

# A Framework for Web Page Rank Prediction

Elli Voudigari<sup>1</sup>, John Pavlopoulos<sup>1</sup>, and Michalis Vazirgiannis<sup>1,2\*</sup>

<sup>1</sup> Department of Informatics, Athens University of Economics and Business, Greece,  
elliv@aueb.gr, annis.pavlo@gmail.com, mvazirg@aueb.gr,

<sup>2</sup> Institut Télécom, Ecole de Télécom ParisTech, Département Informatique et Réseaux, Paris, France

**Abstract.** We propose a framework for predicting the ranking position of a Web page based on previous rankings. Assuming a set of successive top-k rankings, we learn predictors based on different methodologies. The prediction quality is quantified as the similarity between the predicted and the actual rankings. Extensive experiments were performed on real world large scale datasets for global and query-based top-k rankings, using a variety of existing similarity measures for comparing top-k ranked lists, including a novel and more strict measure introduced in this paper. The predictions are highly accurate and robust for all experimental setups and similarity measures.

**Keywords:** Rank Prediction; Data Mining; Web Mining; Artificial Intelligence.

## 1 Introduction

The World Wide Web is a highly dynamic structure continuously changing, as Web pages and hyperlinks are created, deleted or modified. Ranking of the results is a cornerstone process enabling users to effectively retrieve relevant and important information. Given the huge size of the Web graph, computing rankings of Web pages requires awesome resources-computations on matrices whose size is of the order of Web size ( $10^9$  nodes).

On the other hand the owner of the individual web page can see its ranking only in the case of the web graph by submitting queries to the owner of the graph (i.e. a search engine). Given a series of time-ordered rankings of the nodes of a graph where each bears its ranking for each time stamp, we develop learning mechanisms that enable predictions of the nodes ranking in future times. The predictions require only local feature knowledge while no global data are necessary. Specifically, an individual node can predict its ranking only knowing the values of its own ranking. In such a case the node could plan actions for optimizing its ranking in future.

In this paper we present an integrated effort for a framework towards Web page rank prediction considering different learning algorithms. We consider i)

---

\* Partially supported by the DIGITEO Chair grant LEVETONE in France and the Research Centre of the Athens University of Economics and Business, Greece.

variable order Markov Models (MMs), ii) regression models and iii) an EM based approach with Bayesian learning. The final purpose is to represent the trends and predict future rankings of Web pages. All the models are learned from timeseries datasets where each training set corresponds to pre-processed rank values of Web pages observed over time.

For all methods, prediction quality is evaluated based on the similarity between the predicted and actual ranked lists, while we focus on the top-k elements of the Web page ranked lists, as top pages are usually more important in Web search.

Preliminary work on this topic was presented in [13] and [15]. The current work significantly differs and advances previous works of the authors in the following ways: a) Refined and careful re-engineering of the MMs' parameter learning procedure by using cross validation, b) Integration and elaboration of the results of [15] in order to validate the performance comparison between regression (boosted with clustering) and MM predictors, in large scale real world datasets, c) Namely we adopt: Linear Regression, random  $1^{st}/2^{nd}/3^{rd}$  order Markov models proving the robustness of the model, d) A new top-k list similarity measure (*R-Sim*) is introduced and used for the evaluation of predictors and more importantly, e) Additional, extensive and robust experiments took place using query based on top-k lists from Yahoo! and Google Search engine.

## 2 Related Work

The ranking of query results in a Web search-engine is an important problem and has attracted significant attention in the research community.

The problem of predicting PageRank is partly addressed in [9]. It focuses on Web page classification based on URL features. Based on this, the authors perform experiments trying to make PageRank predictions using the extracted features. For this purpose, they use linear regression; however, the complexity of this approach grows linearly in proportion to the number of features. The experimental results show that PageRank prediction based on URL features does not perform very well, probably because even though they correlate very well with the subject of pages, they do not influence page's authority in the same way.

A recent approach towards page ranking prediction is presented in [13] generating Markov Models from historical ranked lists and using them for predictions.

An approach that aims at approximating PageRank values without the need of performing the computations over the entire graph is [6]. The authors propose an algorithm to incrementally compute approximations to PageRank, based on evolution of the link structure of Web graph (a set of link changes). Their experiments demonstrate that the algorithm performs well both in speed and quality and is robust to various types of link modifications. However, this requires continuous monitoring of the Web graph in order to track any link modifications. There has also been work in adaptive computation of PageRank ([8], [11]) or even estimation of PageRank scores [7].

In [10] a method called predictive ranking is proposed, aiming at estimating the Web structure based on the intuition that the crawling and consequently the ranking results are inaccurate (due to inadequate data and dangling pages). In this work, the authors do not make future rank predictions. Instead, they estimate the missing data in order to achieve more accurate rankings.

In [14] the authors suggest a new measure for ranking scientific articles, based on future citations. Based on publication time and author's name, they predict future citations and suggest a better model.

### 3 Prediction Methods

In this section, we present a framework that aims to predict the future rank position of Web pages based on their trends shown the past. Our goal is to find patterns in ranking evolution of Web pages. Given a set of successive Web graph snapshots, for each page we generate a sequence of rank change rates that indicates the trends of this page among the previous snapshots. We use these sequences of previous snapshots of the Web graph as a training set and try to predict the trends of a Web page based on previous. The remaining of this section is organized as follows: In Sect. 3.1 we train MMs of various orders and try to predict the trends of a Web page. Section 3.2 discusses an approach that uses a separate linear regression model for each web page, while Sect. 3.3 combines linear regression with clustering based on an EM probabilistic framework.

**Rank Change Rate** In order to predict future rankings of Web pages, we need to define a measure introduced in [12] suitable for measuring page rank dynamics. We briefly present its design.

Let  $G_{t_i}$  be the snapshot of the Web graph created by a crawl and  $n_{t_i} = |G_{t_i}|$  the number of Web pages at time  $t_i$ . Then,  $rank(p, t_i)$  is a function providing the ranking of a Web page  $p \in G_{t_i}$ , according to some criterion (i.e. PageRank values). Intuitively, an appropriate measure for Web pages trends is the rank change rate between two snapshots, but as the size of the Web graph constantly increases the trend measure should be comparable across different graph sizes. Thus, we utilize the normalized rank ( $nrank$ ) of a Web page, as it was defined in [12].

For a page  $p$  ranked at position  $rank(p, t_i)$ :  $nrank(p, t_i) = \frac{2 \cdot rank(p, t_i)}{n_{t_i}^2}$ , which ranges between  $2n_{t_i}^{-2}$  and  $2n_{t_i}^{-1}$ . Then, using the normalized ranks, the Rank Change Rate (*Racer*) is given by  $racer(p, t_i) = 1 - \frac{nrank(p, t_{i+1})}{nrank(p, t_i)}$ .

#### 3.1 Markov Model Learning

Markov Models (MMs) [1] have been widely used for studying and understanding stochastic processes and behave very well on modeling and predicting values in various applications. Their fundamental assumption is that the future value

depends on a number of  $m$  previous values, where  $m$  is the order of the MM. They are defined based on a set of states  $S = \{s_1, s_2, \dots, s_n\}$  and a matrix  $T$  of transition probabilities  $t_i$  each of which represents the probability that a state  $s_i$  occurs after a sequence of states.

Our goal is to represent the Web pages ranking trends across different web graph snapshots. We use the *racer* values to describe the rank change of a Web page between two snapshots and we utilize *racer* sequences to learn MMs. Obviously, stable ranking across time is represented by a zero *racer* value, while all other trends by real numbers generating a huge space of discrete values. As expected (intuitively most pages are expected to remain stable for some time irrespective to their rank at the time), the zero value has an unreasonably high frequency compared to all other values which means that all states besides the zero one should be formed by inherent ranges of values instead of a single discrete. In order to ensure equal probability for transition between any pair of states, we guaranteed equiprobable states by forming ranges with equal cumulative frequencies (showing *racer* value within the range) with each other.

In order to calculate the state number for our MMs, we computed the relative cumulative frequency of the zero *racer* state  $RF_{Racer=0}$  and used this to find the optimum number of states  $n_s = \frac{l}{RF_{Racer=0}}$ . Next, we formed  $n_s$  equiprobable partitions and used the ranges' mean average values as states to train our model. We should note that within the significantly high frequency of the zero *racer* values, are also considered pages initially obtained within the top-k list and then fell (and remained) out. We remove any bias from  $RF_{Racer=0}$ , excluding any values not corresponding to stable rank and obtaining  $RF_{Racer=0} \approx 0.1$  which in turn suggested 10 equiprobable states.

**Predictions with Racer** Based on the set of states mentioned above and formed to represent Web page trends, we are able to train MMs and predict the trend of a Web page in the future according to past trends. By assuming  $m+1$  temporally successive crawls, resulting in respective snapshots, a sequence of  $m$  states (representative of *racer* values) are constructed for each Web page. These are used to construct an  $m$ -order MM. Note that the memory  $m$  is an inherent feature of the model. After computing transition probabilities for every path, using the generated states, the future states can be predicted by using the chain rule [1]. Thus, for an  $m$ -order Markov Model, the path probability of a state sequence is  $P(s_1 \rightarrow \dots \rightarrow s_m) = P(s_1) \cdot \prod_{i=2}^m P(s_i | s_{i-m}, \dots, s_{i-1})$ , where each  $s_i$  ( $i \in \{1, 2, \dots, n\}$ ) for any time interval may vary over all the possible states (ranges of *racer* values). Then, predicting the future trend of a page is performed by computing the most likely next state given the so far state path.

In specific, assuming  $m$  time intervals, the next most probable state  $X$  is computed as:  $X = \arg \max_X P(s_1 \rightarrow \dots \rightarrow s_{m-1} \rightarrow X)$ .

Using that, we predict future states for each page. As each state is the mean of a *Racer* range, we compute back the future nrank. Therefore, we are able to predict future top-k ranking by sorting the *racer* of Web pages in ascending order.

### 3.2 Regression Models

Assume a set of  $N$  Web pages and observations of *nrank* values at  $m$  time steps. Let  $x_i = (x_{i1}, \dots, x_{im})$  be the *nrank* values for Webpage  $i$  at the time points  $t = (t_1, \dots, t_m)$ , where the  $(N \times m)$  design matrix  $X$  stores all the observed *nrank* values so that each row corresponds to a Webpage and each column to a time point. Given these values we wish to predict the *nrank* value  $x_{i*}$  for each Webpage at some time  $t_*$  which typically corresponds to a future time point ( $t_* > t_i, i = 1, \dots, m$ ). Next, we discuss a simple prediction method based on linear regression where the input variable corresponds to time and the output to the *nrank* value.

For a certain Webpage  $i$  we assume a linear regression model having the form  $x_{ik} = a_i t_k + b_i + \epsilon_k, k = 1, \dots, m$  ( $\epsilon_k$  denotes a zero-mean Gaussian noise). Note that the parameters  $(a_i, b_i)$  are Webpage-specific and their values are calculated using least squares. In other words, the above formulation defines a separate linear regression model for each Web page thus they treat independently. This can be restrictive since possible existing similarities and dependencies between different Web pages are not taken into account.

### 3.3 Clustering Using EM

We assume that the *nrank* values of each Web page fall into one of  $J$  different clusters. Clustering can be viewed as training a mixture probability model. To generate the *nrank* values  $x_i$  for Web page  $i$ , we first select the cluster type  $j$  with probability  $\pi_j$  (where  $\pi_j \geq 0$  and  $\sum_{j=1}^J \pi_j = 1$ ) and then produce the values  $x_i$  according to a linear regression model  $x_{ik} = a_j t_k + b_j + \epsilon_k, k = 1, \dots, m$ , where  $\epsilon_k$  is independent Gaussian noise with zero mean and variance  $\sigma_j^2$ . This implies that given the cluster type  $j$  the *nrank* values are drawn from the product of Gaussians  $p(\mathbf{x}_i | j) = \prod_{k=1}^m N(x_{ik} | a_j t_k + b_j, \sigma_j^2)$ .

The cluster type that generated the *nrank* values of a certain Web page is an unobserved variable and thus after marginalization we obtain a mixture unconditional density  $p(x_i) = \sum_{j=1}^J \pi_j p(x_i | j)$  for the observation vector  $x_i$ . To train the mixture model and estimate the parameters  $\theta = (\pi_j, \sigma_j^2, a_j, b_j)_{j=1, \dots, J}$ , we can maximize the log likelihood of the data  $L(\theta) = \log \prod_{i=1}^N p(x_i)$  by using the EM algorithm [2]. Given an initial state for the parameters, EM optimizes over  $\theta$  by iterating between  $E$  and  $M$  steps:

The  $E$  step computes the posterior probabilities  $R_j^i = \frac{\pi_j p(x_i | j)}{\sum_{\rho=1}^J \pi_\rho p(x_i | \rho)}$ , for  $j = 1, \dots, J$  and  $i = 1, \dots, N$ , ( $N$  is the total number of web pages).

The  $M$  step updates the parameters according to:  $\pi_j = \frac{1}{N} \sum_{i=1}^N R_j^i$ ,  
 $\sigma_j^2 = \frac{\sum_{i=1}^N R_j^i \sum_{k=1}^m (x_{ik} - a_j t_k - b_j)^2}{\pi_j}$  and  $\begin{bmatrix} a_j \\ b_j \end{bmatrix} = \frac{1}{N_j} \begin{bmatrix} t^T t & t^T \mathbf{1} \\ t^T \mathbf{1} & m \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^N R_j^i x_i^T t \\ \sum_{i=1}^N R_j^i x_i^T \mathbf{1} \end{bmatrix}$ ,  
 $j = 1, \dots, J$ ,  $\mathbf{t}$  is the vector of all time points and  $\mathbf{1}$  is the  $m$ -dimensional vector of ones.

Once we have obtained suitable values for the parameters, we can use the mixture model for prediction. Particularly, to predict the *nrank* value  $x_{i*}$  of Web

page  $i$  at  $t_*$  given the observed values  $x_i = (x_{i1}, \dots, x_{im})$  at previous times, we express the posterior distribution  $p(x_{i*} | x_i)$  using the Bayes rule  $p(x_{i*} | x_i) = \sum_{j=1}^J R_j^i N(x_{i*} | a_j t_* + b_j, s_j^2)$ , where  $R_j^i$  is computed according to E-step. To obtain a specific predictive value for  $x_{i*}$ , we can use the mean value of the above posterior distribution  $x_{i*} = \sum_{j=1}^J R_j^i (a_j t_* + b_j)$  or the median estimate  $x_{i*} = a_j t_* + b_j$ , where  $j = \text{argmax}_\rho R_\rho^i$  that considers a hard assignment of the Web page into one of the  $J$  clusters.

## 4 Top-k List Similarity Measures

In order to evaluate the quality of predictions, we need to measure the similarity of the predicted to the actual top-k ranking. For this purpose, we examine measures commonly used for comparing rankings, point out the shortcomings of existing and define a new similarity measure for top-k rankings, denoted as *RSim*.

### 4.1 Existing Similarity Measures

The first one, denoted as *OSim*( $A, B$ ) [4] indicates the degree of overlap between the top-k elements of two sets  $A$  and  $B$  (each one of size  $k$ ):  $OSim(A, B) = \frac{|A \cap B|}{k}$ .

The second, *KSim*( $A, B$ ) [4], is based on Kendall's distance measure [3] and indicates the degree that the relative orderings of two top-k lists are in agreement:  $KSim(A, B) = \frac{|(u,v):A',B', \text{agree in order}|}{|A \cup B|(|A \cup B| - 1)}$ , where  $A'$  is an extension of  $A$  resulting from appending at its tail the elements  $x \in A \cup (B - A)$  and  $B'$  is defined analogously.

Another interesting measure introduced in Information Retrieval for evaluating the accumulated relevance of a top-k document list to a query is the (*Normalized*) *Discounted Cumulative Gain* (*N(DCG)*) [5]. This measure assumes a top-k list, where each document is featured with a relevance score accumulated by scanning the list from top to bottom. Although DCG could be used for the evaluation of our predictions, since it takes into account the relevance of a top-k list to another, it exhibits some basic features that prevented us from using it in our experiments. It penalizes errors by maintaining an increasing value of cumulative relevance. While this is based on the rank of each document, the size  $k$  of the list is not taken into account – thus the length of the list is irrelevant in DCG. Errors in top ranks of a top-k list should be considered more important than errors in low-ranked positions. This important feature lacks from both DCG and NDCG measures. Moreover, DCG value for each rank in the top-k list is computed taking into account the previous values in the list.

Next, we introduce *Spearman's Rank Correlation Coefficient*, which was used during the experimental evaluation, consists a non-parametric (distribution-free) rank statistic proposed by Spearman (1904) measuring the strength of associations between two variables and is often symbolized by  $\rho$ . It estimates how well the relationship between two variables can be described using a monotonic

function. If there are no repeated data values of these variables (like in ranking problem), a perfect Spearman correlation of +1 or -1 exists if each variable is a perfect monotone function of the other.

It is often confused with the Pearson correlation coefficient between ranked variables. However, the procedure used to calculate  $\rho$  is much simpler. If  $X$  and  $Y$  are two variables with corresponding ranks  $x_i$  and  $y_i$ ,  $d_i = x_i - y_i$ ,  $i = 1, \dots, n$ , between the ranks of each observation on the two variables, then it is given by:

$$\rho = 1 - \frac{6 \cdot \sum_{i=1}^n d_i^2}{n(n^2-1)}.$$

## 4.2 RSim Quality Measure

The observed similarity measures do not cover sufficiently the fine grained requirements arising, comparing top-k rankings in the Web search context. So we need a new similarity metric taking into consideration: a)The absolute difference between the predicted and actual position for each Webpage as large difference indicates a less accurate prediction and b)The actual ranking position of a Web page, because failing to predict a highly ranked Webpage is more important than a low-ranked. Based on these observations, we introduce a new measure, named *RSim*. Every inaccurate prediction made incurs a certain penalty depending on the two noted factors. If prediction is 100% accurate (same predicted and actual rank), the penalty is equal to zero. Let  $B_i$  be the predicted rank position for page  $i$  and  $A_i$  the actual. The Cumulative Penalty Score (CPS) is computed as  $CPS(A, B) = \sum_{i=1}^k |A_i - B_i| \cdot (k + 1 - A_i)$ .

The proposed penalty score *CPS* represents the overall error (difference) between the involved top-k lists A and B and is proportional to  $|A_i - B_i|$ . The term  $(k + 1 - A_i)$  increases when  $A_i$  becomes smaller so errors in highly ranked Web pages are penalized more. In the best case, rank predictions for all Web pages are completely accurate ( $CPS = 0$ ), since  $A_i = B_i$  for any value of  $i$ . In the worst case, the rank predictions for all Web pages not only are inaccurate, but also bear the greatest *CPS* penalty possible. In such a scenario, all the Web pages predicted to be in the top-k list, actually hold the position  $k+1$  (or worse).

Assuming that we want to compare two rankings of length  $k$ , then the maximum *CPS* for even and odd values of  $k$  is equal to  $\frac{2k^3+3k^2+k}{6}$ . The proof for  $CPS_{max}$  final form is omitted due to space limitations.

Based on the above we define a new similarity measure, *RSim*, to compare the similarity between top-k rank lists as follows:

$$RSim(A_i, B_i) = 1 - \frac{CPS(A_i, B_i)}{CPS_{max}(A_i, B_i)}. \quad (1)$$

In the best-case prediction scenario, *RSim* is equal to one, while in the worst-case *RSim* is equal to zero. So the closer the value of *RSim* is to one, the better and more accurate the rank predictions are.

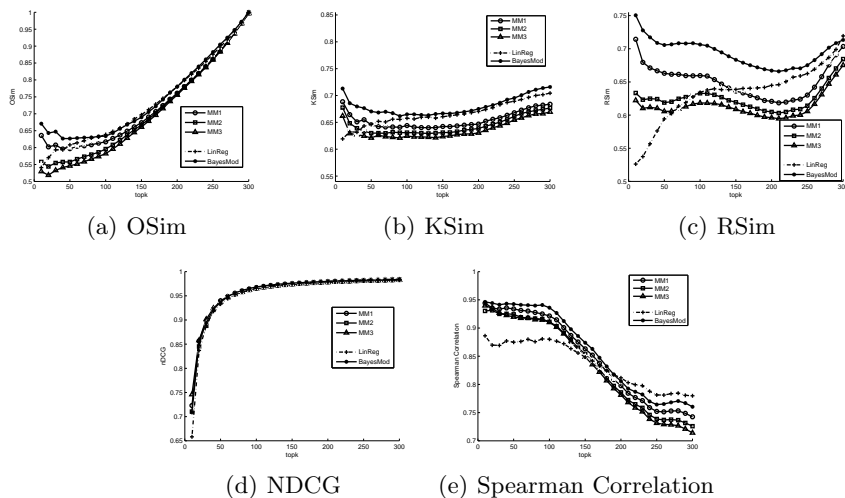


Fig. 1. Prediction accuracy vs Top-k list length - Yahoo dataset.

## 5 Experimental Evaluation

In order to evaluate the effectiveness of our methods we performed experiments on two different real world datasets. These consist collections of top-k ranked lists for 22 queries over a period of 11 days as resulted from the Yahoo!<sup>3</sup> and the Google search engines, produced in the same way. In our experiments, we evaluate the prediction quality in terms of similarities between the predicted and the actual top-k ranked lists using *OSim*, *KSim*, *NDCG*, *Spearman correlation* and the novel similarity measure *RSim*.

### 5.1 Datasets and Query Selection

For each dataset (Yahoo and Google) a wealth of snapshots were available, ensuring we have enough evolution to test our approach. A concise description of each dataset and query-based approach follow. The Yahoo and Google datasets consist of 11 consecutive daily top-1000 ranked lists computed using the Yahoo Search Web Services<sup>4</sup> and the Google Search engine respectively. These sets were picked from popular: a) queries appeared in Google Trends<sup>5</sup> and b) current queries (i.e. euro 2008 or Olympic games 2008).

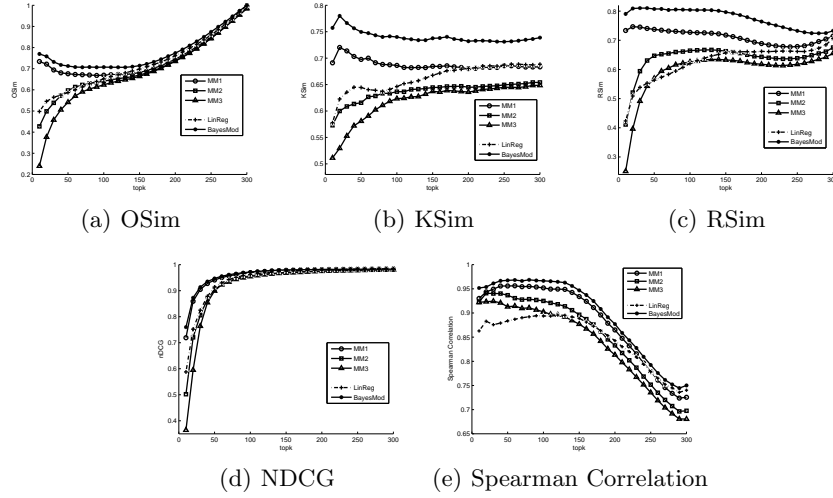
### 5.2 Experimental Methodology

We compared all predictions among the various approaches and we next describe the steps assumed for both datasets. At first, we computed PageRank scores for

<sup>3</sup> <http://search.yahoo.com>

<sup>4</sup> <http://developer.yahoo.com/search/>

<sup>5</sup> <http://www.google.com/trends>



**Fig. 2.** Prediction accuracy vs Top-k list length - Google dataset.

each snapshot of our datasets and obtained the top-k rankings using the scoring function mentioned. Having computed the scores, we calculated the *nrank* (racer values for MMs) for each pair of consecutive graph snapshots and stored them in a matrix  $nrank(racer) \times time$ . Then, assuming an  $m$ -path of consecutive snapshots, we predict the  $m + 1$  state. For each page  $p$ , we predict a ranking comparing it to actual by a 10-fold cross validation process (training 90% of dataset and testing on the remaining 10%).

In the case of the EM approach, we tested the quality of clustering results for clusters cardinality between 2 and 10 for each query and chose the one that maximized the overall quality of clustering. This was defined as a monotone combination of within-cluster  $w_c$  (sum of squared distances from each point to the center of cluster it belongs to) and between-cluster variation  $b_c$  (distance between cluster centers). As score function of clustering, we considered the ratio  $b_c/w_c$ .

### 5.3 Experimental Results

Regarding the Google and Yahoo! dataset results coming out of the experimental evaluation, one can see that the MMs prevail with very accurate results. Regression based techniques (LinReg) reach and outweigh MMs performance as the length of top-k list increases proving their robustness.

In both datasets experiments prove the superiority of EM approach (BayesMod) whose performance is very satisfying for all similarity measures. The MMs come next in the evaluation ranking, where as smaller the order is the better is the prediction accuracy, though one would think of the contrary.

Obviously (figures) the proposed framework offers incredibly high accuracy predictions and is very encouraging, as it ranges systematically between 70% and 100% providing a tool for effective predictions.

## 6 Conclusions

We have described predictor learning algorithms for Web page rank prediction based on a framework of learning techniques (MMs, LinReg, BayesMod) and experimental study showed that they can achieve overall very good prediction performance. Further work will focus in the following issues: a) Multi-feature prediction: we intend to deal with the internal mechanism that produces the ranking of pages (not only rank values) based on multiple features, b) Combination of such methods with dimensionality reduction techniques.

## References

1. Kemeny, J.G., Snell, J.L.: Finite Markov Chains. Princeton. (1963)
2. Dempster, A.P., Laird, N.M., Rubin D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Royal Statistical Society. **39** (1977) 1–38
3. Kendall, M.G., Gibbons, J.D.: Rank Correlation Methods. UK: Charles Griffin (1990)
4. Haveliwala, T.H.: Topic-Sensitive PageRank. Proc. WWW. (2002)
5. Jarvelin, K., Kekalainen, J.: Cumulated gain-based evaluation of IR techniques. TOIS. **20** (2002) 422–446
6. Chien, S., Dwork, C., Kumar, R., Simon, D.R., Sivakumar, D.: Link Evolution: Analysis and Algorithms. Internet Mathematics. **1** (2003) 277–304
7. Chen, Y.-Y., Gan, Q., Suel, T.: Local Methods for Estimating PageRank Values. Proc. of CIKM. (2004)
8. Langville, A.N., Meyer, C.D.: Updating PageRank with iterative aggregation. Proc. of the 13th international World Wide Web conference on Alternate track papers and posters. (2004) 392–393
9. Kan, M.-Y., Thi, H.O.: Fast webpage classification using URL features. Conference on Information and Knowledge Management, ACM. (2005) 325–326
10. Yang, H., King, I., Lu, M.R.: Predictive Ranking: A Novel Page Ranking Approach by Estimating the Web Structure. Proc. of the 14th International WWW Conference. (2005) 1825–1832
11. Broder, A.Z., Lempel, R., Maghoul, F., Pedersen, J.: Efficient PageRank approximation via graph aggregation. Inf. Retrieval. **9** (2006) 123–138
12. Vlachou, A., Berberich, K., Vazirgiannis, M.: Representing and quantifying rank-change for the Web graph. Algorithms and Models for the Web-Graph, 4th International Workshop, WAW: Springer. (2006) 157–165
13. Vazirgiannis, M., Drosos, D., Senellart, P., Vlachou, A.: Web Page Rank Prediction with Markov Models. WWW poster. (2008)
14. Sayyadi, H., Getoor, L.: Future Rank: Ranking Scientific Articles by Predicting their Future PageRank. SIAM Intern. Confer. on Data Mining. (2009) 533–544
15. Zacharouli, P., Titsias, M., Vazirgiannis, M.: Web page rank prediction with PCA and EM clustering. Proc. of the 6th Intern. Workshop on Algorithms and Models for the Web-Graph: Springer-Verlag Berlin. (2009) 104–115