000 results for each query. For example, Figure 3 shows the difference between TD query and NP/HP query.
4) Information of matched titles in the top 20 BM25 results. There are many pages with completely matched titles to the queries of TD. However, there are only a few completely matched pages for NP/HP queries.
5) Information of matched anchors in the top 20 BM25 results.
6) Information of matched URL in the top 20 BM25 results.
7) Proximity in the top 20 BM25 results, including title proximity, anchor proximity and URL proximity.
8) URL depth of top 2 BM25 results. This kind of features are mainly applied to classify NP and HP since the answers of NP are often deeper in a site than those of HP.

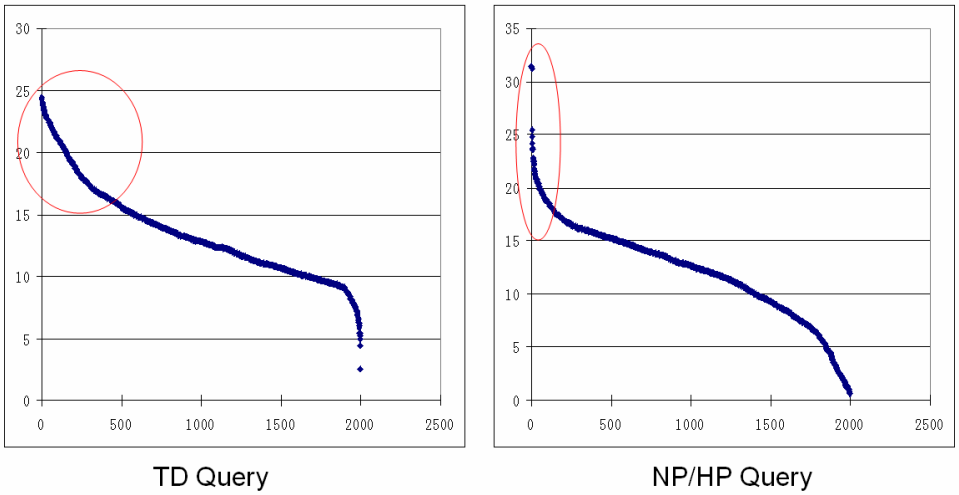

Figure 3. BM25 scores in the top 2000 results

In web track 2003, there are only two kinds of queries: TD and NP/HP. So we focus on classifying TD and NP/HP. Features (1-7) are mainly used to differentiate TD queries and NP/HP queries, and only the last one is used to differentiate NP and HP. This bias can be easily seen from the final results of each task on web track 2004.

With these features, we try two learning methods: SVM [7] and boosting [1], to do classification. We adopt the hierarchical methodology: first we classify all the queries into two categories TD and NP/HP, and then we classify the later one into two categories: NP and HP. We use the queries in web track 2003 as training samples. Our experiments on web track 2003 shows that the overall accuracy is more than 85% for the first hierarchy (TD and NP/HP).

TERABYTE TRACK

We use 6 machines, each of which has 2 CPU's, 4GB memory, and 4 200G IDE disks to index and search the GOV2 terabyte dataset. All HTML Pages are evenly distributed among the machines via URL hashing and every machine accommodates about 4M web pages, while all the non-html pages are discarded. It takes about 12 hours (for each machine) to index the data and most of the time is spent in parsing HTML pages. Since anchor text belongs to the pages the link points to, in processing and indexing the data, each machine needs to send anchor texts to corresponding machines. In the query phase, each query will be broadcasted to all the machines and the results are aggregated from them. It takes averagely about 1.5 seconds to answer a query using our system.
Most the techniques in Web track are tried in the terabyte track too, except for title extraction, HostRank and PageRank.

EXPERIMENTS

Table 6 summarized methods we used in final runs for mixed query task. In the first column, we describe both which ranking function and which fields are used for retrieval. For example, "BM25_whole" means that the ranking function of BM25 is applied on the whole page including anchor, title, meta and body fields. "MSRA1000" was proposed by us at TREC2003 [8], which weights term frequency of fields before applying BM25. "Proximity" is a function that considers the proximity between query term occurrences. "combotitle" refers to the combination of title and extracted title. Linear combination with special normalizations is used to produce the final scores. All coefficients are trained on the dataset of TREC2003. Average precision is our objected measure. Results on the training data is shown in Table 7 and official results on TREC2004 are shown in Table 8. We found our combination is stable. It achieves good performance on both datasets.

Some of our proposed approaches are effective to improve the performance of the TD/HP tasks or the NP task, but not both. Sitemap based feature propagation is a typical example. It worked as well as other runs for TD and HP task when it did not combine some useful scores such as URL score and proximity combotitle. However, this method works worse in terms of MRR for NP task. It is understandable since this method enhances features of a parent from its children and therefore it has an obvious homepage bias. But named pages are usually leaf pages in the site tree. So this method hurts NP queries while improving TD and HP queries.

Table 6. Official Results and Methods for Mixed Query Task

Methods	MSRAmixed1	MSRAx2	MSRAmixed3	MSRAx4	MSRAx5
BM25_whole		Y	Y		
MSRA1000_whole	Y			Y	Y
BM25_combotitle	Y	Y	Y	Y	
BM25_anchor	Y	Y	Y	Y	
URL score	Y	Y	Y	Y	
Proximity_body	Y	Y	Y	Y	
Proximity_combotitle	Y	Y	Y	Y	
Proximity_anchor	Y	Y	Y	Y	
HostRank	Y	Y	Y	Y	Y
Sitemap based feature propagation					Y

Table 7. Results on Training Data for Mixed Query Task

Run	2003.TD	2003.NP
MSRAmixed1	0.198	0.703
MSRAx2	0.199	0.703
MSRAmixed3	0.212	0.713
MSRAx4	0.200	0.700
MSRAx5	0.1965	0.6447

Table 8. Official Results for mixed query task

Run	TD			NP	HP	Overall			
	P@10	MAP	R-Pre	MRR	MRR	S@1	S@5	S@10	AveP
MSRAmixed1	0.2507	0.1780	0.2045	0.6720	0.7250	0.5333	0.8000	0.8800	0.5175
MSRAx2	0.2413	0.1775	0.2095	0.6723	0.7291	0.5200	0.7911	0.8756	0.5190
MSRAmixed3	0.2400	0.1685	0.1963	0.6684	0.6856	0.4889	0.7733	0.8622	0.5012
MSRAx4	0.2413	0.1751	0.2017	0.6846	0.7208	0.5156	0.7956	0.8711	0.5201
MSRAx5	0.2387	0.1751	0.1987	0.5810	0.7202	0.4756	0.7911	0.8711	0.4811

Table 9 and Table 10 show our official runs and results for query classification. MSRAQC2 and MSRAC3 use the boosting classification method, MSRAQC4, and MSRAQCSVM54 use the SVM classification method. MSRAC3 and MSRAQCSVM54 use all the feature (1-8 kinds) we extracted, and MSRAQC2 and MSRAQC4 only use some (1-6 kinds) of those features.. The overall accuracy of MSRAC3 and MSRAQCSVM54 is better than that of MSRAQC2 and MSRAQC4. This approves the effectiveness of these features we extracted: the more features we used, the better performance we get.

Table 9 Official Results and Methods for Query Classification Task

Run	Method
MSRAQC2	boosting with 1-6 kinds of features.
MSRAQC3	boosting with 1-8 kinds of features.
MSRAQC4	SVM with 1-6 kinds of features.
MSRAQC5	boosting for the first hierarchy (TD, NP/HP) and SVM for the second hierarchy (NP, HP) 1-8 kinds of features.
MSRAQCSVM54	SVM with 1-8 kinds of features.

For all of these runs, the TD category are best classified, no matter its precision, recall or F1 all overcome those of NP queries and HP queries. This is consistent with our prediction:

1. Most of the features (1-7 kinds) are for TD and NP/HP. That is, we only consider a binary classification problem at beginning.
2. We adopt a hierarchical methodology for this multiple categories classification. Considering that the first hierarchy is TD and NP/HP, the error in this hierarchy will propagate when we try to classify NP and HP in the second hierarchy. From this view, we may get better overall performance if we adopted some better methodology.

MSRAQC5, which uses boosting in the first hierarchy and SVM in the second hierarchy, gets the best result. Somehow it is interesting to us. We just want to try the performance of the mixture of boosting and SVM, and we do not predict this run performs the best.

Table 10. Official Results for Query Classification Task

Run	TD		NP		HP		Overall
	P/R	F	P/R	F	P/R	F	Accuracy
MSRAQC2	0.6806/0.6533	0.6667	0.5244/0.5733	0.5478	0.5070/0.4800	0.4932	0.5689
MSRAQC3	0.6533/0.6533	0.6533	0.5333/0.5333	0.5333	0.5467/0.5467	0.5467	0.5778
MSRAQC4	0.6479/0.6133	0.6301	0.4878/0.5333	0.5096	0.5694/0.5467	0.5578	0.5644
MSRAQC5	0.7727/0.6800	0.7234	0.5595/0.6267	0.5912	0.5333/0.5333	0.5333	0.6133
MSRAQCSVM54	0.6667/0.6933	0.6797	0.5467/0.5467	0.5467	0.5278/0.5067	0.5170	0.5822

For terabyte track, we have submitted 5 runs. In two of the five runs submitted, only title and body are considered. The other three runs use more fields. Please refer to Table 11 for details about the runs. Evaluation results are pending.

Table 11. Submitted Runs for Terabyte Track

Run.	Method
MSRAt1	MSRA1000_noanchor + Proximity_title + Proximity_body
MSRAt2	MSRA1000_noanchor + Proximity_title + Proximity_body (different coefficients)
MSRAt3	MSRA1000_whole
MSRAt4	MSRA1000 _whole + Proximity_title + Proximity_anchor + Proximity_body
MSRAt5	MSRA1000 _whole + Proximity_title + Proximity_anchor + Proximity_body + URL score

CONCLUSION

We analyzed the properties of queries of topic distillation, named page finding and homepage finding. And we agree on the comments in guidelines. Web task differ from adhoc in a number of ways: 1) homepage bias; 2) query by name and 3) TD topic selection. Our proposed approaches are exactly grouped into three corresponding to the differences. 1) HostRank and URL priors target at homepage bias; 2) Title extraction and URL relevance scoring target at query by name; and 3) Sitemap based feature propagation is consistent to the principle of TD topic selection. Given these individual approaches and some of them are biased to a type of queries and may hurt another type, it is tough to combine them together in proper way. Our solution is linear combination with special normalizations. The results indicate this strategy is effective and stable.

REFERENCES

1. Hastie, T., Tibshirani, R., and Friedman, J. The Elements of Statistical Learning, Springer, New York, 2001.
2. Kraaij, W., Westerveld, T., and Hiemstra, D., The Importance of Prior Probabilities for Entry Page Search. In proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02), 2002
3. Ogilvie, P., Callan, J., Combining Structural Information and the Use of Priors in Mixed Named-Page and Homepage Finding, in Text Retrieval Conference 2003, page 177
4. Page, L., PageRank: Bringing Order to the Web, Standford Digital Library Working Paper, 1997.
5. Robertson, S. E. and Walker, S. Okapi/Keenbow at TREC-8, In the Eighth Text REtrieval Conference (TREC 8), 1999, pp. 151-162.
6. Robertson, S. E. and Sparck Jones, K., *Relevance weighting of search terms*, Journal of the American Society of Information Science, Vol. 27, No. May-June, 1976, pp. 129-146.
7. Vapnik, V. Statistical learning theory. Wiley, 1998.
8. Wen, J.-R., Song, R., Cai, D., Zhu, K., Yu, S., Ye, S., Ma W.-Y., Microsoft Research Asia at the Web Track of TREC 2003, in Text Retrieval Conference 2003, page 408