Learning and Nonlinear Models - Journal of the Brazilian Computational Intelligence Society, Vol. 11, Iss. 1, pp. 3–10, 2013.

AN OSCILLATORY CORRELATION MODEL FOR SEMI-SUPERVISED CLASSIFICATION

Marcos G. Quiles, Márcio P. Basgalupp
Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
{quiles,basgalupp}@unifesp.br

Rodrigo C. Barros
Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo (USP)
rcbarros@icmc.usp.br

Abstract – This paper presents a new semi-supervised classification algorithm based on the oscillatory correlation theory. In this approach, the dataset is converted into a network whose nodes represent the samples and the edges represent the similarity among these samples. Each node in the network is modeled by an oscillator. The network clustering is given by the oscillators synchronization phenomenon, whereas the separation of oscillators that represent distinct clusters is induced by a global inhibitor. The previously labeled objects make use of the synchronization dynamics in order to propagate labels among their neighbors. Experiments performed with the proposed approach have shown promising results in a variety of datasets. It has shown to be capable of eventually outperforming traditional methods in the literature..

Keywords – Oscillatory correlation, synchronization, semi-supervised learning.

1. Introduction

The large volume of data generated through a variety of sources is an unprecedented fact in the history of sciences. Examples of data sources are computational simulations, internet text, biomedical signal and image analysis, genome data, just to name a few. The area of *machine learning* [1–3] arose with the goal of supporting domain specialists in the task of data analysis.

In the specialized machine learning literature, it is well-known that the process of learning through machine models can be either *supervised* or *unsupervised*. In an unsupervised learning environment, the main task is to cluster data objects according to a given similarity measure. This process is said to be *data-driven*, considering that no previous knowledge regarding data labels is necessary [1]. Supervised learning methods, on the other hand, aim at inducing concepts from previously labeled data objects, ie, objects whose labels are known *a priori*. Nevertheless, supervised learning requires a considerable amount of labeled objects in order to increase the chances of finding a satisfactory model for the problem at hand.

Generally speaking, the process of data acquisition is not computationally expensive. However, manually labeling objects may become a costly process depending on the problem domain. There are many cases in which this cost refers not only to computational power, but to the price of the project. For instance, the necessity of hiring domain specialists to perform the labeling process can make the data acquisition a costly task in terms of finances. In this sense, a new learning paradigm has emerged in machine learning, namely *semi-supervised learning* [4, 5], in which only a small number of labeled objects are available for building predictive models. Thus, one does not require every object to be labeled in order to build a classifier, which considerably reduces the cost of the process.

Several approaches have been proposed regarding semi-supervised learning. Among these approaches, network-based techniques (which is the topic explored in this paper) have received increased attention in the recent years [5–14]. The main feature of these techniques lies in the way data is represented: the network nodes represent the data objects whereas the edges represent the distances (similarities) among objects.

Most of the network-based techniques are, in essence, transductive, which means they aim at labeling the unlabeled objects within the training data. Hence, there is a clear need of a technique capable of dealing with new, unseen objects, and also of incorporating them dynamically to the dataset. In [15], a new training process was proposed for semi-supervised learning, based on the neuron synchronization within a network. In such a model, each network node (object) is represented by an Integrate& Fire neuron, and the synchronization phenomenon among neuron groups is used to propagate labels to data objects. Even though the results presented in [15] were considered quite satisfactory, there are three main limitations with that technique:

- The neurons are described by differential equations, thus requiring numeric integration;
- The approach does not include a stopping criterion;
- The network average degree required for a good performance is quite high.

In this paper, we propose a new semi-supervised approach based on the oscillatory correlation theory. More specifically, we present a fast label-propagation algorithm, which was extracted from LEGION [16] (*Locally Excitatory Globally Inhibitory*

Network). We show how the new approach solves the limitations observed in [15], besides being a new application of the oscillatory correlation theory.

The remaining of this paper is organized as follows. In Section 2 we describe the LEGION model, which serves as basis to this work. Section 3 describes the process of network construction from a dataset, as well as the proposed algorithm. Experiments and results are presented in Section 4. Finally, Section 5 provides our main conclusions and also points to future work directions.

2. LEGION model

According to [17], investigations regarding the cerebral functions and perception organization indicate the existence of a temporal correlation mechanism that acts like a representation structure (temporal coding). The temporal correlation theory defines that an object is represented by the temporal correlation of the activity of spatially-distributed neural cells, which represent different features from the same object. Conversely, neural cells that represent features from distinct objects do not have correlated activities.

A natural way of performing temporal correlation is through the use of oscillators [16]. Each oscillator may represent a set of features (color, orientation, movement, depth, etc. [18]) in such a way that each segment (object) is represented by a set of oscillators with synchronized activities, whereas distinct segments are represented by a set of desynchronized oscillators. This particular temporal correlation scenario is called theory of *oscillatory correlation* [16]. Features are represented by oscillators and the integration problem is solved by the synchrony or dissynchrony among neural oscillators [18].

Two main aspects can be established regarding the oscillatory correlation [16]. First, the synchronization of oscillators that represent features from a same object should be established. Second, the desynchronization among groups of distinct oscillators should be defined. One of the greatest challenge in the development of oscillatory correlation models is in the simultaneous implementation of these antagonistic mechanisms [18].

A locally-coupled oscillators network architecture called LEGION (*Locally Excitatory Globally Inhibitory Oscillator Networks*) was proposed in [16] with the goal of establishing a formal theory of oscillatory correlation. The LEGION model has been applied in several tasks, such as the numeric and analytic study of the coupled oscillators dynamics [16, 19], image segmentation [19, 20], and object selection [21, 22], just to mention a few.

The LEGION architecture, in its basic form, is comprised of three main elements: neural oscillators, local excitatory couplings, and a global inhibitor. The local excitatory couplings are used to synchronize the oscillators' groups, representing each of the objects in the visual scene. The global inhibitor, on the other hand, is used to generate the desynchronization among the oscillators' groups. Hence, the network owns a local cooperation mechanism and a global competition strategy, which are two requirements for implementing the oscillatory correlation. In its original form, the LEGION model [16, 23] is comprised of a network of **relaxation** oscillators, in which each oscillator is comprised of an excitatory variable x_i, and an inhibitory variable y_i, defined by Equations (1a) and (1b).

$$\dot{x}_i = 3x_i - x_i^3 + 2 - y_i + \mathcal{I}_i + S_i + \rho \tag{1a}$$

$$\dot{y}_i = \epsilon(\alpha(1 + \tanh(x_i/\beta)) - y_i) \tag{1b}$$

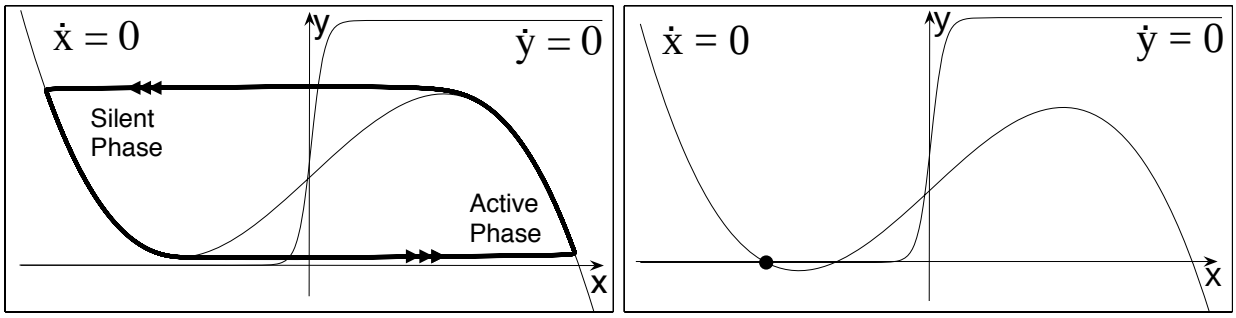
where $\mathcal{I}_i$ is the stimulus that is external to oscillator i, S_i is the coupling with the remaining network oscillators, ϵ is a positive constant with a small value, and ρ is the noise signal whose goal is to verify the robustness of the model and to support the desynchronization of distinct patterns. If $\mathcal{I}_i$ is defined as a constant and the terms S_i and ρ are eliminated, Equations (1a) and (1b) form a typical relaxation oscillator.

Figure 1 presents the nullclines [1] and the oscillator's trajectory is defined by Equations (1a) and (1b), where the nullcline of x is a cubic function and the nullcline of y is a sigmoid function. If there is an external stimulus received by an oscillator, $\mathcal{I}_i > 0$, the x and y nullclines of Equations (1a) and (1b) intersect in a single point. In this case, the oscillator state is called *active* and a stable limit cycle dynamic is observed (Figure 1a). The oscillator's periodic orbit alternates between two well-defined phases, namely *active phase* and silence phase, which correspond to the phases with large and small x values, respectively (see Figure 1a). That means that by observing only the x value it is possible to verify whether an oscillator is triggered or not. This information is used in the derivation of the algorithm presented in Section 3.2. The transition between the two phases is much faster than the internal movement observed in each phase, and thus the transition between the active and silence phases is called *jumping*. Parameter α controls how much the oscillator stays in each phase. When there is not external stimulus received by the oscillator, $\mathcal{I}_i < 0$, the two nullclines from Equations (1a) and (1b) intersect in a stable fixed point in the left side of the cubic function (see Figure 1b). In this case, the oscillator does not generate a periodic orbit and no oscillation is observed whatsoever. However, the oscillator may be induced to oscillate through stimuli received when coupling with neighbor oscillators. In this particular state, the oscillator is called *excitable*. Parameter β controls the sigmoid inclination. Usually, β is set to a near-zero value, so the sigmoid function approximates a step function [16].

The coupling of the system represented by S_i is defined by the following equation:

$$S_i = \sum_{k \in N_i} w_{ik} H(x_k - \theta_x) - w_z H(z - \theta_z) \tag{2}$$

[1] They define the curves when $\dot{x} = 0$ and $\dot{y} = 0$.



(a) (b)

Figure 1: Dynamics of a relaxation oscillator. (a) Behavior of the oscillator in the triggered state. The trajectory is defined by a limit cycle represented by the bold curve, and the arrows indicate the movement direction. (b) Behavior of an oscillator in the excitable state. In this case, a fixed stable point is observed, indicated by a dot in the low left side of the figure.

where w_{ik} is defined as the coupling strength between oscillators i and k, N_i defines the interaction neighborhood of oscillator i, represented by those oscillators that are directly connected to i. Parameter θ_x is a threshold that indicates whether or not an oscillator may affect its neighbors. Usually, θ_x is selected between the active and silence phases. Variable w_z defines the strength of connection between oscillator i and the global inhibitor defined by z, whereas θ_z is a threshold. $H(v)$ is the Heaviside function in which $H(v) = 1$ if $v \geq 0$, and $H(v) = 0$ otherwise.

The dynamics of the global inhibitor z is defined by Equation (3).

$$\dot{z} = \phi(\sigma_\infty - z) \tag{3}$$

where ϕ controls the velocity in which the global inhibitor reacts to signs that come from the oscillators. $\sigma_\infty \equiv 0$ if $x_i < \theta_x$ for all i. Conversely, $\sigma_\infty \equiv 1$ if at least one oscillator $x_i \geq \theta_x$. In this case, if at least one oscillator exceeds the threshold, z is stimulated and approximates 1, thus acting like an inhibitor as z exceeds θ_z (Equation (2)).

Generally speaking, the network dynamics can be summarized as follows. When an oscillator is active, it triggers the global inhibitor, which in turn sends an inhibiting signal for all the network, as described by Equations (1)-(3). Moreover, the active oscillator also propagates its signal to its respective neighbors, which carry on with the process. Hence, the network presents a strategy of local cooperation, whereas the inhibitor is responsible for the global competition. The global inhibitor may be seen as a particular attention mechanism that, once active, the remaining oscillators become inactive. Nevertheless, in its original form, the model presents a limitation regarding the maximum number of segments that can be obtained by the network [16, 20]. Bearing this limitation in mind, [20] propose an image segmentation algorithm extracted from differential equations from the LEGION model, whose essential properties are kept intact. The model proposed in this paper makes use of the same approach used in [20], *i.e.*, we rely on a fast algorithm for simulating the behavior of the network instead of the numeric integration of the original oscillators.

3. Proposed Model

The LEGION model considers a local-cooperation neighborhood, Equation (2), which is responsible for the synchronization phenomenon. The generation of this neighborhood (network generation) among the dataset objects represents the first step of the classification process. Once the network is defined, the neural oscillators dynamics is started and the synchronization between neighbor objects can be observed. At the same time, as described by the oscillatory correlation theory, well-separated objects (represented by node groups) have their trajectory desynchronized by the global inhibitor. The label propagation (classification) becomes a natural consequence of the synchronization between neighbor oscillators, in which a previously labeled node (oscillator) propagates its label to its synchronized nodes.

The next sections present the network generation process and the algorithm extracted from the LEGION model.

3.1. Network Generation and Model Dynamics

Let a dataset be represented by a set of attribute vectors $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$ and a set of labels $\mathcal{L} = \{1, 2, ...c\}$, where n and c represent the number of objects and classes, respectively. In the semi-supervised context, the dataset $\mathcal{S}$ is divided into two subsets: the subset of previously-labeled objects $\mathcal{S}_l = \{\mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_l\}$ and the subset of unlabeled objects $\mathcal{S}_u = \{\mathbf{s}_{l+1}, \mathbf{s}_{l+2}, ..., \mathbf{s}_n\}$. Each object $\mathbf{s}_i$ belonging to the set $\mathcal{S}_l$ has an associated label $l_i \in \mathcal{L}$.

The network generation $G = (V, E)$ for a given dataset $\mathcal{S}$ consists in representing each object $\mathbf{s}_i$ as a node $v_i \in V$, and also in creating the set of edges E that represent the similarity among objects. Here, two approaches were used to create the adjacency matrix: θ_w-cut, where connection happens within the radius defined by θ_w and a kNN approach where each vertex is connected to its k-nearest-neighbor.

The similarity function is defined by the following equation:

$$w_{ij} = \exp\left(\frac{-d(\mathbf{s}_i, \mathbf{s}_j)}{\sigma^2}\right) \tag{4}$$

where $d()$ defines a distance function and σ controls the Gaussian width. By using the similarity function defined above, the adjacency matrix E for the θ_wNN e kNN approaches are generated respectivelly by the following equations:

$$e_{ij} = \begin{cases} w_{ij} & \text{if } w_{ij} \geq \theta_w \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and

$$e_{ij} = \begin{cases} w_{ij} & \text{if } j \in \ k\text{NN}_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where θ_w is a threshold and k is the number of neighbors taken into account.

To avoid the existence of disconnected nodes in the θ_wNN approach, a post-processing step that connects nodes to the network is employed. This step guarantees that every disconnected node is connected to its nearest neighbor within the network (see Equation (7)).

$$e_{ik} = w_{ik} \ \mid \ \arg\max_k w_{ik} \tag{7}$$

Once generated, the network defines the local-cooperation mechanism of the model. Having in mind that the edge generation process is conducted by a similarity function, it is easy to see that objects close to each other in a given region of the attribute space will be strongly connected. Conversely, objects belonging to different regions in the attribute space will be connected by a weak edge, or maybe will not be connected at all. Thus, as explained in Section 2, neighbor oscillators tend to synchronize, whereas the global inhibitor breaks the synchrony between dissimilar groups.

This behavior is strongly related to the task of unsupervised data clustering, since no information regarding labels has been taken into account so far. Therefore, two requirements should be considered in order to adapt the LEGION model to a semi-supervised setting: i) the network should properly represent the dataset, *i.e.*, strongly connected objects should supposedly belong to the same class; and ii) as the network is generated, if two neurons are oscillating, there is a high probability that the objects associated to these neurons belong to the same class. It is worth mentioning that, for a successful semi-supervised classification application, some assumptions should be taken into consideration. For instance, the proposed model relies on the smoothing and clustering assumptions [4, 5, 24].

The classification process occurs alongside the clustering process. Every time a given oscillator i that represents a previously-labeled object is triggered ($x_i > \theta_x$), the state of each neuron j belonging to the neighborhood of i is modified according to Equation (2). If neuron j, stimulated by neuron i, reaches the LK region and is triggered (see Figure 1a), the object represented by neuron j automatically inherits the label belonging to neuron i. In other words, the synchronization process is responsible for both the clustering process and the label propagation task within the network.

The next section presents the algorithm extracted from the LEGION model for semi-supervised data classification.

3.2. Proposed Algorithm

The numeric integration of a network with many oscillators is quite costly in terms of computational resources, as previously stated in Section 2. Hence, we follow the same approach proposed in [20], in which an algorithm is extracted from the equations that model the LEGION network. This algorithm keeps intact the main existing properties of the model's numeric simulation, and it further allows networks with many oscillators to be simulated within acceptable execution time. Moreover, the algorithm explicitly defines a stopping criterion, which is an important feature that cannot be directly derived from the original model based in differential equations.

For the implementation of this algorithm, the following simplifications were performed:

- If all oscillators are inactive (silence phase, near the LK region), the oscillator nearest to LK is selected for being triggered and become active;
- If the full signal received by an oscillator (considering the neighbors and global inhibitor signals) exceeds θ_x, this oscillator becomes active in a single step;
- If no inactive oscillator is able to be triggered, all active oscillators return to the silence phase in a single step. This scenario occurs when all oscillators that were stimulated by the same pattern have already been triggered.

Note that the x value is enough for indicating whether or not an oscillator is in the active phase (triggered). Algorithm 1 defines the semi-supervised classification model proposed in this work.

The silence phase can be described as being the region between the leftmost point of x and the region delimited by LK (see Figure 1a). The network oscillators are initially set to the silence phase according to the following methodology: first, the region

Algorithm 1 Label Propagation Algorithm

```
Set Parameters
Define Connections (Section 3.1)
Set Oscillators to the Silence Phase
Select Neuron for Pulsing (Alg. 2)
repeat
  for all neuron i do
    if x_i(t) = RK e z(t) > z(t − 1) then
      x_i(t) = x_i(t − 1) (oscillator stays in the active phase)
    elseif x_i(t) = RK e z(t) < z(t − 1) then
      x_i(t) = LC (oscillator returns to the silence phase)
      z(t) = z(t − 1)
      if (z(t+1)=0) (there are no more neurons in the active phase) then
        Select Next Neuron (Alg. 2)
      end-if
    else
      Compute coupling S_i regarding neuron i (Eq. (2))
      if S_i > 0 (Excited Neuron) then
        x_i(t + 1) = RK
        z(t + 1) = z(t) + 1
        l_i = ClasseAtiva (the object associated to neuron i is labeled with the active class - see Alg. 2)
      else
        x_i(t + 1) = x_i(t) (neuron stays in the silence phase)
      end-if
    end-if
  end-for
until All neurons have pulsed
```

that represents the silence phase is divided into two parts according to the x values. Every oscillator representing an unlabeled object is randomly initialized in the leftmost part of the silence phase (away from LK), whereas the oscillators representing labeled objects are positioned randomly in the right side of the silence phase (near LK). This approach allows the oscillators that represent labeled objects to be the first to become active. Algorithm 2 presents the strategy for selecting neurons to pulse.

Algorithm 2 Selection Algorithm.

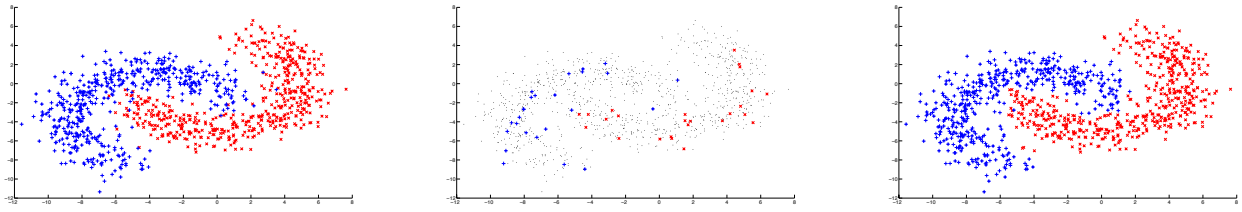
```
Find j among every oscillator in the silence
phase, such that x_j(t) ≥ x_k(t)∀k
x_j(t + 1) = RK
Z(t + 1) = 1
if j ∈ L then
  ActiveClass = l_j (Variable ActiveClass represents the class of the selected neuron, which will be propagated to the neurons that are
  excited by it)
else
  ActiveClass = l_i (arg min_{i∈L} d(i, j)) (If the selected neuron is not associated to an object belonging to the labeled set s_j ∉ S_l, variable
  ActiveClass receives the label associated to the nearest labeled neuron i to j)
end-if
for all k in the silence phase do
  x_k(t + 1) = x_k(t) + (LK − x_j(t))
end-for
```

By analyzing the equations that describe the LEGION model, several parameters need to be set. However, these parameters present well-established values and do not need to be modified during the experiments. Hence, parameter σ, which is used to configure the network that represents the dataset, is the only parameter that needs to be tuned specifically for each dataset.

4. Computer Experiments

This section presents the experiments performed with our model and their respective results. To implement our model, the *iGraph* package was taken into account [25]. This package provides several functions for generating and manipulating networks.

The experiments were divided in two parts. First, a toy dataset was used to illustrate the classification process. Second, several datasets proposed and analyzed in [5] were considered. The use of those datasets was quite interesting, as it allowed for a direct comparison of the model with several other techniques discussed in [5]. It is worth pointing out that all LEGION parameters were held constant: $\theta_x = 0.0$, $\theta_z = 0.5$ e $W_z = 0.2$. The threshold $\theta_w = 0.1$ (Eq. (5)) was also held constant. The parameter σ (Eq. (4)) was always adjusted to obtain a network with average degree $\langle k \rangle \approx 3$ for the θ_w-cut approach. To the kNN approach, k was held constant at 10. These values were obtained through empirical study from diverse datasets.



(a) (b) (c)

Figure 2: Experiment with synthetic data set. a) original dataset, b) dataset with only 5 % of labeled data, c) classification result.

The first experiment was performing with a synthetic dataset containing 1000 samples equally divided into two classes. Figure 2(a) shows the original dataset. Only 5% of the labels associated with randomly selected samples are preserved (see Figure 2(b)). The result of the classification is illustrated in Figure 2(c), which achieved a classification rate of approximately 96% for a set of 200 runs. For each run, the subset of pre-labeled samples was generated randomly. Assuming the illustrative purpose of this first experiment, only the θ_w-cut approach was applied to generate the graph.

The following experiments were conducted with datasets studied in [5][2]. Table 1 provides a brief description of the main characteristics of these datasets.

Table 1: Datasets from [5]

Dataset	Class	Dimension	Samples	Type
g241c	2	241	1500	
g241n	2	241	1500	synthetic
Digit1	2	241	1500	synthetic
USPS	2	241	1500	real
COIL	6	241	1500	real
BCI	2	117	400	real
TEXT	2	11960	1500	real

In order to compare the results to those published in [5] and [15], the same methodology was adopted in this work, *i.e.*, the experiments were divided into two batteries: first, only 10 examples of each dataset were labeled, while in the second battery 100 samples of each dataset were labeled. For each dataset, we used the same 12 *folds* (divisions) provided by [5], allowing a fair comparison of methods. The results obtained by the proposed model were compared with the best, the worst and the average of the results published in [5] and [15]. It is important to mention that in [5] the standard deviation is omitted and only the average of the 12 runs is presented. Thus, no further statistical analysis could be performed to see if the difference is significant or not between the models.

Table 2: Results with 10 labeled examples in $\theta_w$cut networks. The table presents the average precision and the standard deviation obtained in the proposed model, the average accuracy of the model presented in [15] and average, the best and the worst results of the techniques studied in [5].

Dataset	Our model	[15]	Average	Worst	Best
g241c	55.95(3.30)%	-	59.54%	50.41%	77.24%
g241n	56.78(2.89)%	-	55.84%	49.37%	81.27%
Digit1	76.42(5.79)%	-	85.05%	69.40%	94.56%
USPS	80.51(3.55)%	80.65%	80.88%	74.64%	83.93%
COIL	35.50(4.41)%	36.83%	35.83%	32.50%	45.46%
BCI	51.26(2.83)%	51.03%	50.69%	49.64%	52.05%
TEXT	60.88(5.16)%	79.77%	62.73%	54.68%	72.85%

Tables 2 and 3 depicts the results obtained by our method using the θ_w-cut approach. Those tables also present the results shown obtained in [5] and [15]. Analyzing the results, we can see that the proposed model produced results close to the average of [5]. Similar results were obtained with the kNN approach, as shown in Tables 4 and 5. Although the proposed model has not achieved better results than the best techniques, the results are compared to the average of several techniques. It means, the model is able to perform semi-supervised classification in different contexts. Furthermore, as mentioned at the beginning of the section, the average degree of the network required for application of the technique is much lower than the model proposed

[2]Available online at `http://www.kyb.tuebingen.mpg.de/ssl-book`

Table 3: Results with 100 labeled examples in θ_w-cut networks. The table presents the average precision and the standard deviation obtained in the proposed model, the average accuracy of the model presented in [15] and average, the best and the worst results of the techniques studied in [5].

Dataset	Our model	[15]	Average	Worst	Best
g241c	59.72(2.25)%	-	72.91%	55.95%	86.51%
g241n	62.51(1.61)%	-	72.32%	56.79%	95.05%
Digit1	94.16(1.38)%	-	96.21%	93.85%	97.56%
USPS	91.37(0.97)%	84.77%	93.13%	90.23%	95.32%
COIL	81.02(1.83)%	81.49%	80.05%	71.29%	90.39%
BCI	55.17(1.77)%	64.07%	58.36%	52.11%	66.75%
TEXT	69.10(0.98)%	94.34%	73.88%	67.17%	76.91%

in [15], $\langle k \rangle \approx 3$ for the θ_w-cut approach and $k = 10$ for the kNN approach versus $\langle k \rangle \approx 25$. The networks taken into account by our model are more sparse than those treated in [15], thus , the manipulation of the network becomes more efficient. This fact associated with the algorithm which does not require numerical integration, allows the treatment of larger datasets.

Table 4: Results with 10 labeled examples in KNN networks. The table presents the average precision and the standard deviation obtained in the proposed model, the average accuracy of the model presented in [15] and average, the best and the worst results of the techniques studied in [5].

Dataset	Our model	[15]	Average	Worst	Best
g241c	55.84(3.39)%	-	59.54%	50.41%	77.24%
g241n	56.79(2.90)%	-	55.84%	49.37%	81.27%
Digit1	75.69(6.83)%	-	85.05%	69.40%	94.56%
USPS	80.53(3.55)%	80.65%	80.88%	74.64%	83.93%
COIL	34.93(3.86)%	36.83%	35.83%	32.50%	45.46%
BCI	51.09(2.39)%	51.03%	50.69%	49.64%	52.05%
TEXT	60.55(4.94)%	79.77%	62.73%	54.68%	72.85%

Table 5: Results with 100 labeled examples in KNN networks. The table presents the average precision and the standard deviation obtained in the proposed model, the average accuracy of the model presented in [15] and average, the best and the worst results of the techniques studied in [5].

Dataset	Our model	[15]	Average	Worst	Best
g241c	59.62(2.39)%	-	72.91%	55.95%	86.51%
g241n	62.65(1.67)%	-	72.32%	56.79%	95.05%
Digit1	93.90(1.35)%	-	96.21%	93.85%	97.56%
USPS	91.21(0.93)%	80.65%	93.13%	90.23%	95.32%
COIL	80.90(1.98)%	36.83%	80.05%	71.29%	90.39%
BCI	55.06(1.74)%	51.03%	58.36%	52.11%	66.75%
TEXT	69.27(1.07)%	79.77%	73.88%	67.17%	76.91%

Another advantage of our model compared to the one proposed in [15] is the existence of a stopping criterion. In this model, simply wait until all the oscillators have pulsed once.

5. Conclusions

This paper proposes a new model for semi-supervised classification based on the oscillatory correlation theory. Amidst the main contributions, the following can be highlighted: i) a new application of the oscillatory correlation theory, ii) the proposed model showed competitive classification accuracy in comparison to state-of-art techniques, iii) the model has solved some of the limitations observed in [15], and iv) in contrast to the current graph-based models, the proposed model is dynamic, it does not require a new learning process when a new instance is inserted into the dataset. In these situations, the proposed model requires only a new iterations in order to acomodate the new samples.

As future work, we intend to investigate in deep the reasons why the model achieved low accuracies for some datasets considered in this work. Moreover, we also intend to apply the model in dynamic datasets and specific areas of problems, such as interactive segmentation of images.

Acknowledgments

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

REFERENCES

[1] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[2] R. O. Duda, P. E. Hart and D. G. Stork. *Pattern Classification*. Wiley, segunda ed. edition, 2000.

[3] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[4] X. Zhu. "Semi-supervised learning literature survey". Technical Report 1530, Computer Science, University of Wisconsin-Madison, 2008.

[5] O. Chapelle, B. Schölkopf and A. Zien. *Semi-supervised learning*. The MIT Press, 2006.

[6] X. Zhu. "Semi-supervised learning with graphs". Ph.D. thesis, School of Computer Science, Carnegie Mellon University, 2005.

[7] F. Wang and C. Zhang. "Label propagation through linear neighborhoods". *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55–67, 2008.

[8] Y. Xu, J. S. Dyer and A. B. Owen. "Empirical Stationary Correlations for Semi-Supervised Learning on Graphs". *The Annals of Applied Statistics*, vol. 4, no. 2, pp. 589–614, 2010.

[9] F. Nie, S. Xiang, Y. Liu and C. Zhang. "A general graph-based semi-supervised learning with novel class discovery". *NEURAL COMPUTING & APPLICATIONS*, vol. 19, no. 4, pp. 549–555, 2010.

[10] J. Gui, S.-L. Wang and Y.-K. Lei. "Multi-step dimensionality reduction and semi-supervised graph-based tumor classification using gene expression data". *Artificial Intelligence in Medicine*, vol. 50, pp. 181–191, 2010.

[11] M. Fan, N. Gu, H. Qiao and B. Zhang. "Sparse regularization for semi-supervised classification". *Pattern Recognition*, vol. 44, pp. 1777–1784, 2011.

[12] W. Liu, J. He and S.-F. Chan. "Large Graph Construction for Scalable Semi-Supervised Learning". In *Proceedings of the 27th International Conference on Machine Learning (ICML'2010)*, pp. 1–8, Haifa, Israel, 2010.

[13] Y. Wang, X. Xu, H. Zhao and Z. Hua. "Semi-supervised learning based on nearest neighbor rule and cut edges". *Knowledge-Based Systems*, vol. 23, pp. 547–554, 2010.

[14] 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*, vol. DOI: 10.1109/TKDE.2011.119, pp. 1–16, 2012.

[15] M. G. Quiles, L. Zhao, F. A. Breve and A. Rocha. "Label Propagation Through Neuronal Synchrony". In *The 2010 International Joint Conference on Neural Networks (IJCNN'2010)*, pp. 2517–2524, Barcelona, Espanha, 2010.

[16] D. Terman and D. Wang. "Global competition and local cooperation in a network of neural oscillators". *Physica D*, vol. 81, pp. 148–176, 1995.

[17] C. von der Malsburg. "The Correlation Theory of Brain Function". Internal Report 81-2, Department of Neurobiology, Max-Planck-Institute for Biophysical Chemistry, Göttingen, Germany, 1981.

[18] D. Wang. "The time dimension for scene analysis". *IEEE Transactions on Neural Networks*, vol. 16, pp. 1401–1426, 2005.

[19] S. R. Campbell, D. Wang and C. Jayaprakash. "Synchrony and desynchrony in integrate-and-fire oscillators". *Neural Computation*, vol. 11, pp. 1595–1619, 1999.

[20] D. Wang and D. Terman. "Image segmentation based on oscillatory correlation". *Neural Computation*, vol. 9, pp. 805–836, 1997.

[21] D. Wang. "Object selection based on oscillatory correlation". *Neural Networks*, vol. 12, pp. 579–592, 1999.

[22] M. G. Quiles, D. Wang, L. Zhao and R. A. F. Romero. "Selecting salient objects in real scenes: An oscillatory correlation model". *Neural Network*, vol. 24, pp. 54–64, 2011.

[23] D. Wang and D. Terman. "Locally excitatory globally inhibitory oscillator networks". *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 283–286, 1995.

[24] 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 16*, volume 16, pp. 321–328. MIT Press, 2004.

[25] G. Csárdi and T. Nepusz. "The igraph software package for complex network research". *InterJournal*, vol. Complex Systems, pp. 1695, 2006.