Complex network component unfolding using a particle competition technique

Paulo Roberto Urio

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Paulo Roberto Urio

Complex network component unfolding using a particle competition technique

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Zhao Liang

USP – São Carlos
July 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

U76c Urio, Paulo Roberto
 Complex Network Component Unfolding using a
 Particle Competition Technique / Paulo Roberto
 Urio; orientador Zhao Liang. -- São Carlos, 2017.
 75 p.

 Dissertação (Mestrado - Programa de Pós-Graduação
 em Ciências de Computação e Matemática
 Computacional) -- Instituto de Ciências Matemáticas
 e de Computação, Universidade de São Paulo, 2017.

 1. Semi-supervised learning. 2. Complex
 networks. 3. Machine learning. 4. Data clustering.
 5. Community detection. I. Liang, Zhao, orient. II.
 Título.

Paulo Roberto Urio

Desdobramento de componentes de redes complexas usando técnica de competição de partículas

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Zhao Liang

**USP – São Carlos
Julho de 2017**

ACKNOWLEDGEMENTS

I convey my heartfelt gratitude to all the people who have helped and inspired me during my graduate studies at ICMC-USP.

I would like to thank my professor, Dr. Zhao Liang, for his insights, advice, patience, dedication, and for keeping me on the line while extending, at the same time, freedom to drive decisions.

To my colleague and friend, Filipe Verri, with whom significant work has been done, I am also grateful. I like to believe that, while working side by side, we have inaugurated a new form of communication marked by endless brainstorming, treating each other as sounding boards for ideas.

The members of the Bio-inspired Computing Laboratory have been an invaluable resource for ideas and assistance, in particular, Daniel Cestari, for both his insights and patience.

Though far away, my family is my source of strength, and has always supported and advised me over the years toward a meaningful life. Mom, Dad, Angela, Ricardo, Vlade, Victor, Laura, and Quim, thank you.

Finally, I thank the university staff for being so helpful and am grateful for the financial support offered by the Coordination for the Improvement of Higher Education Personnel (CAPES) Brazilian government agency.

ABSTRACT

Urio, Paulo Roberto (2017). “Complex network component unfolding using a particle competition technique”. Master’s dissertation. São Carlos: Instituto de Ciências Matemáticas e de Computação (ICMC/USP), 75

This work applies complex network theory to the problem of semi-supervised and unsupervised learning in networks that are representations of multivariate datasets. Complex networks allow the use of nonlinear dynamical systems to represent behaviors according to the connectivity patterns of networks. Inspired by behavior observed in nature, such as competition for limited resources, dynamical system models can be employed to uncover the organizational structure of a network. In this dissertation, we develop a technique for classifying data represented as interaction networks. As part of the technique, we model a dynamical system inspired by the biological dynamics of resource competition. So far, similar methods have focused on vertices as the resource of competition. We introduce edges as the resource of competition. In doing so, the connectivity pattern of a network might be used not only in the dynamical system simulation but in the learning task as well.

Keywords: Semi-supervised learning. Complex networks. Machine learning. Data clustering. Community detection.

RESUMO

Urio, Paulo Roberto (2017). “Complex network component unfolding using a particle competition technique”. Dissertação de mestrado. São Carlos: Instituto de Ciências Matemáticas e de Computação (ICMC/USP), 75

Este trabalho aplica a teoria de redes complexas para o estudo de uma técnica aplicada ao problema de aprendizado semissupervisionado e não-supervisionado em redes, especificamente, aquelas que representam conjuntos de dados multivariados. Redes complexas permitem o emprego de sistemas dinâmicos não-lineares que podem apresentar comportamentos de acordo com os padrões de conectividade de redes. Inspirado pelos comportamentos observados na natureza, tais como a competição por recursos limitados, sistemas dinâmicos podem ser utilizados para revelar a estrutura da organização de uma rede. Nesta dissertação, desenvolve-se uma técnica aplicada ao problema de classificação de dados representados por redes de interação. Como parte da técnica, um sistema dinâmico inspirado na competição por recursos foi modelado. Métodos similares concentraram-se em vértices como o recurso da concorrência. Neste trabalho, introduziu-se arestas como o recurso-alvo da competição. Ao fazê-lo, utilizar-se-á o padrão de conectividade de uma rede tanto na simulação do sistema dinâmico, quanto na tarefa de aprendizado.

Palavras-chave: Aprendizado semissupervisionado. Redes complexas. Aprendizado de máquina. Agrupamento de dados. Detecção de comunidades.

CONTENTS

Acknowledgements	v
Abstract	vii
Resumo	ix
Contents	xi
1 Introduction	13
1.1 Machine learning	16
1.2 Objectives	19
1.3 Organization	20
2 Semi-supervised Learning by Edge Domination in Complex Networks	21
3 Semi-Supervised Classification by Particle Competition in Complex Network's Edges	29
4 Features of Edge-Centric Collective Dynamics in Machine Learning Tasks	49
5 Data Clustering with Edge Domination in Complex Networks	55
6 Conclusions	69
6.1 Future works	70
6.2 List of publications	70
Bibliography	73

INTRODUCTION

Had randomness not been brought aboard the fleet of computer science, many celebrated algorithms in the Deterministic Bay would be missing today. It brings efficiency in areas of communication, cryptography, sorting and searching, data structures, distributed computing, graph theory, and machine learning (Karp, 1991). The randomness in some of those areas differs from true randomness. True random events are unpredictable and present no structure or pattern. This property may be desired by applications in which randomness holds an essential role, such as in the cryptography processes of securing data. Apart from few delicate cases, randomness in computer modeling is rather controlled by exact processes; they generate random events that signify properties that are well defined. While they increase the uncertainty over choices, the sequence and the distribution of numbers are fully regulated; permitting scientists to control the reproducibility of experiments and to support analytical guarantees.

Though randomness has been powerful in the scientific world, it does not imply that randomness governs the real world. History reveals to us that people have been likely to call random any event they fail to guess most of the time. For example, since antiquity, dice rolling has been considered unpredictable. Today, we know that exact equations of motion describe the dynamics of a dice. In theory, the outcome is deterministic and predictable, but in practice, the model is so sensitive to the initial position that a random process predicts it better (Kapitaniak *et al.*, 2012). It does not only predict better, but it is also simpler and more intuitive: the exact equations of motion require accurate and precise details on the initial position of the dice and the friction of the surface on which the dice lands, whereas the random process is a single random variable for six possible outcomes. The intuition for the alternative random-process comes naturally: People perceive dice rolling as

a random action, just like any other activity that, to the naked eye, is too complicated or too difficult to pin down as regards its exact nature. Often, when intuition doesn't match the objective reality but approximates it adequately, the intuitive approach inspires a simpler model of reality.

Much of our scientific knowledge has been acquired through human perceptions of nature. Trigonometry, for instance, was inspired by observational astronomy, and calculus was inspired by experimental physics. Today, despite their original focuses, they solve several problems of different natures. In computing, neuroscience has inspired models of artificial neuronal network algorithms, which are based on rough intuitions of how biological neuronal networks work in reality. Although the architecture of those artificial networks undermines the actual complexity of our brain, the addition of parallelism and layering to a learning algorithm is itself a good idea. There would be no need to call them neuronal networks, nor to adopt terms like neurons, axons, soma, and dendrites. By the time scientists comprehend the underlying mechanisms, any mention to the original inspirations could be dropped for more meaningful names, but it is undeniable that the inspiration from neuroscience has helped scientists discover and explore those properties in learning algorithms. Such inspirations play key roles in the study of fresh ideas; admitting an open mind for the discovery of concepts that may bring new ways to solve problems.

Besides the growing diversity of approaches, the problems have changed as well. When scientists solve a problem, they either increase the details in the same problem or move to new problems. In both cases, as simple problems are solved, the newer problems are consequently more complex. For example, modern problems have aimed at understanding nontrivial interactions of the dynamics manifested in populations. They describe scenarios of aging, growth, disease propagation, evolution, and behavior of the Earth's fauna and flora. In earlier times, the models used to deal with few interacting elements and plain, simple interactions. Today, scientists deal with millions and billions of interacting elements and different patterns of interaction. The models have become more complex, often inspired by and targeted to the real complexity of biological systems.

A model's complexity varies according to the rules of interaction, which affect *how* individuals interact, and *when* they interact. The aspect of how they interact specifies the individuals' actions and any resulting consequences. It is specified at a local level, from the perspective of an individual, and can be either deterministic or stochastic. The aspect of when they interact specifies whether two or more individuals should interact with one another at a given moment. To tell when interactions occur, the individuals are set in regular or irregular structures of interaction. Regular structures are the simplest, with individuals interacting according to the connections of chains, grids, or Euclidean lattices (Wang and Chen, 2003). These structures add a little complexity to a model, enabling scientists to understand better how the dynamics of interaction manifests alone (Strogatz, 2001).

Irregular structures, on the other hand, increase the complexity of interactions. They arbitrarily diversify the interactions: Some parts of the structure may connect to few others, whereas other parts connect to several parts. Such irregular structures of interaction are effectively represented by graphs of discrete mathematics. In this context, the graph represents a *complex network*.

Together, the components of interaction and the rules of interaction constitute a *dynamical system*, which evolves in time. The interest lies in setting rules that act upon single and simple individuals, at the local level, that collectively emerge in dynamics at higher and a populational level.

The study of systems with the emergence of collective dynamics has led to the development of algorithms based on the behavior of birds, bees, ants, and bacterial colonies and many other forms of intelligence. The behaviors of cooperation and competition or a compound of the two often inspires the machinery for solutions. Competition, for instance, is the concept of two or more individuals striving to outdo one another in controlling a territory and resources, seducing a mate, or rising to a leadership position. Suppose competition is motivated by limited resources. In the case of dynamic systems, the resource may be the interaction structure where individuals are set, which is the analogue to the competition for territory. In an irregular structure, the individuals would compete to dominate parts of the complex network. Since the network is represented by a graph, the individuals are effectively competing for vertices or edges. In an application, an algorithm simulates the competition process in a complex network, which results in ownership of territory either by the individuals or by the group to which they belong. The algorithm thereafter translates the ownership to a solution of the problem or an explicative variable.

Often in models of collective dynamics, scientists trade perfection for efficiency by randomizing their models. Instead of dealing with a myriad of rules, with precise and ever increasing information, they focus on rules that choose randomly over the possible outcomes. In information theory, when Shannon (1948) introduced entropy, he handled randomness as a measure of the amount of information introduced. Similarly, a model of collective dynamics may use randomization to increase the information in the system. Uncertain choices help to diversify the dynamics. If the individuals should move on a network and explore territories, randomized individuals will further explore the network. An individual with greedy behavior, such as choosing the best neighborhood to move in could trap himself in the same region of the network every time, whereas randomization would allow him to escape and explore other places. Thus, randomness brings the advantages of simplifying models of collective dynamics, and can also increase efficiency.

With such collective dynamics in complex networks, I am interested in developing and studying simple and efficient models that solve the problems of machine learning.

1.1 Machine learning

Machine learning is a field dedicated to the study of methods by which a machine can learn from data by experience (Mitchell, 1997; Alpaydin, 2010; Flach, 2012). Three aspects characterize the techniques and problems in this field: what information is available, how the algorithm will learn from the data, and what the algorithm should learn from the data.

In the first aspect, the information available determines *a priori* of knowledge and the domain of the data. The data may be quantitative (discrete or continuous) or qualitative (nominal, ordinal, or binary). They are presented to an algorithm in structured formats (table or graph), in semi-structured formats (described with markup languages like XML or electronic file formats like email), or in unstructured formats (text, image, or video). The order of information may be relevant (in a time series or tagged with timestamps), or irrelevant (the order of words in a text may be ignored). Additionally, the data can be identified with the expected outcomes, which helps an algorithm to guide its learning process.

The second and third aspects, how and what an algorithm should learn, are explained by grouping problems in supervised learning, semi-supervised learning, and unsupervised learning.

In *supervised learning*, the problem is finding a relationship between a limited number of observations and their outcomes (Vapnik, 1995). The supervision manifests in two cases. Either the data are presented together with the expected outcome and the algorithm takes advantage of this knowledge, or the algorithm iteratively guesses and receive feedback from a supervisor with the correctness of the outcomes. Regarding the third aspect of a problem of machine learning, the goal of a supervised algorithm may be the construction of a regression model, estimation of density, or classification of data items (Flach, 2012).

In classification, the data are partitioned into classes, and the outcomes are the classes of each data item. The algorithm may be inductive or transductive. Inductive classifiers aim at building a model similar to the process that has generated the data. If the induced model correctly identifies the classes of unseen examples, then the algorithm has learned by correctly generalizing from a limited set of examples to a generalized model (Mitchell, 1997). Transductive classifiers, on the other hand, do not generalize nor perform an induction of a model. Instead of building a model that estimates classes for every possible unseen example, they classify only the examples at the learning process (Vapnik, 1998).

In *semi-supervised learning*, the problems relate closely to the problems of transductive learning in classification (Chapelle, Schölkopf, and Zien, 2006). In classification, the transductive step learns from labeled examples and classifies the unknown examples. In semi-supervised learning, the transductive step learns from the labeled examples, together with the unlabeled examples, prior to classifying the unlabeled examples used in learning. They take advantage of the unknown data during the learning step, but can not always

build models from the learning process. Hence, to classify any new data item requires the algorithm to repeat the learning process with all the data. In problems for this group, few labeled data and massive amounts of unlabeled data are available to the algorithm.

In *unsupervised learning*, an algorithm works without the knowledge of any expected outcome; even the number of possible outcomes might be unknown. In this case, the learning process lacks a gold standard to take as guidance. Scenarios with this setting have enormously increased with the recent evolution of data-acquisition and storage technologies. The problem is partitioning data into clusters, which is defined according to some presumption of the underlying structure of the data or by regions with a high density of examples in the feature space (Kononenko and Kukar, 2007). A cluster might be, for instance, defined by the spatial closeness of examples that are considered similar (Jain and Dubes, 1988; Hastie, Tibshirani, and Friedman, 2009), or by regions with high density of examples in the feature space (Ester *et al.*, 1996).

There are many ways to approach learning problems. Particularly, there has been a remarkable growth in the study of learning algorithms based on a graph representation of data. While there is little effort on utilizing graph-based algorithms for inductive learning (Zhu, 2005; Zhu and Lafferty, 2005), the transductive learning is the most common setting (Chapelle, Schölkopf, and Zien, 2006; Subramanya and Talukdar, 2014). Originally, graph representations have emerged through the assumption that the data lied on a low-dimensional manifold, and that a graph could approximate the manifold (Saul *et al.*, 2006). This approach allowed researchers to perform nonlinear dimensionality reduction of data (Tenenbaum, Silva, and Langford, 2000; Belkin and Niyogi, 2003), and to work with graphs on semi-supervised learning and unsupervised learning problems (Blum and Chawla, 2001; Szummer and Jaakkola, 2001; Zhu, Ghahramani, and Lafferty, 2003; Bousquet, Chapelle, and Hein, 2004; Karypis, Han, and Kumar, 1999).

In the context of network theory, the problem of *community detection* in complex networks relates to the problem of data clustering in unsupervised learning (Fortunato, 2010). Each data item is a vertex, and a connection denotes similarity between two items. This way, an algorithm that solves community detection can also solve data clustering. The community detection problem is finding a way that the vertices of a network can be grouped, so that there are more connections between vertices within a group than connections between different groups. Each group of vertices is a community, and all groups form a community structure. To solve the problem efficiently, an algorithm can partition vertices of a network into groups, maximizing some quality function of community structure (Newman, 2010). Different approaches like the simulation of collective dynamics may be explored to solve the problem.

1.1.1 Collective dynamics in machine learning and complex networks

Quiles *et al.* (2008) solved the problem of community detection with an algorithm inspired by the concept of competition for territory. Learning agents, called particles, walk a network while competing to conquer and maintain ownership of vertices. To find K communities, they put K particles performing synchronous walking. A particle only walks within its group of vertices, but it may try to move to rival vertices. With a fixed probability, the walk mechanism alternates between choosing the next vertex at random, and choosing to reinforce ownership of an owned vertex, whilst avoiding to return to the last vertex. After simulating several iterations, the resulting groups of vertices correspond to the community structure. Since the number of groups is unknown to the algorithm, they offer a strategy to find the ideal number. They collect the accuracy by matching the results with the actual community structure for a different number of communities and select the number of communities that yielded the best accuracy. This paper introduced the concept of particle competition in complex networks, studying the algorithm's ability to detect the structure of many artificial networks and real networks.

Silva and Zhao (2012b) provided a formal definition of an algorithm for community detection and data clustering with particle competition in complex networks. Inspired by the technique of Quiles *et al.*, they modeled a nonlinear, stochastic system for the dynamics of competition with a few modifications to the original model. Particles are not confined to their groups, and they walk on rival groups of vertices. Instead of a fixed probability to alternate the rule of walking, both terms of random walk and preferential walk are combined together into single rule. Because a particle can walk in rival territories, when a particle goes too far or stays too long at a rival territory, it teleports to an owned vertex. This mechanism is controlled by an energy level that increases and decreases when particles walk in owned or rival territories, in that order. An additional step that converts an instance of data clustering into network allows the technique to solve data clustering problems as well. They compared the technique's accuracy with many other techniques of data clustering and community detection.

With a few modifications, Silva and Zhao (2012a) proposed the formal model adapted to semi-supervised learning problems. The most important modification introduces a cooperation mechanism to the competition model. Instead of a single particle, a team of particles represents each group; characterizing a competition among teams of cooperating particles. To benefit from the knowledge of the number of possible classes, together with a small set of items labeled with their classes available in semi-supervised problems, the vertices representing labeled items have fixed ownership. With a rigorous mathematical model, computer simulations have reinforced the theoretical results. They presented

a parameter analysis, visualization of the results with toy datasets, and assessed the performance on real datasets.

In the three techniques, the competition always results in the ownership of vertices. The result is straightforward because ownership is directly translated to the problem result. Also, they all have a mechanism of a preferential walk, which controls the particles' decisions toward randomly walking or preferring to reinforce vertex.

In those works, the modeled system is also the system used for simulation; hence, if the dynamics determine the behavior of a single element, then each element is simulated separately. Such simulation has a computational complexity that depends on the size of the network and the number of interacting elements. Conversely, an outcome of this dissertation will be the contribution of an alternative modeling that allows a simulation with running time, independent of the number of interacting elements, which will be obtained in a deterministic manner.

1.2 Objectives

The goal of this dissertation is to develop machine learning techniques with dynamical processes on complex networks that will perform a *network component unfolding*.

The focus will be on a transductive semi-supervised learning model that employs a dynamical system in complex networks. In this dynamical system, particles compete for *edges* in a network. Subnetworks are generated with the edges grouped by class dominance. The specific goals are:

- To capture the connectivity pattern of the input data with competition of edges instead of vertices.
- To model a simpler walk mechanism with a variable number of active particles, replacing the former preferential walk mechanism by a mechanism of generation and absorption of particles.
- To offer an equivalent deterministic version of the stochastic competition model and enable faster running times and a competition technique that is independent of the number of particles.

An unsupervised learning model, with minor changes of the developed semi-supervised technique, will be designed and tested for feasibility to find community structures as solutions to data clustering problems.

1.3 Organization

The remainder of this dissertation is organized as follows. The semi-supervised technique is studied in Chapters 2 to 4. The unsupervised technique is studied in Chapter 5. Finally, Chapter 6 concludes this dissertation.

SEMI-SUPERVISED LEARNING BY EDGE DOMINATION IN COMPLEX NETWORKS

Paulo R. Urío, Filipe A. N. Verri, and Liang Zhao

Semi-supervised Learning by Edge Domination in Complex Networks

2015 11th International Conference on Natural Computation

Copyright © 2015 IEEE

Semi-supervised Learning by Edge Domination in Complex Networks

Paulo Roberto Urio, Filipe Alves Neto Verri

i p Z
Z v) JV

Liang Zhao

v v) n
0 0

Abstract—Bio-inspired dynamical processes are able to identify nonlinear features in data. We present a dynamical process model of particle competition in complex networks applied to transductive semi-supervised learning. Particles carry labels and compete for the domination of edges. The process results consist of sets of edges arranged by label dominance. The sets are analyzed as subnetworks for the data classification. Computer simulations show that this model can identify nonlinear data forms in both real and artificial data, including overlapping structure of data.

Index Terms—Complex networks, nonlinear dynamical systems, semi-supervised learning, particle competition.

INTRODUCTION

Machine learning is a branch of artificial intelligence that aims to develop algorithms that can learn from data and make predictions or decisions based on the learned information. In the context of complex networks, machine learning algorithms are used to analyze the structure and dynamics of the network, and to extract useful information from the data. This paper presents a semi-supervised learning algorithm based on edge domination in complex networks. The algorithm is inspired by bio-inspired dynamical processes, where particles compete for the domination of edges. The process results consist of sets of edges arranged by label dominance. The sets are analyzed as subnetworks for the data classification. Computer simulations show that this model can identify nonlinear data forms in both real and artificial data, including overlapping structure of data.

MODEL DESCRIPTION

Complex networks are characterized by a large number of nodes and edges, and by a high degree of connectivity. The nodes represent entities, and the edges represent interactions between them. In this paper, we consider a complex network with nodes labeled with classes. The goal is to learn a classification function that can assign new nodes to one of the classes based on their neighbors. The proposed algorithm is based on edge domination. Each node is associated with a particle, and the particles compete for the domination of edges. The process results consist of sets of edges arranged by label dominance. The sets are analyzed as subnetworks for the data classification. Computer simulations show that this model can identify nonlinear data forms in both real and artificial data, including overlapping structure of data.

This research was supported by the São Paulo State Research Foundation (FAPESP), the Coordination for the Improvement of Higher Education Personnel (CAPES), and the Brazilian National Research Council (CNPq).

B. An Illustrative Example

skmq V Wqmq mw a Pk k t v a t l a W W r t P P k a t k W P P t a m k ~
mXt~ Wq mS W r a i i v r t a q l P a W O k t t W W X t D i R R W ~
mXt~ W_ v q t W W q W W n P q t a S k t W r W z r t W j k V_ q P z ~
P q q m a k V a S P t a W q t W W P q t a S P W W W S a V W j W m W ~
a k t~ W W y a t W q P F a m a G R W q W P S W V_ W k V a S P t W ~
current edge dominance, x` W q t W W P i a W t~ W q m n m q t a m k ~
mX~ q W m k P q t a S i P t P r r W W q m v t~ W W_ W v q a k_ W ~
i P r a t t W q P F a W k Q m R P R a n i X P t n P q t a S W a W P R r m q R W W ~
a r a k w W q R W i z ~ n q m m m m q a q u k W W t~ W n j a k P k S W O ~
P R r m q P W P P t a S j i n W W n~ P R r m q R a t k P u O ~
k t t~ W a_ t t D r a P W t~ W_ v q W W W r m X t~ W W q P t a m k ~
a r ~ m x K O S P k W W_ R t~ W P q t a S i V g B P k V a x W q W ~
P R r m q R W W E n P q r a S P a r W P r t r~ q m t~ W W_ W i k V ~
V a K m S ~ P k t W W v q q W t~ W n j a k P k S W O ~ q q W k t W ~
W m j a k P k S W V P t x W t~ W q m n m q m X m W P q t a S i P r ~
V S S W r r X R i i i W v q m v t~ W W_ W O W m v q S W r P q W W ~
~ q W W W q t a S W r E_ W R a W q P i n W a q W W n P q t a S a i B W ~
S a d B P k V a e A t n S m j n W k r P n q W W R r m q R W j k t a S i W k O ~
T m P q t a S P r W W r q m p k W W_ W B W n j a k P k S W O ~ W ~
S i a W a W v p P i a x t q a R v m W W ~ S i P r r W r O ~

$$W \sim r m q j \quad P i \quad k \quad W \sim W \quad k \quad W \sim W \quad m j \quad n \quad W t \quad a \quad P a \quad m \quad P z \sim k \quad P j \quad a \quad S \quad P i \sim$$

$$0 \quad q \quad m \quad S \quad W \quad r \quad w \quad \bar{n} \quad \frac{c}{i} \quad \frac{c}{i} \quad \bar{n} \quad \{ \quad \pi \quad \{$$

$$\phi: \begin{cases} n_i^c(t+1) &= n_i^c(t) + \sum_j (n_{j \rightarrow i}^c(t+1) - n_{i \rightarrow j}^c(t+1)), \\ \delta_{ij}^c(t+1) &= \delta_{ij}^c(t) + n_{i \rightarrow j}^c(t+1). \end{cases}$$

TABLE I
TEST ERRORS (%) FOR SPLITS WITH 10 LABELED TRAINING POINTS

	g241c	g241d	DigitI	U S P S	C O I L	B C I	Text	Avg. Rank
I - N N	47.88	46.72	13.65	16.66	63.36	49.00	38.12	9.4
S V M	47.32	46.66	30.60	20.03	68.86	49.85	45.37	13.1
M V U + I - N N	47.15	45.56	14.42	23.34	62.62	47.95	45.32	9.1
L E M + I - N N	44.05	43.22	23.47	19.82	65.91	48.74	39.44	8.9
Q C + C M R	39.96	46.55	9.80	13.61	59.63	50.36	40.79	7.0
Discrete regularization	49.59	49.05	12.64	16.07	63.38	49.51	40.37	10.4
T S V M	24.71	50.08	17.77	25.20	67.50	49.15	31.21	9.9
C l u s t e r k e r n e l	48.28	42.05	18.73	19.41	67.32	48.31	42.72	10.0
L D S	28.85	50.63	15.63	17.57	61.90	49.27	27.15	8.0
L a p l a c i a n R L S	43.85	45.68	5.44	18.99	54.54	48.97	33.68	5.4
L G C	45.82	44.09	9.89	9.03	63.45	47.09	46.83	6.7
L P	42.61	41.93	11.31	14.83	55.82	46.37	49.53	5.3
L N P	47.82	46.24	8.58	17.87	55.50	47.65	41.06	7.0
Vertice Competition	41.17	43.51	8.10	15.69	54.18	48.00	34.84	3.7
Our method	46.30	46.29	5.86	14.77	59.99	48.78	36.04	6.0

TABLE II
TEST ERRORS (%) FOR SPLITS WITH 100 LABELED TRAINING POINTS

	g241c	g241d	DigitI	U S P S	C O I L	B C I	Text	Avg. Rank
I - N N	43.93	42.45	3.89	5.81	17.35	48.67	30.11	11.6
S V M	23.11	24.64	5.53	9.75	22.93	34.31	26.45	8.0
M V U + I - N N	43.01	38.20	2.83	6.50	28.71	47.89	32.83	10.6
L E M + I - N N	40.28	37.49	6.12	7.64	23.27	44.83	30.77	10.6
Q C + C M R	22.05	28.20	3.15	6.36	10.03	46.22	25.71	6.6
Discrete regularization	43.65	41.65	2.77	4.68	9.61	47.67	24.00	7.1
T S V M	18.46	22.42	6.15	9.77	25.80	33.25	24.52	7.6
C l u s t e r k e r n e l	13.49	4.95	3.79	9.68	21.99	35.17	34.28	7.3
L D S	18.04	23.74	3.46	4.96	13.72	43.97	23.15	5.1
L a p l a c i a n R L S	24.36	26.46	2.92	4.68	11.92	31.36	23.57	4.2
L G C	41.64	40.08	2.72	3.68	45.55	43.50	56.83	9.1
L P	30.39	29.22	3.05	6.98	11.14	42.69	40.79	8.1
L N P	44.13	38.30	3.27	17.22	11.01	46.22	38.45	11.2
Vertice Competition	21.41	25.85	3.11	4.82	10.94	41.57	27.92	5.0
Our method	41.16	39.77	2.37	5.59	15.11	45.44	28.43	7.9

Zmflvtwtwflfi sflymtfi mww-myclégnf ymflyclwtg wflsm
w-fflv mwfflclwmiéclwflfi flclrnflfi yclémrnfygtvt
mfims hsgtvsclymtslet-y t flvrncltmsvflmntswb-vymflsvstvl-my
mwyvclmlygtv clwfflcltymfktflrm gtv fclyclrn-wyflvncflfi
etss-smy fifyflyeymts3

Ifwtfmyfwmfclymwgcléftvgtvscsefl-ymfclwmiéclwmtt
mwclymflwmsmyfmsymnwfltsrn ymfivmtvyflvclyméfly flfis
ymflflvymeflws ymfw-éfly tvrw mcl fléflfis etswmfivflvflfi
éflrnmfletss-sméclémrnmyfly flfis tclmvtg flvymeflwa;d scl
éfl clgl tvclémrnflnyflvsclym fl3

Msymfg-y-vfl fl mrrngvymflms flwymclwmtflfi clsclyclftg
fi sclsmewmsflfi IfwRtvft flv- wy-fi msflsclymflsclymeclrn tvtflv
ymfite st.etsftflymymtsveflsclymflfi
 $\alpha \rightarrow \infty$)

REFERENCES

- [1] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.
- [2] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
- [3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.
- [4] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, Sep. 2006.
- [5] X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*, 2009, vol. 3.
- [6] A. Barrat, M. Barthélemy, and A. Vespignani, *Dynamical processes on complex networks*. New York, NY, USA: Cambridge University Press, 2008.
- [7] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang, "Complex networks: Structure and dynamics," *Physics Reports*, vol. 424, no. 4-5, pp. 175-308, Feb. 2006.
- [8] M. E. J. Newman, *Networks: An Introduction*, 1st ed. New York, NY, USA: Oxford University Press, 2010.
- [9] T. Cupertino and L. Zhao, "Using interacting forces to perform semi-supervised learning," in *Neural Networks (SBRN), 2012 Brazilian Symposium on*, Oct 2012, pp. 91-96.
- [10] R. A. Gueleri, T. H. Cupertino, A. C. de Carvalho, and L. Zhao, "A flocking-like technique to perform semi-supervised learning," in *2014 International Joint Conference on Neural Networks (IJCNN)*, 2014, pp. 1579-1586.
- [11] T. C. Silva and L. Zhao, "Network-based stochastic semisupervised learning," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 3, pp. 451-66, Mar. 2012.
- [12] —, "Detecting overlapping structures via network-based competitive learning," in *Proceedings of the International Joint Conference on Neural Networks*, 2012.
- [13] R. Duin, P. Juszczak, P. Paclík, E. Pekalska, D. de Ridder, D. Tax, and S. Verzakov, "PR-Tools 4.1, a matlab toolbox for pattern recognition," 2007.
- [14] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*, 2nd ed. Wiley-Interscience, Jan. 1999.
- [15] E. Estrada and N. Hatano, "Communicability in complex networks," p. 20, Jul. 2007.

SEMI-SUPERVISED CLASSIFICATION BY PARTICLE COMPETITION IN COMPLEX NETWORK'S EDGES

Paulo R. Urrio, Filipe A. N. Verri, and Liang Zhao

**Semi-Supervised Classification by Particle Competition in Complex Network's
Edges**

International Journal of Pattern Recognition and Artificial Intelligence

Copyright © 2016 World Scientific

Semi-Supervised Classification by Particle Competition in Complex Network's Edges

Paulo Roberto Urio* and Filipe Alves Neto Verri

Institute of Mathematical and Computer Sciences

University of São Paulo

São Carlos, SP, Brazil

**paulourio@gmail.com*

Liang Zhao

Ribeirão Preto School of Philosophy, Science and Literature

University of São Paulo

Ribeirão Preto, SP, Brazil

Received 13 November 2015

Accepted 24 May 2016

Published 22 July 2016

We present a biologically inspired model for transductive semi-supervised learning tasks. Specifically, this model consists of a set of particles that walk and compete in a complex network. From an input dataset, similarities between labeled and unlabeled data points derive a network representation. As particles walk the network, they compete to dominate the edges. Over the process, particles can become inactive, and, to compensate, labeled vertices will feed new particles to the system. Resulted from the model simulation, we analyze sets of edges arranged by their label dominance. Each set forms a subnetwork that is used to classify connected vertices. Our computer simulations on artificial and real datasets show that this technique can classify nonlinearly formed data and detect vertices of different classes in overlapping regions.

Keywords: Complex networks; nonlinear dynamical systems; particle competition; semi-supervised learning.

1. Introduction

Over the last few years, the world has been experiencing a data deluge. An impressive volume of data is produced in a short period, far more than people would be able to absorb and process mentally. This situation motivates us to build computer systems that can learn from an enormous volume of data. Consisted of a combination of computer science and statistics tools, the field of *machine learning* has emerged dedicated to the study of such computer systems.^{4,18}

Many machine learning techniques have been developed, and we can classify them by the available information in the problem dataset. In supervised learning problems,

the dataset consists of completely labeled data, while, in unsupervised learning problems, the dataset is formed by only unlabeled data.^{18,22} The third class of problems is the semi-supervised learning, in which a mixture of labeled and unlabeled data forms the dataset, usually with several unlabeled data and only a few labeled.^{8,35} This latter scenario of partially labeled data scenario is increasingly common in our lives with the recent data boom. Several man-hours of domain experts would be necessary to assign manually labels to a significant portion of the currently produced data. As a consequence, little-labeled information is made available.

In the recent years, many semi-supervised learning techniques have been developed, including generative models,²⁰ discriminative models,¹⁷ clustering and labeling techniques,²⁹ multi-training,^{5,33} low-density separation models,²⁸ and graph-based methods.^{9,24,32,34} In techniques based on a graph representation of datasets, a vertex is a single data instance, and it connects to other vertices according to a predefined affinity rule. Most of such techniques propagate labels (one label per class) in a graph using an optimization heuristic.^{2,3,21,34} Recent graph-based techniques have been using concepts from complex networks.

Complex networks are graphs that model natural phenomena with nontrivial topology.¹ They represent the topological and dynamical relationship among parts of a system.⁶ A *dynamical system* is a set of mathematical rules that control the evolution of a system. In the context of complex networks, these rules govern the interaction between adjacent vertices.^{1,19} Several studies explore the ability of dynamical processes to identify nonlinear features in data.^{10,13}

Cooperation and competition are concepts intrinsic to the nature of relationships, and they are readily understandable for us. Competition for limited resources, for instance, has originated the scheduling problems in computing, which are efficiently solved by optimization algorithms.^{15,16,26,30,31} As we show in this paper, the same concept of competition can rather be a mechanism to solve problems. That is, we present a learning algorithm that employs a competition mechanism to solve general semi-supervised classification problems.

Silva and Zhao²⁵ and Breve *et al.*⁷ studied data classification with a system of particles competing in vertices of complex networks. In their models, particles navigate within the network, to explore new territories and defend conquered territory against rival particles. A problem discussed by Silva and Zhao²⁵ is the overlapping nature of the vertices — this happens when data points of different classes share the same region in the attribute space. This subject has already been investigated by the same authors,²³ but we believe the edges of the network representation provide us more information about overlapping regions.

This paper extends the results of our previous paper,²⁷ a transductive semi-supervised learning model that employs a dynamical process in complex networks. In this process, namely Edge Competition, particles compete for dominance of *edges* in a network. After the end of the process, subnetworks are created with the edges, which are grouped by class dominance. Regarding the model proposed by Silva and

Zhao,²⁵ our learning technique differs in the following aspects:

- *Rather than incorporating a preferential walk mechanism where particles tend to avoid particles of different classes, we allow the direct conflict among different classes causing particles to “die” and be absorbed.* Eliminating the preferential walking mechanism largely simplifies the dynamical rules of particle competition model. The new model features the competition of only random walking particles. Such a simplification together with the continuous particle generating mechanism permits us to find out an equivalent asymptotic deterministic version to improve the model's efficiency significantly.
- *An asymptotically infinite number of particles compete instead of a fixed number.* The original particle competition is intrinsically stochastic. Thus, each simulation generates a different result inflicting the need for several simulation runs to obtain an average of the result. The stochastic model, therefore, has a high computation cost. Conversely, we consider an asymptotically infinite number of particles, leading to the discovery of a deterministic transition matrix that describes the dynamics of the expected number of particles in each vertex, and the number of particles passing through each edge. The deterministic system has a running time independent of the number of particles. Moreover, the deterministic model ensures a stable learning model. In other words, for the same input, the new model generates the same output for each run. Therefore, with the new model, only one run is enough for each input data set. Such a feature is quite attractive in practical applications.
- *Particles compete for the dominance of edges instead of vertices.* Our intuition is that the edge domination model gives us more detail than the vertex domination model. This advantage is due to each vertex having from one to several edges. Edges determine the network's topological structure. Consequently, our model has the benefit of granting essential information concerning overlapping vertices (vertices, which belong to more than one classes) by simply counting the edges dominated by each class of particles. Currently, we are conducting studies to prove the advantages of edge-centric dynamics over models of vertex-centric dynamics.

The studied model offers satisfactory performance on semi-supervised learning problems. Moreover, it is suitable for detecting overlapping regions of the data.

Section 2 introduces the proposed model and Sec. 3 addresses its time and space complexity order. Section 4 presents results of computer simulations that were performed on both artificial and real-world data to assess the model's performance. Finally, Sec. 5 concludes this paper.

2. Model Description

In this section, we give an introduction to our framework model, explaining its basic design. Whenever pertinent we go into detail for further clarification.

2.1. Overview

Consider a complex network expressed by a simple, unweighted, undirected graph $G = (\mathcal{V}; \mathcal{E})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. If two vertices are considered similar, an edge connects them. This network contains $|\mathcal{V}| = l + u$ vertices that can be either labeled or unlabeled data points. The set $\mathcal{L} = \{v_1; \dots; v_l\}$ contains the labeled vertices, where a vertex $v_i \in \mathcal{L}$ is labeled with $y_i \in \{1; \dots; C\}$. The set $\mathcal{U} = \{v_{l+1}; \dots; v_{l+u}\}$ contains the unlabeled vertices. The network is represented by the adjacency matrix $A = (a_{ij})$ where $a_{ij} = a_{ji} = 1$ if v_i is connected to v_j . In semi-supervised learning, the goal is to assign correctly existing labels to the unlabeled vertices $\mathcal{U}$, using the labeled vertices $\mathcal{L}$ for training.

In this model, particles are the objects that flow through the network while carrying a label. They are generated by the labeled vertices, which are called sources. After being released, a particle randomly walks the network. When walking the network, there are two ways a particle can be absorbed. If it encounters a sink — a labeled vertex whose class is different from the particle's — it will be absorbed. Also, it has a chance to be absorbed while passing through an edge that has particles carrying different labels. Details on the probability of a particle absorption will be discussed later.

As said before, labeled vertices are *sources* for particles of the same class and *sinks* for particles of other classes. This behavior is part of the competition model, in which a sink acts as a barrier to the particles of other classes. We also provide means by which a source may decrease its particle production over time. Since production can eventually halt and the system becomes absent of particles, this mechanism entails a stopping criterion. As it is not sure that every particle will be absorbed, the simulation needs to be time-limited as well.

By observing particles passing through an edge, we can establish the edge domination — by the end of the process, the dominating class of an edge has the highest share of visits of particles that have passed through the edge. The survival of a particle, however, is only determined by the particles present in the edge at the same time. If a particle survives, it reinforces its class presence on the edge and may continue walking; otherwise, the particle is absorbed, and it ceases to affect the system.

The process will eventually stop, and the result consists of edges grouped by their dominating class. For every class, the set of dominated edges is a subnetwork of G . Although an edge appears in only one set, an unlabeled vertex may appear in more than one subnetwork. For an unlabeled vertex, we expect to assign the same label as the subnetwork's label. However, if an unlabeled vertex appears in more than one subnetwork, we assign to it the label of its closest source.

2.2. An illustrative example

To provide an intuitive understanding of the process, we illustrate an iteration in Fig. 1. At the left-hand side of the figure, there are 13 particles in the system and gray

Fig. 1. Illustration of one iteration of the particle competition process. The network has seven vertices, where colored vertices are sources and sinks. The number by an edge represents the proportion of light-green particles that passed through the edge in the last iteration. In (Left), gray arrows indicate where each particle decided to move in the next iteration. (Right) is the iteration result; at the bottom, † indicates the absorbing state (color online).

arrows indicating where the particles have decided to move in the next iteration. The number by each edge indicates the *current edge dominance*, where the value is the proportion of light-green particles that passed through the edge during the last iteration. The probability of a particle to be absorbed is inversely proportional to the current edge dominance. If absorbed, a particle is moved to the absorbing state †.

At the right-hand side of the figure, the result of the iteration is shown. We can see that the particles 7, 9, 11, and 13 were absorbed — these particles failed to pass through the edges and did not change the current edge dominance. The current edge dominance updates with the proportion of the particles that successfully passed through the edge. The sources — red and green vertices — generated a total of three new particles (14, 15, and 16) to compensate for the absorbed particles. When no particle passes through an edge, the dominance level at the time is equally distributed among the classes.

2.3. Mathematical model of the dynamical process

Recurrent notations in this section are listed in Table 1.

Formally, we define Edge Competition as a dynamical process $\mathbf{X}$ whose state at time t is

$$\mathbf{X}(t) = [\mathbf{n}^c(t) \Delta^c(t)]^T; \quad (1)$$

where the number of particles belonging to class c in each vertex is $\mathbf{n}^c(t) = [n_i^c(t)]_i$, and the dominance of class c over each edge is the matrix $\Delta^c(t) = (\delta_{ij}^c(t))_{ij}$. Indexes i and j are vertices in the network; ij is the edge between v_i and v_j .

Although we have an undirected network model, particle movement is directed. The matrix Δ^c is the count of the total number of particles that have passed through each edge. The dominance of class c over edge ij is the sum $\delta_{ij}^c + \delta_{ji}^c$.

Without going into detail yet, we define the evolution of the system. Let $n_{i \rightarrow j}^c(t)$ be the number of particles belonging to class c that moved from vertex i to vertex j at

Table 1. Recurrent notations used throughout the text.

Notation	Description
$G = (\mathcal{V}; \mathcal{E})$	Simple, unweighted, undirected graph
$G^c = (\mathcal{V}; \mathcal{E}^c)$	Resulting subnetwork with edges dominated by label c
$\mathcal{L} = \{v_1; \dots; v_l\}$	Set of labeled vertices
$\mathcal{U} = \{v_{l+1}; \dots; v_{l+u}\}$	Set of unlabeled vertices
$y_i \in \{1; \dots; C\}$	Label of v_i
$A = (a_{ij})$	Adjacency matrix
$n^c(t) = [n_i^c(t)]$	Number (or fraction) of particles labeled with c in v_i
$\Delta^c(t) = (\bar{c}_{ij}(t))$	Dominance of class c over each edge
$n_{i \rightarrow j}^c$	Number of particles labeled with c traveling from v_i to v_j
$S = \{\dagger; 1; \dots; \mathcal{V} \}$	State space
$\mathcal{A}_c$	Absorbing states for class c
$\mathcal{X}_c$	Unreachable states for class c
$\mathcal{T}_c$	Transient states for class c
	Competition parameter
	Production decay parameter
$W^c = (w_{ij}^c)$	Motion probability for particles labeled with c
$\Gamma^c = (\bar{c}_{ij})$	Survival probability for particles labeled with c
P^c	Transition matrix for particles labeled with c

time t . The evolution function of the dynamical system is

$$: \begin{cases} n_i^c(t+1) = n_i^c(t) + \sum_j (n_{j \rightarrow i}(t+1) - n_{i \rightarrow j}^c(t+1)); \\ \bar{c}_{ij}(t+1) = \bar{c}_{ij}(t) + n_{i \rightarrow j}^c(t+1); \end{cases} \quad (2)$$

Intuitively, the number n_i^c of particles that are in a vertex is the number of previous particles, plus the number $n_{j \rightarrow i}^c$ of particles arriving, minus the number $n_{i \rightarrow j}^c$ of particles leaving. Moreover, to calculate the total number $\bar{c}_{ij}$ of particles that have passed through an edge, we simply add up the number of particles at each time. Values $n_{i \rightarrow j}^c$ are obtained stochastically according to the dynamics of walking, absorption, and generation. Next, we present the dynamics in detail.

2.3.1. Particle motion

As previously stated, a particle randomly walks within the network. Formally, the movement of particles belonging to class c is expressed by the transition matrix $W^c = (w_{ij}^c)$. The element w_{ij}^c is the probability of a particle moving from v_i to v_j . We define the symbolic vertex $\dagger$ to represent the place where absorbed particles go. From now on, we can treat the motion of the particles as a Markov chain whose state space is given by $S = \{\dagger; 1; \dots; |\mathcal{V}|\}$.

For each class c , the state space is split into three disjoint sets:

$$\mathcal{A}_c = \{\dagger\}; \quad (3)$$

$$\mathcal{X}_c = \{i | v_i \in \mathcal{L} \text{ and } y_i \neq c\}; \quad (4)$$

$$\mathcal{T}_c = S \setminus (\mathcal{A}_c \cup \mathcal{X}_c): \quad (5)$$

They represent, respectively, the absorbing, the unreachable, and the transient states. The state $\dagger$ is the single absorbing state of the system, Eq. (3). We say that a vertex labeled with c is a *sink* — an absorbing state — for any particle that does not carry the same label. Whenever a particle chooses to move into a sink, it is absorbed at the time of the decision, before reaching the sink. Thus, every sink is unreachable, Eq. (4). All the remaining states are transient, Eq. (5).

Now, we can explicitly calculate the transition probabilities w_{ij}^c , with $\deg v_i$ denoting the degree of vertex i . If $i \in \mathcal{T}_c$, then

$$w_{ij}^c = \begin{cases} \frac{1}{\deg v_i} a_{ij} & \text{if } j \in \mathcal{T}_c; \\ \frac{1}{\deg v_i} \sum_{k \in \mathcal{X}_c} a_{ik} & \text{if } j = \dagger; \\ 0 & \text{otherwise:} \end{cases} \quad (6)$$

On the other hand, if $i \in \mathcal{X}_c \cup \mathcal{A}_c$, then $w_{ij}^c = \mathbb{1}_{[j=\dagger]}$, where $\mathbb{1}_{[\cdot]}$ is the indicator function that yields one if the argument is logically true, and zero otherwise.

2.3.2. Particle absorption

Moving into a sink is not the only way a particle can be absorbed. When a particle decides to go through an edge ij such that $i; j \in \mathcal{T}_c$, the presence of other particles determines its survival. Mathematically, the survival probability ζ_{ij}^c of a class c particle to reach v_j given it is in v_i and it has chosen to move to v_j is

$$\zeta_{ij}^c(t) = (1 - \gamma) + \bar{n}_{i \leftrightarrow j}^c(t); \quad (7)$$

where $\gamma \in [0; 1]$ is the competition parameter and $\bar{n}_{i \leftrightarrow j}^c(t)$ is the *current relative domination* of class c in edge ij ,

$$\bar{n}_{i \leftrightarrow j}^c = \frac{n_{i \rightarrow j}^c + n_{j \rightarrow i}^c}{\sum_{k=1}^C (n_{i \rightarrow j}^k + n_{j \rightarrow i}^k)}: \quad (8)$$

We go further and incorporate the survival probability into the transition matrix W^c . First, we need to define the matrix $\Gamma^c(t) = (\zeta_{ij}^c(t))_{i,j \in S}$ whose entries are given by Eq. (7), for all $i; j \in \mathcal{T}_c$. For other cases, we set $\zeta_{ij}^c = 0$. Finally, let $\circ$ be the Hadamard (entrywise) product operator. The updated transition matrix is $\tilde{W}^c(t) = \Gamma^c(t) \circ W^c + (\tilde{d}_{ij}^c(t))_{i,j \in S}$, where

$$\tilde{d}_{ij}^c(t) = \begin{cases} 1 - \sum_{k \in S} \zeta_{ik}^c(t) w_{ik}^c & \text{if } j = \dagger; \\ 0 & \text{otherwise:} \end{cases} \quad (9)$$

2.3.3. Particle production

In our model, a source generates particles according to its degree and the current number of particles in the system. Also, the particle production rate may decay over time. Let $n^c(t) = \sum_{i \in \mathcal{T}} n_i^c$ be the number of active particles belonging to class c in the system at time t . We define $e^c(t) = n^c(0) t^{-\alpha}$, where $\alpha \in [0; \infty)$ is the production decay parameter. The lower the α , the longer sources will keep generating particles. When $\alpha = 0$, particle production keeps the system with a constant number of active particles.

At time t , a source generates new particles if $n^c(t) < e^c(t)$. Also, we guarantee that $n^c(t)$ is a nonincreasing function so that the number of particles generated at some time never surpasses the number of particles in vertex $\dagger$.

Since the number of generated particles depends on the number of absorbed particles, we denote g_i^c as the expected number of generated particles belonging to class c in the vertex i . Let $\mathcal{G}^c = \{v_i | v_i \in \mathcal{L} \text{ and } y_i = c\}$ be the sources for particles that belong to class c . At some time,

$$g_i^c = \frac{c_i}{\deg v_i} \cdot (e^c - N^c) \cdot \mathbb{1}_{[n^c < e^c]}; \quad (10)$$

where

$$c_i = \begin{cases} \frac{\deg v_i}{\sum_{v_j \in \mathcal{G}^c} \deg v_j} & \text{if } v_i \in \mathcal{G}^c; \\ 0 & \text{otherwise;} \end{cases} \quad (11)$$

This behavior can be incorporated into the transition matrix. The transition matrix $P^c(t) = (p_{ij}^c(t))$ that embraces the whole dynamics of the process is

$$P^c(t) = \Gamma^c(t) \circ W^c + (d_{ij}^c(t))_{i,j \in S}; \quad (12)$$

where

$$d_{ij}^c(t) = \begin{cases} 1 - \sum_{s \in S} \frac{c_s(t)}{\deg v_s} w_{is}^c & \text{if } i \neq \dagger \text{ and } j = \dagger; \\ 1 - \frac{e^c(t) - n^c(t)}{n^c(0)} \mathbb{1}_{[n^c(t) < e^c(t)]} & \text{if } i = \dagger \text{ and } j = \dagger; \\ \frac{1}{n^c(0)} g_i^c(t) & \text{if } i = \dagger \text{ and } j \neq \dagger; \\ 0 & \text{otherwise;} \end{cases} \quad (13)$$

2.4. Process simulation

We simulate the system with an asymptotically infinite number of initial particles. The vector n^c is understood as a stochastic row vector. The expected value for $n_{i \rightarrow j}^c(t+1)$ is equal to $n_i^c(t) \cdot p_{ij}^c(t)$. At the beginning of the process, n_i^c is proportional to the degree of vertex i . The simulation halts in two situations: when the system is considered to be absent of particles — that is, when $n_{\dagger}^c \approx 1$ for all label c —, and when the time limit has been reached.

2.5. Subnetwork construction

After the end of the process, we construct C complex networks $G^c = (\mathcal{V}; \mathcal{E}^c)$ where $\{\mathcal{E}^c\}_c$ is a partition of $\mathcal{E}$.

An edge ij is in $\mathcal{E}^c$ if and only if it is dominated by class c . That is, $ij \in \mathcal{E}^c \Leftrightarrow \arg \max_k \bar{k}_{ij}^{-k} = c$, where the relative domination is

$$\bar{k}_{ij}^{-c} = \frac{\bar{k}_{ij}^c + \bar{k}_{ji}^c}{\sum_{k=1}^C (\bar{k}_{ij}^k + \bar{k}_{ji}^k)}.$$

2.6. Vertex labeling

The networks G^c are used in the vertex classification. We assign a label $y_j \in \{1; \dots; C\}$ for each unlabeled vertex j , $v_j \in \mathcal{U}$, based on the information provided by the networks G^c and the labeled vertices, $v_i \in \mathcal{L}$. We denote l_{ji}^c for the length of the shortest path from v_j to v_i within the network G^c . If there are no paths between v_j and v_i in G^c , then $l_{ji}^c = \infty$.

A vertex is assigned with the label of its nearest labeled vertex in some network G^c . If two labeled vertices are equally distant from vertex j , instead of the nearest labeled vertex, the label is determined by the largest number of labeled vertices in the same component as vertex j . Formally, let $\epsilon < |\mathcal{L}|^{-1}$ be fixed and $\mathcal{R}_j^c = \{i | l_{ji}^c < \infty\}$. The label for vertex j is

$$y_j = \arg \min_c \left(\min_i l_{ji}^c - |\mathcal{R}_j^c| \right). \quad (14)$$

Note that due to the rules of the system evolution, when there is at least one path between vertex j and vertex i in the process network, $l_{ji}^c = \infty$ if $y_i \neq c$. Hence, $\mathcal{R}_j^c$ contains only vertices i such that $y_i = c$. Although unlikely, if two or more classes minimize Eq. (14), the label is chosen at random.

3. Computational Complexity

Here, we provide the theoretical time and space complexity of our algorithm in the worst case scenario.

To simulate the process, we first initialize the scalars $n_i^c(0)$ and $\bar{k}_i^c$ for every vertex v_i and class c . Also, we initialize matrix W^c for each class c . Since W^c consists of $|\mathcal{E}|$ nonzero entries, using sparse matrices this step has time and space complexity $O(C|\mathcal{V}| + C|\mathcal{E}|)$.

After the initialization, we simulate the system until it halts. Matrix P^c is calculated every iteration according to Eq. (12). Calculating $|\mathcal{E}|$ entries of matrix Γ^c takes time complexity $O(C|\mathcal{E}|)$. Thus, an element-wise multiplication of Eq. (12) requires $O(C|\mathcal{E}|)$ operations. The second part of the same equation requires $O(|\mathcal{V}|^2)$ calculations. In conclusion, P^c comprises $O(|\mathcal{E}| + |\mathcal{V}|)$ nonzero entries obtained by $O(C|\mathcal{E}| + |\mathcal{V}|^2)$ operations.

In each iteration, we multiply the current vector n^c and matrix P^c . This multiplication takes $O(|\mathcal{E}| + |\mathcal{V}|)$ operations since P^c is sparse. Consequently, simulating

the process until it halts has time complexity order of $O(\mathcal{TC}^2|\mathcal{E}| + \mathcal{TC}|\mathcal{V}|^2)$, where T is the number of iterations that depends on the user-defined time limit. Finally, the subnetwork construction requires $O(C|\mathcal{E}|)$ operations.

The vertex labeling step is the last step of the algorithm. The time complexity of this step can be efficiently calculated by using a breadth-first search in G^c starting at the labeled vertices. Hence, the worst-case time complexity $O(C|\mathcal{U}|(|\mathcal{V}| + |\mathcal{E}|))$.

In summary, considering the discussion above, our algorithm runs at the worst-case in $O(\mathcal{TC}^2|\mathcal{V}|^2)$ and has space complexity $O(C|\mathcal{V}| + C|\mathcal{E}|)$ because $|\mathcal{V}|^2 > |\mathcal{E}|$.

4. Computer Simulations

To assess the model's performance, in this section, we present the results for both artificial and real datasets. First, we show the process on synthetic data. Then, we present the simulation results for well-known benchmark datasets.

4.1. Simulations on artificial datasets

For a better understanding of the steps, we illustrate the process on three synthetic datasets. Each dataset has a different class distribution — a banana-shaped distribution, a ring-shaped distribution, and a Highleyman distribution. (The first and third datasets were generated with the PRTools framework.¹¹) The data of a class in the banana dataset are uniformly distributed along a banana shape and then superimposed on a normal distribution with the standard deviation of 1 along the axes. The ring-shaped dataset consists of data points distributed along five intersecting rings, a ring for each class. In Highleyman distribution, two classes are defined by bivariate normal distributions with different parameters.

Because the datasets are not in a network representation, we use the k-nearest neighbor (k-NN) graph construction method before applying our technique. In the constructed network, a vertex represents a data point, and it connects to its k-NNs — determined by Euclidean distance. We set parameters $k = 1$ for the first and third datasets. For the dataset with ring shapes, we set $k = 0.5$ and $k = 1$.

The first synthetic dataset has 600 samples equally split into two classes. Figure 2(a) depicts the initial state of the process where the dataset is represented by a network. (The network representation is obtained by setting $k = 4$ for the k-NN graph construction.) At this stage, the edges are not dominated by any of the classes. Though the particles are not shown, Figs. 2(b) and 2(c) are snapshots of the process evolution — at time 3 and 15 — where the edges are colored with the dominating classes at the time. In these illustrations, a solid line in dark red stands for an edge ij such that $\bar{ij}^{\text{red}} > \bar{ij}^{\text{green}}$, while a dashed line in light green stands for the opposite. When $\bar{ij}^{\text{red}} = \bar{ij}^{\text{green}}$, an edge is drawn in a solid light gray line. As expected, edges close to sources are dominated first, and farther edges are progressively dominated. At time 15, Fig. 2(c), every edge has been dominated, and the process stops. Figure 2(d) shows the classification following the process result. With 1% of the points in the labeled set, the technique can correctly identify the pattern formed by each class.

Fig. 2. Process evolution on a banana-shaped distribution dataset. Dark red and light green colors represent the two classes. Unlabeled points are black; labeled points are larger and colored. Edges are colored according to the dominating class at the time, where light gray stands for *not dominated*. (a) The network representation of the dataset at the beginning of the process. (b) and (c) Process at time 7 and 15, respectively. (d) The result of the dataset classification (color online).

The ring-shaped dataset consists of five classes with 200 samples each, with 8% of the data points randomly selected as labeled. Figure 3(a) depicts this dataset. (We set $k = 7$ for the k -NN algorithm.) Differently from the other two artificial experiments, this dataset has many intersecting regions. Five rings are disposed, side-by-side, in columns with three rings on the first column and two other in the second column, obtaining five intersecting regions. These intersecting regions have up to three different classes. Figure 3(b) shows the result of classification after a single epoch of the competition process simulation. Many edges were never visited by any of

Fig. 3. (a) Ring-shaped distribution dataset. White colored points are unlabeled data items. Labeled data points are larger and colored according to its class. (b)–(d) Classification results on a ring-shaped distribution dataset at time 1, 5, and 30. Vertex colors indicate the predicted class, and edge colors indicate the class of particles that dominated at the time. Edges in light gray were not dominated by any of particles, having dominance of $1/5$ for any class. Labeled data points are depicted in larger vertices (color online).

particles, letting a domination level set to $1/5$ for each class. Therefore, the neighboring vertices of those edges were randomly classified.

In this second experiment, we set $\alpha = 0.5$, which contrasts with the other two experiments set to the maximum competition. A smaller α allows particles to move farther into regions already dominated by rival particles. Particularly, in this dataset, particles may traverse the rival ring to reach an interlocked region of the ring. After a few simulation epochs, Fig. 3(c), particles have dominated most of the edges, but there were still some unvisited edges randomly classified. The boundaries were beginning to become well defined. Finally, Fig. 3(d) shows the classification result when the system had been converged to a state with stable boundaries between classes. Although some parts of the rings are encircled by data points of a different class (the interlocked parts of the rings), the presence of a labeled data point within the inner region, and even in the overlapping regions resulted with a better classification performance.

The third synthetic dataset has 300 samples, of which six are labeled. This dataset is shown in Fig. 4(a). (We set $k = 20$ for the k -NN algorithm.) We can observe that the labeled vertices of light-green class form a barrier to particles belonging to the dark-red class. After the convergence, however, the constructed subnetworks are still connected — there is a single component connecting all the vertices of the subnetwork (see Fig. 4(b)). This result reveals that the competition process in edges provides more information than the competition in vertices because it can identify the overlapping vertices as a part of the process.

An interesting question worth looking into is the subnetwork convergence. The process has a convergent state of subnetworks if there is some iteration at the process evolution that all subnetworks G^c keep unchanged in the next iterations. Particularly, we consider a case where the number of active particles is always greater than zero, no particle production, and the competition setting is at maximum. That is, when $\alpha = 0$ and $\beta = 1$.

Fig. 4. (a) Highleyman dataset with 600 samples. Larger points are labeled. The color indicates the class. (b) Subnetworks generated by the proposed model at time $t = 100$. Larger vertices are the source of particles (color online).

Fig. 5. Simulation to assess the subnetworks stability. Three colored lines indicate the average number of changes along the process evolution, for simulations that started with 0.5%, 1%, and 2% of labeled vertices; a 95% confidence interval estimative is depicted in light gray. The number of changes is the average number of edges whose dominant class has changed between time $t - 1$ and t (color online).

The subnetwork convergence is assessed by simulating the dynamical process in the ring-shaped dataset with different sets of labeled vertices. We varied the fraction of labeled vertices in 0.5%, 1%, and 2%. For each fraction, the system was simulated with 30 random initial configurations.

In Fig. 5, the number of edges whose dominant class has changed in one iteration is depicted over the process evolution. A colored line depicts the average number of changes in a system that started with a given proportion of labeled vertices. For every tested combination of parameters, the process converges with a speed proportional to the number of particles' sources. Furthermore, in every combination, the average number of changes is fewer than 1 for all $t \geq 31$.

4.2. Simulations on benchmark datasets

We compare our model with 14 semi-supervised techniques tested on Chapelle's benchmark.⁸ The benchmark consists of seven datasets that have 1500 data points each, except for BCI, which has 400 points. The datasets are described in Ref. 8.

The benchmark provides, for each dataset, 24 distinct, unbiased sets (splits) of labeled points. Half of the splits are formed by 10 labeled points and the other half by 100 labeled points. The benchmark ensures that each split contains at least one data point of each class. The result from a technique is the average test error — the proportion of data points incorrectly labeled — over the splits. Simulation results were collected from Ref. 8, except for LGC, LP, LNP, and Vertice Competition that were found in Ref. 25.

Table 2. Test errors (%) for splits with 10 labeled training points.

	g241c	g241d	Digit1	USPS	COIL	BCI	Text	Avg. Rank
1-NN	47.88	46.72	13.65	16.66	63.36	49.00	38.12	9.4
SVM	47.32	46.66	30.60	20.03	68.86	49.85	45.37	13.1
MVU + 1-NN	47.15	45.56	14.42	23.34	62.62	47.95	45.32	9.1
LEM + 1-NN	44.05	43.22	23.47	19.82	65.91	48.74	39.44	8.9
QC + CMR	39.96	46.55	9.80	13.61	59.63	50.36	40.79	7.0
Discrete regularization	49.59	49.05	12.64	16.07	63.38	49.51	40.37	10.4
TSVM	24.71	50.08	17.77	25.20	67.50	49.15	31.21	9.9
Cluster kernel	48.28	42.05	18.73	19.41	67.32	48.31	42.72	10.0
LDS	28.85	50.63	15.63	17.57	61.90	49.27	27.15	8.0
Laplacian RLS	43.85	45.68	5.44	18.99	54.54	48.97	33.68	5.4
LGC	45.82	44.09	9.89	9.03	63.45	47.09	46.83	6.7
LP	42.61	41.93	11.31	14.83	55.82	46.37	49.53	5.3
LNP	47.82	46.24	8.58	17.87	55.50	47.65	41.06	7.0
Vertice competition	41.17	43.51	8.10	15.69	54.18	48.00	34.84	3.7
<i>Our method</i>	46.30	46.29	5.86	14.77	59.99	48.78	36.04	6.0

For the dataset network representation, k-NN is used. However, the network may have components without labeled vertices, preventing particles from circulating through all vertices. Therefore, we connect two components with an edge between the closest pair of vertices until there is only one component.

For the simulation, we discretize the intervals of the parameters in $\alpha = \{0; 0.25; \dots; 1\}$ and $\beta = \{0; 0.50; \dots; 3\}$. We also vary the k-NN parameter $k \in \{1; 2; \dots; 10\}$. We tested every combination of k , α , and β . We halt the process at time $t = 1000$ if the system does not become absent of particles up to this iteration. For each run, the combination that yields best accuracy is selected as the result.

The results of test errors for splits with 10 labeled points are shown in Table 2; results for splits with 100 labeled points are shown in Table 3. In addition to the results from each dataset, the average performance rank of a technique over the

Table 3. Test errors (%) for splits with 100 labeled training points.

	g241c	g241d	Digit1	USPS	COIL	BCI	Text	Avg. Rank
1-NN	43.93	42.45	3.89	5.81	17.35	48.67	30.11	11.6
SVM	23.11	24.64	5.53	9.75	22.93	34.31	26.45	8.0
MVU + 1-NN	43.01	38.20	2.83	6.50	28.71	47.89	32.83	10.6
LEM + 1-NN	40.28	37.49	6.12	7.64	23.27	44.83	30.77	10.6
QC + CMR	22.05	28.20	3.15	6.36	10.03	46.22	25.71	6.6
Discrete regularization	43.65	41.65	2.77	4.68	9.61	47.67	24.00	7.1
TSVM	18.46	22.42	6.15	9.77	25.80	33.25	24.52	7.6
Cluster kernel	13.49	4.95	3.79	9.68	21.99	35.17	34.28	7.3
LDS	18.04	23.74	3.46	4.96	13.72	43.97	23.15	5.1
Laplacian RLS	24.36	26.46	2.92	4.68	11.92	31.36	23.57	4.2
LGC	41.64	40.08	2.72	3.68	45.55	43.50	56.83	9.1
LP	30.39	29.22	3.05	6.98	11.14	42.69	40.79	8.1
LNP	44.13	38.30	3.27	17.22	11.01	46.22	38.45	11.2
Vertice competition	21.41	25.85	3.11	4.82	10.94	41.57	27.92	5.0
<i>Our method</i>	41.16	39.77	2.37	5.59	15.11	45.44	28.43	7.9

datasets is shown in the last column. The ranking for a dataset is sorted by accuracy. The technique studied in this paper has a higher rank for splits with 10 labeled points. More labeled points allow our method to get better results though it does not benefit as much as other techniques. The fact that the technique can perform well with few labels is an interesting behavior since manual data labeling is an expensive process.

Notably, the proposed technique is not among the best for the g241c and g241d datasets. These two datasets are formed by two Gaussian-distributed classes with an overlapping region. In these two, the data points in overlapping regions are the most difficult to classify, even for an expert person. However, the benefit of this technique is the availability of edge information. It is expected to have different classes dominating the edges adjacent to vertices in the overlapping region. This feature implies that the vertices in overlapping regions will appear in more than one subnetwork during the classification step. As the framework model allows different methods to decide which class the vertex belongs to (we used the shortest path distance to a source), this could be the case for active learning, i.e. asking an expert to label a vertex in an overlapping region.

Table 4 shows the results with the standard deviation over the splits along with the combination of parameters that led to the best accuracy result.

We statistically compare the results shown in Tables 2 and 3. For all tests, we set our level of significance at 5%. First, we use a test based on the average rank of each method to evaluate the null hypothesis that all the techniques are equivalent. With the Friedman test,¹⁴ there is statistically significant difference between the rank of the techniques ($p < 0.01$). At a 5% significance level, the null hypothesis is rejected. Therefore, we cannot say that all techniques are equivalent.

Since the Friedman test result reports statistical significance, we can use the Wilcoxon signed-rank test¹⁴ for all possible pair combinations of techniques. In this paired difference test, the null hypothesis is that the first technique has greater or equal error results than the second. If rejected at a 5% significance level, then we say the first technique is superior to the second. By analyzing results for splits with 10 and 100 labeled points together, we conclude that our technique is superior to 1-NN ($p < 0.01$) and MVU + 1-NN ($p < 0.04$). Examining separately, for splits with 10 labeled points, our method is also superior to discrete regularization and SVM

Table 4. Test errors (%) with standard deviation and best parameters.

	10 labeled	k			100 labeled	k		
g241c	46:30 ± 2:49	6	0.75	0	41:16 ± 1:95	9	1	1
g241d	46:29 ± 3:42	7	1	0	39:77 ± 1:53	10	1	1.5
Digit1	5:86 ± 2:51	4	0.5	0.5	2:37 ± 0:48	4	0.75	0.5
USPS	14:77 ± 5:65	4	0.25	2.5	5:59 ± 0:77	4	1	1.5
COIL	59:99 ± 3:62	1	0	1	15:11 ± 2:88	1	0	0
BCI	48:78 ± 1:82	6	1	1.5	45:44 ± 2:96	1	1	0.5
Text	36:04 ± 2:98	8	0.75	0	28:43 ± 0:92	3	0.75	0.5

($p < 0.01$); whereas for splits with 100 labeled points, Laplacian RLS, LDS, and Vertex Competition ($p < 0.03$) are superior to ours. This test result confirms that our method performs well with a small set of labeled points while not benefiting greatly from larger labeled sets.

5. Conclusion

We presented a two-step semi-supervised learning technique based on a dynamical process on complex networks. First, the Edge Competition process is simulated on an input network. Second, the process results are used to generate subnetworks for each class in the problem. Labels are assigned according to the closest labeled vertex in the subnetworks.

The novel process is mathematically modeled in detail. In summary, an asymptotically infinite number of particles compete for edges in a network. When a particle passes by an edge, it increases its dominance whilst decreasing other classes' dominance. Three dynamics provide a biologically inspired scenario of competition and cooperation.

The proposed method is suitable for the task of semi-supervised data classification and revealed its ability to provide information about overlapping regions. Furthermore, it is straightforward to adapt the model for data clustering and community detection.

Despite the satisfactory performance, the classification step is rather limited. In this step, only the shortest paths between the vertices in the subnetworks have been considered. We believe communicability between pairs of vertices¹² may be a favorable alternative. In the future, we will further investigate the advantages of dynamics in edges. Moreover, studying mathematical properties of no-competition scenarios ($\alpha = 0$, and $\beta = 0$ with $\gamma \rightarrow \infty$), we will better comprehend complex cases ($\alpha > 0$).

Acknowledgments

This research was supported by the São Paulo State Research Foundation (FAPESP), the Coordination for the Improvement of Higher Education Personnel (CAPES), and the Brazilian National Research Council (CNPq).

References

1. A. Barrat, M. Barthélemy and A. Vespignani, *Dynamical Processes on Complex Networks* (Cambridge University Press, New York, NY, USA, 2008).
2. M. Belkin and P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, *Neural Comput.* **15**(6) (2003) 1373–1396.
3. M. Belkin, P. Niyogi and V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, *J. Mach. Learn. Res.* **7** (2006) 2399–2434.
4. C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006).

5. A. Blum and S. Chawla, Learning from labeled and unlabeled data using graph mincuts, *Int. Conf. Machine Learning*, 2001, pp. 19–26.
6. S. Boccaletti, V. Latora, Y. Moreno, M. Chavez and D. U. Hwang, Complex networks: Structure and dynamics, *Phys. Reports* **424**(4–5) (2006) 175–308.
7. F. Breve, L. Zhao, M. G. Quiles, W. Pedrycz and J. Liu, Particle competition and co-operation in networks for semi-supervised learning, *IEEE Trans. Knowl. Data Eng.* **24**(9) (2012) 1686–1698.
8. O. Chapelle, B. Schölkopf and A. Zien (eds.), *Semi-Supervised Learning* (MIT Press, Cambridge, MA, 2006).
9. L. Cheng and S. J. Pan, Semi-supervised domain adaptation on manifolds, *IEEE Trans. Neural Networks and Learning Syst.* **25**(12) (2014) 2240–2249.
10. T. Cupertino and L. Zhao, Using interacting forces to perform semi-supervised learning, *2012 Brazilian Symp. Neural Networks (SBRN)* (IEEE, 2012), pp. 91–96.
11. R. Duin, P. Juszczak, P. Paclík, E. Pekalska, D. de Ridder, D. Tax and S. Verzakov, PR-Tools 4.1, a matlab toolbox for pattern recognition (2007).
12. E. Estrada and N. Hatano, Communicability in complex networks, *Phys. Rev. E* **77**(3) (2008) 036111.
13. R. A. Gueleri, T. H. Cupertino, A. C. de Carvalho and L. Zhao, A flocking-like technique to perform semi-supervised learning, *2014 Int. Joint Conf. Neural Networks (IJCNN)* (IEEE, 2014), pp. 1579–1586.
14. M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*, 2nd edn. (Wiley-Interscience, 1999).
15. Y. Jiang, Z. Shao and Y. Guo, A DAG scheduling scheme on heterogeneous computing systems using tuple-based chemical reaction optimization, *The Scientific World J.* **2014** (2014) 404375.
16. C. Liu, K. Li, C. Xu and K. Li, Strategy configurations of multiple users competition for cloud service reservation, *IEEE Trans. Paralle. Distr. Syst.* **27**(2) (2016) 508–520.
17. M. Loog and A. C. Jensen, Semi-supervised nearest mean classification through a constrained log-likelihood, *IEEE Trans. Neural Networks and Learning Syst.* **26**(5) (2015) 995–1006.
18. T. M. Mitchell, *Machine Learning*, 1st edn. (McGraw-Hill, Inc., New York, NY, USA, 1997).
19. M. E. J. Newman, *Networks: An Introduction*, 1st edn. (Oxford University Press, New York, NY, USA, 2010).
20. K. Nigam, A. K. McCallum, S. Thrun and T. Mitchell, Text classification from labeled and unlabeled documents using EM, *Machine Learning* **39**(2–3) (2000) 103–134.
21. S. T. Roweis and L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, *Science* **290** (2000) 2323.
22. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd edn. (Prentice Hall Press, Upper Saddle River, NJ, USA, 2009).
23. T. C. Silva and L. Zhao, Detecting overlapping structures via network-based competitive learning, in *Proc. Int. Joint Conf. Neural Networks (IJCNN)* (IEEE, 2012), pp. 1–8.
24. T. C. Silva and L. Zhao, Semi-supervised learning guided by the modularity measure in complex networks, *Neurocomputing* **78**(1) (2012) 30–37.
25. T. C. Silva and L. Zhao, Network-based stochastic semisupervised learning, *IEEE Trans. Neural Networks and Learning Syst.* **23**(3) (2012) 451–66.
26. T. K. Truong, K. Li and Y. Xu, Chemical reaction optimization with greedy strategy for the 0-1 knapsack problem, *Appl. Soft Comput. J.* **13**(4) (2013) 1774–1780.
27. P. R. Urio, F. A. N. Verri and L. Zhao, Semi-supervised learning by edge domination in complex networks, *Proc. IEEE 11th Int. Conf. Natural Computation (ICNC'15)*, Zhangjiajie, China (IEEE, 2015), pp. 516–521.

28. V. N. Vapnik, *Statistical Learning Theory* (Wiley, New York, NY, 1998).
29. K. Wagstaff, C. Cardie, S. Rogers and S. Schroedl, Constrained k-means clustering with background knowledge, in *Proc. 18th Int. Conf. Machine Learning* (Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 2001), pp. 577–584.
30. Y. Xu, K. Li, L. He, L. Zhang and K. Li, A hybrid chemical reaction optimization scheme for task scheduling on heterogeneous computing systems, *IEEE Trans. Parall. Distr. Syst.* **9219**(c) (2014) 1.
31. Y. Xu, K. Li, J. Hu and K. Li, A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues, *Inform. Sci.* **270** (2014) 255–287.
32. K. Zhang, L. Lan, J. T. Kwok, S. Vucetic and B. Parvin, Scaling up graph-based semi-supervised learning via prototype vector machines, *IEEE Trans. Neural Networks and Learning Syst.* **26**(3) (2015) 444–457.
33. Z.-H. Zhou and M. Li, Tri-training: Exploiting unlabeled data using three classifiers, *IEEE Trans. Knowl. Data Eng.* **17**(11) (2005) 1529–1541.
34. X. Zhu, Semi-supervised learning with graphs, doctoral thesis, Carnegie Mellon University (2005).
35. X. Zhu and A. B. Goldberg, Introduction to semi-supervised learning, *Synthesis Lectures on Artificial Intelligence and Machine Learning* (Morgan and Claypool Publishers, 2009), pp. 1–130.

Paulo R. Urio obtained his B.Sc. degree in Computer Science from the Central Western Parana State University, Guarapuava, Brazil, in 2014. Currently, he is an M.Sc. student at the University of São Paulo, São Carlos, Brazil. His research interests include machine learning and complex networks.

Filipe A. N. Verri received his B.Sc. degree in Computer Science from the University of São Paulo, São Carlos, Brazil, in 2014 as the highest-ranked student. Currently, he is a PhD candidate in Computer Sciences at the University of São Paulo, São Carlos, Brazil, and a visiting scholar in the School of Electrical, Computer and Energy Engineering at Arizona State University, Tempe, USA. His research interests are data classification, pattern recognition, complex networks, and complex systems.

Liang Zhao received his B.Sc. degree from Wuhan University, Wuhan, China, and his M.Sc. and Ph.D. degrees from the Aeronautic Institute of Technology, São José dos Campos, Brazil, in 1988, 1996, and 1998, respectively, all in Computer Science. He joined the University of São Paulo, Brazil, in 2000, where he is currently a Full Professor with the Department of Computing and Mathematics. From 2003 to 2004, he was a visiting researcher with the Department of Mathematics, Arizona State University, Tempe, USA. His current research interests include artificial neural networks, machine learning, nonlinear dynamical systems, complex networks, bioinformatics, and pattern recognition. He is an Associate Editor of Neural Networks and he is also a recipient of the Brazilian Research Productivity Fellowship.

FEATURES OF EDGE-CENTRIC COLLECTIVE DYNAMICS IN MACHINE LEARNING TASKS

Paulo R. Urio, Filipe A. N. Verri, and Liang Zhao

Features of Edge-Centric Collective Dynamics in Machine Learning Tasks

2016 6th International Conference on Nonlinear Science and Complexity

INPE – National Institute for Space Research
São José dos Campos – SP – Brazil – May 16-20, 2016

FEATURES OF EDGE-CENTRIC COLLECTIVE DYNAMICS IN MACHINE LEARNING TASKS

Paulo Roberto Urio¹, Filipe Alves Neto Verri², Liang Zhao³

¹Institute of Mathematical and Computer Sciences, University of São Paulo, São Carlos, Brazil, urio@usp.br

²Institute of Mathematical and Computer Sciences, University of São Paulo, São Carlos, Brazil, and School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe AZ, USA, filipeneto@usp.br

³Ribeirão Preto School of Philosophy, Science and Literature, University of São Paulo, Ribeirão Preto, Brazil, zhao@usp.br

Abstract: We study how collective dynamics can solve Machine Learning tasks. The Edge Domination System is an algorithm to reveal patterns and obtain information of the underlying complex network. The algorithm consists in the simulation of a dynamical system based on particle competition for the dominance of edges. In this paper, we apply this method to semi-supervised tasks of data learning problems. We propose a vertex-centric version of this model and assess the differences between the edge-centric model.

keywords: Analysis and Control of Nonlinear Dynamical Systems with Practical Applications, Complex Networks, Nonlinear Dynamics and Complex Systems, Collective Dynamics and Particle Competition, Machine Learning.

1. INTRODUCTION

We study how collective dynamics can benefit us on data learning tasks. Among the proposed models found in the literature, we have those which feature self-organized and distributed behavior, through the topological structure of the interacting network, and simple evolution rules. Examples of decentralized natural systems include models of ant colonies, birds with flocking behavior, and models of competition for resources and cooperation between species — such models are applied to optimization and learning problems and are increasingly found in the literature [1–3].

A model we have proposed before is the Edge Domination System (EDS) [4], which is a deterministic algorithm

used to obtain information of the underlying complex network by exploring patterns revealed by the collective dynamics of particle competition. The model consists of particles randomly walking a complex network. Particles carry fixed labels and try to dominate the edges of the network by passing through more frequently than rival particles. Unlike similar approaches, in this model the resource of competition is the edges instead of vertices. However, the advantages of such approach are not clear. Therefore, in this paper, we explore the differences between resources of competition. We propose a comparable model of particle competition on vertex, and we compare with the EDS.

2. MODEL DESCRIPTION

The Edge Domination System [4] models a particle competition process in a complex network. Particles compete for the domination of edges of the network. The network is a simple, unweighted, undirected graph $G = (\mathcal{V}; \mathcal{E})$, where $\mathcal{V} = \mathcal{L} \cup \mathcal{U}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. The sets $\mathcal{L} = \{v_1, \dots, v_l\}$ and $\mathcal{U} = \{v_{l+1}, \dots, v_{l+u}\}$ contain the labeled and the unlabeled vertices, respectively. Each labeled vertex $v_i \in \mathcal{L}$ has a label $y_i \in \{1, \dots, C\}$. The number of labels C is greater than 1. We denote $(i; j)$ as the edge between vertices v_i and v_j . The number of vertices is $|\mathcal{V}| = l + u$, and we suppose that $l \ll u$. The network is represented by the adjacency matrix $A = (a_{ij})$ where $a_{ij} = a_{ji} = 1$ if v_i is connected to v_j .

Particles randomly walk from vertex to vertex while carrying a fixed label. For a particle carrying label c , any particle/vertex of a label different from c is a rival particle/vertex. The edge counts the number of particles that pass through it, separately by each label. The label with the highest count is the label which dominates that edge. In this system, a variable number of particles can be generated into or be removed from the system. The three action rules described below.

Walking. A particle in any vertex chooses the next vertex with equal probability between the neighbors of the current vertex. Let $\deg v_i$ be the degree of v_i . A particle in v_i chooses to move to v_j with probability $a_{ij} = \deg v_i$.

Absorption. The current label domination in that edge determines whether the particle either goes to the vertex or it vanishes from the system (absorbed). This rule applies unless the next vertex is a rival source (sink), in which case the particle is absorbed. Thus, if the destination vertex is not a sink and the rule is applicable, the survival probability is $1 - \frac{c_{ij}}{c_{ij}}$, where $\frac{c_{ij}}{c_{ij}} \in [0; 1]$ is the competition parameter and $\frac{c_{ij}}{c_{ij}}$ is the proportion of rival particles that passed through edge $(i; j)$ in the previous iteration.

Generation. A labeled vertex (source) generates particles according to its degree and the current number of active particles in the system. Let n^c and n_0^c be the current and the initial number of particles carrying label c , respectively, and $\mathcal{G}^c$ be the set of sources for particles carrying label c . If the total number of particles is fewer than the initial number, then each source performs $n_0^c - n^c$ trials with probability $\frac{c}{i} \propto \deg v_i$ of generating a new particle.

The original model considers an accumulated domination to produce the unfoldings [4]. Conversely, we use only the current domination to simplify our investigation. The convergence of the system justifies this decision because, after several iterations, the current domination becomes proportional to the accumulated domination.

We present now the slightly modified version of the EDS and its vertex-based counterpart model.

2.1. Edge Domination System

The Edge Domination System is a nonlinear Markovian dynamical system with the state and evolution function

$$X(t) = \begin{bmatrix} \mathbf{n}^c(t) = [n_i^c(t)]_{i=1, \dots, jVj} \\ N^c(t) = (n_{ij}^c(t))_{i,j=1, \dots, jVj} \end{bmatrix}_{c=1, \dots, C} \quad (1)$$

$$\begin{cases} \mathbf{n}^c(t+1) = \mathbf{n}^c(t) \times P^c(N^1(t); \dots; N^C(t)) + \mathbf{g}^c(\mathbf{n}^c(t)) \\ N^c(t+1) = \text{diag } \mathbf{n}^c(t) \times P^c(N^1(t); \dots; N^C(t)), \end{cases} \quad (2)$$

where $P^c(\cdot) = (p_{ij}^c(\cdot))_{i,j}$,

$$p_{ij}^c(\cdot) = \begin{cases} 0 & \text{if } v_j \in \mathcal{L} \text{ and } y_j \neq c, \\ \frac{a_{ij}}{\deg v_i} (1 - \frac{c_{ij}}{c_{ij}}(\cdot)) & \text{otherwise,} \end{cases} \quad (3)$$

$$c_{ij}(N^1; \dots; N^C) = \begin{cases} 1 - \frac{n_{ij}^c + n_{ji}^c}{\sum_{q=1}^C n_{ij}^q + n_{ji}^q} & \text{if } \sum_{q=1}^C n_{ij}^q + n_{ji}^q > 0, \\ 1 - \frac{1}{C} & \text{otherwise,} \end{cases} \quad (4)$$

and $\mathbf{g}^c(\mathbf{n}) = [\mathbf{g}_i^c(\mathbf{n})]_i$, where

$$\mathbf{g}_i^c(\mathbf{n}) = \frac{c}{i} \max\{0; 1 - \mathbf{1} \cdot \mathbf{n}\}. \quad (5)$$

The initial state of the system X is given by an arbitrary discrete *distribution* $\mathbf{n}^c(0)$ of initial active particles and $n_{ij}^c(0) = 0$ for all $i; j; c$.

We group the edges by label domination at the current time. For each label c , the subset of edges $\mathcal{E}^c(t) \subseteq \mathcal{E}$ is

$$\mathcal{E}^c(t) = \left\{ ij \left| \arg \max_q (n_{ij}^q(t) + n_{ji}^q(t)) = c \right. \right\}. \quad (6)$$

Thus, the unfolding G^c that comprises the edges dominated by the label c at time t is $G^c(t) = (\mathcal{V}; \mathcal{E}^c(t))$.

2.2. Vertex Domination System

Based on the EDS, we model a vertex-centric system with state and evolution function

$$X(t) = \left[\mathbf{n}^c(t) = [n_i^c(t)]_{i=1, \dots, jVj} \right]_{c=1, \dots, C} \quad (7)$$

$$\mathbf{n}^c(t+1) = \mathbf{n}^c(t) \times P^c(\mathbf{n}^1(t); \dots; \mathbf{n}^C(t)) + \mathbf{g}^c(\mathbf{n}^c(t)), \quad (8)$$

where $P^c(\cdot)$ is eq. (3) with

$$c_{ij}(\mathbf{n}^1; \dots; \mathbf{n}^C) = \begin{cases} 1 - \frac{n_j^c}{\sum_{q=1}^C n_j^q} & \text{if } \sum_{q=1}^C n_j^q > 0, \\ 1 - \frac{1}{C} & \text{otherwise,} \end{cases} \quad (9)$$

and $\mathbf{g}^c(\mathbf{n})$ is eq. (5).

The initial state of the system X is given by an arbitrary discrete *distribution* $\mathbf{n}^c(0)$ of initial active particles.

3. COMPUTER SIMULATIONS

In this section, we present empirical experiments that address the differences between both systems — Edge Domination System (EDS) and Vertex Domination System (VDS).

First, we formulate a simple example where the system behaves quite differently. Afterward, we compare the performance of each system in artificial semi-supervised tasks. Finally, we measure properties of both systems over time.

3.1. Motivation

Consider the network whose vertices are points in a 2-dimensional plane depicted in fig. 1a. The vertices of this network can be either positive (green, light, shaping a vertical pattern) or negative (red, dark, shaping a horizontal pattern). The central vertex (blue, dark) does not have a defined label. Four vertices (square-shaped) are labeled. We connect each vertex to its nearest neighbor. Using this network as the input of EDS and VDS, one shall expect the fraction of positive and negative particles to be the same in the central vertex for both systems. Figure 2a demonstrates this behavior.

Now, consider adding a new vertex to this dataset as shown in fig. 1b. Notice that the modification barely changes

Figure 1 – Toy networks with (a) 9 vertices and (b) 10 vertices. Square-shaped vertices are initially labeled.

Figure 2 – Fraction of positive particles in each vertex over time with (a) 9-vertices and (b) 10-vertices networks as input.

the network’s topology. However, using this network as input both systems behaves quite differently, see fig. 2b. On the one hand, the proportion of different kinds of particles in the central vertex alternates about the same in the edge-centric system. On the other hand, the central vertex is dominated by positive-labeled particles in the vertex-centric system.

This example motivates us to investigate the advantages of the edge-centric model against the vertex-centric one.

3.2. Systems’ comparison

Before comparing both systems, we propose a random network with labeled vertices.

The undirected random network $G(n; m; p; q)$ has $n + m$ vertices of which n are positive and m negative labeled. Let $y_i \in \{+, -\}$ be the label of vertex v_i , the probability of existing an edge between $(i; j)$ is p if $y_i = y_j$, and q if $y_i \neq y_j$. Networks G that $n; m > 0$, $p; q \in [0; 1]$, and $p > q$ are a good representation of common machine learning tasks.

We compare classification accuracy of the systems for varying levels of difficulty. The mean accuracy is calculated over 30 independently sampled networks $G(250; 250; 0.6; q)$ for all $q \in \{0.1; 0.2; \dots; 0.6\}$. We fixed $\beta = 1$ and set the fraction of initially labeled vertices in $\{0.01; 0.02; 0.05\}$.

Figure 3 – Mean accuracy of both systems in different networks varying the fraction of labeled input vertices.

To classify an unlabeled vertex v_i , we adopt the original strategy for the EDS at time $t = 500$, $y_i = \arg \max_c |\mathcal{E}(\mathcal{N}_i^c(t))|$, where $|\mathcal{E}(\mathcal{N}_i^c(t))|$ stands for the number of edges in the neighborhood of v_i in $G^c(t)$.

In VDS, we just consider the dominating label of the vertex at time $t = 500$, $y_i = \arg \max_c n_i^c(t)$.

The accuracy results varying the network G and the number of sources is shown in fig. 3. Consistently, as q increases and the number of sources decreases, the problem becomes harder, and the systems obtain lower accuracies.

When $p = q = 0.6$, both systems fail to discriminate the label of the vertices because the network has no meaningful topology. Besides the number of sources compensating the difficulty of the problem, EDS performs better in realistic scenarios where sources are enough to discover a non-trivial class topology.

Now, we detail some general dynamical properties of the systems. Let $\delta(\cdot)$ be 1 for a true argument and 0 otherwise; the properties are listed below.

Positive domination. Defined as the fraction of vertices with the number of positive particles greater than the negative ones,

$$\frac{1}{|\mathcal{V}|} \sum_i (\delta(n_i^+(t) > n_i^-(t))). \quad (10)$$

Active particles. Defined as the sum of active particles in the system,

$$\sum_i n_i^+(t) + n_i^-(t). \quad (11)$$

Mean entropy. Defined as the average of the entropy considering the labels of active particles in each vertex,

$$-\frac{1}{|\mathcal{V}|} \sum_i n_i^+(t) \log_2 n_i^+(t) + n_i^-(t) \log_2 n_i^-(t). \quad (12)$$

The experimental results for the three properties over time are shown in fig. 4. Each result corresponds to the average of 30 independent simulations with $\beta = 1$ and networks

Figure 4 – Properties of EDS and VDS over time.

Figure 5 – Active particles and mean entropy properties varying the survival parameter λ .

$G(250;250;0.6;0.3)$ with one positive and one negative labeled vertex. Since the network has the same size and topology for the two labels, half network is dominated by each label in both systems. However, the number of active particles and the entropy are higher in the edge-centric system.

High entropy suggests that particles propagate farther, reaching larger regions dominated by rival particles. As a result, a greater exploration behavior emerges, and particles are more likely to learn from the network’s topology. Conversely, a large number of active particles means that fewer particles are absorbed — it could be a consequence of better boundaries between positive and negative labeled vertices, justifying the better accuracy of EDS in these networks.

Using the same network configuration and the number of executions, we also study how two properties — active particles and mean entropy — behave according to the setting of the survival parameter λ at time $t = 50$. The results are shown in fig. 5. On the one hand, as $\lambda \rightarrow 0$, the two properties in EDS and VDS become identical, which is expected since the two systems operate identically, without competition. On the other hand, as λ approaches 1, EDS presents a higher number of active particles and higher entropy.

4. FUTURE WORKS

In the previous section, we presented experimental investigations that address the differences between the systems EDS and VDS. In future works, we will provide analytical studies that explain the advantages of EDS over VDS. To do so, we will consider two simplifications in the systems: (a) ignore sink absorption, and (b) suppose that $r_{ij}^c(t) + r_{ji}^c(t) > 0$ for all i, j, t for at least one label c .

Since we expect few labeled vertices, we can neglect the sinks effect in the system. The second simplification means that we expected that there are no regions in the network that lack particles walking in. The resulting system evolution equations for EDS and VDS, respectively, are

$$\begin{cases} r_i^c(t+1) = r_i^c(t) \left(1 - \sum_j r_j^c(t) \right) + \sum_j r_{ji}^c(t+1) \\ r_{ij}^c(t+1) = r_{ij}^c(t) \frac{a_{ij}}{\deg v_i} \left(1 - \left[1 - \frac{r_{ij}^c(t) + r_{ji}^c(t)}{\sum_q r_{ij}^q(t) + r_{ji}^q(t)} \right] \right) \end{cases}, \quad (13)$$

$$\sum_j r_j^c(t) \frac{a_{ji}}{\deg v_j} \left(1 - \left[1 - \frac{r_i^c(t)}{\sum_q r_i^q(t)} \right] \right). \quad (14)$$

5. CONCLUSION

In this paper, we present a vertex version of the model of particle competition, which is similar to the already proposed Edge Domination System. We also perform experiments to study their differences.

Since both systems have the same time and memory complexity order, we conclude that using EDS is preferred over VDS, as the former have better features for learning tasks.

ACKNOWLEDGMENTS

This research was supported by the São Paulo State Research Foundation (FAPESP), the Coordination for the Improvement of Higher Education Personnel (CAPES), and the Brazilian National Research Council (CNPq).

References

- [1] R. A. Gueleri, T. H. Cupertino, A. C. de Carvalho, and L. Zhao, “A flocking-like technique to perform semi-supervised learning,” in *IJCNN*, pp. 1579–1586, 2014.
- [2] Z. Zhang, K. Long, J. Wang, and F. Dressler, “On swarm intelligence inspired self-organized networking: Its bionic mechanisms, designing principles and optimization approaches,” *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 513–537, 2014.
- [3] F. Breve, M. G. Quiles, and L. Zhao, “Interactive image segmentation using particle competition and cooperation,” in *IJCNN*, pp. 1–8, IEEE, 2015.
- [4] F. A. N. Verri, P. R. Urio, and L. Zhao, “Network Unfolding Map by Edge Dynamics Modeling,” 2016. arXiv:1603.01182.

DATA CLUSTERING WITH EDGE DOMINATION IN COMPLEX NETWORKS

Paulo R. Urío and Liang Zhao

Data Clustering with Edge Domination in Complex Networks

Submitted to BRACIS 2017

Data clustering with edge domination in complex networks

Paulo Roberto Urio¹ and Zhao Liang²

¹ Institute of Mathematical and Computer Sciences
University of Sao Paulo
Sao Carlos, SP, Brazil
urio@usp.br

² Ribeiro Preto School of Philosophy, Science and Literature
University of Sao Paulo
Ribeirao preto, SP, Brazil

Abstract. This paper presents a model for a dynamical system where particles dominate edges in a complex network. The proposed dynamical system is then extended to an application on the problem of community detection and data clustering. In the case of the data clustering problem, 6 different techniques were simulated on 10 different datasets in order to compare with the proposed technique. The results show that the proposed algorithm performs well when prior knowledge of the number of clusters is known to the algorithm.

1 Introduction

Consider a dataset that is represented by a weighted, undirected graph where a vertex represents a data point and an edge a relationship of similarity. Particularly, if a dynamical system can take place in this graph representation then this graph can be studied with tools of the complex network theory [2].

In machine learning, methods that are based on networks have increasingly being studied. The representation of a dataset as a network allows the method to work not only with a similarity score among pairs of nodes, but it also provides a topological information. Within this context, the goal of this paper is to present a dynamical system of particle competition system on edges of complex networks for the community detection and the data clustering problem.

Regarding previous work with semi-supervised learning, in this paper the presented model modifies dynamics of particle generation and introduces the information of weights in edges. In Section 2 the model is introduced with an overview and a mathematical description. In Section 3 simulations on a real network, artificial datasets, and real-world datasets are presented. Finally in Section 4, some final considerations regarding the studied model are discussed.

2 Edge Domination System

In this section, I give an introduction to the proposed technique, namely Edge Domination System. First an overview and then its mathematical modeling.

2.1 Overview

Consider a complex network expressed by a simple, undirected, weighted graph $\mathbf{G} = (\mathcal{V}; \mathcal{E})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. If two vertices are considered similar, then they are connected by an edge. Denote $(i; j)$ to be the edge between vertices $\mathbf{v}_i$ and $\mathbf{v}_j$, which has a weight $\mathcal{W}(i; j) \in \mathbb{R}$. The weights represent the similarity between two vertices where a larger value indicates higher similarity. The graph that expresses this network is represented by the weighted adjacency matrix $\mathbf{W} = (\mathbf{w}_{ij})$ where $\mathbf{w}_{ij} = \mathbf{w}_{ji} = \mathcal{W}(i; j)$ if $\mathbf{v}_i$ is connected to $\mathbf{v}_j$ and zero otherwise. The process' result consists of K groups of vertices that are not necessarily connected.

In this model, particles are the objects that flow within the network. Every particle belongs to a class, defined at time of their creation. After being released in any node of the network, a particle randomly walks the network. The probability among adjacent vertices to be chosen as the next vertex follows the distribution of the weights of the possible edges. Consider a particle that is in $\mathbf{v}_i$. This particle decides to move to $\mathbf{v}_j$ with probability

$$\frac{\mathbf{w}_{ij}}{\sum_{k=1}^{|\mathcal{V}|} \mathbf{w}_{ik}}.$$

The particle's decision, however, does not imply it will succeed at moving to the neighboring vertex. If the edge that is connecting this vertex has been visited by particles of different classes, this particle might be absorbed before reaching the vertex and then it will cease to affect the system. If the particle succeeds at moving to the neighboring vertex, it is said that this particle has survived—and it will continue walking through the network. This walking dynamics is modeled in terms of level of subordination and domination of a class in relation to all other classes of particles.

In order to determine the level of domination and subordination of each class in an edge we observe the active particles at a given time in the system. Define *current directed domination* $\tilde{n}_{ij}^c(\mathbf{t})$ to be the number of active particles that belong to class $\mathbf{c}$ that have decided to move from $\mathbf{v}_i$ to $\mathbf{v}_j$ at time $\mathbf{t}$ and survived. Similarly, define *current relative subordination* $\tilde{\sim}_{ij}^c$ to be the fraction of active particles that do not belong to class $\mathbf{c}$ and have successfully passed through edge $(i; j)$ *regardless of direction* at time $\mathbf{t}$. The latter is defined as

$$\tilde{\sim}_{ij}^c := \begin{cases} 1 - \frac{\tilde{n}_{ij}^c + \tilde{n}_{ji}^c}{\sum_{q=1}^K \tilde{n}_{ij}^q + \tilde{n}_{ji}^q} & \text{if } \tilde{n}_{ij}^q + \tilde{n}_{ji}^q > 0; \forall q, \\ \frac{1}{K} & \text{otherwise.} \end{cases}$$

The survival of a particle depends on the current relative domination of the edge and the destination vertex. The survival probability is

$$1 - \tilde{\sim}_{ij}^c(\mathbf{t})$$

where $\alpha \in [0; 1]$ is the competition parameter.

Since particles are absorbed through the dynamics of competition, a mechanism to perform replacement of absorbed particles is needed in order to avoid a state where there are no active particles in the system. This replacement is done by creating new particles according to the distribution of the current active particles of a given class in the system.

Let $n_i^c(t)$ be the number of active particles that belong to class c at time t . The number of new particles that will belong to class c in v_i at time t follows the distribution

$$\begin{cases} B(\lfloor \tilde{n}^c(0) - \tilde{n}^c(t) \rfloor; \frac{\alpha}{i}) & \text{if } \lfloor \tilde{n}^c(0) - \tilde{n}^c(t) \rfloor > 0, \\ B(1; 0) & \text{otherwise,} \end{cases}$$

where

$$\frac{\alpha}{i} := \mathbf{P} \frac{\tilde{n}_i^c}{\sum_{j=1}^{|V|} \tilde{n}_j^c}$$

and $B(n; p)$ is a binomial distribution. In other words, if the number of active particles is less than the initial number of active particles, then it is performed $\lfloor \tilde{n}^c(0) - \tilde{n}^c(t) \rfloor$ trials with probability $\frac{\alpha}{i}$ at generating a new particle in v_i .

It follows that the expected number of new particles that belong to class c in v_i at time t is

$$\begin{cases} \frac{\alpha}{i} \lfloor \tilde{n}^c(0) - \tilde{n}^c(t) \rfloor & \text{if } \lfloor \tilde{n}^c(0) - \tilde{n}^c(t) \rfloor > 0, \\ 0 & \text{otherwise.} \end{cases}$$

The information of the current directed domination $\tilde{n}_{ij}^c(t)$ determines the class of particles that dominates each edge in the system. The edges of the network are grouped in sets by the class that dominates them. For each class c , the subset of edges $\mathcal{E}^c(t) \in \mathcal{E}$ is

$$\mathcal{E}^c(t) := \{ij \mid \arg \max_q \tilde{n}_{ij}^q(t) + \tilde{n}_{ji}^q(t) = c\}.$$

Define the unweighted subnetwork

$$\mathbf{G}^c(t) := (\mathcal{V}; \mathcal{E}^c(t)) \quad (1)$$

to be the *unfolding* of network $\mathbf{G}$ according to class c at time t . This subnetwork can be interpreted as a subspace with the most relevant relationships for a given class. The available information in these subnetworks will be utilized for the study of community detection and data clustering.

Next, a formal modeling of this dynamical system is presented.

2.2 Mathematical Modeling

Formally, we define Edge Domination System as a dynamical system $\tilde{X}(t)$. Let $\tilde{n}_i^c(t)$ be the number of active particles that belong to class c in v_i at time t . The

internal state of this dynamical system is

$$\tilde{\mathbf{X}}(\mathbf{t}) := [\mathbf{n}^c(\mathbf{t})]^T, \quad (2)$$

where

$$\mathbf{n}^c(\mathbf{t}) := \tilde{\mathbf{n}}_i^c(\mathbf{t}) \quad_i.$$

Let $\tilde{\mathbf{g}}_i^c(\mathbf{t})$ and $\tilde{\mathbf{a}}_i^c(\mathbf{t})$ be, respectively, the number of particles generated and absorbed in $\mathbf{v}_i$ at time $\mathbf{t}$. The evolution function $\tilde{\cdot}$ of the dynamical system is

$$\tilde{\cdot} : \tilde{\mathbf{n}}_i^c(\mathbf{t} + 1) = \tilde{\mathbf{n}}_i^c(\mathbf{t}) + \sum_j \tilde{\mathbf{n}}_{ji}^c(\mathbf{t} + 1) - \tilde{\mathbf{n}}_{ij}^c(\mathbf{t} + 1) + \tilde{\mathbf{g}}_i^c(\mathbf{t} + 1) - \tilde{\mathbf{a}}_i^c(\mathbf{t} + 1).$$

Intuitively, the number $\tilde{\mathbf{n}}_i^c$ of active particles that are in a vertex is the total number of particles arriving minus the number of particles leaving and particles that have been absorbed. Additionally, there is a term for the number $\tilde{\mathbf{g}}_i^c$ of generated particles. Values $\tilde{\mathbf{n}}_{ij}^c$, $\tilde{\mathbf{g}}_i^c$, and $\tilde{\mathbf{a}}_i^c$ are obtained *stochastically* according to the dynamics of walking, absorption, and generation. The initial state of the system is given by an arbitrary number $\tilde{\mathbf{n}}_i^c(0)$ of initial active particles and $\tilde{\mathbf{n}}_{ij}^c(0) = 0$.

In order to achieve the desirable network unfolding, it is necessary to average the results of several simulations of the system with a very large number of initial particles $\tilde{\mathbf{n}}_i^c(0)$. However, the computational cost of a such simulation is very high. Alternatively, a system $\mathbf{X}(\mathbf{t})$ that achieves similar results in a *deterministic* manner can be modeled. This alternative system considers that exists an asymptotically infinite number of initial active particles.

2.3 Alternative Mathematical Modeling

Consider the dynamical system whose internal state is

$$\mathbf{X}(\mathbf{t}) := \begin{matrix} 2 \\ 4 \end{matrix} \begin{matrix} \mathbf{n}^c(\mathbf{t}) = \mathbf{n}_i^c(\mathbf{t}) \\ \mathbf{N}^c(\mathbf{t}) = \mathbf{n}_{ij}^c(\mathbf{t}) \end{matrix} \begin{matrix} 3 \\ 5 \end{matrix}_i$$

that is a nonlinear Markovian dynamical system with the deterministic evolution function

$$\begin{cases} \mathbf{n}^c(\mathbf{t} + 1) = \mathbf{n}^c(\mathbf{t}) \times \mathbf{P}^c(\mathbf{X}(\mathbf{t})) (\mathbf{n}^c(\mathbf{t}) \times \mathbf{P}^c(\mathbf{X}(\mathbf{t})) \cdot \mathbf{1})^{-1} \\ \mathbf{N}^c(\mathbf{t} + 1) = \text{diag } \mathbf{n}^c(\mathbf{t}) \times \mathbf{P}^c(\mathbf{X}(\mathbf{t})), \end{cases} \quad (3)$$

where

$$\begin{aligned} \mathbf{P}^c(\mathbf{X}) &:= \mathbf{p}_{ij}^c(\mathbf{X}) \quad_{ij}, \\ \mathbf{p}_{ij}^c(\mathbf{X}) &:= \mathbf{P} \frac{w_{ij}}{\sum_{k=1}^{|V|} w_{ik}} \quad \mathbf{1} - \quad_{ij}^c(\mathbf{X}) \quad, \end{aligned} \quad (4)$$

$$\mathbf{n}_{ij}^c(\mathbf{X}) := \begin{cases} 1 - \mathbf{P}_{q=1}^K \frac{\mathbf{n}_{ij}^c + \mathbf{n}_{ji}^c}{\mathbf{n}_{ij}^q + \mathbf{n}_{ji}^q} & \text{if } \mathbf{n}_{ij}^c + \mathbf{n}_{ji}^c > 0, \forall \mathbf{c}, \\ \frac{1}{K} & \text{otherwise.} \end{cases} \quad (5)$$

The initial state of system $\mathbf{X}$ is given by an arbitrary discrete *distribution* $\mathbf{n}^c(0)$ of initial active particles and $\mathbf{n}_{ij}^c(0) = 0$. This system performs an unfolding in $\mathcal{O}(K |\mathcal{E}|)$ operations.

2.4 Community Detection

In the context of community detection, we simulate the alternative model with K classes—here K is also the number of communities—up to time T and work with the unweighted subnetworks $\mathbf{G}^c(T)$ in order to determine the community structure within the network $\mathbf{G}$. A vertex is said to belong to community $\mathbf{c}$ if the density of edges in a neighborhood is higher in $\mathbf{G}^c(T)$ than in every other unfolding. Formally, the community to which $\mathbf{v}_j$ belongs to is

$$\mathbf{y}_j = \arg \min_{c \in \{1, \dots, K\}} \mathbf{S}_{c,j}^{\text{density}},$$

where $\mathbf{S}_{c,j}^{\text{density}} \in [0; 1]$ is the grade of membership of $\mathbf{x}_j$ on community $\mathbf{c}$. Let $\mathcal{N}_{c,j}^o$ be the neighborhood of a given order o of $\mathbf{v}_j$ in the unfolding $\mathbf{G}^c(T)$. Denote the number of edges in this neighborhood as $\mathcal{E}(\mathcal{N}_{c,j}^o)$. Thus, the grade of membership by density of $\mathbf{v}_j$ in unfolding $\mathbf{G}^c(T)$ is

$$\mathbf{S}_{c,j}^{\text{density}} := \mathbf{P}_{q=1}^K \frac{\mathcal{E}(\mathcal{N}_{c,j}^o)}{\mathcal{E}(\mathcal{N}_{q,j}^o)}. \quad (6)$$

2.5 Data Clustering

For the application of data clustering, the community structure itself can be the dataset partition. This might be enough for data that can be easily partitioned. If the data, however, are nonlinearly distributed then a community structure with the same number of communities as of classes might not yield satisfactory results. In such cases, disconnected groups of vertices within a community can be the unfolding result. Depending on the definition of community in networks, treating a disconnected group as a single community is an inconsistency. For the problem of data clustering, I assume a cluster of data point is not divided in two by a second cluster, and therefore this is the case where disconnected groups within communities is an undesirable behavior. One way to circumvent this, is to superestimate the actual number of clusters by imposing a little larger number K of classes of particles. After the end of the process, the community structure is reduced to $\mathbf{C}$ communities, resulting thus in the final data clustering.

The reduction step tries to combine adjacent communities in such a way that the network's modularity is maximized. The modularity of a community

structure within a network is a quality function of how well-defined are the communities [15]. The intuition of modularity is that there should be more links within a community than links among communities. In this aspect, the goal in the first step of the proposed technique is to simplify the problem by creating an initial number of communities and then, in the second step, the problem is treated as an optimization of the network's modularity.

Since there are $\binom{K}{C}$ possible combinations of communities for the reduction, in this paper a greedy construction of the dendrogram is performed. Starting from the initial community structure given by $\mathbf{y}_j$, the merging of two adjacent communities that yields larger modularity is sought out. For a partition with K communities, the algorithm tries all possible merges of adjacent communities that result in $K - 1$ communities. As a consequence, up to $\binom{K}{K-1}$ merges and modularity evaluations are performed. Next, by picking the partition with highest modularity, the next best merging is determined by performing up to $\binom{K-1}{K-2}$ evaluations. Repeat this until there are C communities. Thus, the maximum number of evaluated merges is

$$\sum_{i=C}^{K-1} \binom{i+1}{i} = \frac{1}{2} (K - C) (K + C + 1) = \mathcal{O} K^2 - C^2 = \mathcal{O} K^2 .$$

Since only sparse networks do present a community structure [8], the number of adjacent communities and, hence, the number of merge evaluations are considerably lower. Together with the process, the entire technique is simulated in the order of $\mathcal{O}(K |\mathcal{E}| + K^2)$.

3 Computer Simulations

In order to assess the proposed model, computer simulations were performed. This section starts with simulations of the proposed model for the problem of community detection in complex networks. Afterwards, the data clustering technique is applied for both artificial and real-world datasets.

3.1 Simulation for Community Detection

A real-world network borrowed from social science literature is here utilized for evaluating the ability to detect a community structure in a network. The chosen network is the Zachary's karate club [16], a very well-known network that has become a benchmark test for community detection methods. This network describes the relationship of 34 members of a US university club in the 1970s. Each member is a node in the network, and a link between two nodes indicates whether two members know each other. As the time passed an internal conflict between the president, John A., and the instructor, Mr. Hi, divided the group between members that supported the president and the members that supported the instructor. The goal is to split the club members in a group that supports the president and a group that supports the instructor.

In Figure 1, the process' result for community detection on the karate club is shown. In edges, the colors represent the class of particles that dominated the edge at the end of simulation. In vertices, the color is the result of community detection by using the density information. The color of vertices represent the community structure detected by the proposed model. Even though this is an unweighted network (the links denote only whether two members know each other) the proposed model was able to detect the correct group of members by using the weight of 1 for all links.

Fig. 1. Community detection result for the proposed model on Zachary's karate club network. Red and blue colors denote a group. For edges the color is the class of particles that dominates the edge at the end of the simulation. For vertices the color is the community to which a vertex has been grouped. All members were correctly grouped.

3.2 Simulations for the Data Clustering Problem

Six techniques with different approaches to tackle the problem of data clustering were selected to compare with the proposed technique: K -means [10], Fuzzy c -means [3], Hierarchical DBSCAN* with Density-Based Clustering Validation index (DBSCAN + DBCV) [4, 14], Chameleon [13], Expectation Maximization algorithm with parameterized Gaussian mixture models [9], and an adapted greedy algorithm based on modularity of networks [5].

In order to evaluate the quality of a partition found by an algorithm, the measure of Adjusted Rand Index [11] was calculated for each generated partition. This measure index is defined in the range $[-1; 1]$ where values close to 1 indicate that the partition is close to the a prior knowledge of the dataset's clusters and values close to zero indicate the partition is likely to be as good as an algorithm that partitions the dataset at random.

Table 1. Adjusted Rand Index values for simulations on four artificial datasets.

	Banana	Highleyman	Lithuanian	Spirals	Avg. Rank
<i>K</i> -means	.2429	.2617	-.0016	-.0020	6.2
Fuzzy <i>c</i> -means	.2442	.3201	-.0017	-.0019	5.8
HDBSCAN + DBCV	.4714	.2085	.7024	.2507	4
Chameleon	.9215	.4348	.9343	.0119	3
Expectation Maximization	.3304	.7977	-.0015	-.0020	4.5
Modularity	.3510	.5033	.4259	1.0000	3
<i>Proposed technique</i>	.9408	.7164	.9538	1.0000	1.5

Three techniques rely on stochastic initialization (*K*-means, fuzzy *c*-means, and Expectation Maximization)—for these, the reported results are the average over 50 runs for each dataset. For the simulations all techniques had prior knowledge of the number of classes in the data clustering problem. That way, the reported results assess the ability of a technique to perform the dataset partitioning given that the number of clusters is known. In case of Chameleon and the proposed technique, the algorithms start with an possibly larger number of clusters/communities before agglomerating to the desirable number. For Chameleon, the initial number of clusters varied within $\{\mathbf{C}; \mathbf{C}+1; \dots; |\mathcal{X}|\}$, where $|\mathcal{X}|$ is the number of data points in the dataset. For the proposed technique, the competition parameter is fixed at $\alpha = 0.5$ with the number of classes of particles varying in $\{2; 5; \dots; 30\}$. Moreover, modularity and the proposed technique are graph-based methods, which implies the dataset input must be in a graph representation. The *k*-NN method was used, varying $\mathbf{k} \in \{1; 2; \dots; 30\}$.

Simulations on Artificial Datasets Four artificial datasets were generated with PRTTools framework [7]. The datasets are formed by points in two-dimensional space equally splitted into two classes that form two clusters to be detected by the algorithm. Except for the dataset of spirals that contain 500 points, the other four datasets are formed by 600 points. Because the datasets are not in a network representation, the *k*-Nearest Neighbor (*k*-NN) graph construction method is employed to obtain a graph that is the system’s network. In the constructed network, a vertex represents a data point, and it connects to its $\mathbf{k}$ nearest points determined by Euclidean distance.

Figures 2 to 5 show the generated dataset, the result obtained by application of *K*-means, and the result of the proposed technique. In Table 1, the adjusted rand index of each technique is shown. The last column is the average ranking position that a technique obtained. The technique is able to correctly detect the shape distribution in the four datasets. The graph representation naturally determine how particles flow through the network. If the clusters have a slow interconnectivity, as the case of spirals (see Figure 5), both modularity and the proposed, which both are network-based techniques, are able to correctly partition the data without any mistake. But the network is not the only important aspect. For instance, modularity could not partition data points of different

classes in overlapping regions, as the case of Highleyman dataset, whereas the proposed was able to find a disconnected community (see Figure 3).

Fig. 2. Banana-shaped distribution dataset with two classes. (a) Result obtained by K-means. (b) Result obtained by the proposed technique.

Fig. 3. Highleyman dataset with two classes. (a) Result obtained by K-means. (b) Result obtained by the proposed technique.

Fig. 4. Lithuanian dataset with two classes. (a) Result obtained by K-means. (b) Result obtained by the proposed technique.

Fig. 5. Spirals dataset with two classes. (a) Result obtained by K-means. (b) Result obtained by the proposed technique.

Simulations on Real Datasets Ten real-world datasets were selected from the UCI repository [1]. A brief description of the datasets is given in Table 2. Results of adjusted rand index for simulations of these datasets is shown in Table 3. In this table, the last column is the average ranking position of each technique over all datasets. The ranking is sorted accordingly to the adjusted rand index. A higher index indicates that a technique has partitioned the dataset in a way more similar to the prior knowledge. Thus, the technique that obtained the highest index for a dataset has a ranking position of 1 for that dataset; the second highest index is associated to a ranking position of 2, and so on.

The parameter combination the obtained the highest adjusted rand index is shown in Table 4. Notably for all real-world datasets, simulating the system with a higher value for the number $\mathbf{K}$ of classes of particles, and thereafter in a second step merging to $\mathbf{C}$ communities has given the best results.

Table 2. Description of the real-world datasets, with the number of data points, the number of dimensions in the attribute space, and the number of clusters the data points are partitioned.

	Instances	Dimensions	Classes
Breast cancer	699	9	2
Car evaluation	1728	6	4
Credit approval	690	15	2
Contraceptive method	1473	9	3
Glass	214	9	6
Ionosphere	351	34	2
Iris	150	4	3
Vowel	90	10	11
Wine	178	13	3
Seeds	270	7	3

It is expected that the modularity algorithm do not have a significant difference, since the intuition of the present technique is to simplify the problem by finding some communities in a first step for later optimize the partitions according to a quality function, such as the modularity. An interesting aspect of the reported results is that the average rank of presented technique do not vary much accordingly to the dataset domain, which could mean the technique does not have a great bias of domain. Nevertheless, the artificial toy datasets have a show case where the technique is able to get a better partitioning by loosing the definition of community and allowing disconnected communities.

Chameleon has a very similar insight as the proposed technique in the way it approaches the data clustering problem. First, Chameleon splits the network representation of the dataset into small groups of vertices. Afterwards, it agglomerates these groups into larger ones until obtained the desired number of clusters. One problem is the step of defining the small groups. If the small groups of vertices are not well defined—such as in cases of data with overlapping regions—, the agglomerative step will result in a poor data clustering. Conversely, if well-defined then it obtains good results as in the case of the Glass and Ionosphere datasets, which were the two datasets the Chameleon obtained a higher value for adjusted rand index.

In order to evaluate whether the average ranking position of the techniques are significantly different, the average rankings are examined in a statistical manner. The methodology applied here follows a procedure described in Demšar [6] and Japkowicz and Shah [12]. The first test, Friedman test compares the techniques under the null-hypothesis that all the techniques are equivalent and so their average rank positions. According to Demšar [6], since we have $\mathbf{N} = 10$ datasets and $\mathbf{k} = 7$ techniques, we have the degrees of freedom $df1 = \mathbf{k} - 1 = 6$

Table 3. Comparison of the Adjusted Rand Index of seven different techniques on 10 real-world datasets.

	Breast cancer	Car eval.	Credit approval	Contr. method	Glass	Ion.	Iris	Vowel	Wine	Seeds	Avg. Rank
<i>K</i> -means	.7302	.0294	.2389	.0215	.1610	.1776	.6540	.1736	.8582	.7049	4.8
Fuzzy <i>c</i> -means	.7305	.0307	.3725	.0242	.1632	.1727	.7287	.0892	.8498	.7266	4.3
HDBSCAN+DBCV	.2556	.1313	.0794	.0236	.2575	.7030	.5657	.0814	.3385	.4303	5.4
Chameleon	.7192	.1496	.1653	.0253	.2918	.6767	.6844	.1949	.8249	.7436	3.6
E. Maximization	.6955	.0367	.1987	.0112	.1571	.1547	.9222	.1541	.9472	.6671	4.8
Modularity	.4474	.1872	.1734	.0329	.2118	.0708	.9038	.2505	.8858	.8125	3.5
<i>Proposed technique</i>	.7930	.1880	.4890	.0433	.2377	.3057	.9222	.2259	.9488	.8377	1.6

Table 4. Best parameter combinations for both artificial and real-world datasets. The best combination is associated to the highest adjusted rand index obtained. The listed parameters are the k -NN, the number of classes of particles, and the order of the neighborhood considered for grouping vertices in communities.

	k -NN	K Neighborhood
Banana shape	4 2	4
Lithuanian	7 24	2
Highleyman	11 8	4
Spirals	5 18	1
Breast cancer	4 30	3
Car evaluation	16 11	4
Credit approval	28 27	3
Contraceptive method	10 11	2
Glass	28 8	3
Ionosphere	9 30	2
Iris	8 5	1
Vowel	7 30	2
Wine	8 24	3
Seeds	4 21	3

and $df2 = (N - 1)(k - 1) = 54$, resulting in the critical value $F(6; 54) \approx 2.27$. With such values, we get a Friedman statistic $F_F \approx 4.88$. Since $F_F > F(6; 54)$ the null-hypothesis is rejected at a 5% significance level, allowing us to proceed with a post-hoc test.

Fix a significance level of 5% for the post-hoc. The interest is at comparing the proposed model with the other six techniques. For that the Bonferroni–Dunn test is employed, because there are multiple techniques over multiple datasets, and only the difference between a fixed technique and all others is of interest [12]. Therefore, six null-hypothesis of the proposed technique’s average rank being equivalent to an other technique are tested. To accomplish this, the t statistic between a fixed and an other technique must be by at least a Critical Difference (CD). If they differ that much, then the null-hypothesis is rejected and therefore there is a significant statistical difference between the average ranking position of the two techniques. For the setting in this paper, we get a $CD \approx 3.33$. The proposed technique has significant statistical difference when compare with K -means, Fuzzy c -means, HDBSCAN + DBCV, and Expectation Maximization. The techniques Chameleon and Modularity do not present significant statistical difference. But, nevertheless, these two together with the proposed technique,

Fig. 6. Visualization of post-hoc Bonferroni-Dunn test. Fixing the proposed technique as the control, the average ranking of the other six techniques were compared. All techniques with average rankings outside the bold-marked interval are significantly different from the proposed technique, with a significance level of 5%.

put network-based methods in a good position at data clustering problems. A visualization of the post-hoc test, as suggested in [6], is shown in Figure 6.

4 Conclusion

In this paper, a novel model for particle competition in complex networks is presented. Particles flow through the network and their presence dominates an edge. The information is then used to determine the pertinence of vertices in communities of the network. A second step allows the maximization of the modularity by locally maximizing the modularity of the community structure. This final structure is used to the problem of data clustering.

Simulations on both artificial and real-world datasets show that the proposed technique performs well and have a significant better performance under the condition that the algorithms have prior knowledge of the number of clusters. Furthermore, the technique has a deterministic alternative model with a low computational complexity.

The information of dominance of edges have a potentially higher granularity information about the interaction network formed by the graph representation of the dataset. This information might allow to grasp more properties of the process and consequently help into find a better approach to the local maximization of the community structure.

Finally, in a data clustering setting it is not uncommon to run algorithms on datasets with noise and no prior knowledge of the number of clusters. The proposed technique here present could be extended for such applications.

References

- [1] D J Newman A. Asuncion. *UCI Machine Learning Repository*. 2007.
- [2] Alain Barrat, Marc Barthlemy, and Alessandro Vespignani. *Dynamical Processes on Complex Networks*. 1st. New York, NY, USA: Cambridge University Press, 2008. ISBN: 0521879507, 9780521879507.
- [3] James C. Bezdek, Robert Ehrlich, and William Full. “FCM: The fuzzy c-means clustering algorithm”. In: *Computers & Geosciences* 10.2–3 (1984), pp. 191–203. ISSN: 0098-3004. DOI: 10.1016/0098-3004(84)90020-7.

- [4] Rjgb Campello, Davoud Moulavi, and Joerg Sander. “Density-based clustering based on hierarchical density estimates”. In: *Advances in Knowledge Discovery . . .* (2013), pp. 160–172.
- [5] Aaron Clauset, M. E J Newman, and Cristopher Moore. “Finding community structure in very large networks”. In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 70.6 2 (2004), pp. 1–6. ISSN: 15393755. DOI: 10.1103/PhysRevE.70.066111. arXiv: 0408187 [cond-mat].
- [6] Janez Demšar. “Statistical Comparisons of Classifiers over Multiple Data Sets”. In: *J. Mach. Learn. Res.* 7 (2006), pp. 1–30.
- [7] R.P.W. Duin et al. *PR-Tools4.1, A Matlab Toolbox for Pattern Recognition*. 2007.
- [8] Santo Fortunato. “Community detection in graphs”. In: *Physics Reports* 486.3-5 (2010), pp. 75–174. ISSN: 03701573. DOI: 10.1016/j.physrep.2009.11.002. arXiv: 0906.0612.
- [9] Chris Fraley et al. “mclust Version 4 for R : Normal Mixture Modeling for Model-Based Clustering , Classification , and Density Estimation”. In: *Technical Report 597, University of Washington* (2012).
- [10] J. a. Hartigan and M. a. Wong. “A K-Means Clustering Algorithm”. In: *Journal of the Royal Statistical Society* 28.1 (1979), pp. 100–108.
- [11] Lawrence Hubert and Phipps Arabie. “Comparing partitions”. In: *Journal of Classification* 2.1 (1985), pp. 193–218. ISSN: 01764268. DOI: 10.1007/BF01908075.
- [12] Nathalie Japkowicz and Mohak Shah. *Evaluating Learning Algorithms: A Classification Perspective*. 2011, p. 423. ISBN: 9780521196000. DOI: <http://dx.doi.org/10.1017/CB09780511921803>. URL: <http://books.google.com/books?id=VoWIIQKVzR4C>.
- [13] George Karypis, Eui-Hong Han, and Vipin Kumar. “Chameleon: hierarchical clustering using dynamic modeling”. In: *Computer* 32.8 (1999), pp. 68–75. ISSN: 00189162. DOI: 10.1109/2.781637.
- [14] Davoud Moulavi, Pa Jaskowiak, and Rjgb Campello. “Density-Based Clustering Validation”. In: *Dbs.Ifj.Lmu.De i* (2014).
- [15] M. E. J. Newman. “Fast algorithm for detecting community structure in networks”. In: *Physical Review E* 69.6 (June 2004), p. 066133. ISSN: 1539-3755. DOI: 10.1103/PhysRevE.69.066133.
- [16] W Zachary. “An Information Flow Model for Conflict and Information Fission in Small Groups”. In: *Journal of anthropological research* 33.4 (1977), pp. 452–473. ISSN: 00917710. DOI: 10.2307/3629752.

CONCLUSIONS

The developed work in this dissertation solved machine learning problems with complex networks and dynamical processes. The main goal was to develop a transductive technique for semi-supervised and unsupervised learning problems, with a new approach to particle competition in a two-step design process. First, a model was created as a stochastic process with interactions determined at an individual level. Afterward, an equivalent deterministic system was found at a populational level.

In the semi-supervised technique, the classification process is decomposed into three steps. First, the input data are mapped into an interaction network. Then, the model of particle competition is simulated on this network, leading to the unfolding of components of the network. At this stage, teams of particles compete to dominate the edges of the network. Whenever a particle passes through an edge, the team's domination over that edge is reinforced while decreasing the rivals' dominance. Such competition process is modeled with three dynamics—of walking, absorption, and production—inspired by behaviors observed in nature. The dominated territories become labels, which are used to signal to which component each edge belongs.

After modeling the dynamics separately, the deterministic equivalent system brought advantages over its stochastic counterpart. The time complexity of the deterministic one does not depend on the number of particles; it benefited from better results when considering a continuously varying number of initial particles. Besides, the transductive semi-supervised learning technique becomes stable and with a subquadratic order of complexity.

In the unsupervised technique, the lack of initially labeled data allowed a simplified model at the competition step but required a more sophisticated community optimization

process. With the dynamics of particle production changed to generated particles according to their team distribution, the modified model used a simpler particle production system. In contrast, a larger number of teams are required before an agglomerative step of the unfolded components to optimize the modularity of the community structure.

6.1 Future works

The technique is constituted by components that may be replaced by those specific to an application. We have aimed at building a network representation of data on a metric space with a simple graph construction method—the k -nearest neighbors. The unweighted graph is used as the interaction network of the particle competition to perform the network component unfolding. Not only a different graph construction method can be studied, but the weights of the edges may lead to better results, although modifications on the competition and the network unfolding model need to be elaborated.

Regarding the classification step, which is done by the network unfolding, the current method is a straightforward decomposition of unweighted subgraphs. We have observed the existence of overlap of components during the network unfolding process, which remain to be explored. Since there is a competition process, and teams of particles try to dominate the territories of rivals, instead of a crisp division of network components, the unfolding may be regarded as a fuzzy pertinence.

Such modifications may lead to increasing the model's complexity, but the other way round is also an interesting path. Since simpler models may require less computational time without relinquishing too much learning performance, some aspects of the technique may be further simplified. With a thorough investigation of the technique's behavior, a parameter analysis should help better understand which scenarios are adequate for some combination of parameters.

As regards the unsupervised model, currently, only data clustering problems have been studied as community detection problems. The initialization of the particle distributions remains a problem because the current methods may lead to unconnected communities when the number of initial particle teams is too low for the dataset. While this has been dealt with by applying a preprocessing step to the numerical data, such initialization would not be possible on data that is already in a network structure.

6.2 List of publications

The results have contributed to machine learning, with results reported published in form of scientific papers in international journals and conferences.

6.2.1 Publications in international journals

Urio, P. R., Verri, F. A. N., and Zhao, L. (2016). “Semi-Supervised Classification by Particle Competition in Complex Network’s Edges”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 30.09, p. 1660006. DOI: 10.1142/S0218001416600065. Included in Chapter 3.

Verri, F. A. N., Urio, P. R., and Zhao, L. (2016b). “Network Unfolding Map by Vertex-Edge Dynamics Modeling”. In: *IEEE Transactions on Neural Networks and Learning Systems* PP.99, pp. 1–14. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2626341.

Verri, F. A. N., Urio, P. R., and Zhao, L. (2016a). “Advantages of Edge-Centric Collective Dynamics in Machine Learning Tasks”. In: *Journal of Applied Nonlinear Dynamics*.

6.2.2 Publications in international conference proceedings

Urio, P. R., Verri, F. A. N., and Zhao, L. (2015b). “Semi-supervised learning by edge domination in complex networks”. In: *11th International Conference on Natural Computation (ICNC)*, pp. 514–519. DOI: 10.1109/ICNC.2015.7378041. Included in Chapter 2.

Urio, P. R., Verri, F. A. N., and Zhao, L. (2015a). “Features of edge-centric collective dynamics in machine learning tasks”. In: *6th International Conference on Nonlinear Science and Complexity*. Vol. 1. São José dos Campos, SP. Included in Chapter 4.

Urio, P. R. and Liang, Z. (2017). “Data clustering with edge domination in complex networks”. In: *ArXiv e-prints, submitted to BRACIS 2017*. arXiv: 1705.05494. ArXiv preprint included in Chapter 5.

BIBLIOGRAPHY

- Alpaydin, E. (2010).** *Introduction to Machine Learning*. 2nd ed. MIT press, p. 584. ISBN: 9780262012430. (Cited on page 16.)
- Belkin, M. and Niyogi, P. (2003).** “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation”. In: *Neural Computation* 15.6, pp. 1373–1396. ISSN: 0899-7667. DOI: 10.1162/089976603321780317. (Cited on page 17.)
- Blum, A. and Chawla, S. (2001).** “Learning from labeled and unlabeled data using graph mincuts”. In: *International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 19–26. (Cited on page 17.)
- Bousquet, O., Chapelle, O., and Hein, M. (2004).** “Measure based regularization”. In: *Advances in Neural Information Processing Systems*, pp. 1221–1228. ISSN: 10495258. (Cited on page 17.)
- Chapelle, O., Schölkopf, B., and Zien, A. (2006).** “Semi-Supervised Learning”. In: *Semi-Supervised Learning*. 1st ed. The MIT Press. Chap. Analysis o, pp. 377–393. ISBN: 0262514125. (Cited on pages 16, 17.)
- Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996).** “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*. Ed. by Simoudis, E., Han, J., and Fayyad, U. M., pp. 226–231. ISBN: 1577350049. DOI: 10.1.1.71.1980. (Cited on page 17.)
- Flach, P. (2012).** *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. New York, NY, USA: Cambridge University Press, p. 409. ISBN: 9781107422223. (Cited on page 16.)
- Fortunato, S. (2010).** “Community detection in graphs”. In: *Physics Reports* 486.3-5, pp. 75–174. ISSN: 03701573. DOI: 10.1016/j.physrep.2009.11.002. arXiv: 0906.0612. (Cited on page 17.)
- Hastie, T., Tibshirani, R., and Friedman, J. (2009).** *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Ed. by Bickel, P., Diggle, P., Fienberg, S. E., Gather, U., Olkin, I., and Zeger, S. 2nd ed. Vol. 1. Springer Series in Statistics, p. 767. ISBN: 9780387848570. DOI: 10.1007/b94608. (Cited on page 17.)
- Jain, A. K. and Dubes, R. C. (1988).** *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., p. 304. ISBN: 0-13-022278-X. (Cited on page 17.)

- Kapitaniak, M., Strzalko, J., Grabski, J., and Kapitaniak, T. (2012).** “The three-dimensional dynamics of the die throw”. In: *Chaos* 22.4. ISSN: 10541500. DOI: 10 . 1063/1.4746038. (Cited on page 13.)
- Karp, R. M. (1991).** “An introduction to randomized algorithms”. In: *Discrete Applied Mathematics* 34.1-3, pp. 165–201. ISSN: 0166218X. DOI: 10 . 1016 / 0166 – 218X(91) 90086–C. (Cited on page 13.)
- Karypis, G., Han, E.-H., and Kumar, V. (1999).** “Chameleon: Hierarchical clustering using dynamic modeling”. In: *Computer* 32.8, pp. 68–75. ISSN: 00189162. DOI: 10 . 1109/2.781637. (Cited on page 17.)
- Kononenko, I. and Kukar, M. (2007).** *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publishing Limited, p. 480. ISBN: 9781904275213. (Cited on page 17.)
- Mitchell, T. M. (1997).** *Machine Learning*. 1st ed. McGraw-Hill, Inc., p. 421. ISBN: 9780070428072. (Cited on page 16.)
- Newman, M. (2010).** *Networks: An Introduction*. 1st ed. New York, NY, USA: Oxford University Press Inc., p. 720. ISBN: 978-0199206650. (Cited on page 17.)
- Quiles, M. G., Zhao, L., Alonso, R. L., and Romero, R. a. F. (2008).** “Particle competition for complex network community detection.” In: *Chaos (Woodbury, N.Y.)* 18.3, p. 033107. ISSN: 1089-7682. DOI: 10 . 1063/1.2956982. (Cited on page 18.)
- Saul, L., Weinberger, K., Ham, J., and Sha, F. (2006).** “Spectral methods for dimensionality reduction”. In: *Semi-supervised Learning*. Ed. by Chapelle, O., Schölkopf, B., and Zien, A. 1st ed. The MIT Press, pp. 293–306. ISBN: 9780262033589. DOI: 10 . 1234/12345678. (Cited on page 17.)
- Shannon, C. E. (1948).** “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.July 1928, pp. 379–423. ISSN: 07246811. DOI: 10 . 1145 / 584091 . 584093. arXiv: 9411012 [chao-dyn]. (Cited on page 15.)
- Silva, T. C. and Zhao, L. (2012a).** “Network-based stochastic semisupervised learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* 3, pp. 451–466. ISSN: 2162237X. DOI: 10 . 1109/TNNLS . 2011 . 2181413. (Cited on page 18.)
- Silva, T. C. and Zhao, L. (2012b).** “Stochastic competitive learning in complex networks.” In: *IEEE Transactions on Neural Networks and Learning Systems* s 23.3, pp. 385–98. ISSN: 2162-2388. DOI: 10 . 1109/TNNLS . 2011 . 2181866. (Cited on page 18.)
- Strogatz, S. H. (2001).** “Exploring complex networks”. In: *Nature* 410.March, pp. 268–276. (Cited on page 14.)
- Subramanya, A. and Talukdar, P. (2014).** *Graph-Based Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. ISBN: 9781627052023. (Cited on page 17.)
- Szummer, M. and Jaakkola, T. (2001).** “Partially labeled classification with Markov random walks”. In: *Advances in Neural Information Processing Systems*. MIT Press, pp. 945–952. (Cited on page 17.)

- Tenenbaum, J. B., Silva, V. D., and Langford, J. C. (2000).** “A Global Geometric Framework for Nonlinear Dimensionality Reduction”. In: *Science* 290.December, pp. 2319–2323. ISSN: 0036-8075. DOI: 10.1126/science.290.5500.2319. (Cited on page 17.)
- Urio, P. R. and Liang, Z. (2017).** “Data clustering with edge domination in complex networks”. In: *ArXiv e-prints, submitted to BRACIS 2017*. arXiv: 1705.05494. (Cited on page 71.)
- Urio, P. R., Verri, F. A. N., and Zhao, L. (2015a).** “Features of edge-centric collective dynamics in machine learning tasks”. In: *6th International Conference on Nonlinear Science and Complexity*. Vol. 1. São José dos Campos, SP. (Cited on page 71.)
- Urio, P. R., Verri, F. A. N., and Zhao, L. (2015b).** “Semi-supervised learning by edge domination in complex networks”. In: *11th International Conference on Natural Computation (ICNC)*, pp. 514–519. DOI: 10.1109/ICNC.2015.7378041. (Cited on page 71.)
- Urio, P. R., Verri, F. A. N., and Zhao, L. (2016).** “Semi-Supervised Classification by Particle Competition in Complex Network’s Edges”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 30.09, p. 1660006. DOI: 10.1142/S0218001416600065. (Cited on page 71.)
- Vapnik, V. N. (1995).** *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., p. 314. ISBN: 0-387-94559-8. (Cited on page 16.)
- Vapnik, V. N. (1998).** *Statistical Learning Theory*. 1st ed. Wiley-Interscience, p. 768. ISBN: 9780471030034. (Cited on page 16.)
- Verri, F. A. N., Urio, P. R., and Zhao, L. (2016a).** “Advantages of Edge-Centric Collective Dynamics in Machine Learning Tasks”. In: *Journal of Applied Nonlinear Dynamics*. (Cited on page 71.)
- Verri, F. A. N., Urio, P. R., and Zhao, L. (2016b).** “Network Unfolding Map by Vertex-Edge Dynamics Modeling”. In: *IEEE Transactions on Neural Networks and Learning Systems* PP.99, pp. 1–14. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2626341. (Cited on page 71.)
- Wang, X. F. and Chen, G. (2003).** “Complex networks: Small-world, scale-free and beyond”. In: *IEEE Circuits and Systems Magazine* 3.1, pp. 6–20. ISSN: 1531-636X. DOI: 10.1109/MCAS.2003.1228503. (Cited on page 14.)
- Zhu, X. (2005).** *Semi-Supervised Learning Literature Survey*. Tech. rep. Computer Sciences, University of Wisconsin-Madison, p. 60. (Cited on page 17.)
- Zhu, X., Ghahramani, Z., and Lafferty, J. (2003).** “Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions”. In: *Twentieth International Conference on Machine Learning (ICML-2003)*. Washington, DC: AAAI Press, pp. 912–921. (Cited on page 17.)
- Zhu, X. and Lafferty, J. (2005).** “Harmonic mixtures: Combining mixture models and graph-based methods for inductive and scalable semi-supervised learning”. In: *Proceedings of the 22nd international conference on Machine learning*, pp. 1052–1059. DOI: 10.1145/1102351.1102484. (Cited on page 17.)