Multi-Task Learning for Learning to Rank in Web Search

Jing Bai
Yahoo! Labs
701 First Avenue, Sunnyvale, CA
jingbai@yahoo-inc.com

Hongyuan Zha
College of Computing
Georgia Institute of Technology
Atlanta, GA
zha@cc.gatech.edu

Ke Zhou, Guirong Xue
Dept. of Computer Science and Engineering
Shanghai Jiao-Tong University
zhouke, grxue@apex.sjtu.edu.cn

Gordon Sun, Belle Tseng, Zhaohui Zheng, Yi Chang
Yahoo! Labs
701 First Avenue, Sunnyvale, CA
gzsun, belle, zhaohui, yichang@yahoo-inc.com

ABSTRACT

Both the quality and quantity of training data have significant impact on the performance of ranking functions in the context of learning to rank for web search. Due to resource constraints, training data for smaller search engine markets are scarce and we need to leverage existing training data from large markets to enhance the learning of ranking function for smaller markets. In this paper, we present a boosting framework for learning to rank in the multi-task learning context for this purpose. In particular, we propose to learn non-parametric common structures adaptively from multiple tasks in a stage-wise way. An algorithm is developed to iteratively discover super-features that are effective for all the tasks. The estimation of the functions for each task is then learned as a linear combination of those super-features. We evaluate the performance of this multi-task learning method for web search ranking using data from a search engine. Our results demonstrate that multi-task learning methods bring significant relevance improvements over existing baseline methods.

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

1. INTRODUCTION

Ranking functions are at the core of search engines and they directly influence the relevance of search results and user search experience. Machine learning approaches for learning ranking functions, entails the collection of training data, in the form of labeled data constructed from relevance assessment by human editors. This approach has proven to be effective for large search markets for which we have a large amount of training data. However, there are a number of small markets for which it is difficult to acquire large quantities of relevance judgments. One idea to alleviate this problem is to leverage the existing training data that have been collected for source markets, and use them to help training the ranking function for a target market with insufficient training data. Multi-task learning and transfer learning, which have been well studied in machine learning community, can be used to deal with this problem. However, to our knowledge, these approaches have not been used and tested on large datasets from search engines. In this paper, we investigate the possibility of using these approaches to train ranking functions for search markets where human labeled training data are limited.

A prerequisite for multi-task learning to be advantageous is that the tasks share some common characteristics. This is the case for search engine markets (i.e. with different languages and regions). While each search engine market has specific characteristics regarding the language, region, etc, different markets have much in common. For example, search engines rely on features such as term occurrences and co-occurrences in text and anchor texts, for learning ranking functions. These features are common across markets and they are often used in similar ways across markets. Therefore, the ranking functions for different markets also have much to share, and this provides the basis to take advantage of training data from multiple search engine markets. Why can multi-task learning be a good solution? On one hand, compared to separate learning for each task, by grouping the training data of several tasks, we have a larger amount of training data. If the underlying characteristics among the training data are similar, the resulting ranking functions can be better. On the other hand, in separate learning, the resulting ranking function for a task can be easily over-fitted when the training data is limited. By grouping several tasks in multi-task learning, we try to extract the features that are important for all the tasks, thereby reducing the risk of over-fitting to a particular task.

In this paper, we use multi-task learning framework to learn ranking functions for several markets. Each market

CIKM'09, November 2–6, 2009, Hong Kong, China.

is considered as a separate task. In particular, we propose a boosting framework that adaptively learns transferable representations called *super-features* from multiple tasks. We develop an algorithm that adaptively learns the super-features among multiple tasks in a stage-wise manner similar to that used in gradient boosting [5]. At each iteration, a super-feature is constructed based on the training data from all the tasks and the corresponding coefficients are then learned with respect to each task independently, allowing us to account for differences between tasks. Our experiments show general improvements in search relevance using multi-task learning, not only when we combine markets in the same language or region, but also when we combine markets in different languages and regions.

In the next sections, we will first describe previous studies on learning to rank and multi-task learning. Then our method and experimental results will be presented in detail. Finally, conclusions and future work will be summarized.

2. PRIOR WORK

2.1 Learning to Rank

In recent years, ranking problem is frequently formulated as a supervised machine learning problem. These learning-to-rank approaches are capable of combining different features to train ranking functions. For example, RankSVM [10] uses support vector machines to learn a ranking function from preference data. RankNet [3] applies neural network and gradient descent to obtain a ranking function. RankBoost [11] applies the idea of boosting to construct an efficient ranking function from a set of weak learners. The studies reported in [12] proposed a framework called GBRank using gradient descent in function spaces, which is able to learn relative preference data in web search. However, the above approaches are all proposed to learn a single ranking function for a market. In the case where the training data is limited, they cannot produce a good ranking function.

2.2 Multi-Task Learning

Multi-Task learning has been shown to be able to improve the generalization performance through exploring the common structures among multiple tasks and transfer knowledge between related tasks [4, 7]. This method can be generally classified into two groups, which target two types of common structure. The first family of approaches assumes that all functions to be learned for the tasks are similar to each other with respect to some norm [2,7]. In these methods, the common structures are specified through selecting a proper norm to measure the similarity of the functions. The second family of approaches assume that common structures can also be represented by super-features shared among multiple tasks [1,4]. Super-features can be in the form of units in the hidden layers of neural networks [9] or linear combinations of the original features [1]. However, all these studies are based on the assumption that these common structures have particular parametric forms. This assumption makes these methods less flexible to deal with the case where the common structures shared by tasks have more complex forms. In the case of search engines, we do not know what forms should be used for ranking functions. Therefore, a more flexible learning method without such assumption is desired, so that the proper feature structures can be discovered during the learning process.

3. MULTI-TASK LEARNING FOR RANKING FUNCTIONS

3.1 Problem Formulation

We consider T learning tasks with common input-output space $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is a feature space and the output space $\mathcal{Y}$ is the real line $\mathcal{R}$. Suppose that the t-th task has N_t labeled training data:

$$S_t = \{(x_{t1}, y_{t1}), \ldots, (x_{tN_t}, y_{tN_t})\}$$

that are i.i.d. samples from a distribution $\mathcal{P}_t$ over $\mathcal{X} \times \mathcal{Y}$. The goal is to obtain a function $h_t : \mathcal{X} \to \mathcal{Y}$ for each task t that can predict the label y of unseen x. In the context of web ranking, the training set S_1 for Task 1 may contain the labeled training data for a English search engine market and the training set S_2 for Task 2 contains the labeled training data for a Chinese market and so on.

We assume that there is a loss function $L_t(y, h_t(x))$ for each task t. For this task, the empirical risk $\mathcal{R}_t(h_t)$ is defined as the sum of loss over the training set of this task:

$$\mathcal{R}_t(h_t) \quad = \quad \sum_{i=1}^{N_t} L_t(y, h_t(x))$$

The empirical risk of different tasks is then combined into one unified objective function in order to learn them simultaneously:

$$\mathcal{R}(h_1, h_2, \ldots, h_T) = \sum_{t=1}^{T} \mathcal{R}_t(h_t) = \sum_{t=1}^{T} \sum_{i=1}^{N_t} L_t(y_{ti}, h_t(x_{ti}))$$

In multi-task learning the distributions $\mathcal{P}_t$ can be different for different tasks, so it is important to transfer knowledge among different tasks. Therefore, the connections among different tasks should be properly modeled. In our approach, we assume that tasks share some common internal representations. Specifically, we assume that all h_t's linearly depend on a common set of *super-features*:

$$g(x) = (g^{(1)}(x), g^{(2)}(x), \ldots, g^{(M)}(x))$$

However, these super-features cannot be predetermined, and they depend on the original feature set in a more complex way. We will define later an algorithm, which tries to determine them automatically.

Formally, a ranking function is defined as follows:

$$h_t(x) = \sum_m w_t^{(m)} g^{(m)}(x) = w_t^T g(x), \quad t = 1, 2, \ldots, T$$

where $w_t = [w_t^{(1)}, \ldots, w_t^{(M)}]$ are the linear coefficients of the super-features for task t. Super-features $g^{(m)} : \mathcal{X} \to \mathcal{R}$ are shared among all tasks. Therefore, our goal is to learn all the tasks simultaneously via minimizing this objective function:

$$\underset{w_1, \ldots, w_T, g}{\operatorname{argmin}} \sum_{t=1}^{T} \sum_{i=1}^{N_t} L_t(y_{ti}, \sum_m w_t^{(m)} g^{(m)}(x_t i))) \qquad (1)$$

In the above optimization problem, we combine estimating the function $h_t(x)$ for each task with learning the super-features $g^{(1)}(x), g^{(2)}, \ldots, g^{(M)}$ that are shared among tasks. In this paper, we use gradient boosting trees [6] to represent the super-features.

3.2 A Multi-task Learning Algorithm

Our goal is to construct a function:

$$h_t(x) = \sum_{m=1}^{M} w_t^{(m)} g^{(m)}(x)$$

for each task t such that the objective function defined in Eqn (1) is minimized. Generally, it is difficult to optimize the problem in Eqn (1) directly. Therefore, we propose to learn the super-features $g^{(m)}(x)$ and their coefficients $w_t^{(m)}$ in a stage-wise manner, i.e. we first try to determine a super-feature according to all the training data, and then estimate the coefficient of the super-feature for each task. The two stages are performed iteratively until convergence.

More specifically, the super-feature $g^{(m)}$ and its coefficient $w_t^{(m)}$ at each iteration m are determined such that:

$$(g^{(m)}, \{w_i^{(m)}\}) = \text{argmin} \sum_t \mathcal{R}_t(h_t^{(m)} + w_t^{(m)} g^{(m)}) \quad (2)$$

where $h_t^{(m)}$ is the estimation from the previous iteration.

The problem in Eqn (2) can be solved through alternating optimization. The algorithm optimizes Eqn (2) by alternatively performing the following two steps. We first optimize Eqn (2) with respect to $g^{(m)}$ with $w_i^{(m)}$ fixed. Formally, we solve the following problem (step 1):

$$g^{(m)} = \underset{g \in \mathcal{C}}{\text{argmin}} \sum_t \mathcal{R}_t(h_t^{(m)} + w_t^{(m)} g) \quad (3)$$

Then $w_1^{(m)}, \ldots, w_T^{(m)}$ is obtained by optimizing Eqn (2) with $g^{(m)}$ fixed. Since coefficient $w_t^{(m)}$ depends on $\mathcal{R}_t(h_t)$, we can solve $w_t^{(m)}$ with respect to each task respectively (step 2):

$$w_t^{(m)} = \underset{w}{\text{argmin}}\, \mathcal{R}_t(h_t^{(m)} + w g^{(m)}) \quad (4)$$

In our case of learning to rank for a search engine, we use the squared loss $L_t(y, \hat{y}) = (y - \hat{y})^2$. Substitute $L_t(y, \hat{y})$ into Eqn (1), we have the following Eqn:

$$\underset{w_1,\ldots,w_T,g}{\text{argmin}} \sum_{t=1}^{T} \sum_{i=1}^{N_t} (y_{ti} - w_t^T g(x_{ti}))^2 \quad (5)$$

Then the problem of Eqn (3) becomes:

$$\begin{aligned} & \text{argmin}_{g \in \mathcal{C}} \textstyle\sum_{t=1}^{T} \sum_{i=1}^{N_t} \left(y_{ti} - h_t^{(m)}(x_{ti}) - w_t^{(m)} g(x_{ti}) \right)^2 \\ = \quad & \text{argmin}_{g \in \mathcal{C}} \textstyle\sum_{t=1}^{T} \sum_{i=1}^{N_t} (w_t^{(m)})^2 \left(\frac{y_{ti} - h_t^{(m)}(x_{ti})}{w_t^{(m)}} - g(x_{ti}) \right)^2 \end{aligned}$$

From the above equation, we can see that $g^{(m)}$ is obtained by solving a weighted regression problem: $g^{(m)}$ should fit the training data:

$$\left\{ \left(x_{ti}, \frac{y_{ti} - h_t^{(m)}(x_{ti})}{w_t^{(m)}}, (w_t^{(m)})^2 \right) \; | t = 1, \ldots, T, \quad i = 1, \ldots, N_t \right\}$$

in which the three elements are respectively the feature vector, the target value and the weight of training examples.

Once $g^{(m)}$ has been estimated, the linear coefficients $w_t^{(m)}$ can be determined by solving:

$$w_t^{(m)} = \underset{w}{\text{argmin}} \sum_{i=1}^{N_t} \left(y_{ti} - h_t^{(m)}(x_{ti}) - w g_t^{(m)}(x_{ti}) \right)^2$$

In this case, we have a closed form solution for $w_t^{(m)}$:

$$w_t^{(m)} = \frac{\sum_{i=1}^{N_t} g_t^{(m)}(x_{ti})(y_{ti} - h_t^{(m)}(x_{ti}))}{\sum_{i=1}^{N_t} (g_t^{(m)}(x_{ti}))^2}$$

In principle, any weighted regression algorithm can be applied to fit $g^{(m)}$. In this paper, we use gradient boosting trees as the base learner.

4. EXPERIMENTS

In the following series of experiments, we will examine the following questions: 1) We have several small search engine markets with limited training data. By exploring the training data within multi-task learning framework, can each market benefit from the common super-features extracted? 2) Can the transfer learning methodology be applied in the same manner to markets in the same language as well as in different languages?

4.1 Experimental setting

In order to test the approach on realistic data, we use data from a search engine in our experiments. Document relevance is judged by human editors. Human judgments are organized into sets: each set contains the judgments for around 1000 queries and their associated documents using corresponding relevance scores. Each query-document pair is represented by a feature vector, and features can be generalized into 3 types: 1) Query-based feature, which depends on the query only; 2) Document-based feature, which depends on the document only; 3) Query-document-based feature, which depends on the relations between the query and the document.

In our experiments, we consider 3 markets in two languages. For each task, we will use up to 4 sets of judgments as training data and another 2 sets of judgments as testing data. These sets are determined randomly. Here we use these data to test different scenarios by varying the number of training data sets: smaller tasks are simulated by using less training data and larger tasks are simulated by using more training data. A number of parameters, such as the number of trees and the number of nodes in each tree, are fixed according to our previous experience. In our experiments, we use DCG-5 [8] as our evaluation metric, t-test is also performed for statistical significance.

4.2 Combining different tasks

One of our goals of using multi-task learning is to create better ranking functions for multiple tasks when they have very limited training data. It is expected that the super-features learnt from the data of both tasks can better reflect the desired ranking functions than the features learnt for each task separately. In this series of tests, we use multi-task learning on two groups of tasks: Task1 & Task2 in the same language, Task2 & Task3 in two different languages. In each run, we will use the same amount of training data from each task, from 1 to 4 sets of human judgments. Our goal is to see if the resulting ranking functions are better than the models trained separately on each task, and how the size of training data and the language differences impact on the learned ranking functions.

In Table 1 and 2, we show the results obtained by using the following models: Dedicated models (Ded): the models trained on data from the target task only; Combined

Table 1: Combining different tasks in same language (DCG5, "*" statistical significance p<0.05)

T1+	T1			T2		
T2	Ded	Comb	MT	Ded	Comb	MT
1	7.264	7.338	7.378	10.113	10.606*	10.695*
2	7.145	7.315	7.561*	10.437	10.526	10.641*
3	7.438	7.555	7.587	10.573	10.626	10.803*
4	7.378	7.535	7.720*	10.639	10.705	10.862*

models (Comb): the models trained on combined training data by grouping different tasks together; Multi-task learning models (MT): the models trained by multi-task learning technology. The models on combined training data aim to simulate the simple combination approaches.

First, on dedicated models, we observe that in general, when the amount of training data increases, the performance of dedicated models usually increases. This is consistent with the observations from other studies that the quality of the learnt model strongly depends on the amount of training data. However, we also observe that when the amount of training data increases, the dedicated models are not always improved. For example, when set 2 or 4 is added for Task1, DCG is decreased slightly. This shows that the final effectiveness depends on not only the quantity, but also the quality of training data.

For models trained on combined data, we can also see that when more data are used, the models usually become better. However, as for dedicated models, the quality of the training data can strongly influence the performance of combined models. For example, when data set 2 is added to Task1, the resulting Comb model using combined data performs worse on Task1 than with data set 1 alone. In addition, the same combined model also performed worse on Task2 test data. This observation confirms that simple combination of training data will not necessarily produce training data of better quality.

In contrast, MT models are more resistant to the variation of quality of a particular set of training data. For example, in the cases mentioned above, while the DCG of both dedicated and combined models decreased, the DCG of the MT models increased. This comparison indicates that multi-task learning is more capable of capturing the true common features from different tasks than the simple data-combination approach. This makes it less subject to the particular problems in some training data from a particular market. In addition, we also observe that several improvements using MT models are statistically significant, in particular, in the case of T1+T2 for the same language.

Our experiments also show that transfer learning can be done between tasks in the same language (see Table 1) as well as in different languages (see Table 2). In the first case, the improvements are generally larger than in the second case. This can be explained by the fact that the characteristics of relevant documents in two different languages have larger differences. While our multi-task learning is able to account for some differences between tasks, the tasks themselves can be more different in nature, thus less common features can be extracted by our approach. Nevertheless, in any case, we still notice that MT models are always better than combined models as well as the dedicated models. This shows that multi-task learning can also be used to take advantage of training data from other source markets in different languages.

Table 2: Combining different tasks in different languages (DCG5, "*" statistical significance p<0.05)

T3+	T3			T2		
T2	Ded	Comb	MT	Ded	Comb	MT
1	10.084	10.368*	10.369*	10.113	10.367	10.395*
2	10.474	10.499	10.569	10.437	10.464	10.618
3	10.571	10.499	10.625	10.573	10.498	10.638
4	10.575	10.552	10.681	10.639	10.678	10.733

5. CONCLUSIONS

In this paper, we adapted a multi-task learning method to ranking problem in order to take advantage of the training data from different markets. The principle is to extract the common super-features from data of different markets, and then to learn appropriate coefficients of them for different markets. This allows us to recognize the common characteristics across markets, while taking into account the possible differences between them. To solve the underlying general optimization problem, we proposed a stage-wise approach, which tries to optimize first a super-feature and then its corresponding coefficients iteratively. We have tested the proposed approach on several sets of research data from a search engine. Our results showed that multi-task learning can produce models that are significantly better than the dedicated models as well as the data combined model. To our knowledge, this is the first study that shows the high potential benefits of multi-task learning method for search engines on multiple markets.

6. REFERENCES

[1] A. Argyriou, T. Evgeniou and M. Pontil. Multi-task feature learning. *Advances in Neural Information Processing Systems 19*, pages 41-48. MIT Press, Cambridge, 2007.

[2] B. Bakker and T. Heskes. Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research*, 4:83-99, 2003.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine learning*, 2005.

[4] J. Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28(1):7-39, 1997.

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

[6] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38, 2002.

[7] K. Yu, V. Tresp and A. Schwaighofer. Learning gaussian processes from multiple tasks. *ICML*, volume 119 of *ACM International Conference Proceeding Series*, pages 1012-1019, 2005.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. ACM Transactions on *Information Systems*, 20, 422-446. 2002

[9] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41-75, 1997.

[10] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of ACM SIGKDD*, 2002.

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

[12] Z. Zheng, K. Chen, G. Sun and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th ACM SIGIR conference*, 2007.