SCIENTIFIC REPORTS

OPEN

A multi-similarity spectral clustering method for community detection in dynamic networks

Received: 28 January 2016
Accepted: 15 July 2016
Published: 16 August 2016

Xuanmei Qin[1], Weidi Dai[2], Pengfei Jiao[2], Wenjun Wang[2] & Ning Yuan[3]

Community structure is one of the fundamental characteristics of complex networks. Many methods have been proposed for community detection. However, most of these methods are designed for static networks and are not suitable for dynamic networks that evolve over time. Recently, the evolutionary clustering framework was proposed for clustering dynamic data, and it can also be used for community detection in dynamic networks. In this paper, a multi-similarity spectral (MSSC) method is proposed as an improvement to the former evolutionary clustering method. To detect the community structure in dynamic networks, our method considers the different similarity metrics of networks. First, multiple similarity matrices are constructed for each snapshot of dynamic networks. Then, a dynamic co-training algorithm is proposed by bootstrapping the clustering of different similarity measures. Compared with a number of baseline models, the experimental results show that the proposed MSSC method has better performance on some widely used synthetic and real-world datasets with ground-truth community structure that change over time.

Complex networks have been studied in many domains, such as genomic networks, social networks, communication networks and co-author networks[1]. The community structure has revealed important structure in these complex networks[2–6]. A great deal of research has been devoted to detecting communities in complex networks, such as graph partitioning[7,8], hierarchical clustering[9], modularity optimization[10], spectral clustering[11,12], label propagation, game theory and information diffusion[13], a detailed review is available in the literature[14]. However, most existing methods are designed for static networks, and not suitable for real-world data networks with dynamic characteristics. For example, the interactions among users in the blogosphere or circles of friends are not stationary because some interactions disappear, and some new ones appear each day.

Recently, some methods have been proposed to find community structures and their temporal evolution in dynamic networks. An intuitive idea is to divide the network into discrete time steps and to use static methods to the snapshot networks[15–22]. The so-called two-stages methods, analyse the community extraction and the community evolution in two separated stages. In other words, the communities are extracted at a given snapshot while ignoring the changing trends among and within communities of the dynamic networks. These two-stage methods are extremely noise-sensitive and produce unstable clustering results. For example, nodes or links disappear or emerge in the subsequent snapshot, which is impossible to detect using the two-stage methods. A better choice is to consider multiple time steps as a whole and the evolutionary clustering algorithm is proposed[23], which can detect communities of the current snapshot by joining with the community structure of the previous snapshot.

In fact, evolutionary clustering algorithm enables one to detect current communities using community structures from the previous steps by introducing an item called the temporal smoothness. The general framework for evolutionary clustering was first formulated by Chakrabarti *et al.*[23]. In this framework, they proposed heuristic solutions to evolutionary hierarchical clustering and k-means clustering. The framework FacetNet, which was proposed by Lin *et al.*[24], relies on non-negative matrix factorization. A density-based clustering method, which was proposed by Kim and Han[25], and uses a cost embedding technique and optimal modularity, can efficiently find temporally smoothed local clusters of high quality.

The existing evolutionary clustering methods that are most similar to MSSC are the PCQ (preserving cluster quality) and PCM (preserving cluster membership) methods[26]. PCQ and PCM are two proposed frameworks that

[1]School of Computer Software, Tianjin University, Tianjin, 300350, China. [2]School of Computer Science and Technology, Tianjin University, Tianjin, 300350, China. [3]Department of Basic Courses, Academy of Military Transportation, PLA, Tianjin 300161, China. Correspondence and requests for materials should be addressed to P.J. (email: pjiao@tju.edu.cn)

incorporate the temporal smoothness in spectral clustering. In both frameworks, a cost function is defined as the sum of the traditional cluster quality cost and the temporal smoothness item. Our method follows the evolutionary clustering strategy, but with one major difference. The intuitive goal of spectral clustering is to detect latent communities in networks such that the points are similar in the same community and different in different communities. There are several similarity measurements to evaluate the similarities between two vertices. A common approach is to encode prior knowledge about objects using a kernel, such as the linear kernel, Gaussian kernel and Fisher kernel. A large proportion of existing spectral clustering algorithms use only one similarity measurement. However, there is a problem in that the clustering results based on different similarity matrices may be notably different[11,27]. Here, we introduce a multi-similarity method to the evolutionary spectral clustering algorithm, which simultaneously considers multiple similarity matrices.

Inspired by Abhishek Kumar *et al.*[28], we propose a multi-similarity spectral clustering (MSSC) method and a dynamic co-training algorithm for community detection in dynamic networks. The proposed method preserves the evolutionary information of community structure by combining the current data and historic partitions. The idea of co-training was originally proposed in semi-supervised learning for bootstrapping procedures where two hypotheses are trained in different views[29]. The cotraining idea assumes that the two views are conditionally independent and sufficient, i.e., each view can conditionally independently give the classifiers and be sufficient for classification on its own. Then the classification is restricted in one view to be consistent with those in other views. Co-training has been used to classify web pages using the text on the page as one view and the anchor text of hyperlinks on other pages that point to the page as the other views[30]. In another words, the text in a hyperlink on one page can provide information about the page to which it links. Similarity to semi-supervised learning, the clustering, which is based on different similarity measures, is obtained using information from one another by co-training in the proposed dynamic co-training approach. This process is repeated in a pre-defined number of iterations.

Moreover, the problem of how to determine the weight of the temporal penalty to the historic partitions, which reflects the user preferences on the historic information, remains. In many cases, this parameter depends on the users' subjective preference[26], which is undesirable. We propose an adaptive model to dynamically tune the temporal smoothness parameter.

In summary, we introduce multiple similarity measures in the evolutionary spectral clustering method. We propose a dynamic co-training method, which accommodates multiple similarity measures and regularizes current communities according to the temporal smoothness of historic ones. Then, an adaptive approach is presented to learn the change in weight of the temporal penalty over time. Based on these ideas, a multi-similarity evolutionary spectral clustering method is presented to discover communities in dynamic networks using the evolutionary clustering[23] and dynamic co-training method. The performance of the proposed MSSC method is demonstrated on some widely used synthetic and real-world datasets with ground-truths.

Results

To quantitatively compare our algorithm and others, we compare the values of the normalized mutual information (NMI)[31] and the sum of squares for error (SSE)[32] for various networks from the literature. The NMI is a well known entropy measure in information theory, which measures the similarity of two clusters (in this paper, between the community structures $\hat{G}$ obtained using our method and G obtained from the ground truth). Assume that the $i-th$ row of $\hat{G}$ indicates the community membership of the $i-th$ node (i.e., if the ith node belongs to the k-th community, then $\hat{g}_{ik} = 1$ and $\hat{g}_{ik'} = 0$ for $k \neq k'$). NMI can be defined as $NMI = 2\frac{I(\hat{G}; G))}{H(\hat{G}) + H(G)}$, which is the normalization of mutual information $I(\hat{G};\ G)$ by the average of two entropies $H(\hat{G})$ and $H(G)$. The NMI value is a quantity between 0 and 1, a higher NMI indicates higher consistency, and $NMI = 1$ corresponds to being identical. SSE can be defined as $\|\hat{G}\hat{G}^T - GG^T\|_F^2$, which measures the distance between the community structure represented by $\hat{G}$ and that represented by G. A smaller SSE, indicates a smaller difference between the prediction values and the factual values.

We compare the accuracy against three previously published spectral clustering algorithms for detecting communities in dynamic networks: the preserving cluster quality method (PCQ)[26], the preserving cluster membership method (PCM)[26] and the traditional two-stage method. PCQ and PCM are two proposed frameworks that incorporate temporal smoothness in spectral clustering. In both frameworks, a cost function is defined as the sum of the traditional cluster quality cost and a temporal smoothness one. Although these two frameworks have similar expressions for the cost function, the temporal smoothness cost in PCQ is expressed as how well the current partition clusters historic data, which makes the clusters depend on both current data and historic data, whereas the temporal cost in PCM is expressed as the difference between the current partition and the historic partition, which prevents the clusters from dramatically deviating from the recent history. The traditional two-stage method divides the network into discrete time steps and performs static spectral clustering[11] at each time step. Each approach is repeated for 10 times, and the average result and variance are presented. The parameter for PCQ and PCM is $\alpha = 0.9$. We begin by inferring communities in three synthetic datasets with known embedded communities. Next, we study two real-world datasets, where communities are identified by human domain experts. For concreteness and simplicity, we restrict ourselves in this paper to the case of two similarity measures. The proposed method can be extended for more than two similarity matrices. We choose to use the Gaussian kernel and linear kernel as the similarity measures among different data points. Then, the similarity matrices are $W_t^{(1)}(i, j) = e^{\{-\|\vec{v}_i - \vec{v}_j\|^2/(2\sigma^2)\}}$ and $W_t^{(2)} = \vec{v}_i^T \vec{v}_j$, where $\vec{v}_i$ and $\vec{v}_j$ represent the m-dimensional feature vectors and $i \neq j$. In our experiments, $\vec{v}_i$ is a column vector of the adjacency matrix A at snapshot t, which is represented by A_t. In other words, $\vec{v}_i$ is an n-dimensional feature vector. The σ is taken equal to the median of the pair-wise Euclidean distances between the data points.

	Cn		aveage degree = 16			aveage degree = 20		
			z = 4	z = 5	z = 6	z = 4	z = 5	z = 6
PCQ-NA	1	NMI	0.3562 ± 0.0455	0.0393 ± 0.0178	0.0253 ± 0.0072	0.9111 ± 0.0311	0.5383 ± 0.1074	0.1129 ± 0.0629
		SSE	4224.38 ± 260.79	6069.06 ± 81.79	6101.72 ± 70.14	813.00 ± 340.81	3071.78 ± 827.96	5636.56 ± 336.75
	3	NMI	0.3333 ± 0.0437	0.0384 ± 0.0169	0.0252 ± 0.0076	0.9142 ± 0.0384	0.5268 ± 0.1042	0.1055 ± 0.0552
		SSE	4378.28 ± 289.09	6076.68 ± 75.84	6131.48 ± 84.28	675.24 ± 263.41	3133.74 ± 792.06	5682.42 ± 298.19
	6	NMI	0.3165 ± 0.0597	0.0400 ± 0.0138	0.0253 ± 0.0056	0.9256 ± 0.0389	0.5084 ± 0.1072	0.1059 ± 0.0577
		SSE	4470.96 ± 366.32	6074.34 ± 86.10	6119.46 ± 81.34	556.82 ± 342.32	3288.68 ± 766.29	5686.82 ± 291.96
PCQ-NC	1	NMI	0.4059 ± 0.1443	0.0404 ± 0.0204	0.0274 ± 0.0105	0.9034 ± 0.0307	0.5589 ± 0.1128	0.1106 ± 0.0476
		SSE	3815.34 ± 995.63	6001.10 ± 107.43	6078.12 ± 91.28	898.88 ± 298.82	2827.82 ± 838.13	5615.80 ± 250.09
	3	NMI	0.3921 ± 0.1405	0.0394 ± 0.0194	0.0290 ± 0.0099	0.9288 ± 0.0346	0.5349 ± 0.1262	0.1031 ± 0.0428
		SSE	3898.54 ± 967.97	5999.10 ± 106.08	6053.18 ± 88.14	503.94 ± 211.46	3002.60 ± 943.07	5672.86 ± 240.05
	6	NMI	0.3670 ± 0.1218	0.0394 ± 0.0177	0.0267 ± 0.0098	0.9137 ± 0.0316	0.4958 ± 0.1084	0.0966 ± 0.0424
		SSE	4021.20 ± 858.63	6010.98 ± 88.34	6071.34 ± 70.74	646.28 ± 219.35	3295.12 ± 783.13	5710.98 ± 210.87
PCM-NA	1	NMI	0.3116 ± 0.0621	0.0412 ± 0.0158	0.0257 ± 0.0056	0.8999 ± 0.0387	0.4974 ± 0.1133	0.1054 ± 0.0568
		SSE	4539.50 ± 394.25	6053.88 ± 104.21	6121.96 ± 71.94	810.52 ± 390.29	3358.52 ± 784.04	5680.16 ± 305.19
	3	NMI	0.3109 ± 0.0709	0.0395 ± 0.0164	0.0251 ± 0.0072	0.8877 ± 0.0344	0.4980 ± 0.1169	0.1039 ± 0.0542
		SSE	4545.74 ± 419.80	6057.90 ± 93.30	6126.38 ± 83.91	948.72 ± 312.82	3346.96 ± 797.57	5682.32 ± 278.17
	6	NMI	0.3098 ± 0.0656	0.0403 ± 0.0144	0.0239 ± 0.0052	0.9294 ± 0.0323	0.4945 ± 0.1149	0.1058 ± 0.0559
		SSE	4550.38 ± 380.21	6073.36 ± 81.51	6126.62 ± 70.79	467.30 ± 173.87	3386.82 ± 824.11	5683.04 ± 293.82
PCM-NC	1	NMI	**0.6412** ± 0.1278	0.0392 ± 0.0191	0.0314 ± 0.0125	0.9004 ± 0.0484	**0.7819** ± 0.1129	0.1197 ± 0.0370
		SSE	**2155.12** ± 935.50	6012.28 ± 114.99	6049.72 ± 83.70	889.90 ± 396.37	**1286.42** ± 812.42	5578.26 ± 257.97
	3	NMI	**0.4307** ± 0.0931	0.0408 ± 0.0197	0.0267 ± 0.0057	0.8737 ± 0.0461	**0.5893** ± 0.0993	0.0852 ± 0.0242
		SSE	**3550.10** ± 735.88	5984.40 ± 107.92	6046.64 ± 40.66	860.80 ± 290.84	**2407.00** ± 657.48	5789.26 ± 122.49
	6	NMI	0.2804 ± 0.0525	0.0402 ± 0.0188	0.0233 ± 0.0067	0.8951 ± 0.0400	0.3785 ± 0.0931	0.0609 ± 0.0239
		SSE	4627.50 ± 341.80	6007.58 ± 77.13	6080.16 ± 38.87	701.52 ± 251.36	4026.64 ± 570.82	5915.96 ± 158.57
StaticSpectral	1	NMI	0.3741 ± 0.1315	0.0396 ± 0.0170	0.0284 ± 0.0107	0.9166 ± 0.0289	0.4940 ± 0.1198	0.0992 ± 0.0456
		SSE	3971.22 ± 924.64	6012.54 ± 98.31	6068.14 ± 108.58	619.68 ± 274.70	3305.72 ± 880.41	5667.04 ± 268.22
	3	NMI	0.3732 ± 0.1316	0.0394 ± 0.0176	0.0263 ± 0.0104	0.9117 ± 0.0349	0.5000 ± 0.1159	0.0985 ± 0.0454
		SSE	3988.72 ± 933.37	6005.98 ± 88.83	6083.30 ± 106.78	674.82 ± 338.08	3260.94 ± 822.32	5688.84 ± 235.99
	6	NMI	**0.3771** ± 0.1300	0.0421 ± 0.0186	0.0264 ± 0.0112	0.9080 ± 0.0483	0.4898 ± 0.1156	0.0995 ± 0.0451
		SSE	**3964.94** ± 928.45	5997.50 ± 96.17	6086.88 ± 106.90	704.44 ± 421.43	3351.84 ± 848.26	5687.16 ± 233.97
MSSC	1	NMI	0.4684 ± 0.0597	**0.0623** ± 0.0248	**0.0396** ± 0.0094	**0.9806** ± 0.0144	0.6462 ± 0.1311	**0.1352** ± 0.0669
		SSE	3461.16 ± 471.45	**5915.66** ± 132.83	**5996.36** ± 57.92	**100.40** ± 78.85	2284.44 ± 969.86	**5484.36** ± 363.86
	3	NMI	0.4108 ± 0.0747	**0.0562** ± 0.0200	**0.0378** ± 0.0099	**0.9693** ± 0.0238	0.5639 ± 0.1142	**0.1257** ± 0.0571
		SSE	3840.66 ± 526.34	**5957.56** ± 86.20	**6018.72** ± 80.61	**154.56** ± 120.58	2836.34 ± 844.02	**5560.14** ± 319.88
	6	NMI	0.3727 ± 0.0702	**0.0551 ± 0.0183**	**0.0405** ± 0.0089	**0.9641** ± 0.0276	**0.5428** ± 0.1113	**0.1265** ± 0.0594
		SSE	4059.82 ± 471.45	**5959.74** ± 113.17	**6013.50** ± 81.21	**182.72** ± 140.66	**3063.80** ± 831.10	**5548.00** ± 336.35

Table 1. The performance in different GN-benchmark networks. When parameter z = 4, 5 and 6, the average degree of each node is 16 and 20 at each snapshot, we randomly select 1, 3 and 6 nodes change their cluster membership, respectively. Notice that the value of NMI and SSE is the average for 10 snapshots.

Synthetic Datasets. *GN-benchmark network #1.* The first dataset is generated according to the description by Newman *et al.*[33]. This dataset contains 128 nodes, which are divided into 4 communities, each of which has 32 nodes. We generate data for 10 consecutive snapshots. In each snapshot from 2 to 10, the dynamics are introduced as follows: from each community we randomly select certain members to leave their original community and randomly join the other three communities. Pairs of nodes are randomly linked with a higher probability p_{in} for within-community edges and a lower probability p_{out} for between-community edges. Aloughth p_{out} is freely varied, the value of p_{in} is selected to maintain the expected degree of each vertex as a constant. When the average degree for the nodes is fixed, parameter z, which represents the mean number of edges from a node to the nodes in other communities, is sufficient to describe the data. With the increase in z, the community structure becomes indistinct. We consider three values (4, 5 and 6) for z; the average degree of each node is 16 and 20 at each snapshot. We randomly select 1, 3 and 6 nodes to change their cluster membership. The performance is significantly improved, as shown in Table 1, where *Cn* is the number of changed nodes. In general, our method has a higher NMI and a smaller SSE in most situations except when $z = 4$ (the average degree is 16) and $z = 5$ (the average degree is 20), where the NC-based PCM outperforms the MSSC method. Because space is limited, the values of NMI and SSE are the average values from snapshot 1 to 10. In Fig. 1, we intuitively show the performance under the conditions that parameter z is 5, the average degree of each node is 16 and at each snapshot, 3 nodes change their cluster membership. Figure 1(a) shows that the MSSC method always outperforms the baselines, which indicates that our method has a better accuracy. In addition, Fig. 1(b) shows that the MSSC method has a lower error in cluster membership with respect to the ground truth from a general view. In both figures, our method significantly improves the accuracy and reduces the error compared with PCQ, PCM and static spectral clustering.

GN-benchmark network #2. To compare the effectiveness as the number of communities varies, we use the second dataset with two types of data sets, which were generated by Francesco Folino and Clara Pizzuti[34]: SYN-FIX with a fixed number of communities and SYN-VAR with a variable number of communities. For SYN-FIX, the data generating method is identical to the GN-benchmark network #1. The network consists of 128 nodes, which

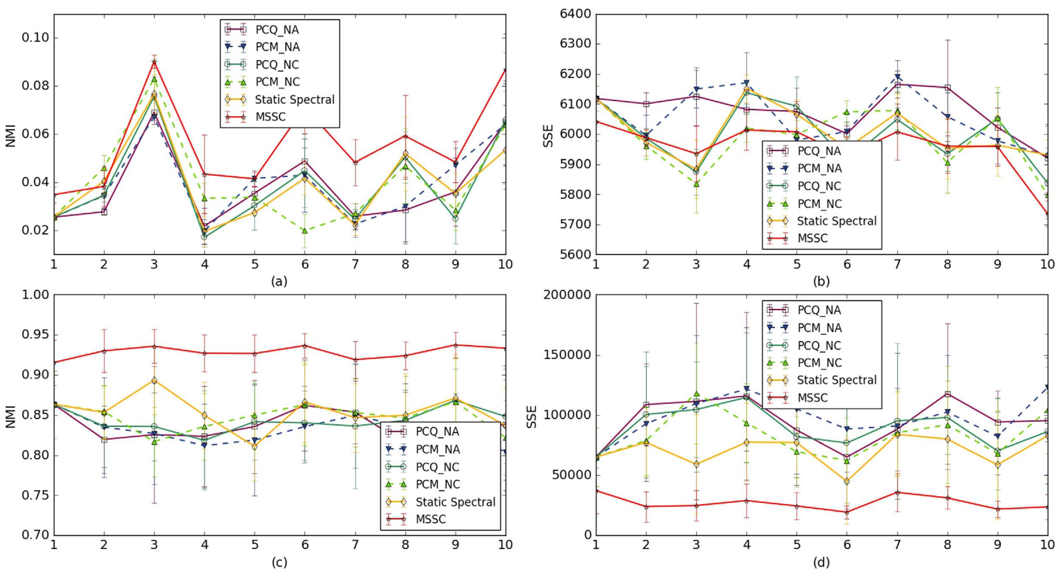

Figure 1. The performance of different methods in synthetic networks. (**a,b**) Normalized mutual information and the sum of the squared errors of different methods at 10 snapshots in synthetic networks, where the parameter z is 5, the average degree of each node is 16 and at each snapshot, 3 nodes change their cluster membership. (**c,d**) Performance for a single contraction event with 1000 nodes over 10 snapshots; the nodes have a mean degree of 15, a maximum degree of 50, and a mixing parameter value of $\mu = 0$, which controls the overlapping among communities. Notice that the x-axes show the snapshots.

		syn-fix		syn-var	
	z	NMI	SSE	NMI	SSE
PCQ-NA	3 5	0.6028 ± 0.2033 0.6069 ± 0.2016	9640.60 ± 4779.74 9340.80 ± 4642.21	0.6132 ± 0.1862 0.6116 ± 0.1928	9259.42 ± 3910.06 9174.40 ± 4197.49
PCQ-NC	3 5	**0.6133** ± 0.1840 0.6054 ± 0.1884	9319.38 ± 4134.37 9466.48 ± 4214.53	0.6002 ± 0.1798 0.5984 ± 0.1922	9841.54 ± 3830.96 9656.26 ± 4299.85
PCM-NA	3 5	0.5963 ± 0.1996 0.5993 ± 0.1978	9460.58 ± 4666.48 9304.36 ± 4568.22	0.5926 ± 0.1872 0.5834 ± 0.1928	9720.78 ± 3991.91 9855.60 ± 4235.54
PCM-NC	3 5	0.5978 ± 0.1862 **0.6109** ± 0.1943	9916.08 ± 4193.82 9271.82 ± 4401.95	0.6070 ± 0.1854 0.6139 ± 0.1907	9536.20 ± 3861.53 9123.94 ± 4136.11
StaticSpectral	3 5	0.5835 ± 0.1954 0.5781 ± 0.2052	9668.48 ± 4259.43 9755.16 ± 4679.10	0.5863 ± 0.1896 0.5812 ± 0.1862	9482.16 ± 3893.93 9659.56 ± 3723.53
MSSC	3 5	0.5852 ± 0.2052 0.6091 ± 0.2094	**2340.58** ± 1232.92 **2171.14** ± 1246.76	**0.6458** ± 0.1805 **0.6475** ± 0.1747	**8261.06** ± 4294.08 **8263.74** ± 4162.19

Table 2. The performance in different GN-benchmark networks #2. The performance for SYN-FIX and SYN-VAR with $z = 3$ and $z = 5$, respectively. For SYN-FIX, the number of communities is fixed. For SYN-VAR, a new community is created once at each timestamp between $2 \leq t \leq 5$.

are divided into four communities of 32 nodes. Every node has an average degree of 16 and shares z links with other nodes of the network. Then, 3 nodes are randomly selected from each community and randomly assigned to the other three communities. For SYN-VAR, the generating method for SYN-FIX is modified to introduce the forming and dissolving of communities and the attaching and detaching of nodes. The initial network contains 256 nodes, which are divided into 4 communities of 64 nodes. Then, 10 consecutive networks are generated by randomly choosing 8 nodes from each community, and a new community is generated with these 32 nodes. This process is performed for 5 timestamps before the nodes return to the original communities. Every node has an average degree of 16 and shares z links with the other nodes of the network. A new community is created once at each timestamp between $2 \leq t \leq 5$. Therefore, the numbers of communities between $1 \leq t \leq 10$ are 4, 5, 6, 7, 8, 8, 7, 6, 5, and 4. At each snapshot, 16 nodes are randomly deleted, and 16 new nodes are added to the network for $2 \leq t \leq 10$. Table 2 shows the accuracy and error of the community membership that are obtained by the four algorithms for SYN-FIX and SYN-VAR with $z = 3$ and $z = 5$. Table 2 shows that the MSSC method can handle dynamic networks well when the number of community varies, and when $z = 3$, the community structure is easy to detect because there is less noise. Hence, although MSSC does not perform well in NMI, it has a lower error for SYN-FIX.

Synthetic dataset #3. The third synthetic dataset is used to study the MSSC method in dynamic networks, where the number of nodes changes. Greene *et al.*[31] developed a set of benchmarks based on the embedding of events in

		birthdeath	expand	contraction	mergesplit	switch
PCQ-NA	NMI	0.8398 ± 0.0118	0.8485 ± 0.0122	0.8365 ± 0.0175	0.8515 ± 0.0096	0.8381 ± 0.0187
	SSE	74170.32 ± 18518.48	90065.30 ± 13012.11	94808.78 ± 19132.44	84288.78 ± 8577.71	101091.10 ± 20912.82
PCQ-NC	NMI	0.8504 ± 0.0247	0.8457 ± 0.0176	0.8430 ± 0.0143	0.8373 ± 0.0160	0.8432 ± 0.0143
	SSE	68511.50 ± 20307.64	92582.98 ± 14792.11	89217.64 ± 15869.03	94506.46 ± 15785.94	97056.32 ± 16190.39
PCM-NA	NMI	0.8356 ± 0.0129	0.8368 ± 0.0175	0.8316 ± 0.0175	0.8374 ± 0.0188	0.8446 ± 0.0117
	SSE	77350.20 ± 21198.80	99648.56 ± 16180.79	97976.98 ± 17861.72	90110.12 ± 14109.52	93093.64 ± 13556.13
PCM-NC	NMI	0.8419 ± 0.0168	0.8369 ± 0.0193	0.8469 ± 0.0172	0.8407 ± 0.0190	0.8359 ± 0.0153
	SSE	74547.64 ± 16656.38	101543.20 ± 17771.21	83593.96 ± 18388.87	90853.56 ± 12056.83	101398.38 ± 15773.68
StaticSpectral	NMI	0.8548 ± 0.0176	0.8574 ± 0.0154	0.8540 ± 0.0219	0.8477 ± 0.0160	0.8480 ± 0.0193
	SSE	52299.48 ± 12337.69	70764.06 ± 8715.67	70587.36 ± 12970.81	75613.28 ± 9825.15	101398.38 ± 12010.70
MSSC	NMI	**0.9335** ± 0.0076	**0.9303** ± 0.0046	**0.9284** ± 0.0076	**0.9306** ± 0.0082	**0.9360** ± 0.0084
	SSE	**18842.36** ± 5317.05	**26923.20** ± 2996.34	**27066.20** ± 5976.27	**25206.58** ± 3995.91	**24462.02** ± 4547.29

Table 3. The performance for five dynamic networks. Dynamic networks for five different event type: birth and death, expansion, contraction, merging and splitting, switch nodes.

synthetic graphs. Five dynamic networks are generated without overlapping communities for five different event types: birth and death, expansion, contraction, merging and splitting, and switch. A single birth event occurs when a new dynamic community appears, and a single death event occurs when an old dynamic community dissolutions. A single mergeing event occurs if two distinct dynamic communities observed at snapshot $t-1$ match to a single step community at snapshot t and a single splitting event occurs if a single dynamic community at snapshot $t-1$ is matched to two distinct step communities at snapshot t. The expansion of a dynamic community occurs when its corresponding step community at snapshot t is significantly larger than the previous one and the contraction of a dynamic community occurs when its corresponding step community at snapshot t is significantly smaller than the previous one. The switch event occurs when the nodes move among the communities. The performance of a small example dynamic graph produced by the generator is shown in Fig. 1(c,d), which involves 1000 nodes, 17 embedded dynamic communities and a single contraction event. To evaluate methods, we constructed five different synthetic networks for five different event types, which covered 1000 nodes over 10 snapshots. In each of the five synthetic datasets, 20% of node memberships were randomly permuted at each snapshot to simulate the natural movement of users among communities over time. The snapshot graphs share a number of parameters: the nodes have a mean degree of 15, a maximum degree of 50, and a mixing parameter value of $\mu = 0$, which controls the overlap between communities. The number of communities were constrained to have sizes in the range of [20, 100]. In each of the five synthetic datasets, the node memberships were randomly permuted at each step to simulate the natural movement of users among communities over time. Table 3 shows the performance of five different methods in different events. We also find that the standard deviation for MSSC is smaller, which implies that the clustering results are more stable.

Real-World Datasets.

NEC Blog Dataset. The blog data were collected by an NEC in-house blog crawler. Given seeds of manually picked highly ranked blogs, the crawler discovered blogs that were densely connected with the seeds, which resulted in an expanded set of blogs that communicated with each other. The NEC blog dataset has been used in several previous studies on dynamic networks[24,26,35]. The dataset contains 148, 681 entry-to-entry links among 407 blogs crawled during 15 months, which start from July 2005. First, we construct an adjacency matrix, where the nodes correspond to blogs, and the edges are interlinks among the blogs (obtained by aggregating all entry-to-entry links). In the blog network, the number of nodes changes in different snapshots. The blogs roughly form 2 main clusters, the larger cluster consists of blogs with technology focuses and the smaller cluster contains blogs with non-technology focuses (e.g., politics, international issues, digital libraries). Therefore, in the following studies, we set the number of clusters to be 2. Figure 2 shows the performance. Because the edges are sparse, we take 4 weeks as a snapshot and aggregate all edges in every month into an affinity matrix for that snapshot. Figure 2(a) shows that although MSSC does not perform as well as NA-based PCQ and PCM in the first few snapshots, MSSC begins to outperform NA-based PCQ and PCM as time progresses. In addition, MSSC retains a lower variance than NA-based PCQ and PCM. This result suggests that the benefits of MSSC accumulate more over time than those of NA-based PCQ and PCM. Furthermore, Fig. 2(b) shows that MSSC has lower errors although it does not outperform the baselines in NMI at few snapshots.

KIT E-mail Dataset. Furthermore, we consider a large number of snapshots of the e-mail communication network in the Department of Informatics at KIT[36]. The network of e-mail contacts at the department of computer science at KIT is an ever-changing graph during 48 consecutive months from September 2006 to August 2010. The vertices represent members, and the edges correspond to the e-mail contacts weighted by the number of e-mails sent between two individuals. Because the edges are sparse, we construct the adjacency matrix among 231 active members. In the E-mail network, the clusters are different departments of computer science at KIT. The number of clusters is 14, 23, 25, 26, and 27, for the snapshots of 1, 2, 3, 4, and 6 months, respectively, because the smaller divided intervals correspond to more data points that are treated as isolated points. Therefore, when we take one month as a snapshot, the number of clusters is the smallest. Because of limited space, we show the NMI scores and SSE values for the 8 snapshots situation (each snapshot is six months) in Fig. 2(c,d). We observe that MSSC outperforms the baseline methods. To study the effect of considering historic information, Table 4 takes 1, 2, 3, 4, and 6 months as a snapshot. We observe that the more snapshots, correspond to more know historic

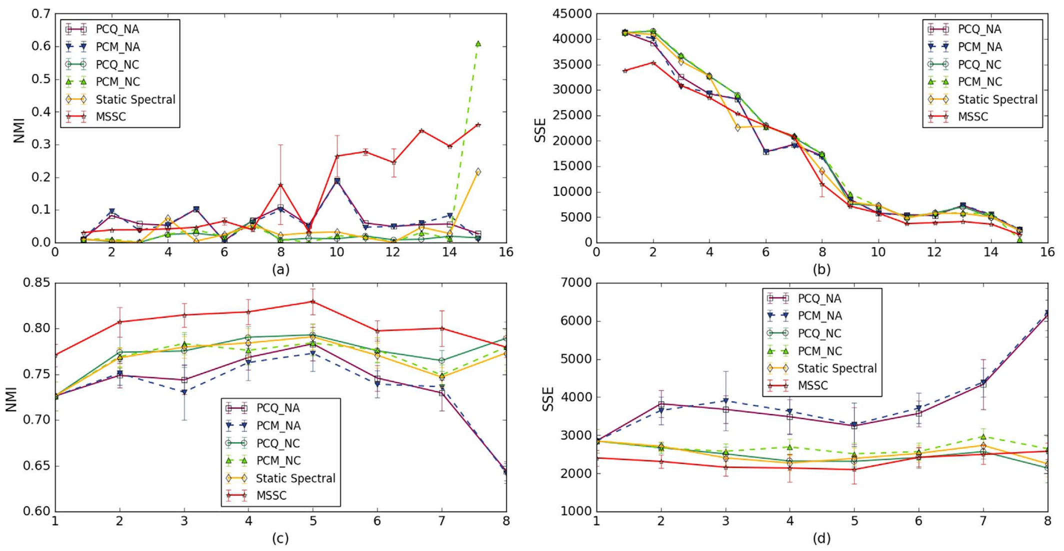

Figure 2. The performance for real-world dataset. (**a**,**b**) This NEC blog dataset contains 407 blogs crawled during 15 consecutive months, which begin from July 2005, where each month is a snapshot. (**c**,**d**) The network of e-mail contacts at the department of computer science at KIT is an ever-changing network during 48 consecutive months, where the snapshot is six months.

		T = 48	T = 24	T = 16	T = 12	T = 8
PCQ-NA	NMI	0.7567 ± 0.0365	0.7850 ± 0.0398	0.7760 ± 0.0396	0.7479 ± 0.0360	0.7362 ± 0.0416
	SSE	369.70 ± 58.83	1259.69 ± 265.12	1915.06 ± 463.29	2766.32 ± 581.48	3892.13 ± 1007.15
PCQ-NC	NMI	0.8120 ± 0.0401	0.8105 ± 0.0284	0.7933 ± 0.0287	0.7807 ± 0.0292	0.7736 ± 0.0215
	SSE	253.33 ± 60.63	924.83 ± 156.76	1371.71 ± 202.50	1862.48 ± 215.96	2476.00 ± 224.26
PCM-NA	NMI	0.7466 ± 0.0433	0.7773 ± 0.0386	0.7680 ± 0.0434	0.7378 ± 0.0398	0.7326 ± 0.0400
	SSE	382.39 ± 69.97	1290.58 ± 258.89	1949.60 ± 462.78	2856.55 ± 628.81	3954.75 ± 1018.30
PCM-NC	NMI	0.8290 ± 0.0345	0.8150 ± 0.0214	0.8076 ± 0.0248	0.7796 ± 0.0317	0.7680 ± 0.0203
	SSE	**232.27** ± 58.91	924.99 ± 109.74	1296.94 ± 174.63	1884.85 ± 230.36	2686.10 ± 152.40
StaticSpectral	NMI	0.8018 ± 0.0398	0.8059 ± 0.0249	0.7871 ± 0.0284	0.7795 ± 0.0287	0.7674 ± 0.0212
	SSE	265.45 ± 56.69	933.73 ± 135.10	1418.89 ± 226.12	1848.40 ± 201.88	2518.20 ± 224.53
MSSC	NMI	**0.8333** ± 0.0214	**0.8448** ± 0.0206	**0.8271** ± 0.0273	**0.8086** ± 0.0270	**0.8021** ± 0.0196
	SSE	241.52 ± 28.73	**846.61** ± 77.28	**1297.94** ± 177.73	**1676.28** ± 226.60	**2328.38** ± 177.97

Table 4. The performance for the KIT E-mail Dataset. The e-mail networks taking 1, 2, 3, 4, 6 months as a snapshot, respectively.

information and smaller error. Therefore the SSE is smallest when the dynamic networks are considered as 48 snapshots.

Discussion

In this paper, to find a highly efficient spectral clustering method for community detection in dynamic networks, we propose an MSSC method by considering different measures together. We first construct multiple similarity matrices for each snapshot of dynamic networks and present a dynamic co-training method that bootstrapping the clustering of different similarity measures using information from one another. Furthermore, the proposed dynamic co-training method, which considers the evolution between two neighbouring snapshots can preserve the historic information of community structure. Finally, we use a simple but effective method to adaptively estimate the temporal smoothing parameter in the objective.

We have evaluated our MSSC method on both synthetic and real-world networks with ground-truths, and compared it with three state-of-the-art spectral clustering methods. The experimental results show that the method effectively detects communities in dynamic networks for most analysed data sets with various network and community size.

In all of our experiments, we observe that the major improvement in performance is obtained in the first iteration. The performance varies around that value in subsequent iterations. Therefore, in this paper, we show the results after the first iteration. In general, the algorithm does not converge, which is also the case with the semi-supervised co-training algorithm[28].

However, the number of clusters or communities must be pre-designed in each snapshot. Determining the number of clusters is an important and difficult research problem in the field of model selection. There is currently no good resolution method for this problem. Some previously suggested approaches to this problem are cross-validation[37], minimum description length methods that use two-part or universal codes[38], and maximization of a marginal likelihood[39]. Our algorithms can use any of these methods to automatically select the number of cluster k because our algorithm still uses the fundamental spectral clustering algorithm. Additionally, as a spectral clustering method, MSSC must construct an adjacency matrix and calculate the eigen-decomposition of the corresponding Laplacian matrix. Both steps are computationally expensive. For a data set of n data points, these two steps have complexities of $O(n^2)$ and $O(n^3)$, which are unbearable burdens for large-scale applications[40]. There are some options to accelerate the spectral clustering algorithm, such as landmark-based spectral clustering (LSC), which selects $p\,(\ll n)$ representative data points as the landmarks and represents the remaining data points as the linear combinations of these landmarks[41,42]. Liu *et al.*[43] introduced a sequential reduction algorithm based on the observation that some data points quickly converge to their true embedding, so that an early stop strategy will speed up the decomposition. Yan, Huang, and Jordan[44] also provided a general framework for fast approximate spectral clustering.

Methods

Traditional spectral clustering. In this section, we review the traditional spectral clustering approach[11]. The basic idea of spectral clustering is to cluster based on the spectrum of a Laplacian matrix. Given a set of data points $\{x_1, x_2, \ldots, x_n\}$, the intuitive goal of clustering is to find a reasonable method to divide the data points into several groups, with greater similarity in each group and dissimilarity among the groups. From the view of graph theory, the data can be represented as a similarity-based graph $G=(V, E)$ with vertex set V and edge set E. Each vertex v_i in this graph represents a data point x_i, and the edge between vertices v_i and v_j is weighted by similarity W_{ij}. For any given similarity matrix W, we can construct the unnormalized Laplacian matrix by $L=D-W$ and the normalized Laplacian matrix by $\mathbb{L} = I - D^{-1/2}WD^{-1/2}$, where the degree matrix D is defined as a diagonal matrix with elements $d_{ii} = \sum_{j=1}^{n} W_{ij}$. The adjacency matrix is a square matrix A, such that its element A_{ij} is one when there is an edge from vertex v_i to vertex v_j and is zero when there is no edge. Two common variants of spectral clustering are average association and normalized cut[45]. The two partition criteria that maximize the association with the group and minimize the disassociation among groups are identical (the proof is provided in the literature[45]). Unfortunately, each variant is associated with an NP-hard problem. The relaxed problems can be written as[11,26,45]

$$\min_{Z\in R^{n\times k}} tr(Z^T\mathbb{L}Z) \quad subject\ to\ Z^TZ = I \tag{1}$$

In our algorithm, we will use the normalized cut as the partition criteria. The optimal solution to this problem is to set Z to be the eigenvectors that correspond to the k smallest eigenvalues of $\mathbb{L}$. Then, all data points are projected to the eigen-space and the k-means algorithm is applied to the projected points to obtain the clusters. The focus of our work is the definition of the similarity matrix in the spectral clustering algorithm, i.e. computing the relaxed eigenvectors Zs with different similarity measurements.

Different similarity measures. In spectral clustering, a similarity matrix should be constructed to quantify the similarity among the data points. The performance of the spectral clustering algorithm heavily depends on the choice of similarity measures[46]. There are several constructions to transform a given set of data points into their similarities. A common approach in machine learning is to encode prior knowledge about the data vertices using a kernel[27]. The linear kernel which is given by the inner products between implicit representations of data points, is the simplest kernel function. Assume that the ith node in V can be represented by an m-dimensional feature vector $\vec{v}_i \in R^m$, and the distance between the ith and jth nodes in V is $\|\vec{v}_i - \vec{v}_j\|$, which is the Euclidean distance. The linear kernel can be used as a type of similarity measure, i.e., similarity matrix W can be solved by $W_{ij} = \vec{v}_i^{\,T}\vec{v}_j$. The Gaussian kernel function is one of the most common similarity measures for spectral clustering[11], which can be written as $W_{ij} = e^{\{-\|\vec{v}_i-\vec{v}_j\|^2/(2\sigma^2)\}}$, where the standard deviation of the kernel σ is equal to the median of the pair-wise Euclidean distances between the data points.

There are also some specific kernels for the similarity matrix. Fischer and Buhmann[47] proposed a path-based similarity measure based on a connectedness criterion. Chang *et al.*[48] proposed a robust path-based similarity measure based on the M-estimator to develop the robust path-based spectral clustering method.

Different similarity measures may reveal similarity between data points from different perspectives. For example, the Gaussian kernel function is based on Euclidean distances between the data points, whereas the linear kernel function is based on the inner products of the implicit representations of data points. Most studies of spectral clustering are based on one type of similarity measure, and notably few works consider multiple similarity measures. Therefore, we propose a method to consider multiple similarity measures in spectral clustering. In other words, our goal is to find a spectral clustering method based on multiple similarity matrice.

Multi-similarity spectral clustering. First, we introduce basic ideas on multi-similarity spectral clustering in the dynamic networks. We assume that the clustering from one similarity measurement should be consistent with the clustering from the other similarity measurements, and we bootstrapping the clustering of different similarities using information from one another by a dynamic co-training. The dynamic co-training method based on the idea of evolutionary clustering can preserve historic information of community structure. After a new similarity matrix is obtained by the dynamic co-training, we follow the standard procedures in traditional spectral clustering and obtain the clustering result. Figure 3 graphically illustrate the dynamic co-training process.

Specifically, we first compute the similarity matrices with different similarity measures at snapshot t, and the pth similarity matrix is denoted by $W_t^{(p)}$. Following most spectral clustering algorithms, a solution to the problem of minimizing the normalized cut is the relaxed cluster assignment matrix $Z_t^{(p)}$ whose columns are the eigenvec-

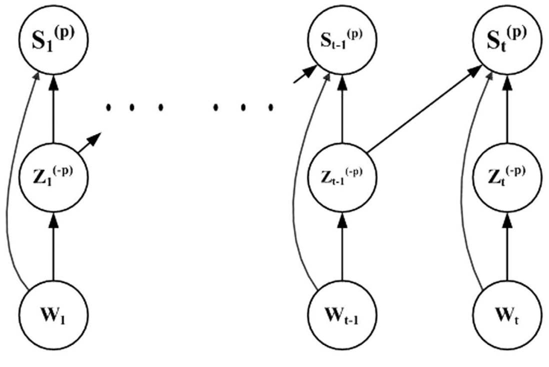

Figure 3. The graphical illustration of the dynamic co-training method. W_t represents the similarity matrix at snapshot $t\,S_t^{(p)}$ represents the new similarity matrix after the dynamic co-training. $Z_t^{(-p)}$ denotes the discriminative eigenvector in the Laplacian matrix obtained from the 1, 2, 3 … sth except for the pth similarity measures.

tors associated with the first k eigenvalues of the normalized Laplacian matrix $\mathbb{L}_t^{(p)}$. Then all data points are projected to the eigen-space, and the clustering result is obtained usin the k-means algorithm. For a Laplacian matrix with exactly k connected components, its first k eigenvectors are are the cluster assignment vectors, i.e., these k eigenvectors only contain discriminative information among different clusters, while ignoring the details in the clusters[11]. However, if the Laplacian matrix is fully connected, the eigenvectors are no longer the cluster assignment vectors, but they contain discriminative information that can be used for clustering. From the co-training, we can use the eigenvectors from one similarity matrix to update the other one. The updated similarity matrix on the pth similarity measure at snapshot t can be defined as

$$S_t^{(p)} = sym\left(\left(\sum_{q\neq p} Z_t^{(q)} Z_t^{(q)^T}\right) W_t^{(p)}\right) \tag{2}$$

$$sym(Y) = (Y + Y^T)/2 \tag{3}$$

where $Z_t^{(q)}$ denotes the discriminative eigenvector in the Laplacian matrix from the qth similarity measure, p, $q = 1, 2, 3, \ldots s$ and $p \neq q$. Equation (3) is the symmetrization operator to ensure that the projection of similarity matrix $W_t^{(p)}$ onto the eigenvectors is a symmetric matrix. Then, we use $S_t^{(p)}$ as the new similarity matrix to compute the Laplacians and solve for the first k eigenvectors to obtain a new cluster assignment matrix $Z_t^{(p)}$. After the co-training procedure is repeated for a pre-selected number of iterations, matrix $V = Z_t^{(p)}$ is constructed, where p is considered the most informative similarity measure in advance. Alternatively, if there is no prior knowledge of the similarity informativeness, matrix V can be set to be the column-wise concatenation of all $Z_t^{(p)}$s. For example, we generate two cluster assignment matrices $Z_t^{(1)}$ and $Z_t^{(2)}$, which are combined to form $V = [Z_t^{(1)} Z_t^{(2)}]$. Finally, the clusters are obtained using the k-means algorithm on V.

As descibed, we can solve the problem to accommodate multiple similarities. A further consideration is to follow the evolutionary clustering strategy to preserve the historic information of the community structure based on the co-training method. A general framework for evolutionary clustering was proposed by a linear combination of two costs[26]:

$$Cost = \alpha \cdot CS + (1 - \alpha) \cdot CT \tag{4}$$

where CS measures the snapshot quality of the current clustering result with respect to the current data features, CT measures the goodness-of-fit of the current clustering result with respect to either historic data features or historic clustering results.

Here, we assume that the clusters at any snapshot should mainly depend on the current data and should not dramatically shift to the next snapshot. Then, a better approximation to the inner product of the feature matrix and its transposition is define as

$$Z_t^{(q)} Z_t^{(q)^T} = \alpha_t^{(q)} Z_t^{(q)} Z_t^{(q)^T} + (1 - \alpha_t^{(q)}) Z_{t-1}^{(q)} Z_{t-1}^{(q)^T} \tag{5}$$

where $0 \leq \alpha_t^{(q)} \leq 1$, and $\alpha_t^{(q)}$ is the temporal penalty parameter that controls the weight on the current information and historic information. Notice that $Z_t^{(q)} Z_t^{(q)^T}$ is determined by both current eigenvectors and historic eigenvectors, so the updated similarity $S_t^{(p)}$ defined in Equation (2), which considers the history, produces stable and consistent clusters. With the increase in $\alpha_t^{(q)}$, more weight is placed on the current information, and less weight is placed on the historic information. Algorithm 1 describes the MSSC algorithm in detail.

```
Algorithm 1 multi-similarity spectral clustering algorithm
Input: multiple similarity matrices, $\{W_t^{(p)}\}_{t=1}^T$ for $p = 1, 2, \ldots, s$;
Output: Assignments to $k$ clusters at time $t$;
1:  Initial: Computing Laplacian matrices $\{\mathbb{L}_t^{(p)}\}_{t=1}^T$;
2:  compute the original cluster assignment matrices
3:  $Z_t^{(p)} = \underset{Z \in R^{n \times k}}{argmintr} (Z_t^{(p)^T} \mathbb{L}_t^{(p)} Z_t^{(p)})\ \ s.t.\ \ Z_t^{(p)^T} Z_t^{(p)} = I$;
4:  for $t = 1$ to $T$ do
5:      for $i = 1$ to $iternum$ do
6:          co-training to obtain the new similarity matrices $S_t^{(p)} = sym\left(\left(\sum_{z \neq p} \alpha_t^{(q)} Z_{t,i-1}^{(q)} Z_{t,i-1}^{(q)^T} + (1 - \alpha_t^{(q)}) Z_{t-1}^{(q)} Z_{t-1}^{(q)^T}\right) W_t^{(p)}\right)$, $q = 1, 2, \ldots, s$;
7:          use $S_t^{(p)}$ as new similarity matrices to compute Laplacian matrices and solve for the first $k$ eigenvectors $Z_{t,i}^{(p)}$;
8:      end for
9:      row-normalized $Z_t^{(p)}$. $Y_t^p(i, j) = Z_t^p(i, j) / \left(\sum_j Z_t^{(p)}(i, j)^2\right)^{1/2}$;
10:     constructing $V = Y_t^{(p)}$, where $p$ considered the most informative similarity measure in advance. If there is no more prior knowledge on the informativeness, $V$ can also be set as the column-wise concatenation of multiple feature matrices $Y_t^{(p)}$s;
11:     apply the k-means algorithm to $V$ to obtain the clusters;
12: end for
```

Determining α. We have presented our proposed MSSC method. However, the temporal smoothing parameter $\alpha_t^{(p)}$ remains unknown, which prevents the clustering result at any snapshot from significantly deviating from the clustering result in the neighbouring snapshot. In many cases, the parameter depends on the subjective preference of the user. To work around this problem, Kevin S. Xu[49] presented a framework that adaptively estimated the optimal smoothing parameter using shrinkage estimation. In this section, we propose a different approach to adaptively estimate the parameter, which can be defined as

$$\alpha_t^{(p)} = 1 - \frac{\| W_t^{(p)} - W_{t-1}^{(p)} \|_F}{\| W_t^{(p)} \|_F} \tag{6}$$

Note that $\alpha_t^{(p)}$ can be easily estimated because $W_t^{(p)}$ is known. In this model, more weight is placed on the current similarity, because the data should not dramatically shift to the neighbouring snapshot. Further more, a large difference in W indicates a small α, so it takes more information from the past.

Changing community numbers. We have assumed that the number of community k is fixed, which is a notably strong restriction to the application of our approach. In fact, our approach can handle variations in community numbers. When the community numbers are different at two neighbouring snapshots, the approximation in Equation (5) is free from the effect of changes in clusters, i.e., $Z_t^{(q)} Z_t^{(q)^T}$ and $Z_{t-1}^{(q)} Z_{t-1}^{(q)^T}$ is independent of the community numbers.

Inserting and removing nodes. In many real-world networks, new nodes join or existing nodes leave the networks often. Assume that at time t, old nodes are removed from and new nodes are inserted into the network. We handle this problem by applying some heuristic solution to transform $W_{t-1}^{(p)}$ and $Z_{t-1}^{(p)}$ to the same dimension as $W_t^{(p)}$ and $Z_t^{(p)}$, respectively[26]. When old nodes are removed, we can remove the corresponding rows from $Z_{t-1}^{(p)}$ in Equation (5) to obtain $\tilde{Z}_{t-1}^{(p)}$(assuming that $\tilde{Z}_{t-1}^{(p)}$ is $n_1 \times k$). When new nodes are inserted, we must extend $\tilde{Z}_{t-1}^{(p)}$ to $\hat{Z}_{t-1}^{(p)}$, which has the identical dimension as $Z_t^{(p)}$(assuming the dimension of $Z_t^{(p)}$ is $n_2 \times k$). Then, $\hat{Z}_{t-1}^{(p)}$ is defined as

$$\hat{Z}_{t-1}^{(p)} = \begin{bmatrix} \tilde{Z}_{t-1}^{(p)} \\ G_{t-1} \end{bmatrix} \ \ for\ \ G_{t-1} = \frac{1}{n_1} \vec{1}_{n_1} \vec{1}_{n_2 - n_1}^T \tilde{Z}_{t-1}^{(p)} \tag{7}$$

For Equation (6), when old nodes are removed, we can remove the corresponding rows and columns from $W_{t-1}^{(p)}$ to obtain $\tilde{W}_{t-1}^{(p)}$ (assuming that $\tilde{W}_{t-1}^{(p)}$ is $n_1 \times n_1$). When new nodes are inserted, we add the corresponding rows and columns to obtain $\hat{W}_{t-1}^{(p)}$, which has the identical dimension as $W_t^{(p)}$(assuming that the dimension of $W_t^{(p)}$ is $n_2 \times n_2$). $\hat{W}_{t-1}^{(p)}$ can be defined as

$$\hat{W}_{t-1}^{(p)} = \begin{bmatrix} \tilde{W}_{t-1}^{(p)} & E_{t-1} \\ E_{t-1}^T & F_{t-1} \end{bmatrix} \ \ for \ \ \begin{cases} E_{t-1} = \dfrac{1}{n_1} \tilde{W}_{t-1}^{(p)} \vec{1}_{n_1} \vec{1}_{n_2 - n_1}^T \\ F_{t-1} = \dfrac{1}{n_1^2} \vec{1}_{n_2 - n_1} \vec{1}_{n_1}{}^T \tilde{W}_{t-1}^{(p)} \vec{1}_{n_1} \vec{1}_{n_2 - n_1}^T \end{cases} \tag{8}$$

References

1. Girvan, M. & Newman, M. E. J. Community structure in social and biological networks. *PNAS* **99,** 7821–7826 (2002).
2. Adamic, L. A. & Adar, E. Friends and neighbors on the web. *SN* **25,** 211–230 (2003).
3. Barabási, A. L. *et al.* Evolution of the social network of scientific collaborations. *Phys. A.* **311,** 590–614 (2002).
4. Eckmann, J.-P. & Moses, E. Curvature of co-links uncovers hidden thematic layers in the World Wide Web. *PNAS* **99,** 5825–5829 (2002).
5. Flake, G., Lawrence, S., Giles, C. & Coetzee, F. Self-organization and identification of Web communities. *Computer* **35,** 66–71 (2002).
6. Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N. & Barabási, A. L. Hierarchical organization of modularity in metabolic networks. *Science* **297,** 1551–1555 (2002).
7. Kernighan, B. & Lin, S. An Efficient Heuristic Procedure for Partitioning Graphs. *AT&T. Tech. J.* **49,** 291–307 (1970).
8. Barnes, E. R. An algorithm for partitioning the nodes of a graph. *SIAM J. Algebra Discr.* **3,** 541–550 (1982).
9. Dempster, A. P., Laird, N. M. & Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *J. Roy. Stat. Soc. B.* **39,** 1–38 (1977).
10. Newman, M. E. J. Modularity and community structure in networks. *P. Natl. Acad. Sci.* **103,** 8577–8582 (2006).
11. Von Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **17,** 395–416 (2007).
12. Rohe, K., Chatterjee, S. & Yu, B. Spectral clustering and the high-dimensional stochastic blockmodel. *Ann. Statist.* **39,** 1878–1915 (2011).
13. Alvari, H., Hajibagheri, A. & Sukthankar, G. Community detection in dynamic social networks: A game-theoretic approach. In *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*, 101–107 (2014).
14. Fortunato, Santo Community detection in graphs. *Phys. Rep.* **486,** 75–174 (2010).
15. Asur, S., Parthasarathy, S. & Ucar, D. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *TKDD* **3,** 16:1–16:36 (2009).
16. Kumar, R., Novak, J., Raghavan, P. & Tomkins, A. *On the bursty evolution of blogspace* ACM Press, 568–576 (2003).
17. Kumar, R., Novak, J. & Tomkins, A. Structure and evolution of online social networks. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 611–617 (ACM, 2006).
18. Leskovec, J., Kleinberg, J. & Faloutsos, C. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* 177–187 (ACM, 2005).
19. Lin, Y.-R., Sundaram, H., Chi, Y., Tatemura, J. & Tseng, B. L. Blog community discovery and evolution based on mutual awareness expansion. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, 48–56 (IEEE Computer Society, 2007).
20. Palla, G., Barabasi, A.-l. & Vicsek, T. Quantifying social group evolution *Nature* **446,** 664–667 (2007).
21. Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y. & Schult, R. Monic: Modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 706–711 (ACM, 2006).
22. Toyoda, M. & Kitsuregawa, M. Extracting evolution of web communities from a series of web archives. In *Proceedings of the Fourteenth ACM Conference on Hypertext and Hypermedia* (eds Ashman, H., Brailsford, T. J., Carr, L. & Hardman, L.) 28–37 (ACM, 2003).
23. Chakrabarti, D., Kumar, R. & Tomkins, A. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 554–560 (ACM, 2006).
24. Lin, Y.-R., Chi, Y., Zhu, S., Sundaram, H. & Tseng, B. L. Facetnet: A framework for analyzing communities and their evolutions in dynamic networks. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, 685–694 (ACM, 2008).
25. Kim, M.-S. & Han, J. A particle-and-density based evolutionary clustering method for dynamic networks. *PVLDB* **2,** 622–633 (2009).
26. Chi, Y., Song, X., Zhou, D., Hino, K. & Tseng, B.-L. On evolutionary spectral clustering. *ACM Trans. Knowl. Discov. Data* **3,** 17:1–17:30 (2009).
27. Srebro, N. How good is a kernel when used as a similarity measure. In *Learning Theory: 20th Annual Conference on Learning Theory, COLT 2007, San Diego, CA, USA; June 13–15, 2007. Proceedings* Vol. 4539 (eds Bshouty, N. H. & Gentile, C.) 323–335 (Springer, 2007).
28. Kumar, A. & Daumé, H., III A co-training approach for multi-view spectral clustering. In *ICML* (eds Getoor, L. & Scheffer, T.) 393–400 (Omnipress, 2011).
29. Blum, A. & Mitchell, T. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 92–100 (ACM, 1998).
30. Lee, L. of referencing in *Computer Science: Reflections on the Field, Reflections from the Field* Ch. 6, 101–126 (The National Academies Press, 2004).
31. Greene, D., Doyle, D. & Cunningham, P. Tracking the evolution of communities in dynamic social networks. In *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, 176–183 (IEEE Computer Society, 2010).
32. Lin, Y.-R., Chi, Y., Zhu, S., Sundaram, H. & Tseng, B. L. Analyzing Communities and Their Evolutions in Dynamic Social Networks. In *ACM Trans. Knowl. Discov. Data* **3,** 8:1–8:31 (2009).
33. Newman, M. E. J. & Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **69,** 026113 (2004).
34. Folino, F. & Pizzuti, C. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE T. Knowl. Data En.* **26,** 1838–1852 (2014).
35. Yang, T., Chi, Y., Zhu, S., Gong, Y. & Jin, R. Detecting communities and their evolutions in dynamic social networks—abayesian approach. *Mach. Learn.* **82,** 157–189 (2010).
36. Karlsruhe institue of technology Dynamic network of email communication at department of informatics at karlsruhe institue of technology (kit). Available at: http://i11www.iti.uni-karlsruhe.de/en/projects/spp1307/emaildata (Accessed: August 2015) (2011).
37. Airoldi, E. M., Blei, D. M., Fienberg, S. E. & Xing, E. P. Mixed membership stochastic blockmodels. *J. Mach. Learn. Res.* **9,** 1981–2014 (2008).
38. Rosvall, M. & Bergstrom, C. T. An information-theoretic framework for resolving community structure in complex networks. *P. Natl. Acad. Sci.* **104,** 7327–7331 (2007).
39. Hofman, J. M. & Wiggins, C. H. Bayesian approach to network modularity. *Phys. Rev. Lett.* **100** 258701 (2008).
40. Chen, X. & Cai, D. Large scale spectral clustering with landmark-based representation. In *AAAI* (eds Burgard, W. & Roth, D.) (AAAI Press, 2011).
41. Cai, D. Compressed spectral regression for efficient nonlinear dimensionality reduction. In *IJCAI* (eds Yang, Q. & Wooldridge, M.) 3359–3365 (AAAI Press, 2015).
42. Cai, D. & Chen, X. Large scale spectral clustering via landmark-based sparse representation. *IEEE T. Cybernetics* **45,** 1669–1680 (2015).
43. Liu, T.-Y., Yang, H.-Y., Zheng, X., Qin, T. & Ma, W.-Y. Fast large-scale spectral clustering by sequential shrinkage optimization. In *ECIR*, 4425 (eds Amati, G., Carpineto, C. & G, R.) 319–330 (Springer, 2007).
44. Yan, D., Huang, L. & Jordan, M. I. Fast approximate spectral clustering. In *KDD* (eds IV, J. F. E., Fogelman-Soulié, F., Flach, P. A. & Zaki, M.) 907–916 (ACM, 2009).
45. Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22,** 888–905 (2000).

46. Bach, F. R. & Jordan, M. I. Learning spectral clustering. In *NIPS* (eds Thrun, S., Saul, L. K. & Schölkopf, B.) 305–312 (MIT Press, 2003).
47. B, F. & Buhmann, J. M. Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **25,** 513–518 (2003).
48. Chang, H. & Yeung, D.-Y. Robust path-based spectral clustering. *Pattern Recogn.* **41,** 191–203 (2008).
49. Xu, Kevin S., Kliger, Mark & Hero, III, Alfred O. Adaptive evolutionary clustering. *Data Min. Knowl. Discov.* **28,** 304–336 (2013).

Acknowledgements
This work was supported by the Major Project of National Social Science Fund(14ZDB153)and the major research plan of the National Natural Science Foundation (91224009, 51438009).

Author Contributions
X.Q. and P.J. conceived and designed the experiments; X.Q., P.J., W.D., W.W. and N.Y. analysed the results and reviewed the manuscript.

Additional Information
Competing financial interests: The authors declare no competing financial interests.

How to cite this article: Qin, X. *et al.* A multi-similarity spectral clustering method for community detection in dynamic networks. *Sci. Rep.* **6**, 31454; doi: 10.1038/srep31454 (2016).