Semi-Supervised Learning from Imperfect Data through Particle Cooperation and Competition

Fabricio A. Breve, Liang Zhao, and Marcos G. Quiles

Abstract— **In machine learning study, semi-supervised learning has received increasing interests in the last years. It is applied to classification problems where only a small portion of the data points is labeled. In these situations, the reliability of these labels is extremely important because it is common to have mislabeled samples in a data set and these may propagate their wrong labels to a large portion of the data set, resulting in major classification errors. In spite of its importance, wrong label propagation in semi-supervised learning has received little attention from researchers. In this paper we propose a particle walk semi-supervised learning method with both competitive and cooperative mechanisms. Then we study error propagation by applying the proposed model in modular networks. We show that the model is robust against mislabeled samples and it can produce good classification results even in the presence of considerable proportion of mislabeled data. Moreover, our numerical analysis uncover a critical point 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. These studies have practical importance to design secure and robust machine learning techniques.**

I. Introduction

The advances in computation and communication in the recent years have been quickly increasing our capacity of generating and collecting large amounts of data. However, most of this data is in its raw form, and it should be processed to extract useful information. The set of methods and techniques responsible for transforming data into potential useful information is called Data Mining. Data mining is a multidisciplinary field, drawing researchers from areas including statistics, machine learning, artificial intelligence, neural networks, pattern recognition, data management and databases, information retrieval, data visualization, and others [1], [2], [3], [4], [5].

One of the common tasks in Data Mining is *classification*, which is the process of arranging the data in pre-defined groups, commonly accomplished by machine learning algorithms. There are two major groups which embraces most machine learning algorithms: *supervised learning* and *unsupervised learning*. In supervised learning, algorithms learn from labeled examples, and thus a sufficient number of data points must be labeled, usually by a human expert in the specific domain of application. The labeled data is presented to the algorithm during its training phase before it is able to predict labels from new data points. On the other hand, in unsupervised learning, algorithms learn from only unlabeled examples and they try to identify how the data is organized. *Clustering* is a popular subgroup of unsupervised learning techniques, in which the data set is split into subsets (*clusters*), so that data points in the same clusters have some similarities [2], [6], [7], [8], [9], [10], [11], [12].

Supervised learning algorithms have been successfully applied to solve many practical problems, however, as mentioned before, nowadays data sets sizes are constantly increasing and labeling enough samples for the training process is expensive, time consuming, and it often requires the work of human experts. Therefore, it is common to encounter data sets in which only a small subset of samples is labeled and most data points are unlabeled. In these situations, the performance of supervised learning techniques are largely lowered because a lot of information carried by the unlabeled data are simply ignored. On the other hand, unsupervised learning methods have problems as well because they cannot take advantage of the available label information. In order to make use of both labeled and unlabeled data, a new class of machine learning algorithms, called *semi-supervised learning*, arose. These algorithms combine a few labeled data points with a large amount of unlabeled data in order to produce better classifiers with less human effort [13], [14], [15]. Semi-supervised learning includes some generative models [16], [17], cluster-and-label techniques [18], [19], co-training and tri-training techniques [20], [21], [22], [23], low-density separation models [24], graph-based methods, like Mincut [25], Local and Global Consistency [26], label propagation techniques [27], [28] and others. In the last years, most research in semi-supervised learning has been in graph-based methods [14], however most of them are similar and they may be seen as regularization framework [13], differing only in the particular choice of the loss function and the regularizer [25], [26], [29], [30], [31], [32].

In both supervised and semi-supervised learning algorithms, the quality of the training data is very important. We know that humans can easily compensate for imperfect data in their learning process, and animals can learn from conditioning even when they are rewarded inconsistently. But that is not the case for machine learning systems, in which fault-tolerance is usually difficult to be implemented. Most algorithms assume that the input label information is completely reliable, however, in practice mislabeled samples are common in data sets. This problem is usually refereed as *learning from imperfect data*, *learning from imperfect*

Fabricio A. Breve and Liang Zhao are with the Department of Computer Sciences, Institute of Mathematics and Computer Science (ICMC), University of São Paulo (USP), Av. Trabalhador São-carlense, 400, 13560-970, São Carlos, SP, Brazil (phone: +55 16 3373-9713; e-mail: fabricio, zhao@icmc.usp.br).

Marcos G. Quiles is with the Department of Science and Technology (DCT), Federal University of São Paulo (Unifesp), São José dos Campos, Sp, Brazil, (phone: +55 12 3942-5568; email: quiles@unifesp.br).

978-1-4244-8126-2/10/$26.00

teacher, and sometimes *learning from probabilistic teacher*. It is a significant issue in supervised learning, but the situation gets more critical in semi-supervised learning since there is few labeled data in this case, and errors (wrong labels) are easier to be propagated to a large portion of the data set. Besides its importance and vast influence on classification, error propagation or semi-supervised learning from imperfect data has received little attention from researchers and there are only a few recent works on this subject [33], [34], [35], [36], [37].

Recently, a biologically inspired clustering algorithm which uses a particle walking and competition approach was developed to detect communities in networks [38]. Particles compete with each other in order to possess nodes of the network, and at the end they will be naturally confined within a cluster. This method was later extended to realize semi-supervised learning [39]. The extended version basically introduces static nodes corresponding to the labeled samples, and each particle represents a class of the problem. In order to allow label propagation, particles are periodically reset to one of their corresponding labeled samples. That algorithm provides classification results similar to some well known methods, but with lower order of computational complexity.

In this paper, we present a new particle walking approach to perform semi-supervised learning with some interesting features. Firstly, inspired from social behavior of animals and collective sports, we use multiple particles, which are organized in teams, to represent each class. A cooperative mechanism is introduced among teammates, and at the same time, competition occurs between different teams. Another novelty is that here each particle has a single node as home, and they keep track of the distances from their home to other nodes, in a way that they will prioritize domination of their respective neighborhoods, help their teammates with their neighborhoods, and eventually try to invade opponent's territories. Finally, we introduce a new labeling process, together with the new particle dynamics, providing not only an efficient and effective semi-supervised learning algorithm, but also an algorithm that has a high level of robustness to mislabeled samples, preventing error propagation. We have also conducted numerical study of the algorithm performance as we increase the proportion of mislabeled samples in networks with different sizes and mixture levels, discovering the existence of critical points in the performance curve and their relationship with these network features.

This paper is organized as follows. The proposed model is described in section 2. In Section 3, simulation results by using different network configurations are presented. Finally, Section 4 concludes the paper.

II. Model Description

In this section, we introduce the particle competition and cooperation based semi-supervised learning algorithm. 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 with 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. The initial configuration is shown in Subsection II-A. At each iteration of the algorithm, each particle will choose a neighbor node to visit by using a random-deterministic rule, which is presented in Subsection II-C. The chosen node is called *target node*, and the particle which chooses the target node will increase its team domination level on this node, at the same time that it will decrease other teams domination levels on it. Each particle also has a strength level, which lowers or raises according to its team domination level on the target node. The rules to update nodes strength and particles domination levels are presented in Subsection II-B. At the end of the iterative process, the unlabeled nodes are labeled according to the rules described in Subsection II-D.

A. Initial Configuration

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.

Graph-based methods define a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, with $\mathbf{V} = \{v_1, v_2, \ldots, v_n\}$, where each node v_i corresponds to a sample x_i. The edges in E are defined accordingly to the similarity between nodes, i.e., there will be an edge between each pair of nodes v_i and v_j if they are similar enough according to any chosen distance measure and threshold.

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. From now on, each node v_i will be called the *home node* of its corresponding particle ρ_i. Particles change their position through time, and they keep track of the distance between their actual position and their home node. Particles representing samples with the same class labels will act like a team, collaborating with each other and competing with particles from other teams.

The system has two different kind of dynamics: particle dynamics and node dynamics. Each particle ρ_j holds two variables: $\rho_j^{\omega}(t)$ and $\rho_{\mathbf{j}}^{\mathbf{d}}(\mathbf{t})$. The first one, $\rho_j^{\omega}(t) \in [0, 1]$, is the particle strength, which indicates how much the particle can change nodes levels at time t. The second variable 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 particle home node v_j and the node v_i.

Each node v_i has one vector variable $\mathbf{v_i^{\omega}}(\mathbf{t}) = \{v_i^{\omega_1}(t), v_i^{\omega_2}(t), \ldots, v_i^{\omega_c}(t)\}$, 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:

$$\sum_{\ell=1}^{c} v_i^{\omega_\ell} = 1. \quad (1)$$

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:

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

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

$$\rho_j^\omega(0) = 1. \quad (3)$$

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

$$\rho_j^{d_i}(t) = \begin{cases} 0 & \text{if } \; i = j \\ n-1 & \text{if } \; i \neq j \end{cases}. \quad (4)$$

At each iteration, each particle will calculate the distance between its home node and the its target node, and update its table if necessary.

B. Node and Particle Dynamics

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. However, this rule apply only to unlabeled nodes, because labeled nodes have their domination levels fixed. Therefore, for each node selected as target v_i, the domination level $v_i^{\omega_\ell}(t)$ is updated as follows:

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

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). Labeled nodes have their domination levels v_i^ω always fixed, as defined by the third case in (5).

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:

$$\rho_j^\omega(t+1) = \rho_j^\omega(t) + \Delta_\rho(v_i^{\omega_\ell}(t+1) - \rho_j^\omega(t)), \quad (6)$$

where $0 < \Delta_\rho \leq 1$ is a parameter to control the amplitude of the particle strength change, lower values will cause slow changes and higher values will lead to quick changes, 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 approximate 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 their home nodes, 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 distance table $\rho_j^{d_k}(t)$ at each iteration t as follows:

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

where $\rho_j^{d_i}(t)$ and $\rho_j^{d_k}(t)$ are the distances to its home node 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, 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 its home node. Every time a particle chooses a target node, it will check its distance in its 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.

Finally, 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.

C. Random Walk and Deterministic Walk

The system includes two different rules to determine how particles choose their target nodes. Those are *random walk* and *deterministic walk*. When using *random walk*, a particle randomly chooses any of its neighbors to target with no concern about domination levels or distances. This rule is important for exploration and acquisition of new nodes. On the other hand, when performing *deterministic walk* particles have preference to target nodes closer to their home nodes and nodes in which their team have higher domination level. This rule is important for territory defense and team behavior. Particles must balance the choice of those rules in order to achieve an equilibrium between exploratory and defensive behavior.

In this manner, in *Random walk* the particle ρ_j chooses its target node v_i randomly among their neighbors:

$$p(v_i|\rho_j) = \frac{W_{ki}}{\sum_{q=1}^{n} W_{qi}}, \tag{8}$$

where k is the index of the node being visited by particle ρ_j, thus $W_{ki} = 1$ if there is an edge between the current node and any node v_i, and $W_{ki} = 0$ otherwise.

On the other hand, in *deterministic movement* the particle chooses its target 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 its home node v_j as follows:

$$p(v_i|\rho_j) = \frac{W_{ki} v_i^{\omega_\ell} \frac{1}{(1+\rho_j^{d_i})^2}}{\sum_{q=1}^{n} W_{qi} v_i^{\omega_\ell} \frac{1}{(1+\rho_j^{d_i})^2}}, \tag{9}$$

and, once more, 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.

Each particle will choose a rule for each iteration with probability p_{det} of choosing deterministic walk and probability $1 - p_{\text{det}}$ of taking random walk, with $0 \leq p_{\text{det}} \leq 1$. Once the rule has been chosen, the target neighbor will be randomly chosen with probabilities defined by (8) or (9), respectively.

D. Labeling the Unlabeled Nodes

At a first glance, nodes' domination levels $\mathbf{v_i^\omega(t)}$ represent all the information we need in order to label the unlabeled nodes at the end of the process, as they indicate which team dominated each node. Although this is true in most common cases, the domination levels are very volatile under certain circumstances. For instance, when there are overlap nodes or mislabeled nodes, the dominating team may change frequently, and a snapshot of the domination levels in the last iteration may not reflect what happened during the whole system execution. In many cases, a mislabeled node implies that a particle will have its home node placed inside other team neighborhood. Therefore, it is likely that this particle will not be able to compete for its closest neighbors, it will probably be expelled and migrate to their teammates neighborhood. However, sometimes this particle will try to return home, as the deterministic rule takes the distance table in consideration. A snapshot taken in one of these occasions will not reflect what happened most of the time.

In order to avoid this problem, a new vector variable called *accumulated domination levels* is introduced. It represents temporal averaged domination levels for each team at each node. The accumulated domination levels starts from zero and increases every time a node is chosen as target by a particle taking the random walk rule. The particle will raise its team accumulated domination level, but it will not change the other team levels. There is no upper limit and the increase is always proportional to the particle strength. Notice that accumulated domination levels are not changed when deterministic walk rule is chosen, otherwise it would amplify the visiting advantage of dominating particle, which is not desirable. The new variable presentation was delayed until now because it has no effect on system dynamics, it is actually used only to register and take advantage of temporal information from the system in order to label the unlabeled nodes at the end.

Mathematically, accumulated domination levels are defined as $\mathbf{v_i^\lambda}$, which is a vector $\mathbf{v_i^\lambda(t)} = \{v_i^{\lambda_1}(t), v_i^{\lambda_2}(t), \ldots, v_i^{\lambda_c}(t)\}$ of the same size as L, and $v_i^{\lambda_\ell}(t) \in [0 \quad \infty]$ holds accumulated domination level by team ℓ over node v_i. At each iteration, for each selected node v_i (in *random movement*), the accumulated domination level $v_i^{\lambda_\ell}(t)$ is updated as follows::

$$v_i^{\lambda_\ell}(t+1) = v_i^{\lambda_\ell}(t) + \rho_j^{\omega}(t) \tag{10}$$

where ℓ is the class label of particle ρ_j.

After the last iteration, each unlabeled node is labeled according to its highest accumulated domination level:

$$y_i = \arg\max_{\ell} v_i^{\omega_\ell}(\infty). \tag{11}$$

III. Computer Simulations

In this section, we present simulation results to show the effectiveness and robustness of our method in the presence of mislabeled data. The following parameters were held constant in all simulations in this paper: $\Delta_v = 0.1$, $\Delta_\rho = 1.0$, and $p_{\text{det}} = 0.5$. These values were obtained by empirical optimization and produce good results in the type of networks studied here.

The networks are generated by using the method proposed by [40]. 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 our first set of experiments, we generate networks with increasing number of nodes, $n = \{256, 512, 768, 1024\}$, divided equally into 4 classes, with $\langle k\rangle = n/8$ and $z_{out}/\langle k\rangle = 0.250$, 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 l elements ($L \subset N$) to be labeled, while the others are presented to the algorithm without labels, the labeled set size is tested with increasing values, $l/n = \{0.05, 0.10, 0.15, \ldots, 0.50\}$. 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. This mislabeled set is also tested with increasing values, $q/l = \{0.00, 0.05, 0.10, \ldots, 1.00\}$. So, there is a total of 1680 different configurations and each of them is repeated 50 times to take an average. The results are presented in Figures 1a, 1b, 1c and 1d for $n = 256$, 512 , 768 and 1024 respectively.

In our second set of experiments, we generate networks with increasing mixture, $z_{out}/\langle k\rangle = \{0.125, 0.250, 0.375, 0.500\}$, while the network size and average node degree are fixed, $n = 512$ and $\langle k\rangle = 64$. Again, all elements are equally divided into 4 classes. The labeled subset and the mislabeled subset are selected the same way as in the previous experiment. Again, there is a total of 1680 different configurations and each of them is repeated 50 times to take an average. The results are presented in Figures 2a, 2b, 2c and 2d for $z_{out}/\langle k\rangle = 0.125$, 0.250, 0.375 and 0.500 respectively.

In all cases, 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 all the figures, showing the robustness of our 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. This interesting phenomenon is due to the competition mechanism of the algorithm, where the correct label propagation and wrong label propagation compete with each other. If there is not many wrong label samples, they may lose the competition and consequently the wrong label propagation can be impeded. We also notice that the plateau gets larger as the network grows and smaller as the mixture of the network increases. This means 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 the network size increases, the angles formed in the critical points become narrow. On the other hand, as the network mixture increases, the angles formed in the critical points become wider.

In order to better understand the behavior of the critical points, we run other two sets of experiments with more points and repetitions. So, in the third set of simulations, we generate networks with 16 different sizes $n = \{64, 128, 192, 256, \ldots, 1024\}$, divided equally into 4 classes, average node degree constant, $\langle k\rangle = n/8$, and fixed mixture, $z_{out}/\langle k\rangle = 0.25$. The labeled subset is set to $l/n = 0.1$ (10% labeled nodes is a typical semi-supervised learning problem), and the mislabeled subset is again variable, $q/l = \{0.00, 0.02, 0.04, \ldots, 1.00\}$, but now with a smaller step size (0.02). So, now 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 3 and by analyzing them we have the confirmation that as the network size increases ($n \to +\infty$), the performance curve with variable mislabeled samples set size becomes rougher and the critical points have a narrower angle.

The forth set of experiments is similar to the third one, but now we fixed network size to $n = 512$, divided equally into 4 classes, we also 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 4, and by analyzing them we have the confirmation that as the network mixture increases ($z_{out}/\langle k\rangle \to 1$), the performance curve with variable mislabeled samples set size becomes smoother and the critical points 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.

We are not concerned about 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 closer it is to the second critical point, the higher is the robustness of the algorithm in that specific configuration. In our experiments, the first critical point was closer to the second as the classes become more well separated and as the network size grows, which means the algorithm gets more robust in those cases, as we expected. Figures 5 and 6 show the first critical points extracted from the experiments shown in Figures 3 and 4, respectively.

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 Tables I and II we can observe the maximum size of the mislabeled subset that still produces

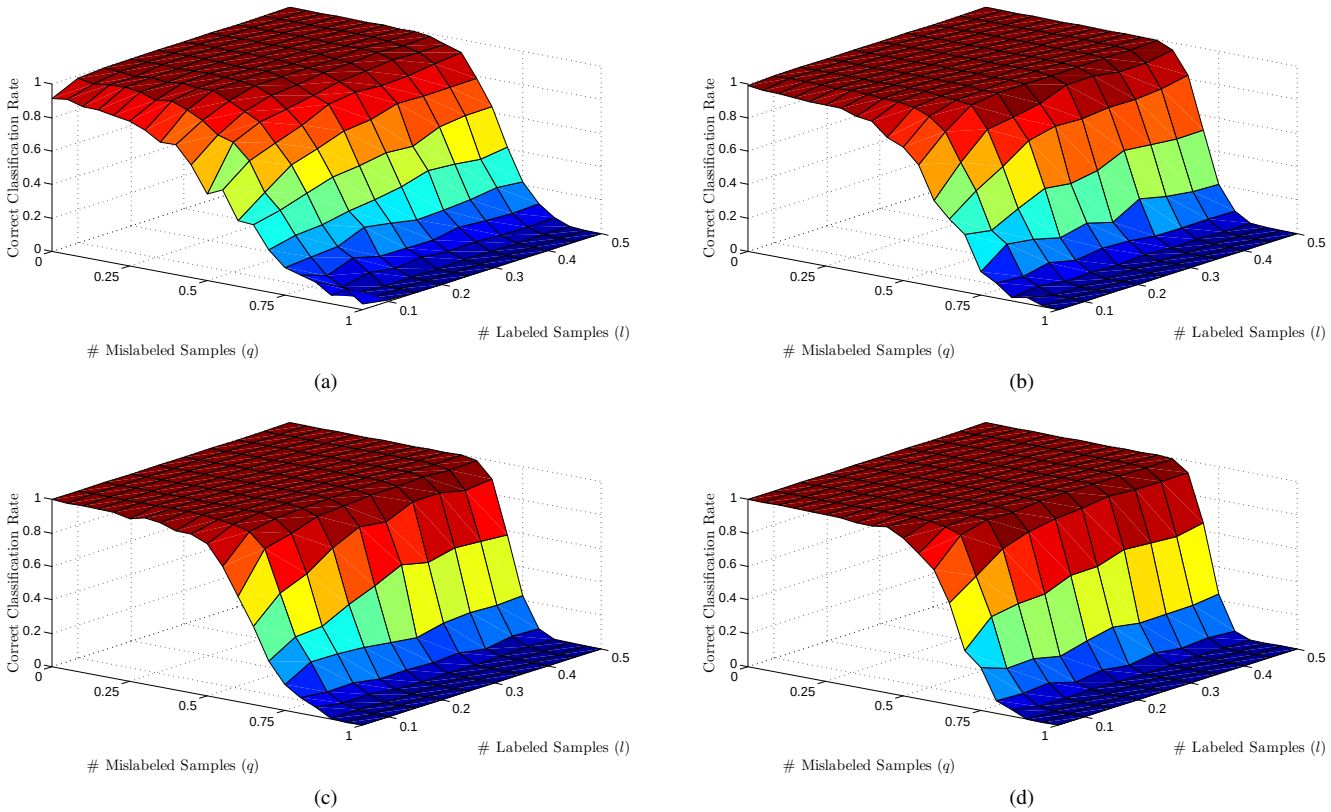

Fig. 1: Correct Classification Rate with varying labeled subset sizes and mislabeled subset sizes, network mixture $z_{out}/k = 0.250$ and different network sizes: (a) $n = 256$, (b) $n = 512$, (c) $n = 768$, and (d) $n = 1024$

TABLE I: Maximum mislabeled subset size for different network sizes (n). $z_{out}/\langle k \rangle = 0.250$, $l/n = 0.1$.

	Correct Classification Rate			Correct Classification Rate	
n	$> 90\%$	$> 80\%$	n	$> 90\%$	$> 80\%$
64	-	8%	576	52%	58%
128	10%	26%	640	54%	58%
192	26%	44%	704	56%	60%
256	40%	48%	768	56%	60%
320	44%	48%	832	56%	60%
384	48%	52%	896	58%	62%
448	48%	56%	960	58%	62%
512	52%	56%	1024	60%	64%

TABLE II: Maximum mislabeled subset size for different network mixtures ($z_{out}/\langle k \rangle$). $n = 512$, $l = 64$.

	Correct Classification Rate			Correct Classification Rate	
$z_{out}/\langle k \rangle$	$> 90\%$	$> 80\%$	$z_{out}/\langle k \rangle$	$> 90\%$	$> 80\%$
0.0313	52%	56%	0.2813	50%	54%
0.0625	52%	56%	0.3125	48%	54%
0.0938	50%	58%	0.3438	48%	52%
0.1250	52%	56%	0.3750	46%	50%
0.1563	52%	56%	0.4063	40%	48%
0.1875	52%	58%	0.4375	32%	44%
0.2188	50%	56%	0.4688	22%	40%
0.2500	52%	56%	0.5000	-	26%

good results (over 80% and 90% of correct classification rate) for different network sizes and mixtures, respectively.

IV. Conclusions

This paper proposes a biologically inspired method for semi-supervised classification using teams of walking particles competing for network nodes. Particles corresponding to labeled data points spread their labels while they expand their domain by walking in a network, competing with particle from other teams while cooperating with their teammates.

Particles use a concept of home node in order to keep them around to dominate and protect neighboring nodes. Collaborative behavior takes place when particles from the same team work together to protect their neighborhood from being invaded by other teams. They also work together to attack other nodes in order to raise their team domination level and take over them. Moreover, when a node is away from its home node, teammates may be visiting and helping to protect its neighborhood.

The model dynamics together with the labeling scheme provide a natural way of preventing error propagation from mislabeled nodes. Computer simulation results indicates that the proposed model is robust to the presence of mislabeled data. Moreover, analysis of the results indicate the presence of critical points in the performance curve as the mislabeled samples subset grows. Additional experiments showed how these critical points are related to the network size and

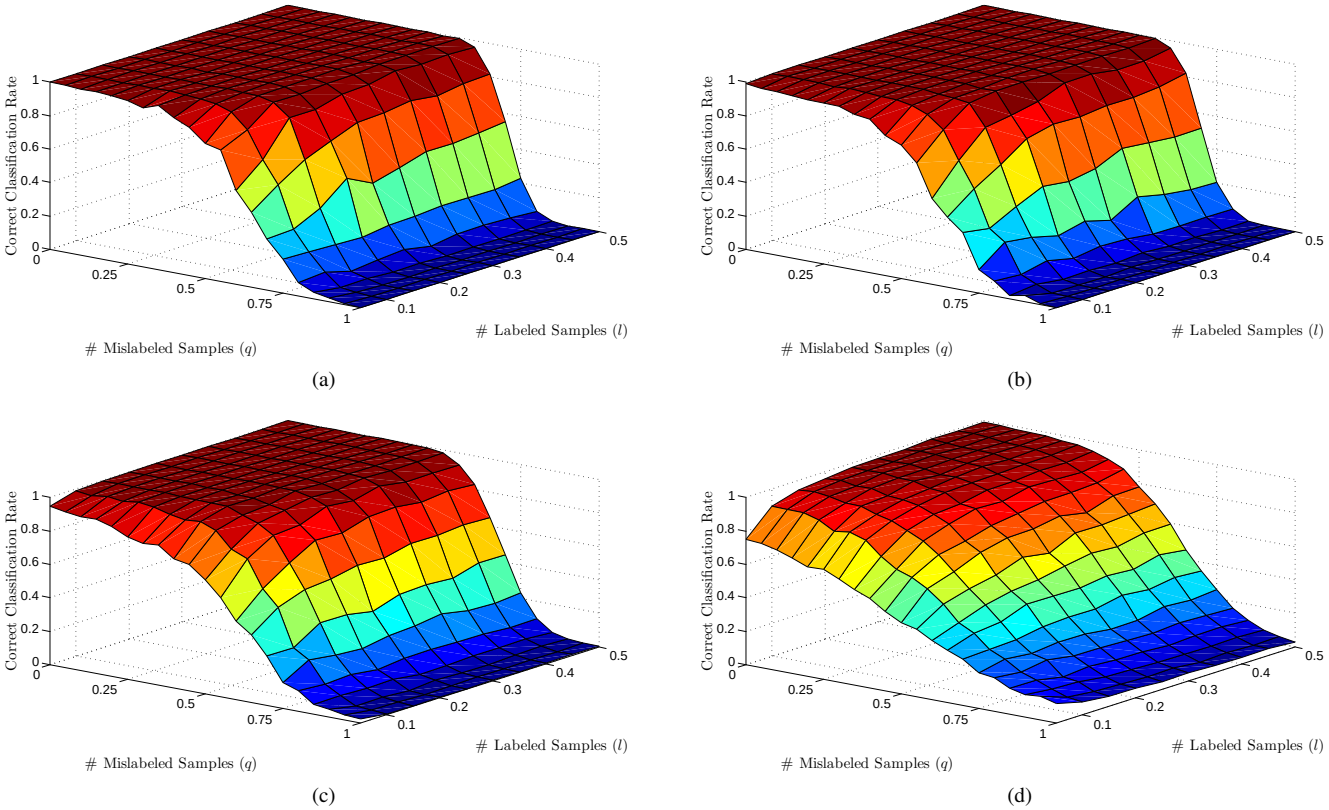

Fig. 2: Correct Classification Rate with varying labeled subset sizes and mislabeled subset sizes, network size $n = 512$ and different network mixtures: (a) $z_{out}/\langle k \rangle = 0.125$, (b) $z_{out}/\langle k \rangle = 0.250$, (c) $z_{out}/\langle k \rangle = 0.375$, and (d) $z_{out}/\langle k \rangle = 0.500$

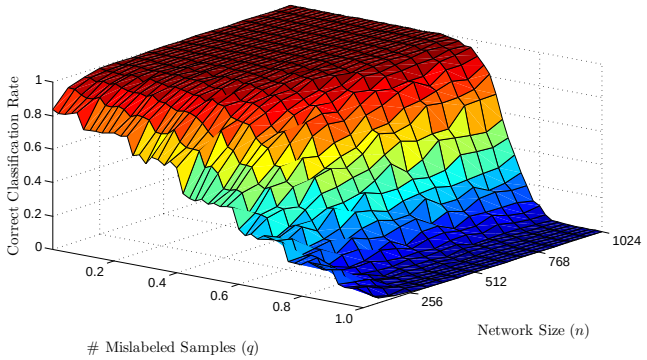

Fig. 3: Correct Classification Rate with different network sizes and mislabeled subset sizes. $z_{out}/\langle k \rangle = 0.25$, $l/n = 0.1$.

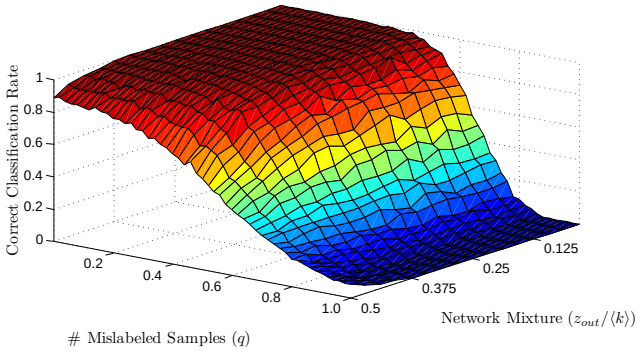

Fig. 4: Correct Classification Rate with different network mixtures and mislabeled subset sizes. $n = 512$, $l = 64$.

mixture. The results obtained in this paper are useful for designing secure and robust machine learning techniques.

ACKNOWLEDGMENT

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

REFERENCES

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. Morgan Kaufmann, 2006.

[2] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kauffman, 2005.

[3] D. J. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. MIT Press, 2001.

[4] S. M. Weiss and N. Indurkhya, *Predictive Data Mining: A Practical Guide*. Morgan Kaufmann, 1998.

[5] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson Addison Wesley, 2005.

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

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

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

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

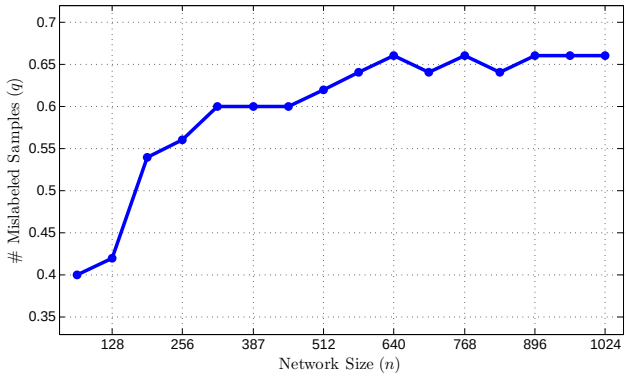

Fig. 5: The first critical point in the mislabeled samples curves with different network sizes. $z_{out}/\langle k\rangle = 512$, $l/n = 0.1$.

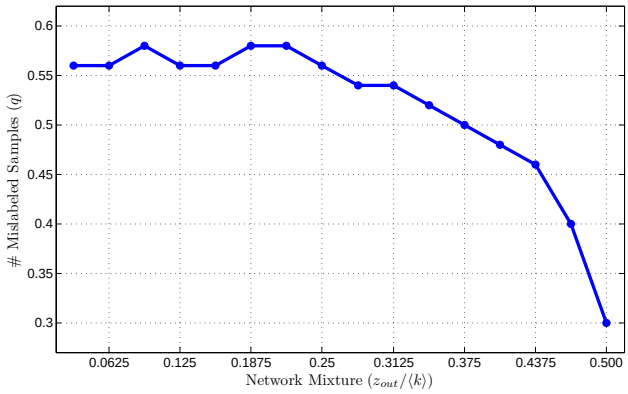

Fig. 6: The first critical point in the mislabeled samples curves with different network mixtures. $n = 512$, $l = 64$.

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

[11] 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.

[12] 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.

[13] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

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

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

[16] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," in *Machine Learning*, vol. 39, 2000, pp. 103–134.

[17] A. Fujino, N. Ueda, and K. Saito, "A hybrid generative/discriminative approach to semi-supervised classifier design," in *AAAI-05, Proceedings of the Twentieth National Conference on Artificial Intelligence*, 2005, pp. 764–769.

[18] A. Demiriz, K. P. Bennett, and M. J. Embrechts, "Semi-supervised clustering using genetic algorithms," in *Proceedings of Artificial Neural Networks in Engineering (ANNIE-99*. ASME Press, 1999, pp. 809–814.

[19] R. Dara, S. Kremer, and D. Stacey, "Clustering unlabeled data with soms improves classification of labeled real-world data," in *Proceedings of the World Congress on Computational Intelligence (WCCI)*, 2002, pp. 2237–2242.

[20] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *COLT: Proceedings of the Workshop on Computational Learning Theory*, 1998, pp. 92–100.

[21] T. M. Mitchell, "The role of unlabeled data in supervised learning," in *Proceedings of the Sixth International Colloquium on Cognitive Science*, 1999.

[22] Z.-H. Zhou and M. Li, "Tri-training: exploiting unlabeled data using three classifiers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1529–1541, Nov. 2005.

[23] ——, "Semisupervised regression with cotraining-style algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 11, pp. 1479–1493, Nov. 2007.

[24] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley-Interscience, September 2008.

[25] 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*, 2003, pp. 912–919.

[26] 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*, vol. 16. MIT Press, 2004, pp. 321–328.

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

[28] 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.

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

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

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

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

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

[34] D. K. Slonim, "Learning from imperfect data in theory and practice," Cambridge, MA, USA, Tech. Rep., 1996.

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

[36] M.-R. Amini and P. Gallinari, "Semi-supervised learning with an imperfect supervisor," *Knowl. Inf. Syst.*, vol. 8, no. 4, pp. 385–413, 2005.

[37] ——, "Semi-supervised learning with explicit misclassification modeling," in *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 555–560.

[38] 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, p. 033107, 2008.

[39] F. A. Breve, L. Zhao, and M. G. Quiles, "Particle competition in complex networks for semi-supervised classification," in *Complex (1)*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, J. Zhou, Ed., vol. 4. Springer, 2009, pp. 163–174.

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