X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará
© Sociedade Brasileira de Inteligência Computacional (SBIC)

PREVENTING ERROR PROPAGATION IN SEMI-SUPERVISED LEARNING USING TEAMS OF WALKING PARTICLES

Fabricio Breve, and **Liang Zhao**
Institute of Mathematics and Computer Science (ICMC), University of São Paulo (USP)
{fabricio,zhao}@icmc.usp.br

Abstract – Semi-supervised learning algorithms are applied to classification problems where only a small portion of the data points is labeled. In these cases, the reliability of the labels in the labeled subset is very important, because mislabeled samples may propagate their wrong labels to a large portion of the data set. This paper presents a novel and efficient semi-supervised learning graph-based method specifically designed to handle data sets with mislabeled samples. It uses walking particles with cooperative and competitive behavior in order to propagate labels. The proposed model also incorporates some features to make it robust to large amounts of mislabeled samples. Computer simulations show the performance of the method in the presence of different amounts of mislabeled data, in networks of different sizes and mixtures. These simulations identify critical points of mislabeled subset size, below which the network is free of wrong label contamination, but above which the mislabeled samples start to propagate their labels to the rest of the network. Moreover, the proposed method is compared to other representative semi-supervised learning graph-based methods and its performance in real-world data sets is increasingly better than the others as the amount of mislabeled samples in the data set increases.

Keywords – Semi-supervised learning, Learning from imperfect data, Particle competition and cooperation, Error propagation analysis.

1. INTRODUCTION

Machine learning is the scientific field concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data. Most machine learning algorithms fall in one of the two major groups, which are *supervised learning* and *unsupervised learning*. In supervised learning, algorithms learn from a sufficient number of labeled data points. The labels are usually provided by a human expert in the specific domain of application. During the training phase, the labeled data is presented to the algorithm so it will later be able to predict labels from new data points. On the other hand, unsupervised learning algorithms learn from only unlabeled examples, trying to identify how the data is organized. [1–8].

Though semi-supervised learning had solved many practical problems in the last decades, the size of the data sets being treated is constantly increasing, thus labeling enough samples for the training process is an expensive and time consuming task, and it usually requires the work of human experts. Therefore, many data sets being treated today are composed by only a small subset of labeled samples, while the remaining samples are left unlabeled. In these situations, the efficacy of supervised learning techniques can be quite limited, as all the information carried by the unlabeled samples is simply ignored. On the other hand, unsupervised learning methods cannot take advantage of the available label information. The *semi-supervised learning* algorithms handle these problems by learning from a few labeled data points combined with a large amount of unlabeled data, with the objective of producing better classifiers while less human effort is required [9–11]. Most of the semi-supervised learning methods proposed in the last years fall into the subgroup of graph-based methods [10]. This subgroup includes methods like Mincut [12], Local and Global Consistency [13], label propagation techniques [14, 15] and others. However most of these methods are similar as they may be seen as regularization frameworks [9], differing only in the particular choice of the loss function and the regularizer [12, 13, 16–19].

The quality of the training data is very important in supervised learning, and even more in semi-supervised learning, as less labeled data is available. Humans and other animals can easily compensate for imperfect data in their learning process. Behavioral experiments show that animals can successfully learn from conditioning even when they are inconsistently rewarded. The same does not apply to machine learning systems, in which fault-tolerance is usually hard to achieve. Most algorithms just assume that the input label information is completely reliable, but in practice mislabeled samples are commonly found in the data sets. This problem is commonly refereed as *learning from imperfect data* or *learning from imperfect teacher* [20, 21]. This is an important issue in supervised learning, and it is even more critical in semi-supervised learning, where fewer labeled data items are available, and errors (wrong labels) may easily propagate to a large portion of the data set. Though this is an important topic, it has not received much attention from researchers and there are only a few recent works on semi-supervised learning from imperfect data [22–24].

Recently, a biologically inspired clustering algorithm used particle walking and competition to detect communities in networks [25]. Particles compete with each other in order to possess nodes of the network, naturally confining themselves within a cluster. Later, this approach was extended to realize semi-supervised learning [26, 27]. In this version, there are team of particles which cooperates with their teammates and compete against other teams. Particles are created for each labeled node of the

network, and they try to conquer and defend their neighborhood. Those particle walking based methods provide classification results similar to those from some well known methods, but with lower order of computational complexity.

The particle competition and cooperation approach [26] naturally has some tolerance to mislabeled data [28], though it was not designed to solve this particular problem. In this paper, we present a new semi-supervised learning particle walking algorithm which is specifically designed to handle data sets with large amounts of mislabeled data. Among the features introduced to improve its robustness in the presence of mislabeled samples, the most important are: labeled nodes that have the same label are all interconnected independently of their distance; there is only a distance table for each team, which is shared by all teammates; and all node potentials are variable, even those from labeled nodes. These features allow particles to leave mislabeled nodes, which will be usually inside other class neighborhood, and help their teammates in the neighborhood of its respective class. In our experiments, when there is no mislabeled nodes, the proposed algorithm correct classification rates are compatible with those achieved by the particles competition and cooperation method [26] in the same data set. The advantage of the proposed algorithm appears when there are mislabeled samples in the data sets, in these cases it performs better than the particles competition and cooperation method and other representative semi-supervised learning graph-based methods.

This paper is organized as follows. The proposed model is described in Section 2. In Section 3, we present some simulation results that show the effectiveness and robustness of the proposed method in the presence of mislabeled data, including a study on how the performance is affected as the network size and mixture varies, and a comparison among the proposed method and other representative semi-supervised learning graph-based methods applied to real-world data sets with mislabeled data. Finally, in Section 4 we draw some conclusions.

2. MODEL DESCRIPTION

In this section, we present the proposed semi-supervised learning method, which relies on particle competition and cooperation. A set of particles, each of them representing a labeled data item, are put in an unweighted network. A subset of particles representing nodes with the same label is called a *team*. These teams will compete against each other to possess nodes of the network. Each node has a vector to represent the domination level of each team on it. While teammates particles act cooperatively to possess the nodes of the network, particles belonging to different teams will compete with each other trying to avoid rivals to enter their territory. At each iteration of the algorithm, each particle will choose a neighbor node to visit. The chosen node is called *target node*, and the particle will increase its team domination level on it, at the same time that it will decrease other teams domination levels on this same node. Each particle also has a strength level, which lowers or raises according to its team domination level on the target node.

The semi-supervised learning problem is described as follows. Given a data set $\chi = \{x_1, x_2, \ldots, x_l, x_{l+1}, \ldots, x_n\} \subset \mathbb{R}^m$ and the corresponding label set $L = \{1, 2, \ldots, c\}$, the first l points $x_i (i \leq l)$ are labeled as $y_i \in L$ and the remaining points $x_u (l < u \leq n)$ are left unlabeled, i.e, $y_u = \emptyset$. The goal is to assign a label to each of these unlabeled samples.

The first step of the algorithm is to build a network from a given data set. Define a undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. $\mathbf{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes, where each one v_i corresponds to a sample x_i, and $\mathbf{E}$ is the set of edges (v_i, v_j). Two nodes v_i and v_j are connected if v_j is among the k-nearest neighbors of v_i or vice-versa using Euclidean distance. Also, v_i and v_j are connected if they are both labeled nodes with the same label, i.e., $y_i = y_j$ and $\{y_i, y_j\} \in L$. Otherwise, v_i and v_j are disconnected.

For each network node $v_i \in \{v_1, v_2, \ldots, v_l\}$, corresponding to a labeled data point $x_i \in \{x_1, x_2, \ldots, x_l\}$, there is a particle $\rho_i \in \{\rho_1, \rho_2, \ldots, \rho_l\}$ which initial position is at v_i, i.e., the set of particles and the set of labeled nodes have the same size. Particles representing samples with the same class labels will act like a team, collaborating with each other and competing with particles from other teams. Particles change their position through time, and they keep track of the distance between their actual position and the closest labeled node of its team (label).

The system has two different kind of dynamics: particle dynamics and node dynamics. Each particle ρ_j holds a variable $\rho_j^{\omega}(t) \in [0, 1]$ corresponding to the particle strength, which indicates how much the particle can change nodes levels at time t. Teams have a variable $\rho_{\mathbf{j}}^{\mathbf{d}}(\mathbf{t})$ which values are shared by the whole team. It is a distance table, i.e., a vector $\rho_{\mathbf{j}}^{\mathbf{d}}(\mathbf{t}) = \{\rho_i^{d_1}(t), \rho_i^{d_2}(t), \ldots, \rho_i^{d_n}(t)\}$ with the same size as V, where each element $\rho_j^{d_i}(t) \in [0 \quad n-1]$ holds the distance measured between the closest labeled node of its team and the node v_i.

Each node v_i has one vector variable $\mathbf{v}_{\mathbf{i}}^{\omega}(\mathbf{t}) = \{v_i^{\omega_1}(t), v_i^{\omega_2}(t), \ldots, v_i^{\omega_c}(t)\}$, which is the same size of L, where each element $v_i^{\omega_\ell}(t) \in [0, 1]$ corresponds to team ℓ domination level over node v_i. For each node, the sum of the domination levels is always constant, this happens because a particle increases its team domination level on the node, at the same time that it decreases other teams domination levels. Therefore, the following relationship holds:

$$\sum_{\ell=1}^{c} v_i^{\omega_\ell} = 1. \tag{1}$$

The initial domination levels are set differently for nodes corresponding to labeled and unlabeled samples. For those corresponding to labeled samples, the team which corresponds to its class has its domination level set to the highest value, while the other teams have their domination levels set to the lowest value. Meanwhile, the nodes corresponding to unlabeled samples have

all teams domination levels set equally. Therefore, for each node v_i, the initial level of domination vector $\mathbf{v_i^\omega}$ is set as follows:

$$v_i^{\omega_\ell}(0) = \begin{cases} 1 & \text{if } y_i = \ell \\ 0 & \text{if } y_i \neq \ell \text{ and } y_i \in L \\ \frac{1}{c} & \text{if } y_i = \emptyset \end{cases} . \tag{2}$$

Each particle will have its initial position set to its corresponding labeled node, while its initial strength is set to maximum, as follows:

$$\rho_j^\omega(0) = 1. \tag{3}$$

Regarding the distance tables, each particle ρ_j will know only the distance from itself to the nodes its team already visited or targeted. Therefore, at start, they will know no distances except for its team labeled nodes, which are set to 0, and the other distances will be set to the largest possible value $(n-1)$, as follows:

$$\rho_j^{d_i}(t) = \begin{cases} 0 & \text{if } \quad y_i = \rho_j^f \\ n-1 & \text{otherwise} \end{cases} , \tag{4}$$

where ρ_j^f is the class label of its team. At each iteration, each particle will calculate the distance between its target node and the closest labeled node of its team, and update its team distance table if necessary.

When it comes to the node dynamics, at each iteration t, each particle p_j selects a target neighbor node it will try to visit. Remember that each node holds a vector where the elements represent each team domination level. Different teams compete with each other for owning the network nodes, therefore particles will increase their team domination level in the target node, at the same time that they will decrease the domination level of the other teams in this same node. Therefore, for each node selected as target v_i, the domination level $v_i^{\omega_\ell}(t)$ is updated as follows:

$$v_i^{\omega_\ell}(t+1) = \begin{cases} \max\{0, v_i^{\omega_\ell}(t) - \frac{\Delta_v \rho_j^\omega(t)}{c-1}\} \\ \quad \ell \neq \rho_j^f \\ v_i^{\omega_\ell}(t) + \sum_{q \neq \ell} v_i^{\omega_q}(t) - v_i^{\omega_q}(t+1) \\ \quad \ell = \rho_j^f \end{cases} , \tag{5}$$

where $0 < \Delta_v \leq 1$ is a parameter to control changing rate of the domination levels and ρ_j^f represents the class label of particle ρ_j. If Δ_v takes a low value, the node domination levels change slowly, while if it takes a high value, the node domination levels change quickly. Each particle ρ_j will change the target node v_i by increasing the domination level of its team ($v_i^{\omega_\ell}, \ell = \rho_j^f$) while decreasing the domination levels of other teams ($v_i^{\omega_\ell}, \ell \neq \rho_j^f$), always holding to the conservation law defined by (1).

Regarding the particle dynamics, a particle will get weaker or stronger according to their current strength and the domination level of its team in the target node. If the domination level is higher than the current strength, it will become stronger, otherwise, it will become weaker. Therefore, at each iteration t, a particle strength $\rho_j^\omega(t)$ is updated as follows:

$$\rho_j^\omega(t+1) = v_i^{\omega_\ell}(t+1), \tag{6}$$

where v_i is the target node, and $\ell = \rho_j^f$, i.e., ℓ is the class label of particle ρ_j. In other words, each particle ρ_j has its strength ρ_j^ω set to the value of its team domination level $v_i^{\omega_j}$ on the target node. Therefore, a particle usually gets stronger if it targets a node his team is dominating, while it usually gets weaker if it tries to invade a node dominated by another team.

The distance table is introduced in order to keep particles aware of how far they are from a labeled node of its team, so they will avoid going too far away, situation that could let its neighborhood vulnerable to attacks from other teams. Domination levels and distance tables are designed to prevent particles from losing all their strength when they walk into enemies neighborhoods at the same time that they keep particles around to protect their own neighborhood. Each particle ρ_j updates its team distance table $\rho_j^{d_k}(t)$ at each iteration t as follows:

$$\rho_j^{d_k}(t+1) = \begin{cases} \rho_j^{d_i}(t) + 1 & \text{if } \rho_j^{d_i}(t) + 1 < \rho_j^{d_k}(t) \\ \rho_j^{d_k}(t) & \text{otherwise} \end{cases} , \tag{7}$$

where $\rho_j^{d_i}(t)$ and $\rho_j^{d_k}(t)$ are the distances to the closest labeled node of its team from the current node and from the target node, respectively.

Distance calculation is dynamic and simple: particles have limited knowledge of the network, they do not know how nodes are connected, they only know all labeled nodes with the same label are connected, so they assume the worst case, i.e., all the nodes can be reached only with a number of steps as high as the number of nodes minus one $(n-1)$ starting from any of its team labeled nodes. Every time a particle chooses a target node, it will check its team distance table, if this distance is higher than the distance of the current node plus 1, it will update the table. In other words, unknown distances are calculated on the fly and updated as particles naturally find shorter paths.

At each iteration, a particle will randomly choose any of its neighbors to target with more probability to nodes closer to the labeled nodes of its team and nodes in which its team have higher domination level. Therefore, the particle ρ_j chooses its target node v_i with probabilities defined according to its team domination level on that neighbor $\rho_j^{\omega_\ell}$ and the inverse of the distance ($\rho_j^{d_i}$) from that neighbor, v_i, to the closest labeled node of its team, v_j, as follows:

$$p(v_i|\rho_j) = (1-\alpha)\frac{W_{qi}}{\sum_{\mu=1}^{n} W_{q\mu}} + \alpha\frac{W_{qi}v_i^{\omega_\ell}\frac{1}{(1+\rho_j^{d_i})^2}}{\sum_{\mu=1}^{n} W_{q\mu}v_i^{\omega_\ell}\frac{1}{(1+\rho_j^{d_i})^2}}, \tag{8}$$

where k is the index of the node being visited by particle ρ_j and $\ell = \rho_j^f$, where ρ_j^f is the class label of particle ρ_j. $0 < \alpha < 1$ is a parameter that defines the weight of team domination levels and distances on the probabilities. When α is low, exploratory behavior dominates; and when α is high, defensive behavior dominates. Best classification performance is achieved when there is an equilibrium between exploratory and defensive behavior. Therefore, we usually set $\alpha \approx 0.5$.

A particle will actually visit the target node only if its team domination level on that node is higher than those from all other teams; otherwise, a shock happens and the particle will stay at the current node until the next iteration. This mechanism prevents a particle from entering other team territories.

Usually, after a sufficient number of iterations, most nodes will be completely dominated by a single team. However, some border nodes will still change their domination level, and sometimes they will even change their domination team. Therefore, we cannot always expect a convergence of all nodes labels, therefore we monitor the average maximum domination levels of the nodes ($\langle v_i^{\omega_\ell}\rangle$, $\ell = \arg\max_q v_i^{\omega_q}$) and stop the algorithm when there is no increasing of this quantity.

After the last iteration of the algorithm, each unlabeled node is labeled after the team which has the highest domination level on it:

$$y_i = \arg\max_{\ell} v_i^{\omega_\ell}(\infty). \tag{9}$$

Overall, the proposed algorithm can be outlined as follows:

Algorithm 1: The Proposed Algorithm

```
Build the network from the data set;
Set nodes' domination levels by using Eq. (2);
Set initial positions of particles at their corresponding nodes by using Eq. (3);
Set particle strength and distance tables by using Eq. (4);
repeat
    for each particle do
        Select the target node by using Eq. (8);
        Update target node domination levels by using Eq. (5);
        Update particle strength by using Eq. (6);
        Update particle distance tables by using Eq. (7);
until the stopping criterion is satisfied;
Label each unlabeled data by using Eq. (9);
```

3. COMPUTER SIMULATIONS

In this section, we present simulation results to show the effectiveness and robustness of our method in the presence of mislabeled data, and also a comparison of the proposed method and other representative semi-supervised learning graph-based methods. First, we evaluate the performance of the proposed method applied to networks with different sizes, mixtures and average node degrees in the presence of mislabeled data; and then we compare the proposed method with other representative semi-supervised learning graph-based methods when applied to real-world networks with mislabeled data. The following parameters were held constant in all simulations in this paper: $\Delta_v = 0.1$, and $\alpha = 0.5$. These values were obtained by empirical optimization and produce good results in the type of networks studied here.

The networks for the first group of experiments are generated by using the method proposed by [29]. In this method, pairs of nodes which belong to the same class are linked with probability p_{in}, whereas pairs belonging to different classes are connected with probability p_{out}. The average degree is defined by $\langle k\rangle$. The value of p_{out} is taken so the average number of links from a node to the nodes of any other classes, z_{out}, can be controlled. At the same time, the value of p_{in} is chosen to keep the average node degree $\langle k\rangle$ constant. Thus, $z_{out}/\langle k\rangle$ defines the mixture of the classes, and as $z_{out}/\langle k\rangle$ increases from zero, the classes become more diffuse and harder to identify.

In the first set of simulations, we generate networks with increasing number of nodes $n = \{64, 128, 192, 256, \ldots, 1024\}$, divided equally into 4 classes, average node degree proportional to the network size, $\langle k\rangle = n/8$, and fixed mixture, $z_{out}/\langle k\rangle = 0.25$, thus the average node degree increases proportionally to the network size and the mixture is kept constant. For each of these configurations we randomly select a subset of elements ($L \subset N$) to be labeled, while the others are presented to the algorithm without labels, the labeled subset size is set to $l/n = 0.1$ (10% labeled nodes is a typical semi-supervised learning problem). In

order to test robustness to mislabeled samples, we randomly choose q elements from the labeled subset L ($Q \subset L$) to have their labels changed to any of the other classes chosen randomly for each sample, thus producing mislabeled nodes. These mislabeled subsets are created with increasing sizes, $q/l = \{0.00, 0.05, 0.10, \ldots, 1.00\}$. So, we have 800 different configurations and each of them is repeated 100 times, with different generated networks and different samples in the labeled and in the mislabeled subsets, in order to obtain an average. The results are presented in Figure 1a and by analyzing them we notice that as the network size increases ($n \to +\infty$), the performance curve with variable mislabeled samples subset size becomes rougher and the critical points, beyond which the performance drops, have a narrower angle.

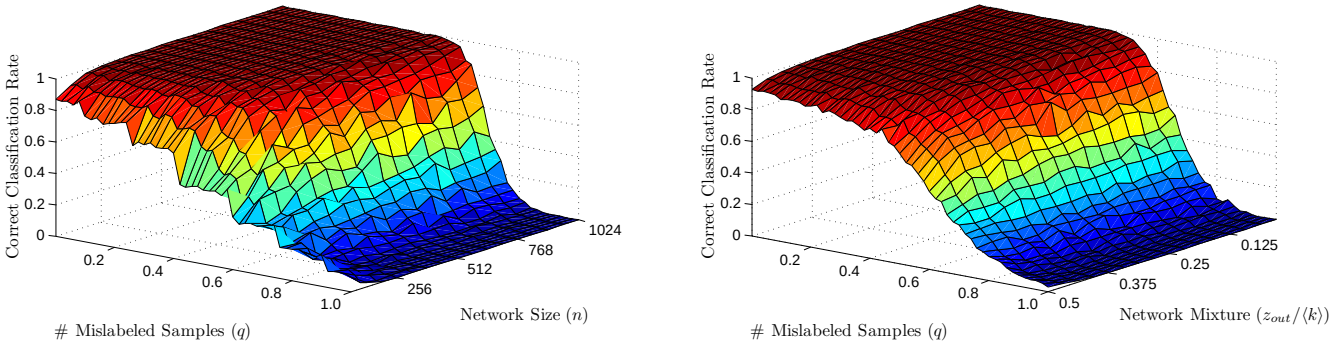


Figure 1: Correct Classification Rate with: (a) different network sizes and mislabeled subset sizes, $\langle k \rangle = n/8$, $z_{out}/\langle k \rangle = 0.25$, $l/n = 0.1$; and (b) different network mixtures and mislabeled subset sizes, $n = 512$, $\langle k \rangle = 64$, $l = 64$.

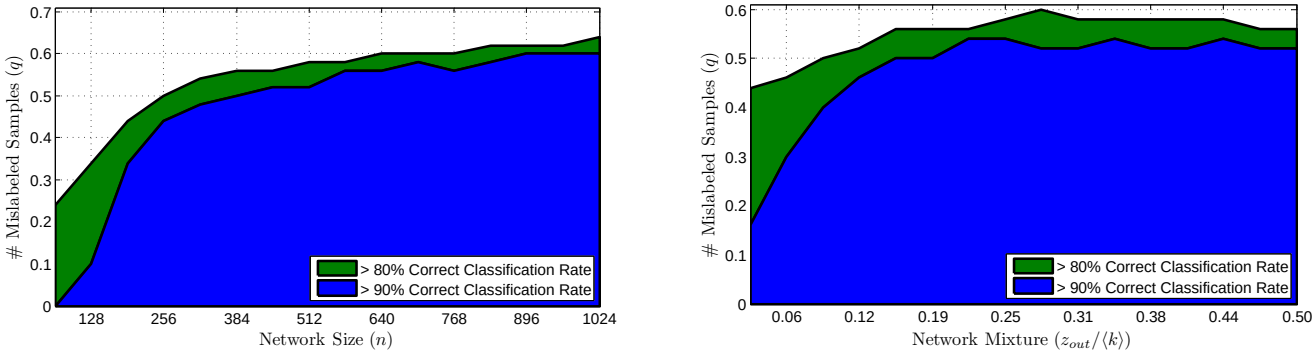
The second set of experiments is similar to the first one, but now we have fixed the network size to $n = 512$, divided equally into 4 classes, we kept the average node degree constant $\langle k \rangle = 64$, and the networks were generate with 16 levels of mixture, $z_{out}/\langle k \rangle = \{0.0625, 0.125, 0.1875, 0.25, \ldots, 0.5\}$. The labeled subset is fixed, $l = 64$, and the mislabeled subset size is variable again, $q/l = \{0.00, 0.02, 0.04, \ldots, 1.00\}$. Once more, we have 800 different configurations and each of them is repeated 100 times in order to obtain an average. The results are presented in Figure 1b, and by analyzing them we notice that as the mixture increases from low to moderate ($z_{out}/\langle k \rangle \leq 0.3125$) the performance of the algorithm and the critical points varies only within the margin of error, showing the robustness of the method. The performance only drops when $z_{out}/\langle k \rangle > 0.3125$. In those cases, as the mixture increases, the performance curve with variable mislabeled samples subset size becomes smoother and the critical points, beyond which the performance drops, have a wider angle. This is expected, since in a completely random network there would be no cluster structures, the algorithm would output random labels and therefore we could expect a correct classification rate of $\sim 25\%$ (4 equiprobable classes problem) no matter the size of the mislabeled nodes subset.

In both experiments, we notice that as the subset of mislabeled samples grows from small to moderate, there is very little effect on the performance of the algorithm, as indicated by the plateau region formed in both the figures, showing the robustness of the proposed method. In fact, in most cases we get correct classification rate higher than 90% even when more than half of labeled subset is composed by mislabeled nodes, which is pretty impressive. The competition mechanism of the algorithm and the features we have introduced in this new algorithm are responsible for this interesting phenomenon. If there is not many wrong label samples, the corresponding particles may lose the competition and consequently the wrong label propagation can be impeded. Among the features we introduced, the connections created among labeled nodes with the same label work like a highway that allows particles generated for a mislabeled sample to quickly leave the enemy territory and go help their own team. The distance table shared by the whole team also helps the particles leave mislabeled nodes and not return. Finally, as labeled nodes now have variable potentials, a mislabeled node may be taken by the particles in its neighborhood, preventing it from propagating the wrong label.

By analyzing the shape of the graphs on Figures 1a and 1b, we notice that as the mislabeled subset grows, the algorithm performance is high and almost constant in the beginning, then there is a critical point beyond the performance drops really fast, and finally there is another critical point beyond the performance stabilizes with a low value. These critical points vary with the size and mixture of the network, as expected. We are not concerned with algorithms performance beyond the second critical point because in those cases the quality of the labeled subset is worse than random labeling. In these bad cases, it would be better to use some unsupervised learning algorithm. The first critical point, on the other hand, is an important indicator of the robustness of the algorithm. The robustness of an algorithm in the presence of mislabeled samples may be measured by the size of the area before the first critical point, and also by the angle formed by the performance curve as the mislabeled subset increases. A large area before the first critical point followed by a sharp angle in the performance curve indicates that the algorithm is robust to mislabeled samples in that specific configuration. On the other hand, when the algorithm is not robust to mislabeled samples, the performance curve tends to a straight line and the critical points are hard to identify as they form wide angles. In our experiments, the larger area before the critical points and the critical points with sharper angles are obtained when the classes become more well separated and as the network size grows, which means the algorithm gets more robust in those cases. The area before the first critical point is quite large in most cases, which also points towards the robustness of the method.

The performance of the algorithm in those typical semi-supervised learning setups is also impressive, as it managed to keep

high correct classification rates even when there is a large percentage of mislabeled nodes, with different network sizes and mixtures. In Figures 2a and 2b we can observe the maximum size of the mislabeled subset that still produces good results (over 80% and 90% of correct classification rate) for different network sizes and mixtures, respectively.



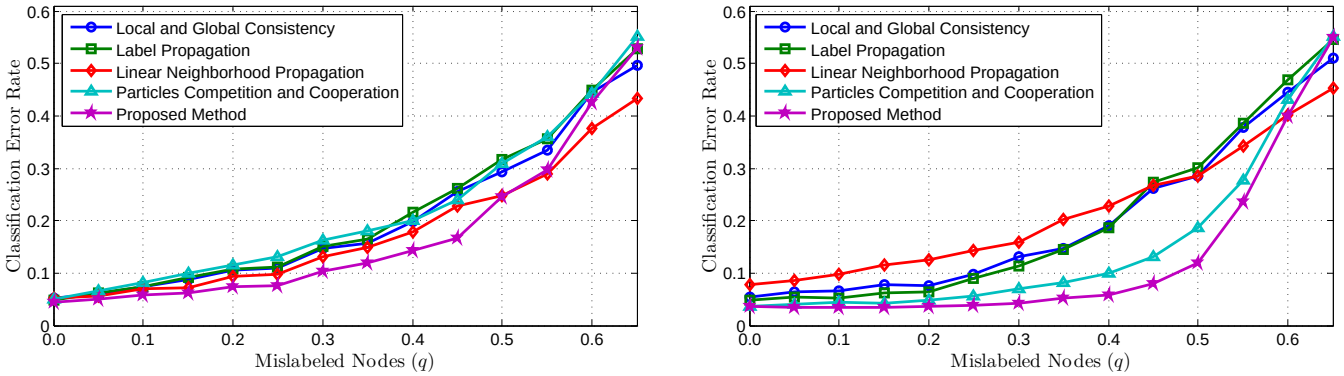
(a) (b)

Figure 2: Maximum mislabeled subset size for 80% and 90% of correct classification rate with: (a) different network sizes, $\langle k \rangle = n/8$, $z_{out}/\langle k \rangle = 0.25$, $l/n = 0.1$; and (b) different network mixtures ($z_{out}/\langle k \rangle$), $n = 512$, $\langle k \rangle = 64$.

The next step is to compare the proposed method with other representative semi-supervised learning graph-based methods when applied to real-world networks with mislabeled data. Here the performance of the proposed method is compared to those of Local and Global Consistency (LGC) [13], Label Propagation (LP) [14], Linear Neighborhood Propagation (LNP) [15], and the original Particle Competition and Cooperation (PCC) method [26]. The σ parameters of the LGC and the LP methods, and the k parameters of LNP, PCC and the proposed method, are all optimized using the genetic algorithm available in the Global Optimization Toolbox of MATLAB. For the LGC and LNP methods, we have fixed $\alpha = 0.99$, as done in [13] and [15], respectively. For the PCC method, the following parameters are kept fixed: $p_{\text{grd}} = 0.5$, $\Delta_v = 0.1$. Finally, for the proposed method, we have fixed $\alpha = 0.5$ and $\Delta_v = 0.1$.

For each data set, we randomly select a subset of elements ($L \subset N$) to be labeled, while the others are presented to the algorithm without labels. In order to test robustness to mislabeled samples, we randomly choose q elements from the labeled subset L ($Q \subset L$) to have their labels changed to any of the other classes chosen randomly for each sample, thus producing mislabeled nodes. These mislabeled subsets are created with increasing sizes, $q/l = \{0.00, 0.05, 0.10, \dots\}$. Each configuration is repeated at least 50 times, with different samples in the labeled and in the mislabeled subsets, in order to obtain an average.

Figure 3a shows the performance comparison when the semi-supervised learning graph-based methods are applied to the Iris Data Set [30], which contains 4 features from 3 different types of iris plant. There are 150 samples (50 from each class), 40 samples are randomly chosen to compose the labeled subset. When all the samples in the labeled subset are correctly labeled, the proposed algorithm performs slightly better than the others. As the mislabeled subset increases, the performance of all algorithms decreases as expected, but the proposed method performance degrades less than the others, and its advantage becomes increasingly more visible with up to 45% of the labeled subset composed by mislabeled samples.



(a) (b)

Figure 3: (a) Classification error rate in the Iris data set with different mislabeled subset size; (b) Classification error rate in the Wine data set with different mislabeled subset size.

Finally, Figure 3b shows the performance comparison when the semi-supervised learning graph-based methods are applied to the Wine Data Set [30]. This data set results from a chemical analysis of 178 samples of wines grown in the same region in Italy,

but derived from three different cultivars (classes). The analysis determined the quantities of 13 constituents (attributes) found in these three types of wines. 40 samples are randomly chosen to compose the labeled subset. When all the samples in the labeled subset are correctly labeled, the proposed algorithm already performs a little better than the others. As the mislabeled subset increases, this difference becomes more notable because the classification error rates of the other algorithms increases more than the classification error rate of the proposed method. With up to 30% of mislabeled samples in the labeled subset, the proposed algorithm seems not to be affected at all by the wrong labels as it keeps a stable performance. With 35% of mislabeled samples and beyond, there is a decrease in the proposed method performance. However, it still manages to perform much better than LGC, LP, and LNP with up to 55% of mislabeled samples. When 50% of the labeled subset is composed by mislabeled samples, the proposed method still makes less than half of the amount of classification mistakes made by LGC, LP and LNP.

4. CONCLUSIONS

This paper proposes a new biologically inspired method for semi-supervised classification, which is specifically designed to handle data sets with mislabeled subsets. It uses teams of walking particles competing for network nodes. Each team corresponds to one of the classes label. Particles which belongs to the same team act cooperatively to spread the team label to unlabeled nodes, at the same time that they protect their neighborhood from invading teams. Teammates also work together to attack other nodes in order to raise their team domination level and take them over. In this way, a mislabeled node may have its label changed when the team which has its correct label first dominates the nodes around it, then attacks it, and finally takes it over, thus stopping wrong label propagation from this node.

Computer simulations results indicates that the proposed model is robust to the presence of mislabeled data. Analysis of the results indicate the presence of critical points in the performance curve as the mislabeled samples subset grows. These experiments also showed how these critical points are related to the network size and mixture. In the comparison against other representative semi-supervised graph-based methods, the proposed algorithm performed better than them when applied to real-world data sets with mislabeled samples. It performed even better than the original particle competition and cooperation method, showing that the features we introduced to handle data sets with mislabeled samples are effective.

In future works, we intend to expand the analysis to cover the impact of average node degree and other networks measures in the performance of the algorithm, as well as expanding the comparison to include more artificial and real-world data sets with mislabeled nodes.

5. ACKNOWLEDGMENT

This work was supported by the São Paulo State Research Foundation (FAPESP) and by the Brazilian National Research Council (CNPq).

REFERENCES

[1] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kauffman, second edition, 2005.

[2] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] E. Alpaydin. *Introduction to machine learning*. MIT Press, 2004.

[5] G. E. Hinton and T. J. Sejnowski. *Unsupervised Learning: Foundations of Neural Computation*. MIT Press, 1999.

[6] K. J. Cios, W. Pedrycz, R. W. Swiniarski and L. A. Kurgan. *Data Mining: A Knowledge Discovery Approach*. Springer, 2007.

[7] C. Aggarwal and P. Yu. "A Survey of Uncertain Data Algorithms and Applications". *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 5, pp. 609–623, May 2009.

[8] R. Wolff, K. Bhaduri and H. Kargupta. "A Generic Local Algorithm for Mining Data Streams in Large Distributed Systems". *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 4, pp. 465–478, April 2009.

[9] X. Zhu. "Semi-Supervised Learning Literature Survey". Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[10] O. Chapelle, B. Schölkopf and A. Zien, editors. *Semi-Supervised Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, 2006.

[11] S. Abney. *Semisupervised Learning for Computational Linguistics*. CRC Press, 2008.

[12] X. Zhu, Z. Ghahramani and J. Lafferty. "Semi-supervised learning using Gaussian fields and harmonic functions". In *Proceedings of the Twentieth International Conference on Machine Learning*, pp. 912–919, 2003.

[13] D. Zhou, O. Bousquet, T. N. Lal, J. Weston and B. Schölkopf. "Learning with local and global consistency". In *Advances in Neural Information Processing Systems*, volume 16, pp. 321–328. MIT Press, 2004.

[14] X. Zhu and Z. Ghahramani. "Learning from labeled and unlabeled data with label propagation". Technical Report CMU-CALD-02-107, Carnegie Mellon University, Pittsburgh, 2002.

[15] F. Wang and C. Zhang. "Label Propagation through Linear Neighborhoods". *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[16] A. Blum and S. Chawla. "Learning from labeled and unlabeled data using graph mincuts". In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 19–26, San Francisco, 2001. Morgan Kaufmann.

[17] M. Belkin, I. Matveeva and P. Niyogi. "Regularization and semisupervised learning on large graphs". In *Conference on Learning Theory*, pp. 624–638. Springer, 2004.

[18] M. Belkin, N. P. and V. Sindhwani. "On manifold regularization". In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTAT 2005)*, pp. 17–24, New Jersey, 2005. Society for Artificial Intelligence and Statistics.

[19] T. Joachims. "Transductive learning via spectral graph partitioning". In *Proceedings of International Conference on Machine Learning*, pp. 290–297. AAAI Press, 2003.

[20] D. K. Slonim. "Learning from Imperfect Data in Theory and Practice". Technical report, Cambridge, MA, USA, 1996.

[21] T. Krishnan. "Efficiency of learning with imperfect supervision". *Pattern Recognition*, vol. 21, no. 2, pp. 183–188, 1988.

[22] P. Hartono and S. Hashimoto. "Learning from imperfect data". *Applied Soft Computing*, vol. 7, no. 1, pp. 353–363, 2007.

[23] M.-R. Amini and P. Gallinari. "Semi-supervised learning with an imperfect supervisor". *Knowledge and Information Systems*, vol. 8, no. 4, pp. 385–413, 2005.

[24] M.-R. Amini and P. Gallinari. "Semi-supervised learning with explicit misclassification modeling". In *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*, pp. 555–560, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[25] M. G. Quiles, L. Zhao, R. L. Alonso and R. A. F. Romero. "Particle competition for complex network community detection". *Chaos*, vol. 18, no. 3, pp. 033107, 2008.

[26] F. A. Breve, L. Zhao, M. G. Quiles, W. Pedrycz and J. Liu. "Particle Competition and Cooperation in Networks for Semi-Supervised Learning". *IEEE Transactions on Knowledge and Data Engineering*, 2011. PrePrint - DOI: 10.1109/TKDE.2011.119.

[27] F. A. Breve, L. Zhao and M. G. Quiles. "Particle Competition in Complex Networks for Semi-supervised Classification". In *Complex (1)*, edited by J. Zhou, volume 4 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 163–174. Springer, 2009.

[28] F. A. Breve, L. Zhao and M. G. Quiles. "Semi-supervised learning from imperfect data through particle cooperation and competition". In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pp. 1 –8, july 2010.

[29] L. Danon, A. Díaz-Guilera, J. Duch and A. Arenas. "Comparing community structure identification". *Journal of Statistical Mechanics: Theory and Experiment*, vol. 9, pp. P09008, 2005.

[30] A. Frank and A. Asuncion. "UCI Machine Learning Repository", 2010.