Analysis of Domain Specification in Ranking Model Adaptation

Shanthi Sri Regulagadda[1] and T. Ashok Kumar [2]

1. M.Tech., Department of Computer Science & Engineering, Chirala Engineering College, Prakasam District, Andhra Pradesh, India.
2. Associate Professor, Department of Computer Science, Chirala Engineering College, Prakasam District, Andhra Pradesh, India

Abstract: In the market, various domain - specific search engines emerged, which are restricted to specific topicalities or document formats, and vertical to the broad based search. Simply applying the ranking model trained for the broad - based search to the verticals cannot achieve a sound performance due to the domain differences, while building different ranking models for each domain is both laborious for Labeling sufficient training samples and time consuming or the training process. In this paper, to address the above difficulties, we investigate two problems (1) Whether we can adapt the ranking model learned for existing Web page search or verticals, to the new domain, so that the amount of labeled data and the training cost is reduced, while the performance requirement is still satisfied (2) How to adapt the ranking model from auxiliary domains to a new target domain. We address the second problem from the regularization framework and an algorithm called ranking adaptation SVM is proposed. The results demonstrate the applicability's of the proposed ranking model adaptation algorithm and the ranking adaptability measurement. In this paper we built a prototype application that demonstrates ranking model adaption using a novel ranking model meant for ranking the search results besides adapting to new domains. The experimental results revealed that the proposed application is useful in searching data across the domains.

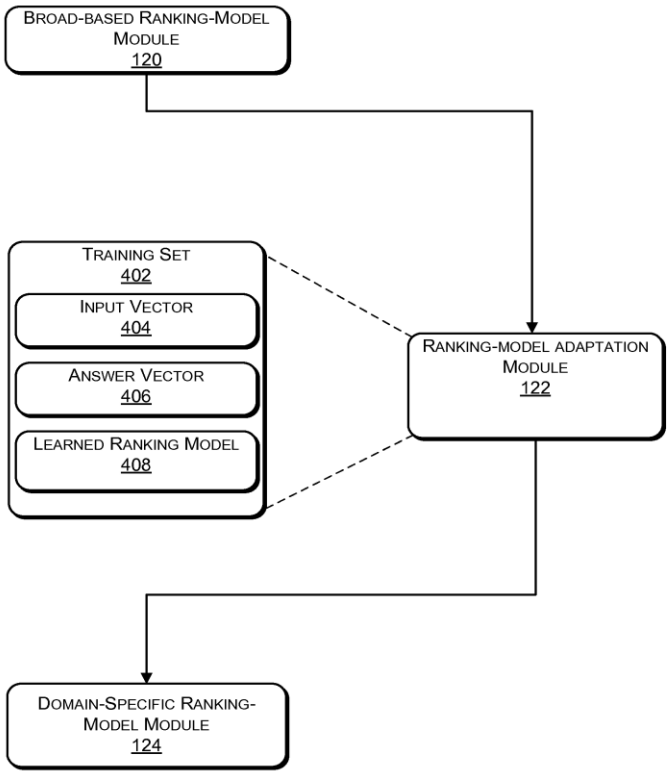
Keywords: Search engine, Ranking Models, Framework, domain specific search, Support Vector Machines, Domain Adaptation.

Introduction:

We consider the ranking problem for Information Retrieval (IR), where the task is to order a set of results (documents, images or other data) by relevance to a query issued by a user. Ranking is a core technology that is fundamental to widespread applications such as internet search and advertising, recommender systems, and Social networking systems. In this paper, we propose a new ranking algorithm that combines the strengths of two previous approaches Lambda Rank and boosting. Ranking has been shown to be a very effective ranking a logarithm for optimizing IR measures. It leverages the fact that neural net training needs only the gradients of the cost function, not the function values themselves, and it models those gradients using the sorted positions of the documents for a given query. This bypasses two significant problems, namely that typical IR measures. Domain specific search engines are becoming increasingly popular because they increased accuracy and extra features not possible with general, Web-wide search engines. Unfortunately, they are also desalt and time consuming to maintain. This paper proposes the use of machine learning techniques to greatly automate the creation and maintenance of domain-specific search engines. We describe new research in enforcement learning, text classification and information extraction that enables exigent speeding, populates topic hierarchies, and ideates informative text segments. Using these techniques, we have built a demonstration system. Domain-

specific Web search engines are effective tools for reducing the difficulty experienced when acquiring information from the Web. Existing methods for building domain-specific Web search engines require human expertise or specific facilities. However, we can build a domain-specific search engine simply by adding domain-specific keywords, called "keyword spices," to the user's input query and forwarding it to a general-purpose Web search engine. Keyword spices can be effectively discovered from Web documents using machine learning technologies. The paper describes domain-specific Web search engines that use keyword spices for locating recipes, restaurants. It is well known that the popular search engine Google uses PageRank (1) as its core algorithm for ranking documents. PageRank is a very successful algorithm, because it associates document as being important based on who links to it and the importance of those links. However, sometimes Page Rank isn't enough. In clothing, links can imply relation (such as recommended items), but it is not always the case. In addition, it is difficult to define the importance of a specific article of clothing. What makes one better than another? It is very much a matter of taste. Therefore,it is important to use context. Context is approached in this system using natural language processing techniques such as giving weight to domain related features and using part of speech to find the focus of the query. Section 2 lists the components of the project, Section 3 discusses websites that provide clothing search, Section 4 describes the data and resources used, Section 5 explains the ranking methods, Section 6 evaluates the system, section 7 discusses the feasibility of the project, and section 8 discusses future work.

Ranking Model Adaptation: Inspired by the linear regression based model adaptation methods in speech recognition [15, 18], we propose a general framework to perform ranking model adaptation. We assume that a global ranking model is trained based on a large user-independent training set. For each user, an adapted model is obtained by applying a set of learned linear trans-formations, e.g., scaling and shifting, to the parameters of the global model based on each individual user's adaptation data, e.g., query with corresponding clicks. In the following discussions, we first describe our general framework of ranking model adaptation, and then we take three frequently used learning to rank algorithms, i.e., RankNet [4], LambdaRank [22] and RankSVM [16], as examples to demonstrate the detailed procedures of applying the proposed adaptation framework.



Learning to rank" approach enables a ranking model to be automatically constructed for a broad-based search engine based upon training data. The training data represents documents returned to a search query and are labeled by humans according to the relevance with respect to the query. The purpose of the broad-based search engine ranking model is to rank the data from various domains in a way that is similar to rankings in the training data. Applying ranking models constructed for use with a broad-based search engine to a domain-specific environment may present many obstacles. For example, the broad-based ranking models are typically built upon data retrieved from multiple domains, and can therefore be difficult to adapt for a particular domain with special search intentions. Alternatively, creating a ranking model specific to each of the various specific domains available on the web is both time consuming and computational expensive as the resources required for such an undertaking would be considerable

In the setting of the proposed ranking adaptation, both the number of queries m and the number of the returned documents n(qi) in the training set are assumed to be small. They are insufficient to learn an effective ranking model for the target domain. However, an auxiliary ranking model fa, which is well trained in another domain over the labeled data Qa and Da, is available. It is assumed that the auxiliary ranking model fa contains a lot of prior knowledge to rank documents, so it can be used to act as the base model to be adapted to the new domain. Few training samples can be sufficient to adapt the ranking model since the prior knowledge is available.

Ranking Adaptability : Though the ranking adaptation can mostly provide benefits for learning a new model, it can be argued that when the data from auxiliary and target domains share little common knowledge, the auxiliary ranking model can provide little help or even negative influence, to the ranking of the documents in the target domain. Consequently, it is imperative to develop a measure for quantitatively estimating the adaptability of the auxiliary model to the target domain. However, given a ranking model and a dataset collected for a particular target domain, it's nontrivial to measure their correlations directly, because neither the distribution of the ranking model nor that of the labeled samples in the target domain is trivial to be estimated. Thus, we present some analysis on the properties of the auxiliary model, based on which the definition of the proposed *ranking adaptability* is presented. Based on the above analysis of fa, we develop the *ranking adaptability* measurement by investigating the correlation between two ranking lists of a labeled query in the target domain, i.e., the one predicted by fa and the ground-truth one labeled by human judges. Intuitively, if the two ranking lists have high positive correlation, the auxiliary ranking model fa is coincided with the distribution of the corresponding labeled data, therefore we can believe that it possesses high ranking adaptability towards the target domain, and vice versa. This is because the labeled queries are actually randomly sampled from the target domain for the model adaptation, and can reflect the distribution of the data in the target domain.

Framework: For a given set of queries $Q^u = \{q^{u1}, q^{u2}, \ldots, q^{um}\}$ from user *u*, each query q_i^u is associated with a list of document-labes pairs $\{(x^u_{i1}, y^u_{i1}),(x^u_{i2}, y^u_{i2}), \ldots ,(x^u_{in}, y^u_{in})\}$ where x^u_{ij} denotes a retrieved document represented by a V-dimensional vector of ranking features, and y^u_{ij} is the corresponding relevance label indicating if the document x^u_{ij} is relevant to user u (e.g., clicks).

Since our focus of this work is on user-level ranking model adaptation, in the following discussions we ignore the superscript u to make the notations concise when no ambiguity is involved. A ranking model f is defined as a mapping from a document x_{ij} to its ranking scores ij, i.e.,f:xij→sij, such that when we order the retrieved documents for query q by f, a certain ranking metric, .g., an average precision(MAP) or precision at k (P@k) [2], is optimized. Such rank-ing model can be anually set, or estimated by an automatic algorithm based on a collection of annotated queries [19]. In this work, we focus online arran king models, which can be characterized by a parametric form of linear combination of ranking features, i.e., f(x) =wTx, where w is the linear coefficients for the corresponding an king features. Denoting fs(x) =ws Txas the given global ranking model estimated in a user-independent annex, the adaptation offs(x) for each individual user is performed via the line artrans formations defined by V×(V+1) dimensional matrix Au, by which three linear operations, i.e., scaling, shifting and rotation, can encoded. More precisely

$f^u(x) = (A^{\hat{u}} w^s)^T x$

There are two major considerations in designing such a transformation matrix A^u . First, a full transformation matrix has the number of O(V2) free parameters, which is redundant and even larger than the number of parameters needed to estimate a new ranking model for each user (i.e., O(V)). As a result, it is infeasible for us to estimate a fulltransformation matrix for every user. To reduce the size of free parameters in A u , we decide to only focus on the scaling and shifting operations for adapting the parameters in fs(x). This reduces the size of free parameters in A u from O (V^2) to O (V). Second, a more important consideration is how to alleviate the problem of sparse observation of ranking features in the limited adaptation data. Because some advanced ranking features used in modern search engines, e.g., topic category of documents, might not be trigged in the scattered adaptation queries, one will encounter missing feature values. In order to properly update the parameters for unseen features during adaptation, we organize the features in groups and share the same shifting and scaling transformations to the parameters within the same group. Based on the above considerations, we design the transformation matrix A^u to be the following specific form,

$$\mathbf{A}^u = \begin{pmatrix} a^u_{g(1)} & 0 & \cdots & b^u_{g(1)} \\ 0 & a^u_{g(2)} & \cdots & b^u_{g(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & a^u_{g(V)} & b^u_{g(V)} \end{pmatrix}_{V\times(V+1)}$$

Where g (·) is a feature grouping function, which maps V original ranking features to K different groups, a^u_k and b^u_k denote the scaling and shifting operations applied to the linear coefficients ws of the source model f s (x) in group k. As a result, Eq (1) can be realized as

$$f^u(x) = \sum_{k=1}^{K} \sum_{g(i)=k} (a^u_k w^s_i + b^u_k) x_i$$

The grouping function g (·) defines the transformation sharing among the original ranking features. It enables the observations from seen features to be propagated to unseen features within the same group during adaptation, which is critical in addressing the problem of sparsity in the limited adaptation data. However, defining the optimal grouping of ranking features is non-trivial we postpone the discussion of constructing g (·) to Section

Once the grouping function g (·) is given, another important component in our adaptation framework is the criterion to estimate the optimal transformation matrix A^u. An ideal transformation should be able to adjust the generic ranking model to meet each individual's ranking preference, i.e., maximizing the search utility for each user. In the study of learning to rank algorithms in IR, various types of objective functions, e.g., pairwise and listwise, have been proposed to realize the goal of optimizing ranking metrics. There fore, to make the proposed framework generally applicable, we do not restrict our adaptation objective to any specific form, but instantiate it with the objective function from the ranking algorithm we choose to adapt

We want to emphasize that although in our framework we utilize the objective function from the ranking algorithm to be adapted as the criterion to estimate the transformation matrix

A^u, it does not necessarily restrict the global model to being estimated by the same ranking algorithm. As long as the global model and adapted model share the same model structure, e.g., neural network structure in RankNet and linear model in RankSVM, the proposed adaptation framework is applicable.

$$\min_{A^u} \quad L_{\text{adapt}}(A^u) = L(Q^u; f^u) + \lambda R(A^u)$$
$$\text{where} \quad f^u(x) = (A^u \tilde{w}^s)^{\mathsf{T}} x \text{ and } \tilde{w}^s = (w^s, 1)$$

in which $L(Q^u;f^u)$ is the objective function defined in the ranking algorithm we choose to adapt, e.g., cross-entropy in Rank Net or hinge loss in Rank SVM, $R(A^u)$ is a regularization function defined on the transformation matrix A^u,λ is a trade-off parameter, and w s is the linear coefficients for ranking features in the global ranking model

Adapting RankNet & LambdaRank :

RankNet [.] is a probabilistic learning-to-rank algorithm, which models the probability that a document x^i_j is ranked higher than x^i_l for query q^i, i.e., $P(y^i_j > y^i_l)$. A logistic function is employed to map the predicted ranking scores of two documents, e.g., s^i_j and s^i_l, to probability of ordering

$$P(y_{ij} > y_{il}) = \frac{1}{1 + e^{-(s_{ij} - s_{il})}}.$$

The training objective function in RankNet is defined as the cross-entropy between the predicted pairwise ordering probabilities and the observed pairwise preferences in the training data, i.e.,

$$L_{\text{RankNet}} = \sum_{q_i} \sum_{y_{ij}, y_{il}} -\bar{P}(y_{ij} > y_{il}) \log P(y_{ij} > y_{il}) - (1 - \bar{P}(y_{ij} > y_{il})) \log(1 - P(y_{ij} > y_{il}))$$

where $\bar{P}(y_{ij} > y_{il})$ is the empirically estimated probability that x_{ij} is ranked higher than x_{il}. RankNet is usually optimized via a neural network. Because in each layer of a neural network, every neuron's out put is linearly combined to feed into the next layer, our adaption framework can be smoothly applied to the linear weights for each neuron (e.g., different transformation matrices for each neuron in the hidden layers). In order to understand the effect of the proposed adaptation in RankNet, we will use a RankNet with no hidden layers for discussion, but the same procedure can be applied to general RankNet with an arbitrary number of hidden layers. To adapt RankNet, we take the same cross-entropy function defined in as our adaptation objective, and define the following regularization function on matrix A^u

$$R(A^u) = \frac{1}{2}\sum_{k=1}^{K}(a_k^u - 1)^2 + \frac{\sigma}{2}\sum_{k=1}^{K} b_k^{u\,2}$$

Based on the discussion of adapting RankNet within the proposed framework, it is straightforward to adapt LambdaRank [22] in a similar manner. As a listwise learning to-rank algorithm, LambdaRank modifies the error term in RankNet by adding an additional correction term and names such modified error as lambda function Therefore, to adapt LambdaRank within our framework, we only need to replace the error function of the output layer in RankNet with the lambda function defined in and all the other procedures remain the same as in RankNet.

Adapting RankSVM:

RankSVM [16] is a classic pairwise learning-to-rank algorithm, in which the learning problem is formalized as

$$\begin{aligned}\min_{w,\xi_{ijl}} \quad & \frac{1}{2}||w||^2 + C\sum_{q_i}\sum_{j,l}\xi_{ijl} \\ \text{s.t.} \quad & w^{\mathsf{T}}\Delta x_{ijl} \geq 1 - \xi_{ijl}, \forall q_i, x_{ij}, x_{il} \\ & \xi_{ijl} \geq 0 \\ \text{where} \quad & y_{ij} > y_{ij} \text{ and } \Delta x_{ijl} = x_{ij} - x_{il}\end{aligned}$$

Where C is a trade-off parameter to control the balance between model complexity and empirical hinge loss over the identified preference pairs from the training data. To adapt RankSVM, we keep the hinge loss defined in our adaptation objective, and use the same regularization function for A^u defined.

By taking the linear transformation wu=Aũws, we get the adapted problem for RankSVM as

$$\begin{aligned}\min_{\mathbf{a}^u,\mathbf{b}^u,\xi_{ij}} \quad & \frac{1}{2}\sum_{k=1}^{K}(a_k^u - 1)^2 + \frac{\sigma}{2}\sum_{k=1}^{K}(b_k^u)^2 + C\sum_{q_i}\sum_{j,l}\xi_{ijl} \\ \text{s.t.} \quad & w^{u\mathsf{T}}\Delta x_{ijl} \geq 1 - \xi_{ijl}, \forall q_i, x_{ij}, x_{il} \\ & \xi_{ijl} \geq 0 \\ & w^u = A^u\tilde{w}^s, y_{ij} > y_{il}, \text{ and } \Delta x_{ijl} = x_{ij} - x_{il}\end{aligned}$$

Since the input for RankSVM training is document pairs, in the following discussion, we briefly denote Δ xijl as ⃗xt, in which the subscript t ranges over all the preference pairs in the adaptation set, to simplify the notations. Following the conventional derivation of RankSVM, we get the dual problem by introducing a set of Lagrange multipliers α

The effect of the proposed adaptation on RankSVM is clearly depicted in its dual form. First, as we know that the linear coefficients in front of the Lagrange multipliers α in correspond to the separation margin for each training instance in SVM. In the adapted problem, the margin is rescaled according to the global model fs(⃗xt)'s prediction on the adaptation data if the global model can well separate the adaptation pair⃗xt, i.e.,fs(⃗xt)>0, the margin decreases, indicating this case is not crucial for adaptation f the global model fails to correctly predict the order for this pair, i.e.,fs(⃗xt)≤0, the margin increases, and ⃗xt becomes a more important instance in adaptation for this particular user. This precisely interprets the effect of model based adaptation we only update the global model when it makes a mistake on the adaptation data; otherwise keep it intact. Second, the proposed linear transformations induce two new kernels in a compressed space K1(⃗xt, ⃗xs), corresponding to the scaling operation, defines a compound polynomial kernel over the ranking features projected by the global ranking model ws; and K2(⃗xt, ⃗xs), corresponding to the shifting operation, defines another compound polynomial kernel over the original ranking features. Both kernels work in a compressed K-dimensional space determined by the feature group mapping function g(·), and are interpolated by the balance parameter σ between the regularizations for shifting and scaling operations. As a result, non-linearity is

introduced to the original linear RankSVM model, and such non-linearity helps the model to leverage the observations from seen features to the unseen ones in the same group.

Work Analysis: In order to evaluate the proposed adaptation framework, we performed a series of experiments on large-scale search query logs sampled from Bing.com. A set of state-of-the-art ranking model adaptation methods were included as base lines to validate the effectiveness of the proposed method to apply the proposed adaptation method and compare with the aselines according to user click feedback, we can only use the queries with clicks. Therefore, in our experiment, we filtered out the queries without clicks and required each user to have at least two queries with clicks, i.e., one for adaptation and one for testing. We also sampled a large set of manually annotated query logs from our existing data collection as the user-independent training set for adaptation. Each query-document pair in this annotation set is labeled with a five-grade relevance score, i.e., from 0 for "bad" to 4 for "perfect." Documents in both the selected user data set and annotation data set are represented by a set of 1,830 ranking features selected from their overlapped feature set, including frequently used ranking features such as BM25, language model score and PageRank. Using the language of domain adaptation, we treat the collection of annotated queries as our source domain and each user's queries with clicks as target domain. This setting provides a good simulation for real Web search scenario, where the generic rankers in use are usually trained on offline annotated data, and thus it helps us compare the effectiveness of different ranking model adaptation methods. The basic statistics of the annotation set and selected user set are summarized in Preference pairs are extracted from user's clicks to reflect their unique search requirements. In order to meliorate the positional biases inherent in click data [1], we followed Joachims et al.'s method to extract the click preference pairs. In particular, we employed two click heuristics: for a given query q with a ranked document list {(x1, y1),(x2, y2),. . . ,(xn, yn)} returned by the search engine, In order to avoid defining different feature grouping functions for different ranking algorithms we selected to adapt, e.g., in RankNet each neuron in the hidden layers needs a possibly different grouping function but in RankSVM only one grouping function is needed for the original features, we decided not to use hidden layers in the neuron networks for RankNet and LambdaRank in our experiment. As a result, the same grouping function defined on the original ranking features can be directly used in RankNet, LambdaRank and RankSVM. A LambdaRank model optimizing NDCG@10 is trained on the annotation set and used as the global ranking model for adaptation in the following experiments (denoted as Source-Only)1. The trade-off parameter λ(in Eq (3))andσ(in Eq (5)) in our method are selected by 5-fold cross validation on the whole user set in advance. To quantitatively compare different adaptation method's performance, we employed a set of standard IR evaluation metrics: by treating all the clicked documents as relevant, we calculated Mean Average Precision (MAP), Precision at 1 (P@1), Precision at 3 (P@3) and Mean Reciprocal Rank (MRR). Definitions of these metrics can be found in process

Analysis of Feature Grouping: The grouping of features has a substantial impact on the adaptation performance in our method, since transformations will be shared for the arameters of features in the same group. Ideally, we should put parameters that need to be updated synchronously in the same

group. In this experiment, we evaluated the three feature grouping methods, i.e.,Name , SVD, and Cross, proposed. For comparison purposes, we also included two trivial grouping methods: 1) "Full," which creates a group for every single feature, i.e., no transformation is shared across features; 2)"RND," which randomly allocates features intoK groups. In our data set, according to the naming scheme of features, i.e., feature Type source seqID , 413 feature groups are extracted by the Name method. The two data-driven p-proaches, SVD and Cross , were performed on the annotation set, but we have to specify the group size K for them in advance. To analyze the effect of group size K in our proposed adaptation framework, we evaluated the adaptation performance of RankNet by varying the setting of K . To control the number of adaptation queries in each user, which influences the adaptation performance, we selected a subset of users, where each user has at least six queries with clicks (close to the average number of queries with clicks per user in our collection), and used the first three queries for adaptation and last three queries for testing in each user. This leads to a collection of 8,879 users with 112,069 queries. The MAP ranking performance of adapted RankNet with different feature grouping methods is shown in Figure 1 (a). First, it is clear that a properly set K is crucial for both SVD and Cross methods. The more groups we set, the more adaptation parameters need to estimate based on the limited adaptation data; but if we set too few feature groups, the Since we only used a subset of annotated queries and features, the results here do not reflect the actual performance of the search engine. discriminations among the features will be lost due to inaccurate parameter updating by adaptation sharing. Besides, (a) also shows that the adaptation performance is less sensitive to K around its optimal value, i.e., the perfor-mance as indicated by MAP is stable in a wide range of K from 400 to 800, for both SVD and Cross . Another observation (a) is that Cross performed consistently better than the other grouping methods under the same setting of K. Because in the Cross method features with similar contributions (i.e., linear coefficients) to document ranking are grouped together, and they tend to update synchronously. Sharing transformations among such features is more desirable. In contrast, other grouping methods cannot exploit such relationship among the features, e.g., SVD only exploits the co-occurrence relationship between features, and thus they achieved worse results. In order to understand the in-depth effect of feature grouping in our adaptation framework, we computed the average number of updated parameters in the adapted ranking model for each user with respect to different group size K and illustrated the results in Figure 1 (b). We can note that on average only 316 features (with a standard deviation of 214) can be observed in the adaptation data according to the result of Full method. However, because of adaptation transformation sharing across features in our framework, the number of parameters that have been actually adjusted is much larger. For example, with 800 groups, about 870 parameters (with a standard deviation of 220) on average are effectively updated by the Cross method, indicating that more than 60% of updated parameters are adapted with-out actual observations. On the other hand, when K becomes smaller, the number of updated parameters increases rapidly. Consequently, using too fewer groups forces less relevant features get updated by the shared transformations, which in turn degrades the overall adaptation performance. Similar adaptation results with respect to group size K were also observed in LambdaRank and RankSVM. In the following experiments, to avoid selecting K for each individual user and the

variation of performance introduced by this factor, we fix K to be 800 for both SVD and Cross.

In this section, we perform several experiments under two different settings, to demonstrate the effectiveness of the proposed RA-SVM based algorithms and the *ranking adaptability* measurement.

Datasets and Evaluation Measure: In imbalanced datasets, not only is the class distribution is skewed, the misclassication cost is often uneven to o. The minority class examples are often more imp ortant than the ma jority class examples. The cost of misclassify a minority class example i far greater than misclassify a ma jority class example, for example, fraud detection or cancer diagnosis. We will briey discuss some relevant evaluation measurements, starting with confusion matrix, which is closely related to many evaluation techniques and can be found in most data mining textb o oks. Shows a confusion matrix for outcome of a two class problem. As an example, using this table, we can dene the overall accuracy. We firstly conduct the experiments over the Letor benchmark dataset [20], and adapt the ranking model learned from TD2003 dataset to the ranking of TD2004 dataset. Letor TD2003 and TD2004 datasets are gathered from the topic distillation task of TREC 2003 and TREC 2004, with 50 queries for TD2003 and 75 ones for TD2004. The documents are collected by crawling from the .gov domain. For each query, about 1000 associated documents are returned, and labeled with a binary judgment, i.e., relevant or irrelevant. The features of TD2003 and TD2004 include the low-level features such as term frequency, inverse document frequency, and document length, as well as high-level features such as BM25, LMIR, PageRank, and HITS, for totally 44 dimensional features. However, Letor is a comparatively small dataset, and each document is only labeled with a binary relevance degree, which cannot reflect the practical Web search scenarios with multiple relevance degrees. Also, there are no domainspecific features for the target domain data, where we cannot demonstrate the effectiveness of the proposed ranking adaptation with domain-specific feature algorithms. Therefore, to give a more thorough analysis of our proposed RA-SVM based methods and to demonstrate the effectiveness of domain specific features, we collect more large scale datasets from a commercial internet search engine. Two datasets are separately gathered from different domains, i.e. the Web page search and the image search engines. There are totally 2625 queries for theWeb page search domain, and 1491 queries for image. At most 50 documents for each query are crawled and labeled, and therefore we obtain 122815 query-document pairs for Web page search and 71246 query-image pairs, resulting in 46.79 documents returned for eachWeb page search query for each image query on average. We take the visual features of images as domain-specific features for the image search domain, and try to utilize these features to boost the performance of adaptation from Web page search to image search. Note that the dataset of image search is a subset of the one used in our conference version [10]. This is because we have to crawl the images from the Web to extract their visual features as domain-specific features, in order to test the performance of RA-SVM-MR and RA-SVMSR. However, the URLs of some images are currently

invalid and we cannot download the corresponding images. Therefore, we have to select a subset of image queries from the original dataset, where each query has at least 30 images successfully downloaded. Query-dependent textual features are extracted for all query-document pairs based on

different document sources, e.g., the anchor text, the URL, the document title and the body. Some other features are also incorporated, such as the static rank, the junk page and so on. Totally 354 dimensional textual features are extracted. Each query-document pair is labeled with a relevance Confusion matrix is useful when accessing the performance without taking cost in to consideration. It is

used as a basis for various measures, such as precision and recal.

Cost curve was intro deuced by Drummond and Holte (2000), and they have also provided a detailed comparison between ROC curve and cost curve in Drummond and Holte (2004). Basically, cost curve lo oks at how classiers perform across a range of deferent mist classification cost. It can b e seen as dierent slop e line tangent to the ROC curve, therefore every ROC curve has a corresponding cost curve. This view of a slop e line b ears similarity to the discussion about baseline performance

Conclusion:

In this work, we proposed a general ranking model adaptation framework for personalized search. A series of learned linear transformations, e.g., scaling and shifting, were performed on the parameters of a generic linear ranking model in a per-user basis, such that the adapted model can better

fit each individual user's search result ranking preferences. By sharing transformations across features in a group-wise manner, unseen features can also be properly updated given only limited number of adaptation queries. We instantiated the proposed framework with three frequently used learning to rank algorithms, i.e., RankNet, LambdaRank and RankSVM, and the adaptation method achieved significant improvement in, not only adaptation efficiency, but also ranking performance of the adapted ranking models, against several state-of-the-art ranking model adaptation methods in extensive experimentation. In our current solution, the feature grouping function and transformation matrix are estimated independently. It would be meaningful to jointly estimate the two components for better adaptation performance. Besides, the proposed linear transformation based ranking model adaptation framework opens an interesting new direction for personalization: rich signals, e.g., user-specific profiles and features, could also be included to affect the transformation in order to better reflect users' individual search interests. In this paper we implemented a new ranking model adaptation framework which combines the SVM

and AdaBoost algorithms. The framework is built in such a way that it can be adapted to different domains. The domain specific search is possible without building different ranking models for different domains. This can't be achieved with domain specific search engines and broad based searching algorithms. Our decision to use AdaBoost along with SVM for ranking has shown positive results. We also built a prototype application with web based interface which demonstrates the proof of concept. The application is tested with data sets such as TD2003

and TD2004. The empirical results revealed that the proposed framework is adaptive to different domains

References:

[1] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for

predicting web search result preferences. In *SIGIR'06*, 2006.

[2] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[3] P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short- and long-term behavior on search personalization. In *SIGIR'12*, pages 185–194, 2012.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML'05*, pages 89–96. ACM, 2005.

[5] D. Chen, Y. Xiong, J. Yan, G. Xue, G. Wang, and Z. Chen. Knowledge transfer for cross domain learning to rank. *Information Retrieval*, 13(3):236–253, 2010.

[6] D. Chen, J. Yan, G. Wang, Y. Xiong, W. Fan, and Z. Chen. Transrank: A novel algorithm for transfer of rank learning. In *ICDMW'08*, pages 106–115. IEEE, 2008.

[7] P. Chirita, W. Nejdl, R. Paiu, and C. Kohlschu¨tter. Using odpmetadata to personalize search. In *SIGIR'05*, pages 178–185.ACM, 2005.

[8] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD'01*, pages 269–274. ACM, 2001.

[9] Z. Dou, R. Song, and J. Wen. A large-scale evaluation and analysis of personalized search strategies. In *WWW'07*, pages 581–590. ACM, 2007.

[10] K. Duh and K. Kirchhoff. Learning to rank with partially labeled data. In *SIGIR'08*, pages 251–258. ACM, 2008.

[11] R. Fidel and M. Crandall. Users' perception of the performance of a filtering system. In *SIGIR'97*, volume 31, pages 198–205. ACM, 1997.

[12] J. Gao, Q. Wu, C. Burges, K. Svore, Y. Su, N. Khan, S. Shah, and H. Zhou. Model adaptation via model interpolation and boosting for web search ranking. In *EMNLP'09*, pages 505–513,

[13] W. Gao, P. Cai, K. Wong, and A. Zhou. Learning to rank only using training data from related domain. In *SIGIR'10*, pages 162–169. ACM, 2010.

[14] B. Geng, L. Yang, C. Xu, and X. Hua. Ranking model adaptation for domain-specific search. *Knowledge and DataEngineering, IEEE Transactions on*, 24(4):745–758, 2012.

[15] X. He and W. Chou. Minimum classification error linear regression for acoustic model adaptation of continuous density hmms. In *ICME'03*, pages I–397. IEEE, 2003.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *KDD'02*, pages 133–142, 2002.

[17] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR'05*, pages 154–161. ACM, 2005.

[18] C. Leggetter and P. Woodland. Maximum likelihood linea regression for speaker adaptation of continuous density hidden markov models. *Computer speech and language*, 9(2):171, 1995.

[19] T. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[20] T. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR'07 workshop on learning to rank for information retrieval*, pages 3–10, 2007.

[21] S. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[22] C. Quoc and V. Le. Learning to rank with nonsmooth cost functions. In *NIPS'07*, volume 19, page 193, 2007.

[23] X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *SIGIR'05*, pages 43–50. ACM, 2005.

[24] D. Sontag, K. Collins-Thompson, P. N. Bennett, R. W. White, S. Dumais, and B. Billerbeck. Probabilistic models for personalizing web search. In *WSDM'12*, pages 433–442, 2012.

[25] B. Tan, X. Shen, and C. Zhai. Mining long-term search history to improve search accuracy. In *KDD'06*, pages 718–723, 2006.

[26] J. Teevan, S. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR'05*, pages 449–456. ACM, 2005.

Biographies:

Shanthi Sri.Regulagadda pursuing her M.Tech in Computer Science and Engineering at Chirala Engineering College, Prakasam District, Andhra Pradesh, India. Her research interest includes Analysis of Domain Specification in Ranking Model Adaptation. .

T.Ashok kumar, Associate Professor, M.Tech in Computer Science and Engineering at Chirala Engineering College, Prakasam District, Andhra Pradesh, India. He Guided for the project and involved in all technical aspects