Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 320 (2004) 71–87
Theoretical Computer Science
www.elsevier.com/locate/tcs

Neural computation, social networks, and topological spectra

Michelangelo Diligenti*, Marco Gori, Marco Maggini
Dipartimento di Ingegneria dell'Informazione, Università di Siena, Siena, Italy

Abstract

This paper emphasizes some intriguing links between neural computation on graphical domains and social networks, like those used in nowadays search engines to score the page authority. It is pointed out that the introduction of web domains creates a unified mathematical framework for these computational schemes. It is shown that one of the major limitations of currently used connectionist models, namely their scarce ability to capture the topological features of patterns, can be effectively faced by computing the node rank according to social-based computation, like Google's PageRank. The main contribution of the paper is the introduction of a novel graph spectral notion, which can be naturally used for the graph isomorphism problem. In particular, a class of graphs is introduced for which the problem is proven to be polynomial. It is also pointed out that the derived spectral representations can be nicely combined with learning, thus opening the doors to many applications typically faced within the framework of neural computation.

Keywords: Neural computation; Social networks; Topological spectra; PageRank

1. Introduction

A strong limitation of most connectionist-based models is that they are not well-suited for capturing topological features, which often play a crucial role in decision making. This limitation is due to the flat data representation currently adopted also in other machine learning approaches, where the links amongst samples of the training set are not typically taken into account. For instance, in some fields like pattern recognition, the application of widely disseminated multilayer networks requires the dissipation of learning capabilities to incorporate translation and rotation invariance. As a consequence, Backpropagation cannot focus only on the actual detection of the

* Corresponding author.

0304-3975/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.tcs.2004.03.044

distinguishing features of the patterns, thus limiting its capabilities. This limitation has been recognized by many researchers and, recently, the development of more general learning models capable of dealing with structured domains has been the subject of detailed investigation [11]. Basically, instead of processing a flat representation, new models have been conceived which operate on graphs, thus extending the notion of learning environment to a collection of graphs.

Interestingly, the computation of the rank of web pages proposed in [6], which is formulated into the framework of social networks and which is very much related to the previous work on the estimation of qualifications in self-evaluating groups [2,3], follows a computational scheme which resembles very much the one adopted in neural networks for graphical domains [10]. The model adopted for computing the PageRank is different with respect to the developed neural models in structured domains in that it does not learn parameters, but nicely operates on a unique domain, namely the Web, by a computation which is guaranteed to converge thanks to the special linear structure of the model.

The main contribution of the paper is the introduction of a novel graph spectral notion, which can be naturally used for the graph isomorphism problem. This notion is based on the page rank regarded as a function of the damping factor and turns out to be very adequate to face the graph isomorphism problem. In particular, it is proven that the problem is polynomial for a given class of graphs, which does not appear straightforwardly reducible to already known classes for which the isomorphism is known to be polynomial. The result is not only interesting in itself, but also for its potential implications in partial graph matching and related problems that can be framed as learning in web domains, which are abstractions of the Web in that they are a unique, typically huge, graph over which a function is defined. The aim of web learning is to infer that function relying on the knowledge of a subgraph of the Web. Notice that the Web is just an example of a web domain, which could be the appropriate abstraction for different problems, especially in pattern recognition. Unlike the developed learning models for structured domains, we deal with the case in which data are embedded into a unique graph, the Web, not in a collection. We give a general view of the notion of web supervised learning in the framework of function optimization which incorporates many interesting models recently proposed in the literature.

The paper is organized as follows. In Section 2, we introduce the notion of web domains and present recursive neural networks and social networks for page rank as special cases. In Section 3 we introduce the notion of topological spectra while in Section 4, we face the problem of reconstructing the graph from the knowledge of its spectrum and propose a simple polynomial scheme for solving the graph isomorphism. In Section 5, a discussion is proposed for extending the spectral concepts within the more general framework of learning, while in Section 6 some conclusions are drawn.

2. Functions on web domains

The theory of multi-dimensional systems [4] can be given a nice extension in the case in which the domain of the function is a graph. Instead of the traditional grid on

which functions are defined, the domain can be generalized to a graph, whose labelled nodes contain vectors of real numbers.

Hence in our view, a web domain can be formalized as follows:

Definition 2.1. Let V be the set of vertices and let $\ell : V \to \mathbb{R}^m : v \to \boldsymbol{u}_v$ be. Any directed graph $G \subset V \times V$ defined over the set of vertices enriched by the *labelling function* $\ell(G)$ is referred to as a *WEB DOMAIN*.

Definition 2.2. Let G be a web domain. Given any node v, we can construct a *LOCAL MAP* on G based on the following dynamical system Σ:

$$\begin{aligned} \boldsymbol{x}_v &= f(\boldsymbol{x}_{\mathrm{ch}(v)}, \boldsymbol{u}_v; \Theta_x), \\ \boldsymbol{y}_v &= g(\boldsymbol{x}_v; \Theta_y), \end{aligned} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{y}_v \in \mathbb{R}^o$, and $\Theta = [\Theta_x' \Theta_y']' \in \mathbb{R}^p$ is a set of (learnable) parameters. The symbol $\boldsymbol{x}_{\mathrm{ch}(v)}$ denotes an ordered list of states associated with the children of node v. The web domain G equipped with the parametrical dynamical system Σ, is referred to as a *WEB*, and is denoted by $\{G, \Sigma, \Theta\}$.

In the following, we report two noticeable examples of different forms of web computation.

2.1. Recursive neural networks

Like data, the model itself can be structured in the sense that the generic variable $x_{i,v}$ might be dependent on $q_k^{-1} x_{j,v}$, where, following the notation introduced in [10], q_k^{-1} is the operator which denotes the kth child of node v. The structure of dependence among the variables represents a form of prior knowledge. Basically, the knowledge of a recursive network yields topological constraints which often make it possible to cut the number of trainable parameters significantly. Recursive neural networks have been introduced to process a specific subclass of graphs, i.e. the Directed Ordered Acyclic Graphs (DOAGs), but can be applied to more general classes under appropriate hypotheses [1,13].

From the encoding network depicted in Fig. 1, we can see a pictorial representation of the computation taking place in the recursive neural network. Each `nil` pointer is associated with a *frontier state* $\hat{\boldsymbol{x}}_v$, which is in fact an initial state that turns out to be useful to terminate the recursive equation. The graph plays its own role in the computation both because of the information attached to its nodes and for its topology. A formal description of the computation of the input graph requires sorting the nodes, so as to define for which nodes the state can be computed first. In the literature, this problem is referred to as *topological sorting*. A sort of data flow computation takes place where the state of a given node can only be computed once all the states of its children are known. To some extent, the computation of the output $\boldsymbol{y}_v$ can be regarded as a *transduction* of the input graph $\boldsymbol{u}$ to an output graph $\boldsymbol{y}$ with the

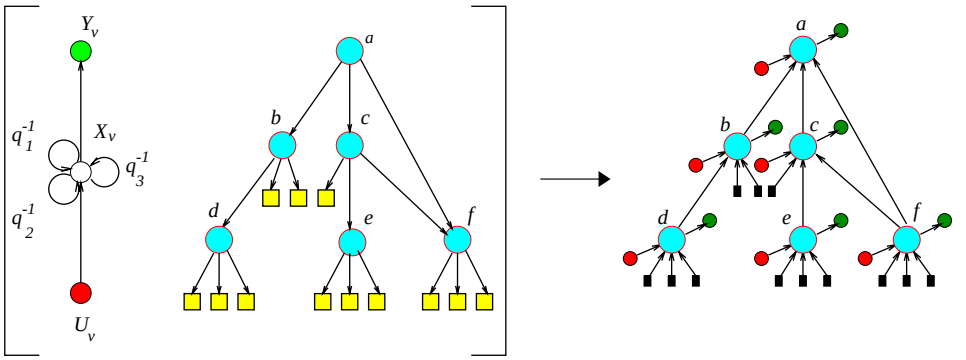

Fig. 1. Compiling the encoding network from the recursive network and the given data structure.

same skeleton [1] as $\boldsymbol{u}$. These IO-*isomorph* transductions are the direct generalization of the classic concept of transduction of lists. When processing graphs, a more general concept of transduction can be considered for the case in which also the skeleton of the graph is modified. However, many interesting task just require the application of IO-isomorph mappings. For example, the classification of DOAGs, required in many applications in pattern recognition, can be viewed as an IO-isomorph transduction. The output of the classification process corresponds with $\boldsymbol{y}_s$, that is the output value of the variables attached to the supersource in the encoding network. [2] Basically, when the focus is on classification, we disregard all the outputs $\boldsymbol{y}_v$ apart from the final value $\boldsymbol{y}_s$ of the forward computation.

The information attached to the recursive network, however, needs to be integrated with a specific choice of the functions f and g which must be suitable for learning the parameters. The connectionist assumption for the functions f and g turns out to be adequate especially to fulfill computational complexity requirements.

Let o be the maximum outdegree of the given directed graph. The dependence of node v on its children ch$[v]$ can be expressed by *pointer matrices* $\boldsymbol{A}_v(k) \in \mathbb{R}^{n,n}$, $k = 1, \ldots o$. Likewise, the information attached to the nodes can be propagated by the weight matrix $\boldsymbol{B}_v \in \mathbb{R}^{n,m}$. Hence, the first-order connectionist assumption yields

$$\boldsymbol{x}_v = \sigma\left(\sum_{k=1}^{o} \boldsymbol{A}_v(k) \cdot q_k^{-1}\boldsymbol{x}_v + \boldsymbol{B}_v \cdot \boldsymbol{u}_v\right). \tag{2}$$

Like for sequence processing the output can be computed by means of $\boldsymbol{y}_v = \sigma(\boldsymbol{C} \cdot \boldsymbol{x}_v)$.

The strong consequence of this graphical representation for f and g is that, for any input graph, an encoding neural network can be created which is itself a graph with neurons as nodes. Hence, the connectionist assumption makes it possible to go one step further the general dependence constraints expressed by means of the concept of recursive

[1] The skeleton of a graph is the structure of the data regardless of the information attached to the nodes.

[2] The supersource is a node having no incoming links. It is always possible to add a single supersource to a given DOAG.

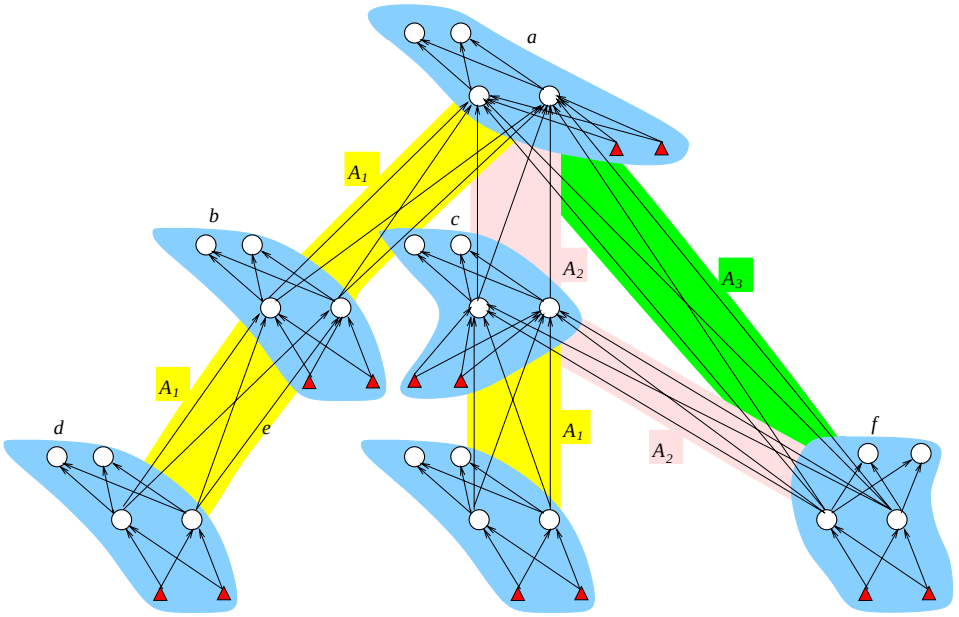

Fig. 2. The construction of a first-order recursive neural network from the encoding network. The construction holds under the assumption that the frontier states are null.

network. The corresponding encoding network turns out to be a graph whose links arise either because of the graph topology or because of the dependence between variables or because of the connectionist representation of the functions f and g themselves. The encoding networks associated with Eqs. (1) and (2) are depicted in Fig. 2 in the particular case of stationary models, in which the parameters are independent of the node v. Encoding neural networks turn out to be directed weighed graphs, that is there is always a real variable attached to the arcs (weight). Note that the architectural choice expressed by Eq. (2) can be regarded as a way to produce a multilayer-based state transition map which, however, transforms the input $\boldsymbol{u}_v$ by means of one layer only. Obviously, one could also adopt a multilayer-based architecture for implementing $f(\boldsymbol{x}_{\mathrm{ch}[v]}, \boldsymbol{u}_v, \Theta_x)$. Likewise, the function $g(\boldsymbol{x}_v, \Theta_y)$ can be implemented by a multilayer perceptron. In Fig. 2, this function is created by means of one layer of sigmoidal neurons only. Finally, in the framework of supervised learning, we can easily extend backpropagation to adjust the shared parameters of the encoding neural networks. The backpropagation takes place on neural nets which inherit the structure of the data and, therefore, the corresponding learning algorithm is referred to as *backpropagation through structure*.

2.2. Google's PageRank

The basic idea of PageRank is that of introducing a notion of page authority which is independent of the page content. Such an authority measure only emerges from the

topological structure of the Web. In PageRank, the authority reminds the notion of citation in the scientific literature. In particular, the authority of a page p depends on the number of incoming hyperlinks (number of citations) and on the authority of the page q which cites p with a forward link. Moreover, selective citations from q to p are assumed to provide more contribution to the score of p than uniform citations. Hence, the PageRank x_p of p is computed by taking into account the set of pages pa$[p]$ pointing to p. According to [6]

$$x_p = d \sum_{q \in \mathrm{pa}[p]} \frac{x_q}{h_q} + (1-d). \tag{3}$$

Here $d \in (0,1)$ is a DAMPING FACTOR and h_q is the HUBNESS[3] of q, that is the number of hyperlinks outcoming from q. When stacking all the x_p into a vector $\boldsymbol{x}$, we get

$$\boldsymbol{x} = d\boldsymbol{W}\boldsymbol{x} + (1-d)\mathbb{1}_N, \tag{4}$$

where $\mathbb{1}_N = [1,\ldots,1]'$ and $\boldsymbol{W} = \{w_{i,j}\}$—the TRANSITION MATRIX—is such that $w_{i,j} = 1/h_j$ if there is a hyperlink from j to i and $w_{i,j} = 0$, otherwise. Thus, $\boldsymbol{W}$ is a non-null matrix, where each column either sums to 1 or 0. More precisely, the jth column $\boldsymbol{w}_j$ is null if page j does not contain hyperlinks. Otherwise, $\boldsymbol{w}_j$ can be constructed by the normalization of the jth row of the Web adjacency matrix.

Notice that Eq. (3) reduces to the general web computational scheme (1).

In [6], the authors report a simple iterative algorithm based on Eq. (3). They introduce the PageRank dynamics

$$\boldsymbol{x}(t) = d\boldsymbol{W}\boldsymbol{x}(t-1) + (1-d)\mathbb{1}_N. \tag{5}$$

It can easily be proven that system is stable and that the sequence $\{\boldsymbol{x}(t)\}$ always converges to the stationary solution of the linear system (4), provided that $d < 1$. Actually, the method used by Google and defined by Eq. (5) is just the Jacobi algorithm for solving linear systems (see [12, pp. 506–509]).

Proposition 2.1. *The unique fixed point of Eq.* (5) *is*

$$\begin{aligned} \boldsymbol{x}^* &= (1-d)(I - d\boldsymbol{W})^{-1}\mathbb{1}_N \\ &= (1-d)\left(\sum_{k=0}^{\infty} d^k \boldsymbol{W}^k \cdot \mathbb{1}_N\right). \end{aligned} \tag{6}$$

Proof. Using induction on t we find that[4]

$$\boldsymbol{x}(t;d) = d^t \boldsymbol{W}^t \boldsymbol{x}_o + (1-d)\left(\sum_{k=0}^{t-1} d^{t-1-k}\boldsymbol{W}^{t-1-k}\right)\mathbb{1}_N.$$

[3] In graph theory, the hubness is also referred to as the outdegree of node q.

[4] The equations is the well-known global response of a discrete-time linear system.

Eq. (6) follows as $t \to \infty$. In fact, the first term depending on $\boldsymbol{x}_o$ vanishes because of the damping factor d and the properties of the matrix $\boldsymbol{W}$. □

A slightly different approach to the computation of PageRank was proposed in [5,7]. In that case, the following equation

$$\boldsymbol{x}(t) = d\boldsymbol{W}\boldsymbol{x}(t-1) + \frac{\alpha(t-1)}{N}\mathbb{1}_N \tag{7}$$

is assumed, where, for each t, $\alpha(t-1) = \|\boldsymbol{x}(t-1)\| - \|d\boldsymbol{W}\boldsymbol{x}(t-1)\|$ in order to force the condition $\|x(t)\|_1 = 1$. System (7) produces a normalized version of PageRank, and converges to $\boldsymbol{x}^*/\|\boldsymbol{x}^*\|_1$, where $\boldsymbol{x}^*$ is the solution of Eq. (4).

Some of the elementary properties of matrix $\boldsymbol{W}$ will be particularly useful in order to discuss our results. First of all, notice that $\boldsymbol{W}$ is a stochastic matrix except for the null rows.[5] The pages that do not contain hyperlinks are called *sinks* and will have a special role in the following discussion. In fact, the presence of sinks prevents the direct application of the results from the theory of stochastic matrices (see [15]). A simple trick to eliminate sinks consists of introducing a dummy page which has a link to itself and is pointed by every sink page. Thus, if P is the set of vertices and $H \subset P \times P$ is the set of arcs in the original graph, the extended graph turns out to be $\overline{\boldsymbol{G}}_W = (\overline{P}, \overline{H})$, where $\overline{P} = P \cup \{N+1\}$ and $\overline{H} = H \cup \{(i, N+1) | \nexists j, (i,j) \in H\}$. The transition matrix $\overline{\boldsymbol{W}}$ that corresponds to $\overline{\boldsymbol{G}}_W$ is

$$\overline{\boldsymbol{W}} = \begin{pmatrix} \boldsymbol{W} & \boldsymbol{0} \\ \boldsymbol{R} & 1 \end{pmatrix},$$

where $\boldsymbol{R} = [r_1, \ldots, r_N]$, and if i is a sink then $r_i = 1$, else $r_i = 0$. After such a transformation, any sink is removed (see Fig. 3) and $\overline{\boldsymbol{W}}$ is a stochastic matrix. For instance,

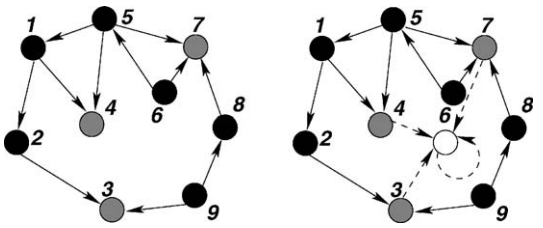

Fig. 3. A trick to eliminate sinks: a dummy node with a self-loop is added.

[5] Stochastic matrices are non-negative matrices having all columns that sum up to 1.

referring to Fig. 3, the transition matrix $\overline{\boldsymbol{W}}$ is

$$\overline{\boldsymbol{W}} = \left[\begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 \\ 1/2 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 1/2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right].$$

3. Topological spectra

The computation defined by Eq. (1) makes it possible to extract interesting topological features, which is commonly regarded as a difficult problem in neural computation.

Definition 3.1. Let us consider a special web domain composed of a collection of graphs defined by matrices $\boldsymbol{W} \in \mathscr{W}$ and let $\boldsymbol{P} \in \mathbb{R}^{N,N}$ be a permutation matrix. Any function

$$\mathscr{T}_s : \mathscr{W} \to \mathbb{R}^N : \boldsymbol{W} \to \mathscr{T}_s(\boldsymbol{W}) = \mathscr{T}_s(\boldsymbol{P}'\boldsymbol{W}\boldsymbol{P}) \tag{8}$$

is referred to as a *TOPOLOGICAL SPECTRUM* for G.

Basically, a topological spectrum is a way to compute properties of a given graph, represented by matrix $\boldsymbol{W}$, which are invariant under permutations of nodes.

Definition 3.2. Given any graph G, the function

$$\mathscr{T}_s(G; n) : [0, 1) \to \mathbb{R}^N : d \to \boldsymbol{x}(n; d) \tag{9}$$

is referred to as the *NODE TOPOLOGICAL SPECTRUM* of degree n for graph G (briefly, node-ranking spectrum). As $n \to \infty$ $\boldsymbol{x}(n; d) \to \boldsymbol{x}^*(d)$.

The definition holds for directed graphs, but it can be extended to undirected graphs by assuming that any undirected arc is replaced by the two directed arcs (u, v) and (v, u).

Remark 3.1. Relations with the notion of *GRAPH SPECTRUM*.

Note that the spectrum of a graph is related to the given definition of topological spectrum. The first notion involves the eigenvalues of the adjacency matrix of the graph. If we replace the adjacency matrix with $\boldsymbol{W}$ the determination of the spectrum

consists of finding the roots z of

$$\det(z\boldsymbol{I} - \boldsymbol{W}) = 0.$$

In order to gain coherence with Definition 3.1, the notion of topological spectrum is obtained by associating a graph with the set of eigenvectors of matrix $\boldsymbol{W}$. We can promptly see that Definition 3.1 holds because of the well-known invariance property of the eigenvectors.

Remark 3.2. Let $\boldsymbol{\ell}_p \in \mathbb{R}^N$ such $\boldsymbol{\ell}_p(q) = 1$ iff $p = q$. Then

$$\begin{aligned}\mathscr{T}_{\mathrm{s}}^{p}(G,d) &= \boldsymbol{\ell}_p \boldsymbol{x}^*(d) \\ &= (1-d)\sum_{k=0}^{\infty}(\boldsymbol{\ell}_p \boldsymbol{W}^k \mathbb{1}_N)d^k \\ &= (1-z^{-1})\sum_{k=0}^{\infty}\gamma_k^p \cdot z^{-k}, \end{aligned} \tag{10}$$

where $\gamma_k^p \doteq \boldsymbol{\ell}_p \boldsymbol{W}^k \mathbb{1}_N$ and $z \doteq d^{-1}$. Interestingly, the topological spectrum restricted to node p, turns out to be directly related to the *Z-transform* of function γ_k^p. For this reason, in the following the damping factor d will be replaced by z^{-1}.

Thus, if we rewrite $d = z^{-1}$ and we consider the node topological spectrum $\boldsymbol{x}^*(d)$ ordering its components such that

$$x_{i(h)}(z^{-1}) \geqslant x_{j(k)}(z^{-1})$$

iff $i(h) < j(h)$, we obtain the damping transform $\boldsymbol{x}(z^{-1})$. This vector of features is topologically invariant as stated in the following proposition.

Proposition 3.1 (TOPOLOGICAL INVARIANCE). *The damping transform $\boldsymbol{x}(z^{-1})$ is topologically invariant, that is for any given graph G, $\boldsymbol{x}(z^{-1})$ is independent of the permutation chosen for the labelling of the nodes.*

Proof. Let $\boldsymbol{P}$ be a permutation matrix which expresses the change of node labelling. Then the matrix $\boldsymbol{W}$ is transformed by

$$\hat{\boldsymbol{W}} = \boldsymbol{P}'\boldsymbol{W}\boldsymbol{P}.$$

The node topological spectrum associated to $\hat{\boldsymbol{W}}$ becomes

$$\begin{aligned}\hat{\boldsymbol{x}}(z^{-1}) &= (1-z^{-1})(\boldsymbol{I} - z^{-1}\hat{\boldsymbol{W}})^{-1}\mathbb{1}_N \\ &= (1-z^{-1})\sum_{k=0}^{\infty}(\boldsymbol{P}'\boldsymbol{W}\boldsymbol{P})^k z^{-k}\mathbb{1}_N \\ &= \boldsymbol{P}'\left((1-z^{-1})\sum_{k=0}^{\infty}\boldsymbol{W}^k z^{-k}\boldsymbol{P}\mathbb{1}_N\right)\end{aligned}$$

$$= \boldsymbol{P}' \left((1 - z^{-1}) \sum_{k=0}^{\infty} \boldsymbol{W}^k z^{-k} \mathbb{1}_N \right)$$
$$= \boldsymbol{P}' \boldsymbol{x}(z^{-1}).$$

Thus the two topological spectra contain the same values yielding the same ordered vectors corresponding to their damping transforms. □

The same property of invariance holds for $\boldsymbol{x}(n, z^{-1})$. The proof follows easily by induction on n.

Proposition 3.2 (TOPOLOGICAL SPECTRUM FOR *DAGs*). *The topological spectrum of a DAG with depth m is a polynomial of finite degree*

$$x_p = (1 - d) \sum_{k=0}^{m-1} \gamma_k^p d^k. \tag{11}$$

Proof. The proof is given by induction on p. We assume to label all the nodes of the graph in such a way to reflect the partial ordering induced by the DAG.

Basis: $p = 1$, trivial.

Induction step: Let pa$[p]$ be the set of parents of node p. For all $q \in \text{pa}[p]$

$$x_q = (1 - d) \sum_{h=0}^{m-2} \omega_q(h) d^h.$$

Because of Eq. (3)

$$x_p = 1 - d + d \sum_{q \in \text{pa}[p]} \gamma_q x_q$$
$$= 1 - d + (1 - d) d \sum_{q \in \text{pa}[p]} \sum_{h=0}^{m-2} \gamma_q \omega_q(h) d^h$$
$$= (1 - d) \left(1 + \sum_{h=1}^{m-1} \left(\sum_{q \in \text{pa}[p]} \gamma_q \omega_q(h) \right) d^h \right)$$
$$= (1 - d) \sum_{h=0}^{m-1} \omega_p(h) d^h,$$

where

$$\omega_p(0) \doteq 1,$$
$$\omega_p(h) \doteq \sum_{q \in \text{pa}[p]} \gamma_q \omega_q(h).$$

Thus, $x_p(d)$ is a polynomial of degree $m - 1$. □

We can define a matrix-based representation of the topological spectrum by considering a sampling for the parameter $z = d^{-1}$.

Definition 3.3 (*Discrete topological spectrum*). The collection of samples of $\{\mathscr{T}_s(G,z_k^{-1})\}_{k\in\mathscr{S}}$ with $\mathscr{S}\doteq[1,\ldots,N]$ is referred to as the *discrete topological spectrum* of G over $\mathscr{Z}=\{z_1^{-1},\ldots,z_N^{-1}\}$.

The discrete topological spectrum $\{\mathscr{T}_s(G,z_k^{-1})\}_{k\in\mathscr{S}}$ can be kept in the square matrix

$$\boldsymbol{X}(z^{-1})\doteq[\boldsymbol{x}(z_1^{-1})|\cdots|\boldsymbol{x}(z_N^{-1})]$$

In particular, we can define a reference sampling to define the columns of $\boldsymbol{X}(z^{-1})$.

Definition 3.4. A *monotonic sampling* of $\mathscr{T}_s(G,z^{-1})$ is any sampling $\mathscr{Z}=\{z_1^{-1},\ldots,z_N^{-1}\}$ such that if $k>h$ then $z_k<z_h$.

The matrix-based representation $\boldsymbol{X}(z^{-1})$ of the topological spectrum is strictly related to the adjacency matrix of the corresponding graph. Each row of matrix $\boldsymbol{X}(z^{-1})$ is in fact related to the corresponding node in the graph. As a result, the way the nodes are numbered affects matrix $\boldsymbol{X}(z^{-1})$. We can remove this dependence by defining a way to order the matrix rows as done when we introduced the damping transform. Let us consider a monotonic sampling and let z_1^{-1} be the reference damping factor. If we reorder the rows such that

$$x_{i(h)}(z_1^{-1})\geqslant x_{j(k)}(z_1^{-1})$$

iff $i(h)<j(h)$, the first column of matrix $\boldsymbol{X}(z^{-1})$ has a well-defined structure. In case there are sets of equal entries in the first column we can order the corresponding rows by considering the values on the other columns. Basically, we obtain a permutation of nodes which yields a sorted list of node rankings with respect to damping factors using z_1^{-1} as a first reference. The original spectral matrix $\boldsymbol{X}(z^{-1})$ is then mapped to the sorted spectral matrix $\boldsymbol{X}_s(z^{-1})$ which directly incorporates some interesting topological properties. The following proposition shows that, unfortunately, the sorting is affected by the choice of the damping factor z_k^{-1}.

When d is small only the nearest relatives give an actual contribution to x_p, whereas, as $d\to 1$, far ancestors are also significant for the evaluation of x_p.

Example 3.1. Let p_1 and p_2 be connected to the graph as in Fig. 4(a). Suppose that p_1 has more ancestors than p_2, which are far from p_1, whereas p_2 has less but closer relatives. When $d\approx 1$, x_{p_1} will be larger than x_{p_2}, but when $d\approx 0$ the converse holds. In particular,

$$x_{p_1}=(1-d)+d(1-d)+4d^2(1-d),$$

$$x_{p_2}=(1-d)+2d(1-d),$$

and $x_{p_1}>x_{p_2}$ if $d>\frac{1}{4}$.

Remark 3.3. Let us consider trees t_p and t_q which, by construction, have p and q as their root and are constructed by following back the arcs leading to p and q,

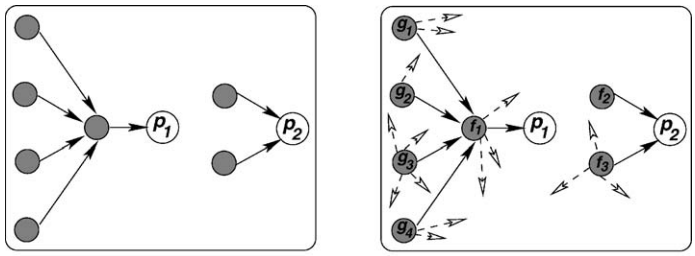

Fig. 4. (a) An example of how d can influence the ordering of the nodes. Node p_1 has many far relatives, whereas p_2 has a few near ancestors. (b) The effect of the hubness in the PageRank extinguishes the effect of d.

respectively. Let the hubness be constant for each node. If t_p and t_q have the same number of nodes in levels $0, 1, \ldots, i-1$, but t_p has fewer nodes than t_q at level i and a greater number of nodes at level $i+1$, then $x_p < x_q$ as d approaches 0. When the hypothesis on the hubness is relaxed, the score of each node is deeply influenced by the importance of the arcs pointing to it.

Example 3.2. Let us consider again the scores of p_1 and p_2 with the connectivity described in Fig. 4(a), but taking into account parent nodes with distinct hubness for both nodes (see Fig. 4(b)). In this case $x_{p_2} \geqslant x_{p_1} \forall d \in [0,1)$. In particular, we can notice that nodes f_1 and f_3 give the same contribution to the PageRank of p_1 and p_2, respectively. Nevertheless, when $d \approx 1$, the PageRanks due to nodes g_i, $i = 1, \ldots, 4$, sum up to approximately $\frac{1}{3}$, which is far away from the unitary score given by f_2.

4. Inverse topological spectrum

Now suppose you are given the topological spectrum $\mathscr{T}_s(G, z^{-1})$ and you want to reconstruct the corresponding graph G. In particular, we will consider the graph discrete topological spectrum of Definition 3.3, represented by the $N \times N$ matrix $\boldsymbol{X}(z^{-1})$. However, as it will be shown, the reconstruction cannot always be done univocally.

Definition 4.1 (*Discrete differential spectrum*). We introduce the definition of *Discrete differential spectrum* as follows:

$$z^{-1}\boldsymbol{X}(z^{-1}) \doteq [z_1^{-1} \cdot \boldsymbol{x}(z_1^{-1}) | \cdots | z_N^{-1} \cdot \boldsymbol{x}(z_N^{-1})].$$

Let us introduce the notation $(1 - z^{-1})\mathbb{1}_N \doteq [1 - z_1^{-1}, \ldots, 1 - z_N^{-1}]'$.

Proposition 4.1 (discrete spectrum inversion). *Let $\boldsymbol{X}(z^{-1})$ be the discrete topological transform of a given graph G for a given sampling $\mathscr{X}$. If rank $\boldsymbol{X}(z^{-1}) = N$ then the*

graph G can be univocally reconstructed and its matrix $\boldsymbol{W}$ *is*

$$\boldsymbol{W} = [\boldsymbol{X}(z^{-1}) - (1 - z^{-1})\mathbb{1}_N] \cdot [z^{-1}\boldsymbol{X}(z^{-1})]^{-1}. \tag{12}$$

Proof. From the definition of $\boldsymbol{X}(z^{-1})$

$$\boldsymbol{X}(z^{-1}) = z^{-1}\boldsymbol{W}\boldsymbol{X}(z^{-1}) + (1 - z^{-1})\mathbb{1}_N.$$

Since $rank\, \boldsymbol{X}(z^{-1}) = N$ then $\boldsymbol{X}(z^{-1})$ is invertible and, consequently, $z^{-1}\boldsymbol{X}(z^{-1})$ is invertible, too. Hence, the thesis follows straightforwardly from the previous equation solving it with respect to $\boldsymbol{W}$. □

The rank of $\boldsymbol{X}(z^{-1})$ is related to the rank of the reachability matrix of the pair $\{\boldsymbol{W}, \mathbb{1}\}$ defined as

$$\boldsymbol{R} \doteq [\mathbb{1}_N \quad \boldsymbol{W}\mathbb{1}_N \quad \boldsymbol{W}^2\mathbb{1}_N \cdots \boldsymbol{W}^{N-1}\mathbb{1}_N].$$

Proposition 4.2 (Reachability matrix and discrete topological spectrum). *If* $rank\, \boldsymbol{R} = r$ *then* $rank\, \boldsymbol{X}(z^{-1}) \leqslant r$ *for any sampling* $\mathscr{Z}$.

Proof. The thesis derives straightforwardly from a well-known result in linear system theory concerning state reachability (see [14, Theorem 5, p. 272]) which is based on the Cayley–Hamilton theorem. In particular, from Eq. (6) it derives that $\boldsymbol{x}^*(z^{-1}) \in \mathrm{span}(\boldsymbol{R})$ for any value of z^{-1} and thus

$$\boldsymbol{X}(z^{-1}) = \boldsymbol{R}\boldsymbol{V}(\mathscr{Z}),$$

where $\boldsymbol{V}(\mathscr{Z})$ is a $N \times N$ matrix. The thesis is obtained directly from the previous factorization. □

Remark 4.1. The proposition motivates the definition of the discrete topological spectrum since it indicates how to choose the sampling $\mathscr{Z}$. In particular, it is clear that the sampling can be limited to N values of z^{-1}.

Finally, when the condition on the rank of the discrete topological spectrum holds, this representation can be used to verify the isomorphism of two graphs.

Theorem 4.1 (graph isomorphism). *Given any two graphs* G_1 *and* G_2, *let* $\boldsymbol{X}_1(z^{-1})$ *and* $\boldsymbol{X}_2(z^{-1})$ *be their spectra and assume that*
1. *there exists a permutation matrix* $\boldsymbol{P}$ *such that* $\boldsymbol{X}_2(z^{-1}) = \boldsymbol{P} \cdot \boldsymbol{X}_1(z^{-1})$
2. *matrix* $\boldsymbol{X}_2(z^{-1})$ *is full rank*
then $G_1 \equiv G_2$ (G_1 *is isomorph to* G_2).

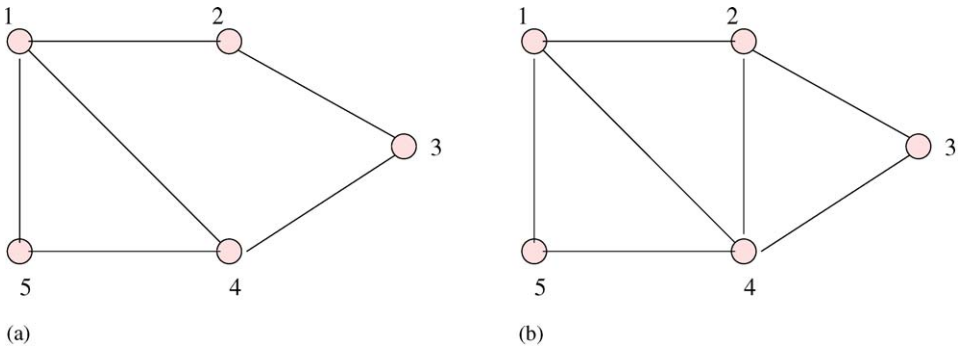

Fig. 5. (a) Graph in which *rank* $\boldsymbol{R}=5$ (b) Graph in which *rank* $\boldsymbol{R}=3$. In this case the isomorphism cannot be based straightforwardly on the computation of the spectrum.

Proof. Since $\boldsymbol{X}_2(z^{-1})$ is full rank we can apply Eq. (12) and by substituting $\boldsymbol{X}_2(z^{-1})$ with $\boldsymbol{P}\cdot\boldsymbol{X}_1(z^{-1})$ we obtain

$$\begin{aligned}\boldsymbol{W}_2 &= [\boldsymbol{P}\cdot\boldsymbol{X}_1(z^{-1}) - (1-z^{-1})\boldsymbol{P}\cdot\mathbb{1}_N\cdot\boldsymbol{P}']\cdot[z^{-1}\boldsymbol{P}\cdot\boldsymbol{X}_1(z^{-1})]^{-1}\\ &= \boldsymbol{P}\cdot[\boldsymbol{X}_1(z^{-1}) - (1-z^{-1})\mathbb{1}_N]\cdot[z^{-1}\boldsymbol{X}_1(z^{-1})]^{-1}\cdot\boldsymbol{P}'\\ &= \boldsymbol{P}\cdot\boldsymbol{W}_1\cdot\boldsymbol{P}',\end{aligned}$$

where we used the fact that $\boldsymbol{P}$ is an orthonormal matrix and $\boldsymbol{X}_1(z^{-1})$ is full rank, too. Thus, it yields that $G_1\equiv G_2$. □

Remark 4.2. Interestingly, the conditions of the theorem are "frequently met" in random graphs. Note that the straightforward application of the theorem yields a polynomial algorithm for the graph isomorphism problem within the class of graph defined by the hypotheses. As shown in the following example, however, there are cases in which the hypotheses do not hold.

Example 4.1. Let us consider the example with the two graphs represented in Fig. 5. For graph (a) $\operatorname{rank}\boldsymbol{X}(z^{-1})\leqslant\operatorname{rank}\boldsymbol{R}=5$ and, therefore the graph may be reconstructed directly from Eq. (12). On the opposite, for graph (b), Eq. (12) does not allow one a straightforward unique reconstruction since $\operatorname{rank}\boldsymbol{X}(z^{-1})\leqslant\operatorname{rank}\boldsymbol{R}=3$.

5. Learning in web domains

A given web exhibits a dynamics that is typically dependent on a set of parameters Θ, which makes it suitable for learning. Similar to artificial neural networks defined on traditional learning environments, one can construct learning theories which basically consist of adapting the parameters Θ. Both the supervised and unsupervised learning

protocols can be conceived which, however, must take into account topological issues. Interestingly, the concepts to be discovered by learning from examples are now at node level and the nodes are inherently embedded into the web domain. For a concept to be meaningful, it has to be somehow related to the content and to the topological properties of the node. In this paper, we restrict the attention to supervised learning, where a teacher provides a real number to be attached to each node, so as to generate the LEARNING ENVIRONMENT $\{(v, t(v)),\ v \in V_L \subset V\}$. Consequently, the degree of fitting of the learning environment is evaluated by minimizing the COST FUNCTION, generally subjected to constraints

$$\min_{\Theta} \sum_{v \in V_L} \delta(t(v) - y(v; \Theta)), \tag{13}$$

$$\phi(\boldsymbol{y}_L(\Theta)) = \mathbf{0}, \tag{14}$$

where $\delta(\cdot)$ is a metrics on $\mathbb{R}$ and $\phi(\cdot)$ expresses a set of constraints on the values $y(v; \Theta)$. This problem arises when one wants to provide a score to web pages on the basis of indications and constraints acquired by human experience, but there are plenty of applications in different fields and, especially, in pattern recognition.

It is worth mentioning that in most interesting real-world problems, $|V_L| \ll |V|$, since there are regularities amongst nodes on both content and topology. Basically, reasonable concepts to be learned require that one can infer properties on other nodes and, therefore, problems of overfitting suggest the adoption of an appropriate number of parameters.

Here we revisit the two classes of models considered in the previous section to give some preliminary insights on learning.

- *Recursive neural networks*: In recursive neural networks the learning of the parameters Θ is made possible by the hypothesis of dealing with directed ordered graphs. Similar to the case of sequences, we can unfold the network along the structure and use BACKPROPAGATION THROUGH STRUCTURE [10]. The minimization of $\sum_{v \in V_L} \delta(t(v) - y(v; \Theta))$ assumes an appropriate sharing of parameters Θ which allows very good generalization to new examples in many interesting real-world problems [9]. Notice that the connectionist unfolding, which produces feedforward networks in the case of directed acyclic graphs, gives rise to neural networks with cycles requiring a relaxation to an equilibrium point in the general case depicted in Fig. 6.
- *Learning the page rank on the web*: An example of learning in web domains has been recently proposed in [16] in which, apart from the web pages whose PageRanks need to be modified, for the rest of the pages, we wish to minimize the modification of their PageRank. Unlike the case of recursive neural networks, in which the learning parameters are associated with the arcs, the chosen learning parameter Θ is associated with each node. Hence, the learning scheme is based on $\boldsymbol{x}_a = (1-d)(\boldsymbol{I} - d\boldsymbol{W})^{-1}\boldsymbol{e} \doteq (1-d)\boldsymbol{M}\cdot\boldsymbol{e}$, where $\boldsymbol{e}$ is a $|V|$-dimensional real vector, acting as a learning parameter. Another approach to learn the PageRank has been recently proposed in [8].

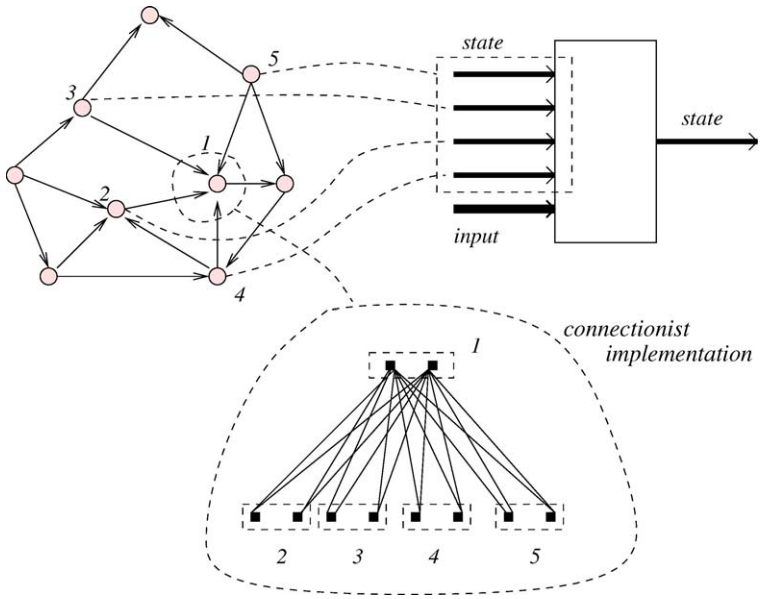

Fig. 6. The local computation hypothesis in web domains and the connectionist implementation of function f.

6. Conclusions

One of the main motivations for proposing the reformulation of learning in web domains is that data often exhibit relationships, which are typically neglected. In particular, topological features, which are efficiently expressed by spectral analysis and random walk models, can hardly be learned by most traditional connectionist models.

In this paper, we have introduced a new general framework for learning which is based on the concept of web. Beginning from present existing limitations of traditional learning schemes, we have stressed the importance of providing structured representation of the data. This is claimed by many people and is becoming the subject of studies which emphasize learning so as to exhibit robustness to noise [11]. The introduction of webs represents an extension of machine learning approaches based on collections of graphs, in that they represent a domain where the function to be learned is defined.

Interestingly, the neural computation and the social networks used to rank Web pages, that are unified in this paper, share principles that can be nicely exploited to face classic problems like graph isomorphism. In particular, a novel spectral analysis of graphs is proposed which gives rise to a polynomial algorithm for a class of graphs that does not trivially appear as a special case of those known in the literature.

Acknowledgements

We thank Antonio Pasini (University of Siena), Ah Chung Tsoi (Faculty of Computer Science, University of Wollongong), Prabhakar Raghavan (Verity Inc.) for

fruitful discussions which contributed to shed lights on some topics covered in this paper.

References

[1] M. Bianchini, M. Gori, F. Scarselli, Processing directed acyclic graphs with recursive neural networks, IEEE Trans. Neural Networks 12 (6) (2001) 1464–1470.
[2] I. Bomze, W. Gutjahr, The dynamics of self-evaluation, Appl. Math. Comput. 64 (1994) 47–63.
[3] I. Bomze, W. Gutjahr, Estimating qualifications in a self-evaluating group, Quality Quantity 29 (1995) 241–250.
[4] N. Bose, Applied Multidimensional Systems Theory, Van Nostrand Reinhold Co., NY, 1982.
[5] S. Brin, R. Motwani, L. Page, T. Winograd, What can you do with a web in your pocket? IEEE Bull. Tech. Committe on Data Eng. IEEE Comput. Soc. 21 (2) (1998) 37–47.
[6] S. Brin, L. Page, The anatomy of a large-scale hypertextual Web search engine, in: Proc. 7th World Wide Web Conf. (WWW7), April 14–18, 1998, pp. 107–117.
[7] S. Brin, L. Page, R. Motwani, T. Winograd, The PageRank citation ranking: bringing order to the Web, Tech. Report 1999–1966, Stanford University, 1999; Available on the Internet at http://dbpubs.stanford.edu:8090/pub/1999-66.
[8] M. Diligenti, M. Gori, M. Maggini, A learning algorithm for web page scoring systems, in: Proc. 18th Internat. Joint Conf. on Artificial Intelligence (IJCAI 2003), Acapulco, Mexico, August 9–15, 2003.
[9] P. Frasconi, C. Goller, M. Gori, A. Kuchler, A. Sperduti, From sequences to data structures: theory and applications, in: J. Kolen, S. Kremer (Eds.), A Field Guide to Recurrent Neural Networks, IEEE Press, New York, 2001, pp. 351–374.
[10] P. Frasconi, M. Gori, A. Sperduti, A general framework for adaptive processing of data structures, IEEE Trans. Neural Networks 9 (5) (1998) 768–786.
[11] P. Frasconi, M. Gori, A. Sperduti, Guest editors' introduction: special section on connectionist models for learning in structured domains, IEEE Trans. Knowledge Data Eng. 13 (3) (2001) 145–147.
[12] G.H. Golub, C.F. Van Loan, Matrix Computation, The Johns Hopkins University Press, Baltimore, MD, 1993.
[13] M. Gori, M. Maggini, L. Sarti, A recursive neural network model for processing directed acyclic graphs with labeled edges, in: Proc. Internat. Joint Conf. on Neural Networks (IJCNN), vol. 2, Portland, Oregon, July 20–24, 2003, pp. 1351–1355.
[14] L. Padulo, M. Arbib, System Theory—A Unified Approach to Continuous and Discrete Systems, W.B. Saunders Company, Philadelphia, London, Toronto, 1974.
[15] E. Seneta, Non-negative Matrices and Markov Chains, Springer, Berlin, 1981, pp. 112–158 (Chapter 4).
[16] A.C. Tsoi, G. Morini, F. Scarselli, M. Hagenbuchner, M. Maggini, Adaptive ranking of web pages, in: Proc. 12th World Wide Web Conf. (WWW12), May 20–24, 2003, pp. 356–365.