Graph Evolution via Social Diffusion Processes

Dijun Luo, Chris Ding, and Heng Huang

Department of Computer Science and Engineering,
University of Texas, Arlington, Texas, USA
dijun.luo@gmail.com, {chqding,heng}@uta.edu

Abstract. We present a new stochastic process, called as Social Diffusion Process (SDP), to address the graph modeling. Based on this model, we derive a graph evolution algorithm and a series of graph-based approaches to solve machine learning problems, including clustering and semi-supervised learning. SDP can be viewed as a special case of *Matthew effect*, which is a general phenomenon in nature and societies. We use social event as a metaphor of the intrinsic stochastic process for broad range of data. We evaluate our approaches in a large number of frequently used datasets and compare our approaches to other state-of-the-art techniques. Results show that our algorithm outperforms the existing methods in most cases. We also applying our algorithm into the functionality analysis of microRNA and discover biologically interesting cliques. Due to the broad availability of graph-based data, our new model and algorithm potentially have applications in wide range.

1 Introduction

Data clustering, assignment, and dimensional reduction have been the focuses for exploring unknown data [1,2]. Among them, graph-based data analysis techniques have recently been investigated extensively in traditional machine learning problems. One reason for the popularity of graph-based approaches is the broad availability of graph data. For example, social objects (users, blog items, photos) are generated with relational links, and for objects represented in Euclidean space, one can easily obtain a graph by using similarity measurements (*e.g.* Gaussian kernels). Graph-based approaches fall into two categories. The first one is *spectral graph partitioning* methods which address the group detection problem by identifying an approximately minimal set of edges to remove from the graph to achieve a given number of groups [3,4,5,6]. Impressive results have been shown in these methods which have been applied in many practical applications. These approaches relax NP-hard combinatorial problems into continuous optimization problems which can be solved by eigenvector decompositions.

Another approach category is *stochastic modeling*. In stochastic models, the observed data are assumed to be drawn from some distribution and generative assumptions [7,8,9,10,11]. These approaches often lead to a maximum likelihood problems that can be solved by Expectation Maximization (EM) or approximately Variational EM algorithms [12].

Among these models, the Chinese Restaurant Processes (CRPs) consider a sequence of customers coming to a restaurant according to the convention of Chinese people: one tends to stay in a place where there are more people. Each customer seeks some previously occupied table and the probability is proportional to the number of customers already sitting there. The new customer also sits in a new table with probability proportional to some parameter. CRP and its variations have been theoretically and empirically studied in many previous researches [10,11,13,14]

In a CRP mixture, customers are data points, and customers sitting at the same table belong to the same cluster. Since the number of occupied tables is random, the resulting posterior distribution of seating assignments provides a distribution of clusterings where the number of clusters is determined by the data.

In this paper, we propose a novel stochastic process which further considers the social events among social members as a metaphor of the intrinsic stochastic process for broad range of data. We call this process as Social Diffusion Process. The basic assumption in this model is that two social members tend to communicate if they are familiar with each other or have many common friends, and that the more times they communicate, the more they are familiar with each other.

Based on our model, we derive an iterative evolution algorithm to model the social structures of the members. The major characteristic of our algorithm which differs from most of previous research is that we do not need to impose latent variables which leads to maximum likelihood estimation. Instead, our evolutionary algorithm iteratively generates a new relational graph among social members in which the social structures become more and more clear, please see Figure 1 for a toy example. In this example, our algorithm starts from a random binary network and ends with clearly separated subgraphs. Details can be found in Section 3.2.

The similar algorithm which is closest to our intuition is Markov Clustering (MCL) [15] from the point of view of graph evolution. However, MCL is not suitable for the purpose in this paper. We perform the MCL evolution on the same graph in Figure 1 (a) and the results for MCL are demonstrated in Figure 2. One can observe that the result in Figure 1 is much more reasonable than that in Figure 2.

The results of the evolution algorithm can be viewed as a special case of the *Matthew effect*, in which “The rich get richer”. This is a general phenomenon in nature and societies [16,17,18]. One interesting observation in our algorithm is that the evolution of a graph by the SDP enhance the *qualities* of the graph in a wide range of applications. This phenomenon suggests that the SDP assumptions are natural in general. Due to the broad availability of graph-based data, our new model and algorithm have potential applications in various areas.

In the rest of the paper, we first introduce the Social Diffusion Process in Section 2 and the derived algorithm in Section 3. In section 4, we show the evidence of improvement of the quality by our algorithm using extensive experiments.

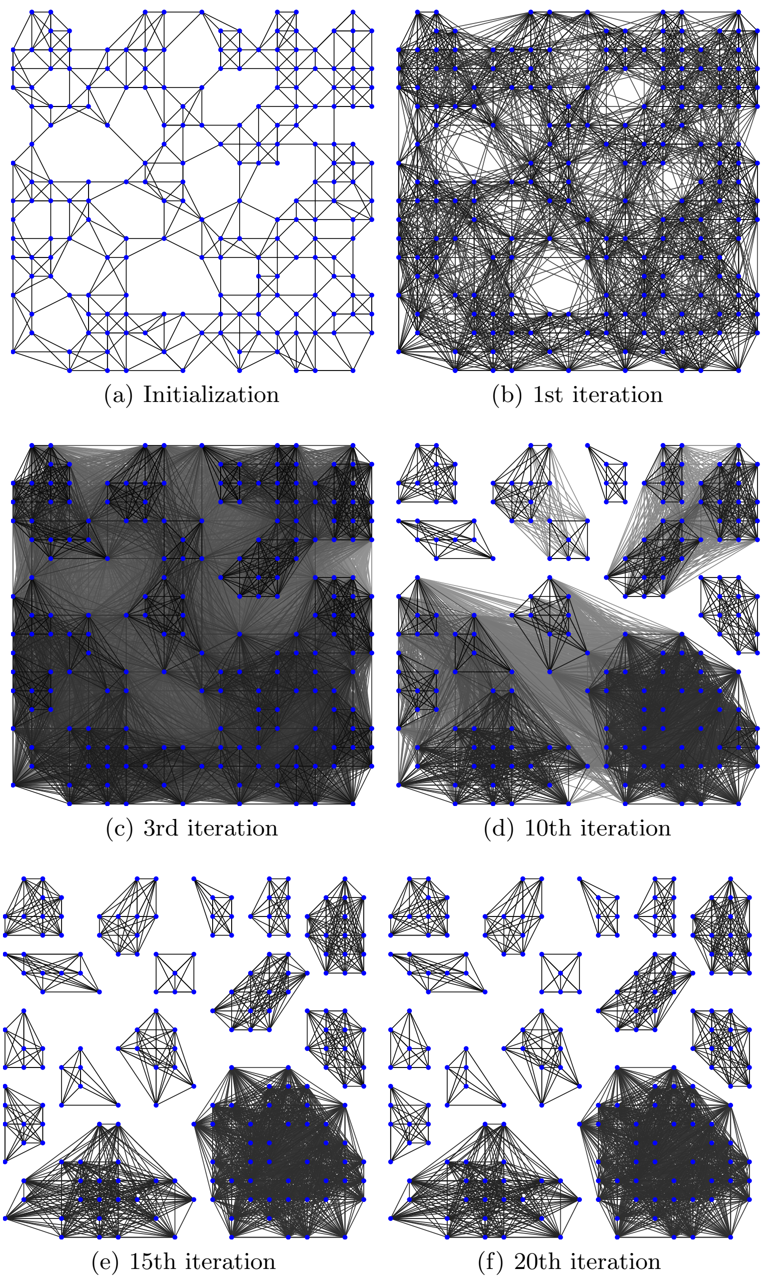
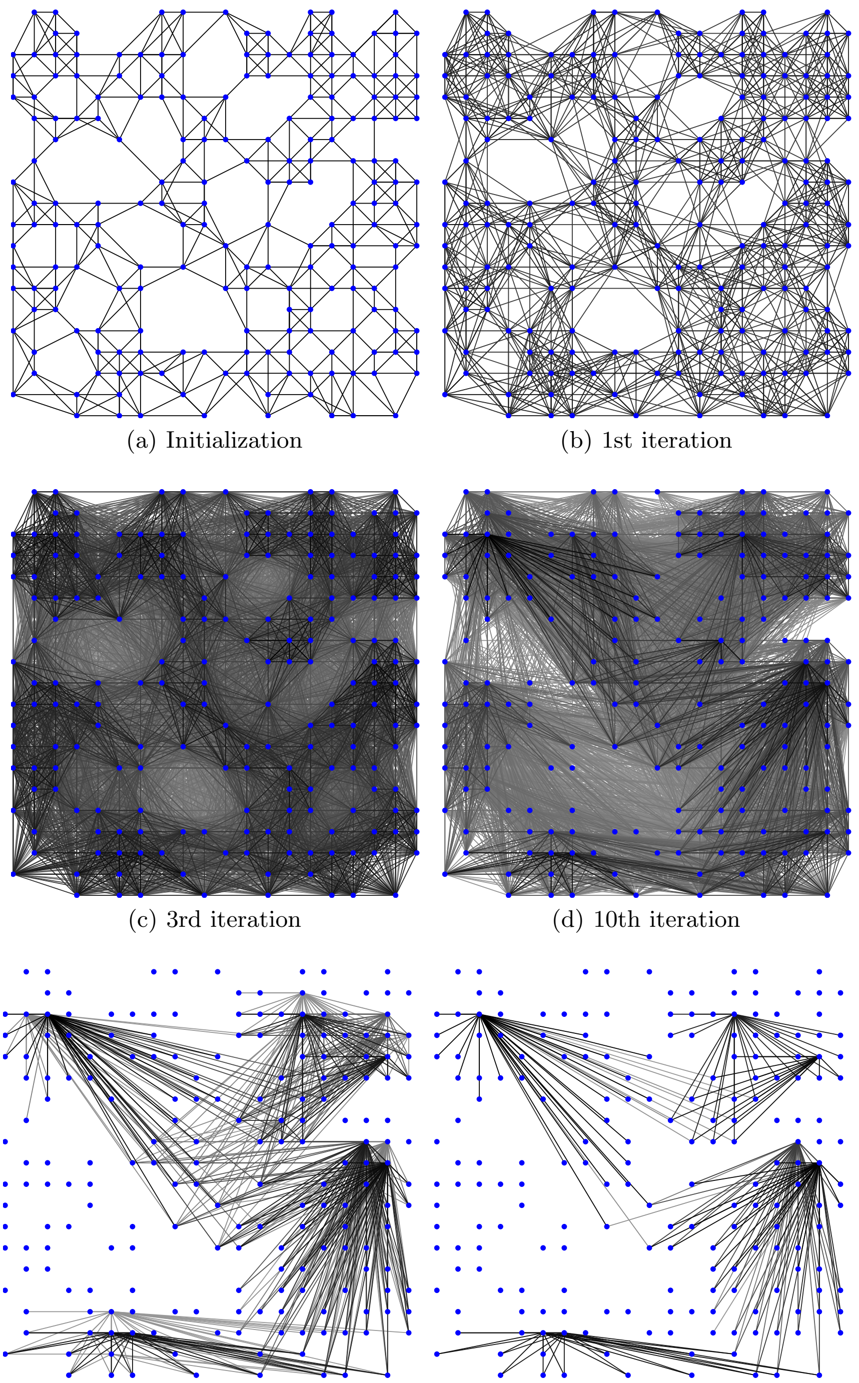


Fig. 1. Graph evolution results on the grid toy data based on Social Diffusion Process. Each point (blue dot) represents a social member and the edge between two social members represents the familiarness between them. (a): the original graph. (b)– (f): the condensation results of the 1st, 3rd, 10th, 15th, and 20th iterations of our evolution algorithm. The darkness of the edge represents the familiarness between the social members (the darker the higher).


(a) Initialization (b) 1st iteration

(c) 3rd iteration (d) 10th iteration

(e) 15th iteration (f) 20th iteration

Fig. 2. Graph evolution results on the grid toy data based on Markov Clustering

2 Social Diffusion Process for Friendship Broadening

In this section we introduce the Social Diffusion Process based on the notations of graph.

2.1 Preliminaries

Let $G = \{V, W\}$ denote an undirected weighted graph, where $V = \{v_1, v_2, \cdots, v_n\}$ is the set of nodes, $W \in \mathbb{R}^{n\times n}$ is a $n \times n$ matrix, and W_{ij} denotes the weight of the edge between nodes v_i and v_j. $W_{ij} = 0$, if there is no edge between v_i and v_j.

2.2 Social Events and Broadening of Friendship

We consider the following scenario: A and B are friends. Suppose A brings a friend A_f and meets with B. Now A_f and B become known to each other. If B also brings a friend B_f to the meeting, i.e., the four (A, A_f, B, B_f) meet. Then A_f become known to both B also B_f, i.e., the friendship circle for A_f is broadened. This happens to A, B, B_f as well.

In graph terminology, the initial friendship between A and B is represented by an edge connecting A and B. The broadened friendship between A_f and B (assuming they are not connected at initial stage) has a connection strength somewhere between 0 and 1. In other words, if two persons C and D don't know each other, the existence of a mutual friend connects C and D. Further more, even if A and B are friends (i.e., an edge exists between A and B), their friendship is further enhanced due to the existence of mutual friends. Our main goal is to formally define this friendship broadening process and compute the **friendship enhancement probability**. We expect this enhanced friendship provide a more clear social community structure as shown in Figure 1.

Formally, we define the following events among social members: (1) **Date** (v_i, v_j): v_i and v_j initial a dating. (2) **Bring**(v_i, v_k): v_i brings v_k after the event **Date**(v_i, v_j) for some j. (3) **Meet**(v_p, v_q): v_p and v_q meet in the same table.

We further impose the following rules: (1) If **Date**(v_i, v_j) happens, **Meet**(v_i, v_j) happens, or (2) If **Date**(v_i, v_j) and **Bring**(v_i, v_k) happen, **Meet**(v_k, v_j) happens. (3)If **Date**(v_i, v_j), **Bring**(v_i, v_k), and **Bring**(v_j, v_l) happen, **Meet**(v_j, v_l) happens.

Here we assume **Date**(v_i, v_j) is equivalent to **Date**(v_j, v_i) and **Meet**(v_k, v_l) is equivalent to **Meet**(v_l, v_k).

We use the following to denote the rules above

$$\text{Rule 1:} \quad \mathbf{Date}(v_i, v_j) \Rightarrow \mathbf{Meet}(v_i, v_j) \tag{1}$$

$$\text{Rule 2:} \quad \left.\begin{matrix} \mathbf{Date}(v_i, v_j) \\ \mathbf{Bring}(v_i, v_k) \end{matrix}\right\} \Rightarrow \mathbf{Meet}(v_j, v_k) \tag{2}$$

$$\text{Rule 3:} \quad \left.\begin{matrix} \mathbf{Date}(v_i, v_j) \\ \mathbf{Bring}(v_i, v_k) \\ \mathbf{Bring}(v_j, v_l) \end{matrix}\right\} \Rightarrow \mathbf{Meet}(v_k, v_l) \tag{3}$$

2.3 Social Diffusion Process

Now we are ready to introduce the Social Diffusion Process. The process starts with a graph $G = \{V, W\}$ where $V = \{v_1, v_2, \cdots, v_n\}$ denotes a set of social members and W denotes the familiarness between social members, *i.e.* W_{ij} represents the familiarness between v_i and v_j, $i, j = 1, 2, \cdots, n$. We assume that $W_{ij} = W_{ji}$. The SDP happens as following,

(1) Choose a threshold $t \sim U(0, \mu)$ where $\mu = \max_{ij} W_{ij}$ and U denotes the uniform distribution.
(2) **Date**(v_i, v_j) happens with a constant probability δ if $W_{ij} \geq t$.
(3) **Bring**(v_i, v_k) and **Bring**(v_j, v_l) happen with probability $p(i, k, t)$, $p(j, l, t)$, respectively, where

$$p(i,k,t) = \begin{cases} \frac{1}{|\mathcal{N}_{i,t}|} & \text{if } \ v_k \in \mathcal{N}_{i,t} \\ 0 & \text{otherwise} \end{cases},$$

$$p(j,l,t) = \begin{cases} \frac{1}{|\mathcal{N}_{j,t}|} & \text{if } \ v_k \in \mathcal{N}_{j,t} \\ 0 & \text{otherwise} \end{cases},$$

$\mathcal{N}_{i,t} = \{q : W_{iq} \geq t\}, \mathcal{N}_{j,t} = \{q : W_{jq} \geq t\}$, and $|\cdot|$ denotes the cardinality of the set.
(4) Apply rules (1)–(3). For any p, q, if **Meet**(v_p, v_q), $W_{pq} \leftarrow W_{pq} + \alpha\mu$.

The threshold t can be interpreted as the importance of the dating event. Two friends do not date if they are not familiar with each other enough (thresholded by t)[1]. When a social member brings some friend, he/she only considers those friends who are familiar enough with (thresholded by t). The set $\mathcal{N}_{i,t}$ is the friends the social member v_i can bring with this threshold t. Eq. (4) indicates that social member v_i chooses friends in $\mathcal{N}_{i,t}$ with uniform distribution. Notice that there are two parameters in this model δ and α. In section 3, we will introduce an algorithm based on the SDP, in which the two parameters can be eliminated by natural normalization.

3 Graph Evolution Based on Social Diffusion Process

3.1 The Evolution Algorithm

We first denote A^t as the following

$$(A^t)_{ij} = \begin{cases} 1 & \text{if } W_{ij} \geq t \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

[1] The reason why we use a thresholding of W_{ij} instead of directly using W_{ij} for event **Date**(v_i, v_j) is following. Assume we want to date with some one on the wedding of Royal wedding for William and Kate, who are we going to date? Probably one of our most important friends. In the same event, if we want to bring guest to meet our friend in the date, who are we going to bring? Probably another one of our most important friends. In reality, social events happen according to their importance, denoted as threshold t in the paper. We believe this model is much accurate than directly using W_{ij} as the probability of **Date**(v_i, v_j).

where t is a positive threshold. Consider two social members v_i and v_j. The events in which they meet each other can be divided into three cases:

Case (1). $\mathbf{Date}(v_i, v_j)$. In this case the probability that they meet is

$$P(\mathbf{Meet}(v_i, v_j)) = \delta(A^t)_{ij}.$$

Case (2). $\mathbf{Date}(v_i, v_k)$ and $\mathbf{Bring}(v_k, v_j)$. By definition $|\mathcal{N}_{k,t}| = \sum_j A^t_{jk} = d^t_k$, where d^t_k is the degree k in A^t. In this case,

$$\begin{aligned} & P(\mathbf{Meet}(v_i, v_j)) \\ =& \sum_k P(\mathbf{Meet}(v_i, v_j) | \mathbf{Date}(v_i, v_k), \mathbf{Bring}(v_k, v_j)) \\ =& \sum_k \delta(A^t)_{ik} \frac{A^t_{jk}}{d_k} = \delta(A^t D^{-1} A^t)_{ij}, \end{aligned}$$

where $D = \mathbf{diag}(d_1, d_2, \cdots, d_n)$.

Case(3). $\mathbf{Date}(v_k, v_l)$, $\mathbf{Bring}(v_k, v_i)$, and $\mathbf{Bring}(v_l, v_j)$. Similar with case (2), we have

$$\begin{aligned} P(\mathbf{Meet}(v_i, v_j)) &= \sum_{kl} \delta(A^t)_{kl} \frac{A^t_{ik}}{d_k} \frac{A^t_{jl}}{d_l} \\ &= \delta(A^t D^{-1} A^t D^{-1} A^t)_{ij}. \end{aligned}$$

By summing up the three cases, we have

$$\begin{aligned} & P(\mathbf{Meet}(v_i, v_j)) \\ =& \delta A^t_{ij} + \delta(A^t D^{-1} A^t)_{ij} + \delta(A^t D^{-1} A^t D^{-1} A^t)_{ij}. \end{aligned}$$

From the definition of updating of W, we have

$$\begin{aligned} & \mathbb{E}(\varDelta W_{ij}) \\ =& \alpha\mu\delta \left(A^t_{ij} + (A^t D^{-1} A^t)_{ij} + (A^t D^{-1} A^t D^{-1} A^t)_{ij} \right) \\ \triangleq& \alpha\mu\delta M^t_{ij}. \end{aligned} \tag{5}$$

Here $A^t_{ij} + (A^t D^{-1} A^t)_{ij} + (A^t D^{-1} A^t D^{-1} A^t)_{ij}$ is denoted by M^t_{ij}. This suggests that the expectation $\mathbb{E}(\varDelta W_{ij})$ is proportional to M^t_{ij}. In our implementation we normalize M^t_{ij} by $M^t_{ij} \leftarrow M^t_{ij} / \sum_{i'j'} M^t_{i'j'}$, which leads to the following algorithm,

In this algorithm, we use an evenly distributed threshold t to approximate the uniform distribution from which t should be drawn from. In our experiments, we set $T = 50$. One should notice that no matter what the choice of the normalization is, the algorithm has the following properties.

Algorithm 1. $\tilde{W} =$ **GraphEvolution**(W)

```
**Input**: Graph $W$
**Output:** Graph $\tilde{W}$
$\mu = \max_{ij} W_{ij}, \tilde{W} = \mathbf{0}$
**for** $i = 1 : T$ **do**
  $t = i\mu/T$
  Calculate $M^t$ using Eq. (5)
  Normalize $M^t : M^t_{ij} \leftarrow M^t_{ij} / \sum_{i'j'} M^t_{i'j'}$
  $\tilde{W} \leftarrow \tilde{W} + M^t$
**end for**
**Output:** $\tilde{W}$
```

Property 1. The result of **GraphEvolution** is scale invariant, *i.e.* $\forall \beta > 0$,

$$\textbf{GraphEvolution}(W) = \textbf{GraphEvolution}(\beta W).$$

This is because the threshold t is always evenly distributed in the interval $[0, \max_{ij} W_{ij}]$ and M^t remains the same. In other words, the choice of the normalization does not change any terms in M^t.

Property 2. If W is a set of disconnected full cliques with same size and same weight, *i.e.* there is a partition $\Pi = \{\pi_1, \pi_2, \cdots, \pi_K\}, \pi_k \cap \pi_l = \Phi, 1 \leq k, l \leq K, \cup_k \pi_k = \{v_1, v_2, \cdots, v_n\}$ such that $\forall i, j \in \pi_k, W_{ij} = c$ where c is a constant, and $\forall i \in \pi_k, j \in \pi_l, k \neq l, W_{ij} = 0$, then

$$W \propto \textbf{GraphEvolution}(W).$$

This is easy to show since if W is a set of disconnected full cliques with the same weight, A^t is the same for every $t : A^t_{ij} = 1$ if $A_{ij} \neq 0$, $A^t_{ij} = 0$ otherwise. Thus $M^t \propto W$, which leads to $W \propto$ **GraphEvolution**(W). This property shows a hint of conditions in which the algorithm of $W \leftarrow$ **GraphEvolution**(W) converges, which will be discussed later.

3.2 Application of Graph Evolution

The algorithm **GraphEvolution** can be used in different purposes. The basic idea is that it improves the quality in terms of the natural structure underlying the graph data. In this paper, we investigate two applications: clustering and semi-supervised learning.

For the purpose of clustering, one can simply iteratively perform the following

$$W \leftarrow \textbf{GraphEvolution}(W). \tag{6}$$

As iterations continue, the structures of the graph is clearer and clearer. We show results of the evolution algorithm on a toy grid data, see Figure 1.

In this example, we randomly generate 198 points in a 20×20 grid. We obtain an unweighted graph as follows. If node i is one of K-nearest neighbors of node j, or node j is one of the K-nearest neighbors of node i, we set $W_{ij} = 1$, and $W_{ij} = 0$ otherwise. $K = 7$ in this example and the neighborhood is computed using the Euclidean distance of the nodes on the 2-dimensional grid coordinate. The original graph is shown in Figure 2(a).

Starting from this graph, we run the **GraphEvolution** algorithm for 20 iterations and the results of the first, third, 10th, 15th, and 20th iterations are shown in Figure 1 (b)–(d). In the third iteration (Figure 2(c)), the structure of the data is observable. In the 10th iteration (Figure 2(d)), the structure is even more clear. Finally, in the 20th iteration, (Figure 2(f)), the clusters are completely separated.

After the graph evolution iterations, the cluster structure encoded in the edge weight matrix is usually obvious to human. In practice, the number of clusters discovered by the algorithm is different from expected number of clusters. We use the following partition scheme to reach a desired number of cluster. We run algorithm in Eq. (6) until there are two disconnected subgraphs. Then pick up the subgraph which has a large number nodes to run algorithm in Eq. (6), and do the same strategy until we reach a specified number of clusters.

For the purpose of semi-supervised learning, we just use $\tilde{W} =$ **Graph Evolution**(W) as preprocessing, where W is the input of and $\tilde{W}$ is the output. Instead of performing semi-supervised learning on W, we do it on $\tilde{W}$. We show that the qualities of the $\tilde{W}$ are much higher than W.

4 Experimental Results

In this section, we first demonstrate the convergence of algorithm and then show experimental evidence of the quality improvement by apply our graph evolution algorithm. In the clustering comparison, we specify the number of clusters. However, in a microRNA pattern discovery application, we run our algorithm until convergence and let the algorithm determine the number of clusters.

4.1 Convergence Analysis

We first demonstrate the convergence of our algorithm on a toy data, which is a 9×9 binary graph, shown in the left most panel of the bottom row of Figure 3. There are two cliques in this graph: nodes 1–4 and nodes 5–9. We add some noise by setting $W_{13} = W_{58} = W_{79} = 0$ and $W_{45} = 1$. We run algorithm in Eq. (6) for 30 iterations. One can observe that our algorithm converges fast and at the convergent graph, all edges within the same clique have the same value. Also as highlighted in Figure 3, the noise values of W_{13}, W_{58}, W_{79}, and W_{45} are corrected by our algorithm.

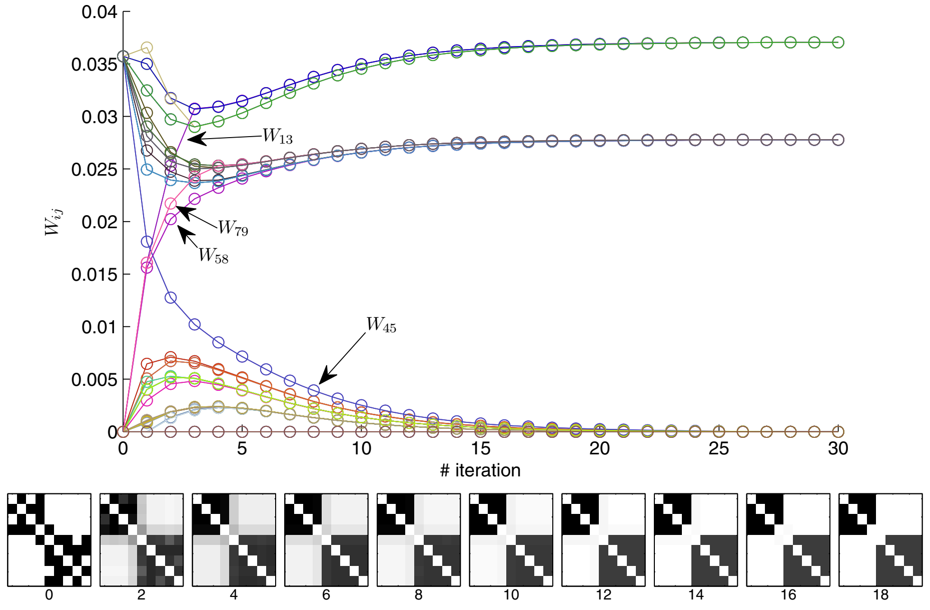


Fig. 3. Convergence curves and adjacency matrix of our algorithm on a 9×9 toy data. The left most panel of the bottom row is the initial binary graph (black represents 1 and white represents 0) and the rest of the bottom row is the evolution result of 2nd, 4th, $\cdots$, 18th iterations. Initially, nodes 1–4 is a pseudo-clique, as well as nodes 5–9. $W_{13} = W_{58} = W_{79} = 0$ and $W_{45} = 1$. After around 18 iterations, the two cliques become separated and the nodes within the two cliques become full connected. The top panel show the convergence of all the elements in W. Highlighted are the values of W_{13}, W_{58}, W_{79}, and W_{45}, which are corrected by our algorithm.

4.2 Clustering

In this experiment, we extensively compare our algorithm with standard clustering algorithms (K-means, Spectral Clustering, Normalized Cut[2]) in 20 data sets. These data sets come from a wide range of domains, including gene expressions including gene expressions (PR1,SRB, LEU, LUN, DER, AML, GLI, MAL, MLL), images (ORL, UMI, COI, JAF, MNI) and other standard UCI data sets (ION, PR2, SOY, ECO, GLA, YEA, ZOO, CAR, WIN, IRI)[3]. We use accuracy, normalized mutual information (NMI) and purity as the measurement of the clustering qualities and the results are shown in Table 1. Our method achieves the best results in 22 out of 24 data sets. Here notice that for Spectral Clustering and Normalized Cut, we tune the graph construction parameters. More

[2] We also compared with MCL. However the accuracies are much (more than 10%) lower than all the method we compare here. We believe MCL is not suitable for the purpose in this paper. One can find visual evidence in Figure 2.

[3] All the mentioned data can be downloaded at parchive.ics.uci.edu/ml/ or csie.ntu.edu.tw/ cjlin/.

Table 1. Accuracy, normalized mutual information (NMI), and purity comparison of K-mean (Km), Spectral Clustering (SC), Normalized Cut (Ncut), and Graph Evolution (GE). Both Spectral Clustering and Normalized Cut are achieved by tuning the graph construction parameters and the best results are reported.

	Accuracy				**NMI**				**Purity**			
	Km	SC	Ncut	GE	Km	SC	Ncut	GE	Km	SC	Ncut	GE
UMI	0.458	0.471	0.498	**0.644**	0.641	0.646	0.649	**0.763**	0.494	0.505	0.505	**0.667**
COI	0.570	0.614	0.792	**0.839**	0.734	0.750	0.860	**0.879**	0.623	0.658	0.817	**0.840**
ION	0.707	0.702	0.684	**0.880**	0.123	0.193	0.107	**0.446**	0.707	0.730	0.684	**0.880**
JAF	0.744	0.799	0.965	**0.967**	0.809	0.849	0.959	**0.962**	0.774	0.819	0.965	**0.967**
MNI	0.687	0.713	0.820	**0.833**	0.690	0.698	0.748	**0.769**	0.705	0.733	0.820	**0.833**
ORL	0.582	0.683	0.756	**0.775**	0.786	0.834	0.866	**0.891**	0.624	0.713	0.773	**0.802**
PR1	0.716	0.675	0.562	**0.899**	0.129	0.176	0.102	**0.458**	0.726	0.757	0.708	**0.899**
PR2	0.580	0.566	0.569	**0.706**	0.019	0.017	0.013	**0.136**	0.580	0.566	0.569	**0.706**
SOY	0.908	0.871	**1.000**	**1.000**	0.903	0.859	**1.000**	**1.000**	0.924	0.893	**1.000**	**1.000**
SRB	0.480	0.622	**0.699**	0.639	0.232	0.411	**0.454**	0.421	0.512	0.645	**0.699**	0.639
YEA	0.132	0.327	0.302	**0.395**	0.013	0.129	0.126	**0.231**	0.328	0.430	0.436	**0.540**
ZOO	0.264	0.674	0.629	**0.723**	0.116	0.615	0.570	**0.751**	0.423	0.750	0.737	**0.871**
AML	0.688	0.678	0.659	**0.847**	0.100	0.100	0.073	**0.394**	0.696	0.692	0.666	**0.847**
CAR	0.623	0.729	0.719	**0.799**	0.655	0.743	0.738	**0.779**	0.691	0.789	0.788	**0.822**
WIN	0.961	0.936	0.978	**0.983**	0.863	0.845	0.907	**0.926**	0.961	0.943	0.978	**0.983**
LEU	0.879	0.840	0.958	**0.972**	0.559	0.513	0.735	**0.806**	0.879	0.860	0.958	**0.972**
LUN	0.663	0.672	**0.748**	0.704	0.495	0.485	**0.547**	0.473	0.864	0.860	**0.911**	0.828
DER	0.766	0.848	0.955	**0.964**	0.838	0.818	0.905	**0.931**	0.853	0.876	0.955	**0.964**
ECO	0.552	0.496	0.505	**0.631**	0.467	0.458	0.487	**0.549**	0.739	0.770	0.808	**0.851**
GLA	0.452	0.446	0.453	**0.565**	0.320	0.298	0.333	**0.399**	0.549	0.572	**0.652**	0.650
GLI	0.585	0.548	0.559	**0.700**	0.465	0.410	0.398	**0.505**	0.619	0.569	0.601	**0.700**
IRI	0.802	0.746	0.843	**0.953**	0.640	0.514	0.655	**0.849**	0.815	0.758	0.843	**0.953**
MAL	0.911	0.731	0.902	**0.929**	0.569	0.299	0.544	**0.624**	0.911	0.743	0.902	**0.929**
MLL	0.669	0.637	0.687	**0.861**	0.435	0.376	0.426	**0.681**	0.692	0.651	0.687	**0.861**

explicitly the graph is constructed as $W_{ij} = \exp\left(-\|x_i - x_j\|^2/(\gamma\bar{r}^2)\right)$ where $\bar{r}$ denotes the average pairwise Euclidean distances among the data points and γ is chosen from $[2^{-2}, 2^{-1}, \cdots, 2^5]$ and the best results are reported.

4.3 Semi-supervised Learning

We first run graph evolution algorithm (Eq. (6)) for one iteration. After that we use the result weights as input to run Zhu *et al.*'s [19] (marked as HF in the Figure 5) and Zhou *et al.*'s [20] (marked as GC) approaches. We compare four methods, HF, GC, HF on resulting graph (HF_GE), GC on resulting graph (GC_GE), on four face image datasets. We tested the methods on AT&T[4], BinAlpha[5],

[4] http://people.cs.uchicago.edu/~dinoj/vis/ORL.zip

[5] http://www.cs.toronto.edu/~roweis/data.html

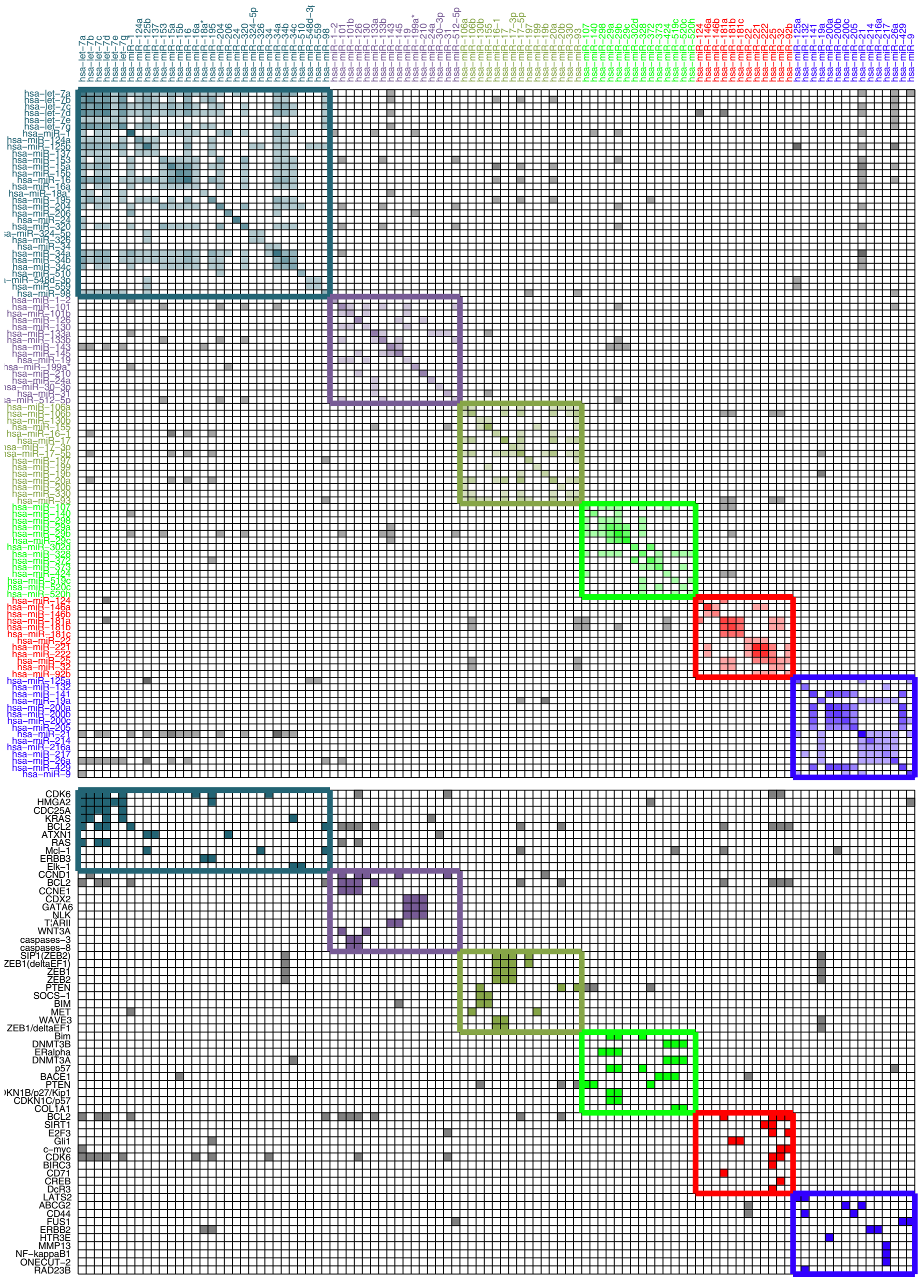


Fig. 4. 6 miRNA cliques found by Graph Evolution. Top panel is the miRNA graph in which the values denotes the number of common targeting genes of two miRNAs. The bottom panel is the top 10 targeting genes for each clique. The cliques are separated by different colors. The left top part of the top panel is the *let-7* miRNA family and the right bottom part of the top panel is the *hsa-mir-200* family.

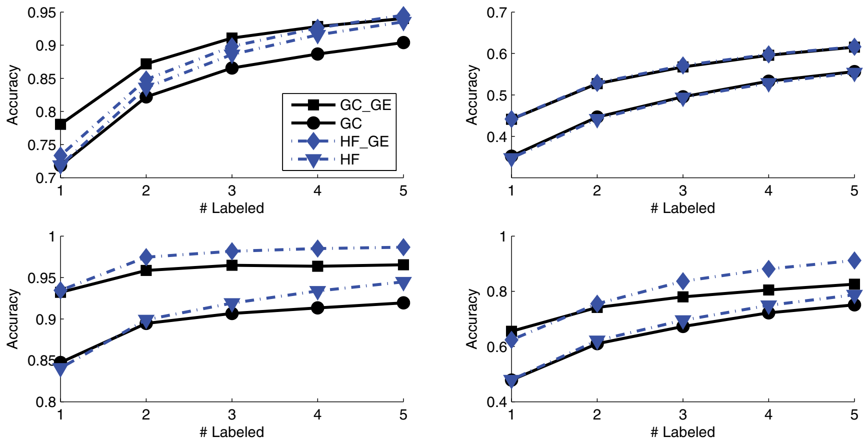


Fig. 5. Semi-supervised learning on 4 datasets(from left to right): AT&T, BinAlpha, JAFFE, and Sheffield datasets. Classification accuracies are shown for four methods: HF, GC, HF using condensated graph (HF_GE), GC using condensated graph (GC_GE). For each dataset, number of labeled data per class are set to 1, 2, 3, 4, 5. Using the graph evolution consistently improves over original methods.

JAFFE[6], and Sheffield[7] data sets. For all the methods and datasets, we randomly select N labeled images for each class, $N = 1, 2, 3, 4, 5$, and use the rest as unlabled images. We try 50 random selections for each dataset, and computer the average of the semi-supervised classification accuracy.

The results are shown in Figure 5. In all these case, we always obtain higher classification accuracy by applying graph condensation. For datasets BinAlpha, JAFFE, and Sheffield, our methods are consistently 5%–10% better than the standard semi-supervised learning methods.

4.4 Graph Evolution for microRNA Functionality Analysis

In this experiment, we are interested in the interaction network between microRNAs (miRNAs) and genes. MiRNAs play important regulatory roles by targeting messenger RNAs (mRNAs) for degradation or translational repression, and have become one of the focuses of post-transcriptional gene regulation in animals and plants[21,22,23] and have been an active research topic in various domains [24,25,26,27]. A database of verified miRNA/target gene relationship can be found in [28]. Here we apply our algorithm to investigate the relationships between the miRNAs and the genes. The main purpose is to discover new interaction patterns in the miRNA regulatory network.

We use the data with version of Nov. 6, 2010. We use the number of targeting genes as the weights of two miRNAs, *i.e.* $W_{ij} = \sum_k B_{ik} B_{jk}$ where $B_{ik} = 1$

[6] http://www.cs.toronto.edu/~roweis/data.html
[7] http://www.shef.ac.uk/eee/vie/face.tar.gz

indicates miRNA i targets gene k, $B_{ik} = 0$ otherwise. We select the largest disconnected component which has 103 miRNAs and run the **GraphEvolution** algorithm until converges. Finally, we discover 6 separated subgroups of miRNAs, which are shown in Figure 4. The following is the outline of our discovery in this experiment. (1) the *let-7* [29,30] miRNA family is correctly clustered into the same group. (2) The *hsa-mir-200* family are highly connected with each other, which is not reported in literature so far.

5 Conclusions

In this paper we present the Social Diffusion Process, which is motivated from the Matthew effect in social phenomenons. We develop the stochastic model by the assumption that social members tend to be together with someone who is familiar with. We also derive an graph evolution algorithm based on the presented mode. Empirical studies show significant improvement of the qualities of the graph data by the Social Diffusion Process, indicating that the assumptions in our model are natural in general. We also discover a new miRNA family in the experiment on miRNA functionality analysis.

Acknowledgment. This research is partially supported by NSF-CCF-0830780, NSF-DMS-0915228, NSF-CCF-0917274.

References

1. Kalton, A., Langley, P., Wagstaff, K., Yoo, J.: Generalized clustering, supervised learning, and data assignment. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 299–304. ACM, New York (2001)
2. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation 15, 1373–1396 (2003)
3. Shi, J., Malik, J.: Normalized cuts and image segmentation. In: IEEE Conf. Computer Vision and Pattern Recognition (1997)
4. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) NIPS, pp. 849–856. MIT Press, Cambridge (2001)
5. Chan, P.K., Schlag, M.D.F., Zien, J.Y.: Spectral K -way ratio-cut partitioning and clustering. In: DAC, pp. 749–754 (1993)
6. Hagen, L.W., Kahng, A.B.: New spectral methods for ratio cut partitioning and clustering. IEEE Trans. on CAD of Integrated Circuits and Systems 11, 1074–1085 (1992)
7. Nowicki, K., Snijders, T.A.B.: Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96, 1077–1088 (2001)
8. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels (2008)
9. Pitman, J.: Combinatorial stochastic processes. Technical report. Springer, New York (2002)

10. Blei, D., Griffiths, T., Jordan, M.: The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM 57, 21–30 (2010)
11. Blei, D., Frazier, P.: Distance dependent chinese restaurant processes. In: ICML (2010)
12. Jordan, M.I., Ghahramani, Z., Jaakkola, T., Saul, L.K.: An introduction to variational methods for graphical models. Machine Learning 37(2), 183–233 (1999)
13. Ishwaran, H., James, L.F.: Generalized weighted chinese restaurant processes for species sampling mixture models. Statistica Sinica 13, 1211–1236 (2003)
14. Ahmed, A., Xing, E.P.: Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In: SDM, pp. 219–230. SIAM, Philadelphia (2008)
15. van Dongen, S.: Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht (2000)
16. Merton, R.: The matthew effect in science. Science 159, 56–63 (1968)
17. Rossiter, M.W.: The matthew matilda effect in science. Social Studies of Science 23, 325–341 (1993)
18. Stanovich, K.E.: Matthew effects in reading: Some consequences of individual differences in the acquisition of literacy. Reading Research Quarterly 21, 360–407 (1986)
19. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: ICML, pp. 912–919 (2003)
20. Zhou, D., Bousquet, O., Lal, T., Weston, J., Scholkopf, B.: Learning with local and global consistency. In: NIPS, p. 321 (2003)
21. Bartel, D.P.: Micrornas: target recognition and regulatory functions. Cell 136, 215–233 (2009)
22. Bearfoot, J.L., et al.: Genetic analysis of cancer-implicated microrna in ovarian cancer. Clinical cancer research: an official Journal of the American Association for Cancer Research 14(22), 7246–7250 (2008)
23. Blenkiron, C., et al.: Microrna expression profiling of human breast cancer identifies new markers of tumor subtype. Genome Biology 8, R214+ (2007)
24. Ng, K., Mishra, S.: De novo svm classification of precursor micrornas from genomic pseudo hairpins using global and intrinsic folding measures. Bioinformatics 23, 1321–1330 (2007)
25. Huang, J.C., Frey, B.J., Morris, Q.: Comparing sequence and expression for predicting microRNA targets using genMIR3. In: Altman, R.B., Dunker, A.K., Hunter, L., Murray, T., Klein, T.E. (eds.) Pacific Symposium on Biocomputing, pp. 52–63. World Scientific, Singapore (2008)
26. Dahiya, N., Morin, P.: Micrornas in ovarian carcinomas. Endocrine-Related Cancer (2009) (in press), doi:10.1677/ERC–09–0203
27. Berezikov, E., et al.: Approaches to microrna discovery. Nat. Genet. 38(suppl.), S2–S7 (2006)
28. Jiang, Q., et al.: mir2disease: a manually curated database for microrna deregulation in human disease. Nucleic Acids Res. 37, D98–D104 (2009)
29. Hu, G., et al.: Microrna-98 and let-7 confer cholangiocyte expression of cytokine-inducible src homology 2-containing protein in response to microbial challenge. J. Immunol. 183, 1617–1624 (2009)
30. Abbott, A., et al.: The let-7 microrna family members mir-48, mir-84, and mir-241 function together to regulate developmental timing in caenorhabditis elegans. Dev. Cell 9, 403–414 (2005)