World Applied Sciences Journal 6 (1): 06-15, 2009
ISSN 1818-4952

A Neural Framework for Web Ranking Using Combination of Content and Context Features

Amir Hosein Keyhanipour, Maryam Piroozmand and Kambiz Badie

IT Research Faculty, Iran Telecommunication Research Center (ITRC), Tehran, Iran

Abstract: Containing enormous amounts of various types of data, web has become the main source for finding the desired information. Meanwhile retrieving the desired information in such a vast heterogeneous environment is much difficult. This situation has led to a drastic increase in the popularity of internet search engines. Undoubtedly, designing both efficient and effective ranking strategies as the basic core of web information retrieval systems are unavoidable. Unfortunately most of the proposed ranking algorithms do not work very well over general datasets because of their fixed configurations. Many of these algorithms also suffer from their computational costs. Regarding these shortcoming, in this paper, a new ranking framework named "NNRank" is proposed which uses the primitive features of web documents from the categories of content and context using an artificial neural network. The neural networks selected in our approach is a radial basis function or a principle component analysis neural network which due to their high convergence rate, have the capability to exhibit a high performance with a limited number of features. Experimental results based on TREC 2004 gathered in Microsoft LETOR dataset, indicate a noticeable enhancement comparing to the well-known ranking algorithms such as TF-IDF, PageRank and HITS. The results are also comparable with those of BM25.

Key words: Ranking Algorithm · Search Engine · Neural Networks · Feature Selection · LETOR, TREC

INTRODUCTION

The web has emerged as a main medium of information dissemination in recent years. According to the available statistics, about 55 billions of web pages have been available at October 2008 [1]. At any time, hundred millions of web pages are created, modified or even deleted in a fully dynamic manner by millions of publishers. To facilitate the tedious task of information discovery, there are a number of search tools such as search engines, web directories and also meta-search tools. Among these tools, search engines are the most favorite options because based on the statistics, more than 80% of online users find their desired information via search engines [2]. Studies done on the search behavior of search engines' users, depict that most of the people, tend to just click on the top of results list. In fact, the top 30 results get over 90% of search traffic and the top 10 results receive nearly 80% more traffic than those in positions 11-30 achieve [3]. This shows how important it would be to rank the retrieved results in an appropriate manner, since it implies the users' satisfaction by an information retrieval system.

Due to its importance, ranking has been the case of many investigations in the academic societies, as well as in the business market. Although many successful companies such as Google [4] have never published their undertaken algorithms, but details of many powerful techniques are publicly accessible. Generally we can categorize the available ranking models in three broad categories: content-based, hyperlink-based and hybrid algorithms.

Most ranking algorithms for web resources are classified in the first category. They mainly use similarity measures derived from the vector-space model. For instance, in *TF-IDF* algorithm, documents and users' queries are converted to vectors of words and the similarity between a specific query and a document is defined as the dot product of document's corresponding vector and query's vector [5].

The second category includes algorithms which use hyperlink features extracted from the web graph corresponding to the set of documents. As examples of these techniques, one can point out the *PageRank* [6], *HITS* [7] and recent works such as *DistanceRank* [8]. *PageRank* is proposed by the Stanford University and

Corresponding Author: Amir Hosein Keyhanipour, Faculty member, Iran Telecommunication Research Center (ITRC), Postal Address: No. 5, Zolfaghary Alley, Shadmehr Street, P.O.Box: 14569-36511. Tehran, Iran

has been sometimes undertaken by Google. *PageRank* relies on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value [9]. The *Page Rank* metric, $PR(p)$ defines the importance of a page p to be the weighted sum of its back-links.

The third category contains algorithms that use combinations of context and context features. As an example, we can refer to the "*TF-IDF of Anchor Terms*". Experiments have shown that this category has a better performance than those of the former categories [10]. In the meantime these techniques suffer from their inflexibility and computational complexity which prevents them from being widely used in practice.

We can assume a fourth category of other ranking methods which use types of other information to improve the quality of ranking. A good example of these techniques is *BM25* which directly applies some features of the users' queries in its ranking formula [11].

The rest of this paper is organized as follows: Section 2 presents a literature review of the problem and related works. Section 3 would discuss the proposed framework. Introduction of the LETOR dataset and details of the experiments and interpretation of the results are given in Section 4. Finally, the conclusions and further directions for investigation are discussed in Section 5.

Survey on Related Works on Neural Net Based Web Ranking: The use of learning approaches in general and the application of neural networks especially in development of ranking frameworks in particular has been the case of many published investigations. The aim of such techniques is the provision of a flexible strategy to overcome with the dynamic nature of web environment as well as the changes made in the user information needs. Although in this investigation we concentrate on the application of neural networks in the web ranking problem, but other techniques such as fuzzy-logic theory [12] and genetic programming methods [13] are also investigated.

On the other hand, in the literature, many learning-based ranking algorithms are proposed; typical examples include RankBoost, RankSVM and RankNet. Ranking SVM employs support vector machine (SVM) to classify object pairs in consideration of large margin rank boundaries [14]. RankBoost conducts boosting to find a combined ranking which minimizes the number of disordered pairs of objects [15]. RankNet defines "Cross Entropy" as a probabilistic cost function for mis-ranking pairs of documents. Training is done in a neural net framework via the gradient descent method [16]. As could be seen, the common point between the aforementioned algorithms is that these methods mainly construct pairs between documents and use machine learning techniques for minimizing the number of disordered pairs. However, in this investigation it is intended to propose a ranking approach based on the capabilities of the neural networks while focusing on the simplistic features of web documents. These features are assumed as information resources which will be fused via a neural network to provide an acceptable ranking. We do not intend to minimize the distance order of the documents but want to close the obtained relevancy of documents to the relevancy of the most well known approaches as much as possible. Based on this idea, in the rest of this section we will concentrate on the review of ranking algorithms proposed based on the neural networks framework.

On 1995, Caruana *et al*. proposed a neural net ranking model named as "*RankProp*" which provides a mapping of the data onto a large number of targets which reflect the desired ranking, which performs better than just regressing to the original, scaled rank values [17]. However it is not known under what conditions it converges and it does not give a probabilistic model. This work was extended by Diligenti *et al*. by the use of Error Back-Propagation mechanism to learn the final score of Web pages in a neural net framework [18].

In general, static ranking can be seen as a regression problem. If we let X_i be the set of features related to the page i and Y_i be the corresponding output value (say, the rank) for this page, we could learn a regression function that maps each page's features onto their rank. However, this situation may load some over-constrains to the problem that was wished to be resolved. The order of the pages was the main concern, not the actual value assigned to them [19].

For such cases, let $Z = \{(i, j)\}$ be a collection of pairs of items, where item i should be assigned a higher value than those of item j. The goal of the ranking problem, then, is to learn a function f such that, by using the set of features, can be able to provide a mapping onto the proper order of items.

Based on the above, Herbrich *et al*. (2000) proposed a new approach to cast the problem of learning to rank as ordinal regression, which is, learning to provide an adequate mapping from an input vector onto a member of an ordered set of numerical ranks [20].

Following that work, in 2002 a new method named *"Prank"* was proposed which used a MLP[1] network to compare the set of items and find the best relative position of them [21]. In other words, it trains a Perceptron model to directly maintain a totally-ordered set via projections. Though this method showed acceptable performance, but due to its computational cost, it has never been widely used in operation.

Another successful approach is named *"RankNet"* which by the use of a back-prop model, attempts to minimize a cost function by adjusting each weight in the network according to the gradient of the cost function with respect to that weight [22].

Other new methods are based on the use of a new neural network model called *"Graph Neural Network" (GNN)* which are able to directly process web graphs [23]. For example, Scarselli *et al*. at 2005 adapted the *GNN* model for the task of computing customized web page rank values [24, 25]. In that study, web was modeled as an asymmetric Hopfield Net and used in the search process to extract some specified properties.

It is mentionable that, in spite of the fact that the aforementioned techniques demonstrate approaches to use some features of documents to provide suitable rankings, they do not pay attention to the features themselves. Features as the representation of documents, contain specific aspects of the documents and it is a crucial task to select a set of features adequate in representation aspects.

They also suffer from the application of test on toy data. Furthermore they do not provide comparison with the well-known ranking algorithms which are widely used in the web retrieval systems.

In this paper we would use combination of simple features extracted from content as well as context of web pages to improve the practicality of the proposed framework. We also would use the TREC 2004 data embedded in the Microsoft LETOR dataset [26] as our benchmark, which gives us the opportunity to be able to straightforwardly compare our method with the others.

The Proposed Neural Framework: The aim of this study is introduce a flexible and applicable framework using simple features extracted from the content of web pages as well as their context which are generally computed from the web graph. Our motivation for this purpose was initiated via studying the structure of formulas of well known ranking algorithms such as *TF-IDF*, *PageRank*, *HITS* and *BM25*. This analysis shows that most of these formulas are constructed from very simple features of documents. This fact was our primary motivation to focus on the simple features of documents instead of the complicated ones. On the other hand, LETOR as a subset of TREC 2004 dataset which has been recently introduced by Microsoft Research Asia has extracted a large variety of features from documents in different levels of complexity.

Among the most suitable techniques which have been widely used in operation, we can point out *PageRank*, *TF-IDF* and *BM25*. As mentioned earlier, *PageRank* which was at times used in the Google search engine, deals with the analysis of web graph in an offline manner and has a relatively low complexity, though its performance is moderate. In contrast, *TF-IDF* which is based on the analysis of contents of documents, has more computational complexity while its performance is more than *PageRank*. At last, we can refer to the *BM25* algorithm which uses the features of users' queries as well as those of web documents and can therefore obtain higher performance levels and definitely more complexity, because of dealing with the users' query in a real time manner.

As shown in Figure 1, we intend to move toward the performance of well known methods through using simple features of documents in a flexible framework but with less complexity. In this Figure, it is seen that the proposed method, tends to get close to the performance interval of $[P_1, P_2]$ which is between the performance of the best known approaches and those of our desired neural framework. Meanwhile its complexity is less than those of the best known techniques (interval of $[C_1, C_2]$) that do the ranking in a real-time manner.

These circumstances directed us to organize a research on the use of these characteristics of documents in the framework of neural networks. Figure 2 shows a general schema of the proposed framework.

To achieve the above purpose, we have selected both supervised and unsupervised approaches to compare their performance. Here we will present a brief description of PCA[2] and RBF[3] networks as examples of unsupervised and supervised models.

[1]Multi-Layer Perceptron

[2]Principle Component Analysis

[3]Radial Basis Function

Radial Basis Function (RBF) networks are nonlinear hybrid networks which usually make use of Gaussian transfer functions, rather than the standard *Sigmoidal* functions that are typically employed by MLPs. In such networks, the centers and widths of the Gaussians are set by unsupervised learning rules. The supervised learning process could then be applied to the output layer. These networks tend to learn much faster than MLPs. For standard RBF's, the supervised segment of the network only needs to produce a linear combination of the output at the unsupervised layer. Therefore 0 hidden layers is the default. Adding hidden layers provides the supervised learning instead of a simple linear Perceptron.

On the other hand, Principal component analysis (PCA) networks combine unsupervised and supervised learning in the same topology. PCA is an unsupervised linear procedure that finds a set of uncorrelated features, principal components, from the input. A MLP is used to perform the nonlinear classification from these components. PCA is a data reduction method, which condenses the input data down to a few principal components. As with any data reduction method, there is the possibility of losing important input information. Here, the number of principal components selected can be a compromise between training efficiency (few PCA components) and accurate results (a large number of PCA components).

Based on aforementioned capabilities of RBF and PCA networks, we decided to concentrate on RBF and PCA networks. In the meantime, we studied the performance of MLP. However, through simulation and looking at experimental results, It was found that MLP can not be a suitable candidate.

Computer Experimentation and Analyzing the Results: In this section, first we introduce the LETOR dataset and then, the details of experimental results as well as their analysis are described in detail.

Experimental Dataset: Using machine learning techniques to find out the ranking function has been a promising research direction. However, the lack of public benchmark datasets (e.g. standard features set, relevance judgments, data partitioning and also evaluation metrics) makes the existing works difficult to be compared with each other. To resolve this problem, LETOR which is a package of benchmark data sets for "*LEarning TO Rank*", is released by Microsoft Research Asia [26]. Fortunately, it has been widely used in recent investigations [27-29].

Table 1: Sample data in LETOR dataset

Query	PageRank	TF-IDF	BM25	...	Relevancy
qid: 1	0.0023	9.23	19.31	...	1
qid: 1	0.0098	11.3	29.12	...	0
...	...	...	...	...	...
qid: 10	0.000	7.2	15.22	...	0
...	...	...	...	...	...
qid: 75	0.0076	12.23	30.1	...	1

The part of LETOR dataset which is dedicated to the TREC 2004, contains 1000 result items for 75 queries which provides 75000 pairs of query-item. Table 1 shows a view of data which is available in LETOR.

Providers of LETOR have extracted features for each query-document pair in the OHSUMED and TREC collections, which are widely used in the literature of information retrieval. The extracted features cover most of the 'standard' features discussed about in the Information Retrieval literature, including classical features (such as term frequency, inverse document frequency, *BM25* and language models for IR) and the features proposed in SIGIR papers in recent years (such as *HostRank*, *Feature propagation* and also *TopicalPageRank*). These features could be categorized in four broad types [10]:

- Low-level content-based features such as: term frequency (TF), inverse document frequency (IDF) and document length (DL), which have been extracted from different fields of documents like: body, anchor, title and URL. These are 16 features in total.
- High-level content-based features, which are 12 features, resulted from the application of the well known ranking algorithms such as *BM25* and *LMIR* applied in different parts of documents.
- Hyperlink-derived features (7 features) extracted from the context graph in which the web documents are settled.
- Hybrid features which are 8 features that refer to those features containing both content and hyperlink information, including "*hyperlink-based relevance propagation*" and "*sitemap-based relevance propagation*" [30].

Experiments with Pca and Rbf as the Proposed Networks: Experiments are based on 16 content-based features plus 7 hyperlink-based ones. To compute the desired relevancy which is the target of network, 5 features were selected which and their relevancy was aggregated using Borda-Voting algorithm [29]. This set includes

Feature#1 (BM25), Feature#14 (PageRank), Feature#27 (Sitemap based feature propagation), Feature#33 (TFIDF of Anchor) and Feature#42 (Hyperlink base feature propagation: weighted in-link). The assigned weights to these features were 0.25, 0.10, 0.25, 0.15 and 0.25 respectively. The desired relevancy is higher (at least about 10%) than those of each of the underlying ranking methods. This fact causes that the subject neural network tries to approach to these target relevancies which is higher than those of any ranking method, meanwhile due to its dynamic nature, *NNRank* may reaches to higher relevancy. Nonetheless in the worst case, its performance would reach to those of the underlying methods which are the results of the well-known rankings such as *BM25* or even a bit lower. It should be noticed that this improvement is gained by a flexible and low computational cost neural network configuration which uses a basic set of primitive features.

We have also repeated our experiments by a selected set of features including 11 simple and basic features as inputs of neural network: Features#2-5, Feature#6, Features#9-12, Feature#32 and also Feature#34. They are: DL[4] of body, DL of anchor, DL of title, DL of URL, HITS authority, IDF of body, IDF[5] of anchor, IDF of title, IDF of URL, TFIDF of body and TFIDF[6] of title.

In our experimentation, we have utilized PCA and RBF networks with one hidden layer and also 23 inputs. In the case of PCA network, the used learning rule was SangersFull which is a normalized implementation of the Hebbian learning rule [31]. For the RBF network, we used the competitive learning rule of ConscienceFull which is an intrinsic probability distribution of the input data. The selected transfer function for both networks was Hyperbolic Tangent (TanhAxon).

From the operational point of view, *NNRank* has the capability to rank the results in the real time fashion using very simple features extracted from the set of documents. These features which could be easily extracted from the set of result items are fed into the trained ranking network. Based on its training experiences, the network assigns a rank value to the corresponding document.

The main issue is that based on this simple framework, *NNRank* provides rankings which are completely comparable with those of famous ranking strategies such as *BM25*, *PageRank* and *HITS*.

The evaluation tool contained in LEOTR can output evaluation criteria such as *P@n*, *MAP* and *NDCG* for any given ranking algorithm. According to the following formula, the *P@n* criterion or Precision at *n,* measures the relevance of the top *n* results of the ranking list with respect to a given query.

$$P@n = \frac{\text{No. of relevent docs in top n results}}{\text{N}}$$

The *"Mean Average Precision" (MAP)*, is computed for a single query and is defined as the average of the *P@n* values for all relevant documents.

$$MAP(n) = \frac{\sum_{\text{n=1}}^{\text{N}} \left(\text{P@n} \times rel(n)\right)}{\text{Number of total relevant docs for this query}},$$

$$rel(n) = \begin{cases} 1, \text{ if the } n^{\text{ th}} \text{ doc is relevant} \\ 0, \text{ otherwise} \end{cases}$$

Note that *P@n* and *MAP* can only handle cases with binary judgment: “relevant” or “irrelevant”. Recently, a new evaluation metric called *Normalized Discount Cumulative Gain (NDCG)* has been proposed, which can handle multiple levels of relevancy and is defined as:

$$NDCG(n) = Z_n \sum_{j=1}^{n} \frac{2^{r_j} - 1}{\log(1 + j)}$$

In the above formula, $r(j)$ is the rating of the j^{th} document in the list and the normalization constant Z_n is chosen so that the perfect list gets the *NDCG* score of one.

In the remaining of this section, we will describe the details of obtained results as well their interpretation.

Due to the computational cost, we have limited our investigation to 20 queries and 1,000 result items for each query which is 20,000 pairs of query-item. It also should be notified that all the experiments were repeated for 1,000 episodes.

Figure 3 illustrates that the PCA implementation of *NNRank*, has higher precision than those of other ranking methods but just *BM25*. Meanwhile it exceeds all of the rankings for n>13. We can see that there is a neglect able difference between the performance of *NNRank* by using all 23 features and 11 selected features set.

[4]Document Length

[5]Inverse Document Frequency

[6]Term Frequency / Inverse Document Frequency

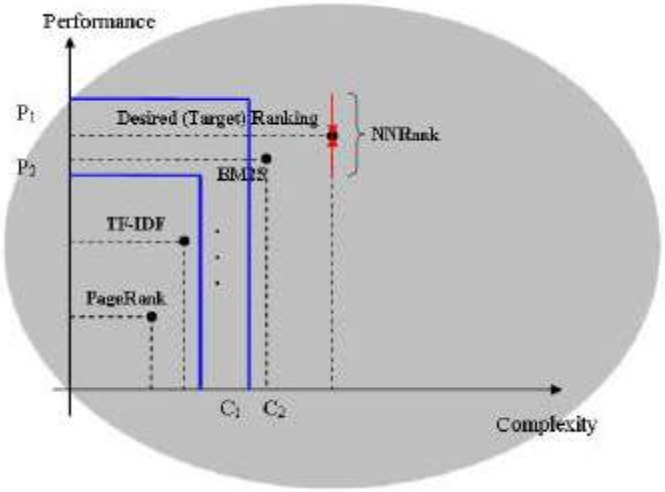

Fig. 1: Comparison of performance and complexity of the proposed techniques with those belonging to the others

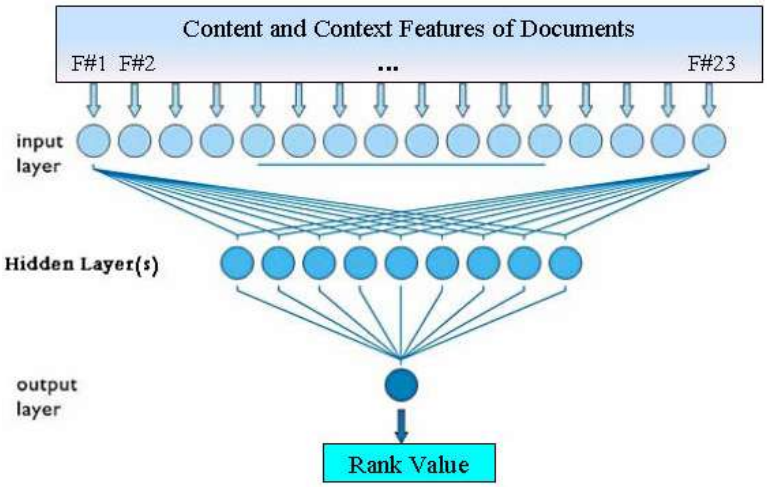

Fig. 2: Overall schema of the proposed framework

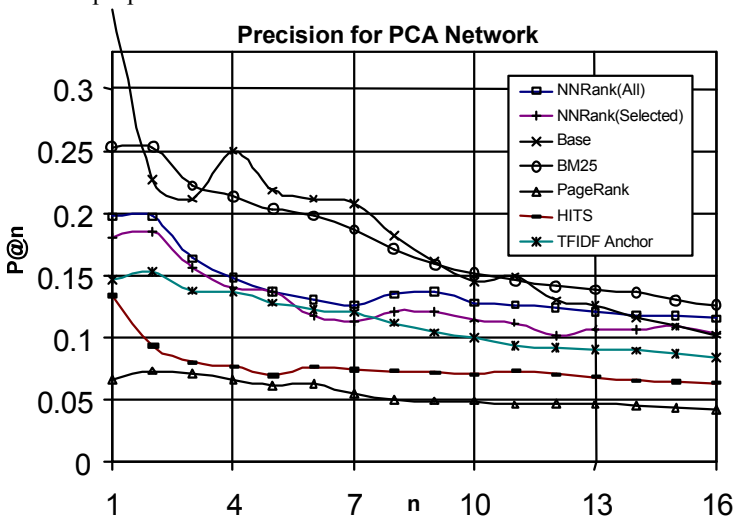

Fig. 3: Precision obtained by NNRank (PCA implementation) vs. other ranking methods

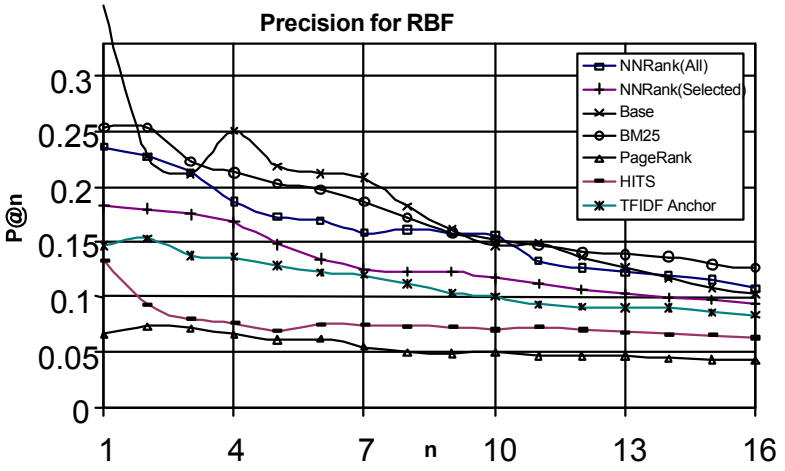

Fig. 4: Precision acquired by NNRank (RBF implementation) vs. other ranking methods

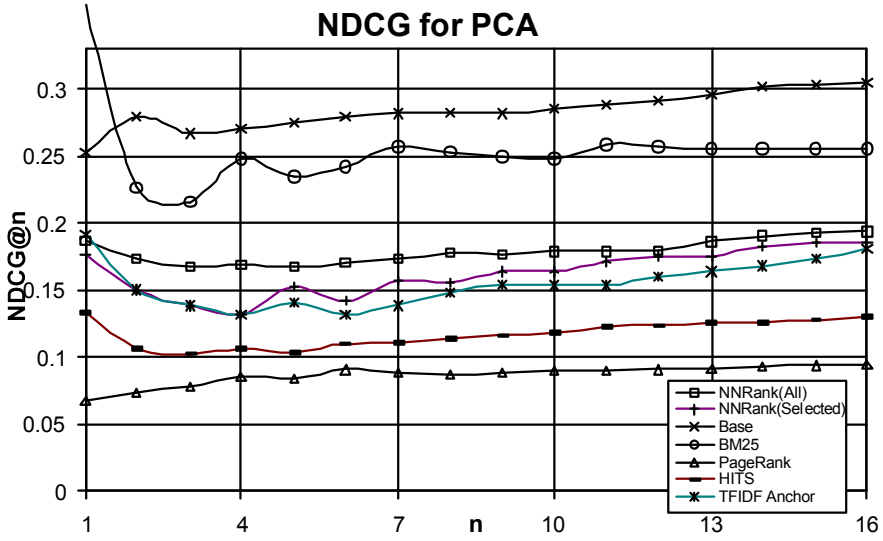

Fig. 5: NDCG values of NNRank (PCA implementation) in comparison with other ranking methods

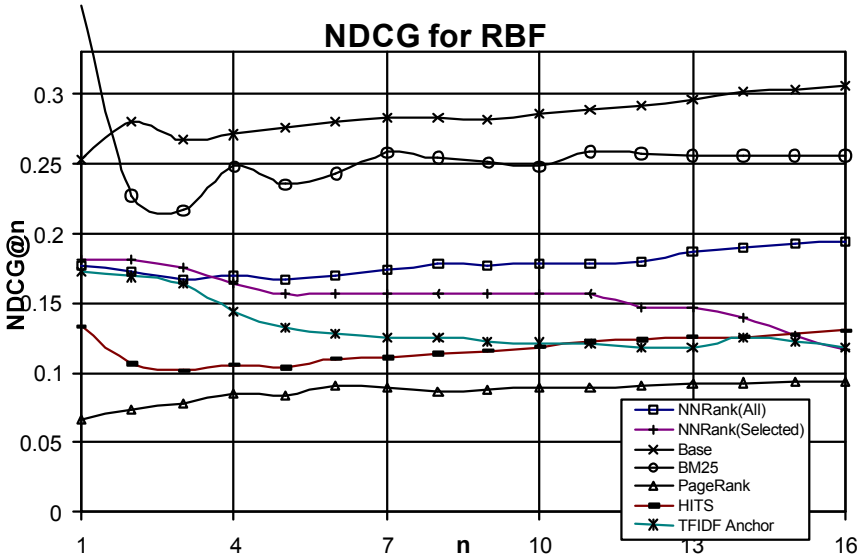

Fig. 6: NDCG values of NNRank (RBF implementation) network against other ranking methods

Table 2: MAP values of different ranking methods

Algorithm	MAP
NNRank-RBF(All)	0.146598557
NNRank-RBF(Selected)	0.138360259
NNRank-PCA(All)	0.14267908
NNRank-PCA(Selected)	0.135690285
BM25	0.181581566
Base Algorithm	0.241054597
PageRank	0.064253624
HITS	0.094470889
TF-IDF Anchor	0.14239113
TF-IDF Body	0.097537812

Performance of the *RBF* configuration of *NNRank* is shown in figure 4. We can see the same trend is repeated here, in the interim an interesting observation is that the *NNRank* (All - using all features) goes over the *"Base"* algorithm which is the aggregation of 5 selected features.

Figures 5 and 6 respectively illustrate the *NDCG@n* values corresponding to the PCA and RBF implementations of *NNRank* in comparison with those of the other algorithms.

Generally, it is evident that the RBF implementation of the *NNRank* has better *P@n* values which have fluctuations for different values of *n* in comparison with the PCA variation, which has leaded to lower *NDCG* values.

The *MAP* values of different implementations of *NNRank* and other ranking algorithms are briefly presented in Table 2. It could be easily seen that *NNRank* outperforms famous rankings such as *TF-IDF*, *HITS* and *PageRank* and is relatively comparable with *BM25* which is one of the best known techniques in the area of ranking.

These statistics reveal the plausibility of the idea of using simple and basic features of web documents. In fact, a web document could be represented as a set of content and context features. Each feature could be imagined as an information sensor which reflects the corresponding document to some extent and has a certain computational cost. The aim of NNRank is to select those features which contain the most information and also have the least computational cost. In NNRank, the feature selection has been done by the use of neural networks. As mentioned former, the target of this neural network is the aggregation of some of the best-known ranking algorithms which are integrated via an information aggregation operator named as Borad-voting. To sum up, NNRank uses the capabilities of neural networks to provide a ranking. In this regard, NNRank tries to minimize the distance between the predicted relevancy values and those obtained by the well-know ranking methods.

The results achieved, confirm that although NNRank uses a limited number of simple features to rank the web documents, its performance is comparable with those of the well-known ranking algorithms such as *BM25*, *PageRank* and *HITS*. However, NNRank can not exceed the Base algorithm because it is the combination of some of the most successful ranking techniques.

Analysis of the Experimental Results: A close look at the results indicates that many of the proposed ranking techniques have relatively poor performance in real environments. However, applying the proposed framework led us to the point that a relatively fine performance can be obtained when a hybridation of content and context features are used. The interesting fact is that rankings such as *"TF-IDF Anchor"*, which is based on hybrid features of documents' content as well as their context, do much better than the conventional methods.

Meanwhile, the overview of the results regarding the usage of basic features extracted from content and context of web documents, can lead to a promising ranking method. It can be seen that, despite the fact that *NNRank* uses a set of basic features extracted from documents, it has in reality done better than many famous ranking methods. Another noticeable finding, is the little difference between the performance of *NNRank* in its full mode employing the entire 23 features and in its reduced mode which uses only 11 selected features. As other advantages of *NNRank*, the simplicity and flexibility of the proposed method which uses simple features easily extractable from web documents, can be mentioned.

CONCLUDING REMARKS

Due to the relatively high computational cost of current ranking algorithms as well as their rigidity over different datasets, which in turn leads to a relatively low performance, we were motivated to develop a new ranking framework. In this regard, a framework named *"NNRank"* was proposed, which makes use of a combination of basic and simple features that are extracted from both the content and context of web documents.

Here the ultimate endeavor is to provide an adequate mapping from the combination of these basic features onto the desired output which is to be aligned with the product of applying an aggregation operator to top 5

ranking methods, as discussed in the paper. As an alternative for the neural network, we made use of *RBF* and *PCA* networks due to their high performance in the supervised and unsupervised modes of operation. It should however be mentioned that *RBF* implementation of *NNRank* functions a bit better than its *PCA* alternative, since it makes use of supervisory information which does not exist in the case of *PCA* configuration.

As is seen from the tentative results, the proposed *NNRank* provides a ranking which is quite comparable with the best known ranking strategies. The interesting point with respect to both *RBF* and *PCA* implementations of *NNRank* is that, these networks, due to their relatively high convergence speed, have the capability to end up a plausible convergence under the presence of a limited number of input features. Experiments done by a selected set of 11 features which were selected by trial and error, approves the above fact.

NNRank, while being benefited by a high flexibility in performance, can not behave better than *BM25*, because the later, beside the ordinary content features, makes use of the query features that are more informative than the context features.

It is seen that the ranking problem deals with a major paradigm that is the movement between full-content and full-context modes as the sources for feature extraction. Regarding the fact that in the best case, context is a full reflection of content, we need to establish a sort of balance between these two types of features.

On the other hand, precise analysis of successful techniques such as *BM25*, reveals the fact that most of these techniques not only use features that are directly derivable from content, but also use the features that could be derived indirectly. In the case of *BM25*, we can point out the query-driven features that could be imagined as the implicit presentation of their desired documents. From a general point of view, having such an insight into the ranking problem, would ultimately lead to a hybridation of features derivable from content and context which may provide more powerful ranking strategies.

As a further research work, application of feature reduction techniques to the union set of content and context features is proposed. This could be done by mapping the features set onto a semantic graph including the inter-relations between the concepts of these features and then reducing the appropriate candidates in a gradual and systematic manner just like the case of simulated annealing.

REFERENCES

1. World Wide Web Size, www.worldwidewebsize.com, October 2008.
2. Forrester Research, http://www.forrester.com, November 2005.
3. Websearchworkshop's website: www.websearchworkshop.co.uk/stats.php, December 2007.
4. Google's website: www.google.com, October 2008.
5. Baeza-Yates, R. and B. Ribeiro-Neto, 1999. Modern Information Retrieval. Addison-Wesley/ACM Press.
6. Brin, S. and L. Page, 1998. The anatomy of a large-scale hypertextual Web search engine. J. Computer Networks and ISDN Systems, 30(1-7): 107-117.
7. Kleinberg, J.M., 1999. Authoritative sources in a hyperlinked environment. J. ACM, 46(5): 604-632.
8. Zareh Bidoki, A.M. and N. Yazdani, 2008. DistanceRank: An intelligent ranking algorithm for web pages. Journal of Information Processing and Management, 44(2): 877-892.
9. Brin, S. and L. Page, 1998. The anatomy of a large-scale hypertextual Web search engine. In the Proceedings of the 7th International Conference on World Wide Web 7, pp: 107-117.
10. Qin, T., T.Y. Liu and H. Li, 2007. The TREC Datasets in LETOR. Microsoft Research Asia.
11. Robertson, S.E., S. Walker, S. Jones, M. Hancock-Beaulieu and M. Gatford, 1994. Okapi at TREC-3. In the Proceedings of the third Text REtrieval Conference, pp: 26-109.
12. Martin-Bautista, M.J., M.A. Vila and H.L. Larsen, 1999. A fuzzy genetic algorithm approach to an adaptive information retrieval agent. J. Am. Soc. Inform. Sci., 50(9): 760-771.
13. Fan, W., M.D. Gordon and P. Pathak, 2004. A generic ranking function discovery framework by genetic programming for information retrieval. J. Inform. Processing and Management, 40(4): 587-602.
14. Herbrich, R., T. Graepel and K. Obermayer, 2000. Large margin rank boundaries for ordinal regression. Book Chapter in: Advances in Large Margin Classifiers, 115-132, MIT Press.
15. Freund, Y., R. Iyer, R.E. Schapire and Y. Singer, 2003. An efficient boosting algorithm for combining preferences. J. Machine Learning Res., 4: 933-969.
16. Burges, C., T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton and G. Hullender, 2005. Learning to rank using gradient descent. In Proceedings of the 22nd International Conference on Machine Learning, pp: 89-96.

17. Caruana, R., S. Baluja and T.M. Mitchell, 1996. Using the Future to Sort Out the Present: Rankprop and Multitask Learning for Medical Risk Evaluation. J. Advances in Neural Infor. Processing Sys., 8: 959-965.
18. Diligenti, M., M. Gori and M. Maggini, 2005. Learning Web Page Scores by Error Back-Propagation. In the Proceedings of the 9th International Joint Conference on Artificial Intelligence, pp: 684-689.
19. Herbrich, R., T. Graepel and K. Obermayer, 1999. Support vector learning for ordinal regression. In the Proceedings of the 9th International Conference on Artificial Neural Networks, pp: 97-102.
20. Herbrich, R., T. Graepel and K. Obermayer, 2000. Large Margin Rank Boundaries for Ordinal Regression. Book Chapter in: Advances in Large Margin Classifiers. MIT Press.
21. Crammer, K. and Y. Singer, 2002. Pranking with ranking, Advances in Neural Information Processing Systems, 14(1): 641-647.
22. Burges, C., T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton and G. Hullender, 2005. Learning to rank using gradient descent. In the Proceedings of the 22nd International Conference on Machine Learning, pp: 89-96.
23. Gori, M., M. Hagenbuchner, F. Scarselli and A.C. Tsoi, 2004. Graphical based learning environment for pattern recognition. Lecture notes in computer science, 3138: 42-56.
24. Scarselli, F., S.L. Yong, M. Hagenbuchner and A.C. Tsoi, 2005. Adaptive page ranking with neural networks. In the Proceedings of the 14th International Conference on World Wide Web, pp: 936-937.
25. Scarselli, F., S.L. Yong, M. Gori, M. Hagenbuchner, A.C. Tsoi and M. Maggini. Graph Neural Networks for Ranking Web Pages. In the Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, pp: 666-672.
26. LETOR's website, http://research.microsoft.com/users/tyliu/LETOR, October 2008.
27. Burges, C., 2007. Learning to Rank for Web Search: Some New Directions. SIGIR Ranking Workshop.
28. Taylor, M., J. Guiver, S. Robertson and T. Minka, 2007. SoftRank: Optimising Non-Smooth Rank Metrics. SIGIR Ranking Workshop.
29. Zareh Bidoki, A.M., M. Azadnia, N. Yazdani and A.H. Keyhanipour, 2007. ComRank: A Novel Combinational Adaptive Web Ranking Algorithm. In the Proceedings of the 13th International CSI Computer Science, pp: 341-346.
30. Qin, T., T.Y. Liu, X.D. Zhang, Z. Chen and W.Y. Ma, 2005. A study of relevance propagation for web search. In the Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp: 408-415.
31. Fausett, L.V., 1994. Fundamentals of Neural Networks: Architectures, Algorithms and Applications. Prentice Hall.