

Learning to Rank: A Machine Learning Approach to Static Ranking

Li-Tal Mashiach

049011 Algorithms for Large Data Sets

July 9, 2006

1 Introduction

Over the last decade, the Web has grown exponentially in size. Unfortunately, the number of incorrect, spamming, and malicious sites has also grown rapidly. Despite of that, users continue to rely on the search engines to separate the good from the bad, and rank results in such a way the best pages are suggested first. The probably most prominent ranking method is PageRank [4]. Although Google, the most popular search engine, ranking algorithm originally based on the PageRank algorithm, there has recently been work showing that PageRank may not perform any better than other simple measures on certain tasks. For example Amento et al. [1] found that simple features, such as the number of pages on a site, performed as well as PageRank.

We will introduce and discuss two solutions, RankNet [2] and fRank [5], which follow a line of research completely different from PageRank, Hits [3] or other similar ranking algorithms which are based on link analysis. In [2] Burges *et al.* proposed RankNet, a probabilistic cost for training systems to learn ranking functions using pairs of training examples. In [5] Brill *et al.* combined all kinds of Web page's features using RankNet to achieve a ranking system, called fRank (feature Rank), that according to their experiments, is significantly better than PageRank.

2 RankNet

Much work in machine learning has been done on the problems of classification and regression. Lets describe formally the classification problem. Let $X = \{x_i\}$ be a collection of feature vectors (typically, a feature is any real valued number), and $Y = \{y_i\}$ be a collection of associated classes, where y_i is the class of the object described by feature vector x_i . The classification problem is to learn a function f that maps $y_i = f(x_i)$, for all i . When y_i is real-valued as well, this is called regression. Static ranking can be seen as a regression problem. If we let x_i represent features of page i , and y_i be the rank of

each page, we could learn a regression function that mapped each page's features to their rank. However, this over-constrains the problem we wish to solve. All we really care about is the order of the pages, not the actual value assigned to them.

RankNet [2] is a ranking machine learning algorithm, which was adopted in August 2005 by Microsoft for its MSN search engine. The approach in RankNet is that all we really care about is the order of the items. Therefore RankNet attempts to optimize the ordering of the items, rather than the value assigned to them. The learning algorithm is given a set of pairs of items $Z = \langle i, j \rangle$, together with target probabilities \overline{P}_{ij} that item i is to be ranked higher than item j .

RankNet uses a neural network for the learning. The network gets Web page's features as input data, propagates it through the network that contains different weights to each input, and produces output value. RankNet uses the cost entropy function as a cost function, But while a typical neural network cost function is based on the difference between the network output and the desired output, the RankNet cost function is based on the difference between a pair of network outputs. RankNet attempts to minimize the value of the cost function by adjusting each weight in the network according to the gradient of the cost function with respect to that weight.

The paper shows that RankNet gives excellent performance on a real world ranking problems with large amount of data. Comparing RankNet to other learning systems clearly shows the benefit of using pair-based cost function.

3 fRank

The paper [5] proposes fRank, which uses RankNet and a set of features to learn a ranking function for Web pages. The set of features includes PageRank, and other features divided into the next categories:

- Popularity - The actual popularity of a Web page, measured as the number of times that it has been visited by users over some period of time. Such data may be archived in 3 ways: from users who have installed the MSN toolbar, from proxy logs and from record of search engines of which results their users clicked on. It includes features such as the number of times a page was viewed and the number of times any page in the domain was viewed.
- Anchor text and in links - These features are based on the information associated with links to the page in question. It includes features such as the total amount of text in links pointing to the page, the number of unique words in that text, etc.
- Page - This category consists of features which may be determined by looking at the page (and its URL) alone. It includes features such as the number of words in the body, the frequency of the most common term, etc.
- Domain - This category contains features that are computed as averages across all pages in the domain. For example the average number of out links on any page and the average PageRank.

The most interesting parts of the paper are the experiments and the results parts. The authors used human judgments to define the correct ordering. For each query, rating

was assigned manually to a number of results. The queries were selected by randomly choosing queries from among those issued to the MSN search engine. To convert the data from query-dependent to query-independent, they simply removed the query, taking the maximum over judgments for a URL that appears in more than one query.

The authors showed that fRank significantly outperforms PageRank for the purpose of static ranking, and more than that, every single feature set individually gets more accurate results than PageRank. They found that the Page feature set is the most important out of all feature sets, and the second most important is the popularity feature set. This is surprising because Page features are not depend on the overall graph structure of the Web. This is contrary to the common belief that the Web graph structure is the key to finding a good static ranking of Web pages. The last interesting result was that by collecting more popularity data, fRank's performance can continue to improve. The relation is logarithmic: doubling the amount of popularity data provides a constant improvement in pairwise accuracy.

4 Discussion

RankNet seems to be a good method to learn any ranking function from all fields. I will concentrate on the second paper in my discussion.

- To train and test fRank, human judges were used. When the judges ranked the Web pages they did it in a context of queries. It is unclear to me how can they use rank that is query dependent to train query independent rank function. for my understanding the training for static ranking cannot be depend on queries in order to get accurate results.
- The results of the experiments show the importance of the popularity features. In my opinion statistics information of popularity will become significant direction for search engines. Information such as amount of visiting of a Web page, the time of stay, the frequency users return to the page, the way users leave the page - by clicking back or by in link and so on will be collected and will be used for ranking.
- Although the experiments showed that Page features had the highest performance, in particular better than PageRank, PageRank is much more protected from spams in compare to Page features. The web page writer can influence directly on all Page features, but it will be much harder for him to influence the Web graph structure. Unfortunately, the paper does not explore that issue, which makes it hard to interpret their findings.
- fRank method, with all of the popularity features, can only be applied to pages that have already been crawled. It is impossible to use fRank for directing the crawler.

5 Future work - Predict Popularity Rank

As a future work I will suggest a new ranking measure - Predict Popularity Rank (PP-Rank). The idea in general is to use the popularity data to train a machine to predict

popularity of Web pages. PP-Rank will keep using the rest of the feature sets, and even expend them, especially the Page feature set. The main difference from fRank is that fRank uses human judgments to define the correct ordering, which as i mention, seems problematic in the case of static ranking learning. PP-Rank will use popularity features to define the correct ordering.

5.1 Popularity data for training

In order to train and evaluate the quality of the static ranking, we need to define the correct ordering for a set of pages. We would like to order the Web pages according to their predicted popularity, with the thought that popular pages are also relevant. For that we need to define what makes a Web page popular. we can think about a set of popularity measures that a combination of them will give us the correct ordering we are looking. Here are Examples of such measures:

- Amount of visits
- How long users stay in the page
- Did they leave by clicking back or by clicking a link.

Those measures should be normalize for each user according to his pattern of surfing in the internet. For example, user that visits once a day the Web page X but X is the only page he visits should affect more than user that visits once a day the Web page X but visits other thousands of pages a day. In the same way user can stay in the page for an hour, but this is his usual behavior, and it should be count differently than a user that stays in the page for an hour but usually devotes few minutes for each page.

5.2 Protection from spams

Without any special care PP-Rank might be very weak against malicious users that will try to raise the popularity of sites by entering many time from many computers. To protect PP-Rank from that kind of spam we can use two methods: First of all we can limit the ability of one IP address to influence the popularity, for example, we will count only one entrance to specific site each day for each IP address. We still might have a problem incase a malicious user succeed to enter from lots of computers. In that case we can consider using group of trust users that only them will be the ones to influence the popularity measure.

5.3 Advantages

PP-Rank can predict popularity of pages that were just created. For such pages, PageRank calculation will be useless, as no page points to those new pages yet, and therefor it is impossible to identify good or bad pages and all new pages will get the same starting rank from PageRank. PP-Rank will predict the pages that are going to be popular, and while search engine that base on PageRank will raise their rank only after the web graph will react to them, search engine that uses PP-Rank can immediately recognize them and

give them high rank. As a result of that search engine that uses PP-Rank will give more accurate results, which includes new Web pages, than search engine that uses PageRank.

Another advantage of PP-Rank is that it can be a measure for directing the crawler. All features that PP-Rank uses are static features that can be calculated on a Web page that has just been crawled. High rank can direct the crawler to keep crawling pages in the close environment of that page.

Finally, as the training was according to users' preferences, the rank will not be according to what web masters find interesting (PageRank), but according to what users find interesting.

6 Conclusions

A good static ranking is an important component for search engines. The learning machine approach looks promising. As a future work, we can use the popularity features to train a machine to predict popularity.

References

- [1] B. Amento, L. Terveen, and W. Hill. Does "Authority" Mean Quality? Predicting Expert Quality Ratings of Web Documents. In *Proceedings of the 23th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000.
- [2] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, , and G. Hullender. Learning to Rank Using Gradient Descent. In *ICML*, pages 89–96, 2005.
- [3] Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [4] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.
- [5] Matthew Richardson, amit Prakash, and Eric Brill. Beyond PageRank: Machine Learning for Static Ranking. In *Proceedings of the 15th International World-Wide Web Conference (WWW)*, 2006.