COSNet: a Cost Sensitive Neural Network for Semi-supervised Learning in Graphs

Alberto Bertoni, Marco Frasca, and Giorgio Valentini

DSI, Dipartimento di Scienze dell' Informazione,
Università degli Studi di Milano,
Via Comelico 39, 20135 Milano, Italia.
{bertoni,frasca,valentini}@dsi.unimi.it

Abstract. The semi-supervised problem of learning node labels in graphs consists, given a partial graph labeling, in inferring the unknown labels of the unlabeled vertices. Several machine learning algorithms have been proposed for solving this problem, including Hopfield networks and label propagation methods; however, some issues have been only partially considered, e.g. the preservation of the prior knowledge and the unbalance between positive and negative labels. To address these items, we propose a Hopfield-based cost sensitive neural network algorithm (*COSNet*). The method factorizes the solution of the problem in two parts: 1) the subnetwork composed by the labelled vertices is considered, and the network parameters are estimated through a supervised algorithm; 2) the estimated parameters are extended to the subnetwork composed of the unlabeled vertices, and the attractor reached by the dynamics of this subnetwork allows to predict the labeling of the unlabeled vertices. The proposed method embeds in the neural algorithm the "a priori" knowledge coded in the labelled part of the graph, and separates node labels and neuron states, allowing to differentially weight positive and negative node labels. Moreover, *COSNet* introduces an efficient cost-sensitive strategy which allows to learn the near-optimal parameters of the network in order to take into account the unbalance between positive and negative node labels. Finally, the dynamics of the network is restricted to its unlabeled part, preserving the minimization of the overall objective function and significantly reducing the time complexity of the learning algorithm. *COSNet* has been applied to the genome-wide prediction of gene function in a model organism. The results, compared with those obtained by other semi-supervised label propagation algorithms and supervised machine learning methods, show the effectiveness of the proposed approach.

1 Introduction

The growing interest of the scientific community in methods and algorithms for learning network-structured data is motivated by emerging applications in several domains, ranging from social to economic and biological sciences [1, 2]. In this context a fundamental problem is represented by the supervised or semi-supervised node classification, i.e. predicting node labels by exploiting the relationships between labeled and unlabeled nodes of the network. Instances are connected via a set of links, and a learner relies on the assumption that linked entities tend to be assigned to the same class label. For example, in protein-protein interaction networks genetic or physical interactions coded in the links of the network bear witness to common biological processes or molecular function activities between linked proteins [3]; in social networks, people that are friends often share similar characteristics or commons interests [4]; in document classification, texts that are linked through common citations often share similar topics [5].

Several approaches have been proposed in literature to classify networked data. They usually represent data through an undirected graph $G = (V, W)$, where nodes $v \in V$ correspond to instances to be classified, and W defines the weights of the edges according to the "strength" or the evidence of the relationships between pairs of nodes.

The first and simplest algorithms proposed were based on "guilt-by-association" methods, by which unlabeled nodes are set according to the majority or the weighted majority of the labels in their neighborhoods [6, 7]. By extending this approach, nodes can "propagate" their labels to their neighbors iteratively by repeating this "label propagation" process until convergence [8, 9]. In this context Markov Random Walks can be applied to tune the amount of propagation we allow in the graph, by setting the length of the walk across the graph [10, 11]. Other related methods are based on smoothness considerations that yields to graph regularization [12, 13], or exploit the properties of the graph Laplacian associated to the weight matrix of the network [14]. Algorithms based on the evaluation of the functional flow in graphs [3, 15], on Markov [16] and Gaussian Random Fields [17, 18] have been applied to the prediction of gene functions in biological networks. Hopfield networks [19] shares common elements with label propagation algorithms. Indeed labels are iteratively propagated across the neighbors of each node and a quadratic cost function related to the consistency of the labeling of the nodes w.r.t. the network topology is minimized by the network dynamics. From this standpoint Hopfield networks and most of the proposed graph-based algorithms for the prediction of node labels can be cast into a recently proposed common framework where a quadratic cost objective function is minimized [20]. Nevertheless, there are some issues that have been only partially considered in classifying networked data. Many of the graph-based approaches do not preserve prior information coded in nodes labeling, and are unable to effectively predict node labels when data are unbalanced, e.g. when negative nodes significantly outnumber positives. This issue is particularly relevant when label propagation algorithms are applied to predict the functions of genes, since positive annotations are usually much less than negative ones [18]. Despite some cost-sensitive variants of Gaussian Random fields have been proposed, they are based on simple class rescaling so that their respective weights over unlabeled examples match the prior class distribution estimated from labeled examples [8, 18]. Finally, many approaches based on neural networks do not distinguish between the node labels and the values of the neuron states [21], thus resulting in a lower predictive capability of the network.

To address these issues, we propose a cost-sensitive neural algorithm (*COS-Net*), based on Hopfield networks, whose main characteristics are the following:

1. Available a priori information is embedded in the neural network and preserved by the network dynamics.
2. Labels and neuron states are conceptually separated. In this way a class of Hopfield networks is introduced, having as parameters the values of neuron states and the neuron thresholds.
3. The parameters of the network are learned from the data through an efficient supervised algorithm, in order to take into account the unbalance between positive and negative node labels.
4. The dynamics of the network is restricted to its unlabeled part, preserving the minimization of the overall objective function and significantly reducing the time complexity of the learning algorithm.

In sect. 2 the classification of nodes in networked data is formalized as a semi-supervised learning problem. Hopfield networks and the main issues related to this type of recurrent neural network are discussed in Sect. 3 and 4. The problem of the restriction of network dynamics to a subset of nodes is analyzed in Sect 5, while the proposed neural network algorithm, COSNet (*COst Sensitive neural Network*), is discussed in Sect. 6. In the same section we show that *COSNet* covers the main Hopfield networks learning issues, and in particular a statistical analysis highlights that the network parameters selected by *COSNet* lead to significantly lower values of the the energy function w.r.t. the non cost-sensitive version of the Hopfield network. In Section 7, to test the proposed algorithm on a classical unbalanced semi-supervised classification problem, we applied *COS-Net* to the genome-wide prediction of gene functions in a model organism, by considering about 200 different functional classes of the FunCat taxonomy [22], and five different types of biomolecular data. The conclusions end the paper.

2 Semi-supervised Learning in Graphs

Consider a weighted graph $G = (V, W)$, where $V = \{1, \dots, n\}$ is the vertex set and $W = (w_{ij})$ is the symmetric weight matrix: the weight $w_{ij} \in \mathbb{R}$ denotes a similarity index of node i with respect to node j. The vertices in V are labeled with $\{+, -\}$, leading to the subsets P and N of positive and negative vertices, but the labeling is known only for a subset $S \subset V$, while is unknown for $U = V \setminus S$. Let be $S^+ = S \cap P$ and $S^- = S \cap N$: we can refer to S^+, S^- and W as the "prior information".

The semi-supervised classification problem consists in finding a bipartition (U^+, U^-) of nodes in U relying on the prior information. Nodes in U^+ are then considered candidates for the class $P \cap U$.

A reasonable measure of the "correctness" in approximating $P \cap U$ by U^+ and $N \cap U$ by U^-, especially when unbalanced data are considered, is the F_{score}, defined as follows: by calling *false positives* the vertices $FP = U^+ \cap N$, *false negatives* $FN = U^- \cap P$ and *true positives* $TP = U^+ \cap P$, the F_{score} is the harmonic mean between *precision* and *recall*, where $precision = \frac{|TP|}{|TP|+|FP|}$, $recall = \frac{|TP|}{|TP|+|FN|}$. Note that $0 \leqslant F_{score} \leqslant 1$ and $F_{score} = 1$ iff $U^+ = P \cap U$.

3 Hopfield Networks

By slightly generalizing the classical definition of discrete Hopfield networks (DHNs) [19], a Hopfield network H with neurons $V = \{1, 2, \ldots, n\}$ can be described by a triple $H = \, < W, \gamma, \alpha >$, where:

- W is a $n \times n$ symmetric matrix in which $w_{ij} \in \mathbb{R}$ is the connection strength between neurons i and j, with $w_{ii} = 0$ for each i
- $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_n) \in \mathbb{R}^n$ is a vector of activation thresholds
- α is a real number in $[0, \frac{\pi}{2}]$ that determines the two different values $\{\sin\alpha, -\cos\alpha\}$ for neuron states.

At each discrete time t each neuron i has a value $x_i(t) \in \{\sin\alpha, -\cos\alpha\}$ according to the following dynamics:

1. At time 0 an initial value $x_i(0) = a_i$ is given for each neuron i
2. At time $t+1$ each neuron is updated asynchronously in a random order by the following activation rule

$$x_i(t+1) = \begin{cases} \sin\alpha & \text{if } \sum\limits_{j=1}^{i-1} w_{ij} x_j(t+1) + \sum\limits_{k=i+1}^{n} w_{ik} x_k(t) - \gamma_i > 0 \\ -\cos\alpha & \text{if } \sum\limits_{j=1}^{i-1} w_{ij} x_j(t+1) + \sum\limits_{k=i+1}^{n} w_{ik} x_k(t) - \gamma_i \leq 0 \end{cases} \tag{1}$$

The state of the network at time t is the vector $x(t) = (x_1(t), x_2(t), \ldots, x_n(t))$. The main feature of a Hopfield network is the existence of a quadratic state function, i.e. the *energy function*:

$$E(x) = -\frac{1}{2} x^T W x + x^T \gamma \tag{2}$$

This is a non increasing function w.r.t. the evolution of the network according to the activation rules (1), i.e.

$$E(x(0)) \geq E(x(1)) \geq \ldots \geq E(x(t)) \geq \ldots$$

It is easy to show that every dynamics of the network converges to an equilibrium state $\hat{x} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)$, where, by updating each neuron i, the value $\hat{x}_i$ doesn't change for any $i \in \{1, 2, \ldots, n\}$. In this sense a DHN is a local minimizer of the energy function, and $\hat{x}$ is also called "attractor" of the dynamics.

4 Learning Issues in Hopfield Networks

Hopfield networks have been used in many different applications, including content-addressable memory [23, 24, 25], discrete nonlinear optimization [26], binary classification [21]. In particular in [21] is described a binary classifier for gene function prediction, named GAIN, that exploits DHNs as semi-supervised learners. According to the semi-supervised set-up described in Section 2, GAIN considers a set V of genes divided into $(U,\ S)$, together with an index w_{ij} of similarity between genes i and j, with $0 \leq w_{ij} \leq 1$. Finally, S is divided into the genes with positive labels S^+ and negative labels S^-. The aim is to predict a bipartition (U^+, U^-) of genes U.

To solve the problem, a DHN with connection strength w_{ij}, thresholds 0 and neuron states $\{1, -1\}$ is considered; let observe that, up to the multiplicative constant $\frac{\sqrt{2}}{2}$, in our setting the neuron states correspond to $\alpha = \frac{\pi}{4}$. The network is initialized with the state $\overline{x} = (\overline{u}, \overline{s})$ by assigning 1 to neurons in S^+, -1 to neurons in S^- and a random value to those in U (subvector $\overline{u}$). The equilibrium state $\hat{x} = (\hat{u}, \hat{s})$ reached by the asynchronous dynamics is used to infer the bipartition (U^+, U^-) of U by setting $U^+ = \{i \in U \mid \hat{u}_i = 1\}$ and $U^- = \{i \in U \mid \hat{u}_i = -1\}$.

This approach leads to three main drawbacks:

1. *Preservation of the prior knowledge.* During the network dynamics each neuron is updated, and the available prior information coded in the bipartition (S^+, S^-) of S may not be preserved. This happens when the reached state $\hat{x} = (\hat{u}, \hat{s})$ is such that $\hat{s} \neq \overline{s}$.
2. *Limit attractors problem.* By assigning the value 1 to positive labels, -1 to those negative and by setting to 0 the threshold of each neuron, when $|S^+| \ll |S^-|$ the network is likely to converge to a trivial state: in fact, the network dynamics in this case leads to the trivial attractor $(-1, -1, \ldots, -1)$. It is notable that this behaviour has been frequently registered in several real-world problems, e.g. the gene function prediction problem [27, 22].
3. *Incoherence of the prior knowledge coding.* Since the inference criterion is based on the minimization of the overall objective function, we expect that the initial state $\overline{s}$ of labeled neurons is a subvector of a state $(\overline{s}, \hat{u})$ "close" to a minimum of the energy function. Unfortunately, in many cases this is not true.

To address these problems, we exploit a simple property which holds for sub-networks of a DHN, and that we discuss in the next section.

5 Sub-network Property

Let be $H = < W, \gamma, \alpha >$ a network with neurons $V = \{1, 2, \ldots, n\}$, having the following bipartitions: (U, S) bipartition of V, where up to a permutation, $U = \{1, 2, \ldots, h\}$ and $S = \{h+1, h+2, \ldots, n\}$; (S^+, S^-) bipartition of S; (U^+, U^-) bipartition of U.

According to (U, S), each network state x can be decomposed in $x = (u, s)$, where u and s are respectively the states of neurons in U and in S. The energy function of H can be written by separating the contributions due to U and S:

$$E(u, s) = -\frac{1}{2}\left(u^T W_{uu} u + s^T W_{ss} s + u^T W_{us} s + s^T W_{us}^T u\right) + u^T \gamma^u + s^T \gamma^s, \quad (3)$$

where $W = \begin{pmatrix} W_{uu} & W_{us} \\ W_{us}^T & W_{ss} \end{pmatrix}$ and $\gamma = (\gamma^u, \gamma^s)$.

By setting to a given state $\tilde{s}$ the neurons in S, we consider the dynamics obtained by updating only neurons in U, without changing the state of neurons in S. Since

$$E(u, \tilde{s}) = -\frac{1}{2} u^T W_{uu} u + u^T (\gamma^u - W_{us}\tilde{s}) - \frac{1}{2}\tilde{s}^T W_{ss}\tilde{s} + \tilde{s}^T \gamma^s,$$

the dynamics of neurons in U is described by the subnetwork $H_{U|\tilde{s}} =< W_{uu}, \gamma^u - W_{us}\tilde{s}, \alpha >$. It holds the following:

Fact 5.1 (Sub-network property). If $\tilde{s}$ is part of a energy global minimum of H, and $\tilde{u}$ is a energy global minimum of $H_{U|\tilde{s}}$, then $(\tilde{u}, \tilde{s})$ is a energy global minimum of H.

In our setting, we associate the state $x(S^+, S^-)$ with the given bipartition (S^+, S^-) of S:

$$x_i(S^+, S^-) = \begin{cases} \sin\alpha & \text{if } \ i \in S^+ \\ -\cos\alpha & \text{if } \ i \in S^- \end{cases}$$

for each $i \in S$. By the sub-network property, if $x(S^+, S^-)$ is part of a energy global minimum of H, we can predict the hidden part relative to neurons U by minimizing the energy of $H_{U|x(S^+,S^-)}$.

6 *COSNet*

In this section we propose *COSNet* (COst-Sensitive neural Network), a semi-supervised learning algorithm whose main feature is the introduction of a supervised learning strategy which exploits the sub-network property to automatically estimate the parameters α and γ of the network $H =< W, \gamma, \alpha >$. The main steps of *COSNet* can be summarized as follows:

INPUT: symmetric connection matrix $W : V \times V \longrightarrow [0, 1]$, bipartition (U, S) of V and bipartition (S^+, S^-) of S.
OUTPUT: bipartition (U^+, U^-) of U.

Step 1. Generate an initial temporary bipartition (U^+, U^-) of U such that $\frac{|U^+|}{|U|} \simeq \frac{|S^+|}{|S|}$.

Step 2. Find the optimal parameters $(\hat{\alpha}, \hat{\gamma})$ of the Hopfield sub-network $H_{S|x(U^+,U^-)}$, such that the state $x(S^+, S^-)$ is "as close as possible" to an equilibrium state.

Step 3. Extend the parameters $(\hat{\alpha}, \hat{\gamma})$ to the whole network and run the sub-network $H_{U|x(S^+,S^-)}$ until an equilibrium state $\hat{u}$ is reached. The final solution (U^+, U^-) is:

$$U^+ = \{i \in U \mid \hat{u}_i = \sin \hat{\alpha}\}$$
$$U^- = \{i \in U \mid \hat{u}_i = -\cos \hat{\alpha}\}.$$

Below we explain in more details each step of the algorithm.

6.1 Generating a Temporary Solution

To build the sub-network $H_{S|x(U^+,U^-)}$, we need to provide an initial bipartition of U. The adopted procedure is the following:

- generate a random number m according to the binomial distribution $B(|U|, \frac{|S^+|}{|S|})$
- assign to U^+ m elements uniformly chosen in U
- assign to U^- the set $U \setminus U^+$.

This bipartition criterion comes from the probabilistic model described below.

Suppose that V contains some positive and negative examples, a priori unknown, and that all bipartitions $(U,\ S)$ of V are equiprobable, with $|U| = h$. If S contains $|S^+|$ positive examples, while U is not observed, then by setting $P(x) = \mathit{Prob}\,\{|U^+| = x \ \mid\ S \text{ contains } |S^+| \text{ positives}\}$, it is easy to see that the following equality holds:

$$\frac{|S^+|}{|S|} \cdot h = \underset{x}{\operatorname{argmax}} \ \ P(x).$$

In the next section we exploit this labeling of U to estimate the parameters α and γ of the network.

6.2 Finding the Optimal Parameters

By exploiting the temporary bipartition (U^+, U^-) of U found in the previous step, we consider the sub-network $H_{S|x(U^+,U^-)} = < W_{ss}, \gamma^s - W_{us}^T x(U^+, U^-), \alpha >$, where $\gamma_i^s = \gamma \in \mathbb{R}$ for each $i \in \{h+1, h+2, \ldots, n\}$. The aim is to find the values of the parameters α and γ such that the state $x(S^+, S^-)$ is "as close as possible" to an equilibrium state.

For each node k in S let define $\Delta(k) \equiv (\Delta^+(k), \Delta^-(k))$, where

$$\Delta^+(k) = \sum_{j \in S^+ \cup\ U^+} w_{kj}$$
$$\Delta^-(k) = \sum_{j \in S^- \cup\ U^-} w_{kj}.$$

In this way, each element $k \in S$ corresponds to a point $\Delta(k)$ in the plane. In particular, let consider the sets $I^+ = \{\Delta(k), k \in S^+\}$ and $I^- = \{\Delta(k), k \in S^-\}$. It holds the following:

Fact 6.1: I^+ is linearly separable from I^- if and only if there is a couple (α, γ) such that $x(S^+, S^-)$ is an equilibrium state for the network $H_{S|x(U^+,U^-)}$.

This fact suggests a method to optimize the parameters α and γ. Let be $f_{\alpha,\gamma}$ a straight line in the plane that separates the points $I^+_{\alpha,\gamma} = \{\Delta(k) \mid f_{\alpha,\gamma}(\Delta(k)) \geq 0\}$ from points $I^-_{\alpha,\gamma} = \{\Delta(k) \mid f_{\alpha,\gamma}(\Delta(k)) < 0\}$:

$$f_{\alpha,\gamma}(y, z) = \cos\alpha \cdot y - \sin\alpha \cdot z - \gamma = 0 \tag{4}$$

Note that we assume that the positive half-plane is "above" the line $f_{\alpha,\gamma}$.

To optimize the parameters (α, γ) we adopt the F-score maximization criterion, since it can be shown that $F_{score}(\alpha, \gamma) = 1$ iff $x(S^+, S^-)$ is an equilibrium state of $H_{S|x(U^+,U^-)}$. We obtain

$$(\hat{\alpha}, \hat{\gamma}) = \underset{\alpha,\gamma}{\operatorname{argmax}} \; F_{score}(\alpha, \gamma). \tag{5}$$

In order to reduce the computational complexity of this optimization, we propose a two-step approximation algorithm that at first computes the optimum line (in terms of the F_{score} criterion) among the ones crossing the origin of the axes, and then computes the optimal intercept:

1. *Compute* $\hat{\alpha}$. The algorithm computes the slopes of the lines crossing the origin and each point $\Delta(k) \in I^+ \cup I^-$. Then it searches the line which maximizes the F_{score} criterion by sorting the computed lines according to their slopes in an increasing order. Since all the points lie in the first quadrant, this assures that the angle $\hat{\alpha}$ relative to the optimum line is in the interval $[0, \frac{\pi}{2}]$.
2. *Compute* $\hat{\gamma}$. Compute the intercepts of the lines whose slope is $\tan\hat{\alpha}$ and crossing each point belonging to $I^+ \cup I^-$. The optimum line is identified by scanning the computed lines according to their intercept in an increasing order. Let $\hat{q}$ be the intercept of the optimum line, then we set $\hat{\gamma} = -\hat{q}\cos\hat{\alpha}$.

Both step 1 and step 2 can be computed in $\mathcal{O}(n \log n)$ computational time (due to the sorting), where n is the number of points.

6.3 Network Dynamics

The optimum parameters $(\hat{\alpha}, \hat{\gamma})$ computed in the previous step are then extended to the sub-network $H_{U|x(S^+,S^-)} = < W_{uu}, \hat{\gamma}^u - W_{su}^T x(S^+, S^-), \hat{\alpha} >$, where $\hat{\gamma}^u_i = \hat{\gamma}$ for each $i \in \{1, 2, \ldots, h\}$. Then, by running the sub-network $H_{U|x(S^+,S^-)}$, we learn the unknown labels of neurons U, preserving the prior information coded in the labels of neurons in S.

The initial state of the network is set to $u_i = 0$ for each $i \in \{1, 2, \ldots, h\}$. When the position of the positive half-plane in the maximization problem (5) is "above" the line, the update rule for node i at time $t + 1$ is

$$u_i(t+1) = \begin{cases} \sin\hat{\alpha} & \text{if } \sum\limits_{j=1}^{i-1} w_{ij} u_j(t+1) + \sum\limits_{k=i+1}^{h} w_{ik} u_k(t) - \theta_i < 0 \\ -\cos\hat{\alpha} & \text{if } \sum\limits_{j=1}^{i-1} w_{ij} u_j(t+1) + \sum\limits_{k=i+1}^{h} w_{ik} u_k(t) - \theta_i > 0 \end{cases} \tag{6}$$

where $\theta_i = \hat{\gamma} - \sum\limits_{j \in S} w_{ij} x_j(S^+, S^-)$. When the position of the positive half-plane is "below" the line, the disequalities (6) need to be reversed: the first one becomes "$\sin \hat{\alpha}$ if $\ldots > 0$", and the second "$-\cos \hat{\alpha}$ if $\ldots < 0$".

The stable state $\hat{u}$ reached by this dynamics is used to classify unlabeled data. If the known state $x(S^+, S^-)$, with the parameters found according to the procedure described in Section 6.2, is a part of a global minimum of the energy of H, and $\hat{u}$ is an energy global minimum of $H_{U|x(S^+,S^-)}$, the sub-network property (Section 5) guarantees that $(\hat{u}, x(S^+, S^-))$ is a energy global minimum of H.

6.4 *COSNet* Covers Hopfield Networks Learning Issues

In this section we analyze the effectiveness of the proposed algorithm w.r.t the learning issues described in Section 4.

1. Preservation of the prior knowledge. The restriction of the dynamics to the unlabeled data assures the preservation of the prior knowledge coded in the connection matrix and in the bipartition of the labeled data. Note that a similar approach has been proposed in [8], even if in that case the known labels are simply restored at each iteration of the algorithm, without an actual restriction of the dynamics.

In addition, the restriction of the dynamics to the unlabeled neurons reduces the time complexity, since often unlabeled data are much less than the labeled ones. This is an important advantage when huge and complex graphs, e.g. biological networks, are analyzed.

2. Limit attractors problem. This problem may occur when training data are characterized by a large unbalance between positive and negative examples, e.g. when $|S^+| \ll |S^-|$, which is frequent in many real-world problems [18]. In this case the points $\Delta(k) \equiv (\Delta^+(k), \Delta^-(k))$ (Section 6.2) are such that $\Delta^-(k) \gg \Delta^+(k)$. Accordingly, a separation angle $\frac{\pi}{4} \leq \hat{\alpha} \leq \frac{\pi}{2}$ is computed by the supervised algorithm described in Section 6.2. In our setting, such an angle determines a value of the positive states greater than the negative ones, yielding the network dynamics to converge towards non trivial attractors.

3. Incoherence of the prior knowledge coding. We would like to show that the parameters (α, γ) automatically selected by *COSNet* can yield to a "more coherent" state w.r.t. the prior knowledge, in the sense that this state corresponds to a lower energy of the underlying network.

To this end, by considering the data sets used in the experimental validation tests (Section 7), in which a labeling $\overline{x} \in \{1, -1\}^{|V|}$ of V is known, we randomly choose a subset U of V. After hiding the corresponding labels, by applying *COSNet* we approximate the optimal parameters $(\hat{\alpha}, \hat{\gamma})$. Accordingly, we define the state $\overline{x}(\hat{\alpha})$ by setting $\overline{x}_k(\hat{\alpha}) = \sin \hat{\alpha}$ if $\overline{x}_k = 1$ and $\overline{x}_k(\hat{\alpha}) = -\cos \hat{\alpha}$ if $\overline{x}_k = -1$, for each $k \in \{1 \ldots |V|\}$. We show that the state $\overline{x}(\hat{\alpha})$ is "more coherent" with the prior knowledge than $\overline{x}$, by studying whether $\overline{x}(\hat{\alpha})$ is "closer" than $\overline{x}$ to a global minimum of the energy function $E(x)$.

As measure of "closeness" of a given state z to a global minimum of $E(x)$, we consider the probability P_z that $E(x) < E(z)$, where $x = (x_1, x_2, \ldots, x_{|V|})$

is a random state generated according to the binomial distribution $B(|V|, \rho_z)$, where ρ_z is the rate of positive components in z.

To estimate P_z, we independently generate t random states $x^{(1)}$, $x^{(2)}$, ..., $x^{(t)}$ and we set $Y = \sum_{i=1}^{t} \beta(E(z) - E(x(i)))$, where $\beta(x) = 1$ if $x \geq 0$, 0 otherwise. The variable $\frac{Y}{t}$ is an estimator of p_z, and in our setting $Y << t$. For determining the confidence interval of P_z at a $1-\delta$ confidence level, we need to consider three cases:

1. $Y = 0$. We can directly compute the confidence interval $[0, 1 - \delta^{\frac{1}{t}}]$.

2. $1 \leq Y \leq 5$. Y is approximately distributed according to the Poisson distribution with expected value $\lambda = Y$. Accordingly, the confidence interval is $\left[\frac{1}{2n}\chi^2_{2Y,1-\frac{\delta}{2}}, \frac{1}{2n}\chi^2_{2(Y+1),\frac{\delta}{2}}\right]$, where χ^2_k is a chi squared random variable with k degrees of freedom.

3. $Y > 5$. The random variable Y is approximately distributed according to a normal distribution with expected value Y and variance $\frac{Y(1-Y)}{t}$. We adopt the Agresti-Coull interval estimator [28], which is more stable for values of Y closer to the outliers [29]. The resulting confidence interval is $\frac{Y+2}{t+4} \pm \frac{1}{t+4}\sqrt{(Y+2)(t-Y-2)}z_{1-\frac{\delta}{2}}$, where $z_{1-\alpha}$ is the $1-\alpha$ percentile of the standard normal distribution.

By setting $\delta = 0.05$ and $t = 1000$, we estimated the confidence interval for both $P_{\overline{x}(\hat{\alpha})}$ and $P_{\overline{x}}$ for the data sets used in the experimental phase and for all the FunCat classes considered in Section 7. In Table 1 we report the comparison of the confidence intervals of $P_{\overline{x}(\hat{\alpha})}$ and $P_{\overline{x}}$ in the *PPI-VM* data set and for some of the considered FunCat classes. Similar results are obtained also with the other data sets.

We distinguish two main cases: a) both the confidence intervals coincide with the minimum interval $[0, 0.0030]$, case coherent with the prior information; b) both lower and upper bounds of $P_{\overline{x}(\hat{\alpha})}$ are less than the corresponding bounds of $P_{\overline{x}}$. It is worth noting that, in almost all cases, the probability $P_{\overline{x}(\hat{\alpha})}$ has an upper bound smaller than the lower bound of $P_{\overline{x}}$. This is particularly evident for classes "01.03.16.01", "02.13" and "11.02.01"; in the latter the lower bound of $P_{\overline{x}}$ is 0.7761, while the corresponding upper bound of $P_{\overline{x}(\hat{\alpha})}$ is $\simeq 0$.

These results, reproduced with similar trends in other data sets (data not shown), point out the effectiveness of our method in approaching the problem of the incoherence of the prior knowledge coding.

Table 1: Confidence interval estimation for the probabilities $P_{\overline{x}(\hat{\alpha})}$ and $P_{\overline{x}}$ at a confidence level 0.95 (data set PPI-VM).

Data set PPI-VM										
Class	Confidence interval					Class	Confidence interval			
	$P_{\overline{x}}(\hat{\alpha})$		$P_{\overline{x}}$				$P_{\overline{x}}(\hat{\alpha})$		$P_{\overline{x}}$	
	min	max	min	max			min	max	min	max
"01"	0	0.0030	0	0.0030		"02"	0	0.0030	0	0.0030
"01.01"	0	0.0030	0	0.0030		"02.01"	0	0.0030	0.0638	0.0975
"01.01.03"	0.0001	0.0056	0.0433	0.0722		"02.07"	0	0.0030	0.0011	0.0102
"01.01.06"	0.0001	0.0056	0.0442	0.0733		"02.10"	0	0.0030	0.0522	0.0833
"01.01.06.05"	0.0210	0.0427	0.0702	0.1051		"02.11"	0.0002	0.0072	0.0939	0.1332
"01.01.09"	0	0.0030	0.0045	0.0174		"02.13"	0.0312	0.0565	0.3622	0.4226
"01.02"	0.0001	0.0056	0.0067	0.0212		"02.13.03"	0.7139	0.7681	0.7740	0.8236
"01.03"	0	0.0030	0.0620	0.0953		"02.19"	0.0001	0.0056	0.0006	0.0088
"01.03.01"	0.1452	0.1915	0.2232	0.2768		"02.45"	0.1022	0.1428	0.1815	0.2312
"01.03.01.03"	0	0.0030	0.0145	0.0333		"11"	0	0.0030	0	0.0030
"01.03.04"	0.5020	0.5637	0.6280	0.6867		"11.02"	0	0.0030	0	0.0030
"01.03.16"	0.0025	0.0135	0.1189	0.1619		"11.02.01"	0	0.0030	0.7761	0.8255
"01.03.16.01"	0	0.0030	0.3025	0.3608		"11.02.02"	0.2184	0.2716	0.8519	0.8931

7 Results and Discussion

We evaluated the performance of the proposed algorithm on the gene function prediction problem, a real-world multi-class, multi-label classification problem characterized by hundreds of functional classes. In this context the multi-label classification can be decomposed in a set of dichotomic classification problems by which genes can be assigned or not to a specific functional class. Classes are usually unbalanced, that is positive examples are significantly less than negatives, and different biomolecular data sources, able to capture different features of genes, can be used to predict their functions.

7.1 Experimental Set-up

We performed genome-wide predictions of gene functions with the yeast model organism, using the whole FunCat ontology [22], a taxonomy of functional classes structured according to a tree forest [1]. To this end we used five different biomolecular data sources, previously analyzed in [30]. The main characteristics of the data can be summarized as follows:

- *Pfam-1* data are represented as binary vectors: each feature registers the presence or absence of 4,950 protein domains obtained from the Pfam (Protein families) data base. This dataset contains 3529 genes.
- *Pfam-2* is an enriched representation of Pfam domains by replacing the binary scoring with log E-values obtained with the HMMER software toolkit [31].

[1] We used the funcat-2.1 scheme with the annotation data funcat-2.1_data_20070316, available from: ftp://ftpmips.gsf.de/yeast/catalogues/funcat/funcat-2.1_data_20070316.

- *Expr* data contains gene expression measures of 4523 genes relative to two experiments described in [32] and [33].
- *PPI-BG* data set contains protein-protein interaction data downloaded from the BioGRID database [34]. Data are binary: they represent the presence or absence of protein-protein interactions for 4531 proteins.
- *PPI-VM* is another data set of protein-protein interactions that collects binary protein-protein interaction data for 2338 proteins from yeast two-hybrid assay, mass-spectrometry of purified complexes, correlated mRNA expression and genetic interactions [35].

For *PPI* data we adopt the scoring function used by Chua et al [36], which assigns to genes i and j the similarity score

$$S_{ij} = \frac{2|N_i \cap N_j|}{|N_i - N_j| + 2|N_i \cap N_j| + 1} \times \frac{2|N_i \cap N_j|}{|N_j - N_i| + 2|N_i \cap N_j| + 1}$$

where N_k is the set of the neighbors of gene k (k is included). Informally, this score is a way to take in account the interaction partners shared by the two genes: when two genes share a high number of neighboring genes, the score is close to 1, otherwise it is close to 0. When two genes share similar interactions, it is likely that they share also similar biological functions.

The remaining data sets associate to each gene a feature vector; in these cases, the score for each gene pair is set to the Pearson's correlation coefficient of the corresponding feature vectors. For *Expr* data we computed the squared correlation coefficient to equally consider positive and negative correlated expression between genes.

To reduce the complexity of the network and the noise introduced by too small edge weights, as a pre-processing step we eliminated edges below a given threshold. In this way we removed very weak similarities between genes, but at the same time we chose low thresholds to avoid the generation of "singletons" with no connections with other nodes. In brief, we tuned the threshold for each dataset so that each vertex has at least one connection: in this way we obtained a 0.05 threshold for *Expr*, 0.15 for *Pfam-2*, 0.0027 for *Pfam-1*, 0.01 for *PPI-VM* and 0.04 for *PPI-BG*.

Moreover, to avoid training sets with a too small number of positive examples, according to the protocol followed in [30], for each dataset we selected the classes with at least 20 positives, thus resulting in about 200 functional classes for each considered data set.

7.2 Results

We compared COSNet with other semi-supervised label propagation algorithms and supervised machine learning methods proposed in the literature for the gene function prediction problem. We considered the classical *GAIN* algorithm [21], based on Hopfield networks; *LP-Zhu*, a semi-supervised learning method based on label propagation [8]; *SVM-l* and *SVM-g*, i.e. respectively linear and gaussian kernel SVMs with probabilistic output [37]. SVMs had previously been shown

Table 2: Performance comparison between GAIN, *COSNet* , LP-Zhu, SVM-l, SVM-g.

Dataset	**Methods**					**Performance**
	GAIN	*COSNet*	LP-Zhu	SVM-l	SVM-g	**measures**
Pfam-1	0.9615	0.9570	0.9613	0.7528	0.7435	Accuracy
	0.0277	**0.3892**	0.0120	0.2722	0.2355	F-score
Pfam-2	0.9613	0.9020	0.9656	0.7048	0.7515	Accuracy
	0.0296	**0.3233**	0.2117	0.1054	0.0270	F-score
Expr	0.9655	0.4617	0.9655	0.7496	0.7704	Accuracy
	0	**0.0957**	0.0008	0.0531	0.0192	F-score
PPI-BG	0.9666	0.9455	0.9704	0.7679	0.7597	Accuracy
	0.0362	**0.3486**	0.1758	0.1546	0.1178	F-score
PPI-VM	0.9554	0.9363	0.9560	0.7237	0.7222	Accuracy
	0.1009	**0.3844**	0.2106	0.1888	0.2351	F-score

to be among the best algorithms for predicting gene functions in a ”flat” setting (that is without considering the hierarchical relationships between classes) [38, 39].

To estimate the generalization capabilities of the compared methods we adopted a stratified 10-fold cross validation procedure, by ensuring that each fold includes at least one positive example for each classification task. Considering the severe unbalance between positive and negative classes, beyond the classical accuracy, we computed the F-score for each functional class and for each considered data set. Indeed in this context the accuracy is only partially informative, since a classifier predicting always ”negative” could obtain a very high accuracy. Table 2 shows the average F-score and accuracy across all the classes and for each data set.

The results show that *COSNet* achieves the best performances (in terms of the F-score) w.r.t. all the other methods. The *LP-Zhu* method is the second best method in *Pfam-2* and *PPI-BG* data sets, but obtains very low performances with *Pfam-1* and *Expr* data. These overall results are confirmed by the Wilcoxon signed-ranks test [40]: we can register a significant improvement in favour of *COSNet* with respect to all the other methods and for each considered data set at $\alpha = 10^{-15}$ significance level.

In order to understand the reasons for which our method works better, we compared also the overall precision and recall of the methods separately for each data set: we did not consider *GAIN*, since this methods achieved the worst results in almost all the data sets. For lack of room, in Figure 1 we show only the results relative to *Pfam-2* and *PPI-VM* data sets. We can observe that, while *COSNet* does not achieve the best precision or recall, it obtains the best F-score as a result of a good balancing between them. These results are replicated with the other data sets, even if with Pfam-1 and Expr data *COSNet* achieves also the best average precision and recall (data not shown).

We think that these results come from the *COSNet* cost-sensitive approach that allows to automatically find the ”near-optimal” parameters of the network with respect to the distribution of positive and negative nodes (Section 6). It is worth noting that using only single sources of data *COSNet* can obtain a relatively high precision, without suffering a too high decay of the recall. This is of

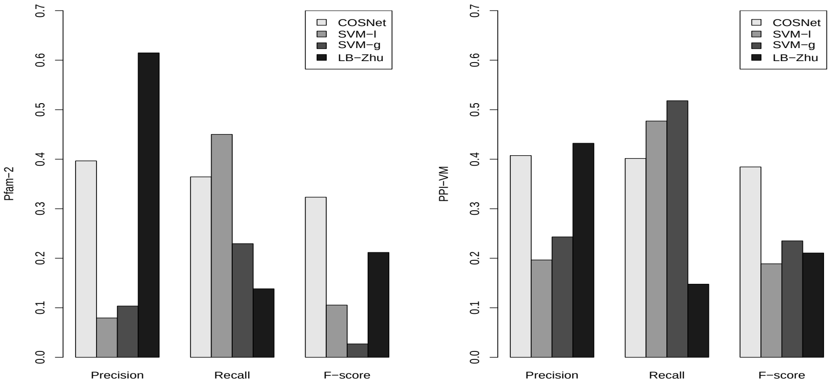

Fig. 1: Average precision, recall and F-score for each compared method (excluding *GAIN*). Left: Pfam-2; Right: PPI-VM

paramount importance in the gene function prediction problem, where ”in silico” positive predictions of unknown genes need to be confirmed by expensive ”wet” biological experimental validation procedures. From this standpoint the experimental results show that our proposed method could be applied to predict the ”unknown” functions of genes, considering also that data fusion techniques could in principle further improve the reliability and the precision of the results [2, 41].

8 Conclusions

We introduced an effective neural algorithm, *COSNet*, which exploits Hopfield networks for semi-supervised learning in graphs. *COSNet* adopts a cost sensitive methodology to manage the unbalance between positive and negative labels, and to preserve and coherently encode the prior information. We applied *COSNet* to the genome-wide prediction of gene function in yeast, showing a large improvement of the prediction performances w.r.t. the compared state-of-the-art methods.

By noting that the parameter γ of the neural network may assume different values for each node, our method could be extended by allowing a different activation threshold for each neuron. To avoid overfitting due to the increment of network parameters, this approach should be paired with proper regularization techniques. Moreover, by exploiting the supervised learning of network parameters, *COSNet* could be also adapted to combine multiple sources of networked data: indeed the accuracy of the linear classifier on the labeled portion of the network could be used to ”weight” the associated source of data, in order to obtain a ”consensus” network, whose edges are the result of a weighted combination of multiple types of data.

Acknowledgments

The authors gratefully acknowledge partial support by the PASCAL2 Network of Excellence under EC grant no. 216886. This publication only reflects the authors' views.

References

[1] Zheleva, E., Getoor, L., Sarawagi, S.: Higher-order graphical models for classification in social and affiliation networks. In: NIPS 2010 Workshop on Networks Across Disciplines: Theory and Applications, Whistler BC, Canada (2010)

[2] Mostafavi, S., Morris, Q.: Fast integration of heterogeneous data sources for predicting gene function with limited annotation. Bioinformatics **26**(14) (2010) 1759–1765

[3] Vazquez, A. et al.: Global protein function prediction from protein-protein interaction networks. Nature Biotechnology **21** (2003) 697–700

[4] Leskovec, J. et al.: Statistical properties of community structure in large social and information networks. In: Proc. 17th Int. Conf. on WWW (2008) 695–704

[5] Bilgic, M., Mihalkova, L., Getoor, L.: Active learning for networked data. In: Proc. of the 27th ICML, Haifa, Israel (2010)

[6] Marcotte, E. et al.: A combined algorithm for genome-wide prediction of protein function. Nature **402** (1999) 83–86

[7] Oliver, S.: Guilt-by-association goes global. Nature **403** (2000) 601–603

[8] Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning with gaussian fields and harmonic functions. In: Proc. of the 20th ICML, Washintgton DC, USA (2003)

[9] Zhou, D., et al.: Learning with local and global consistency. In: Adv. Neural Inf. Process. Syst. Volume 16. (2004) 321–328

[10] Szummer, M., Jaakkola, T.: Partially labeled classification with markov random walks. In: NIPS 2001. Volume 14., Whistler BC, Canada (2001)

[11] Azran, A.: The rendezvous algorithm: Multi- class semi-supervised learning with Markov random walks. In: Proc. of the 24th ICML. (2007)

[12] Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: COLT 2004. (2004)

[13] Delalleau, O., Bengio, Y., Le Roux, N.: Efficient non-parametric function induction in semi-supervised learning. In: Proc. of the Tenth Int. Workshop on Artificial Intelligence and Statistics. (2005)

[14] Belkin, M., Niyogi, P.: Using manifold structure for partially labeled classification. In: Adv. Neural Inf. Process. Syst. Volume 15. (2003)

[15] Nabieva, E. et al.: Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. Bioinformatics **21**(S1) (2005) 302–310

[16] Deng, M., Chen, T., Sun, F.: An integrated probabilistic model for functional prediction of proteins. J. Comput. Biol. **11** (2004) 463–475

[17] Tsuda, K., Shin, H., Scholkopf, B.: Fast protein classification with multiple networks. Bioinformatics **21**(Suppl 2) (2005) ii59–ii65

[18] Mostafavi, S. et al.: GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. Genome Biology **9**(S4) (2008)

[19] Hopfield, J.: Neural networks and physical systems with emergent collective compautational abilities. Proc. Natl Acad. Sci. USA **79** (1982) 2554–2558

[20] Bengio, Y., Delalleau, O., Le Roux, N.: Label Propagation and Quadratic Criterion. In: Semi-Supervised Learning. MIT Press (2006) 193–216

[21] Karaoz, U., et al.: Whole-genome annotation by using evidence integration in functional-linkage networks. Proc. Natl Acad. Sci. USA **101** (2004) 2888–2893

[22] Ruepp, A., et al.: The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. Nucleic Acids Research **32**(18) (2004) 5539–5545

[23] Wang, D.: Temporal pattern processing. In: The Handbook of Brain Theory and Neural Networks. (2003) 1163–1167

[24] Liu, H., Hu, Y.: An application of hopfield neural network in target selection of mergers and acquisitions. Int. Conf on Business Intelligence and Financial Engineering (2009) 34–37

[25] Zhang, F., Zhang, H.: Applications of a neural network to watermarking capacity of digital image. Neurocomputing **67** (2005) 345–349

[26] Tsirukis, A.G., Reklaitis, G.V., Tenorio, M.F.: Nonlinear optimization using generalized hopfield networks. Neural Comput. **1** (1989) 511–521

[27] Ashburner, M., et al.: Gene ontology: tool for the unification of biology. the gene ontology consortium. Nature Genetics **25**(1) (2000) 25–29

[28] Agresti, A., Coull, B.A.: Approximate is better than exact for interval estimation of binomial proportions. Statistical Science **52**(2) (1998) 119–126

[29] Brown, L.D., Cai, T.T., Dasgupta, A.: Interval estimation for a binomial proportion. Statistical Science **16** (2001) 101–133

[30] Cesa-Bianchi, N., Valentini, G.: Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. Journal of Machine Learning Research, W&C Proceedings, Machine Learning in Systems Biology **8** (2010) 14–29

[31] Eddy, S.R.: Profile Hidden Markov models. Bioinformatics **14**(9) (1998) 755–763

[32] Spellman, P.T., et al.: Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. Molecular Biology of the Cell **9**(12) (1998) 3273–3297

[33] Gasch, P., et al.: Genomic expression programs in the response of yeast cells to environmental changes. Mol. Biol. Cell **11**(12) (2000) 4241–4257

[34] Stark, C. et al.: Biogrid: a general repository for interaction datasets. Nucleic acids research **34**(Database issue) (2006) D535–D539

[35] von Mering, C., et al.: Comparative assessment of large-scale data sets of protein-protein interactions. Nature **417**(6887) (2002) 399–403

[36] Chua, H., Sung, W., Wong, L.: An efficient strategy for extensive integration of diverse biological data for protein function prediction. Bioinformatics **23**(24) (2007) 3364–3373

[37] Lin, H.T., Lin, C.J., Weng, R.: A note on platt's probabilistic outputs for support vector machines. Machine Learning **68**(3) (2007) 267–276

[38] Brown, M.P.S., et al.: Knowledge-based analysis of microarray gene expression data by using support vector machines. Proceedings of the National Academy of Sciences of the United States of America **97**(1) (2000) 267–276

[39] Pavlidis, P. et al.: Learning gene functional classifications from multiple data types. Journal of Computational Biology **9** (2002) 401–411

[40] Wilcoxon, F.: Individual comparisons by ranking methods. Journal of Computational Biology **1**(6) (1945) 80–83

[41] Re, M., Valentini, G.: Simple ensemble methods are competitive with state-of-the-art data integration methods for gene function prediction. Journal of Machine Learning Research, W&C Proceedings, Machine Learning in Systems Biology **8** (2010) 98–111