Learning a Ranking from Pairwise Preferences

Ben Carterette
carteret@cs.umass.edu

Desislava Petkova
petkova@cs.umass.edu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003

ABSTRACT

We introduce a novel approach to combining rankings from multiple retrieval systems. We use a logistic regression model or an SVM to learn a ranking from pairwise document preferences. Our approach requires no training data or relevance scores, and outperforms a popular voting algorithm.

Categories and Subject Descriptors: H.3.3 Information Search and Retrieval: Search Process

General Terms: Algorithms, Design, Experimentation, Performance

Keywords: information retrieval, data fusion

1. INTRODUCTION

It has been shown that relevant documents not retrieved by one system might be successfully retrieved by another, and as a result, combining multiple rankings can produce a better new ranking. Previous methods have used a linear combination of document scores [2], voting algorithms based on preference rankings [1], or taking the maximum, minimum, or average of a set of scores [3, 5]. We explore a statistical method based on pairwise preferences of documents. Specifically, given a ranking of documents $d_1 \succ d_2 \succ d_3 \cdots$, where $i \succ j$ means i is ranked above j, we extract all document pairs (d_i, d_j) and assume that if $d_i \succ d_j$, then d_i is preferred to d_j. For example, if one ranking is d_1, d_2, d_3, d_4, we say that system prefers d_1 to d_2, d_3, and d_4; it prefers d_2 to d_3 and d_4, and it prefers d_3 to d_4.

2. LOGISTIC REGRESSION MODEL

Suppose documents have a relevance weight θ_i such that a large θ_i means the document is of "high relevance" and a low θ_i means the document is of "low relevance". The best possible ranking of documents would be in descending order of the weights. A retrieval system can be seen as sampling from a relevance distribution $p_i(\theta_i)$ to estimate the weight for each document, and then ordering accordingly.

SIGIR'06, August 6–10, 2006, Seattle, Washington, USA.

Under this assumption, given two documents d_i and d_j, the probability that d_i is ranked higher than d_j is $P(\theta_i \geq \theta_j)$, where P is the cumulative density function of $p(\theta) = p_1(\theta_1) \ldots p_n(\theta_n)$. Given a set of pairwise preferences, the MLE ranking is by the parameter vector (θ_i) which maximizes the likelihood of the set. For all pairs of documents d_i, d_j, let $y_i = 1$ if $d_i \succ d_j$ and $y_i = 0$ if $d_j \succ d_i$. The likelihood is defined as:

$$L(\theta) = \prod_{(i,j)} [P(\theta_i \geq \theta_j)]^{y_i} [1 - P(\theta_i \geq \theta_j)]^{1-y_i}$$

For simplicity we assume that the relevance weights are distributed normally with mean θ_i and variance $\frac{1}{2}$. Then, the likelihood can be simplified to a logistic regression with probit link function:

$$L(\theta) = \prod_{(i,j)} \Phi(\theta_i - \theta_j)^{y_i} (1 - \Phi(\theta_i - \theta_j))^{1-y_i}$$

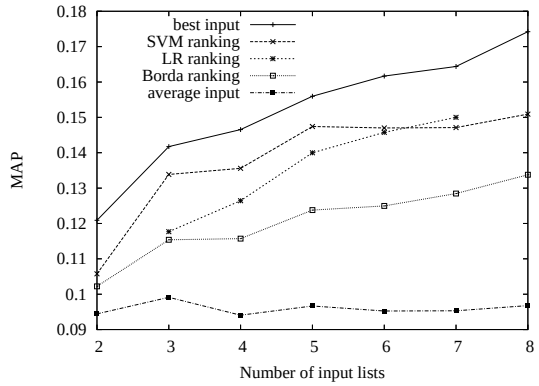
where Φ is the normal cumulative distribution. We can find θ_{MLE} using a mathematical toolkit such as `R` or Matlab. We adapted this model from Mease [6].

With just one system as input, the resulting ranking will be the same as the original. With more than one ranking, documents that are consistently ranked highly are more likely to be relevant, and documents that are ranked higher than documents that are consistently ranked highly are even more likely to be relevant, even if they make fewer appearances in the ranked lists.

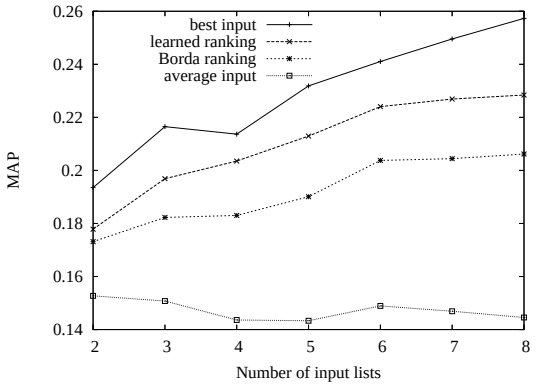
3. EXPERIMENTS

We sampled runs randomly from all 74 retrieval runs submitted to the ad hoc track of TREC-6. For each topic, we extracted all document pairwise preferences from the top 20 documents retrieved by each system. We then found the parameter values that maximized the likelihood function above. We compared the resulting ranking to the set of input rankings. On average the resulting ranking was better than all of the input rankings except the best one, i.e. it is expected that combining rankings will be better than selecting a ranking at random. Figure 1(a) shows results.

A baseline algorithm that, like ours, requires no training and no document scores is the Borda count voting algorithm [1]. A Borda count for a document is simply the number of documents ranked below it. Documents are ranked by their Borda count. The logistic regression model outperforms this algorithm.



(a) Top 20 documents.



(b) Top 100 documents.

Figure 1: Learning a new ranking from randomly-chosen input systems. Both figures show the MAPs of the best and average input ranking as well as our learned logistic regression (LR), SVM, and Borda count rankings. MAP was computed after truncating the lists, then averaged over 100 trials.

4. SVM MODEL

Logistic regression can be interpreted as learning a decision hyperplane. A support vector machine (SVM) can also learn decision hyperplanes, and it has two advantages over logistic regression: first, it makes no assumption about the distribution of relevance, and second, linear kernel SVMs can be optimized more efficiently than the logistic regression likelihood function.

To learn the SVM decision hyperplane, we solve the optimization problem:

$$\begin{aligned} \min \quad & \tfrac{1}{2}||\theta||^2 + C\sum \xi_k \\ \text{subject to} \quad & y_k(\langle \theta, x_k \rangle + b) \geq 1 - \xi_k;\ \xi_k \geq 0 \end{aligned}$$

Documents are ranked by the θ that minimizes this. Here x_k is a vector of length n associated with a pair of documents such that if $d_i \succ d_j$, then $y_k = 1, x_k[d_i] = 1, x_k[d_j] = -1$, and everything else is 0.

5. EXPERIMENTS

We again take random samples of TREC-6 runs and truncate them after the top 20 documents. With the SVM, the learned ranking was on average the best or second best of all input rankings. The results are shown in Figure 1(a). The SVM outperforms both the logistic regression and Borda models.

Linear SVM optimization is very efficient using algorithms such as Keerthi and DeCoste's [4], so the SVM model scales better than the logistic regression model. We can go deeper in the list than we can with logistic regression. Figure 1(b) shows the result of truncating after 100 documents retrieved by each system.

We can reduce the amount of data by randomly sampling document pairs. We find that we can reduce the number of preferences by as much as 75% with no noticible performance degradation.

6. CONCLUSION

We can use statistical estimation models to learn a new ranking of documents given other rankings with no training data. Our models do not use scores, making them more flexible, and they empirically outperform the Borda count voting algorithm.

Our models can be enhanced when training data is available. Given some relevance information, we can exclude any preference that contradicts it. For example, if we know d_i is relevant, d_i should be preferred to everything else; if d_i is nonrelevant, it should not be preferred to anything. Given some knowledge of relative system quality, we can weight preferences according to how good the system that generated them is. A simple weighting scheme is to scale to an integer value n and then duplicate preferences n times. Preliminary experiments show improved rankings when some training data is available.

7. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023 and through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010. Any opinions, findings, and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

8. REFERENCES

[1] J. Aslam and M. Montague. Models for Metasearch. In *Proceedings of SIGIR*, pages 275–285, 2001.

[2] B. Bartell, G. Cottrell, and R. Belew. Automatic Combination of Multiple Ranked Retrieval Systems. In *Proceedings of SIGIR*, pages 173–181, 1994.

[3] E. Fox and J. Shaw. Combination of Multiple Searches. In *Proceedings of TREC-2*.

[4] S. Keerthi and D. DeCoste. A Modified Finite Newton Method for Fast Solution of Large Scale Linear SVMs. *Journal of Machine Learning Research*, pages 341–362, Mar 2005.

[5] R. Manmatha and H. Sever. A Formal Approach to Score Normalization for Metasearch. In *Proceedings of HLT*, pages 88–93, 2002.

[6] D. Mease. A Penalized Maximum Likelihood Approach for the Ranking of College Football Teams Independent of Victory Margins. *The American Statistician*, Nov 2003.