# Incorporating Query Difference for Learning Retrieval Functions in Information Retrieval

[Extended Abstract]

Hongyuan Zha
Department of CSE
Pennsylvania State University
University Park, PA 16802
zha@cse.psu.edu

Zhaohui Zheng
Yahoo Inc.
701 First Avenue
Sunnyvale, CA 94089
zhaohui@yahoo-inc.com

Haoying Fu, Gordon Sun
Yahoo Inc.
701 First Avenue
Sunnyvale, CA 94089
gzsun@yahoo-inc.com

## ABSTRACT

We discuss information retrieval methods that aim at serving a diverse stream of user queries. We propose methods that emphasize the importance of taking into consideration of query difference in learning effective retrieval functions. We formulate the problem as a multi-task learning problem using a risk minimization framework. In particular, we show how to calibrate the empirical risk to incorporate query difference in terms of introducing nuisance parameters in the statistical models, and we also propose an alternating optimization method to simultaneously learn the retrieval function and the nuisance parameters. We illustrate the effectiveness of the proposed methods using modeling data extracted from a commercial search engine.

## Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Retrieval functions*; H.4.m [**Information Systems**]: Miscellaneous—*Machine learning*

## General Terms

Algorithms, Experimentation, Theory

## Keywords

relevance, retrieval function, machine learning, query dependence, least-squares regression, regularization

## 1. PROBLEM FORMULATION

We consider $\mathcal{D}$, the set of all the documents in consideration, $\mathcal{L}$ the set of labels which can be either a finite or infinite set, and $\mathcal{Q}$ the set of all potential user queries. We model each query $q \in \mathcal{Q}$ as a probabilistic distribution $P_q$ over $\mathcal{D} \times \mathcal{L}$, $P_q(d, \ell), \quad d \in \mathcal{D}, \quad \ell \in \mathcal{L}$ which specifies the probability of document $d$ being labeled as $\ell$ under query $q$. Now we define a loss function $L$ over the set $\mathcal{L} \times \mathcal{L}$, $L : \mathcal{L} \times \mathcal{L} \mapsto \mathcal{R}_+^1$, the set of nonnegative real numbers, and we also specify a class of functions $\mathcal{H}$ from which the retrieval function will be extracted, where for $h \in \mathcal{H}$, $h : \mathcal{Q} \times \mathcal{D} \mapsto \mathcal{L}$.


*SIGIR'06*, August 6–11, 2006, Seattle, Washington, USA.
ACM 1-59593-369-7/06/0008.

For each query $q$, we can then specify a learning problem (classification or regression problem): find $h_q^* \in \mathcal{H}$ such that

$$h_q^* = \arg\min_{h \in \mathcal{H}} \mathcal{E}_{P_q} L(\ell, h(q, d)).$$

The goal of information retrieval, is to learn a retrieval function $h^*$ that will be good for all the queries $q \in \mathcal{Q}$. Therefore, we need to deal with potentially *infinite* number of *related* learning problems, each for one of the query $q \in \mathcal{Q}$. To this end, we specify a distribution over $\mathcal{Q}$: $P_Q(q)$ can indicate, for example, the probability that a specific query $q$ is issued to the information retrieval system which can be approximated. Then the optimization problem we need to solve is for the combined risk,

$$\min_{h \in \mathcal{H}} \mathcal{E}_{P_Q} \mathcal{E}_{P_q} L(\ell, h(q, d)) \tag{1}$$

In practice, we sample a set of queries $\{q_i\}_{i=1}^Q$ from the distribution $P_Q$, and for each query $q$, we also sample a set of documents from $\mathcal{D}$ for labeling to obtain

$$\{d_{qj}, l_{qj}\}, \quad q = 1, \ldots, Q, \ j = 1, \ldots, n_q$$

where $l_{qj} \in \mathcal{L}$ are labels obtained from human judges for example after relevance assessment. Ideally, the sampling of the queries should be according to $P_Q$ and the sampling of the documents for each query $q$ should be according to $P_q$, the former is relatively easy to do while the later is a much more difficult issue. The optimization problem for the empirical counterpart of (1) is

$$\min_{h \in \mathcal{H}} \sum_{q=1}^{Q} \sum_{j=1}^{n_q} L(\ell_{qj}, h(q, d_{qj})) + \lambda\, \Omega(h),$$

where we also add a regularization term to control the complexity of $h$ with $\Omega(h)$ measuring the complexity of $h$, and $\lambda$ is the regularization parameter that balances the fit of the model in terms of the empirical risk and the complexity of the model.

## 2. INCORPORATING QUERY DIFFERENCE IN THE EMPIRICAL RISK

To be concrete, we consider the risk minimization problem in the context of regression, i.e., we assume the labels are real numbers, and to be consistent with convention, we will use $y_{qj}$ to denote the label $\ell_{qj}$. We assume the labeled set is represented as $\{x_{qj}, y_{qj}\}, \quad q = 1, \ldots, Q, \ j = 1, \ldots, n_q,$

here $x_{qj}$ denote the feature vector for the query-document pair $\{q, d_{qj}\}$. To this end, we seek to find a function $h \in \mathcal{H}$ to minimize the following empirical risk,

$$L(h) = \sum_{q=1}^{Q} \sum_{j=1}^{n_q} (y_{qj} - h(x_{qj}))^2.$$

To incorporate query-dependent effects, we can consider the following modified empirical risk, and we seek to find $h$ and $g_q, q = 1, \ldots, Q$, to minimize

$$L(h, g) = \sum_{q=1}^{Q} \sum_{j=1}^{n_q} [y_{qj} - g_q(h(x_{qj}))]^2, \quad (2)$$

where $g_q(\cdot)$ is a general monotonically increasing function, and $g = [g_1, \ldots, g_Q]$, i.e., we seek

$$\{h^*, g^*\} = \text{argmin}_{\{h \in \mathcal{H}, g \text{ mono increasing}\}} L(h, g).$$

The intention is that the function $g_q$ incorporate the difference for queries when using $h(x)$ to predict the label $y$. From another viewpoint, for suitably chosen $g_q$, we seek to find $h(x)$ to match $g_q^{-1}(y)$. In this work we focus on the simple case where $g_q(\cdot)$ is a linear function, i.e., $g_q(x) = \beta_q + \alpha_q x, \quad q = 1, \ldots, Q$ with $\alpha_q \geq 0$.

As we mentioned before, to control the size of the parameters $\beta_q, \alpha_q$ and the complexity of $h$, we also need to add regularization terms to the modified empirical risk (2) to obtain the regularized empirical risk

$$L(h, \beta, \alpha) = \sum_{q=1}^{Q} \sum_{j=1}^{n_q} [y_{qj} - \beta_q - \alpha_q h(x_{qj})]^2 +$$

$$+\lambda_\beta \|\beta\|_p^p + \lambda_\alpha \|\alpha\|_p^p + \lambda_h \Omega(h),$$

where $\beta = [\beta_1, \ldots, \beta_Q]$ and $\alpha = [\alpha_1, \ldots, \alpha_Q]$, and $\lambda_\beta, \lambda_\alpha$ and $\lambda_h$ are regularization parameters, and $\| \cdot \|_p$ is the $p$ norm of a vector. We will only consider the case for $p = 1$ or $p = 2$. In summary, we seek to find

$$\{h^*, \beta^*, \alpha^*\} = \text{argmin}_{h \in \mathcal{H}, \beta, \alpha \geq 0} L(h, \beta, \alpha). \quad (3)$$

In general, we will not impose a parametric form for the function $h$, and we will employ the methodology of coordinate descent (alternating optimization) to solve the optimization problem (3). Specifically, we will alternate between optimizing against $h$ and optimizing against $\beta$ and $\alpha$. The regularization parameter $\lambda_h$ will be determined during the nonlinear regression process for finding $h$ discussed below while regularization parameters $\lambda_\beta$ and $\lambda_\alpha$ will be determined by cross-validation. In what follows, we will refer the above algorithm as *adaptive Target Value Tranformation* (aTVT). We will compare aTVT against a particular system that uses nonlinear regression to learn a retrieval function based on the gradient boosting methods.

## 3. EXPERIMENTAL RESULTS

In this section we report some experimental results on data generated from a commercial search engine. In our experiments, a set of queries are sampled from query logs, and a certain number of query-document pairs are labeled according to their perceived relevance judged by human editors. The labels are mapped to numerical values and the goal is to learn a retrieval function that can best mimic the human judgment process. In Table 1, we list some basic statistics for the two data sets.

**Table 1: Number of queries and query-url pairs on US and CN datasets**

| | # queries | # total query-document pairs |
|---|---|---|
| Chinese | 2649 | 78188 |
| English | 910 | 70742 |

**Table 2: The dcgs and percentage dcg increases of retrieval function with aTVT over without aTVT on the English data set, and $p$ values for different regularization parameters: $\lambda_\alpha$ and $\lambda_\beta$ in the $L_2$ case. Notice that the *dcg* for retrieval function without aTVT are 11.30.**

| | $\lambda_\beta$=1 | 10 |
|---|---|---|
| $\lambda_\alpha$=1 | 11.48(+1.55%,0.01) | 11.49(+1.68%,0.006) |
| 10 | **11.53(+1.98%,0.002)** | 11.52(+1.88%,0.002) |
| 50 | 11.50(+1.75%,0.002) | 11.51(+1.79%,0.008) |
| 100 | 11.51(+1.83%,0.007) | 11.49(+1.65%,0.003) |

In this work, we use the recently popularized Discounted Cumulative Gain (DCG) methodology which seems to be more appropriate for assess relevance in the context of search engines. For a ranked list of $N$ documents, we use the following variation of DCG,

$$\text{DCG}_N = \sum_{i=1}^{N} \frac{G_i}{\log_2 (i+1)},$$

where $G_i$ represents the weights assigned to the label of the document at position $i$. We will use the symbol *dcg* to indicate the average of this value over a set of queries in our experiments.

Table 2 lists the *dcg* for retrieval function with aTVT as compared to retrieval function without aTVT in the $L_2$ case for the English data set (similar results hold for the chinese data set). The percentage dcg gains and the $p$-values from Wilcoxon signed rank tests are also presented. From thetable, we can see aTVT gives statistically significant dcg gains. The optimal regularization parameter combinations give about 2% *dcg* gain for the English data.

## 4. CONCLUDING REMARKS

There are many ways to incorporate query difference in learning retrieval functions, the approaches of constructing appropriate query features being one of them even though it is usually not looked at from this viewpoint. In this paper, we present an approach through modifications of the empirical risk using nuisance parameters to accommodate the effects of query difference. The approach can be used even when there are query features contained in the feature vector of query-document pairs.