Cross-Market Model Adaptation with Pairwise Preference Data for Web Search Ranking

Jing Bai
Microsoft Bing
1065 La Avenida
Mountain View, CA 94043
jbai@microsoft.com

Fernando Diaz, Yi Chang, Zhaohui Zheng
Yahoo! Labs
701 First Avenue
Sunnyvale, CA 94089
{diazf,yichang,zhaohui}@yahoo-inc.com

Keke Chen
Computer Science
Wright State
Dayton, Ohio 45435
keke.chen@wright.edu

Abstract

Machine-learned ranking techniques automatically learn a complex document ranking function given training data. These techniques have demonstrated the effectiveness and flexibility required of a commercial web search. However, manually labeled training data (with multiple absolute grades) has become the bottleneck for training a quality ranking function, particularly for a new domain. In this paper, we explore the adaptation of machine-learned ranking models across a set of geographically diverse markets with the market-specific pairwise preference data, which can be easily obtained from clickthrough logs. We propose a novel adaptation algorithm, PairwiseTrada, which is able to adapt ranking models that are trained with multi-grade labeled training data to the target market using the target-market-specific pairwise preference data. We present results demonstrating the efficacy of our technique on a set of commercial search engine data.

1 Introduction

Web search algorithms provide methods for ranking web scale collection of documents given a short query. The success of these algorithms often relies on the rich set of document properties or *features* and the complex relationships between them. Increasingly, machine learning techniques are being used to learn these relationships for an effective ranking function (Liu, 2009). These techniques use a set of labeled *training data* labeled with multiple relevance grades to automatically estimate parameters of a model which directly optimizes a performance metric. Although training data often is derived from editorial labels of document relevance, it can also be inferred from a careful analysis of users' interactions with a working system (Joachims, 2002). For example, in web search, given a query, document preference information can be derived from user clicks. This data can then be used with an algorithm which learns from pairwise preference data (Joachims, 2002; Zheng et al., 2007). However, automatically extracted pairwise preference data is subject to noise due to the specific sampling methods used (Joachims et al., 2005; Radlinski and Joachim, 2006; Radlinski and Joachim, 2007).

One of the fundamental problems for a web search engine with global reach is the development of ranking models for different regional markets. While the approach of training a single model for all markets is attractive, it fails to fully exploit of specific properties of the markets. On the other hand, the approach of training market-specific models requires the huge overhead of acquiring a large training set for each market. As a result, techniques have been developed to create a model for a small market, say a Southeast Asian country, by combining a strong model in another market, say the United States, with a

Coling 2010: Poster Volume, pages 18–26,
Beijing, August 2010

small amount of manually labeled training data in the small market (Chen et al., 2008b). However, the existing Trada method takes only multi-grade labeled training data for adaptation, making it impossible to take advantage of the easily harvested pairwise preference data. In fact, to our knowledge, there is no adaptation algorithm that is specifically developed for pairwise data.

In this paper, we address the development market-specific ranking models by leveraging pairwise preference data. The pairwise preference data contains most market-specific training examples, while a model from a large market may capture the common characteristics of a ranking function. By combining them algorithmically, our approach has two unique advantages. (1) The biases and noises of the pairwise preference data can be depressed by using the base model from the large market. (2) The base model can be tailored to the characteristics of the new market by incorporating the market specific pairwise training data. As the pairwise data has the particular form, the challenge is how to effectively use pairwise data in adaptation. This appeals to the following objective of many web search engines: design algorithms which minimize manually labeled data requirements while maintaining strong performance.

2 Related Work

In recent years, the ranking problem is frequently formulated as a supervised machine learning problem, which combines different kinds of features to train a ranking function. The ranking problem can be formulated as learning a function with pair-wise preference data, which is to minimize the number of contradicting pairs in training data. For example, RankSVM (Joachims, 2002) uses support vector machines to learn a ranking function from preference data; RankNet (Burges et al., 2005a) applies neural network and gradient descent to obtain a ranking function; RankBoost (Freund et al., 1998) applies the idea of boosting to construct an efficient ranking function from a set of weak ranking functions; GBRank (Zheng et al., 2007; Xia et al., 2008) using gradient descent in function spaces, which is able to learn relative ranking information in the context of web search. In addition, Several studies have been focused on learning ranking functions in semi-supervised learning framework (Amini et al., 2008; Duh and Kirchhoff, 2008), where unlabeled data are exploited to enhance ranking function. Another approach to learning a ranking function addresses the problem of optimizing the list-wise performance measures of information retrieval, such as mean average precision or Discount Cumulative Gain (Cao et al., 2007; Xu et al., 2008; Wu et al., 2009; Chen et al., 2008c). The idea of these methods is to obtain a ranking function that is optimal with respect to some information retrieval performance measure.

Model adaptation has previously been applied in the area of natural language processing and speech recognition. This approach has been successfully applied to parsing (Hwa, 1999), tagging (Blitzer et al., 2006), and language modeling for speech recognition (Bacchiani and Roark, 2003). Until very recently, several works have been presented on the topic of model adaptation for ranking (Gao et al., 2009; Chen et al., 2008b; Chen et al., 2009), however, none of them target the model adaptation with the pair-wise learning framework. Finally, multitask learning for ranking has also been proposed as a means of addressing problems similar to those we have encountered in model adaptation (Chen et al., 2008a; Bai et al., 2009; Geng et al., 2009).

3 Background

3.1 Gradient Boosted Decision Trees for Ranking

Assume we have a training data set, $\mathcal{D} = \{\langle (q,d), y\rangle_1, \ldots, \langle (q,d), y\rangle_n\}$, where $\langle (q,d), t\rangle_i$ encodes the labeled relevance, y, of a document, d, given query, q. Each query-document pair, (q,d), is represented by a set of features, $(q,d) = \{x_{i1}, x_{i2}, x_{i3}, ..., x_{im}\}$. These features include, for example, query-document match features, query-specific features, and document-specific features. Each relevance judgment, y, is a relevance grade mapped (e.g. "relevant", "somewhat relevant", "non-relevant") to a real

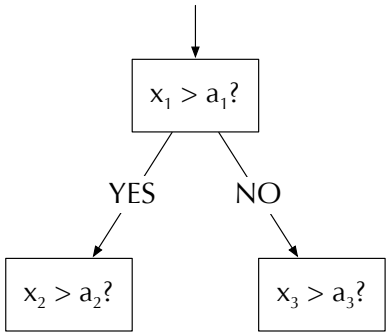

Figure 1: An example of base tree, where x_1, x_2 and x_3 are features and a_1, a_2 and a_3 are their splitting values.

number. Given this representation, we can learn a *gradient boosted decision tree* (GBDT) which models the relationship between document features, (q, d), and the relevance score, y, as a decision tree (Friedman, 2001). Figure 1 shows a portion of such a tree. Given a new query document pair, the GBDT can be used to predict the relevance grade of the document. A ranking is then inferred from these predictions. We refer to this method as GBDT_{reg}.

In the training phase, GBDT_{reg} iteratively constructs regression trees. The initial regression tree minimizes the L_2 loss with respect to the targets, y,

$$L_2(f, y) = \sum_{\langle (q,d), y \rangle} (f(q, d) - y)^2 \qquad (1)$$

As with other boosting algorithms, the subsequent trees minimize the L_2 loss with respect to the residuals of the predicted values and the targets. The final prediction, then, is the sum of the predictions of the trees estimated at each step,

$$f(x) = f^1(x) + \ldots + f^k(x) \qquad (2)$$

where $f^i(x)$ is the prediction of the ith tree.

3.2 Pairwise Training

As alternative to the absolute grades in $\mathcal{D}$, we can also imagine assembling a data set of *relative judgments*. In this case, assume we have a training data set $\mathcal{D}^{\succ} = \{\langle (q, d), (q, d'), \rho \rangle_1, \ldots, \langle (q, d), (q, d'), \rho \rangle_n\}$, where $\langle (q, d), (q, d'), \rho \rangle_i$ encodes the preference, of a document, d, to a second document, d', given query, q. Again, each query-document pair is represented by a set of features. Each preference judgment, $\rho \in \{\succ, \prec\}$, indicates whether document d is preferred to document d' ($d \succ d'$) or not ($d \prec d'$).

Preference data is attractive for several reasons. First, editors can often more easily determine preference between documents than the absolute grade of single documents. Second, relevance grades can often vary between editors. Some editors may tend to overestimate relevance compared to another editor. As a result, judgments need to be rescaled for editor biases. Although preference data is not immune to inter-editor inconsistency, absolute judgments introduce two potential sources of noise: determining a relevance ordering and determining a relevance grade. Third, even if grades can be accurately labeled, mapping those grades to real values is often done in a heuristic or *ad hoc* manner. Fourth, GBDT_{reg} potentially wastes modeling effort on predicting the grade of a document as opposed to focusing on optimizing the rank order of documents, the real goal a search engine. Finally, preference data can often be mined from a production system using assumptions about user clicks.

In order to support preference-based training data, (Zheng et al., 2007) proposed GBRANK based on GBDT_{reg}. The GBRANK training algorithm begins by constructing an initial tree which predicts a constant score, c, for all instances. A pair is contradicting if the $\langle (q, d), (q, d'), \succ \rangle$ and prediction $f(q, d) < f(q, d')$. At each boosting stage, the algorithm constructs a set of contradicting pairs, $\mathcal{D}^{\succ}_{\text{contra}}$. The GBRANK algorithm then adjusts the response variables, $f(q, d)$ and $f(q, d')$, so that $f(q, d) > f(q, d')$. Assume that $(q, d) \succ (q, d')$ and $f(q, d) < f(q, d')$. To correct the order, we modify the target values,

$$\tilde{f}(q, d) = f(q, d) + \tau \qquad (3)$$

$$\tilde{f}(q, d') = f(q, d') - \tau \qquad (4)$$

where $\tau > 0$ is a margin parameter that we

need to assign. In our experiments, we set τ to 1. Note that if preferences are inferred from absolute grades, $\mathcal{D}$, minimizing the L_2 to 0 also minimizes the contradictions.

3.3 Tree Adaptation

Recall that we are also interested in using the information learned from one market, which we will call the *source market*, on a second market, which we will call the *target market*. To this end, the Trada algorithm adapts a GBDT_{reg} model from the source market for the target market by using a small amount of target market absolute relevance judgments (Chen et al., 2008b). Let the $\mathcal{D}_s$ be the data in the source domain and $\mathcal{D}_t$ be the data in target domain. Assume we have trained a model using GBDT_{reg}. Our approach will be to use the decision tree structure learned from $\mathcal{D}_s$ but to adapt the thresholds in each node's feature. We will use Figure 1 to illustrate Trada. The splitting thresholds are a_1, a_2 and a_3 for rank features x_1, x_2 and x_3. Assume that the data set $\mathcal{D}_t$ is being evaluated at the root node v in Figure 1. We will split the using the feature $v_x = x_1$ but will compute a new threshold v'_a using $\mathcal{D}_t$ and the GBDT_{reg} algorithm. Because we are discussing the root node, when we select a threshold b, $\mathcal{D}_t$ will be partitioned into two sets, $\mathcal{D}_t^{>b}$ and $\mathcal{D}_t^{<b}$ representing those instances whose feature x_1 has a value greater and lower than b. The response value for each partition will be the uniform average of instances in that partition,

$$f = \begin{cases} \frac{1}{|\mathcal{D}_t^{>b}|} \sum_{d_i \in \mathcal{D}_t^{>b}} y_i & \text{if } d_i \in \mathcal{D}_t^{>b} \\ \frac{1}{|\mathcal{D}_t^{<b}|} \sum_{d_i \in \mathcal{D}_t^{<b}} y_i & \text{if } d_i \in \mathcal{D}_t^{<b} \end{cases} \quad (5)$$

We would like to select a value for b which minimizes the L_2 loss between y and f in Equation 5; equivalently, b can be selected to minimize the variance of y in each partition. In our implementation, we compute the L_2 loss for all possible values of the feature v'_x and select the value which minimizes the loss.

Once b is determined, the adaptation consists of performing a linear interpolation between the original splitting threshold v_a and the new splitting threshold b as follows:

$$v'_a = pv_a + (1-p)b \quad (6)$$

where p is an adaptation parameter which determines the scale of how we want to adapt the tree to the new task. If there is no additional information, we can select p according to the size of the data set,

$$p = \frac{|\mathcal{D}_s^{<a}|}{|\mathcal{D}_s^{<a}| + |\mathcal{D}_t^{<b}|} \quad (7)$$

In practice, we often want to enhance the adaptation scale since the training data of the extended task is small. Therefore, we add a parameter β to boost the extended task as follows:

$$p = \frac{|\mathcal{D}_s^{<a}|}{|\mathcal{D}_s^{<a}| + \beta|\mathcal{D}_t^{<b}|} \quad (8)$$

The value of β can be determined by cross-validation. In our experiments, we set β to 1.

The above process can also be applied to adjust the response value of nodes as follows:

$$v'_f = pv_f + (1-p)f \quad (9)$$

where v'_f is the adapted response at a node, v_f is its original response value of source model, and f is the response value (Equation 5).

The complete Trada algorithm used in our experiments is presented in Algorithm 1.

Algorithm 1 Tree Adaptation Algorithm

TRADA$(v, \mathcal{D}_t, p)$
1 $b \leftarrow$ COMPUTE-THRESHOLD$(v_x, \mathcal{D}_t)$
2 $v'_a \leftarrow pv_a + (1-p)b$
3 $v'_f \leftarrow pv_f + (1-p)$MEAN-RESPONSE$(\mathcal{D}_t)$

4 $\mathcal{D}'_t \leftarrow \{x \in \mathcal{D}_t : x_i < v'_a\}$
5 $v'_< \leftarrow$ TRADA$(v_<, \mathcal{D}'_t, p)$

6 $\mathcal{D}''_t \leftarrow \{x \in \mathcal{D}_t : x_i > v'_a\}$
7 $v'_> \leftarrow$ TRADA$(v_>, \mathcal{D}''_t, p)$

8 **return** v'

The Trada algorithm can be augmented with a second phase which directly incorporates the target training data. Assume that our source model, $\mathcal{M}_s$, was trained using source data, $\mathcal{D}_s$. Recall that $\mathcal{M}_s$ can be decomposed as a sum of regression tree output, $f_{\mathcal{M}_s}(x) = f^1_{\mathcal{M}_s}(x) + \ldots + f^k_{\mathcal{M}_s}(x)$. *Additive tree adaptation* refers augmenting this summation with a set of regression trees trained on the residuals between the model, $\mathcal{M}_s$, and the target training data, $\mathcal{D}_t$. That is, $f_{\mathcal{M}_t}(x) = f^1_{\mathcal{M}_s}(x) + \ldots + f^k_{\mathcal{M}_s}(x) + f_{\mathcal{M}_t}(x)^{k+1} + \ldots + f_{\mathcal{M}_t}(x)^{k+k'}$. In order for us to perform additive tree adaptation, the source and target data must use the same absolute relevance grades.

4 Pairwise Adaptation

Both GBRANK and Trada can be used to reduce the requirement on editorial data. GBRANK achieves the goal by leveraging preference data, while Trada does so by leveraging data from a different search market. A natural extension to these methods is to leverage both sources of data simultaneously. However, no algorithm has been proposed to do this so far in the literature. We propose an adaptation method using pairwise preference data.

Our approach shares the same intuition as Trada: maintain the tree structure but adjust decision threshold values against some target value. However, an important difference is that our adjustment of threshold values does not regress against some target grade values; rather its objective is to improve the ordering of documents. To make use of preference data in the tree adaptation, we follow the method used in GBRANK to adjust the target values whenever necessary to preserve correct document order. Given a base model, $\mathcal{M}_s$, and preference data, $\mathcal{D}_t^{\succ}$, we can use Equations 3 and 4 to *infer* target values. Specifically, we construct a set $\mathcal{D}^{\succ}_{\text{contra}}$ from $\mathcal{D}_t^{\succ}$ and $\mathcal{M}_s$. For each item (q, d) in $\mathcal{D}^{\succ}_{\text{contra}}$, we use the value of $\tilde{f}(q, d)$ as the target. These tuples, $\langle (q, d), \tilde{f}(q, d) \rangle$ along with $\mathcal{M}_s$ are then are provided as input to Trada. Our approach is described in Algorithm 2.

Compared to Trada, Pairwise-Trada has two important differences. First, Pairwise-Trada can use a source GBDT model trained either against absolute or pairwise judgments. When an organization maintains a set of ranking models for different markets, although the underlying modeling method may be shared (e.g. GBDT), the learning algorithm used may be market-specific (e.g. GBRANK or GBDT_{reg}). Unfortunately, classic Trada relies on the source model being trained using GBDT_{reg}. Second, Pairwise-Trada can be adapted using pairwise judgments. This means that we can expand our adaptation data to include click feedback, which is easily obtainable in practice.

Algorithm 2 Pairwise Tree Adaptation Algorithm

PAIRWISE-TRADA($\mathcal{M}_s, \mathcal{D}_t^{\succ}, p$)

1 $\mathcal{D}_{\text{contra}} \leftarrow$ FIND-CONTRADICTIONS($\mathcal{M}_s, \mathcal{D}_t^{\succ}$)
2 $\tilde{\mathcal{D}}_t \leftarrow \{\langle (q, d), \tilde{f}(q, d) \rangle : (q, d) \in \mathcal{D}_{\text{contra}}\}$
3 **return** TRADA(ROOT($\mathcal{M}_s$), $\tilde{\mathcal{D}}_t, p$)

5 Methods and Materials

The proposed algorithm is a straightforward modification of previous ones. The question we want to examine in this section is whether this simple modification is effective in practice. In particular, we want to examine whether pairwise adaptation is better than the original adaptation Trada using grade data, and whether the pairwise data from a market can help improve the ranking function on a different market.

Our experiments evaluate the performance of Pairwise-Trada for web ranking in ten target markets. These markets, listed in Table 1, cover a variety of languages and cultures. Furthermore, resources, in terms of documents, judgments, and click-through data, also varies across markets. In particular, editorial query-document judgments range from hundreds of thousands (e.g. SEA_1) to tens of thousands (e.g. SEA_5). Editors graded query-document pairs on a five-point relevance scale, resulting in our data set $\mathcal{D}$. Preference labels, $\mathcal{D}^{\succ}$, are inferred from these judgments.

We also include a second set of experiments which incorporate click data.[1] In these experiments, we infer a preference from click data by assuming the following model. The user is presented with ten results. An item $i \succ j$ if i the following conditions hold: i is positioned below j, i receives a click, and j does not receive a click.

In our experiments, we tested the following runs,

- GBDT_{reg} trained using only $\mathcal{D}_s$ or $\mathcal{D}_t$
- GBRANK trained using only $\mathcal{D}_s^{\succ}$ or $\mathcal{D}_t^{\succ}$
- GBRANK trained using only $\mathcal{D}_s^{\succ}$, $\mathcal{D}_t^{\succ}$, and $\mathcal{C}_t$
- Trada with both GBDT_s and GBRANK_s, adapted with $\mathcal{D}_t$.
- Pairwise-Trada with both GBDT_s and GBRANK_s, adapted with $\mathcal{D}_t^{\succ}$ and $\mathcal{C}_t$ at different ratios.

In the all experiments, we use 400 additive trees when additive adaptation is used.

All models are evaluated using discounted cumulative gain (DCG) at rank cutoff 5 (Järvelin and Kekäläinen, 2002).

6 Results

6.1 Adaptation with Manually Labeled Data

In Table 1, we show the results for all of our experimental conditions.

We can make a few observations about the non-adaptation baselines. First, models trained on the (limited) target editorial data, GBDT_t and GBRANK_t, tend to outperform those trained only on the source editorial data, GBDT_s and GBRANK_s. The critical exception is SEA_5, the market with the fewest judgments. We believe that this behavior is a result of similarity between the United States source data and the SEA_5 target market; both the source and target query populations share the same language, a property not exhibited in other markets. Notice that other small markets such as LA_2 and LA_3 see modest improvements when using target-only runs compared to source-only runs. Second, GBRANK tends to outperform GBDT when only trained on the source data. This implies that we should prefer a base model which is based on GBRANK, something that is difficult to combine with classic Trada. Third, by comparing GBRANK and GBDT when only trained on the target data, we notice that the effectiveness of GBRANK depends on the amount of training data. For markets where there training data is plentiful (e.g. SEA_1), GBRANK outperforms GBDT. On the other hand, for smaller markets (e.g. LA_3), GBDT outperforms GBRANK.

In general, the results confirm the hypothesis that adaptation runs outperform all of non-adaptation baselines. This is the case for both Trada and Pairwise-Trada. As with the baseline runs, the Australian market sees different performance as a result of the combination of a small target editorial set and a representative source domain. This effect has been observed in previous results (Chen et al., 2009).

We can also make a few observations by comparing the adaptation runs. Trada works better with a GBDT base model than with a GBRANK base model. We We believe this is the case because the absolute regression targets are difficult to compare with the unbounded output of GBRANK. Pairwise-Trada on the other hand tends to perform better with a GBRANK base model than with a GBDT base model. There are a few exceptions, SEA_3 and LA_2, where Pairwise-Trada works better with a GBDT base model. Comparing Trada to Pairwise-Trada, we find that using preference targets tends to improve performance for some markets but not all. The underperformance of Pairwise-Trada tends to occur in smaller markets such as LA_1, LA_2, and LA_3. This is similar to the behavior we observed in the non-adaptation runs and suggests that, in operation, a modeler may have to decide on the training algorithm based on the amount of data available.

[1] For technical reasons, this data set is slightly different from the results we show with the purely editorial data. Therefore the size of the training and testing sets are different, but not to a significant degree.

	SEA_1	SEA_2	EU_1	SEA_3	EU_2	SEA_4	LA_1	LA_2	LA_3	SEA_5
training size	243,790	174,435	137,540	135,066	101,076	100,846	91,638	75,989	66,151	37,445
testing size	18,652	26,752	11,431	13,839	12,118	12,214	11,038	16,339	10,379	21,034
$GBDT_s$	9.4483	8.1271	9.0018	10.0630	8.5339	5.9176	6.1699	11.4167	8.1416	10.5356
$GBDT_t$	9.6011	8.6225	9.3310	10.7591	9.0323	6.4185	6.8441	11.8553	8.5702	10.4561
$GBRANK_s$	9.6059	8.1784	9.0775	10.2486	8.6248	6.1298	6.2614	11.5186	8.2851	10.5915
$GBRANK_t$	9.6952	8.6225	9.3575	10.8595	9.0384	6.4620	6.8543	11.7086	8.4825	10.3469
Trada										
$GBDT_s, \mathcal{D}_t$	**9.6718**	**8.6120**	**9.3086**	**10.8001**	**9.1024**	**6.3440**	**6.9444**	**11.9513**	**8.6519**	10.6279
$GBRANK_s, \mathcal{D}_t$	**9.6116**	**8.5681**	**9.2125**	**10.7597**	**8.9675**	**6.4110**	**6.8286**	**11.7326**	**8.5498**	10.6508
Pairwise-Trada										
$GBDT_s, \mathcal{D}_t$	**9.7364**	**8.6261**	**9.3824**	**10.8549**	**9.0842**	**6.4705**	**6.9438**	**11.8255**	**8.5323**	10.4655
$GBRANK_s, \mathcal{D}_t$	**9.7539**	**8.6538**	**9.4269**	**10.8362**	**9.1044**	**6.4716**	**6.9438**	**11.8034**	**8.6187**	10.6564

Table 1: Adaptation using manually labeled training data Southeast Asia (SEA), Europe (EU), and Latin America (LA) markets. Markets are sorted by target training set size. Significance tests use a t-test. Bolded numbers indicate statistically significant improvements over the respective source model.

	SEA_1	SEA_2	EU_1	SEA_3	EU_2	SEA_4	LA_1	LA_2	LA_3	SEA_5
training size	194,114	166,396	136,829	161,663	94,875	96,642	73,977	108,350	64,481	71,549
testing size	15,655	11,844	11,028	11,839	11,118	5,092	10,038	12,246	10,201	7,477
$GBRANK_s$	9.0159	8.5763	8.7119	11.4512	9.7641	6.5941	6.894	7.9366	8.058	10.7935
Pairwise-Trada										
$GBRANK_s, \mathcal{D}_t, \mathcal{C}_t$										
editorial	**9.3577**	**8.9205**	**8.901**	**12.2247**	9.9531	6.7421	**7.1455**	**8.2811**	**8.2503**	10.7973
click	9.1149	8.7622	8.8187	**11.9361**	9.8818	6.7703	**7.1812**	**8.264**	**8.2485**	10.9042
editorial+click	**9.4898**	**9.0177**	**8.945**	**12.3172**	**10.1156**	6.8459	**7.2414**	**8.4111**	**8.292**	**11.1407**

Table 2: Adaptation incorporating click data. Bolded numbers indicate statistically significant improvements over the baseline. Markets ordered as in Table 1.

6.2 Incorporating Click Data

One of the advantages of Pairwise-Trada is the ability to incorporate multiple sources of pairwise preference data. In this paper, we use the heuristic rule approach which is introduced by (Dong et al., 2009) to extract pairwise preference data from the click log of the search engine. This approach yields both skip-next and skip-above pairs (Joachims et al., 2005), which are sorted by confidence descending order respectively. In these experiments, we combine manually generated preferences with those gathered from click data. We present these results in Table 2.

We notice that no matter the source of preference data, Pairwise-Trada outperforms the baseline GBRANK model. The magnitude of the improvement depends on the source data used. Comparing the editorial-only to the click-only models, we notice that click-only models outperform editorial-only models for smaller markets (SEA_4, LA_1, and SEA_5). This is likely the case because the relative quantity of click data with respect to editorial data is higher in these markets. This is despite the fact that the click data may be noisier than the editorial data. The best performance, though, comes when we combine both editorial and click data.

6.3 Additive tree adaptation

Recall that Pairwise-Trada consists of two parts: parameter adaptation and additive tree adaptation. In this section, we examine the contribution to performance each part is responsible for. Figure 2 illustrates the adaptation results for the LA_1 market. In this experiment, we use a United States base model and 100K LA_1 editorial judgments for adaptation. Pairwise-Trada is performed on top of differently sized base models with 600, 900 and 1200 trees. The original base model has 1200 trees; we selected the first 600, 900 or full 1200 trees for experiments. The number of trees used in the additive tree adaptation step ranges up to 600 trees. From Figure 2 we can see that the additive adaptation can

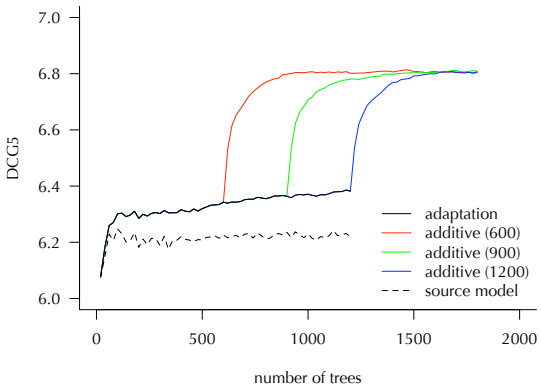

Figure 2: Illustration of additive tree adaptation for LA_1. The curves are average performance over a range of parameter settings.

significantly increase DCG over simple parameter adaptation and is therefore a critical step of Pairwise-Trada. When the number of trees in the additive tree adaptation step reaches roughly 400, the DCG plateaus.

7 Conclusion

We have proposed a model for adapting retrieval models using preference data instead of absolute relevance grades. Our experiments demonstrate that, when much editorial data is present, our method, Pairwise-Trada, may be preferable to competing methods based on absolute relevance grades. However, in real world systems, we often have access to sources of preference data beyond those resulting from editorial judgments. We demonstrated that Pairwise-Trada can exploit such data and boost performance significantly. In fact, *if we omit editorial data altogether we see performance improvements over the baseline model*. This suggests that, in principle, we can train a single, strong source model and improve it using target click data alone. Despite the fact that the modification we made is quite simple, we showed that modification is effective in practice. This tends to validate the general principle of using pairwise data from a different market. This principle can be easily used in other frameworks such as neural networks (Burges et al., 2005b). Therefore, the proposed method also points to a new direction for future improvements of search engines.

There are several areas of future work. First, we believe that detecting other sources of preference data from user behavior can further improve the performance of our model. Second, we only used a single source model in our experiments. We would also like to explore the effect of learning from an ensemble of source models. The importance of each may depend on the similarity to the target domain. Finally, we would also like to more accurately understand the queries where click data improves adaptation and those where editorial judgments is required. This sort of knowledge will allow us to train systems which maximally exploit our editorial resources.

References

Amini, M.-R., T.-V. Truong, and C. Goutte. 2008. A boosting algorithm for learning bipartite ranking functions with partially labeled data. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*.

Bacchiani, M. and B. Roark. 2003. Unsupervised language model adaptation. In *ICASSP '03: Proceedings of the International Conference on Acoustics, Speech and Signal Processing*.

Bai, J., K. Zhou, H. Zha, B. Tseng, Z. Zheng, and Y. Chang. 2009. Multi-task learning for learning to rank in web search. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*.

Blitzer, J., R. McDonald, and F. Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP '06: Proceedings of the 2006 Conference on Empirical Methods on Natural Language Processing*.

Burges, C., T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. 2005a. Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd International Conference on Machine learning*.

Burges, Chris, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005b. Learning to rank using gradient descent. In *ICML '05: Proceedings of the*

22nd international conference on Machine learning, pages 89–96. ACM.

Cao, Z., T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. 2007. from pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*.

Chen, D., J. Yan, G. Wang, Y. Xiong, W. Fan, and Z. Chen. 2008a. Transrank: A novel algorithm for transfer of rank learning. In *ICDM workshop '08: Proceeding of IEEE Conference on Data Mining*.

Chen, K., R. Lu, C. K. Wong, G. Sun, L. Heck, and B. Tseng. 2008b. Trada: tree based ranking function adaptation. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1143–1152, New York, NY, USA. ACM.

Chen, W., T.-Y. Liu, Y. Lan, Z. Ma, and H. Li. 2008c. Measures and loss functions in learning to rank. In *NIPS '08: Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*.

Chen, K., J. Bai, S. Reddy, and B. Tseng. 2009. On domain similarity and effectiveness of adapting-to-rank. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1601–1604, New York, NY, USA. ACM.

Dong, A., Y. Chang, S. Ji, C. Liao, X. Li, and Z. Zheng. 2009. Empirical exploitation of click data for query-type-based ranking. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods on Natural Language Processing*.

Duh, K. and K. Kirchhoff. 2008. Learning to rank with partially-labeled data. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*.

Freund, Y., R. D. Iyer, R. E. Schapire, and Y. Singer. 1998. An efficient boosting algorithm for combining preferences. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*.

Friedman, J. H. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232.

Gao, J., Q. Wu, C. Burges, K. Svore, Y. Su, N. Khan, Shah S., and H. Zhou. 2009. Model adaptation via model interpolation and boosting for web search ranking. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods on Natural Language Processing*.

Geng, B., L. Yang, C. Xu, and X.-S. Hua. 2009. Ranking model adaptation for domain-specific search. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 197–206, New York, NY, USA. ACM.

Hwa, R. 1999. Supervised grammar induction using training data with limited constituent information. In *ACL '99: Proceedings of the Conference of the Association for Computational Linguistics*.

Järvelin, Kalervo and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20(4):422–446.

Joachims, T., L. Granka, B. Pan, and G. Gay. 2005. Accurately interpreting clickthrough data as implicit feedback.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM Press.

Liu, T.-Y. 2009. *Learning to Rank for Information Retrieval*. Now Publishers.

Radlinski, F. and T. Joachim. 2006. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs.

Radlinski, F. and T. Joachim. 2007. Active exploration for learning rankings from clickthrough data.

Wu, M., Y. Chang, Z. Zheng, and H. Zha. 2009. Smoothing dcg for learning to rank: A novel approach using smoothed hinge functions. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*.

Xia, F., T.-Y. Liu, J. Wang, W. Zhang, and H. Li. 2008. Listwise approach to learning to rank: Theorem and algorithm. In *ICML '08: Proceedings of the 25th international conference on Machine learning*.

Xu, J., T.Y. Liu, M. Lu, H. Li, and W.Y. Ma. 2008. Directly optimizing evaluation measures in learning to rank. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*.

Zheng, Z., K. Chen, G. Sun, and H. Zha. 2007. A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 287–294. ACM.