TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 01/10 pp447–457
DOI: 10.26599/TST.2019.9010055
Volume 25, Number 4, August 2020

Complex Network Classification with Convolutional Neural Network

Ruyue Xin, Jiang Zhang*, and Yitong Shao

Abstract: Classifying large-scale networks into several categories and distinguishing them according to their fine structures is of great importance to several real-life applications. However, most studies on complex networks focus on the properties of a single network and seldom on classification, clustering, and comparison between different networks, in which the network is treated as a whole. Conventional methods can hardly be applied on networks directly due to the non-Euclidean properties of data. In this paper, we propose a novel framework of Complex Network Classifier (CNC) by integrating network embedding and convolutional neural network to tackle the problem of network classification. By training the classifier on synthetic complex network data, we show CNC can not only classify networks with high accuracy and robustness but can also extract the features of the networks automatically. We also compare our CNC with baseline methods on benchmark datasets, which shows that our method performs well on large-scale networks.

Key words: complex network; network classification; DeepWalk; Convolutional Neural Network (CNN)

1 Introduction

A complex network is the highly simplified model of a complex system, and it has been widely used in many fields, such as sociology, economics, and biology[1]. Given that complex networks can describe the relationship between events, an increasing amount of research is using complex networks to model problems. For example, we can use a network to model compounds in chemical research, in which nodes and edges represent molecules and chemical bonds between molecules. The compound network can be used to identify substances with the same pattern structure as the toxic compounds. Moreover, nowadays, more and more social data constitute large-scale social networks, in which nodes and links represent individuals and relationships. The analysis of social network can be used to identify key people in society or reveal the social circles of people. Therefore, studying complex networks is crucial.

Most studies on complex networks focus on the properties of a single complex network[2], such as classification and clustering of nodes and link prediction, while paying little attention to comparisons, classifications, and clustering between different complex networks. However, complex networks classification is necessary and important in the study of complex networks. For example, the social network behind an online community impacts the development of the community because social ties between users can be treated as the backbone of the online community[3]. Thereafter, online community can be diagnosed by comparing and distinguishing their connected modes, and the development of online communities can be predicted. As another example, we consider the product flows on the international trade network. We know that correct classification of products not only helps us understand the characteristics of products but also helps trade countries better count the trade volume of products. However, classifying and labeling each

• Ruyue Xin and Jiang Zhang are with the School of Systems Science, Beijing Normal University, Beijing 100875, China. E-mail: xin-yue-2009@163.com; zhangjiang@bnu.edu.cn.

• Yitong Shao is with the School of Mathematical Sciences, Beijing Normal University, Beijing 100875, China. E-mail: 201511130117@mail.bnu.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2019-01-18; revised: 2019-07-12; accepted: 2019-09-09

@ The author(s) 2020. The articles published in this open access journal are distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/).

exchanged product in international trade is tedious and difficult. Conventional method classifies these products according to the attributes of the product manually (https://unstats.un.org/unsd/trade/sitcrev4.htm), which is subjective. However, if a trade network classifier is built, we can classify a new product exclusively according to its network structure, because previous studies point out that different products have completely different structures of international trade networks.

In addition, complex network classification problems can be applied in many practical areas, such as predicting a country's economic development based on industrial networks or predicting a company's performance based on its interactive structure, which all can be converted to a complex network classification task.

The complex network classification task refers to the use of an established network classification model to continuously learn the training data, which is a labeled network dataset, to find the connection between the label and the structure of the network and finally use the learned model to classify the test data, which is a labeled unknown network dataset. Thus, the complex network classification problem is supervised learning similar to traditional classification algorithms. However, because complex networks mainly contain unstructured data, such as network structure, traditional classification algorithms cannot be directly applied. Therefore, new complex network classification algorithms need to be found.

At present, deep learning technology has achieved state-of-the-art results in the processing of Euclidean data. For example, Convolutional Neural Network (CNN)[4] can process image data, and Recurrent Neural Network (RNN)[5] can be used in natural language processing. However, deep learning technology is still under development for graph structure data, such as social networks, international trade networks, and protein structure data.

As for complex network classification problem, some related research mainly studied graph structure data in the past. For example, kernel methods were proposed to calculate the similarity between two graphs[6]. However, the methods can hardly be applied to large-scale and complex networks due to the expensive computational complexity of these graph classification methods.

Network representation learning, which is developed recently, is an important way to study graph structure data. Earlier works, such as local linear embedding[7] and IsoMAP[8], first constructed graphs based on feature vectors. In the past decades, some shallow models such as DeepWalk[9], node2Vec[10], and LINE[11] were proposed, which can embed nodes into high-dimensional space and empirically perform well. However, these methods can only be applied on the tasks (classification, community detection, and link prediction) on nodes but not the whole networks. Some models, such as GNNs[12], GGSNN[13], and GCN[14], use deep learning techniques to deal with network data and learn representations of networks. Nevertheless, most methods also focus on the tasks at the node level but not the graph level. Another limitation of these techniques is the requirement for a fixed network structure. In this paper, we proposed a new method called Complex Network Classifier (CNC) to address the complex network classification problem by combining network embedding and CNN. We first embed a network into high-dimensional space through the DeepWalk algorithm, which preserves the local structures of the network and converts it into a two-dimensional image. Then, we input the image into a CNN for classification. Our model framework has the merits of small size, small computational complexity, scalability to different network sizes, and automatic feature extraction.

The rest of this paper is organized as follows. Section 2 introduces the related research. Section 3 presents the model framework and experiments data. Section 4 shows the experiments and results. Section 5 gives the discussion and conclusion.

2 Related Work

2.1 Complex network

Complex network focuses on the structure of individuals' interrelation in systems and is a way to understand the nature and function of complex systems. Studies of complex networks started from regular networks, such as Euclidean grid or nearest-neighbor network in the two-dimensional plane[15]. In 1959, Gilbert[16] proposed random network theory. In 1998, Watts and Strogatz[17] and Barabási and Albert[18] proposed small-world and scale-free network models, respectively, which depict real-life networks better. Researchers have summarized the classic complex network model, which includes regular networks, random networks, small-world networks, and scale-free networks, and proposed network properties, such as average path length, aggregation coefficient, and degree distribution. Recent studies mainly focus on network reconstruction and network synchronization, and few studies focus on the classification of complex networks.

2.2 Network classification

Classification of network data has important applications, such as protein-protein interaction, predicting the functionality of chemical compounds, diagnosing communities, and classifying product trading networks. In the network classification problem, we are given a set of networks with labels, and the goal is to predict the label of a new set of unlabeled networks. The kernel methods developed in previous research are based on the comparison of two networks and similarity calculation. The most common graph kernels are random walk kernels[19], shortest-path kernels[20], graphlet kernels[21], and Weisfeiler-Lehman graph kernels[22]. However, the main problem of graph kernels is that they can hardly be used on large-scale and complex networks because of the expensive calculation complexity.

2.3 Deep learning on graph structure data

CNN is the most successful model in the field of image processing. It has achieved good results in image classification[4], recognition[23], semantic segmentation[24], and machine translation[25], and can independently learn and extract features of images. However, it can be applied only on regular data, such as images with a fixed size. As for graph structure data, researchers have recently been searching for effective and efficient deep learning methods with deep learning methods. For example, to apply the convolutional operation on graphs, Ref. [26] proposed to perform the convolution operation on the Fourier domain by computing the graph decomposition of the Laplacian matrix. Furthermore, Ref. [27] introduced a parameterization of the spectral filters. Reference [14] proposed an approximation of the spectral filter by Chebyshev expansion of the graph Laplacian. Reference [28] simplified the previous method by restricting the filters to operate in a one-step neighborhood around each node. However, among all the aforementioned spectral approaches, the learned filters based on the Laplacian eigenbasis are dependent on the graph structure. Thus, a model trained on a specific structure cannot be directly applied to a graph with a different structure. We know that a complex network classification problem often includes many samples and each sample has one specific network structure, so we cannot directly use GCN to classify networks.

2.4 Network representation learning

Representation learning has been an important topic in machine learning for a long time, and many works aim at learning representations for samples. Recent advances in deep neural networks have indicated their powerful representation abilities and that they can generate useful representations for many types of data. Network representation learning is an important way to preserve structure and extract features of a network through network embedding, which maps nodes into a high-dimensional vector space based on graph structure. The vector representations of network nodes can be used for classification and clustering tasks. Some shallow models were previously proposed for network representation learning. DeepWalk[9] combined random walk and skip-gram to learn network representations. LINE[11] designed two loss functions attempting to capture the local and global network structure. Node2Vec[10] improved DeepWalk and proposed a two-order random walk to balance the Depth First Search (DFS) and Breath First Search (BFS) search. Reference [29] proposed an approach based on the open-flow network model to reveal the underlying flow structure and its hidden metric space of different random walk strategies on networks.

The most important contribution of network representation learning is that it can extract network features that provide a way to process network data. Thus, we consider using the features extracted by the embedding methods to solve the network classification problem. We recognize DeepWalk as a classic and simple model that can represent the network structure and has high efficiency when dealing with large-scale networks. Moreover, the random walk process in DeepWalk, which obtains the sequences of networks, is adaptable to different networks. For example, we can easily change the random walk mechanism for the international trade network, which is directed and weighted. Therefore, we combine the network representation learning and deep learning method to develop our model, which can perform well in the complex network classification task.

3 Methods of Network Classification

3.1 Model

Our strategy to classify complex networks is to convert networks into images and use the standard CNN model to perform the network classification task. Given the development of network representation techniques, many algorithms can be used to embed the network into a high-dimensional Euclidean space. We select DeepWalk algorithm[9], which was proposed by Bryan Perozzi et al. to obtain the network representation. The algorithm generates numeric node sequences by performing large-scale random walks on the network. Afterwards, the sequences are fed into the skip-gram and negative sampling algorithms to obtain the Euclidean coordinate representation of each node. To increase the number of training samples, we can perform data augmentation by performing the DeepWalk algorithm on a single network several times to obtain more sets of node representations. Obviously, high-dimensional space representation is hard to process. Thus, we use the Principal Components Analysis (PCA) algorithm to reduce the dimension of node representations into two-dimensional space. PCA can simplify information, remove redundant information and noise, and retain the principle components of data[30]. For the high-dimensional space representation of network, PCA can not only reduce the dimension but also retain the main information of the network structure. For example, the karate network is shown in Fig. 1a, and its two-dimensional representation after the DeepWalk embedding and PCA reducing dimension is presented in Fig. 1b. Figure 1b shows that nodes of the same color are clustered, thereby indicating that the network structure information is actually well preserved after embedding and reducing the dimension, thus being suitable for the network classification task. However, the set of nodes is a point cloud that is still irregular and cannot be processed by CNN. Thus, we rasterize the two-dimensional representation into an image. We divide all the areas covered by the 2-dimensional scatter plot into a square area with 48 × 48 grids, and then count the number of nodes in each grid as the pixel grayscale. Afterwards, a standard gray scale image is obtained. We actually change the size of the grid based on the network size during our experiments, reducing the size for small networks. This method can also be applied on directed and weighted networks, such as international trade networks. By adjusting the probabilities according to the weight and direction of each edge for a random walk on a network, we can obtain an embedding image.

The final step is to feed the networks' images into a CNN classifier to complete the classification task. Our CNN architecture includes two convolutional layers (one convolutional operation and one max-pooling operation) and one fully connected layer and one output layer. The whole architecture of our model

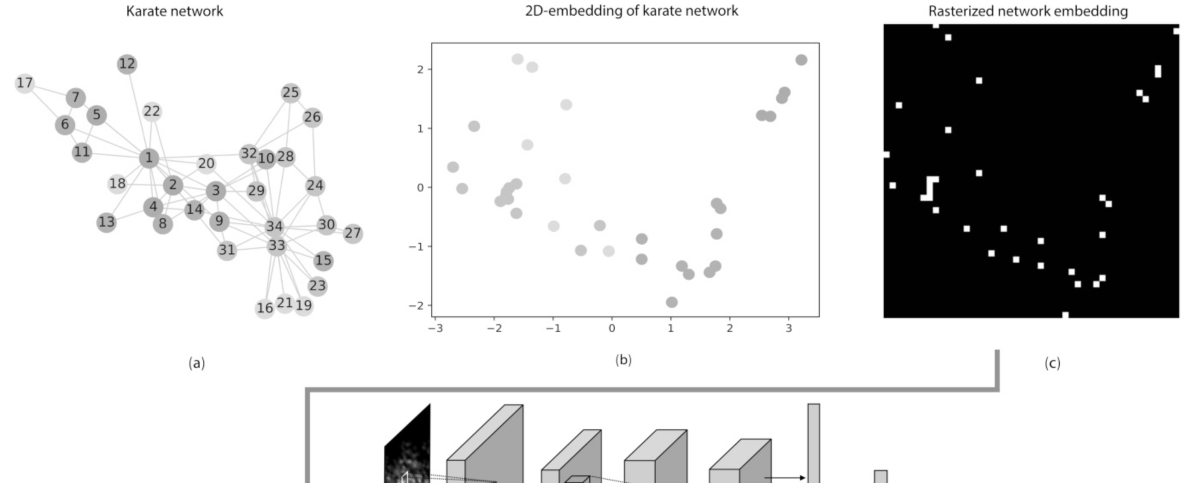


Fig. 1 Pipeline of the CNC algorithm.

is shown in Fig. 1. Figure 1a is the original input network. Figure 1b is the embedding of the network with DeepWalk algorithm. In DeepWalk algorithm, to obtain enough corpus, we set the number of walks to 10 000 times and the sequence length to 10. We then embed the network into a 20-dimensional space and project it on two-dimensional space by using the PCA algorithm. Figure 1c is the rasterized image from the 2D-embedding representation of the network. Figure 1d is the CNN architecture of the CNC algorithm, which includes one input image, two convolutional pooling layers, one fully connected layer, and one output layer. The sizes of the convolutional filters and of the pooling operation are 5 × 5 and 2 × 2, respectively. The first layer has three convolutional filters, the second layer has five convolutional filters, and the fully connected layer has 50 units. In all complex network classification experiments, we set the learning rate to 0.01 and the size of a mini-batch is 100. The CNN architecture is selected to minimize the computational complexity and retain the classification accuracy.

3.2 Experimental data

A large amount of experimental data is needed to train and test the classifier. Thus, we use both synthetic networks generated by network models and empirical networks to test our model.

3.2.1 Synthetic data

The synthetic networks are generated by well-known Barabási-Albert model (BA) and Watts-Strogatz model (WS). According to the evolutionary mechanism of the BA model, which iteratively adds $m = 4$ nodes and edges at each time, the added nodes will preferentially link to the existing nodes with higher degrees until $n = 1000$ nodes are generated, and the average degree $\langle E \rangle$ of the generated network is about 8, which is close to the degree of real networks[31]. We then use the WS model ($n = 1000$, the number of neighbors of each node $k = 8$, and the probability of reconnecting edges $p = 0.1$) to generate a large number of small-world networks with the same average degrees as in the BA model. We then mix the generated 5600 BA networks and WS networks. Then, we separate the set of networks into a training set (with 8000 networks), a validation set (with 2000 networks), and a test set (with 1200 networks).

3.2.2 Empirical data

Product-specific international trade networks are adopted as the empirical data to test our classifier. The dataset is provided by the National Bureau of Economic Research (http://cid.econ.ucdavis.edu/nberus.html) and covers the trade volume between countries of more than 800 different kinds of products that are all encoded by SITC4 digits, a hierarchical classification system for different products from 1962 to 2000. The international trade network is a weighted directed network, in which the weighted directed edges represent the volumes of trading flows between two countries. Thus, the random walk in the DeepWalk algorithm should be based on the weights and directions of edges. We train the CNC to distinguish food products and chemical products. Each product class contains about 10 000 networks obtained by the products and the product combinations within the category.

In addition, to test the efficiency of our model, we applied our framework to benchmark datasets from bioinformatics to social networks (see Table 1 for summary statistics of these datasets).

The bioinformatics dataset NCI1[32], made publicly available by the National Cancer Institute, is a subset dataset of chemical compounds screened for their ability to suppress or inhibit the growth of a panel of human tumor cell lines.

Social network datasets include scientific collaboration dataset and Reddit datasets (Reddit is a popular content-aggregation, website: http://reddit.com). The scientific collaboration dataset COLLAB is derived from three public collaboration datasets[33], which represents the three different research fields. The networks of different researchers were generated from each field, and each network was labeled as the field of the researcher. The task is to determine to which field the collaboration network of a researcher belongs. REDDIT-BINARY (RE_B) is a dataset where each network corresponds to an online discussion thread where nodes correspond to users, and an edge exists between two nodes if at least one of them responded to another's comment. Top submissions

Table 1 Properties of the empirical data.

	Size	Enriched size	Number of average nodes	Number of average edges	Number of classes
NCI1	4110	12 330	29.80	32.30	2
COLLAB	5000	15 000	74.49	2457.78	3
RE_B	2000	6000	429.61	497.75	2
RE_5K	4999	14997	508.50	594.87	5
RE_12K	11 929	35 787	391.40	456.89	12

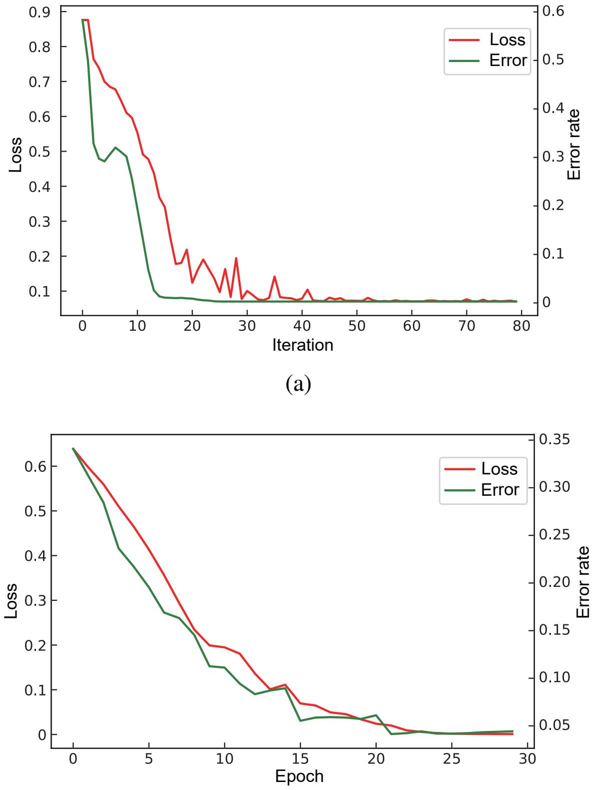
from four popular subreddits were chosen and divided into question/answer-based subreddits and discussion-based subreddits. The task is to identify whether a given network belongs to a question/answer-based community or a discussion-based community. REDDIT-MULTI-5K (RE_5K) is a dataset from the five different subreddits, and we simply label each graph with their correspondent subreddit. REDDIT-MULTI-12K (RE_12K) is a larger variant of RE_5K, consisting of 12 different subreddits. The task in the two datasets is to predict to which subreddit a given discussion network belongs.

4 Experiments and Results

We conduct a large number of network classification experiments, and the results are presented in this section. On the synthetic networks, we not only show the classification results but also present how the CNC can extract the features of networks and the robustness of the classifier on network sizes. On the empirical networks, we show the results that our CNC apply on the trade flow networks, which are directed weighted networks. To compare our method and other existing methods on graph classification, we adopt the empirical networks listed in Table 1.

4.1 Classification on synthetic networks

4.1.1 BA and WS network classification

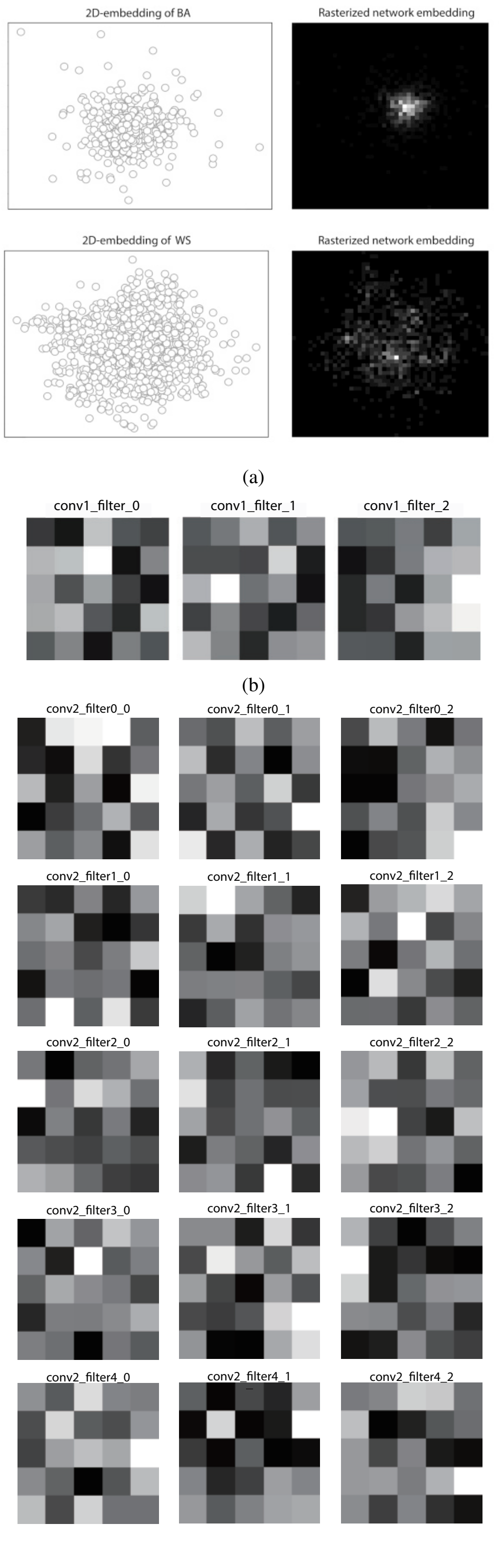


(a)

(b)

Fig. 2 Loss and validation error rate of the classification task on (a) BA vs. WS models and (b) food vs. chemical products.

The first task is to apply CNC to distinguish BA network and WS network. Although we know that the BA network is a scale-free network and WS network is a small-world network with a high clustering coefficient, the machine does not know this. Thus, this series of experiments shows the possibility that the CNC network can extract the key features to distinguish the two networks. We generate 5600 BA networks with $n = 1000, m = 4$ and 5600 WS networks with the same size ($n = 1000$, $k = 8$) and $p = 0.1$, respectively. We combine these networks to form the dataset, which is further randomly separated into training set (with 8000 networks), validation set (with 2000 networks), and test set (with 1200 networks). Figure 2 shows the decay of the loss on the training set and the error rate of the validation set. Finally, we obtain 0.1% of the average error rate on the test set. Thus, we can say the model can distinguish the BA network and the WS network accurately. To understand what has been learned by our CNC model, we can visualize the feature maps extracted from the network representations by the filters of the CNN, which are visualized in Fig. 3. Figure 3a shows 2D representations and rasterized images of a BA network (upper) and a WS network (bottom), Fig. 3b is the visualization of the three filters of the first convolutional layer, and Fig. 3c is the visualization of the five filters (size of $5 \times 5 \times 3$) of the second convolutional layer. However, reading meaningful information is difficult because the network structure cannot correspond to the images. To understand what the filters do, we need to combine the network structure and the feature map. Therefore, we try to map the highlighted areas in the feature maps of each filter on the node set of the network. That is, we wonder which parts of the networks and what kind of local structures are activated by the first convolutional layer filters. We compare the activation modes for the two model networks as input, and the results are shown in Fig. 4. By observing and comparing these figures, we find that the convolutional filters of the first layer have learned to extract the features of the network in different parts. As shown in Fig. 4, Filter 0 extracts the local clusters with a medium density of nodes and connections, Filter 1 tries to extract the local clusters



(a)

(b)

(c)

Fig. 3 Feature maps visualization.

with sparse connections, and Filter 2 tries to extract the local clusters with dense nodes and connections. By comparing the BA and WS model networks, we can observe that the locations and the patterns of the highlighted areas are different. The local areas with dense nodes and connections (Filter 0) locate at the central area of the network representation for both BA network and WS network. The local structures with sparse nodes and connections locate at the peripheral area, which is close to the edges of the image for the WS network but is in the central area for the BA network. This combination of the activation modes on feature maps can help the higher-level filters and fully connected layer distinguish the two kinds of networks.

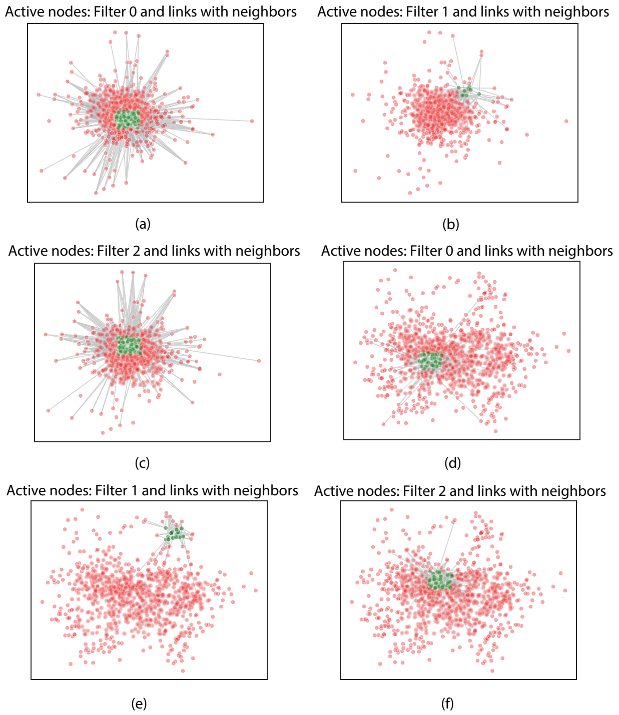

Active nodes: Filter 0 and links with neighbors
Active nodes: Filter 1 and links with neighbors
(a)
(b)
Active nodes: Filter 2 and links with neighbors
Active nodes: Filter 0 and links with neighbors
(c)
(d)
Active nodes: Filter 1 and links with neighbors
Active nodes: Filter 2 and links with neighbors
(e)
(f)

Fig. 4 Active nodes by the three kernels in the first layer, (a)–(c) BA network and (d)–(f) WS network.

4.1.2 Small-world network classification

One may think that distinguishing the BA and WS networks is trivial because they are two different models. Our second experiment will consider whether the classifier can distinguish networks generated by different parameters of the same model, which is harder than the previous task.

To verify the discriminant ability of the model in this task, we use the WS model to generate a large number of experimental networks by changing the value of edge reconnection probability p from 0 to 1 at a 0.1 step, and then we mix the networks with two discriminant p values, e.g., $p = 0.1$ and $p = 0.6$. Then, we train the CNC for networks and test their discriminant ability on the test sets.

We systematically conduct this experiment for any combination of the networks with each two probabilities, and the results are shown in Fig. 5. The

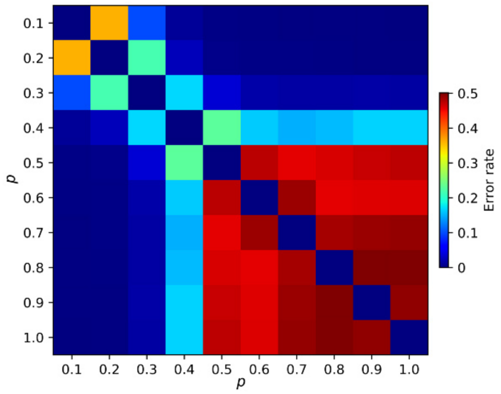
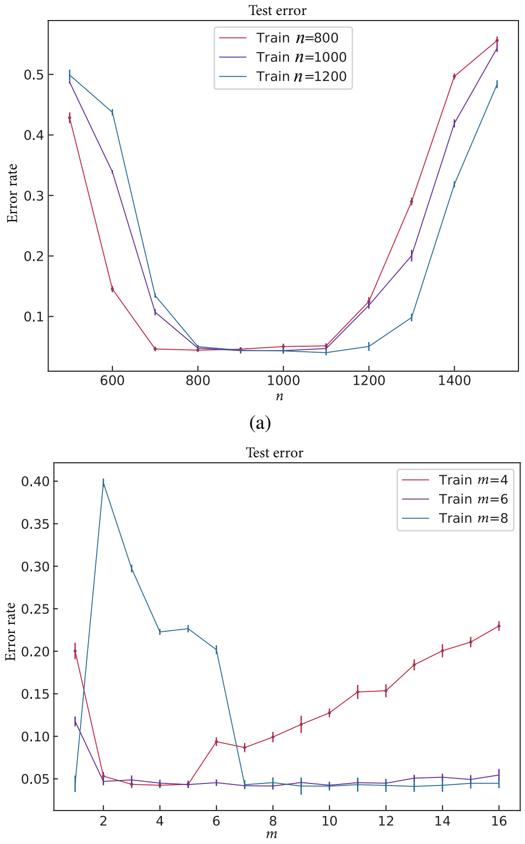


Fig. 5 Classification results of each two small-world networks with different p values.



(a)

(b)

Fig. 6 Dependence of the error rates on (a) the number of nodes and (b) the number of edges in the robustness experiments.

networks generated by p values are less than 0.3 and p values that are greater than or equal to 0.4 are easier to distinguish. Interestingly, the error rate changed suddenly at $p = 0.4$. The classifier cannot distinguish the two networks with $p > 0.4$. This phenomenon may be due to the phase transition from the small-world network to random network because the WS networks with $p > 0.5$ may be treated as random networks.

4.2 Robustness on network sizes

Our model has good classification performances on both synthetic and empirical data. Next, we want to test the robustness of the classification on different sizes (numbers of nodes and edges). All the experiments performed in classification experiments contain the model networks with identical numbers of nodes and edges. Nonetheless, a good classifier should extract the features that are independent in size. Therefore, we examine the robustness of the classifier on various network sizes that are different from those of the training sets. In these experiments, we first apply the trained classifier for BA and WS networks with $n = 1000$ nodes and average degree $\langle E \rangle = 8$, on new networks with different numbers of nodes and edges. We generate 600 mixed networks with parameters m $[1, 2, 3, \ldots, 16]$ for the BA model and k $[2, 4, 6, \ldots, 32]$ for the WS model as the test set, such that their average degrees are similar. We systematically compare how the numbers of nodes and edges on the test sets influence the error rates, as shown in Fig. 6, in Fig. 6a, on the test set, we set n (number of nodes) = $[500, 600, 700, \ldots, 1500]$, and we also retrain networks for $n = 800$ and $n = 1200$, and test them on networks with different n. In Fig. 6b, on the test set, we set m (average number of edges) $= [1, 2, 3, \ldots, 16]$, and we also retrain networks for $m = 6$ and $m = 8$ and test them on different networks with the different m. At first, we observe that the error rates are almost independent on small fluctuations of the number of nodes. However, the error rates increase as size differences increase in the test data. This finding indicates that our classifiers are robust on the size of the networks.

Nevertheless, sudden changes occur in the variants on the number of edges, which indicates that the number of edges has larger impacts on the network structure. We observe that a sudden drop in error rates occurs with an increase of m for the test set when $m = 8$ for the training set. We observed the increased network embedding and determined that this sudden change is due to the emergence of multi-centers on the representation space for the BA model. Therefore, the number of links can change the overall structure in the scale-free network, thereby causing our classifier to fail. Another interesting phenomenon is that the error rates can remain small when the number of edges increases when m in the training set is set to 8. Therefore, the classifier training on the dense networks is more robust on the variance on edge densities.

4.3 Classification on trade flow networks

We want to verify the effectiveness of the model on empirical networks. We conduct a classification task on international trade networks with the dataset obtained from the National Bureau of Economic Research (http://cid.econ.ucdavis.edu/nberus.html). These data cover the trade volume and direction information between countries of more than 800 different kinds of products that are all encoded by SITC4 digits from 1962 to 2000. We select food and chemical products as two labels for this experiment, and their SITC4 codes start with 0 and 5, respectively. For example, 0371 is for prepared or preserved fish and 5146 is for oxygen-function amino-compounds. Figure 7 shows the two-dimensional representations of the 10 products for the two categories. After preprocessing, the total number of the food trade networks is 10 705 (including products and product combinations with SITC4 digits starting with 0) and the number for the chemical trade networks is 10 016 (including products and product combinations with SITC4 digits starting with 5). Then, we divide them into training set, validation set, and test set according to the ratio of 9:1:1. During the training, we adjust the network parameters to 15 convolutional filters in the first layer, 30 convolutional filters in the second layer, and 300 units in the fully connected layer. Figure 2b shows that the classification error rate can be cut down to 5%.

4.4 Comparison with other methods

We compare our model CNC with baseline methods, such as graph kernel methods and deep learning methods. Kernel methods mainly compute the similarity between two graphs. We chose Graphlet Kernel[21] (GK) and Weisfeiler-Lehman kernel[22] (WL), which are two state-of-the-art graph kernels. As for deep learning methods, we chose deep graph kernels[6], which achieve significant improvements in classification accuracy over state-of-the-art graph kernels in some datasets. We also chose PATCHY-SAN (PSCN, $k = 10$)[34], which is the best performing graph CNN model. We applied our model to benchmark datasets and compared the classification accuracy of our model against the baseline methods (see Table 2). From the table, we can see that our framework performs well on Reddit datasets, which are all large-scale networks that have hundreds of nodes and edges. This result implies that our model can learn large-scale networks well because it can extract more meaningful information for these networks through DeepWalk. However, our model cannot perform well on the NCI1 and COLLAB datasets mainly because those networks are too small to produce density information on rasterized images. We also tried using t-SNE methods to reduce the dimension of node representations. But the effect is worse than that of our original PCA method because the nodes become more clustered

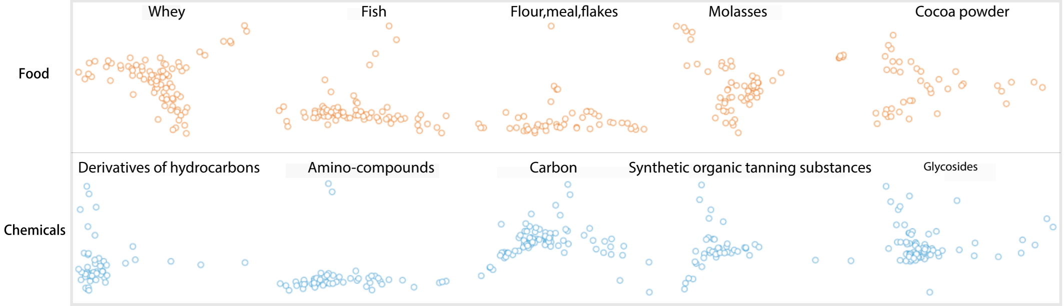


Fig. 7 Network representations of 10 selected products in food (upper) and chemicals (bottom).

Table 2 Comparison of classification accuracy (± standard deviation).

(%)

Method	Dataset				
	NCI1	COLLAB	RE_B	RE_5K	RE_12K
GK	62.28±0.29	72.84 ± 0.28	77.34±0.18	41.01±0.17	31.82±0.08
WL	**80.22±0.51**	**77.82±1.45**	78.52±2.01	50.77±2.02	34.57±1.32
Deep GK	62.48±0.25	73.09±0.25	78.04±0.39	41.27±0.18	32.22±0.10
PSCK, $k = 10$	70.00±1.98	72.60±2.15	86.30±1.58	49.10±0.70	41.32±0.42
CNC_tSNE	63.18±3.35	63.46±1.59	80.17±2.66	46.15±1.55	36.53±0.97
CNC	63.11±0.56	67.79±2.34	**86.72±1.55**	**51.35±3.02**	**41.44±1.64**

after the t-SNE algorithm is applied, thereby leaving less local information when we map them into image representation. Thus, CNN_tSNE has difficulty to learn the effective features.

5 Conclusion and Discussion

In this paper, we propose a model that mainly incorporates DeepWalk and CNN to solve the network classification problem. With DeepWalk, we obtain an image for each network, and then we use CNN to complete the classification task. Our method is independent on the number of network samples, which is a major limitation for the spectral methods on graph classification. We validate our model through experiments with synthetic data and empirical data, which show that our model performs well in classification tasks. To further understand the network features extracted by our model, we visualize the filters in CNN and observe that CNN can capture the differences between WS and BA networks. Furthermore, we test the robustness of our model by setting different sizes for training and testing. We also compare our model with baseline methods, and the result shows that our model performs well on large-scale networks. The biggest advantage of our model is that it can deal with networks with different structures and sizes. In addition, our model has a small architecture and low computational complexity. Several potential improvements and extensions to our model could be addressed in future work. For example, we can develop more methods to deal with the network features in high-dimensional space. We also think that our model can be applied to more classification and forecasting tasks in various fields. Finally, we believe that extending our model to more graph structure data would allow us to address a larger variety of problems.

Acknowledgment

The work was supported by the National Natural Science Foundation of China (No. 61673070) and Beijing Normal University Interdisciplinary Project. The authors would like to thank the referees for their valuable comments and helpful suggestions.

References

[1] S. N. Dorogovtsev and J. F. F. Mendes, Evolution of networks, *Advances in Physics*, vol. 51, no. 4, pp. 1079–1187, 2002.

[2] R. Albert and A. L. Barabási, Statistical mechanics of complex networks, *Reviews of Modern Physics*, vol. 74, no. 1, p. 47, 2002.

[3] R. A. Hanneman and M. Riddle, *Introduction to Social Network Methods*. Riverside, CA, USA: University of California, 2005.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in *Proc. of Advances in Neural Information Processing Systems*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[5] A. Graves, A. Mohamed, and G. Hinton, Speech recognition with deep recurrent neural networks, in *Proc. of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada, 2013, pp. 6645–6649.

[6] P. Yanardag and S. V. N. Vishwanathan, Deep graph kernels, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Sydney, Australia, 2015, pp. 1365–1374.

[7] S. T. Roweis and L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[8] J. B. Tenenbaum, V. De Silva, and J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[9] B. Perozzi, R. Al-Rfou, and S. Skiena, Deepwalk: Online learning of social representations, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2014, pp. 701–710.

[10] A. Grover and J. Leskovec, node2vec: Scalable feature learning for networks, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016, pp. 855–864.

[11] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q Mei, Line: Large-scale information network embedding, in *Proceedings of the 24th International Conference on World Wide Web*, Florence, Italy, 2015, pp. 1067–1077.

[12] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, The graph neural network model, *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[13] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, Gated graph sequence neural networks, arXiv preprint arXiv: 1511.05493, 2015.

[14] M. Defferrard, X. Bresson, and P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in *Proc. of Advances in Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 3844–3852.

[15] S. H. Strogatz, Exploring complex networks, *Nature*, vol. 410, no. 6825, p. 268, 2001.

[16] E. N. Gilbert, Random graphs, *The Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1141–1144, 1959.

[17] D. J. Watts and S. H. Strogatz, Collective dynamics of small-worldnetworks, *Nature*, vol. 393, no. 6684, p. 440, 1998.

[18] A. L. Barabási and R. Albert, Emergence of scaling in random networks, *Science*, vol. 286, no. 5439, pp. 509–

512, 1999.

[19] H. Kashima, K. Tsuda, and A. Inokuchi, Marginalized kernels between labeled graphs, in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, Atlanta, GA, USA, 2003, pp. 321–328.

[20] K. M. Borgwardt and H. P. Kriegel, Shortest-path kernels on graphs, in *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05)*, Washington, DC, USA, 2005, p. 8.

[21] N. Shervashidze, S. V. N. Vishwanathan, T. H. Petri, K. Mehlhorn, and K. M. Borgwardt, Efficient graphlet kernels for large graph comparison, in *Proc. of Artificial Intelligence and Statistics*, Clearwater Beach, FL, USA, 2009, pp. 488–495.

[22] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, Weisfeiler-Lehman graph kernels, *Journal of Machine Learning Research*, vol. 12, no. Sep, pp. 2539–2561, 2011.

[23] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv: 1409.1556, 2014.

[24] J. Long, E. Shelhamer, and T. Darrell, Fully convolutional networks for semantic segmentation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 2015, pp. 3431–3440.

[25] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, A convolutional neural network for modelling sentences, arXiv preprint arXiv: 1404.2188, 2014.

[26] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, arXiv preprint arXiv: 1312.6203, 2013.

[27] M. Henaff, J. Bruna, and Y. LeCun, Deep convolutional networks on graph-structured data, arXiv preprint arXiv: 1506.05163, 2015.

[28] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint, arXiv: 1609.02907, 2016.

[29] W. Gu, L. Gong, X. Lou, and J. Zhang, The hidden flow structure and metric space of network embedding algorithms based on random walks, *Scientific Reports*, vol. 7, no. 1, p. 13114, 2017.

[30] L. I. Smith, A tutorial on principal components analysis, http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf, 2002.

[31] X. F. Wang and G. Chen, Complex networks: Small-world, scale-free and beyond, *IEEE Circuits and Systems Magazine*, vol. 3, no. 1, pp. 6–20, 2003.

[32] N. Wale, I. A. Watson, and G. Karypis, Comparison of descriptor spaces for chemical compound retrieval and classification, *Knowledge and Information Systems*, vol. 14, no. 3, pp. 347–375, 2008.

[33] J. Leskovec, J. Kleinberg, and C. Faloutsos, 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*, Chicago, IL, USA, 2005, pp. 177–187.

[34] M. Niepert, M. Ahmed, and K. Kutzkov, Learning convolutional neural networks for graphs, in *Proceedings of International Conference on Machine Learning*, New York, NY, USA, 2016, pp. 2014–2023.

Ruyue Xin received the master degree from Beijing Normal University in 2019. She is currently perusing the PhD degree in the University of van Amsterdam. Her research interests include graph theory and graph representation learning.

Yitong Shao received the bachelor degree from Beijing Normal University in 2019. He is a master student in Beijing Normal University now. His research interests include Markov chains, especially the branching processes.

Jiang Zhang received the PhD degree from Beijing Jiaotong University, Beijing, China，in 2006. From 2006 to 2008, he worked in China Academy of Sciences and finished his post doctoral research. From 2008 to 2012, he was an assistant professor in Beijing Normal University. From 2012 to 2018, he was an associate professor in Beijing Normal University. And now, he is a professor of the School of Systems Science in Beijing Normal University.

Prof. Zhang has many publications in journals and conferences, such as *Nature Communication*, *PLoS ONE*, *Physica A*, etc. His research interests include network reconstruction, discrete optimization problem on graph, and enterprise financial distress analysis.