JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 10, No. 2, June 2016, pp. 58-67

Truncated Kernel Projection Machine for Link Prediction

Liang Huang*
School of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan, China
46399115@qq.com, lianghuang@whpu.edu.cn

Ruixuan Li
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
rxli@mail.hust.edu.cn

Hong Chen
College of Science, Huazhong Agricultural University, Wuhan, China
chenh@mail.hzau.edu.cn

Abstract

With the large amount of complex network data that is increasingly available on the Web, link prediction has become a popular data-mining research field. The focus of this paper is on a link-prediction task that can be formulated as a binary classification problem in complex networks. To solve this link-prediction problem, a sparse-classification algorithm called "Truncated Kernel Projection Machine" that is based on empirical-feature selection is proposed. The proposed algorithm is a novel way to achieve a realization of sparse empirical-feature-based learning that is different from those of the regularized kernel-projection machines. The algorithm is more appealing than those of the previous outstanding learning machines since it can be computed efficiently, and it is also implemented easily and stably during the link-prediction task. The algorithm is applied here for link-prediction tasks in different complex networks, and an investigation of several classification algorithms was performed for comparison. The experimental results show that the proposed algorithm outperformed the compared algorithms in several key indices with a smaller number of test errors and greater stability.

Category: Smart and intelligent computing

Keywords: TKPM; Link prediction; Empirical feature; Reproducing-kernel Hilbert space; Mercer kernel

I. INTRODUCTION

In recent years, complex networks such as social networks have attracted considerable Internet-based attention. A typical complex network can be represented as the graph $G = (V, E)$, where V represents the entities and E represents the relationships between them. For instance, the World Wide Web consists of pages as well as the hyperlinks among them; moreover, social networks consist of individuals and the relations among them. In the field of bioinformatics, the network structures among biological entities such as genes and proteins, which represent physical interactions and gene regulation, are studied extensively.

Complex networks contain highly dynamic objects, and understanding the mechanisms by which they evolve is

Open Access http://dx.doi.org/10.5626/JCSE.2016.10.2.58 http://jcse.kiise.org

Received 25 August 2015; **Revised** 13 April 2016; **Accepted** 21 May 2016

*Corresponding Author

pISSN: 1976-4677 eISSN: 2093-8020

still an open question. For instance, with the prevalence of the Web 2.0, online social network service (SNS) websites such as Facebook, Flickr, and LiveJournal are increasingly popular. The instantaneous and random interactions between millions of people have created uncountable and highly dynamic relationships of online social networks. These connections grow and change quickly over time through the presence or absence of the interactions between people. For those seeking to mine the link behaviors in complex networks, it is urgent to handle complicated situations like those of SNS relationships.

In this paper, the focus is on solving the link-prediction problem of several complex networks with the use of the proposed empirical-feature-based learning algorithm. Many link-prediction algorithms have been proposed over the past few years. The most direct idea is the use of the topological features of complex networks, whereby the hypothesis that two entities that are "close" in a network will have colleagues in common serves as the basis; this shared factor suggests that the "close" entities are more likely to collaborate or interact in the near future. The most well-known model is the preferential-attachment model that was proposed by Newman [1] and Barabasi et al. [2]. Liben-Nowell and Kleinberg [3] proposed and compared a number of metrics such as the shortest-path distances and the numbers of shared neighbors that are derived from various network models.

A more sophisticated approach to the link-prediction task can be formalized as either a binary-classification problem or a ranking problem on the node pairs. This problem leads to the adoption of several link-prediction learning algorithms. Al Hasan et al. [4] used a set of learning algorithms to evaluate the performances of these algorithms in terms of the link-prediction problem, whereby a comparative analysis was made through the identification of a short list of features for a network-based link-prediction task. Kashima and Abe [5] presented a novel probabilistic model of network evolution that is parameterized for the derivation of an efficient incremental-learning algorithm, which is then used to predict the links among the entities in a number of complex networks including biological networks.

In bioinformatics, the prediction of protein-protein interactions (PPIs) has been intensively studied over recent years. A large amount of work has been completed for the prediction of PPIs whereby machine-learning methods such as the support vector machine (SVM) [6, 7] are used. Ben-Hur and Noble [8] used an SVM method with a pairwise kernel and evaluated a number of sequence features including the spectrum Kernel. Magnan et al. [9] used several machine-learning methods with sequence-based features for the prediction of protein solubility. Impressive performances regarding the use of sophisticated pairwise kernels that allow for the achievement of classifications that are derived using machine-learning methods have consequently been reported.

This work, whereby the proposed sparse-learning algorithm that is based on empirical features leads to an improved accuracy and the greater precision of predictions, differs from the earlier works. The contributions of the present paper are as follows:

1) The classification link-prediction algorithm is called *Truncated Kernel Projection Machine*. For the realization of sparse learning that is based on empirical data, a number of new kernel-projection machines that were studied recently in [10-12] are proposed here. The algorithms can be formulated as a coefficient-regularized framework for the reproduction of the kernel-Hilbert space, e.g., the ℓ_0-regularizer in [10] and the ℓ_1-regularizer in [11]. Differing from these regularized methods, a new empirical feature-based method that is based on an empirical risk-minimization principal is proposed here; furthermore, its representer theorem is established, and the easy and efficient implementation of the algorithm is also shown.

2) The effectiveness of the proposed algorithm is evaluated through an empirical comparison of its predictive performance with those of the KPM and the SVM. The results of the experiments of this paper indicate that, for the complex networks that are used, the predictive performances of the proposed method in several key indices including Accuracy, Precision, Recall, and F1-values are significantly more effective than those of the existing methods. As far as the authors know, a link-prediction study that is according to an empirical feature-based sparse method has not been conducted. This study provides a new insight for dealing with the previously mentioned learning tasks, and it fills the gap regarding the empirical evaluations of the previous algorithms in [10-12].

This paper is organized as follows. In Section II, the empirical feature-based learning algorithm with a truncated sparsity is presented with a theoretical analysis. In Section III, the TKPM is applied in several datasets including the UCI Machine Learning Repository, the DBLP coauthor network, and a PPIs network, and the SVM and the KPM are employed for the comparison. Lastly, the conclusion is given in Section IV.

II. ALGORITHM

Let X be a compact metric space and let Y be contained in $[-M, M]$. It is assumed that the product space $Z := X \times Y$ is measurable and that it is endowed with an unknown probability measure that is denoted by ρ. The input-output pairs (x, y) are sampled according to ρ. For every $x \in X$, let $\rho(y|x)$ be the conditional (w.r.t. x) probability measure on Y, and let $\rho_X(x)$ be the marginal probability measure on X. The error for a measurable function $f: X \rightarrow Y$ is the so-called expected risk, as follows:

$$\varepsilon(f) := \| y - f \|_{L^2_\rho}^2 = \int_Z (y - f(x))^2 d\rho.$$

It is known that the function that minimizes $\varepsilon(f)$ is the regression function that is defined by the following formula:

$$f_\rho(x) = \int_Y y d\rho(y|x),\ x \in X.$$

Set $\mathbb{N}_m \coloneqq \{1,2,..., m\}$ for any $m \in \mathbb{N}$. A training set of size m is drawn by sampling m independently and the identically distributed pairs according to ρ, as follows:

$$\mathbf{z} \coloneqq \{z_i,\ i \in \mathbb{N}_m\} = \{(x_i,\ y_i),\ i \in \mathbb{N}_m\} \in Z^m.$$

The design of the learning algorithm that is investigated in this paper is based on a reproducing kernel-Hilbert space $\mathcal{H}_\mathrm{k}$ that is associated with a Mercer kernel K. An integral operator on $\mathcal{H}_\mathrm{k}$ is defined by the following formula:

$$L_K(f) = \int_X K_x f(x) d\rho_x(x) \in \mathcal{H}_K.$$

Note that L_K is a compact, self-adjoint positive operator; its eigenvalues are denoted with a non-increasing sequence as $\{\lambda_i\}$, and the associated eigenfunctions are denoted using $\{\phi_i\}$. Here, $\{\phi_i\}$ forms the orthonormal basis of $\mathcal{H}_K$. The power r of L_K is also defined using the following formula:

$$L_K^r\left(\sum_i c_i \phi_i\right) = \sum_i c_i \lambda_i^r \phi_i.$$

In this paper, it is assumed that the following equation applies:

$$f_\rho = L_K^r(g_\rho)\text{, for some } r>0 \text{ and } g_\rho \in \mathcal{H}_k. \tag{1}$$

This assumption has been used extensively in machine-learning algorithms, e.g., [11, 13]. If $g_\rho = \sum_i d_i \phi_i$ and $\{d_i\} \in l^2$, then $f_\rho = \sum_i d_i \lambda_i^r \phi_i$; furthermore, $\mathbf{x} = \{x_i\}_{i=1}^m$. The empirical features $\{\phi_i^x\}$ approximate $\{\phi_i\}$ here, where $\{\phi_i^x\}$ represents the eigenfunctions of the empirical operator L_K^x that is defined by the following formula:

$$L_K^X(f) = \frac{1}{m}\sum_{i=1}^m f(X_i) K_{X_i} = \frac{1}{m}\sum_{i=1}^m < K_{X_i}, f >_K K_{X_i},\ f \in \mathcal{H}_K,$$

where L_K^x is a normalized sum of m rank-one operators, and it is self-adjoint and positive with the rank at most m. The eigensystem of L_K^x is denoted by $\{(\lambda_i^x,\ \phi_i^x)\}$, where λ_i^x is arranged in a non-increasing order, and $\lambda_i^x = 0$ when $i > m$ and $\{\phi_i^x\}$ forms the orthonormal basis of $\mathcal{H}_\mathrm{k}$.

The first m eigenfunctions $\{\phi_i^x\}_{i=1}^m$ can be considered the empirical features for learning, and they can be computed easily with the use of the Gramian matrix $\mathbf{K} = (K(x_i,\ x_j))_{i,j=1}^m$ (see Remark 1 in [11] for details).

The algorithms that are based on the empirical features are introduced in [10, 11], whereby the coefficient regularization is derived through the use of the following formula:

$$c_\gamma^z \coloneqq \arg\min_{c \in \mathbb{R}^m} \left\{ \frac{1}{m}\sum_{i=1}^m v\left(\sum_{j=1}^k c_j\ \phi_j^x(x_i),\ y_i\right) + \gamma\Omega(c),\ \gamma > 0 \right\}, \tag{2}$$

where $V : \mathbb{R}^2 \to \mathbb{R}_+$ is a loss function, $\gamma > 0$ is a regularization parameter, and $\Omega(c)$ is the penalty on the coefficients. In [10, 14], $V(y,f(x)) = (y - f(x))^2$, and $\Omega(\mathrm{c}) = \|c\|_{\ell_1} = \sum_{i=1}^m |c_i|$.

The empirical feature-based learning by truncation that is different from the regularized algorithms in [10, 11], is the focus of this paper. A truncated parameter $\varepsilon > 0$ is therefore introduced to control the feature domain $D_{m,\varepsilon} = \{\phi_i^x : i \in N_m,\ \lambda_i^{\mathbf{x}} \geq \varepsilon\} = \{\phi_i\}_{i=1}^k$, where k denotes the number of features in $D_{m,\varepsilon}$. Through the use of the learning algorithm, an empirical risk minimization is implemented to derive the coefficient, as follows:

$$c^{z,k} = (c_1^{z,k},..., c_k^{z,k}) = \arg\min_{c \in \mathbb{R}^k} \frac{1}{m}\sum_{i=1}^m \left(\sum_{j=1}^k c_j\ \tilde{\phi}_j(x_i) - y_i\right)^2. \tag{3}$$

Then, the predictor is $f_k^z = \sum_{i=1}^k c_i^{z,k} \tilde{\phi}_j$.

Now, the representer theorem for the algorithm in (3) can be given.

Theorem 1. *Let $\tilde{\lambda}_i$ be the corresponding eigenvalue of $\tilde{\phi}_i$ for $i \in \mathbb{N}_{\boldsymbol{k}}$. The coefficient $c^{z,k} = (c_1^{z,k},..., c_k^{z,k})$ in (3) is given by*

$$\boldsymbol{c_i^{z,k} = \frac{1}{m\tilde{\lambda}}\sum_{j=1}^m y_j \tilde{\phi}_i(x_j)}.$$

Proof. Since $(\tilde{\lambda}_i,\ \tilde{\phi}_i) \in \{\boldsymbol{\lambda_j^x,\ \phi_j^x}\}_{\boldsymbol{j=1}}^{\boldsymbol{m}}$ for $i \in \mathbb{N}_{\boldsymbol{k}}$, the following formula is derived:

$$\tilde{\lambda}_i\ \tilde{\phi}_i = \boldsymbol{L_k^x \tilde{\phi}_i} = \boldsymbol{\frac{1}{m}\sum_{j=1}^m \tilde{\phi}_i(x_j) K_{x_j}}.$$

As $\{\phi_j^x\}$ forms the orthonormal basis of $\mathcal{H}_k$ and $\tilde{\phi}_i \in \{\phi_j^x\}$, it is possible to observe the following formula:

$$\delta_{i,l}\tilde{\lambda}_i = \langle \tilde{\lambda}_i\ \tilde{\phi}_i,\ \tilde{\phi}_l \rangle_k = \frac{1}{m}\sum_{i=1}^m \tilde{\phi}_i(x_j)\tilde{\phi}_l(x_j), i,l \in \mathbb{N}_k$$

where $\delta_{i,l} = 1$ if $i = l$, and $\delta_{i,l} = 0$ otherwise. The empirical error part can then be rewritten as follows:

$$\frac{1}{m}\sum_{i=1}^m\left(\sum_{j=1}^k c_j\tilde{\phi}_j(x_i) - y_i\right)^2 = \sum_{p,q=1}^k c_p c_q \frac{1}{m}\sum_{i=1}^m \tilde{\phi}_p(x_i)\tilde{\phi}_q(x_i)$$

$$-\frac{2}{m}\sum_{i=1}^m\sum_{j=1}^k y_i c_j \tilde{\phi}_j(x_i) + \frac{1}{m}\sum_{i=1}^m y_i^2 = \sum_{p,q=1}^k c_p c_q \delta_{p,q}\tilde{\lambda}_p$$

$$-2\sum_{j=1}^k c_j\left(\frac{1}{m}\sum_{i=1}^m y_i\tilde{\phi}_j(x_i)\right) + \frac{1}{m}\sum_{i=1}^m y_i^2 = \sum_{j=1}^k\left(\tilde{\lambda}_j c_j^2 - \frac{2c_j}{m}\sum_{i=1}^m y_i\tilde{\phi}_j(x_i)\right) + \frac{1}{m}\sum_{i=1}^m y_i^2.$$

The desired result is then derived through the setting of the following formula:

$$\frac{\partial \frac{1}{m}\sum_{i=1}^{m}\left(\sum_{j=1}^{k} c_j \tilde{\phi}_j(x_i) - y_i\right)^2}{\partial c_j} = 0.$$

Here, the sparsity is controlled by the truncated threshold ε. The details of the Truncated Kernel Projection Machine (TKPM) can now be given by Algorithm 1, as follows:

Algorithm 1. Truncated Kernel Projection Machine

```
Input: Let $\varepsilon > 0$ be the truncated threshold, whereby the training set $z = \{(x_i, y_i)\}_{i=1}^{m}$, the test set $z = \{(x_i, y_i)\}_{i=1}^{m}$, and the kernel function $K$ that is defined on $z$ are given.
Output: The classifier sgn ($f_k^z$)
1: Compute the kernel matrix $\mathbf{K} = (K(x_i, x_j))_{i,j=1}^{m}$ and its eigenvalue sectors $\{\hat{\lambda}_i^x, \hat{u}_i\}$ ;
2: Transformation:
    $\lambda_i^x = \frac{\hat{\lambda}_i^x}{m},\ \phi_i^x = \frac{1}{\sqrt{\hat{\lambda}_i^x}}\sum_{j=1}^{m}(\hat{u}_i)_j K_{x_j},\ \hat{\lambda}_i^x > 0$ ;
3: Feature selection:
    $D_{m,\varepsilon} = \{\phi_i^x : i \in N_m,\ \lambda_i \geq \varepsilon\} = \{\tilde{\phi}_i\}_{i=1}^{k}$ ;
4: Calculating the coefficient
    $c_i^{z,J,k} = \frac{1}{m\hat{\lambda}_i}\sum_{j=1}^{m} y_j \tilde{\phi}_i(x_j)$ of the classifier sgn ($f_k^z$).
```

The proposed binary classifier can be applied in a number of link-prediction datasets by predicting whether a specific link in the network should be added or not; the detailed steps regarding the procedure for the application of the proposed algorithm in the link-prediction problems are provided in the experiments, i.e., the third paragraph in subsection B of Section III. The TKPM was tested for the link-prediction tasks of three different datasets in the experiments. The results show that the proposed TKPM can be implemented easily and efficiently.

III. EXPERIMENTS

The TPKM was tested in several complex networks. In each experiment, three algorithms including the TKPM, the KPM, and the SVM with the RBF kernel were employed for the comparison. The same sample sets and test sets are adopted in each comparison experiment. The TKPM and the KPM are fixed with the same eigen numbers $D = Numbers_{eigenvalue > truncated\ threshold}$. The TKPM and the KPM were implemented in MATLAB R2009a in Linux. For the SVM, the free C library of libsvm is employed in MATLAB.

A. Test TKPM in Several Benchmark Datasets

Six benchmark datasets that consist of some data that are originally from the University of California at Irvine (UCI) repository were taken from [15] and were used in the TKPM test. All of the datasets consist of 100 realizations that had been split into a training sample and a test sample. In all of the datasets except "Banana," the 100 realizations actually contain the same data points and only differ regarding the training/test split. The results obtained from the application of several state-of-the-art classification algorithms have been reported in [15], including the SVM with the RBF kernel. Several optimal results are reported on the website, along with the suitable values (chosen by cross-validation) for the parameters σ (kernel width) and C_G (SVM-regularization parameter). These parameters are also employed in the experiments of this paper.

Table 1 lists the best results of TKPM, KPM, and SVM for each dataset; furthermore, the corresponding truncated thresholds and the parameters D of KPM are also listed. The results are in the form of *average errors ± squared test errors*. Noticeably, the KPM is less competitive here, and the KPM results that are reported in [14] are also included in Table 1 (named "KPM(R)"). Both the TKPM and the SVM perform similarly in all of the datasets. The TKPM performances are better than those of the other algorithms in nearly all of the datasets, with

Table 1. Best test errors of the three algorithms

	SVM	KPM	KPM(R)	D	TKPM	Threshold
Banana	11.56 ± 0.37	17.69 ± 0.63	10.91 ± 0.57	12	**10.52 ± 0.64**	0.15 (40)
Breast Cancer	26.04 ± 0.47	28.81 ± 0.45	28.73 ± 4.42	1	**25.22 ± 0.44**	0.26 (19)
Diabetes	22.53 ± 0.38	24.52 ± 0.32	23.77 ± 1.69	17	**23.17 ± 0.38**	2.1 (17)
Flare Solar	**32.43 ± 0.98**	35.86 ± 0.53	32.52 ± 1.78	8	33.73 ± 0.47	2.1 (8)
German	23.61 ± 0.21	26.77 ± 0.46	24.09 ± 2.38	11	**23.58 ± 0.21**	10 (11)
Heart	15.97 ± 0.32	41.24 ± 1.38	17.53 ± 3.54	20	**15.74 ± 0.28**	0.85 (13)

Values are presented as average errors ± squared test errors.
D is the best parameter of KPM and KPM(R) in each dataset, and threshold is the best parameter of TKPM in each dataset.

Table 2. Best classifier in the family

	SVM	KPM	TKPM
Banana	3	2	**95**
Breast Cancer	27	11	**47**
Diabetes	19	23	**45**
Flare Solar	**40**	23	8
German	**41**	10	35
Heart	31	1	**36**

The numbers in the table denote the number of wins.

"Flare Solar" being the only exception; meanwhile, the TKPM dominates both KPM and KPM(R) in each of the datasets.

In Table 2, for each dataset, the smallest test error of the TKPM is compared with those of the SVM and the KPM; each time, the winner is given one point. The TKPM won more points in four of the datasets, the SVM won in the Flare Solar and "German" datasets, and the TKPM still dominates the KPM in each of the datasets except for Flare Solar. These two tables highlight the superiority of the TKPM over the KPM.

Figs. 1–3 show the average test errors of the TKPM compared to the KPM for the same truncated threshold (D). For each dataset, two algorithms are tested in the eight truncated thresholds $\in \{0.1, 0.2, 0.5, 1, 1.5, 2, 5, 10\}$. In both figures, the average test errors of the TKPM are smaller than those of the KPM regarding the same truncated threshold (D). The performances of the TKPM are very stable in nearly all of the datasets. The curves of the TKPM are nearly straight lines in Figs. 1–3, while those of the KPM change very sharply. The above results show us that, when compared to the KPM, the TKPM can achieve better results while it is not sensitive to the truncated thresholds.

B. Test TKPM in DBLP Coauthor Network

In this subsection, a basic computational problem that underlies the social network evolution, the link-prediction problem of social networks, is defined and studied. Given a snapshot of a social network at time *t*, the aim here is the accurate prediction of the edges that will be added to the network during the interval from time *t* to a given future time *t*'.

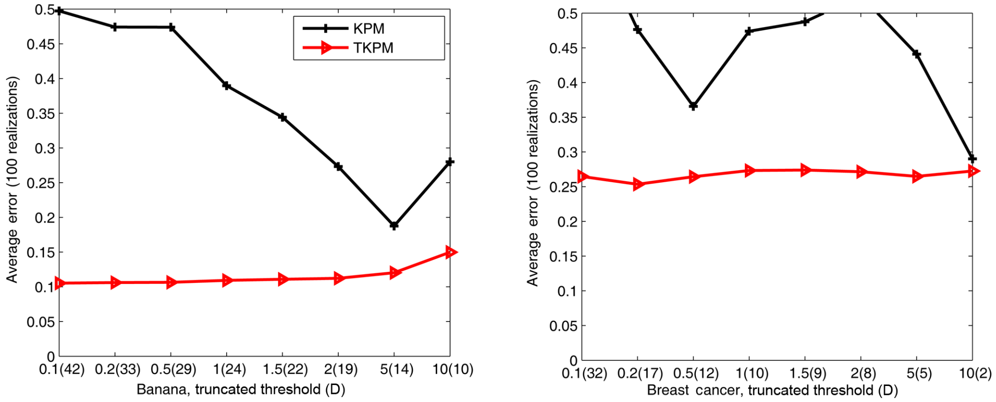

Fig. 1. The average errors of KPM and TKPM in different truncated thresholds (D) in Banana and Breast Cancer.

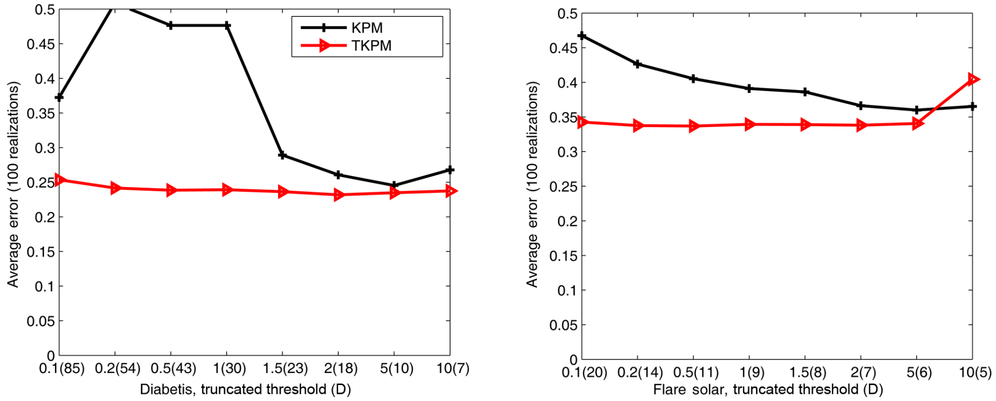

Fig. 2. The average errors of KPM and TKPM in different truncated thresholds (D) in Diabetes and Flare Solar.

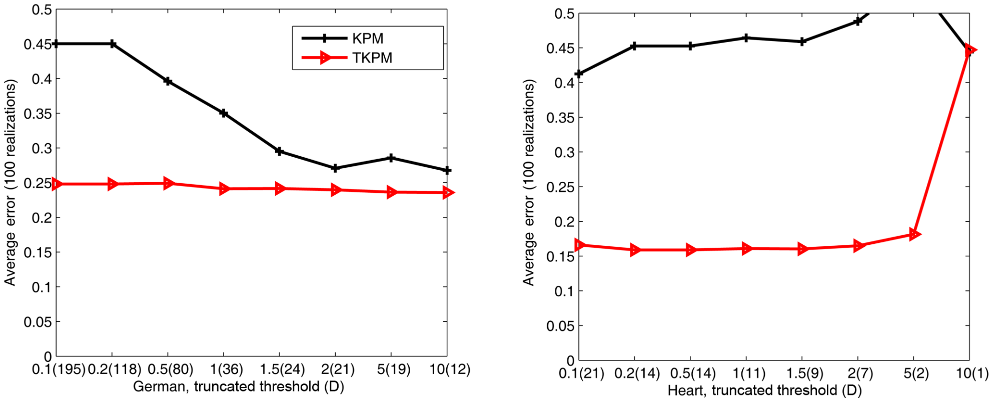

Fig. 3. The average errors of KPM and TKPM in different truncated thresholds (D) in German and Heart.

The DBLP [16] dataset that is used in this experiment provides the bibliographic information of major computer science journals and proceedings. The dataset indexes more than 2.5 million articles and 1 million authors from a period of several decades. An XML document of the article information including the title, author, journal, and year is also provided with the DBLP dataset. Papers that have been published in 18 journals in the field of artificial intelligence and pattern recognition between 1996 and 2010 were chosen for this study. These papers were then split into two sets so that the papers from the years 1996 to 2003 could be used as the training set and those from the years 2004 to 2010 could be used as the test set.

A coauthor relationship is defined if two authors are the coauthors of a paper in the dataset. The training set with 7,729 authors and 26,334 relationships, and the test set with 7,729 authors and 42,185 relationships were constructed in this way.

An experimental setup that is similar to that of [4] was adopted for these experiments. The classification dataset was constructed through the selection of the author pairs that appear in the training set, but that did not publish any papers during those years. Each pair represents either a positive label or a negative label, depending on whether or not the author pairs published at least one paper in the test set. The six features that are reported in [3] were used in the experiments including graph distance, common neighbors, Jaccard's coefficient, Adamic/Adar, preferential attachment, and Katz$_\beta$.

1) Results

Similar to the last subsection, the kernel parameter σ was set to 0.8, and C_G = 16 for the three algorithms. The standard five-fold cross-validation was employed for the verification of the results with the six truncated thresholds $\in \{0.1(148),0.5(107),1(66),3(21),5(15),10(10)\}$. Four basic indices including accuracy, precision, recall, and F1-value formed the focus of each experiment.

Let TP represent the true positive, TN represent the true negative, FP represent the false positive, and FN represent the false negative. The definitions of the four indices can be given as follows:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F1\ value = \frac{2*Precision*Recall}{Precision+Recall}\ .$$

Table 3 lists the performances of the three algorithms, whereby the results are presented in the form of *each index ± squared error*. The TKPM won in terms of the accuracy, recall, and F1-value indices for nearly all of the truncated thresholds (D), as shown by the bold values in Table 3. The KPM won the precision index with a very low recall in each truncated threshold (D). The best performances in terms of accuracy, recall and F1-value were achieved by the TKPM, as shown by the bold values that are also italicized in Table 3. The KPM won the precision index with the lowest recall. Table 3 also shows that the KPM is very unstable regarding the test with the larger test error and changes. Due to the difference regarding the selection of the DBLP datasets and features, the average results of the experiments here are relatively lower than those of [4]. It remains persuasive, however, that the TKPM can achieve better performances in a wide range of truncated thresholds with smaller test errors.

Fig. 4 shows the error bar of the KPM and the TKPM. Both of the TKPM curves are nearly straight lines with very small squared errors; yet, the KPM curves change frequently with large squared errors. The average results of the three algorithms are relatively lower than those of [4] due to the differences between the datasets and parameters of the two studies; however, the results still prove that the TKPM can achieve better performances in

Table 3. Performances of three algorithms for DBLP

Method	Accuracy	Precision	Recall	F1-value	Threshold (D)
KPM	55.23 ± 0.93	**91.04±8.05**	17.41±9.13	22.72±8.64	**0.1 (148)**
TKPM	**66.58 ± 0.47**	79.34±0.80	**44.96±0.92**	**52.37±0.86**	-
KPM	56.31 ± 1.21	**86.50±5.33**	21.46±13.2	25.51±7.91	0.5 (107)
TKPM	**66.44 ± 0.46**	79.32±0.76	**44.59±1.00**	**57.06±0.92**	-
KPM	53.32 ± 1.09	**89.50±9.03**	27.36±16.1	26.79±8.69	1 (66)
TKPM	**66.36 ± 0.45**	79.35±0.79	**44.35±1.01**	**56.87±0.91**	-
KPM	56.56 ± 1.82	**95.30±9.39**	14.21±17.2	22.95±8.95	3 (21)
TKPM	**65.76 ± 0.53**	79.16±0.79	**42.92±1.19**	**55.62±1.07**	-
KPM	60.96 ± 1.69	**93.61±8.21**	23.77±3.91	36.89±5.70	5 (15)
TKPM	**66.04 ± 0.47**	79.51±0.88	**43.36±0.96**	**56.08±0.88**	-
KPM	56.54 ± 2.31	**96.02±10.5**	13.90±5.43	22.20±8.02	10 (10)
TKPM	**65.28 ± 0.14**	78.40±0.77	**42.36±0.94**	**54.96±0.80**	-
SVM	65.92 ± 0.40	75.40±0.71	42.96±0.91	53.41±0.93	-

Values are presented as each index ± squared errors.
$\sigma = 0.8$, truncated threshold (D) $\in$ {0.1(148),0.5(107),1(66),3(21),5(15),10(10)}.

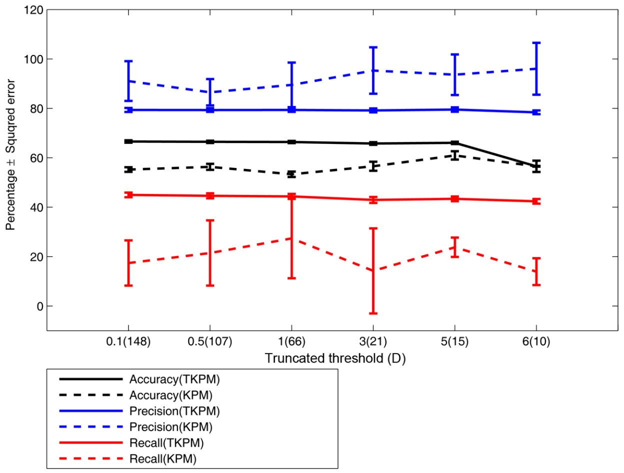

Fig. 4. Error bar of accuracy, precision, and recall of TKPM and KPM in different truncated thresholds (D). The data are represented as percentage ± squared error.

a wide range of truncated thresholds with small test errors.

C. Test TKPM in PPIs Network

In this subsection, a prediction of PPIs that is based only on the amino acid sequence information is the focus. PPIs are central to most biological processes, e.g., metabolic pathways, signaling cascades, and transcription control processes.

Shen et al. [17] proposed a descriptor named "conjoint triad," which is for the properties of one amino acid and its vicinal amino acids in the amino acid sequence of a protein, and regarded any three continuous amino acids as a unit. The triads can therefore be differentiated according to the amino acid classes, i.e., triads composed of three amino acids belonging to the same class. Shen defined 343 amino acid triad classes, and each protein can be represented by a 343-dimension vector. The vector count is the number of occurrence times for each triad in the protein sequence. A protein-protein pair can therefore be represented by a 686-dimension vector through the concatenation of the vectors of two proteins.

In [17], the SVM with a kernel function was adopted for the prediction of the PPIs, and sound results were achieved. In the experiments of this study, the method of [17] was used to extract the features of the PPIs. The results of the TKPM were compared to those of the SVM and the KPM with the same kernel function and parameters.

1) Dataset

The Database of Interacting Proteins (DIP) [18] was adopted for this experiment. The DIP is a database that documents experimentally determined PPIs, providing the scientific community with an integrated set of tools for browsing and the extraction of information regarding the protein interaction networks. As of October 2011, the DIP catalog comprises approximately 73,024 unique interactions from 24,281 proteins that belong to 447 organisms; the vast majority are from yeast, Helicobacter pylori, and the human species.

The *S. cerevisiae* (baker's yeast) that is composed of 4,525 PPIs was taken from the DIP and used for the experimental dataset; therefore, a positive dataset with

4,525 protein-protein pairs was constructed using the method of [17], and each protein-protein pair comprises 686 features. The amino-acid sequence of each protein-protein pair was then shuffled according to the laws of k-let; accordingly, two different datasets were constructed including the 1-let and 2-let datasets, and each negative dataset comprises 4,525 negative protein-protein pairs. The standard five-fold cross-validation was employed for the comparison between the performances of the TKPM, the SVM, and the KPM.

2) Results

In the experiment, the TKPM, SVM, and KPM were compared. The σ was set to 2.5 and the C_G (SVM-regularization parameter) was set to 8. The eigen numbers were still fixed to $D = Numbers_{eigenvalue>truncated\ threshold}$. The six thresholds ∈ {0.1(3958),0.5(2288),1(694),3(133),5(48),10(12)} were employed for the 1-let dataset and the six thresholds ∈ {0.1(3931),0.5(2183),1(684),3(132),5(50),10(13)} were employed for the 2-let dataset here. The results were presented for the three indices accuracy, precision, and sensitivity in the form of percentage ± squared error. The definition of sensitivity here is the same as that of recall, as was previously mentioned. Tables 4 and 5 list the results of the experiment, wherein the three algorithms achieved good results for both the 1-let and the 2-let datasets. The TKPM outperformed the other algorithms in terms of accuracy and precision in the 1-let dataset for each truncated threshold (D) with a relatively small squared error; however, the KPM won the sensitivity index for each truncated threshold (D) in the 1-let dataset. The performance of the TKPM is comparable to that of the KPM for the 2-let dataset, but the former still won more indices in more truncated thresholds (D). The TKPM performed best in the accuracy and precision indices, and the KPM performed best in the sensitivity index, as shown by the italicized, bold values in Tables 4 and 5. The results of the 1-let dataset dominated the 2-let dataset results. The classification results that can be achieved with a 1-let dataset are therefore superior to those of the 2-let dataset.

IV. CONCLUSION

A new algorithm that is used for the prediction of the links in complex networks is proposed in this paper with an empirical-feature-based learning algorithm called TPKM that comprises a truncated sparsity.

The performance of the algorithm was applied to different instances of complex networks for the comparison with the KPM and the SVM. The experiment results show that the results that can be obtained with the TKPM are superior to those of the SVM. The TKPM outperformed the KPM and SVM in nearly all of the datasets with high stability. The TKPM curves are like straight lines compared to those of the KPM which change sharply. These results show that the TKPM is not sensitive to the truncated threshold (D) when compared to the KPM.

The computational cost of the TKPM still outperforms

Table 4. Performances of different algorithms for DIP *S. cerevisiae* with 1-let

Method	Accuracy	Precision	Sensitivity	Threshold (D)
KPM	92.89±0.39	90.29±1.45	**95.03±1.01**	0.1 (3958)
TKPM	**94.72±0.35**	**95.02±0.79**	93.78±0.54	-
KPM	91.52±0.41	87.54±1.33	**94.11±0.98**	0.5 (2288)
TKPM	**93.74±0.31**	**93.81±0.75**	92.95±0.69	-
KPM	88.72±0.46	85.33±1.51	**92.02±1.49**	1 (694)
TKPM	**90.62±0.58**	**90.81±0.17**	89.36±1.11	-
KPM	**85.34±0.53**	83.52±1.25	**86.27±0.88**	3 (133)
TKPM	85.11±0.47	**85.48±1.21**	82.81±0.55	-
KPM	81.56±0.44	79.17±1.62	**83.14±0.98**	5 (48)
TKPM	**81.94±0.61**	**82.85±1.46**	78.45±0.96	-
KPM	48.08±2.23	47.94±2.28	**59.61±0.93**	10 (12)
TKPM	**57.28±0.42**	**55.85±2.45**	55.70±2.40	-
SVM	93.13±0.40	91.40±0.71	87.65±0.91	-

Values are presented as percentage ± squared error.
σ = 2.5, truncated threshold (D) ∈ {0.1(3958),0.5(2288),1(694),3(133), 5(48),10(12)}.

Table 5. Performances of different algorithms for DIP *S. cerevisiae* with 2-let

Method	Accuracy	Precision	Sensitivity	Threshold (D)
KPM	88.32±0.41	87.13±1.98	**87.93±0.38**	0.1 (3931)
TKPM	**89.51±0.38**	**87.74±1.87**	84.31±0.21	-
KPM	85.98±0.21	**86.29±1.01**	81.49±1.52	0.5 (2183)
TKPM	**86.42±0.50**	85.93±1.98	**85.87±0.46**	-
KPM	**82.34±0.63**	**80.58±1.51**	76.58±0.49	1 (684)
TKPM	81.01±0.57	80.24±1.84	**79.64±0.80**	-
KPM	**75.52±0.63**	74.15±0.27	69.75±0.57	3 (132)
TKPM	74.71±0.41	**76.42±1.52**	**73.26±0.93**	-
KPM	69.92±0.75	**71.11±1.69**	61.49±1.10	5 (50)
TKPM	**70.68±0.61**	69.62±2.35	**66.35±5.15**	-
KPM	50.78±0.52	48.62±1.86	**81.19±6.61**	10 (13)
TKPM	**51.28±0.49**	**51.18±2.62**	38.55±5.83	-
SVM	85.84±0.61	83.47±1.32	86.54±1.33	-

Values are presented as percentage ± squared error.
σ = 2.5, truncated threshold (D) ∈ {0.1(3931),0.5(2183),1(684),3(132), 5(50),10(13)}.

that of the KPM, the latter of which needs to solve a linear optimizing problem with a large cost. In the experiments, KPM memory errors were reported frequently, whereas TKPM errors were rarely reported and they comprise more eigen numbers. In the experiments, the speed of the SVM is faster than that of the TKPM, whereby a different implementation caused the different speeds. The optimized and tested libsvm library in C, TKMP, and KPM were implemented in MATLAB.

The proposed algorithm will not only allow for the extension of link prediction and link analysis to different kinds of complex networks, but it will also provide a useful tool for understanding the structure and evolution of these networks.

Two different future directions are evident regarding this work, as follows: 1) following an analysis, the TKPM can be extended into the area of semi-supervised machine learning, whereby better results could be achieved. 2) The methods that have been used in the experiments of this study are limited by static time periods. The time-series-analysis algorithm can be introduced here for the achievement of a better performance regarding the link-prediction task.

It is hoped that the TKPM will be employed successfully in the prediction of new links and missing links in complex networks, and will help to uncover new and interesting properties in this area.

In a future study, the proposed algorithm will be extended through a binding of the graph kernel [19] with the data dependence. The achievement of a better performance is sought through the use of this extended algorithm.

AKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China (NSFC) under Grant No. 11001092 and the Fundamental Research Funds for the Central Universities (Program No. 2011PY130, 2011QC022).

REFERENCES

1. M. E. Newman, “Clustering and preferential attachment in growing networks,” *Physical Review E*, vol. 64, no. 2, article ID. 025102, 2001.
2. A. L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek, “Evolution of the social network of scientific collaborations,” *Physica A: Statistical Mechanics and Its Applications*, vol. 311, no. 3, pp. 590-614, 2002.
3. D. Liben-Nowell and J. Kleinberg, “The link prediction problem for social networks,” in *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM'03)*, New Orleans, LA, 2003, pp. 556-559.
4. M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, “Link prediction using supervised learning,” in *Proceedings of 4th Workshop on Link Analysis, Counter-terrorism and Security*, Bethesda, MD, 2006.
5. H. Kashima and N. Abe, “A parameterized probabilistic model of network evolution for supervised link prediction,” in *Proceedings of 6th International Conference on Data Mining (ICDM'06),* Hong Kong, 2006, pp. 340-349.
6. W. S. Noble, “Support vector machine applications in computational biology,” in *Kernel Methods in Computational Biology*, Cambridge, MA: MIT Press, pp. 71-92, 2004.
7. Y. Guo, L. Yu, Z. Wen, and M. Li, “Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences,” *Nucleic Acids Research*, vol. 36, no. 9, pp. 3025-3030, 2008.
8. A. Ben-Hur and W. S. Noble, “Kernel methods for predicting protein–protein interactions,” *Bioinformatics*, vol. 21, no. suppl 1, pp. i38-i46, 2005.
9. C. N. Magnan, A. Randall, and P. Baldi, “SOLpro: accurate sequence-based prediction of protein solubility,” *Bioinformatics*, vol. 25, no. 17, pp. 2200-2207, 2009.
10. L. Zwald, G. Blanchard, P. Massart, and R. Vert, “Kernel projection machine: a new tool for pattern recognition,” *Advances in Neural Information Processing Systems*, vol. 17, pp. 1649-1656, 2005.
11. X. Guo and D. X. Zhou, “An empirical feature-based learning algorithm producing sparse approximations,” *Applied and Computational Harmonic Analysis*, vol. 32, no. 3, pp. 389-400, 2012.
12. L. Zwald and G. Blanchard, “On the convergence of eigenspaces in kernel principal component analysis,” *Advances in Neural Information Processing Systems*, vol. 18, pp. 1649-1656, 2006.
13. S. Smale and D. X. Zhou, “Online learning with Markov sampling,” *Analysis and Applications*, vol. 7, no. 1, pp. 87-113, 2009.
14. G. Blanchard and L. Zwald, “Finite-dimensional projection for classification and statistical learning,” *IEEE Transactions on Information Theory*, vol. 54, no. 9, pp. 4169-4182, 2008.
15. G. Raetsch, “Benchmark repository used in several Boosting, KFD, and SVM papers,” Available: http://archive.ics.uci.edu/ml/datasets.html.
16. DBLP dataset, http://dblp.uni-trier.de/xml/.
17. J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang, “Predicting protein–protein interactions based only on sequences information,” *Proceedings of the National Academy of Sciences*, vol. 104, no. 11, pp. 4337-4341, 2007.
18. Database of Interaction Proteins, http://dip.doe-mbi.ucla.edu.
19. S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, “Graph kernels,” *Journal of Machine Learning Research*, vol. 11, pp. 1201-1242, 2010.

Liang Huang

Liang Huang received the B.Sc. degree from Hubei University, Wuhan, China, in 2003, and received Ph.D. degree from Huazhong University of Science and Technology in 2012. He is a lecturer at School of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan, China. His current research interests include social networks and data mining.

Ruixuan Li

Ruixuan Li received the B.S., M.S., and Ph.D. degrees in Computer Science from Huazhong University of Science and Technology, China, in 1997, 2000, and 2004, respectively. He was a Visiting Researcher in Department of Electrical and Computer Engineering at University of Toronto from 2009 to 2010. He is currently a Professor in the School of Computer Science and Technology at Huazhong University of Science and Technology. His research interests include peer-to-peer computing, distributed data management, and distributed system security. He is a member of IEEE and ACM.

Hong Chen

Hong Chen received the B.Sc. degree and the Ph.D. degrees from Hubei University, Wuhan, China, in 2003 and 2009, respectively. During 2016, he stayed with the University of Texas at Arlington as a post-doctoral researcher. He is an Associate Professor at the Institute of Applied Mathematics, College of Science, Huazhong Agricultural University, China. His current research interests include machine learning, approximation theory, and pattern recognition.