Predicting Query Performance on the Web

No Author Given

Abstract. Predicting performance of queries has many useful applications like automatic query reformulation and automatic spell correction. However, accurate and effective performance prediction on the Web is a challenge. In particular, measures such as Clarity, that work well on homogeneous TREC like collections are not as effective on the Web. In this paper, we develop an effective and efficient approach for online performance prediction on the Web. We propose use of retrieval scores, and aggregates of the rank-time features used by the document-ranking algorithm to train regressors for query performance prediction. For a set of more than 12,000 queries sampled from the query logs of a major search engine, our approach achieves a linear correlation of 0.78 with DCG, and 0.52 with NDCG. Analysis of the prediction effectiveness shows that (i) *hard* queries are easier to identify while *easy* queries are harder to identify, (ii) NDCG, a non-linear effectiveness measure, is much harder to predict than DCG, and (iii) *long* queries' performance prediction is easier than prediction for *short* queries.

1 Introduction

Query performance prediction has been the focus of numerous research efforts in the past. Its importance is evidenced by its various applications. For example, Yom-Tov et al. [18] used query performance prediction to detect queries for which no relevant content exists in the document collection. Additionally, they used it to perform selective query expansion and merging of results in a distributed information retrieval system. Kumaran and Carvalho [14] choose between reduced versions of long queries using query performance predictors to improve retrieval effectiveness.

However, effective performance prediction techniques like Clarity [6], Coherence [13], and stability of the ranking [20], are computationally expensive as they involve post-processing of the search results. Further, these techniques focused on small and specialized environments like TREC collections, and are not as effective when applied to Web search [11].

In this paper, we propose a new query performance prediction technique that can be seamlessly incorporated into a web search engine's architecture (Section 3). Unlike previous *post-retrieval* techniques that use features computed by analyzing the top results returned by the query, our technique instead uses features readily available to the search engine's ranker: those that are used to rank documents in the first place. We perform a large-scale evaluation of query quality prediction on a set of 12,185 queries sampled from the query logs of a major search engine. The results (Section 5) show that our proposed technique performs better than Clarity, a state-of-the-art baseline, achieving a linear correlation of 0.78 with DCG and 0.52 with NDCG.

Typically, prediction techniques are evaluated using a single metric such as linear correlation or root mean squared error(RMSE) of the predictions. As we will elaborate in Section 6, these measures conceal how prediction accuracies vary for different queries – ranging from *hard* queries with useless results to *easy* queries with perfect results. Understanding distribution of prediction errors is critical from an application standpoint. For example, identifying hard queries is more important for automatic query reformulations whereas, identifying easy queries is more important for query expansion. In addition to evaluation using single metrics, we also analyze the effectiveness of our prediction technique at different effectiveness regions i.e., the effectiveness for finding *hard* and *easy* queries.

The extended analysis of results that we present in Section 6 yields further insights into the problem of predicting query performance. Interesting findings include, predicting *hard queries* is easy and finding *easy queries* is hard, NDCG prediction is harder than DCG prediction, and longer queries' performance is easier to predict compared to shorter queries.

The main contributions of this paper include:

- A novel, highly accurate, and efficient technique suitable for online query performance prediction.
- Web-scale prediction and evaluation on a large sample of real user queries.
- Analysis of prediction effectiveness that emphasize the need to optimize techniques for particular applications, rather than for general metrics such as RMSE or linear correlation.

2 Related Work

Pre-retrieval [10], [9] and post-retrieval measures [6], [13] have been proposed to predict query performance. While pre-retrieval techniques are more efficient than post-retrieval techniques, they are not as accurate. Post-retrieval techniques on the other hand, rely on analyzing the content of search results and are usually more accurate but less efficient. In contrast, our post-retrieval approach is both effective and efficient because we use rank-time features that are designed to capture document relevance.

Pre-retrieval. Pre-retrieval techniques do not analyze the search results. Instead, they use collection statistics of query terms such as inverse document frequency, collection term frequency, and variance of term weights, to predict query performance [12], [2], [19]. Avoiding retrieval and analysis of the search results improves prediction efficiency but it also limits prediction effectiveness, as the quality of the search results depends heavily on the retrieval algorithm.

Post-retrieval. Post-retrieval techniques retrieve search results for the given query, and then extract and analyze their contents to detect properties that indicate high performance. One of the most effective post-retrieval measure is Clarity. Clarity measures the divergence of the document language models from the background collection model [6]. Larger divergences indicate higher query performance. Clarity based measures work effectively on homogenous TREC collections but they are not as effective on Web collections [20], [11].

Other post-retrieval measures include, coherence of the search results [13], agreement between the top results of the full query and the top results of its sub-queries [18], and distribution of topic labels on search results [5]. In contrast, our approach does not depend on effective estimation of language models, similarity between result sets, or a labeling of Web documents.

Web Search. Zhou and Croft [21], highlight the diversity of Web queries, and search results as key challenges in predicting performance in Web search environments. To address these challenges they propose, weighted information gain, query feedback, and a rank sensitivity based measure. Hauff et al. [11], propose *Improved Clarity*, a measure that dynamically chooses the number of documents and the terms used for computing Clarity to address challenges on the Web. Both techniques demonstrate improved performance on the TREC[1] GOV2 collection. However, in the former case, conducting multiple retrievals and analyzing content of the retrieved documents is infeasible due to the small latencies tolerated on the Web, typically in the order of milliseconds. In the latter case, the estimation of language models can be less effective due to the diversity of Web search results and the relatively high proportion of off-topic information even in top ranked documents.

Retrieval score-based features have been used to predict query performance with mixed success [16],[17]. In particular, Grivolla et al. [8], show that mean, max and range of retrieval scores corresponding to the top ranked documents correlate well with average precision and can be used effectively to classify queries as *easy* or *hard*, on a collection of 50 TREC queries. We extend the retrieval score-based features to include measures that capture variance and include features that are used by the retrieval algorithm itself. Also, to the best of our knowledge, this is the first large scale evaluation and analysis of performance prediction on real Web queries.

3 Performance Prediction Overview

In this section, we describe our query performance prediction framework.

Performance Measures. We use discounted cumulative gain (DCG) and normalized discounted cumulative gain (NDCG) to measure query performance. DCG@k is the discounted cumulative gain at rank k calculated as $\sum_{r=1}^{k} \frac{2^{l(r)}-1}{log(1+r)}$, i.e., the sum of gains $2^{l(r)-1}$ at each rank r, discounted by an increasing function of the rank - $log(1+r)$. NDCG@k calculated as $\frac{1}{Z}\times$ DCG@k, normalizes the DCG@k by Z, which is defined as the best discounted gain at k that an optimal retrieval can achieve. Both DCG and NDCG are well suited for evaluating Web search engines as they can handle multiple levels of relevance.

[1] http://trec.nist.gov

Ranking Algorithm. We use LambdaRank [3], a machine learning algorithm that is well suited for learning to rank documents on the Web. Ranking models for the Web use a large number of features which makes it infeasible to directly optimize for retrieval metrics such as NDCG. Instead, LambdaRank computes smooth approximations to gradients of the retrieval metrics and this indirect optimization has been empirically shown to find the local optimum for the model parameters [7].

Intuition. Our approach is based on two observations. First, the LambdaRank scores of the top ranked documents are good estimators of retrieval effectiveness. Second, LambdaRank, combines query dependent, and query-independent document features that are designed to capture relevance. Query performance is intimately related to these feature values. Therefore, we use the retrieval scores and features used by LambdaRank for prediction.

However, retrieval scores for different queries may not be directly comparable. Using statistical aggregates of the scores helps to overcome this issue to some extent. In addition, statistical aggregates such as maximum, mean, and standard deviation capture different aspects of the quality of search results. For example, for queries with low performance, the retrieval scores tend to have low mean and high variance.

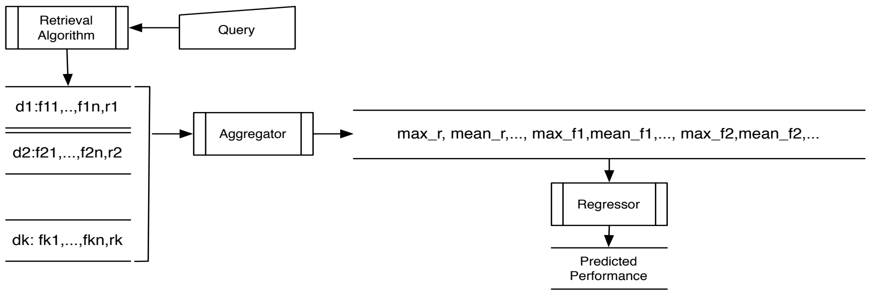


Fig. 1. Query Performance Prediction Framework.

Framework. Figure 1 gives a pictorial representation of our prediction framework. Given a query and a ranking algorithm (LambdaRank in this case), our learning approach is as follows. First, we use the ranking algorithm to score documents and rank them. For the top K ranked documents, we compute statistical aggregates such as mean, max, standard deviation, variance, and coefficient of dispersion for both the retrieval scores, and the retrieval features. Then, we use these aggregated features and the individual retrieval scores to train a regression model for predicting the target metric. We consider both linear and Random Forest-based non-linear regressors [15]. The linear regressors are trained to find feature weights which minimize RMSE of the predictions. The Random Forest regressors grow multiple regression trees by randomly sampling a subset of features from which the most informative feature is added to the tree iteratively. The regression trees are non-linear regressors themselves and hence, can capture non-linear relationships between the features and the predicted metric.

4 Experiments

We conduct experiments to predict DCG@5[2] and NDCG@5, on a collection of 12,185 queries that were sampled from the query logs of a major web search engine. For each query in this collection, we create feature vectors as follows. First, we use LambdaRank to assign scores and rank documents[3]. Our implementation uses several retrieval features such as BM25F-based features, click-based features, query length, and other query independent features such as static rank. For each of these retrieval features we create statistical aggregates. Next, we select the top 100 aggregates that have the highest linear correlation with the target metric on a set of training queries. We refer to these features as regression features henceforth. Finally, we create a query performance prediction dataset by associating with each query, the performance metric, DCG@5 or NDCG@5 and the regression features. Using this dataset, we conduct 3-fold cross-validation experiments to train linear as well as non-linear regressors based on the Random Forest algorithm[4].

We considered Clarity [6], which has been shown to be a competitive technique for query performance prediction, as an experimental baseline. Our implementation of Clarity uses a query model built from the top 100 results returned by the search engine. We build the query models from query-biased snippets rather than from the entire text of the documents. In addition to being efficient, it also helps create good quality query models by focusing on the relevant portion of the web pages and automatically filtering out layout information and advertisements. When compared to the features we use, clarity achieves very low correlation for both DCG and NDCG, as shown in Table 1. Therefore, we do not consider Clarity as a feature for further experiments.

Table 1. Correlation: *Average*, *Best* and *Worst* correspond to the average feature correlation, the highest and lowest correlation of the features used in our approach.

	Clarity	Average	Best	Worst
DCG	0.16	0.57	0.70	0.20
NDCG	0.09	0.27	0.50	0.17

5 Results

Table 2 shows the prediction accuracy in terms of linear correlation and root mean squared error (RMSE) for our approach. We can see that both predicted DCG and predicted NDCG values achieve a high linear correlation and low RMSE. Also, NDCG prediction is much harder as indicated by the low correlation and higher RMSE values. This is mainly because NDCG is a non-linear metric that is computed based on the actual number of relevant documents that exist in the collection. This information cannot be estimated based on the features of the top ranked documents alone. Finally, in terms of correlation and RMSE, there is little difference in prediction effectiveness between simple linear regression and the non-linear random forest based regression.

[2] We normalize DCG@5 by a constant(=perfect DCG@5) to scale values to (0,1).

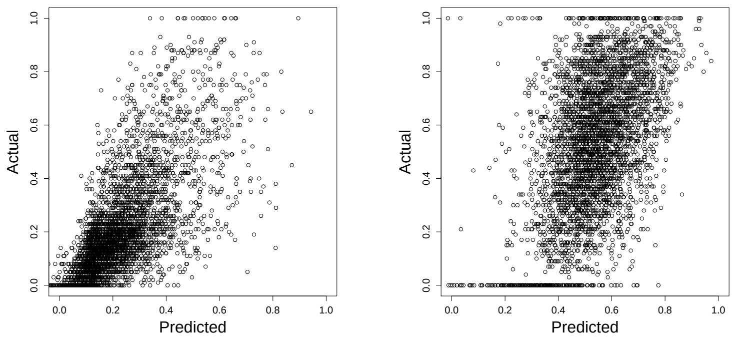
[3] We use an entirely different data set to train LambdaRank parameters.

[4] We used the *randomForest* package available from R with default parameters.

Table 2. Prediction Effectiveness for DCG and NDCG

Method	DCG		NDCG	
	Correlation	RMSE	Correlation	RMSE
Linear	0.78	0.13	0.50	0.23
Random Forest	0.79	0.13	0.52	0.22

The scatter plot in Figure 2(a) illustrate a strong correlation between the predicted and actual DCG values for one fold of the data. Figure 2(b) shows predicted NDCG values which are not as strongly correlated with the actual values. For DCG, when the predicted values are less than 0.2, the actual values are also less than 0.2 in most cases. On the other hand, when the predicted values are greater than 0.4 the actual values are more spread out. This suggests, DCG prediction is more precise for *hard* queries than for *average*, and *easy* queries. Similarly, NDCG prediction is highly precise when predicted values are below 0.3. However, prediction effectiveness degrades quickly when predicted values are greater than 0.4. Thus, for both DCG and NDCG, the high linear correlation and low RMSE values mask the rather poor effectiveness at the extremes.



(a) DCG (b) NDCG

Fig. 2. Linear Regression: Prediction versus Target Metrics for Test fold 1.

Feature Importance. We inspect the features used for the DCG and NDCG regression. Note that the features selected for DCG and NDCG can be different. We consider three subsets: features based on 1) *LambdaRank* scores, 2) *Click*-based features, and 3) *BM25F*-based features. Table 3 shows the predic-

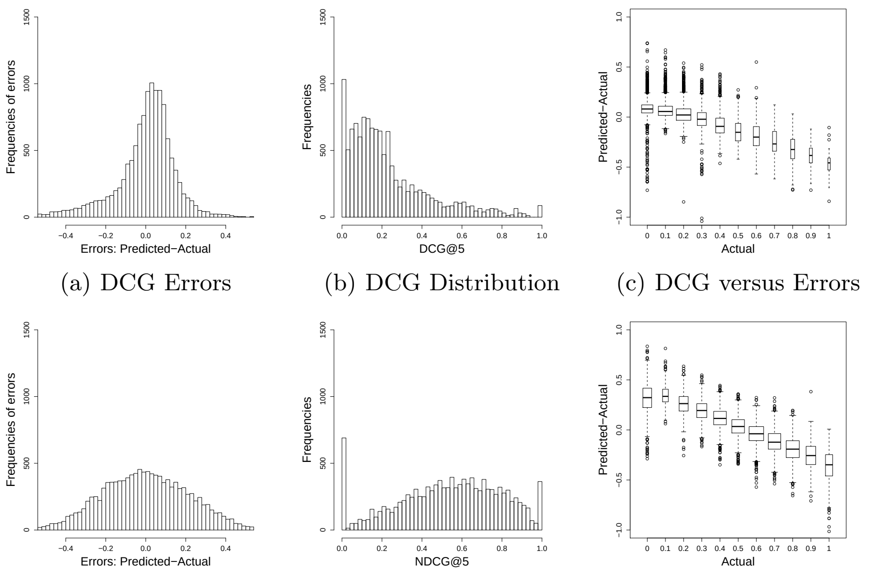
Table 3. Prediction Effectiveness of different feature groups.

Group	DCG		NDCG	
	Correlation	RMSE	Correlation	RMSE
LambdaRank	0.75	0.14	0.50	0.22
Click	0.78	0.13	0.41	0.24
BM25F	0.71	0.14	0.38	0.24
All	0.78	0.13	0.50	0.23

tion effectiveness of the different feature groups for linear regression. For DCG, all feature groups achieve high correlation while for NDCG, click and BM25F features are substantially lower compared to the combined features. Also, relative feature importance differs for DCG and NDCG. For instance, click features are more important for predicting DCG than LambdaRank features while, the relationship is reversed for NDCG. Click features are strong predictors of user preference [1],[4], and it is no surprise that they correlate well with DCG. However, NDCG being a non-linear metric, is harder to predict with click-based features alone. We hypothesize that since LambdaRank combines several features including click features and is trained to optimize for NDCG, the LambdaRank-based features turn out to be better predictors than click features. It is also interesting to note that the click features for DCG and LambdaRank features for NDCG are as effective as all the features combined. This suggests that more careful feature selection can reduce the run-time computations while retaining prediction effectiveness.

6 Analysis

In this section, we further analyze of the effectiveness of our prediction techniques. Unless otherwise stated, all analyses are based on the linear regression using the top 100 highly correlated features.



(a) DCG Errors (b) DCG Distribution (c) DCG versus Errors

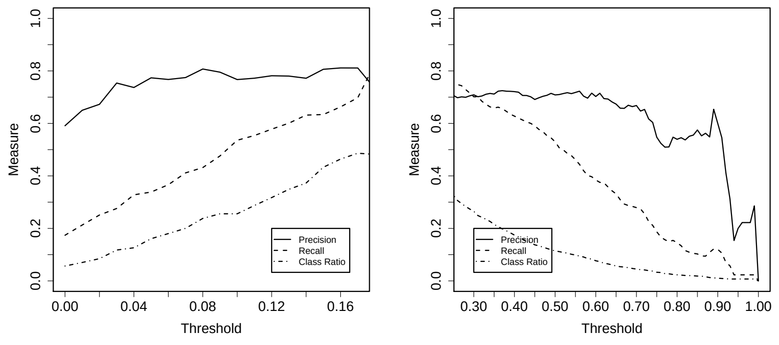
(d) NDCG Errors (e) NDCG Distribution (f) NDCG versus Errors

Fig. 3. Distribution of DCG and NDCG prediction errors.

Distribution of Errors. The distribution of DCG prediction errors is concentrated around zero as shown in Figure 3(a). In fact, around 80% of the errors are within 0.2 of the actual values. However, the errors span a relatively high range of values compared to the true distribution of DCG, shown in Figure 3(b). Figures 3(d) and (e) show corresponding distributions for NDCG. Nearly 80% of the errors are within 0.4 of the actual values. NDCG errors are more spread out compared to DCG errors because NDCG prediction is harder. Also, NDCG values are more evenly distributed, thus increasing the range of possible errors.

Figure 3(c) shows that 1) most of the prediction errors are small for hard queries and 2) errors increase for easy queries. For example, for queries with DCG< 0.1 most errors are less than 0.1 but for queries with DCG> 0.5, most errors are above 0.4. On the other hand, NDCG errors are higher at both extremes, as shown in Figure 3(f). The distribution of errors show that despite the high values for the average metrics, prediction effectiveness for *hard* and *easy* queries needs further improvement.

Effectiveness Regions. Even though metrics such as linear correlation and RMSE are useful for training and evaluation, from an application standpoint these average metrics are not adequate. For example, reliable identification of *easy* queries is useful for selective application of pseudo-relevance feedback [2] whereas, reliable identification of *hard* queries is more useful for enabling query reformulations. In order to highlight prediction effectiveness for different types of queries, we formulate two tasks: 1) Identifying hard queries - queries whose target metric is below a specified threshold, and 2) Identifying easy queries - queries whose target metric is above a specified threshold. Using standard precision and recall metrics, we can then focus on the type of queries that are most useful for the application of interest.

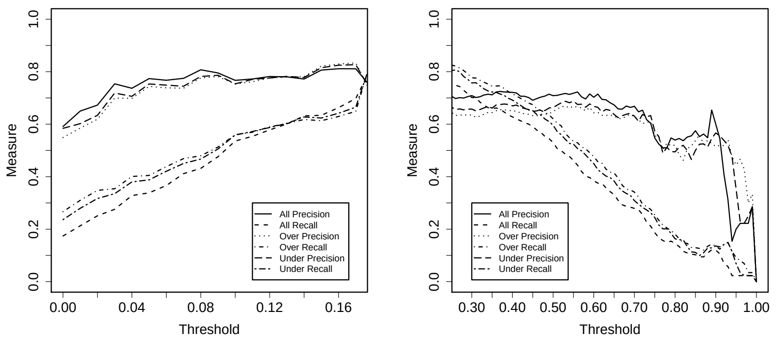


(a) DCG: Hard Queries (b) DCG: Easy Queries

Fig. 4. Prediction accuracy at different effectiveness regions: Precision, Recall and Class Ratio for identifying (a) Hard queries - queries whose actual DCGs are less than a specified threshold (x - axis), and (b) Easy queries - queries whose actual DCGs are greater than a specified threshold (x - axis).

Figure 4(a) shows precision and recall values for the task of identifying hard queries for varying thresholds. Hard queries are identified with high precision but with low recall. For example, for the task of predicting queries with DCG≤ 0.08, the precision is nearly 80% but the recall is less than 50%. Further, as the threshold increases, the task becomes progressively easier, and consequently, both precision and recall improve. For finding *easy* queries, precision is much lower for finding the most *easy queries.* Note that in Figure 4(b) as threshold increases, the task becomes progressively harder. Nonetheless, precision drops more dramatically i.e., it is much harder to identify easy queries reliably.

Sampling. Figures 4(a) and 4(b) show that the recall curve closely follows



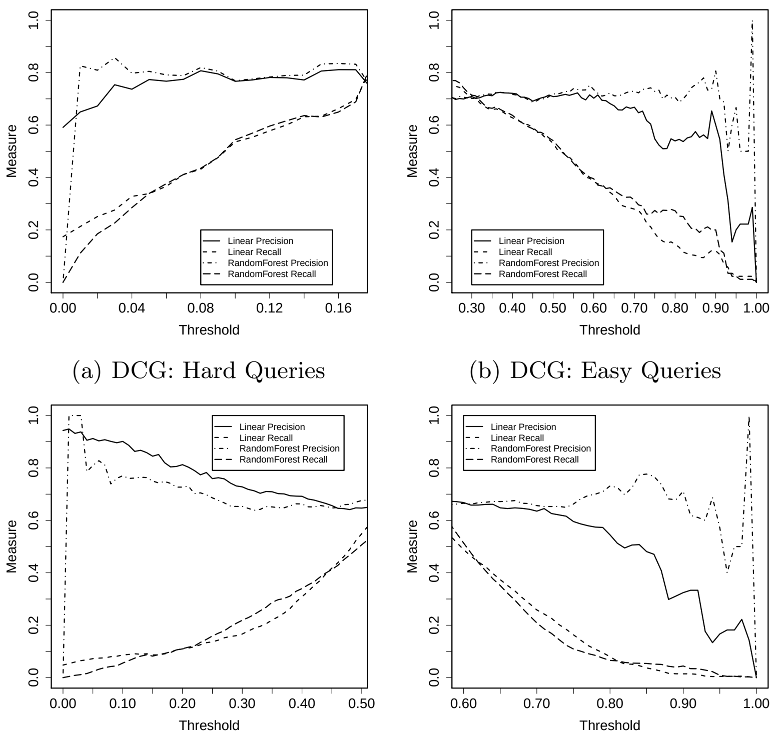
(a) DCG: Hard Queries (b) DCG: Easy Queries

Fig. 5. Impact of Sampling on Linear Regression.

the class ratio − the ratio of positive to negative instances. Less than 10% of the queries have DCG$>$ 0.5 and less than 10% of the queries have DCG$<$ 0.04. Consequently, more queries in the middle region are correctly identified, since the regression focuses on minimizing RMSE, an average metric. This suggests that improving the sample ratio of the extremes i.e., the hard and easy queries, can improve their prediction accuracies. To illustrate the benefits of sampling, we consider two sampling strategies: (a) over sampling of *extremes* - queries with DCG$<$ 0.03 or DCG$>$ 0.5 and (b) under sampling of *middle* region - those with DCG between 0.03 and 0.5.

As shown in Figure 5, both over sampling of the extremes and the under sampling of middle region lead to improved recall with only a small loss in precision, for the task of identifying hard queries. For example, over sampling extremes leads to a nearly 10% absolute increase in recall with less than 3% absolute loss in precision for finding queries with DCG$<$ 0.08. We see a similar trend for identifying easy queries. Although not reported here, corresponding NDCG sampling experiments did not show any improvements.

Linear vs. Non-Linear Regression. Figures 6(a)-6(d) compare the effectiveness of linear and non-linear regressions. For both DCG and NDCG prediction, using a non-linear regressor improves prediction accuracies. Specifically, easy queries prediction improves dramatically, with a relatively small loss in precision for hard queries prediction. We found that some regression features are not monotonically related to the target metrics. In particular, NDCG features that are positively correlated for hard queries (with NDCG < 0.3) are actually negatively correlated for the easy queries (with NDCG > 0.7). We hypothesize



(a) DCG: Hard Queries (b) DCG: Easy Queries

(c) NDCG: Hard Queries (d) NDCG: Easy Queries

Fig. 6. Linear versus Non-linear prediction.

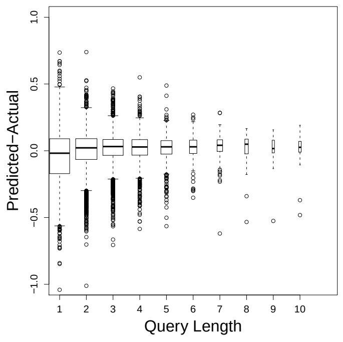
that the random forest regression is better able to handle this non-linearity in feature correlation and hence the improved performance for easy queries.

Query Length. Predicting performance for long queries is easier than for shorter queries. Figure 7(a) shows a box plot of the distribution of DCG prediction errors for queries of different lengths (number of words). As query length increases the range of DCG prediction errors decreases[5]. This is mainly because query length is an important feature in our regression. Query length has a strong negative correlation with retrieval effectiveness, i.e., retrieval is less effective for longer queries compared to shorter keyword queries, as shown in Figure 7(b). For NDCG, Figure 7(c) shows that prediction errors do not decrease for longer queries. This is because as shown in Figure 7(d) the range of NDCG values for long queries is much larger than the range of DCG values, even though the average NDCG values decrease as query length increases.

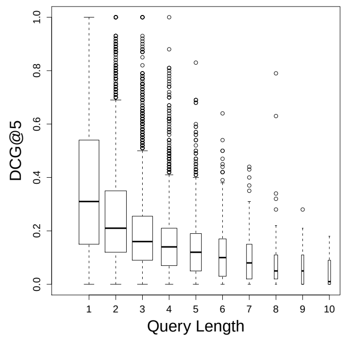
7 Conclusions

In this paper, we describe an effective and efficient Web query performance prediction technique that uses retrieval scores and retrieval features. Our approach

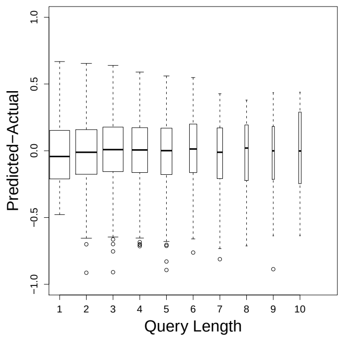
[5] The number of queries decrease for increasing query lengths but we note that more than 10% of the queries have length 5 or more.



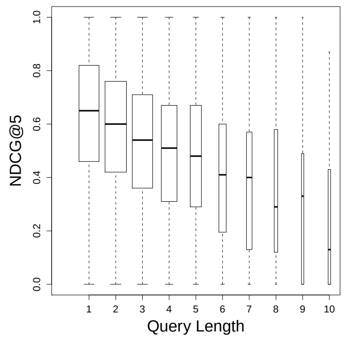
(a) DCG Errors



(b) Query Length vs DCG



(c) NDCG Errors



(d) Query Length vs NDCG

Fig. 7. DCG and NDCG prediction errors distribution for queries of different lengths.

outperforms traditional measures such as Clarity. Prediction effectiveness varies significantly for different types of queries and specifically, prediction for queries with very high or very low performance is hard. Single metrics such as linear correlation and RMSE do not adequately capture this variance. We propose evaluation of prediction techniques using classification tasks that reflect applications of performance prediction. Our classification based evaluation reveals that adjusting query distributions using sampling improves prediction at the extremes and using non-linear regressors improves prediction of *easy* queries.

References

1. E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR '06: 29th Annual ACM SIGIR Conference Proceedings*, pages 19–26, 2006.
2. G. Amati, C. Carpineto, G. Romano, and F. U. Bordoni. Query difficulty, robustness, and selective application of query expansion. In *ECIR '04: Proceedings of the 26th European Conference on Information Retrieval*, pages 127–137, 2004.
3. C. Burges, R. Ragno, and Q. Le. Learning to rank with nonsmooth cost functions. *Advances in Neural Information Processing Systems*, 19:193, 2007.

4. B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. *Advances in Neural Information Processing Systems*, 20:217–224, 2008.
5. K. Collins-Thompson and P. N. Bennett. Estimating query performance using class predictions. In *SIGIR '09: 32nd International ACM SIGIR Conference Proceedings*, pages 672–673, 2009.
6. S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *SIGIR '02: 25th Annual ACM SIGIR Conference Proceedings*, pages 299–306.
7. P. Donmez, K. M. Svore, and C. J. Burges. On the local optimality of lambdarank. In *SIGIR '09: 32nd Annual ACM SIGIR Conference Proceedings*, pages 460–467.
8. J. Grivolla, P. Jourlin, and R. de Mori. Automatic classification of queries by expected retrieval performance. In *SIGIR '05: Proceedings of the Workshop on Predicting Query Difficulty - Methods and Applications*, volume 5, 2005.
9. C. Hauff, L. Azzopardi, and D. Hiemstra. The combination and evaluation of query performance prediction methods. In *ECIR '09: Proceedings of the 31th European Conference on Information Retrieval*, pages 301–312. Springer-Verlag, 2009.
10. C. Hauff, D. Hiemstra, and F. de Jong. A survey of pre-retrieval query performance predictors. In *CIKM '08: Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 1419–1420, 2008.
11. C. Hauff, V. Murdock, and R. Baeza-Yates. Improved query difficulty prediction for the web. In *CIKM '08: Proceedings of the 17th ACM conference on Information and Knowledge Management*, pages 439–448, 2008.
12. B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *The Eleventh Symposium on String Processing and Information Retrieval*, pages 43–54. LNCS, Springer, 2004.
13. J. He, M. Larson, and M. de Rijke. Using coherence-based measures to predict query difficulty. In *ECIR '08: Proceedings of the 30th European Conference on Information Retrieval*, pages 689–694. Springer, Springer, April 2008.
14. G. Kumaran and V. R. Carvalho. Reducing long queries using query quality predictors. In *SIGIR '09: 32nd International ACM SIGIR Conference Proceedings*, pages 564–571, 2009.
15. A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.
16. C. Piatko, J. Mayfield, P. McNamee, and S. Cost. JHU/APL at TREC 2004: Robust and terabyte tracks. In *In Proceedings of TREC 2004*.
17. S. Tomlinson. Robust, Web and Terabyte Retrieval with Hummingbird Search-Server TM at TREC 2004. In *Proceedings of TREC 2004*.
18. E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *SIGIR '05: 28th Annual ACM SIGIR Conference Proceedings*, pages 512–519, 2005.
19. Y. Zhao, F. Scholer, and Y. Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In *ECIR '08: Proceedings of the 30th European Conference on Information Retrieval*, pages 52–64. Springer.
20. Y. Zhou and W. B. Croft. Ranking robustness: a novel framework to predict query performance. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and Knowledge Management*, pages 567–574, 2006.
21. Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In *SIGIR '07: 30th Annual ACM SIGIR Conference Proceedings*, pages 543–550, 2007.