Edge Representation Learning for Community Detection in Large Scale Information Networks

Suxue Li[1], Haixia Zhang[1], Dalei Wu[2], Chuanting Zhang[1], and Dongfeng Yuan[1⋆]

[1] Shandong Provincial Key Laboratory of Wireless Communication Technologies, Shandong University, Jinan, 250100, Shandong, China.
[2] The University of Tennessee at Chattanooga, Chattanooga, TN 37403, USA
li_suxue@163.com,haixia.zhang@sdu.edu.cn,dalei-wu@utc.edu, chuanting.zhang@gmail.com,dfyuan@sdu.edu.cn

Abstract. It is found that networks in real world divide naturally into communities or modules. Many community detection algorithms have been developed to uncover the community structure in networks. However, most of them focus on non-overlapping communities and the applicability of these work is limited when it comes to real world networks, which inherently are overlapping in most cases, e.g. Facebook and Weibo. In this paper, we propose an overlapping community detection algorithm based on edge representation learning. Firstly, we sample a series of edge sequences using random walks on graph, then a mapping function from edge to feature vectors is automatically learned in an unsupervised way. At last we employ the traditional clustering algorithms, e.g. K-means and its variants, on the learned representations to carry out community detection. To demonstrate the effectiveness of our proposed method, extensive experiments are conducted on a group of synthetic networks and two real world networks with ground truth. Experiment results show that our proposed method outperforms traditional algorithms in terms of evaluation metrics.

Keywords: Network · Community detection · Representation learning · Cluster

1 Introduction

Networks that represent real world systems are everywhere in human life, such as biology, sociology, computer science, and academics [5][15]. An information network represents an abstraction of the real world, it provide us a useful tool to mine knowledge from links in it. Network analysis helps people solve real life problems. The study of networks is of great importance and attracts a lot of experts into it. One of the most important properties of network is community structure. The main purpose of community detection is to uncover the inherent structure of networks. Networks in real world are always complicated

⋆ Corresponding Author

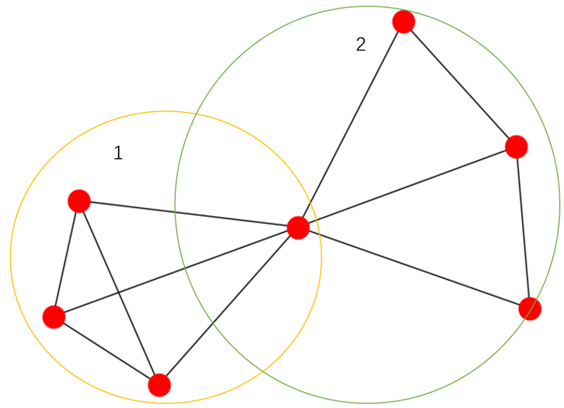


Fig. 1: A sample network

and large-scaled. It is hard to process and analyze it directly. By community detection techniques, networks are divided into different parts, the structure becomes obvious and the networks are more understandable which is helpful for subsequent analysis. Finding communities becomes a critical issue to study and explore networks. There are some typical applications listed as follows:

· **Advertising.** People in the same community often share similar interests. If we know a person buys a product, we can post advertisements about similar products to members who are in the same community with her. In this way, it can help improve the performance of product recommendation system.

· **Recommendation.** In social networking software, e.g. Facebook and Weibo, based on the community structure, we can recommend to a user those who are in the same community but not his friends yet.

· **Information propagation and Disease diffusion.** For overlapping community detection, we can find the fuzzy nodes that belong to more than one communities. Finding this kind of nodes is critical to speed up information propagation and control disease diffusion.

· **Information retrieval.** Words with similar meaning tends to be in the same community. When a user search a keyword, the results of the keyword and its near-synonym can be presented to the user simultaneously. Thus community detection helps promote personalized services.

The majority of existing community detection algorithms focus on finding non-overlapping communities. However networks in real world are complicated and nodes in them often belong to many different communities, especially in social networks. In social networks, this kind of overlapping communities reflect different types of associations between people. [6] For example, two users in Weibo could be relatives, classmates, sport partners, etc. User A could be a classmate of B and a sport partner of C at the same time, so she should be in the community of classmates of B and in the community of sport partners

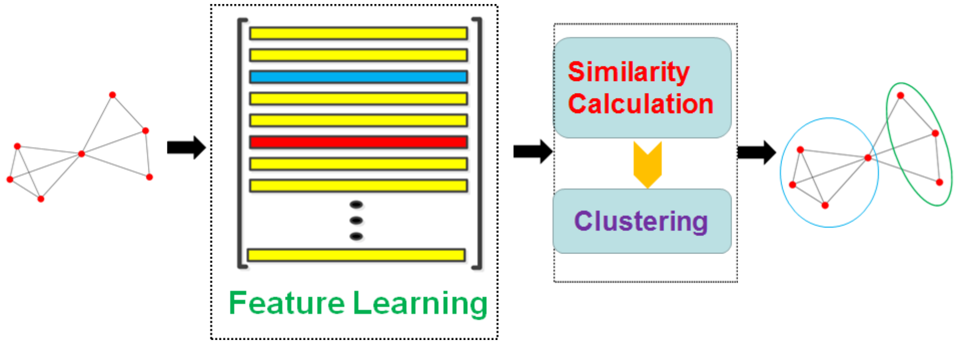


Fig. 2: The architecture of newly proposed machine learning methods.

of C simultaneously. What's more, A is important in the network, because B's classmates and C' sport partners can build connection through A. As shown in Figure 1, there are two communities which both have four members. It is obvious that there is one fuzzy node belongs to both community 1 and community 2. This node plays an important role in this network, it serves as a bridge between two communities. When messages are propagated among different communities in a network, the propagation path will always go through this kind of fuzzy nodes. Thus finding such nodes which is called overlapping community detection is a critical issue in community detection. For this reason, it is an effective way to recover overlapping communities by grouping edges, rather than vertices. Each edge is assigned to a single community, but clusters can still overlap because edges in different communities may share one endpoint.

In recent years, some algorithms based on machine learning methods appear. The architecture of these methods are shown in Figure 2. Given a network, the first thing is to extract the node feature representations. Then the similarity calculation part is done. There are a variety of similarity calculation methods, e.g. cosine similarity. The last step is traditional clustering which is based on the calculated node similarities. For this architecture, it is important how to learn representations of the given networks, because network representations affect the performance of the following community detection progress. In this paper, we propose a new method based on edge representation learning which can not only detect overlapping communities but also improve the performance of community detection. We sample a series of edge sequences by random walks on graph, then utilize unsupervised machine learning method to learn a mapping from edges to feature representations. After the mapping progress, we divide edges into groups by traditional clustering method, e.g. k-means algorithm [8] and its variants. Each edge connects two nodes, that is to say, one edge corresponds two nodes. In this way, we transform edges communities into corresponding nodes communities. As a result, the fuzzy nodes in networks that belong to different communities simultaneously can be detected. Our contributions are:

(1) We propose a scalable edge embedding method to learn edge representations based on language modeling tools.

(2) We apply edge representation learning method to community detection, and propose a new way to find overlapping communities by clustering edges.

(3) We conduct multiple experiments on synthetic networks and real world networks to test the performance of proposed algorithm.

The rest of the paper is organized as follows. Section 2 overviews related work in community detection domain. Section 3 introduces edge representation learning model. In section 4, we present steps on how to perform CD-ERL (community detection based on edge representation learning) algorithm. Section 5 displays the experiment and results. Section 6 concludes the paper.

2 Related Work

In the past few years, a lot of community detection algorithms have been proposed [11][24]. Even though they are both for finding community structure, the principles are different. Some algorithms are based on modularity optimization, such as FN (Fast-Newman) algorithm [17]; some are based on graph partitioning, such as KL (Kernighan-Lin) algorithm [9]; there are also some algorithms which are based on label propagation [19], spectral clustering [16] and dynamic networks [2]. However most of them just fit networks with clear structure.

With the rapid development of representative learning for natural language processing, there are some new representative methods proposed in network field [18] [21] [1] [23], such as node2vec [7]. These methods provide a new guidance for network analysis. However in community detection area, these methods that divide nodes directly can only find non-overlapping communities, while communities in real world networks are always overlapping. They ignore the importance of the nodes that belong to more than one communities.

Inspired by newly proposed representative learning methods, we proposed a CD-ERL algorithm to detect overlapping communities by clustering edges. The same way as representative learning methods for natural language processing, we sample edge sequence according to network links, and learn a mapping from edges to vectors. However, there are many representative learning methods used in community detection area, they only consider nodes representation and nodes clustering which lead to non-overlapping communities. We overcome this by learning edge representations, and by clustering edges, we can get both edge communities and corresponding node communities. The node communities we detect are overlapping, and the fuzzy nodes can be detected.

3 Edge Representation Learning Model

In this paper we formulate the edge representation learning progress as a maximum likelihood problem by extending representative learning for natural language processing into networks. Feature representation learning methods based on Skip-gram architecture[13] have been originally developed in the context of natural language. Here we use Skip-gram model as an example to do community detection task.

Let G = (V, E) be a given network. Here V is the set of nodes, E is the set of edges in the network. We assume G is an undirected, unweighted network here. Our goal is to embed edges in E into d-dimensional feature space R^d. Let $f : E \rightarrow R^d$ be the mapping function from edges to feature matrix, while $R^d(e)$ is the feature representation of edge e. We regard edges in a graph as words in a document, and edge neighbors as words before and after the target word. For e in E, let $N(e)$ represents the neighbors of edge e. In the embedding progress, we want to preserve the neighbor relations as much as possible. Thus the objective function is as follows:

$$\max \sum_{e \in E} \log Pr(N(e)|R^d(e)) \tag{1}$$

We assume that given the feature representation of an edge, finding one neighbor edge is independent of finding another neighbor edge. What's more, interactive edges have symmetry effect on each other. We use a softmax unit to model the conditional likelihood of the source edge and its neighbor. In these conditions, the objective function can be written in this form:

$$\max \sum_{e \in E} [-\log \sum_{e_1 \in E} exp(R^d(e) \cdot R^d(e_1)) + \sum_{i \in N(e)} R^d(i) \cdot R^d(e)] \tag{2}$$

Then we optimize it using stochastic gradient ascent over the model parameters. The detailed steps of edge representation learning algorithm can be summarized in Algorithm 1.

Algorithm 1: Edge Representation Learning Algorithm

Input: Graph(V, E), Dimensions d, Walks per edge r, Walk length l, Context size k

Initialize walks to Empty
for all $iter = 1$ to r **do**
    for all edges $e \in E$ **do**
        Walk = RandomWalk(G, e, l)
        Append walk to walks
        f = StochasticGradientDescent $(k, d, walks)$
        return f
    end for
end for

4 CD-ERL Algorithm

In this section, the details of CD-ERL algorithm are described. There are four steps to complete CD-ERL algorithm.

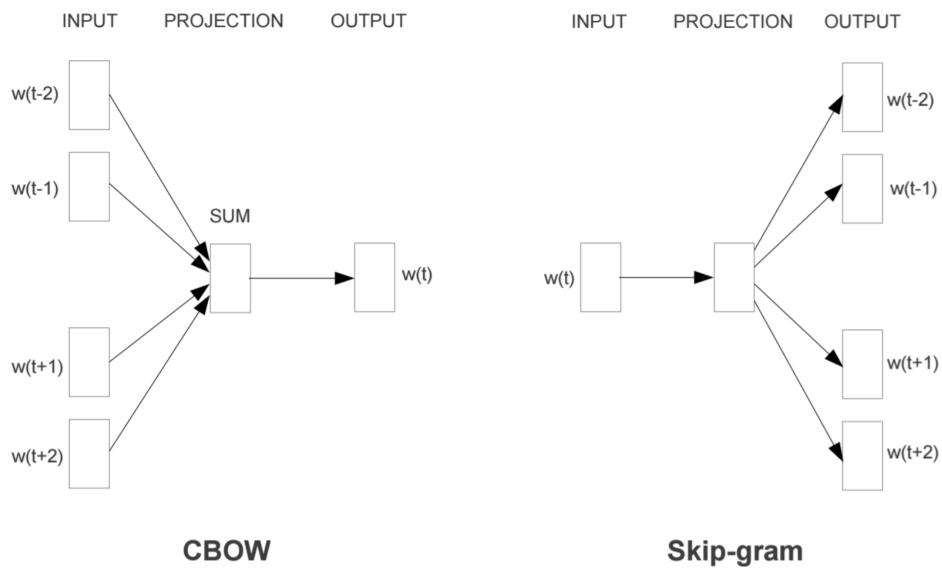


Fig. 3: CBOW and Skip-gram model architectures. [13]

Step 1: **Edge sequences sampling.** Edge sequence in network is non-linear, unlike word sequence in document. It is important how to sample edge sequences. Given a pronounced community structure, we want to achieve this effect that pairs of edges in the same community are much more easily reachable by the sampling strategy than pairs of edges in different communities. In this paper, we choose random walk as the sampling strategy since random walk can well meet our requirements. We collect edge neighbors by random walk, and regard the sampled edge sequences in a network as word sequences in a document.

Step 2: **Edge representations learning.** After getting sampled edge sequences, based on representative learning method for natural language processing, e.g. continuous bag of words [14] and Skip-gram model [13], we transform edges in graph into representative feature matrix R^d using an unsupervised machine learning way. CBOW and Skip-gram model are two widely used methods in natural language processing field. As shown in figure 3, the CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word. At the same time the representation of the inputs are automatically learned. Here we replace the words in the context by sampled edge sequences.

Step 3: **Edges clustering.** In this part, we choose an improved k-means algorithm to complete edge clustering. Given edge feature matrix R^d, the objective function is:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - u_i||_2^2 \tag{3}$$

Our goal is to minimize it. Here u_i is the mean vector of cluster C_i. K-means algorithm[8] adopts greedy strategy and gets the approximate solution by iterative optimization.

The performance of clustering depends on the initial k seeds, so it is crucial how to choose k initial seeds. Here we deal with the seeds using an improved method. Firstly, we choose one center uniformly at random from all data points. Then we compute the distance between data point x and the nearest center that has already been chosen, and choose one new data point at random as a new center, using a weighted probability distribution where a point x is chosen with probability proportional to $D(x)^2$. Last, repeat the above steps until k centers have been chosen. The detailed steps of improved K-means algorithm are summarized in Algorithm 2.

Algorithm 2: Improved K-means Algorithm

Input: Dataset $D = \{ x_1, x_2, ..., x_m \}$, Cluster Number k
Output: Clusters $C = \{ C_1, C_2, ..., C_k \}$

Select one data point u in Dataset D randomly
Repeat
for all $l = 1$ to m **do**
  Compute distance $D(x_l)$ between x_l and the nearest center
  Choose a new center using a probability proportional to $D(x_l)^2$
end for
Until k centers have been chosen
Repeat
Let $C_i = \phi \;\; 1 \leq i \leq k$
for all $j = 1$ to m **do**
  Compute $d_{ji} = ||x_j\text{-}u_i||_2$
  $\lambda_j = argmin_{i \in 1,2,...,k} d_{ij}$
  $C_{\lambda_j} = C_{\lambda_j} \sqcup x_j$
end for
for all $i = 1$ to k **do**
  Compute the new mean vectors : $u_{i1} = \frac{1}{|C_i|} \sum_{x \in C_i} x$
  if $u_{i1} \neq u_i$ **then**
    Update u_i to u_{i1}
  else
    Keep the same u_i
  end if
end for
Until all the mean vectors do not change

Step 4: **Transforming edge communities into node communities.** In graphs, each edge connects two nodes, in other words, one edge corresponds two nodes. In this way, we can transform the edge communities into corresponding node communities. In the example above, the edges connecting A with B and A with C would be placed in different groups, and since they both have A as endpoint, the latter turns out to be an overlapping vertex. Thus nodes belongs to more than one communities can be detected, and we can get both edge communities and node communities.

The complete pseudocode of CD-ERL algorithm is in Algorithm 3.

Algorithm 3: CD-ERL Algorithm

Input: A network $G(V, E)$, edge representation learning parameters, number of communities k

```
1: vectors = ERL(ERL parameters)
2: edge communities = K-means(vectors, k)
3: turn edge communities into node communities
4: return node communities
```

5 Experiments and Results

To better explain the process of the proposed CD-ERL algorithm, we test it on Zachary's karate club dataset [25].

5.1 A running instance

Zachary's karate club dataset is widely used in network analysis field. A social network of a karate club was studied by Wayne W. Zachary for a period of three years from 1970 to 1972. The network captures 34 members of a karate club, documenting 78 pairwise links between members who interacted outside the club. The visualization of this network is shown in Figure 4. In this part, the parameters used in CD-ERL algorithm is set as table 1. We extract 5 walks per edge from the original network. Every walk is a edge sequence, and the length of each edge sequence is 10. We input these extracted walks into the representation learning model and set the feature representation dimensions, then an edge representation will be automatically learned.

The community detection result of CD-ERL algorithm is shown in table 2, it is the detected edge communities. Getting edge communities, we can transform it into node communities based on the correlation of nodes and edges. The final result node communities are shown in table 3. We can see that it can not only detect the overlapping nodes, but also provide the probability that one node belongs to a community.

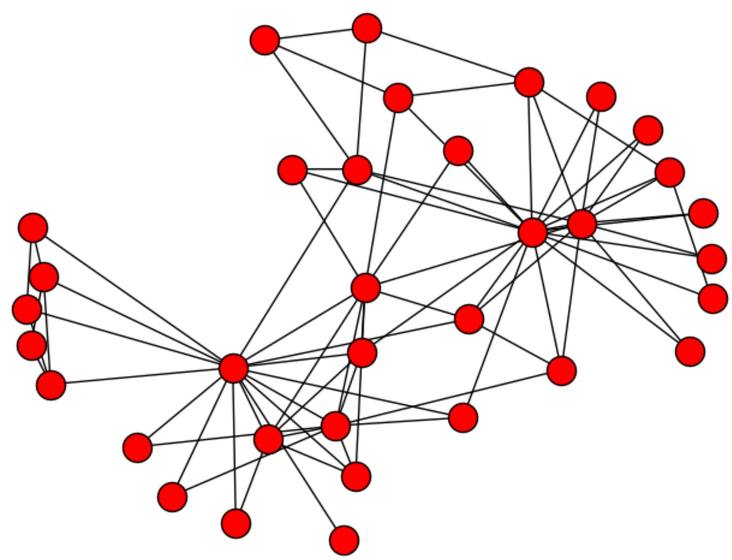


Fig. 4: Zachary's karate club dataset.

Table 1: Experiment Parameters

Parameters	# walks	Walk length	Dimensions	Communities
Zachary's karate club networks	5	10	8	2

To test the performance of proposed CD-ERL algorithm, we do experiments on nine synthetic network benchmarks and two real world networks, and compare it with two traditional community detection methods.

5.2 Evaluation Metrics

NMI In this paper, we use NMI (Normalized Mutual Information) to evaluate the performance of community detection algorithms. The definition of NMI is as follows:

$$\mathrm{NMI} = \frac{2I(X;Y)}{H(X)+H(Y)} \tag{4}$$

If there are N samples in both X and Y, N_i is the number of samples which is equal to i, N_j is the number of samples which is equal to j, N_{ij} is the number

Table 2: Edge communities

	Edge
Community 1	1,3,4,5,6,8,10,11,12,14,15,16,17,20,21,24,26,28,29,31,32,33,34,35,37,39, 40,41,42,43,46,48,49,50,51,52,54,55,56,59,61,63,64,65,67,68,69,73,76
Community 2	0,2,7,9,13,18,19,22,23,25,27,30,36,38,44,45,47,53,57,58,60,62,66,70,71, 72,74,75,77

Table 3: Node communities

Node	**0**	**1**	**2**	**3**	**4**	**... 9**	**10**	**11**	**12**	**...**
Community 1	0.6875	0.4444	0.7000	0.8333	0.6667	**...** 0.5000	0.6667	0	1	...
Community 2	0.3125	0.5556	0.3000	0.1667	0.3333	**...** 0.5000	0.3333	1	0	,,,

of samples which is equal to i in X, equal to j in Y. The calculation formula of NMI[4][22] becomes as follows:

$$\mathrm{NMI} = \frac{-2\sum_{ij} N_{ij} \log \frac{N_{ij} \cdot N}{N_i \cdot N_j}}{\sum_i N_i \cdot \log \frac{N_i}{N} + \sum_j N_j \cdot \log \frac{N_j}{N}} \tag{5}$$

Its lower bound is 0, representing the independence of the result and ground truth, and its upper bound is 1, representing community detection result is the same with ground truth. The closer NMI score is to 1, the better the community detection result is.

V-measure A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster. V-measure[20] is the harmonic mean between homogeneity and completeness.

$$V = \frac{2 \cdot homogeneity \cdot completeness}{homogeneity + completeness} \tag{6}$$

It is symmetric. Its bound is between 0 to 1. 1 stands for perfectly complete community detection. The closer V-measure score is to 1, the better the community detection result is.

Since CD-ERL algorithm can find overlapping communities, in order to perform a better contrast with the traditional non-overlapping community detection algorithms, we made a preprocessing for the results of CD-ERL algorithm. If nodes belongs to more than one communities, we will randomly set this node to one of these communities. In this way, we can contrast the CD-ERL algorithm with traditional non-overlapping community detection methods.

5.3 Baseline Methods

In this paper, we employ two traditional representative community detection algorithms, the LPA (Label Propagation Algorithm) and Louvain algorithm, as the contrast with our method.

Label Propagation Algorithm This algorithm was introduced by Raghavan et al[19]. It proceeds based on the assumption that each node in the network is in the same community with the majority of its neighbors. Every node is initialized a distinct label at the start. In the process of iteration, for a node, count its neighbors' labels and label the node according to the majority of its neighbors' labels. When each node in network has the same label as the majority of its neighbors, the iteration stops. The LPA algorithm is suitable for large scale networks.

Louvain Algorithm The Louvain community detection algorithm[3] optimizes the modularity of a partition of the network by greedy optimization. It optimizes local modularity and generates small scale communities. Then it regards the s-mall communities as nodes to generate a new network. These steps are repeated iteratively until a maximum of modularity is attained and a hierarchy of communities is produced. Relatively speaking, this method is fast and accurate.

5.4 Dataset

Synthetic Benchmark Networks LFR(Lancichinetti-Fortunato-Radicchi) benchmark graph[10] is one of the most frequently-used synthetic network model in community detection filed. This method introduces power-law distributions of degree and community size to the graphs to simulate real world networks. There are six main parameters in this model. N is the number of nodes, k is the average degree, $maxk$ is the maximum degree of nodes, $minc$ is the minimum number of community members, $maxc$ is the maximum number of community members. μ is mix parameter, which is defined as:

$$\mu = \frac{\sum_i k_i^{ext}}{\sum_i k_i^{tot}} \tag{7}$$

Here k_i^{ext} and k_i^{tot} stand for the external degree of node i, i.e. the number of edges connecting it to others that belong to different communities, and the total degree of said node. In this paper, we generate nine networks with different mu. Since the definition of community that in the same communities nodes are linked densely, between different communities the links are sparse, we choose μ from 0.1 to 0.5 for the following experiment. The detail settings of parameters for the synthetic networks are shown in table 4.

Table 4: Parameter Settings of Synthetic Networks

No.	N	k	**maxk**	**minc**	**maxc**	μ
1	10000	8	20	3	1000	0.10
2	10000	8	20	3	1000	0.15
3	10000	8	20	3	1000	0.20
4	10000	8	20	3	1000	0.25
5	10000	8	20	3	1000	0.30
6	10000	8	20	3	1000	0.35
7	10000	8	20	3	1000	0.40
8	10000	8	20	3	1000	0.45
9	10000	8	20	3	1000	0.50

Real World Networks In this paper, we also evaluate our algorithm on two real world networks. These networks with ground truth communities are both from SNAP(Stanford Network Analysis Platform) [12]. We conduct contrast experiments on Amazon and Youtube networks. These two datasets contain not only the graph links, but also the ground truth communities.

We observe that there are some stray nodes with few links in networks which will damage the community detection progress, so we treat them as noisy nodes. Thus, we preprocess these datasets by deleting these stray nodes. After that we get two high-quality networks whose properties are shown in table 5.

Table 5: Properties of Real World Networks

Name	**Type**	**Nodes**	**Edges**	**Average degree**	**Description**
Youtube	Undirected&unweighted	39841	224235	11.2565	Youtube online social network
Amazon	Undirected&unweighted	16716	48739	5.8314	Amazon product network

5.5 Result Analysis

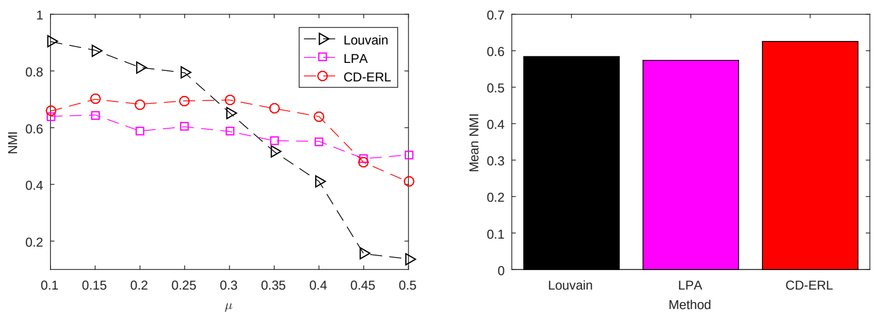
To test the performance, we run CD-ERL algorithm and the contrast methods on both synthetic and real world networks. The parameters used in CD-ERL algorithm is set as table 6.

We generate nine different synthetic networks with different values of μ from 0.1 to 0.5. We repeat experiments for five times on every synthetic networks, and

Table 6: Experiment Parameters

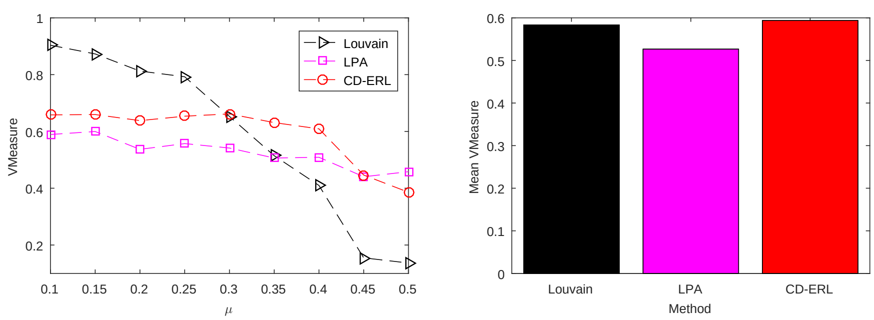
Parameters	# walks	Walk length	Dimensions	Communities
Synthetic networks	20	40	128	1000
Real world networks	20	40	128	5000

regard the average results as the final results. Experiment results on synthetic benchmark networks are shown in Figure 5 and Figure 6.



(a) (b)

Fig. 5: A contrast of NMI using different methods on networks with different μ.



(a) (b)

Fig. 6: A contrast of V-measure using different methods on networks with different μ.

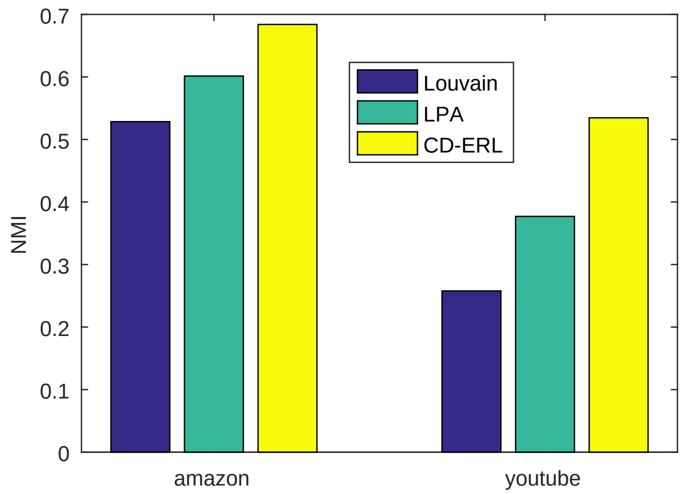


Fig. 7: A contrast of NMI using different methods on different real world networks.

In Figure 5 and Figure 6, horizontal axis represents different mu, while vertical axis represents the value of NMI and V-measure score. We can see that there is a rapid decrease for Louvain algorithm when mu is getting larger. On the contrast, CD-ERL and LPA algorithm is stable with the increase of mu. But the CD-ERL algorithm we proposed in this paper performs better than LPA. For mu from 0.1 to 0.25, Louvain algorithm gets higher value of NMI than the others. This is because when mu is small, networks usually have clear structure, thus Louvain algorithm using modularity optimizing strategy deal with this kind of networks better. But real world networks are always complex and uncertain, Louvain algorithm cannot solve the reality problems stably. CD-ERL and LPA are stable in networks with different structure. But we can see that for the mean NMI and V-measure score, CD-ERL performs best in contrast with other two methods. When dealing with real networks with complex and fuzzy structure, CD-ERL is more suitable.

What's more, we also employ these algorithms on real world networks. Experiment results on real world networks are shown in Figure 7 and Figure 8.

It is obvious in Figure 7 and Figure 8 that the performance of CD-ERL algorithm is much better than LPA and Louvain algorithm. The main idea of Louvain algorithm is to maximize the modularity of networks, it is suitable for networks with obvious community structure. Thus, when the real world network is fuzzy, Louvain algorithm gets bad results. The performance of LPA algorithm is between Louvain algorithm and ours. For LPA algorithm, the label of node in network depends on the labels of its surrounding nodes. When the network is fuzzy, the result becomes bad. CD-ERL algorithm converts the graph links into

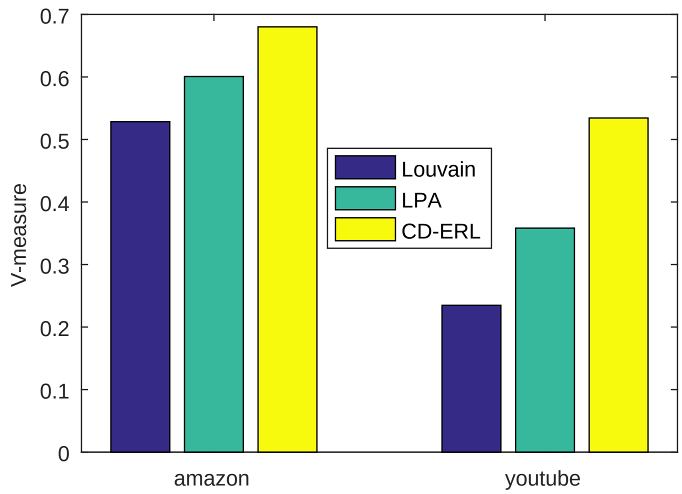


Fig. 8: A contrast of V-measure using different methods on different real world networks.

vectors by edge representation learning, and it uses distance measure instead of modularity in clustering process, this method performs well in real world networks.

5.6 Parameter Sensitivity

In this section, we discuss the parameter sensitivity of proposed CD-ERL algorithm. We do the community detection task on Amazon dataset and show how the NMI score changes when we change the three parameters of CD-ERL algorithm: (1) walk length of random walk (2) number of walks per edge (3) dimensions of the embedding.

As shown in Figure 9, we can see that when feature dimension is set to a small value, the NMI score is higher. With the growth of dimension, the NMI score drops a bit, then keeps a steady state. What's more, we do experiments on Amazon dataset to see how the NMI score changes with different walk length settings. Results in Figure 10 show when walk length is set to 20, the NMI score is low. When we do random walks of walk length larger than 40, the NMI score increases a lot and remains constant. Similarly for the parameter of walks number per edge, Figure 11 shows with a small number of walks per edge, we get a low NMI score. But when the number of walks is larger than 30, the NMI score increases a bit. And with the growth of number of walks per edge, the NMI score stays stable.

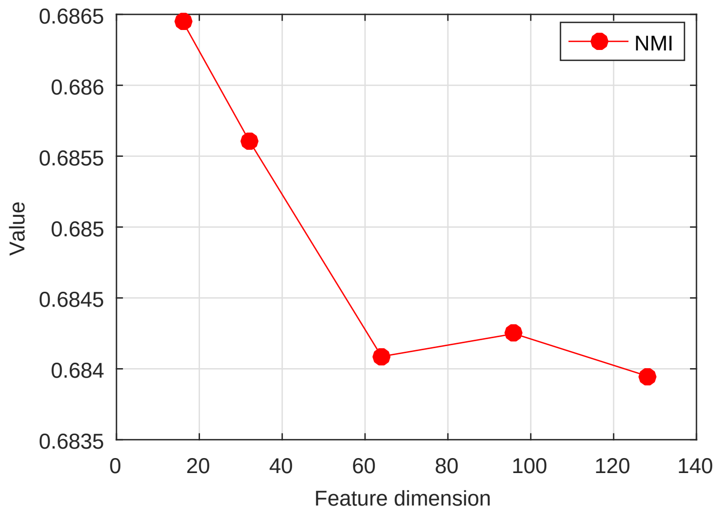


Fig. 9: NMI score on Amazon dataset for various values of dimensions.

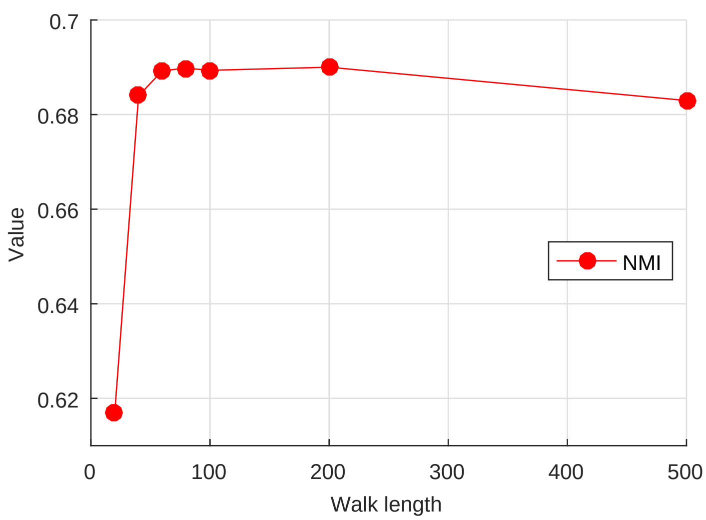


Fig. 10: NMI score on Amazon dataset for various values of walk length.

6 Conclusion

Traditional community detection methods pay more attention to dividing nodes directly, but in this paper we propose a method using traditional clustering algorithm to divide edges into communities by learning edge feature representations.

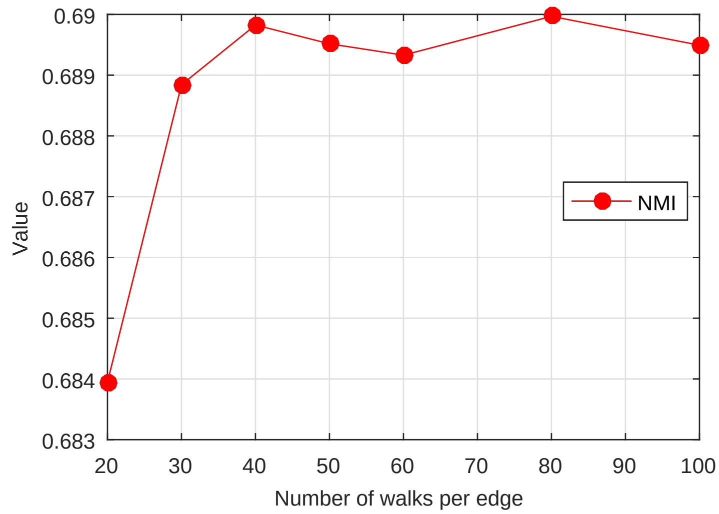


Fig. 11: NMI score on Amazon dataset for various values of number of walks per edge.

Then we turn the edge communities into corresponding node communities to uncover the fuzzy community structure in complex networks. Experiments on synthetic networks and two real world networks show that our algorithm performs well and it is suitable to solve fuzzy networks. CD-ERL algorithm in this paper aims at homogeneous networks, we will study how to extend it to heterogeneous networks next.

Acknowledgments. The work presented in this paper was supported in part by the Special Project for Independent Innovation and Achievement Transformation of Shandong Province (2013ZHZX2C0102, 2014ZZCX03401).

References

1. Adhikari, B., Zhang, Y., Ramakrishnan, N., Prakash, B.A.: Distributed representation of subgraphs. arXiv preprint arXiv:1702.06921 (2017)
2. Bansal, S., Bhowmick, S., Paymal, P.: Fast community detection for dynamic complex networks. In: Complex Networks, pp. 196–207. Springer (2011)
3. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment 2008(10), P10008 (2008)
4. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment 2005(09), P09008 (2005)
5. Fortunato, S.: Community detection in graphs. Physics reports 486(3), 75–174 (2010)

6. Fortunato, S., Hric, D.: Community detection in networks: A user guide. Physics Reports 659, 1–44 (2016)
7. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 855–864. ACM (2016)
8. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern recognition letters 31(8), 651–666 (2010)
9. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. The Bell system technical journal 49(2), 291–307 (1970)
10. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical review E 78(4), 046110 (2008)
11. Leal, T.P., Goncalves, A.C., Vieira, V.d.F., Xavier, C.R.: Decode-differential evolution algorithm for community detection. In: Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on. pp. 4635–4640. IEEE (2013)
12. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data` (Jun 2014)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
15. Newman, M.E.: The structure and function of complex networks. SIAM review 45(2), 167–256 (2003)
16. Newman, M.E.: Finding community structure in networks using the eigenvectors of matrices. Physical review E 74(3), 036104 (2006)
17. Newman, M.E.: Modularity and community structure in networks. Proceedings of the national academy of sciences 103(23), 8577–8582 (2006)
18. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710. ACM (2014)
19. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical review E 76(3), 036106 (2007)
20. Rosenberg, A., Hirschberg, J.: V-measure: A conditional entropy-based external cluster evaluation measure. In: EMNLP-CoNLL. vol. 7, pp. 410–420 (2007)
21. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web. pp. 1067–1077. ACM (2015)
22. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: is a correction for chance necessary? In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 1073–1080. ACM (2009)
23. Wang, S., Tang, J., Aggarwal, C., Chang, Y., Liu, H.: Signed network embedding in social media. SDM (2017)
24. Yang, L., Cao, X., Jin, D., Wang, X., Meng, D.: A unified semi-supervised community detection framework using latent space graph regularization. IEEE transactions on cybernetics 45(11), 2585–2598 (2015)
25. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of anthropological research 33(4), 452–473 (1977)