Random Walks Based Modularity: Application to Semi-Supervised Learning

Robin Devooght
IRIDIA, ULB
50 Av. Fr. Roosevelt
1050 Brussels, Belgium
robin.de@gmail.com

Amin Mantrach
Yahoo Labs
Avinguda Diagonal 177
08018 Barcelona, Spain
amantrac@yahoo-inc.com

Ilkka Kivimäki
ICTEAM, UCLouvain
Av. G. Lemaître, 4 B-1348
1348 Louvain-La-Neuve, Belgium
ilkka.kivimaki@uclouvain.be

Hugues Bersini
IRIDIA, ULB
50 Av. Fr. Roosevelt
1050 Brussels, Belgium
bersini@ulb.ac.be

Alejandro Jaimes
Yahoo Labs
Avinguda Diagonal 177
08018 Barcelona, Spain
ajaimes@yahoo-inc.com

Marco Saerens
ICTEAM, UCLouvain
Av. G. Lemaître, 4 B-1348
1348 Louvain-La-Neuve, Belgium
marco.saerens@uclouvain.be

ABSTRACT

Although criticized for some of its limitations, modularity remains a standard measure for analyzing social networks. Quantifying the statistical surprise in the arrangement of the edges of the network has led to simple and powerful algorithms. However, relying solely on the distribution of *edges* instead of more complex structures such as *paths* limits the extent of modularity. Indeed, recent studies have shown restrictions of optimizing modularity, for instance its resolution limit. We introduce here a novel, formal and well-defined modularity measure based on random walks. We show how this modularity can be computed from *paths* induced by the graph instead of the traditionally used *edges*. We argue that by computing modularity on paths instead of edges, more informative features can be extracted from the network. We verify this hypothesis on a semi-supervised classification procedure of the nodes in the network, where we show that, under the same settings, the features of the random walk modularity help to classify better than the features of the usual modularity. Additionally, the proposed approach outperforms the classical label propagation procedure on two data sets of labeled social networks.

Categories and Subject Descriptors

E.1 [**Data Structures**]: Graphs and networks; G.2.2 [**Graph Theory**]: Network problems, Path and circuit problems

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2567986.

Keywords

Modularity, random walk, social networks, semi-supervised learning, graph mining, statistical physics

1. INTRODUCTION

In the last decade, the modularity introduced by Newman and Girvan [25, 24] has been one of the most used measures for finding community structures in large networks [7]. Its success relies on multiple factors. First, its scalability; indeed, efficient algorithms using modularity, scaling up to very large graphs (more than millions of nodes), have offered good insights for social network analysis on real-world data [4, 5]. Second, different greedy algorithms have been proposed to automatically detect the number of communities, while various state-of-the-art algorithms still require to set a priori the unknown number of communities [7]. One other reason behind the large adoption of modularity is the common acceptance that by maximizing modularity we can obtain better community structures [14].

This large success has led a high number of studies to adopt modularity as a standard measure for community detection in various domains as social science, social media, social networks and web analysis [17]. Studies on modularity itself have been mostly concerned with new (greedy) algorithms that are able to maximize modularity scores on larger and larger networks [24, 4]. However, recent studies have pointed out the limitations of optimizing modularity, including the so-called resolution limit [8]. The multi-resolution modularity is a first attempt to partly overcome this limitation [29, 14].

In this work, based on the initial idea of modularity of detecting stochastic surprises in the arrangement of the network, we propose a novel way to extend modularity, as the number of *paths* (and not of edges) falling within groups minus the expected number in an equivalent network with *paths* placed at random (a mathematical definition is given further in Section 4). The main idea is to consider richer network structures, i.e., paths, rather than "simple" edges when computing modularity. We argue that by using paths one

can extract more informative community structures than by using edges.

To achieve this, we review the probabilistic interpretation of modularity (Section 3) based on a bag-of-edges space. Then, we introduce how by using random walks theory we can generalize the probabilistic interpretation of modularity to a richer Hilbert space: the bag-of-paths. This results in a novel modularity measure, defined on weighted and directed graphs, that captures longer dependencies in the network (Section 4). To validate our argument that, by working in a bag-of-paths space, we capture richer features on the network, we consider a semi-supervised classification problem of the nodes in the graph. For this purpose, we derive an algorithm — scaling on moderately large networks — that extracts the top k eigenvectors of the novel random walks based modularity matrix (Section 5). We test the extent to which these extracted features classify nodes better than when using the same features extracted from the usual modularity matrix. This is done on two labeled social networks publicly available.

The contributions of this work are the following:

- We introduce a novel, formal and well-defined random walks based modularity (Section 4);
- We present a new algorithm that scales on moderately large graphs for extracting the dominant eigenvectors of the introduced modularity matrix (Section 5);
- In a semi-supervised classification procedure of the nodes in the network, we show that the features of the random walks based modularity help to classify better than the features of the usual modularity on labeled social networks (Section 6).

2. RELATED WORK

These last years, modularity has been one of the most used measures for detecting communities in networks. Its success relies on different factors, one being its clear and simple statistic interpretation, another being the availability of simple greedy algorithms scaling up to really large graphs [4, 5]. Therefore, this measure has been the centerpiece of many recent studies in social media and network analysis [17].

The majority of papers on modularity itself have been concerned with new greedy algorithms working faster and providing a higher modularity [7]. This is mainly due to the accepted paradigm that a higher modularity means a better community structure. However, this last claim has been recently criticized in [8], where the authors pointed out the modularity resolution limit: modularity fails in identifying modules smaller than a certain scale. In the same direction, another comparative study has shown that modularity fails to identify clusters on benchmark graphs in comparison with other more effective methods [15]. To overcome the resolution limit, some authors adopted multi-resolution versions of modularity giving the possibility to tune arbitrarily the size of clusters from very large to very small [29, 2]. This approach has been more recently criticized in [14], where the authors underline the fact that real world networks are characterized by clusters of different granularity distributed according to a power law. The authors showed that multi-resolution modularity suffers from this coexistence of small and large clusters depending on the resolution setting.

In this work, we present a novel extension of modularity taking into account larger structures in the graph by relying on *paths* instead of *edges*. By doing so, we formulate a novel random walks based modularity that takes into account longer dependencies. The idea of extending modularity by using random walks theory has been first expressed in [1] where the authors presented different possible extensions of modularity based on various motifs (e.g. triangles or paths) instead of edges. However, no mathematical formulation modeling the problem is presented and the experiments consist of a simple analysis of results obtained on small undirected graphs (e.g. the Zachary's Karate Club, the Southern Women Event Participation network). More recently, another workshop paper [11] has been studying the extension of modularity to paths instead of edges. The authors showed on small undirected graphs (e.g. the Zachary's Karate Club, the College Football and Political Book network) that they can identify better ground-truth communities.

The modularity matrix offers also a way to project the observed graph in a feature space. Newman showed that the solution to the fuzzy clustering problem that maximizes modularity for k clusters is given by the k first eigenvectors of the modularity matrix [23]. Therefore, many works summarize the graph in terms of latent social dimension using the k fuzzy clusters that maximizes modularity. Tang *et al.* [32, 34] proposed to solve a semi-supervised classification problem by extracting the k first eigenvectors of the modularity matrix and use them as features describing the nodes of the graph. While competitive, the results are not necessarily better than label propagation methods (see the experimental section). In this work, we show that by relying on features based on the random walk modularity, instead of features based on the standard, edge based modularity, we largely outperform state-of-the-art approaches on two large scale publicly available data sets.

3. PROBABILISTIC INTERPRETATION OF MODULARITY

In this section, we review the probabilistic interpretation of modularity. Originally, the modularity was built to measure the correlation between the membership of nodes and the links between the nodes. In order to do so, modularity compares the structure of the observed graph with the structure that we would expect if the graph was built independently from the membership of its nodes. Suppose our network contains n nodes partitioned into m categories or groups, $\mathcal{C}_1, \ldots, \mathcal{C}_m$. Let us consider the probability that both the starting node and the ending node of a randomly picked edge are in the category $\mathcal{C}_k$, in other words the probability that an edge is inside that category. If the nodes of the graph were rewired randomly, but conserving the connectivity of each node, the probability that an edge would fall in this category $\mathcal{C}_k$ is simply: $\mathrm{P}(s \in \mathcal{C}_k)\mathrm{P}(e \in \mathcal{C}_k)$, i.e. the probability that a randomly picked edge starts in category $\mathcal{C}_k$ multiplied by the probability that an edge ends in category $\mathcal{C}_k$. If the edges of a graph are such that $\mathrm{P}(s \in \mathcal{C}_k, e \in \mathcal{C}_k) > \mathrm{P}(s \in \mathcal{C}_k)\mathrm{P}(e \in \mathcal{C}_k)$ it means that there are more edges inside the categories than we would expect by chance, or in other words that there is a correlation between the structure of the network and the membership of its nodes.

In this bag-of-edges space, modularity of the partition $\{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$ can be defined as:

$$Q = \sum_{k=1}^{m} [\mathrm{P}(s \in \mathcal{C}_k, e \in \mathcal{C}_k) - \mathrm{P}(s \in \mathcal{C}_k)\mathrm{P}(e \in \mathcal{C}_k)] \quad (1)$$

$$= \sum_{k=1}^{m} \sum_{i,j \in \mathcal{C}_k} [\mathrm{P}(s = i, e = j) - \mathrm{P}(s = i)\mathrm{P}(e = j)] \quad (2)$$

Those probabilities are usually estimated through frequency computation, which allows to express the modularity in a matrix form, using the adjacency matrix $\mathbf{A}$. We define for each category a membership column-vector $\mathbf{u}_k$ with $[\mathbf{u}_k]_j = 1$ if the node j belongs to the category $\mathcal{C}_k$ and 0 else. We also define $\mathbf{e}$ as a column vector of length n, where n is the number of nodes, whose elements are all one. It is easy to see that:

- $\mathbf{u}_k^{\mathrm{T}}\mathbf{A}\mathbf{u}_k$ is the number of edges that start and end in the category $\mathcal{C}_k$,
- $\mathbf{u}_k^{\mathrm{T}}\mathbf{A}\mathbf{e} = \sum_{k'=1}^{n} \mathbf{u}_k^{\mathrm{T}}\mathbf{A}\mathbf{u}_{k'}$ is the number of edges that start in $\mathcal{C}_k$,
- $\mathbf{e}^{\mathrm{T}}\mathbf{A}\mathbf{u}_k$ is the number of edges that end in $\mathcal{C}_k$,
- $\mathbf{e}^{\mathrm{T}}\mathbf{A}\mathbf{e}$ is the total number of edges in the graph.

We can then express modularity as follows (see [22] for a detailed explanation):

$$Q = \frac{1}{\mathbf{e}^{\mathrm{T}}\mathbf{A}\mathbf{e}} \sum_{k=1}^{m} \mathbf{u}_k^{\mathrm{T}}\mathbf{Q}\mathbf{u}_k \quad (3)$$

Where the modularity matrix $\mathbf{Q}$ is defined as follows:

$$\mathbf{Q} = \mathbf{A} - \frac{(\mathbf{A}\mathbf{e})\left(\mathbf{e}^{\mathrm{T}}\mathbf{A}\right)}{\mathbf{e}^{\mathrm{T}}\mathbf{A}\mathbf{e}} \quad (4)$$

In the remainder, we introduce an extension of the modularity based on random walks. The idea of the random walks based modularity (RWM) is to consider the probability that a *path* starts and ends in a category $\mathcal{C}_k$, rather than edges. To compute this probability, we consider a generalization of the random walk with restart [37] by using the bag-of-paths framework [21].

4. RANDOM WALKS BASED MODULARITY

As presented in Equation (1), modularity relies on computing two quantities: The joint probability of starting and ending in a node of the same category: $\mathrm{P}(s \in k, e \in \mathcal{C}_k)$, and its computation assuming independence of the events of starting from a node and ending in a node of the same category: $\mathrm{P}(s \in \mathcal{C}_k)\mathrm{P}(e \in \mathcal{C}_k)$. As shown in Equation (2), these probabilities require to compute the joint probability of starting in node i and ending in node j: $\mathrm{P}(s = i, e = j)$ and its independent counterpart: $\mathrm{P}(s = i)\mathrm{P}(e = j)$. In the remainder, we refer to these two quantities as our values of interest.

The usual modularity model estimates these probabilities from edges. In other words, it assumes having a pool (or bag) of edges from which it samples with replacement. Then, a simple frequency computation is used to estimate the quantities of interest. In the remainder, we propose to sample from an infinite countable bag-of-paths with replacement. In such a space, we can estimate both quantities of interests: the joint probability of starting and ending in a node of the same category, and its independent counterpart. The assumption is that by relying on paths instead of edges one can capture longer dependencies and henceforth extract "richer" structures in the network (see the experimental section for the validation on semi-supervised learning experiments). In the remainder, we introduce a random walk model called the bag-of-paths framework that is used to compute path probabilities (for a thorough introduction to the bag-of-paths framework, see [10]). Notice that other random walk models could have been used such as the random walk with restart, the exponential diffusion kernel or the regularized laplacian [9]. We decide to rely on the bag-of-paths framework since, as we will see, it generalizes the random walk with restart model, and defines a large family of random walks controlled by a temperature parameter. Furthermore, the mathematical foundation of the model allows to derive elegant formulas for computing the values of interest $\mathrm{P}(s = i, e = j)$ and $\mathrm{P}(s = i)\mathrm{P}(e = j)$.

4.1 Notations

We assume that we are working with a directed graph G. We denote the set of paths $\wp$ of length $\leq \tau$ by $\mathcal{P}$, and with $\mathcal{P}_{ij}$ the subset of paths $\in \mathcal{P}$ that start at i and end in j. Each path is associated with a total cost $\tilde{c}(\wp)$. We consider that the cost of a path is given by the sum of the costs c_{ij} of each edge in the path. The graph G is thus defined by the cost matrix $\mathbf{C}$. When there is no edge from node i to node j we consider the cost c_{ij} to be infinite. The graph G is also associated to an adjacency matrix $\mathbf{A}$. The elements a_{ij} of $\mathbf{A}$ are usually defined as $a_{ij} = 1/c_{ij}$, but can be defined independently of the cost matrix. Also, we define the prior probability p_{ij}^{ref} that a natural random walker takes the outgoing link from i to j as the uniform probability defined on the set of outgoing links, weighted by the affinity a_{ij} (all p_{ij}^{ref} define the matrix $\mathbf{P}^{\mathrm{ref}}$). Then, we can define the likelihood of a path $\tilde{\pi}^{\mathrm{ref}}(\wp)$, i.e. the product of the transition probabilities p_{ij}^{ref} associated to each edge on the path $\wp$. After normalization, we obtain $\tilde{\mathrm{P}}^{\mathrm{ref}}(\wp) = \tilde{\pi}^{\mathrm{ref}}(\wp)/\sum_{\wp' \in \mathcal{P}} \tilde{\pi}^{\mathrm{ref}}(\wp')$, which is the prior distribution defined on the set of paths $\mathcal{P}$ according to the natural random walk.

4.2 Probability distribution on paths

In this section, we introduce the mathematical framework we use to estimate the values of interests $\mathrm{P}(s = i, e = j)$ and $\mathrm{P}(s = i)\mathrm{P}(e = j)$. The bag-of-paths framework defines a random walker whose moving strategy consists of minimizing the expected cost of the paths he is taking (i.e. favoring shortest paths or exploitation) while in the meantime trying to explore as much as possible the graph. This exploration–exploitation tradeoff may be formalized in the following way:

$$\left|\begin{array}{ll} \underset{\mathrm{P}(\wp)}{\text{minimize}} & \sum_{\wp \in \mathcal{P}} \mathrm{P}(\wp)\tilde{c}(\wp) \\ \text{subject to} & \sum_{\wp \in \mathcal{P}} \mathrm{P}(\wp)\log(\mathrm{P}(\wp)/\tilde{\mathrm{P}}^{\mathrm{ref}}(\wp)) = J_0 \\ & \sum_{\wp \in \mathcal{P}} \mathrm{P}(\wp) = 1 \end{array}\right. \quad (5)$$

The first Equation expresses the exploitation strategy minimizing the expected cost (or in other words favoring short-

est paths). The second ensures exploration of the graph by constraining the divergence (J_0) from the prior distribution $\mathrm{P}^{\mathrm{ref}}$, which defines a pure exploration strategy of a random walker who always chooses its next possible step with a uniform probability over the outgoing links.

This exploration-exploitation tradeoff problem is typical of statistical physics [12] and has a closed form solution which is a Boltzmann probability distribution:

$$\boxed{\mathrm{P}(\wp) = \frac{\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right]}{\sum_{\wp'\in\mathcal{P}}\tilde{\pi}^{\mathrm{ref}}(\wp')\exp\left[-\theta\tilde{c}(\wp')\right]}} \tag{6}$$

with θ controlling the exploration–exploitation tradeoff. It is easier to express the solution in terms of θ than of J_0.

As a parallel with statistical physics, the denominator is called the partition function:

$$\mathcal{Z} = \sum_{\wp\in\mathcal{P}}\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right] \tag{7}$$

We immediately see that $\theta = 0$ implies $\mathrm{P}(\wp) = \tilde{\mathrm{P}}^{\mathrm{ref}}(\wp)$, and when $\theta \to \infty$ the probability distribution is concentrated on the path(s) with the lowest cost. In other words, as θ increases, the relative entropy decreases and the probability distribution ranges from a random walk model to a shortest path model.

4.3 Computing the probability of a path

Using the bag-of-paths framework, let us see how to compute our two values of interests: $\mathrm{P}(s=i,e=j)$ and $\mathrm{P}(s=i)\mathrm{P}(e=j)$.

First, we define the matrix $\mathbf{W}$:

$$\boxed{\mathbf{W} = \mathbf{P}^{\mathrm{ref}} \circ \exp\left[-\theta\mathbf{C}\right] = \exp\left[-\theta\mathbf{C} + \log\mathbf{P}^{\mathrm{ref}}\right],} \tag{8}$$

where $\circ$ is the elementwise (Hadamard) matrix product, and the exponential and logarithm are taken elementwise.

We can easily see that $\mathcal{Z}$ can be computed from the matrix $\mathbf{W}$ in the following way:

$$\mathcal{Z} = \sum_{\wp\in\mathcal{P}}\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right] = \sum_{i,j=1}^{n}\left[\sum_{t=0}^{\tau}\mathbf{W}^t\right]_{ij} \tag{9}$$

Moreover, $\mathrm{P}(s=i,e=j)$, the probability of picking a path of length up to τ starting in i and ending in j, can be computed from $\mathbf{W}$:

$$\begin{aligned}\mathrm{P}(s=i,e=j) &= \frac{\sum_{\wp\in\mathcal{P}_{ij}}\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right]}{\mathcal{Z}} \\ &= \frac{\left[\sum_{t=0}^{\tau}\mathbf{W}^t\right]_{ij}}{\sum_{i,j=1}^{n}\left[\sum_{t=0}^{\tau}\mathbf{W}^t\right]_{ij}}\end{aligned} \tag{10}$$

We can now extend the measure to all possible path lengths by considering the limit of $\sum_{t=0}^{\tau}\mathbf{W}^t$ for $\tau\to\infty$. This series converges since the spectral radius of $\mathbf{W}$ is less than 1. Indeed, since $\theta > 0$ and $c_{ij} > 0$, the matrix $\mathbf{W}$ is substochastic (the sum of each row is less than 1), which implies that $\rho(\mathbf{W}) < 1$. Therefore, we have:

$$\mathbf{Z} = \sum_{t=0}^{\infty}\mathbf{W}^t = (\mathbf{I}-\mathbf{W})^{-1} \tag{11}$$

We call $\mathbf{Z} = (\mathbf{I}-\mathbf{W})^{-1}$ the fundamental matrix, and z_{ij} the element (i,j) of $\mathbf{Z}$, where

$$z_{ij} = \sum_{\wp\in\mathcal{P}_{ij}}\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right] \tag{12}$$

This allows to compute $\mathcal{Z}$ in the following way

$$\mathcal{Z} = \sum_{i,j=1}^{n} z_{ij} = \mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e} \tag{13}$$

We can now compute our two values of interest crucial for revisiting modularity in terms of random walks. First, the joint probability of picking a path starting in node i and ending in node j.

$$\mathrm{P}(s=i,e=j) = \frac{\sum_{\wp\in\mathcal{P}_{ij}}\tilde{\pi}^{\mathrm{ref}}(\wp)\exp\left[-\theta\tilde{c}(\wp)\right]}{\mathcal{Z}} = \frac{z_{ij}}{\mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e}} \tag{14}$$

And the estimated probability assuming independence:

$$\mathrm{P}(s=i)\mathrm{P}(e=j) = \frac{(\mathbf{e}_i^{\mathrm{T}}\mathbf{Z}\mathbf{e})(\mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e}_j)}{(\mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e})^2} \tag{15}$$

with $\mathbf{e}_i$ being the $n\times 1$ column vector whose elements are all set to 0 except for ith entry set to 1.

4.4 Random Walks based Modularity Computation

Based on the fundamental matrix $\mathbf{Z}$ (Equation (11)), using Equation (1) and (2), we can define the random walk modularity matrix:

$$\boxed{\mathbf{Q}_{\mathrm{RW}} = \mathbf{Z} - \frac{(\mathbf{Z}\mathbf{e})\,(\mathbf{e}^{\mathrm{T}}\mathbf{Z})}{\mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e}}} \tag{16}$$

And the random walk modularity Q_{RW}:

$$\boxed{Q_{\mathrm{RW}} = \frac{1}{\mathbf{e}^{\mathrm{T}}\mathbf{Z}\mathbf{e}}\sum_{k=1}^{m}\mathbf{u}_k^{\mathrm{T}}\mathbf{Q}_{\mathrm{RW}}\mathbf{u}_k} \tag{17}$$

In other words, simply by replacing the adjacency matrix in Equations (3) and (4) by the fundamental matrix $\mathbf{Z}$ of the bag-of-paths framework we extend the classical modularity to a well-defined random walk modularity comparing the probability of finding a *path* (instead of an *edge*) between two nodes of a cluster against the prior probability of that event.

4.5 Relation to other random walk methods

The bag-of-paths model offers a powerful and elegant way to exploit the notion of random walks on graphs. Moreover, it provides a new theoretical background to the famous random walk with restart model [28, 35], which was inspired by the PageRank [27]. Indeed, the random walk with restart kernel is a particular case of the fundamental matrix ($\mathbf{Z}$) of the bag-of-paths model. The cost c_{ij} associated with the edge $(i,\ j)$ in the bag-of-paths model is usually defined as $c_{ij} = 1/a_{ij}$, with a_{ij} being an element of the adjacency matrix of the graphs. However, this cost can be defined at will, and a sensible definition is to use a uniform cost ($c_{ij} = 1$ for all edges) – which is anyhow the case in many real-world graphs. Assuming a uniform cost, we have:

$$\mathbf{W} = e^{-\theta}\mathbf{P}^{\text{ref}} \tag{18}$$

If we define $\alpha = e^{-\theta}$, Equation (11) becomes:

$$\mathbf{Z} = (\mathbf{I} - \alpha\mathbf{P}^{\text{ref}})^{-1}, \tag{19}$$

which is indeed the definition of the random walk with restart kernel with a restart probability of $(1-\alpha)$ [28, 9]. Moreover, Equation (19) is also equivalent to the definition of some diffusion kernels [37, 13]. Therefore, the presented modularity framework encompasses a large family of random walks.

5. APPLICATION TO SEMI-SUPERVISED LEARNING

In order to illustrate the benefit of computing the modularity on paths instead of links, we propose to measure the extent to which features extracted using random walks based modularity can help in semi-supervised learning problems. In this specific setting, we suppose to have a network at our disposal, as well as labels on some of the nodes, and the goal is to predict the labels on the remaining nodes. The problem is semi-supervised because at training time the learning algorithms exploits not only the labels of the training data points, but also the whole structure of the network, including the connections of the test data points.

One of the state-of-the-art algorithms for dealing with graph-based semi-supervised problems consists in extracting in an unsupervised way the modularity features (i.e. the top k eigenvectors of the modularity matrix) from the complete network, and training a classifier on the extracted features to learn how to discriminate between the classes [32, 33]. Computing the k first eigenvectors means that we consider a feature space of k dimensions where the i^{th} element of the j^{th} eigenvector is the j^{th} coordinate of the i^{th} node of the network. The justification of the use of the top eigenvectors as a feature space comes from [23] that showed that the solution of the fuzzy clustering problem that maximizes modularity for k clusters is given by the k first eigenvectors of the modularity matrix. Therefore, those eigenvectors reveal latent social dimensions that can be used as features for a classifier.

In the remainder, we use exactly the same approach, however, this time extracting features from the random walk modularity matrix. By doing so, we outperform significantly the approach relying on traditional modularity features. In this way, we show that random walks based modularity extracts richer features on the tested networks.

5.1 Derived Algorithm

To address the semi-supervised classification problem, we derive a new algorithm, RW ModMax (or *Random Walk Modularity Maximization*) using the dominant spectral components of the random walks based modularity matrix instead of the standard one. The modularity matrix $\mathbf{Q}$ can simply be replaced by the random walks based modularity matrix $\mathbf{Q}_{\text{RW}}$ (see Equation (17)), where the top eigenvectors of $\mathbf{Q}_{\text{RW}}$ are the features to extract. However, remember that the computation of $\mathbf{Q}_{\text{RW}}$ itself relies on the computation of the fundamental matrix $\mathbf{Z}$ obtained after a matrix inversion (Equations (11) and (16)). In case of large networks, this computation is not feasible and therefore computing the dominant eigenvector of $\mathbf{Q}_{\text{RW}}$ is not trivial.

Fortunately, there exist well established algorithms (such as the power method or the more advanced Lanczos method [30]) that can be used to compute the dominant eigenvectors of a matrix without requiring it as input. Instead, these algorithms require as input a fast procedure to compute the product of the matrix with any column vector. In other words, if such a procedure can be designed we can extract the top eigenvectors of $\mathbf{Q}_{\text{RW}}$ without having to compute and store it explicitly.

The product of $\mathbf{Q}_{\text{RW}}$ and any vector $\mathbf{x}$ can be rewritten as:

$$\mathbf{Q}_{\text{RW}}\mathbf{x} = \mathbf{Z}\mathbf{x} - \frac{(\mathbf{Z}\mathbf{e})\left(\mathbf{e}^{\text{T}}\mathbf{Z}\mathbf{x}\right)}{\mathbf{e}^{\text{T}}\mathbf{Z}\mathbf{e}} \tag{20}$$

Thus, if we can compute $\mathbf{y} = \mathbf{Z}\mathbf{x}$ for any $\mathbf{x}$ (including $\mathbf{e}$), we can compute $\mathbf{Q}_{\text{RW}}\mathbf{x}$. Using the definition of $\mathbf{Z}$ we derive:

$$\begin{aligned} &\mathbf{y} = \mathbf{Z}\mathbf{x} \\ &\Rightarrow (\mathbf{I} - \mathbf{W})\mathbf{y} = \mathbf{x} \\ &\Rightarrow \mathbf{y} = \mathbf{W}\mathbf{y} + \mathbf{x} \end{aligned} \tag{21}$$

Equation (21) offers an iterative method to compute $\mathbf{y} = \mathbf{Z}\mathbf{x}$. The iterations converge because the spectral radius of $\mathbf{W}$ is less than one (as explained in Section 4). Algorithm 1 summarizes how the product $\mathbf{Q}_{\text{RW}}\mathbf{x}$ can be computed iteratively. Notice that steps 3 and 4 of Algorithm 1 are independent of $\mathbf{x}$ so they can be done only once for a given graph.

In summary, the dominant eigenvectors of $\mathbf{Q}_{\text{RW}}$ can be extracted using an iterative algorithm (for instance, in our implementation we used the state-of-the-art ARPACK library [16], which is based on the Lanczos method); each step of this algorithm requires the computation of the product $\mathbf{Q}_{\text{RW}}\mathbf{x}$, which in turn implies the computation of the product $\mathbf{Z}\mathbf{x}$ (Equation (20)). This last product is computed iteratively using Equation (21) that only requires the sparse matrix $\mathbf{W}$. This shows that we can compute the top eigenvectors of $\mathbf{Q}_{\text{RW}}$ without having to compute explicitly $\mathbf{Z}$, allowing us to scale up to large networks.

The RW ModMax uses the top eigenvectors of the random walk modularity matrix as features for a classifier (for example a SVM). The complete algorithm is summarized in Algorithm 2.

6. EXPERIMENTS

In this section, we evaluate the performance of RW ModMax on a graph-based semi-supervised classification task with multiple labels. We first describe the three baselines and the two social network data sets used in the experiments. Then, we compare the performances of the baselines with RW ModMax. We analyze the parameter settings and the running time of the proposed model. Finally, the results are analyzed and discussed.

6.1 Baselines for Comparison

We compare the RW ModMax method introduced in Section 5.1 with three state-of-the-art algorithms for graph-based semi-supervised classification.

Algorithm 1 Iterative computation of $\mathbf{Q}_{\mathrm{RW}}\mathbf{x}$

Input:
– The $n \times n$ matrix $\mathbf{W}$.
– The column vector $\mathbf{x}$ of n elements.

1: Initialize $\mathbf{y}^{(0)}$, a column vector of n elements. For example $\mathbf{y}^{(0)} = \mathbf{x}$.
2: Iterate $\mathbf{y}^{(t)} = \mathbf{W}\mathbf{y}^{(t-1)} + \mathbf{x}$ until convergence. $\mathbf{y}^{(\infty)} \approx \mathbf{Z}\mathbf{x}$ (Equation (21)).
3: Initialize $\mathbf{y}_e^{(0)} = \mathbf{e}$, where $\mathbf{e}$ is a column vector of n elements.
4: Iterate $\mathbf{y}_e^{(t)} = \mathbf{W}\mathbf{y}_e^{(t-1)} + \mathbf{e}$ until convergence. $\mathbf{y}_e^{(\infty)} \approx \mathbf{Z}\mathbf{e}$.
5: Compute $\mathbf{Q}_{\mathrm{RW}}\mathbf{x}$ using Equation (20), with the approximations of $\mathbf{Z}\mathbf{x}$ and $\mathbf{Z}\mathbf{e}$ computed at steps 2 and 4.
6: **return** $\mathbf{Q}_{\mathrm{RW}}\mathbf{x}$

Algorithm 2 RW ModMax

Input:
– A graph G containing n nodes.
– The $n \times n$ adjacency matrix $\mathbf{A}$ associated to G, containing affinities.
– The $n \times n$ cost matrix $\mathbf{C}$ associated to G (usually, the costs are the inverse of the affinities, but other choices are possible).
– The parameter $\theta > 0$.
– The number of eigenvectors k.
– The known labels $\mathbf{Y}$, a $n \times m$ matrix whose element (i, j) is equal to 1 if i is known to have the label l_j, and 0 else.

Output:
– A prediction score for each label with regard to each unlabeled node.

1: $\mathbf{D} \leftarrow \mathbf{Diag}(\mathbf{A}\mathbf{e})$ {the row-normalization matrix}
2: $\mathbf{P}^{\mathrm{ref}} \leftarrow \mathbf{D}^{-1}\mathbf{A}$ {the reference transition probabilities matrix}
3: $\mathbf{W} \leftarrow \mathbf{P}^{\mathrm{ref}} \circ \exp[-\theta\mathbf{C}]$ {elementwise exponential and multiplication}
4: Extract the k dominant eigenvectors of the random walks based modularity matrix using a method that only requires to compute the product of $\mathbf{Q}_{\mathrm{RW}}$ (Equation (17)) and a column vector (done by Algorithm 1), e.g. ARPACK library.
5: Train a classifier (e.g. an SVM) with the known labels $\mathbf{Y}$ using the eigenvectors as features of the nodes.
6: Compute scores of each label for unlabeled nodes using the trained classifier.

- **ModMax** uses the k dominant eigenvectors of the standard, edge based, modularity matrix of the graph as features for a subsequent supervised classification [32]. The classification procedure consists in training a linear SVM in a one-versus-the-rest strategy. It differs from our introduced algorithm RW ModMax only in the fact that we use the k dominant eigenvectors of the random walks based modularity matrix instead of the usual modularity matrix. ModMax depends on two parameters, the number of features (k) and the SVM hardness (C).

- **Label diffusion** is based on the idea of propagating the labels throughout the dataset, starting from labeled data and jumping from neighbor to neighbor. The methods based on the principle of diffusion range among of the most widely used methods of semi-supervised classification, probably because they achieve impressive performances with regards to their simplicity of implementation [13, 9, 37, 38]. Our implementation is based on [37], which uses a symmetric normalization of the adjacency matrix. The label diffusion method has one parameter, $\alpha \in]0, 1[$, that determines the rate at which the information vanishes in the diffusion process. For small values of α, the labeled nodes only influence close neighbors, and their zone of influence increases as α tends to 1.

- **EdgeCluster** is based on the idea of extracting features using a fuzzy clustering method, and uses those features for classification with a linear SVM [32]. EdgeCluster adopts an edge-centric view (i.e. the nodes are the features of the edges that link them) and performs a k-means clustering on the edges. The clusters of the edges linked to one node constitute then the features of the said node. As ModMax, EdgeCluster depends on two parameters, the number of clusters (k) and the SVM hardness (C).

6.2 Datasets Description

- **BlogCatalog** is extracted from the website `www.blogcatalog.com`. The website allows blog writers to list their blog under one or more categories, and to specify their friends' blogs, making it a labeled network of blogs. The Blog catalog is a subset of this network containing about 10,000 nodes classified in 39 different categories [32].

- **Flickr** is extracted from the popular photo-sharing website `www.flickr.com`. The users of Flickr can subscribe to interest groups and add other users as friends. The Flickr dataset is a subset of the friends' network of Flickr, and the interest groups are used as label for the user [32].

Both datasets present the scale-free structure often observed in social networks, and their characteristics are listed in Table 1.

6.3 Evaluation Metrics

Each element of the BlogCatalog and Flickr datasets can have multiple labels, and the presented algorithms produce for each element a ranking of the most probable labels. This

Data Set	BlogCatalog	Flickr
Categories	39	195
Nodes	10,312	80,513
Edges	333,983	5,899,882
Maximum Degree	3,992	5,706
Average Degree	65	146

Table 1: Characteristics of the datasets.

ranking is compared to the ground-truth using the micro- and macro-average of the F-measure, respectively noted microF1 an macroF1 [20, 26], a pair of metrics well-known in information retrieval. The F-measure is the harmonic mean of precision and recall. The micro- and macro-average are two ways of computing the F-measure when dealing with multiple labels. The microF1 is defined using directly the global recall (ρ) and precision (π) of the results (as in [32, 33, 34], ρ and π are computed on the ranking truncated to the true number of labels):

$$\text{microF1} = \frac{2\pi\rho}{\pi + \rho} \tag{22}$$

However, if the distribution of label sizes is highly skewed, the microF1 will be mainly influenced by the performances of the method on the most populated labels. The macroF1, on the other hand, gives the same weight to each label:

$$\text{macroF1} = \frac{1}{m}\sum_{i=1}^{m}\frac{2\pi_i\rho_i}{\pi_i + \rho_i} \tag{23}$$

With ρ_i the recall for label i, π_i the precision for label i, and m the number of labels.

We use a third measure, the accuracy (acc), defined as the percentage of elements for which the algorithm correctly predicted the whole set of true labels.

6.4 Settings

The four tested algorithms have one or more parameters whose optimal value depends on the current dataset. In the following experiments, the value of the parameters were chosen automatically at each run among a set of reasonable values using an internal cross-validation method. The cross-validation selected the values yielding the best average microF1. Table 2 shows the set of values tested during the automatic tuning. For the methods with two parameters to tune, every pair of values was tested and the cross-validation selected the pair with the best average score. The value of θ in the RW ModMax method has a negligible influence on the performance (see Section 6.6), hence it was set to an arbitrary value of 5 during the learning process.

6.5 Results

BlogCatalog.

We focus our experiments on classification with small training ratio. Indeed, as it often becomes more and more difficult to gather true labels in real world problems when the size of the dataset increases, it is essential for semi-supervised classification methods designed for large-scale problems to be able to deal with small training ratios. Figure 1 compares the performances of each method on the BlogCatalog dataset for training ratios of 1%, 5% and 10%. The RW ModMax algorithm outperforms the other methods, as confirmed by a Mann-Withney U test with a 1% significance level [19]. The same experiment was made on training ratios of 3%, 7%, 30%, 50%, 70% and 90%, with a systematic dominance of the RW ModMax algorithm. In particular, the RW ModMax method outperforms the original ModMax, demonstrating the benefits of introducing random walks based modularity.

Method	Parameter	Tested values
RW ModMax	θ (inverse temperature)	5
	k (number of features)	100, 200, 500, 800, 1000
	C (SVM hardness)	10, 20, 50, 100, 200, 500
ModMax	k (number of features)	100, 200, 500, 800, 1000
	C (SVM hardness)	10, 20, 50, 100, 200, 500
EdgeCuster	k (number of features)	(0.2, 0.5, 1, 2, 5, 10) $\times 10^3$
	C (SVM hardness)	10, 20, 50, 100, 200, 500
Label diffusion	α (diffusion factor)	0.1, 0.2, 0.3, . . . , 0.9

Table 2: List of the parameters of each method, and set of values tested during the automatic parameters tuning. NB. : on the Flickr dataset C was only chosen among two values, 20 and 500.

Flickr.

Figure 2 shows the performances of the four methods for training ratios of 1%, 5% and 10% ; similar results are observed for training ratios of 3%, 7% and 9%. As for the BlogCatalog dataset, the RW ModMax dominates its competitors. The statistical significance of those results is confirmed by a Mann-Withney U test with a confidence level of 1%.

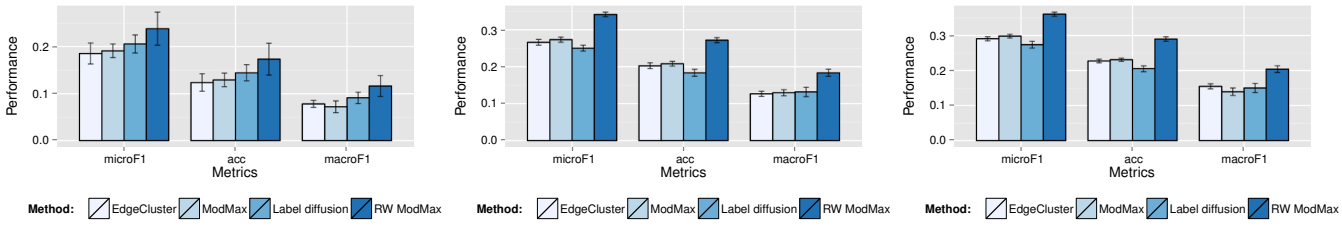
6.6 Parameter Analysis

Figure 3 shows the influence of θ and k on the performances of RW ModMax (on BlogCatalog and Flickr). We do not observe any significant influence of θ in the range $[0.1, 5]$, as confirmed by Mann-Withney U tests.

Table 3 shows the influence of θ on a wider range of values. When $\theta \to 0$, the spectral radius of $\mathbf{W}$ tends to 1, increasing dramatically the convergence time of Algorithm 1. The performance achieved for θ between 0.05 and 10 are similar. A closer look at Figure 3 seems to indicate better results for smaller values of θ, but the difference is of the order of the standard deviation of those results, and is dwarfed by the influence of k on the results. For $\theta = 50$ and higher the quality of the results drops significantly. Indeed, as $\theta \to \infty$, $\mathbf{Z} \to \mathbf{I}$, which in practice is reached around $\theta = 50$ due to numerical precision. At that point, the random walks based modularity matrix becomes independent of the graph itself and the resulting classification is not better than random guessing.

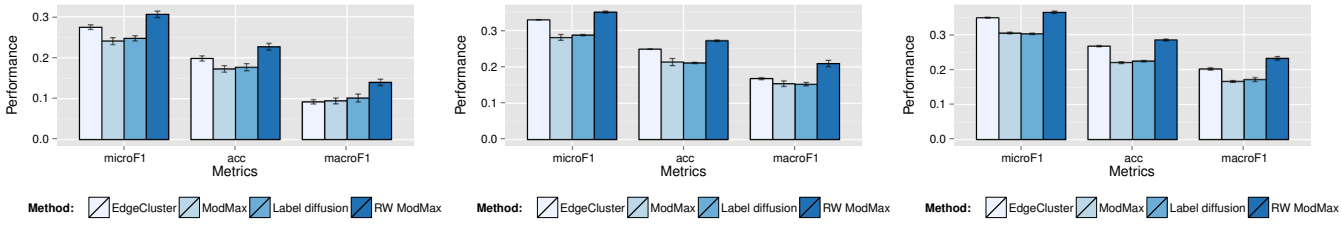
The good stability of the RW ModMax performances with regards to θ — as reported in Figure 3 — allows to easily choose a value of θ (for instance $\theta = 5$) without having to rely on automatic parameter tuning. Therefore, although RW ModMax has one more parameter than ModMax, it does not lead to a harder tuning of the parameters.

The choice of k corresponds to the classical over-fitting/under-fitting trade-off and its value has an important influence on the performance. As one could expect, for an identical training ratio, the Flickr dataset requires more features than the BlogCatalog dataset in order to handle the larger number of nodes and labels. The training ratio also has a strong influence on the optimal value of k. As an example, the



(a) 1% training ratio (b) 5% training ratio (c) 10% training ratio

Figure 1: Comparison of the methods on the BlogCatalog dataset for 1%, 5% and 10% training ratios. Results are averaged over 20 runs.



(a) 1% training ratio (b) 5% training ratio (c) 10% training ratio

Figure 2: Comparison of the methods on the Flickr dataset for 1%, 5% and 10% training ratios. Results are averaged over 10 runs.

θ	0.05	0.1	1	10	50	100
microF1 (%)	40.9	40.9	40.6	40.2	14.0	14.0
macroF1 (%)	26.1	25.9	25.9	25.5	3.7	3.9
Accuracy (%)	33.5	33.7	33.3	32.9	8.1	7.9

Table 3: Influence of θ on the performances of RW ModMax. Dataset: BlogCatalog; training ratio: 50%; $k = 200$; $C = 20$. Results are averaged over 20 runs.

	with online parameters tuning	without online parameters tuning
Label Diffusion	$\pm$ 15s	$<$ 1s
EdgeCluster	$\pm$ 15min	$\pm$ 2min30
RW ModMax	$\pm$ 20min	$\pm$ 5min
ModMax	$\pm$ 20min	$\pm$ 5min

Table 4: Running time on the BlogCatalog dataset with a 50% training ratio. Results obtained on a personal computer (Intel Core i5-2500K 3.3GHz CPU, 16Go RAM). The methods are implemented and run with Matlab R2011b 64-bit.

number of features maximising the macroF1 on the BlogCatalog dataset is about 200 for a 5% training ratio, and about 500 for a 50% training ratio. Unsurprisingly, a larger training set is less prone to over-fitting and can benefit from more features.

Figure 4 shows the influence of C on the microF1 and macroF1 of RW ModMax for two training ratios on BlogCatalog and Flickr, respectively. C has a significant influence on the performance, as confirmed by Mann-Withney U test. We observe that the optimal value of C is dependent on the dataset, with BlogCatalog needing a softer SVM than Flickr.

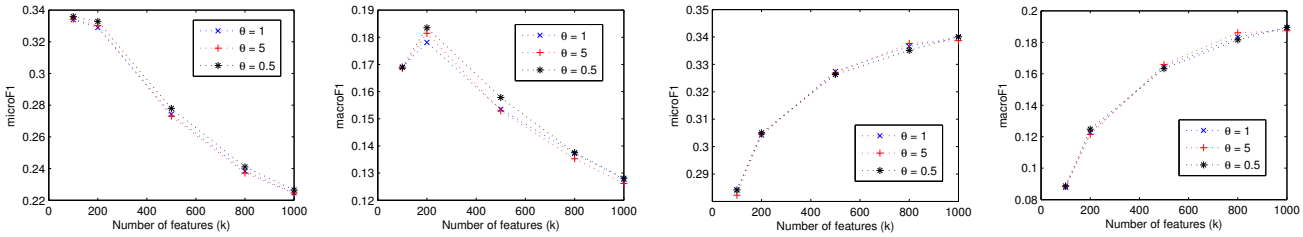
In conclusion, the fine tuning of the number of features (k) is essential for the performance of RW ModMax. A tuning of the SVM hardness (C) can yield further improvement, while the influence of θ is insignificant as long as $\theta \in [0.1, 5]$.

6.7 Running Time Analysis

The RW ModMax algorithm consists of two main phases: (1) the extraction of the top eigenvectors from the random walk modularity matrix and (2) the classification via a linear SVM. We observe experimentally that the running time of the complete algorithm is in $O(k)$. C only influences the second phase of the algorithm for which we observe an increase of the running time as C increases.

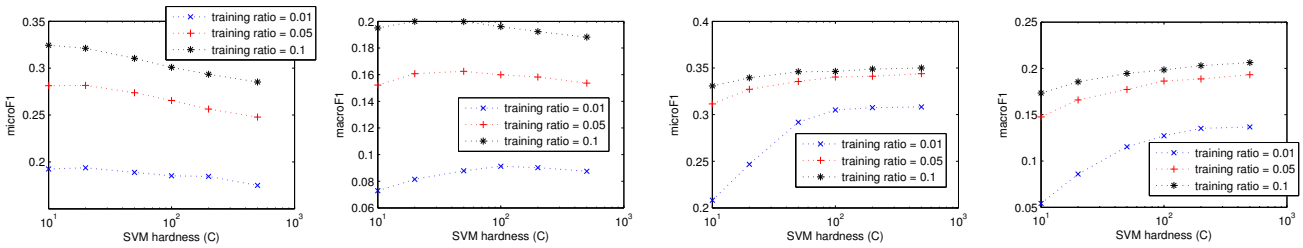
On large graphs with small training ratio the dominant phase (with regards to the running time) is the extraction of eigenvectors, making k the parameter with the biggest influence on the total running time. In this case, the number of features that can be extracted is dictated by time constraints.

Tables 4 and 5 indicate the running time of the four methods on the BlogCatalog and Flickr dataset, respectively. The label diffusion method, although dominated by the RW ModMax, is two orders of magnitude faster than the other methods. The quality of the results offered by RW ModMax comes at the cost of a higher running time. On the other hand, the substitution of traditional modularity by random walks based modularity in the ModMax algorithm causes only a moderate overhead on the Flickr dataset, and is negligible on the smaller BlogCatalog graph.



(a) BlogCatalog (b) Flickr

Figure 3: Influence of θ and k on the microF1 and macroF1 on the BlogCatalog and Flickr datasets for RW ModMax. The training ratio is 5%. The SVM hardness $C = 20$. The results are averaged over 50 runs for BlogCatalog, and over 10 runs for Flickr.



(a) BlogCatalog (b) Flickr

Figure 4: Influence of C on the microF1 and macroF1 on the BlogCatalog and Flickr datasets for RW ModMax. $\theta = 5$ and $k = 500$. The results are averaged over 50 runs for BlogCatalog, and over 10 runs for Flickr. Training ratios of 1%, 5% and 10%

	with online parameters tuning	without online parameters tuning
Label Diffusion	± 20min	± 50s
EdgeCluster	± 15h	± 2h30
RW ModMax	± 1h45	± 1h30
ModMax	± 1h	± 45min

Table 5: Running time on the Flickr dataset with a 5% training ratio. Results obtained on a personal computer (Intel Core i5-2500K 3.3GHz CPU, 16Go RAM). The methods are implemented and run with Matlab R2011b 64-bit.

6.8 Discussions

Advantages.

Our proposed algorithm is able to deal with graphs of the order of 100 000 nodes on a personal computer. Moreover, it outperforms the tested semi-supervised classification methods, including the well-known label propagation method [37]. In particular, RW ModMax significantly improves the labeling performances of the original ModMax algorithm, with similar running time and tuning complexity.

Scaling to Big Data.

The RW ModMax method does need substantial resources when dealing with very large graphs. Indeed, RW ModMax relies on the extraction of a large number of features, and storing 1000 features (in double precision) for a graph of ten millions nodes would require about 70Gb, making it impractical for running on a personal computer. However, the algorithm can be efficiently adapted on distributed systems in order to scale up to big data. Indeed, many solutions have already been proposed for extracting the dominant eigenvectors of very large sparse matrices on grid infrastructures [6, 18] and for using linear classifiers on big data problems [3].

Towards more complex Structures.

We have seen that modularity can benefit from being computed from paths instead of edges. Recent work dealing with kernel methods on graphs revealed new possibilities to embed graphs in richer structural Hilbert spaces, as the bag-of-trees or the bag-of-forests [31]. By working on such spaces, we can derive other modularities exploiting the statistical arrangement of *trees* or *forests* in the network. In this case, by substituting the matrix $\mathbf{Z}$ in Equation (16) by the corresponding kernel we obtain the desired modularity. However, to be able to apply Algorithm 1, depending on the embedding space, one will have to derive a "kernel trick" procedure to avoid storing and computing explicitly the kernel matrix.

Application to community detection.

The experiments revealed on two data sets that the features of the random walks based modularity help to classify better than the features of the usual modularity. This suggests that random walks based modularity is more informative. Therefore, a natural application for it is the problem of community detection. For this purpose, one option is to derive a ΔQ_{RW} function and to apply it in a greedy algorithm as done in [4]. However, in the context of community detection, assessing to which extent a clustering is better remains a difficult question [36].

6.9 Reproducibility of the Experiments

The Matlab implementation of the RW ModMAX is made publicly available at: github.com/rdevooght/RWModMax. A Matlab implementation of the label diffusion of Zhou *et. al* [37] is available at: github.com/rdevooght/Label-diffusion. The Matlab implementations of ModMax and EdgeCluster are publicly available at: leitang.net/code/social-dimension/. Finally, the two datasets benchmarked in this paper, BlogCatalog and Flickr, can be downloaded from: leitang.net/code/social-dimension/.

7. CONCLUSIONS

In this work, we introduced a novel, formal and well-defined, random walks based modularity as the number of *paths* observed within a group minus the probability that a *path* falls in this group by random. On a semi-supervised classification procedure of the nodes in the network using two social networks, we validated that the features of the random walks based modularity help to classify better than the features of the usual modularity. In this way, by computing modularity from *paths* instead of edges, we showed that more informative features can be extracted from the network. In order to scale the method to work with large data, we derived an algorithm that avoids computing and storing explicitly the fundamental matrix associated to the underlying markov chain. Moreover, on both tested data sets we outperformed the label propagation strategy, a state-of-the-art algorithm for graph-based semi-supervised learning.

Future work.

In this study, we restricted our investigation to one specific structure: the *paths* of the network. To push the study further, we would like to extend modularity to richer structures as *trees* and *forests*. Another challenge, lies in the possibility to scale up to very large graphs. Therefore, at the moment, we are investigating different strategies to make our approach benefit of grid infrastructures [6, 18, 3]. Finally, following [36], we plan to test the extent to which our approach succeeds in identifying ground-truth communities.

8. ACKNOWLEDGEMENTS

R. Devooght is supported by the Belgian Fonds pour la Recherche dans l'Industrie et l'Agriculture (FRIA). Part of this work has been funded by the European Union 7th Framework Programme ARCOMEM and Social Sensor projects, by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037 "Social Media", and by projects with the "Region Wallonne" and the "Region Bruxelloise INOVIRIS". We thank these institutions for giving us the opportunity to conduct both fundamental and applied research. We would like to thank Martin Saveski for providing the scripts generating the performance figures, and Daniele Quercia for his comments on the paper.

9. REFERENCES

[1] A. Arenas, A. Fernandez, S. Fortunato, and S. Gómez. Motif-based communities in complex networks. *Journal of Physics A: Mathematical and Theoretical*, 41(22):224001, 2008.

[2] A. Arenas, A. Fernandez, and S. Gomez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[3] R. Bekkerman, M. Bilenko, and J. Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[4] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[5] A. Clauset, M. E. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[6] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox. Twister: a runtime for iterative mapreduce. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, pages 810–818. ACM, 2010.

[7] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[8] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36, 2007.

[9] F. Fouss, K. Francoisse, L. Yen, A. Pirotte, and M. Saerens. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Networks*, 31:53 – 72, 2012.

[10] K. Francoisse, I. Kivimaki, A. Mantrach, F. Rossi, and M. Saerens. A bag-of-paths framework for network data analysis. *Submitted for publication; available on ArXiv as ArXiv:1302.6766*, pages 1–36, 2013.

[11] R. Ghosh and K. Lerman. Community detection using a measure of global influence. In *Advances in Social Network Mining and Analysis*, pages 20–35. Springer, 2010.

[12] E. T. Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[13] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. *Advances in neural information processing systems*, 15:657–664, 2002.

[14] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84(6):066122, 2011.

[15] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110, 2008.

[16] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, volume 6. Siam, 1998.

[17] J. Leskovec, K. J. Lang, and M. Mahoney. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international Conference on World Wide Web*, pages 631–640. ACM, 2010.

[18] J. Lin and M. Schatz. Design patterns for efficient graph algorithms in mapreduce. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*, pages 78–85. ACM, 2010.

[19] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger

than the other. *The annals of mathematical statistics*, 18(1):50–60, 1947.

[20] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008.

[21] A. Mantrach, L. Yen, J. Callut, K. Francoisse, M. Shimbo, and M. Saerens. The sum-over-paths covariance kernel: A novel covariance measure between nodes of a directed graph. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(6):1112–1126, 2010.

[22] M. Newman. *Networks: an introduction*. OUP Oxford, 2009.

[23] M. E. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[24] M. E. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[25] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[26] A. Özgür, L. Özgür, and T. Güngör. Text categorization with class-based and corpus-based keyword selection. In *Computer and Information Sciences-ISCIS 2005*, pages 606–615. Springer, 2005.

[27] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[28] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 653–658. ACM, 2004.

[29] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[30] Y. Saad. *Numerical methods for large eigenvalue problems*, volume 158. SIAM, 1992.

[31] M. Senelle, S. García-Díez, A. Mantrach, M. Shimbo, M. Saerens, and F. Fouss. The sum-over-forests density index: identifying dense regions in a graph. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, To appear, 2014.

[32] L. Tang and H. Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 817–826. ACM, 2009.

[33] L. Tang and H. Liu. Scalable learning of collective behavior based on sparse social dimensions. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1107–1116. ACM, 2009.

[34] L. Tang, X. Wang, H. Liu, and L. Wang. A multi-resolution approach to learning with overlapping communities. In *Proceedings of the First Workshop on Social Media Analytics*, pages 14–22. ACM, 2010.

[35] H. Tong, C. Faloutsos, and J.-Y. Pan. Random walk with restart: fast solutions and applications. *Knowledge and Information Systems*, 14(3):327–346, 2008.

[36] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 3. ACM, 2012.

[37] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16(753760):284, 2004.

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