Neural Process Lett
DOI 10.1007/s11063-013-9295-8

Learning to Rank with Extreme Learning Machine

Weiwei Zong · Guang-Bin Huang

Abstract Relevance ranking has been a popular and interesting topic over the years, which has a large variety of applications. A number of machine learning techniques were successfully applied as the learning algorithms for relevance ranking, including neural network, regularized least square, support vector machine and so on. From machine learning point of view, extreme learning machine actually provides a unified framework where the aforementioned algorithms can be considered as special cases. In this paper, pointwise ELM and pairwise ELM are proposed to learn relevance ranking problems for the first time. In particular, ELM type of linear random node is newly proposed together with kernel version of ELM to be linear as well. The famous publicly available dataset collection LETOR is tested to compare ELM-based ranking algorithms with state-of-art linear ranking algorithms.

Keywords Extreme learning machine · Learning to rank · Linear random node · Linear kernal

1 Introduction

In recent years, relevance ranking has been a popular topic in fields such as document retrieval, collaborative filtering, key term extraction, sentiment analysis, etc. Due to the discriminative capability, machine learning techniques are widely adopted in relevance ranking, namely, "learning-to-rank". As in most of the machine learning algorithms, the framework of a learning-to-rank system usually consists of a learning system and a ranking system. In the stage of learning, a model is built based on the training data, which is capable of ranking the testing data with unknown relevance degree in the following ranking stage.

W. Zong (✉) · G.-B. Huang
School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore
e-mail: zong0003@e.ntu.edu.sg

G.-B. Huang
e-mail: egbhuang@ntu.edu.sg

Published online: 23 March 2013

In literature, learning-to-rank algorithms can be categorized into three types: pointwise, pairwise and listwise. Ranking is perceived as either a regression problem or classification problem in pointwise approaches, where the relationship between the feature (contain information of both query and the document) and target label (relevance judgment) is modeled. However, pairwise ranking is claimed to be more consistent with the nature of ranking, where the pairwise comparison between the documents instead of the pointwise approximation of document-label relationship is considered.

Representative pointwise approaches include subset ranking using regression [1], where relevance judgment was treated as the real value so that the hypothesis was approximated by a regression algorithm; binary/multiclass classification model was used to predict the ranking function [2,3] by treating the relevance judgment as the class label and adopting proper loss function, such as hinge loss for support vector machine (SVM) [4] and least square loss for least square SVM (LS-SVM) [5].

In lieu of sample-label relationship approximation, the ranking function is searched in pairwise approaches to reveal the pairwise preference between two samples. A number of machine learning techniques were applied in modeling. For instance, neural network with gradient descent was used to minimize the cross-entropy loss in RankNet [6]; the same modeling technique to optimize fidelity loss in FRank [7]; boosting algorithms were also validated with exponential loss [8]; in RankSVM [9], the ranking function was searched to minimize the hinge loss function; and finally regularized least squares were used to optimize the least square loss function in [10]. However, extreme learning machine [11–17], as a new powerful machine learning technique, is quite new to the field of relevance ranking.

ELM was originally proposed as a least square based learning algorithm for single hidden-layer neural network (SLFN), where the hidden nodes can be any type of nonlinear piecewise function and whose parameters are randomly generated in prior to the training samples [11–14,17]. Compared with traditional neural networks, the tedious process of iterative parameter tuning is eliminated and problems like slow convergence and local minimum are solved.

The consistency between ELM and SVM, least square support vector machine (LS-SVM) and proximal SVM was studied and analyzed from optimization point of view [15–17]. SVMs are actually a sub-optimal solution to ELM due to an extra bias term in the representation, thus narrowing the searching space where possible optimal solution may reside [16].

In work of [16], it was found that ELM can provide a unified solution for a generalized SLFN, which include but not limit to neural network, support vector network and regularized network. That is to say, the feature mapping function can be any type of nonlinear piecewise function as in conventional ELM random nodes; or an unknown function to form a mercer's kernel as in SVMs and other kernel based algorithms.

To the best of our knowledge, there is no work of applying ELM in relevance ranking so far. Therefore, in this paper, ELM is studied as a learning-to-rank algorithm from the perspective of both pointwise and pairwise.

2 Pointwise ELM

In pointwise approaches, ranking problem is usually formulated as a traditional regression, classification or ordinal regression model. Features of the training samples are extracted from both the query and document. And the relevance judgment of each sample can be interpreted as the output value in regression models or class label in classification models.

Suppose there are n queries, associated with query q_i being m_i number of documents. The training samples are represented as $(\mathbf{x}_j^i, y_j^i),\ i = 1, \ldots, n;\ j = 1, \ldots, m(i)$, where $\mathbf{x}_j^i$

is the feature extracted from the ith query and jth document with y_j^i indicating the relevance degree between them. To model the relationship between query-document pair $\mathbf{x}_j^i$ and their relevance degree, ELM is naturally capable of such task.

The mathematical model of the SLFNs can be represented as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}, \tag{1}$$

where $\mathbf{H}$ is the hidden layer output matrix, $\boldsymbol{\beta}$ is the output weight, and $\mathbf{T} = [y_1^1, \ldots, y_{m(1)}^1, \ldots, y_{m(n)}^n]^T$ is the target vector

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1^1) \\ \vdots \\ \mathbf{h}(\mathbf{x}_{m(1)}^1) \\ \vdots \\ \mathbf{h}(\mathbf{x}_{m(n)}^n) \end{bmatrix}. \tag{2}$$

To minimize the least square loss function of the training error and regulation term $\|\boldsymbol{\beta}\|$

$$\text{Minimize: } \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|^2 \quad \text{and} \quad \|\boldsymbol{\beta}\|, \tag{3}$$

the solution of ELM is as follows

$$\begin{aligned} \text{when } N < L : \boldsymbol{\beta} &= \hat{\mathbf{H}}^{\dagger}\mathbf{T} = \mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{T}, \\ \text{when } L < N : \boldsymbol{\beta} &= \hat{\mathbf{H}}^{\dagger}\mathbf{T} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^T\mathbf{H}\right)^{-1} \mathbf{H}^T\mathbf{T}, \end{aligned} \tag{4}$$

where L is number of hidden nodes in the SLFN, N is the number of total training samples and C is the regularization term specified by users. The two formulas in (4) can be flexibly chosen given different circumstances. When there are relatively fewer training samples, the upper one is computational efficient since it only needs to calculate the matrix inversion of the training sample size. On the other hand, the lower one is recommended when given quite large dataset size. Therefore, the lower formula is adopted in implementation of this paper.

Algorithm 1 Pointwise Ranking ELM Algorithm

Given: Training sample pairs $(\mathbf{x}_j^i, y_j^i)$, number of hidden nodes L, regularization term C, and testing sample $\mathbf{x}$,

Linear Random Node:

1: Generate L hidden nodes with the parameters $(\mathbf{a}, b)$ of each node randomly assigned;
2: Calculate the hidden layer output matrix $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1^1), \ldots]^T$ as shown in (2);
3: Calculate the output weight $\boldsymbol{\beta}$ as shown in (4) and the output function of testing sample as $f(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \cdot \boldsymbol{\beta}$.

Linear Kernel:

4: Calculate the output function of testing sample as shown in (8) and (9).

3 Linear Random Node and Linear Kernel

According to [16], ELM actually provides a unified solution for the generalized SLFN, which extends from traditional neural network to regularized network and support vector network. Therefore, the hidden layer feature mapping $\mathbf{h}(\mathbf{x}_j^i)$ for training sample $\mathbf{x}_j^i$ can be of large variety than any other machine learning technique.

3.1 Random Linear Node

Similar to other SLFN based algorithms, the hidden layer feature mapping has the form of a (row) vector $\mathbf{h}(\mathbf{x}_j^i) = [h_1(\mathbf{x}_j^i), \ldots, h_L(\mathbf{x}_j^i)]$, where training data is mapped from original input space into the hidden layer space with dimensionality L - number of hidden nodes.

The hidden layer function $h_k(\mathbf{x}_j^i) = G(\mathbf{a}_k, b_k, \mathbf{x}_j^i),\ k = 1, \ldots, L$ is termed ELM random node in theory of ELM. The randomness lies in that parameters $(\mathbf{a}_k, b_k)$ of (kth) hidden node are randomly assigned according to any continuous probability distribution without knowledge of the training data. This unique feature of ELM that differs from other machine learning techniques renders a simple and fast algorithm, which has attracted tremendous attention recently.

With randomly generated parameters, the random node function $G(\mathbf{a}_k, b_k, \mathbf{x}_j^i)$ has very flexible form. The universal approximation capability is proved [12,16] with such random nodes using almost any type of piecewise continuous functions. Two popular nonlinear functions are listed

– Sigmoid function

$$G(\mathbf{a}, b, \mathbf{x}) = \frac{1}{1 + \exp(-(\mathbf{a} \cdot \mathbf{x} + b))}, \tag{5}$$

– Gaussian function

$$G(\mathbf{a}, b, \mathbf{x}) = \exp(-b\|\mathbf{x} - \mathbf{a}\|^2). \tag{6}$$

From heuristic point of view, nonlinear hidden node function is generally very helpful for linearly non-separable data or data with small scale. When it comes to ranking problems, where the number of training data usually is up to tens of thousands, nonlinear hidden node function is computationally costy, yet may not guarantee better performance than linear function. Therefore, linear random node is proposed in the paper, which is new in ELM theory

- Linear function

$$G(\mathbf{a}, b, \mathbf{x}) = \mathbf{a} \cdot \mathbf{x} + b, \tag{7}$$

where $\mathbf{a}$ and b are randomly generated as in the traditional ELM random nodes.

Remark 1 As proved in [12,16], ELM with random nodes is actually universal approximator provided the activation function is any type of piecewise continuous function, such as Sigmoid function, RBF function or piecewise function. Obviously the proposed linear random node complies with such function constraint to render a universal approximator. In other words, ELM with linear random nodes is capable of approximating any continuous functions.

3.2 Linear Kernel

On the other hand, the feature mapping $\mathbf{h}(\mathbf{x}_j^i)$ may not be explicitly given. And as in support vector network, solution of ELM is kernerlized in (8) with kernel defined as

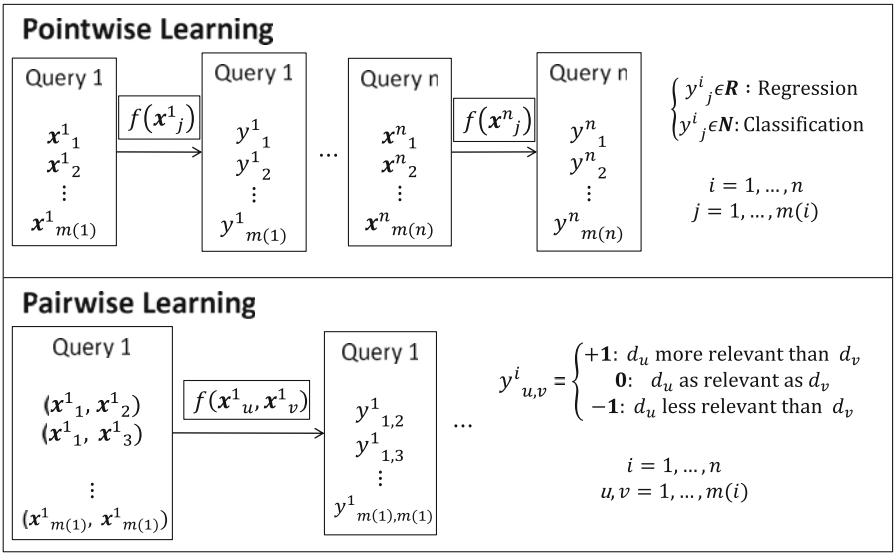


Fig. 1 Illustration of input space and output space for pointwise learning and pairwise learning

$$\mathbf{\Omega}_{ELM} = \mathbf{H}\mathbf{H}^T : \Omega_{ELM\,i,j} = h(\mathbf{x}_i) \cdot h(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j), \tag{8}$$

$$\begin{aligned} \mathbf{f}(\mathbf{x}) &= \mathbf{h}(\mathbf{x})\mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{T} \\ &= \begin{bmatrix} K(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ K(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^T \left(\frac{\mathbf{I}}{C} + \mathbf{\Omega}_{ELM}\right)^{-1} \mathbf{T}. \end{aligned} \tag{9}$$

A widely used kernel is Gaussian kernel $K(\mathbf{u}, \mathbf{v}) = \exp(-\gamma \|\mathbf{u} - \mathbf{v}\|^2)$, where γ is the kernel parameter. However, as far as the computational cost as well as performance efficiency are concerned, the linear kernel $K(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v}$ is adopted in this paper.

4 Pairwise ELM

In pointwise approaches, it is the relationship between the query-document pair $\mathbf{x}^i_j$ (ith query and jth document) and its corresponding relevance degree y^i_j that is approximated by the hypothesis $f(\mathbf{x}^i_j)$. While the principle in pairwise approaches is that, if any document d_u is more relevant to query q_i comparing with document d_v, then relevance degree of y^i_u should be larger than y^i_v. In other words, with respect to each query, it is the preference of one document over another in terms of relevance degree that is of our interest. Afterwards, a regular binary classifier is used to approximate such preference.

4.1 Proposed Pairwise ELM

The aim of pairwise ELM is obviously to search for a hypothesis $f()$ such that $f(\mathbf{x}_u^i) > f(\mathbf{x}_v^i)$ if $y_u^i > y_v^i$ (Fig. 1). Within the framework and traditional least square loss function provided by ELM, the problem of minimizing the training error $\xi_{u,v}^i$ and the norm of output weight can be mathematically written as

$$\begin{aligned} &\text{Minimize: } \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\sum_{i=1}^{n}\sum \|\xi_{u,v}^i\|^2 \\ &\text{Subject to: } (\mathbf{h}(\mathbf{x}_u^i) - \mathbf{h}(\mathbf{x}_v^i))\boldsymbol{\beta} = (y_u^i - y_v^i) - \xi_{u,v}^i,\ i = 1, \ldots, n. \end{aligned} \tag{10}$$

As observed from (10), pairwise comparison can only be conducted among documents associated with the same query. With training data given as $(\mathbf{x}_j^i, y_j^i)$, a prior step is required to partition the original training data depending on the query. Inspired by the work of [10], graph theory helps to store such information in an $N \times N$ symmetric matrix $\mathbf{W}$, where $N = \sum_{i=1}^{n} m(i)$ is the overall number of documents. $\mathbf{W}$ is one type of adjacency graph, where $\mathbf{W}_{i,j} = 1$ if document d_i and d_j are connected to the same query and $\mathbf{W}_{i,j} = 0$ otherwise. All the diagonal entries are set to 1 in $\mathbf{W}$. The corresponding Laplacian matrix $\mathbf{L}$ [18] is then defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} = diag\{\sum_{j=1}^{N} \mathbf{W}_{1,j}, \ldots, \sum_{j=1}^{N} \mathbf{W}_{N,j}\}$. A simple example is demonstrated as follows.

Suppose we are given four samples $\{d_1, d_2, d_3, d_4\}$ and d_1 and d_2 are connected with query 1 while d_3 and d_4 are connected with query 2. We have

$$\mathbf{W} = \begin{pmatrix} 1&1&0&0 \\ 1&1&0&0 \\ 0&0&1&1 \\ 0&0&1&1 \end{pmatrix}; \mathbf{L} = \begin{pmatrix} 2&0&0&0 \\ 0&2&0&0 \\ 0&0&2&0 \\ 0&0&0&2 \end{pmatrix} - \begin{pmatrix} 1&1&0&0 \\ 1&1&0&0 \\ 0&0&1&1 \\ 0&0&1&1 \end{pmatrix} = \begin{pmatrix} 1&-1&0&0 \\ -1&1&0&0 \\ 0&0&1&-1 \\ 0&0&-1&1 \end{pmatrix}. \tag{11}$$

By comparing $\mathbf{L} \cdot [d_1\ d_2\ d_3\ d_4]^T$ and $\mathbf{L} \cdot [y_1\ y_2\ y_3\ y_4]^T$, an equation array (12) is obtained with the first two reflecting query 1 and the last two reflecting query 2. It is not too difficult to imagine how it works for pairwise comparison when there are more than two queries and four documents

$$\begin{cases} d_1 - d_2 & \rightarrow y_1 - y_2 \\ -d_1 + d_2 & \rightarrow -y_1 + y_2 \\ d_3 - d_4 & \rightarrow y_3 - y_4 \\ -d_3 + d_4 & \rightarrow -y_3 + y_4 \end{cases}. \tag{12}$$

Therefore, with the incorporation of graph W, (10) can be rewritten in a way that is more similar to traditional regression/classification form. Hence, only the pairwise relationship between samples associated with the same query maintains

$$\text{Min: } \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\mathbf{W}\sum_{i,j=1}^{N} \|(y_i - y_j) - (f(\mathbf{x}_i) - f(\mathbf{x}_j))\|^2. \tag{13}$$

According to Theorem 1 in [10], to minimize $\mathbf{W}\sum_{i,j=1}^{N} \|(y_i - y_j) - (f(\mathbf{x}_i) - f(\mathbf{x}_j))\|^2$ is equivalent to minimizing $(y_i - f(\mathbf{x}_i))^T\mathbf{L}(y_j - f(\mathbf{x}_j))$. The unconstrained optimization

problem becomes

$$\text{Min: } \frac{1}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + C\frac{1}{2}(\mathbf{T} - \mathbf{H}(\mathbf{x})\boldsymbol{\beta})^T\mathbf{L}(\mathbf{T} - \mathbf{H}(\mathbf{x})\boldsymbol{\beta}). \tag{14}$$

The solution of (14) can be obtained by setting the derivative to zero

$$\boldsymbol{\beta} = (\frac{\mathbf{I}}{C} + \mathbf{H}^T\mathbf{L}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{L}\mathbf{T}. \tag{15}$$

Similar to [16], the kernel version for ranking problems is not difficult to be derived as

$$f(\mathbf{x})_{kernel} = \begin{bmatrix} K(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ K(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^T \left(\frac{\mathbf{I}}{C} + \mathbf{L}\boldsymbol{\Omega}_{ELM}\right)^{-1}\mathbf{L}\mathbf{T}. \tag{16}$$

By combining with graph theory, pairwise ranking is reduced to a regular regression/classification problem. Moreover, as consistent with ELM theory, both random node version (14) and kernel version (16) are available. As mentioned earlier, only linear random node and linear kernel are considered in the experiments.

Algorithm 2 Pairwise Ranking ELM Algorithm

Given: Training sample pairs $(\mathbf{x}_j^i, y_j^i)$, number of hidden nodes L, regularization term C, and testing sample $\mathbf{x}$,

Linear Random Node:

1: Generate L hidden nodes with the parameters $(\mathbf{a}, b)$ of each node randomly assigned;
2: Calculate the hidden layer output matrix $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1^1), \ldots]^T$ as shown in (2);
3: Construct matrix $\mathbf{W}$ to store the information of query-document relationship and compute the corresponding Laplacian matrix $\mathbf{L}$;
4: Calculate the output weight $\boldsymbol{\beta}$ as shown in (15) and the output function of testing sample as $f(\mathbf{x}) = \mathbf{h}(\mathbf{x})\cdot\boldsymbol{\beta}$.

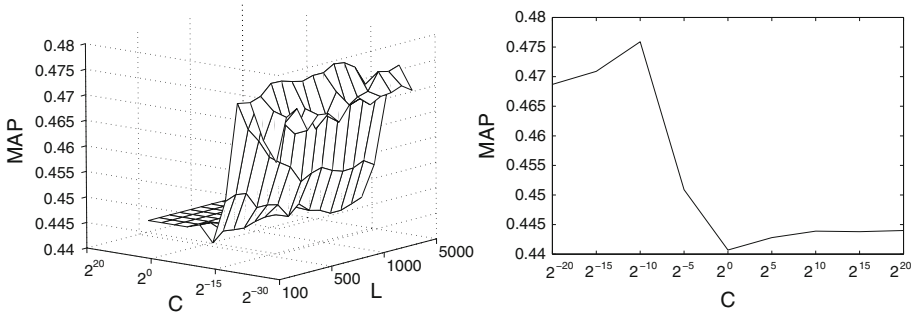
Linear Kernel:

5: Construct matrix $\mathbf{W}$ to store the information of query-document relationship and compute the corresponding Laplacian matrix $\mathbf{L}$;
6: Calculate the output function of testing sample as shown in (8) and (16).

4.2 Discussions with Similar Approaches

A similar work on pairwise ranking can be found in [10], where a regularization framework is built based on the least square loss function together with the graph theory, resulting in similar solution to (14, 16). However, regularized least square actually can be viewed as one special implementation in the ELM framework. The feature mapping function $G(\mathbf{a}, b, \mathbf{x})$ can be known to users, termed as ELM random node, which is the unique feature of traditional ELM. The feature mapping function may not be known, which results in a kernel version otherwise. In fact, solution provided in [10] can be considered as a special case of pairwise RankELM (RELM). When the parameters (**a**,b) in ELM linear random node $G(\mathbf{a}, b, \mathbf{x}) = \mathbf{a}\mathbf{x} + b$ are set as $\mathbf{a} = \mathbf{1}$ and $b = 0$ instead of being randomly generated, the resultant solution is the same as in [10].

A competitive regularized network based pointwise approach, namely ridge regression, which in essence is a regularized regressor with least square loss function. Hence the model



(a) Pointwise ELM with Linear Random Node **(b)** Pointwise ELM with Linear Kernel

Fig. 2 Relationship of the MAP with choices of the parameters tested on Folder 1 from OHSUMED

solution is quite similar to ELM with non-kernel node (4). However, $\mathbf{h}()$ in ELM can be a wide type of ELM random nodes as well as the kernel. Again, this algorithm can be considered as one special case of the ELM framework. Therefore, the performance of pointwise ELM with both random nodes and kernel are compared with this algorithm in the experiment.

By mainly concerning the operation of matrix multiplication and inversion and treating number of hidden nodes as a constant (since it is fixed as 1,000 in the experiment), the complexity of pointwise ELM with linear random node/linear kernel and pairwise ELM with linear random node/linear kernel are respectively $\mathcal{O}(N)$, $\mathcal{O}(N^3)$, $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ where N is the total number of training samples. Note that pointwise ELM with linear random node has shown great advantage over other state-of-art algorithms, say $\mathcal{O}(N \log N)$ for RankSVM given dataset of such large scale in document ranking.

5 Performance Evaluation

The proposed pointwise RankELM and pairwise RankELM with both linear random nodes and liner kernel are evaluated on the publicly available dataset collection LETOR [19], for the task of query-document pair ranking. The performances are further compared with state-of-art linear pointwise approaches (linear regression, regularized regression with L2 loss function) as well as linear pairwise approaches (RankSVM with linear kernel), whose results are made public as well based on uniformly pre-processed datasets. The evaluation metrics are selected as mean average precision (MAP) [20] and normalized discounted cumulative gain (NDCG) [21], which are commonly used in the field of information retrieval.

5.1 Data Specification

Three datasets from LETOR are tested in the experiment, namely OHSUMED, TD2003 and TD2004 dataset. There are 1,6140 query-document pairs from OHSUMED with 106 queries, 4,9058 from TD2003 with 50 queries and 7,4146 from TD2004 with 72 queries. For OHSUMED, the three relevance label are two (relevant), one (possibly relevant) and 0 (irrelevant). The relevance label for TD2003 and TD2004 is either one (relevant) or 0 (irrelevant). The features are normalized version and 5-fold cross validation are used. The

data is partitioned into five parts, where three are used for training, 1 for parameter selection (according to MAP result) and one for performance validation. The final result in comparison is averaged over the 5 folders.

5.2 Parameter Settings

Since the feature mapping in ELM can be known as well as unknown, the linear random node and liner kernel are both validated. Two parameters need to be tuned in linear random node: number of hidden nodes L and regularization term C. According to the literature of ELM, the performance is normally insensitive to choice of L. Figure 2 displays the result given different choices of parameters for pointwise ELM. It is observable that the performance of pointwise ELM is not sensitive with choice of L, as consistent with the ELM theory. And similar observation can be found for pairwise ELM with linear random node. Therefore, L is fixed as 1,000 in the linear random node through the whole experiment. A wide range of trials are made with $C \in \{2^{-30}, 2^{-28}, \ldots, 2^{18}, 2^{20}\}$. In linear kernel implementation, only the regularization term C are required specification, of which a grid search as in linear random node is conducted.

5.3 Experiment Results

Each folder in a dataset contains the partitions for training, validation and testing. Table 1 displays the MAP value of learning models validated on testing partition, folder by folder. Note that RELM_N in the table denotes Ranking ELM with linear random node and RELM_K represents Ranking ELM with linear kernel. And an average MAP value over all the 5 folders is provided as well. It is obvious that learning algorithms with regularization term has significantly improved the performance compared with linear regression without regularization term. On most of the cases, ranking algorithms based on ELM where the feature mapping can be linear random node or linear kernel outperforms regularized linear regression.

In the framework of pairwise learning, RankSVM is selected as a reference to compare with. Traditional SVM with linear kernel is adopted as a binary classifier, with computational complexity as a quadratic function of the number of training samples which is normally quite large. It can be found that the performance of pairwise RELM outperforms RSVM on most of the cases. In addition, the option of linear random node actually helps to significantly reduce the computational complexity. Due to large dataset size, RELM with only random node version is consucted on dataset TD2004.

Since the metric NDCG is suitable for multiple label problems and precision@position is suitable for binary problems, the NDCG result of OHSUMED and precision@position results of TD2003 and TD2004 are provided in Tables 2 and 3.

6 Conclusions

In this paper, ELM is extended to the field of relevance ranking for the first time. According to literature of learn-to-rank algorithms, ranking problems can be formulated in pointwise or pairwise point of view, where in this paper pointwise RankELM and pairwise RankELM are proposed respectively. Ranking algorithms based on ELM actually has a large variety of choices when it comes to the feature mapping functions, namely any type of piecewise continuous function or in the form of a kernel. In the experiment, as far as computation cost is

Table 1 MAP value obtained from various pointwise learning methods and pairwise learning methods of every folder from dataset OHSUMED, TD2003 and TD2004

	Pointwise methods				Pairwise methods		
	Regression	Regularized	RELM_N	RELM_K	RankSVM	RELM_N	RELM_K
OHSUMED							
F1	0.2979	0.3374	0.3421	**0.3459**	0.3038	**0.3421**	**0.3450**
F2	0.4302	0.4351	**0.4510**	0.4466	0.4468	0.4483	**0.4514**
F3	0.4398	0.4703	**0.4734**	0.4751	0.4648	**0.4707**	**0.4696**
F4	0.4978	0.5061	**0.5137**	**0.5127**	0.4990	**0.5118**	**0.5115**
F5	0.4442	**0.4703**	0.4574	**0.4724**	0.4528	0.4639	0.4642
Avg	0.4220	0.4439	0.4475	**0.4505**	0.4334	0.4474	**0.4483**
TD2003							
F1	0.1262	**0.1745**	0.1677	0.1511	**0.1637**	0.1555	**0.1614**
F2	**0.3009**	0.2031	**0.3010**	0.2982	0.2576	0.2965	**0.3138**
F3	0.2665	0.3703	0.2673	**0.4101**	**0.4079**	0.3242	**0.4093**
F4	0.2658	**0.2734**	0.2466	0.2332	0.2356	**0.2707**	0.2411
F5	0.2450	0.1955	0.2437	**0.2536**	0.2490	0.2395	**0.2610**
Avg	0.2409	0.2434	0.2453	**0.2692**	0.2628	0.2573	**0.2773**
TD2004							
F1	0.1725	0.1908	**0.2139**	–	0.2107	**0.2306**	–
F2	**0.2117**	0.1817	**0.2118**	–	0.2091	**0.2186**	–
F3	**0.2196**	0.2035	0.2146	–	**0.2063**	0.1948	–
F4	0.1910	0.1886	**0.2077**	–	0.2185	**0.2263**	–
F5	0.2442	0.2311	**0.2562**	–	**0.2740**	0.2612	–
Avg	0.2078	0.1992	**0.2208**	–	0.2237	**0.2263**	–

Bold values indicate the best performing algorithms

Table 2 NDCG value obtained from various pointwise learning methods and pairwise learning methods of every folder from dataset OHSUMED

OHSUMED	Pointwise methods				Pairwise methods		
	Regression	Regularized	RELM_N	RELM_K	RankSVM	RELM_N	RELM_K
NDCG@1	0.4456	0.5361	**0.5550**	0.5232	0.4958	0.5211	**0.5642**
NDCG@2	0.4532	0.4784	0.4932	**0.5024**	0.4331	0.4977	**0.5019**
NDCG@3	0.4426	0.4740	0.4874	**0.4975**	0.4207	0.4895	**0.4923**
NDCG@4	0.4368	0.4601	0.4739	**0.4896**	0.4240	0.4742	**0.4866**
NDCG@5	0.4278	0.4568	0.4696	**0.4782**	0.4164	0.4691	**0.4755**
NDCG@6	0.4216	0.4536	0.4623	**0.4685**	0.4159	**0.4589**	**0.4611**
NDCG@7	0.4217	0.4449	0.4556	**0.4645**	0.4133	**0.4557**	**0.4567**
NDCG@8	0.4187	0.4444	0.4565	**0.4616**	0.4072	0.4496	**0.4573**
NDCG@9	0.4136	0.4427	0.4529	**0.4627**	0.4124	0.4466	**0.4555**
NDCG@10	0.4110	0.4436	0.4501	**0.4607**	0.4140	0.4422	**0.4574**

Bold values indicate the best performing algorithms

Table 3 Precision value obtained from various pointwise learning methods and pairwise learning methods of every folder from dataset TD2003 and TD2004

	Pointwise methods				Pairwise methods		
	Regression	Regularized	RELM_N	RELM_K	RankSVM	RELM_N	RELM_K
TD2003							
P@1	0.3200	0.3400	**0.3600**	**0.3600**	0.3200	0.3400	**0.3600**
P@2	0.3000	0.3400	0.3400	**0.3500**	0.3100	0.3200	**0.3400**
P@3	0.2600	**0.3000**	0.2867	0.2867	**0.2933**	0.2533	0.2800
P@4	0.2450	0.2550	**0.2750**	0.2600	**0.2850**	0.2400	0.2700
P@5	0.2160	0.2360	**0.2440**	0.2400	**0.2760**	0.2280	0.2600
P@6	0.2133	0.2067	**0.2267**	**0.2267**	**0.2500**	0.2233	0.2333
P@7	0.2057	**0.2086**	0.2029	0.2057	**0.2229**	0.2086	**0.2200**
P@8	0.1950	0.1975	**0.2050**	0.1950	**0.2075**	0.1975	0.2025
P@9	0.1844	0.1822	**0.1911**	**0.1889**	**0.2000**	0.1845	0.1911
P@10	0.1780	0.1760	**0.1820**	0.1780	**0.1880**	0.1760	0.1820
TD2004							
P@1	0.3600	0.2933	**0.4000**	–	0.4133	**0.4533**	–
P@2	0.3400	0.3200	**0.3800**	–	0.3467	**0.4000**	–
P@3	0.3333	0.2978	**0.3511**	–	0.3467	**0.3600**	–
P@4	0.3200	0.2833	**0.3333**	–	0.3333	**0.3433**	–
P@5	0.3120	0.2720	**0.3200**	–	0.3013	**0.3253**	–
P@6	0.2867	0.2644	**0.2933**	–	0.2867	**0.2933**	–
P@7	0.2705	0.2629	**0.2781**	–	0.2800	**0.2857**	–
P@8	0.2683	0.2500	**0.2733**	–	0.2667	**0.2700**	–
P@9	**0.2593**	0.2415	**0.2607**	–	0.2548	0.2578	–
P@10	**0.2493**	0.2347	**0.2507**	–	**0.2520**	0.2480	–

Bold values indicate the best performing algorithms

concerned, the linear random node and linear kernel are used. The experimental results show that ranking algorithms based on ELM outperforms state-of-art linear ranking algorithms on most of cases. To effectively and efficiently extend RankELM with nonlinear feature mappings on large scale ranking data remains an interesting topic and will be our future work.

References

1. Cossock D, Zhang T (2008) Statistical analysis of Bayes optimal subset ranking. IEEE Trans Inf Theory 54(11):5140–5154
2. Nallapati R (2004) Discriminative models for information retrieval. In: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '04. ACM, New York, 25–29 July, pp 64–71
3. Li P, Burges CJC, Wu Q (2007) Mcrank: learning to rank using multiple classification and gradientboosting. In: Platt JC, Koller D, Singer Y, Roweis ST (eds) NIPS. MIT Press, Cambridge
4. Vapnik VN (1998) Statistical learning theory. Wiley, New York
5. Suykens JAK, Vandewalle J (1999) Least squares support vectormachine classifiers. Neural Process Lett 9(3):293–300

6. Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. In: Proceedings of the 22nd international conference on machine learning. ICML '05, ACM, New York, pp 89–96
7. Tsai MF, Liu TY, Qin T, Chen HH, Ma WY (2007) Frank: a ranking method with fidelity loss. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '07, ACM, New York, pp 383–390
8. Freund Y, Iyer R, Schapire RE, Singer Y (2003) An efficient boosting algorithm for combining preferences. J Machine Learn Res 4:933–969
9. Joachims T (2002) Optimizing search engines using clickthrough data. In: proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '02, ACM, New York, pp 133–142
10. Pahikkala T, Tsivtsivadze E, Airola A, Boberg J, Salakoski T (2007) Learning to rank with pairwiseregularized least-squares. In: Joachims T, Li H, Liu TY, Zhai C (eds) Proceedings of the SIGIR 2007 workshop on learning to rank for information retrieval. ACM, Amsterdam, Netherlands, pp 27–33
11. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
12. Huang GB, Chen L, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
13. Huang GB, Chen L (2007) Convex incremental extreme learning machine. Neurocomputing 70:3056–3062
14. Huang GB, Chen L (2008) Enhanced randomsearch based incremental extreme learningmachine. Neurocomputing 71:3460–3468
15. Huang GB, Ding X, Zhou H (2010) Optimization method based extreme learning machine for classification. Neurocomputing 74:155–163
16. Huang GB, Zhou H, Ding X, Zhang R (2012) Extreme learningmachine for regression and multi-class classification. IEEE Trans Syst Man Cybern 42(2):513–529
17. Huang GB, Wang D, Lan Y (2011) Extreme learning machines: a survey. Int J Mach Learn Cybern 2(2):107–122
18. Weisstein EW (1995) Laplacianmatrix. [Online]. Available: http://mathworld.wolfram.com/LaplacianMatrix.html
19. Qin T, Liu TY, Xu J, Li H (2010) Letor: a benchmark collection for research on learning to rank for information retrieval. Inf Retr 13(4):346–374
20. Baeza-Yates RA, Ribeiro-Neto B (1999) Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston
21. Järvelin K, Kekäläinen J (2000) In: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '00, ACM, New York, pp 41–48