Hypergraph based learning method : Embedding, Clustering, and Classification

Loc Tran

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à École Pratique des Hautes Études

Hypergraph based learning method: Embedding, Clustering, and Classification

Soutenue par
Loc Tran
Le 25th March 2023

École doctorale n°472
École doctorale de l'École Pratique des Hautes Études

Spécialité
Informatique, Mathématiques et Applications

Composition du jury :

M. HACENE FOUCHAL Université de Reims Champagne-Ardenne	*Président du jury* *Examinateur*
M. NAHID EMAD UVSQ	*Rapporteur*
M. SOUFIAN BENAMOR UVSQ	*Membre du jury*
M. GUILLAUME GUERARD Léonard de Vinci Pole Universitaire	*Membre du jury*
M. HUY NGUYEN John von Neumann Institute	*Membre du jury*
M. MARC BUI PSL	*Directeur de thèse*

École Pratique des Hautes Études | PSL

Remerciements

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Loc Tran
October 2023

Acknowledgements

I would like to express my sincere gratitude to my advisor Prof. Marc Bui for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

Résumé

Le graphe des Laplaciens a été largement utilisé dans la réduction dimensionnelle méthodes, méthodes de regroupement et méthodes d'apprentissage semi-supervisé depuis les années 2000. Les candidatures de ces méthodes sont énormes comme im- recherche d'âge, séparation de la parole, et la prédiction de la fonction des protéines tion. Comment- jamais, dans de nombreuses applications du monde réel, représentant l'ensemble de données sous forme de graphique n'est pas complet. Com-relation plex comme par paire conduira à la perte d'informations. Le naturel moyen de surmonter la perte d'informations est de représenter l'ensemble de données comme l'hy- pergraphe. Dans cette thèse, la dimension méthodes de réduction fonctionnelle, clustering méthodes et méthodes de classification pour la structure de données hypergraphiques être développé. Ce travail comprend la méthode classique d'apprentissage automatique ods et l'apprentissage profond moderne méthodes de structure de données hypergraphiques ture.

Mots clés : graphe, hypergraphe, laplacien, méthode d'apprentissage semi-supervisé, clustering, plongements, réseau de neurones.

Abstract

The graph Laplacians has been widely used in dimensional reduction methods, clustering methods, and semi-supervised learning methods (i.e., Laplacian Eigenmaps, spectral clustering, and graph-based semi-supervised learning) since 2000s (i.e., the classical machine learning method). The applications of these methods are huge such as image retrieval (Laplacian Eigenmaps), speech separation (spectral clustering), and protein function prediction (graph Laplacian based semi-supervised learning method). However, in many real-world applications, representing the dataset as the graph is not complete. Approximating complex relationship as pairwise will lead to the loss of information. The natural way overcoming the information loss is to represent the dataset as the hypergraph. In this thesis, the dimensional reduction methods, clustering methods, and classification methods for hypergraph data structure (i.e., utilizing the hypergraph Laplacian) will be developed. This work includes the classic machine learning methods and the modern deep learning methods for hypergraph data structure.

Keywords : graph, hypergraph, Laplacian, semi-supervised learning method, clustering, embeddings, neural network.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Why do we need to employ the hypergraph data structure?

In the past, the relational dataset was modeled as the high dimensional samples. In the vector space, the correlations among data points were defined as the distance function or the similarity function. However, modelling the samples with relationships would be more natural than modelling them as the high dimensional vectors. In the other words, the relationships are basically the subsets of samples and the correlations among samples are the intersections among the subsets. In this thesis, we will mainly discuss two types of relationships which are the pairwise relationship and the co-occurrence relationships. A pairwise relationship can also be called a binary relationship. In the other words, it contains just two samples. In the other hands, the co-occurrence relationship is the extension of the pairwise relationship. In the other words, the co-occurrence relationship can contain any number of samples that have co-occurred.

Given a relational dataset, the pairwise relationships among objects/entities/samples in this dataset can be represented as the weighted graph. Then, the un-supervised learning techniques such as representational learning methods/dimensional reduction methods and clustering methods and the semi-supervised learning techniques can be applied to this graph. These techniques (the un-supervised learning techniques and the semi-supervised learning techniques) can be formulated as the operations on this graph. The fundamental matrices used in these techniques are the adjacency matrix of the graph and/or the Laplacian matrix of the graph [1, 2].

However, assuming the pairwise relationships among the objects/entities/samples in this graph representation is not complete. Let's consider the case that we would like to partition/segment a set of articles into different topics (i.e., clustering problem) [3, 4].

Initially, we employ the graph data structure to represent this dataset. The vertices of the graph are the articles. Two articles are connected by an edge (i.e., the relationship) if there is at least one author in common. Finally, we can apply spectral clustering technique [5, 6] to this graph to partition/segment the vertices into groups/clusters.

Obviously, we easily see that in this graph data structure, we **ignore** the information whether **one specific author is the author of three or more articles (i.e., the co-occurrence relationship or high order relationship).**

This will lead to **the loss of information**. In the other words, this will lead to the low performance (i.e., the low accuracy) of the clustering technique.

In order to overcome this difficulty, [3, 4] try to employ the hypergraph data structure to represent for the above relational dataset. In details, in this hypergraph data structure, the articles are the vertices and the authors are the hyper-edges. This hyper-edge can connect more than two vertices (i.e., articles).

Please note that the simplicial complex is the uniform hypergraph which is one specific type of hypergraph. The uniform hypergraph is the hypergraph where all hyperedges have the same cardinality. However, in this thesis, we mainly discuss about the general hypergraph.

The following figure 1.1 shows the example of the hypergraph.

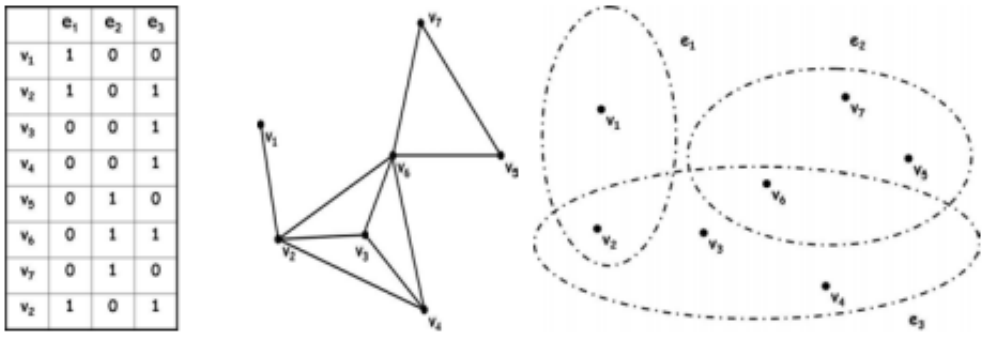

Fig. 1.1 Hypergraph example with 8 vertices and 3 hyper-edges [3]

There are a lot of ways to re-present for this hypergraph data structure: as the incidence matrix (used in my thesis and in [3]) or as the tensor [7, 8]. From [1], we recognize that the outcome of the hypergraph-based clustering technique is quite promising. Its performance is better than the performance of the spectral clustering technique (for graph).

Last but not least, this hypergraph data structure is very useful when it is employed to model social networks (the hyper-edges are group chats in social networks) or circuits (the hyper-edges are electrical components), etc.

1.2 Scientific challenges and contributions

Basically, in this thesis, I will develop the **novel** methods (i.e., novelty property) to solve various machine learning/deep learning problems for hypergraph data structure.

To the best of my knowledge, there are just four main problems in machine learning/deep learning research field such as:

- Representational learning/Dimensional reduction/...
- Clustering
- Classification
- Link prediction/Recommendation system/...

The fundamental matrices used in the techniques solving these four main machine learning/deep learning problems (for hypergraph data structure) are the incidence matrix of the hypergraph and/or the Laplacian matrix of the hypergraph [3, 4].

In details, from 2000 to 2010, computer scientists such as Mikhail Belkin, Ulrike Von Luxburg, and Dengyong Zhou had special interest in employing **graph Laplacian** in dimensional reduction methods [9, 10], clustering methods [11, 12], and semi-supervised learning methods (i.e., graph-based classification methods) [13, 14]. Our objective is to extend their works.

In 2007, Ulrike von Luxburg has shown clearly that there are three main types of graph Laplacians [11] which are:

- Un-normalized graph Laplacian
- Random walk graph Laplacian
- Symmetric normalized graph Laplacian

In order to solve the dimensional reduction problem for graph data structure, in 2002, Mikhail Belkin and Partha Niyogi introduced a theoretical framework for Laplacian Eigenmaps (i.e., dimensional reduction method) [9]. Unlike Principal Component Analysis (PCA),

Laplacian Eigenmaps can preserve the local structure of the data points/samples after the mapping [9]. This is the very strong argument of Laplacian Eigenmaps. However, the authors of Laplacian Eigenmaps methods (i.e., Mikhail Belkin and Partha Niyogi) did not point out their Eigenmaps is the random walk normalized Laplacian Eigenmaps or symmetric normalized Laplacian Eigenmaps when they try to solve the generalized eigenvalue problem

$$Ly = \lambda Dy \tag{1.1}$$

where L is un-normalized graph Laplacian matrix and D is the degree matrix. In particular, there exist two ways to solve this generalized eigenvalue problem and this fact will lead to two completely different Eigenmaps. To the best of my knowledge, no one have pointed out this fact. Moreover, **the un-normalized Laplacian Eigenmaps has not been investigated up to now.**

In order to solve the clustering problem for graph data structure, computer scientists normally employed the adjacency matrix of the graph and the graph Laplacian. In 1972, Donath and Hoffman initially suggested using the eigenvectors of adjacency matrices of graphs to find partitions/clusters/groups [15]. In 1973, Fiedler linked the second smallest eigenvalue of the Laplacian of the graph with its connectivity and proposed partitioning by splitting vertices according to their value in the corresponding eigenvector [16]. The Laplacian of the graph used by Fiedler is in fact the un-normalized or combinatorial graph Laplacian. In the other words, the spectral clustering method used by Fiedler is the un-normalized spectral clustering. Next, in 1992, Lars Hagen and Andrew Kahng applied un-normalized spectral clustering to circuit partitioning problem and justify the un-normalized spectral clustering method by using the graph cut point of view [17]. In 2000, Shi and Malik developed the random walk spectral clustering method and applied this method to image segmentation problem [18]. In 2001, Ng, Jordan, and Weiss developed the symmetric normalized spectral clustering method [12]. Finally, in 2007, Ulrike von Luxburg provided the complete review of spectral clustering methods in [11]. She justified all the spectral clustering methods by using the graph cut point of view and also provided the complete review of the multiway spectral clustering methods.

The advantage of this spectral clustering method is that it is easy to implement and it can be solved efficiently. First, the first phase of the spectral clustering method is to apply the Laplacian Eigenmaps algorithm to the dataset in order to reduce the dimensions of the dataset. Finally, the common k-means clustering algorithm will be applied to this "transformed" dataset.

In order to solve the graph-based classification problem (i.e., graph based semi-supervised learning problem), computer scientists employed the graph Laplacian. In details, in 2002,

Xiaojin Zhu et al. developed the random walk graph Laplacian based semi-supervised learning method and applied this method to various classical applications such as digit recognition [13]. In 2004, Dengyong Zhou et al. developed the symmetric normalized graph Laplacian based semi-supervised learning method and applied this method to various classical applications such as digit recognition and text classification [14]. In 2005 and 2009, Koji Tsuda et al. developed the un-normalized graph Laplacian based semi-supervised learning method and applied this method successfully to the practical bioinformatics problem which is the protein function prediction problem [19, 20].

In [21, 22], the symmetric normalized graph p-Laplacian based semi-supervised learning method has been developed by Dengyong Zhou but has not been applied to any practical applications. To the best of my knowledge, the un-normalized graph p-Laplacian based semi-supervised learning method has not yet been developed and obviously has not been applied to any practical applications. This method is worth investigated because of its difficulty and its close connection to partial differential equation on graph research area.

However, in many real-world applications, representing the set of objects and their relationships as graph data structure is not complete. Approximating complex relationship as pairwise relationship will lead to the loss of information and the low performance of the clustering and the classification methods. Thus, in this thesis, we will try to develop the dimensional reduction methods, the clustering methods, and the semi-supervised learning methods for the hypergraph data structure. Hypergraph is the generalization of the graph. In details, in graph, the edge can connect two vertices of the graph only. However, in hypergraph, edge or hyper-edge can connect more than two vertices of the hypergraph. Similar to graph, we will define three main hypergraph Laplacians which are:

- Un-normalized hypergraph Laplacian
- Random walk hypergraph Laplacian
- Symmetric normalized hypergraph Laplacian

In 2005, Dengyong Zhou et al. have developed the symmetric normalized hypergraph Laplacian Eigenmaps algorithm, the symmetric normalized hypergraph spectral clustering, and the symmetric normalized hypergraph Laplacian based semi-supervised learning algorithm [3].

Inspired from Dengyong Zhou's work, in this thesis, we will develop the un-normalized hypergraph Laplacian Eigenmaps algorithm. Then, we will develop the un-normalized hypergraph Laplacian based semi-supervised learning algorithm.

Moreover, in the developed un-normalized hypergraph Laplacian Eigenmaps algorithm, the weights of all hyper-edges are assumed to be equal to 1. This is not true at all in practical applications. In the other words, some hyper-edges may be more important than other hyper-edges and thus will have the weights that are larger than the weights of other hyper-edges. Thus, in this thesis, we will also develop the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm and the weighted un-normalized hypergraph Laplacian based semi-supervised learning algorithm. These novel techniques will be applied to the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset.

In [21], in 2005, Dengyong Zhou et al. have developed the symmetric normalized graph p-Laplacian based semi-supervised learning methods but has not applied these methods to any practical applications. **Inspired from Dengyong Zhou's work, in this thesis, the un-normalized hypergraph p-Laplacian based semi-supervised learning methods will be developed based on the un-normalized hypergraph p-Laplacian operators' definitions such as the curvature operator of hypergraph (i.e., the un-normalized hypergraph 1-Laplacian operator). Then, the un-normalized hypergraph p-Laplacian based semi-supervised learning methods will be applied to the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset.**

To develop the hypergraph based neural network can be considered the very hard task and this hypergraph based neural network has just been developed since 2019 [23].

However, most of hypergraph based neural networks were developed to solve:

- Classification problem (for example, text classification, image classification, etc.) [24, 25].

to the best of my knowledge

In this thesis, I will develop novel hypergraph based neural networks (i.e., novel versions) to solve three basic problems in machine learning/deep learning research field:

- Classification

- Representational learning/Dimensional reduction/...

- Clustering

There are various clustering techniques, available from Python sklearn package [26] such as k-means [27], hierarchical clustering technique, affinity propagation, etc., that can be employed to solve these clustering problems. However, please note that these techniques can ONLY be applied to feature datasets.

Moreover, there are other clustering techniques (i.e., belong to different class of clustering techniques) that can also be employed to solve the clustering problem but can ONLY be applied to hypergraph datasets such as the symmetric normalized hypergraph spectral clustering [3, 4], maximum modularity approach [28], etc.

The weakness of these two classes of clustering techniques is obviously that they can ONLY be applied to one type of dataset. This will lead to the loss of information. Now, let's consider the case that we have both types of datasets such as the feature dataset and the hypergraph dataset. Assume that the samples in both datasets are the same, how can we apply the clustering technique to both datasets?

In this thesis, we will develop the novel clustering method that utilizes both the feature dataset and the hypergraph dataset.

In details, initially, we will develop the Hypergraph Auto-Encoders method (i.e., the dimensional reduction method). Then, the Hypergraph Auto-Encoders method will be employed to transform both the hypergraph dataset and the feature dataset from the high dimensional space to low dimensional space. Finally, the k-means clustering technique will be applied to the transformed dataset. This novel clustering technique will be called the hypergraph convolutional neural network-based clustering technique.

There are some main advantages of this novel clustering technique such as:

- The memory/space required to store the data is lowered. This will lead to the low space complexity.
- The low space complexity will lead to less computational and training time of the classification/clustering techniques (i.e., the low time complexity).
- The noises and the redundant features are removed from the feature dataset and the hypergraph dataset. This will lead to the high performance of the clustering technique.
- Both feature dataset and network dataset are utilized. This will lead to no information loss.

Last but not least, these clustering techniques are un-supervised learning techniques. Hence, we do not need labeled datasets.

In recent years, deep convolution neural networks have gained much interests from data scientists and have been utilized to solve many classification tasks such as image recognition [29] and speech recognition [30], to name a few. In order to deal with irregular data structure, graph convolution neural networks have been developed by a lot of data scientists such as Thomas Kipf [31]. There are two classes of graph convolution neural network. The

first class of graph convolution neural network is the spatial based approach. This spatial based approach implements the convolution on the graph by accumulating information of the neighbor nodes. The second class of graph convolution neural network is the spectral based approach. This spectral based approach implements a variant of graph convolution neural network based on different graph Laplacians. Easily, we recognize that the time complexity of spectral based approach is much higher than the time complexity of spatial based approach; however, the accuracy of the spectral based approach is higher than the accuracy of the spatial based approach. In this graph data structure, we easily see that the edge can connect only two vertices. In the other words, data scientists have concentrated primarily on developing deep neural network method for the simple un-directed graph.

Inspired from the idea combining the PageRank algorithm with the graph convolution neural network in [32], in this thesis, we propose the novel version of hypergraph neural network method combining the classic hypergraph based semi-supervised learning method [3, 4] with the hypergraph neural network method [23]. This novel hypergraph neural network method then will be employed to solve the "noisy label learning" problem.

In this graph data structure, we easily see that the edge not only can connect two vertices but also carry no information about the direction. **In order to overcome these two information losses which are "the edge carry no information about the direction" and "the edge can connect only two vertices" of the graph data structure which "can" affect the performance of the "node clustering task" and/or the "node classification task", in this thesis, we employ the directed hypergraph data structure [4, 5] and develop the deep neural network method based on this directed hypergraph data structure. This method is called the spectral directed hypergraph neural network method. The development of this directed hypergraph neural network is considered the very hard task and is the novel work.**

1.3 Organization of the thesis

This thesis will contain six chapters:

- Chapter 1: In this chapter, we present why we need hypergraph data structure. Then, the scientific challenges and the contributions of the thesis are presented.

- Chapter 2: In this chapter, initially, we will introduce the definitions of the three hypergraph Laplacians. Then, the un-normalized hypergraph Laplacian Eigenmaps algorithm will be presented in detail. Next, we will present the weighted un-normalized

hypergraph Laplacian Eigenmaps algorithm and the weighted un-normalized hypergraph Laplacian based semi-supervised learning algorithm. Finally, the experimental results of the proposed algorithms in this chapter will be shown. Last but not least, these proposed algorithms will be tested on the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset.

- Chapter 3: In this chapter, we will introduce the definitions of the gradient and divergence operators of hypergraph. Next, we will introduce the definition of Laplace operator of hypergrap and its properties. Then, we will introduce the definition of the curvature operator of hypergraph and its properties. Next, we will introduce the definition of the p-Laplace operator of hypergraph and its properties. In the next section, we will show how to derive the algorithm of the un-normalized hypergraph p-Laplacian based semi-supervised learning method from regularization framework. Finally, we will compare the accuracy performance measures of the un-normalized hypergraph Laplacian based semi-supervised learning algorithm (i.e., the current state of art method) and the un-normalized hypergraph p-Laplacian based semi-supervised learning algorithms.

- Chapter 4: In this chapter, initially, we will define the clustering problem and will present the novel graph/hypergraph convolutional neural network-based clustering technique. Next, we will describe the two datasets that will be used in this chapter which are the Citeseer dataset and the Cora dataset and will compare the performance the hypergraph convolutional neural network based-clustering technique and the performances of the graph convolutional neural network based-clustering technique, the k-means clustering technique and the spectral clustering technique testing on these two Citeseer and Cora datasets.

- Chapter 5: In this chapter, we will introduce the novel version of hypergraph neural network method which is the combination of the classic hypergraph based semi-supervised learning method with the hypergraph neural network method. Next, we will describe the datasets which will be used in this chapter (i.e., the MNIST image dataset, the USPS image dataset, and the FASHION-MNIST image dataset) and will present the experimental results.

- Chapter 6: In this chapter, initially, we will present the preliminary notations and definitions. Next, we will introduce the novel directed hypergraph semi-supervised learning method. Then, the directed hypergraph neural network will be presented.

Finally, we will describe the datasets that will be used in this chapter (i.e., the Cora dataset and the Citeseer dataset) and will present the experimental results.

- Chapter 7: Conclusion

References

[1] R. Merris, "Laplacian matrices of graphs: a survey," *Linear algebra and its applications*, vol. 197, pp. 143–176, 1994.

[2] D. A. Spielman, "Algorithms, graph theory, and linear equations in laplacian matrices," in *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pp. 2698–2722, World Scientific, 2010.

[3] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," *Advances in neural information processing systems*, vol. 19, pp. 1601–1608, 2006.

[4] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," 2005.

[5] J. Liu and J. Han, "Spectral clustering," in *Data Clustering*, pp. 177–200, Chapman and Hall/CRC, 2018.

[6] D. Verma and M. Meila, "A comparison of spectral clustering algorithms," *University of Washington Tech Rep UWCSE030501*, vol. 1, pp. 1–18, 2003.

[7] X. Ouvrard, J.-M. L. Goff, and S. Marchand-Maillet, "Adjacency and tensor representation in general hypergraphs part 1: e-adjacency tensor uniformisation using homogeneous polynomials," *arXiv preprint arXiv:1712.08189*, 2017.

[8] X. Ouvrard, J.-M. L. Goff, and S. Marchand-Maillet, "Adjacency and tensor representation in general hypergraphs. part 2: Multisets, hb-graphs and related e-adjacency tensors," *arXiv preprint arXiv:1805.11952*, 2018.

[9] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[10] M. Belkin and P. Niyogi, "Convergence of laplacian eigenmaps," *Advances in Neural Information Processing Systems*, vol. 19, p. 129, 2007.

[11] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[12] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, pp. 849–856, 2002.

[13] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," 2002.

[14] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in neural information processing systems*, pp. 321–328, 2004.

[15] W. E. Donath and A. J. Hoffman, "Algorithms for partitioning of graphs and computer logic based on eigenvectors of connection matrices," *IBM Technical Disclosure Bulletin*, vol. 15, no. 3, pp. 938–944, 1972.

[16] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973.

[17] L. Hagen and A. B. Kahng, "New spectral methods for ratio cut partitioning and clustering," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 11, no. 9, pp. 1074–1085, 1992.

[18] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[19] K. Tsuda, H. Shin, and B. Schölkopf, "Fast protein classification with multiple networks," *Bioinformatics*, vol. 21, no. suppl_2, pp. ii59–ii65, 2005.

[20] H. Shin, K. Tsuda, and B. Schölkopf, "Protein functional class prediction with a combined graph," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3284–3292, 2009.

[21] D. Zhou and B. Schölkopf, "Regularization on discrete spaces," in *Joint Pattern Recognition Symposium*, pp. 361–368, Springer, 2005.

[22] D. Zhou and B. Schölkopf, "Discrete regularization.," 2006.

[23] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3558–3565, 2019.

[24] K. Ding, J. Wang, J. Li, D. Li, and H. Liu, "Be more with less: Hypergraph attention networks for inductive text classification," *arXiv preprint arXiv:2011.00387*, 2020.

[25] M. Liao, J. Duan, R. Zhang, X. Zhou, X. Wu, X. Wang, and J. Hu, "A hypergraph-embedded convolutional neural network for ice crystal particle habit classification," *INTELLIGENT AUTOMATION AND SOFT COMPUTING*, vol. 29, no. 3, pp. 787–801, 2021.

[26] "2.3. clustering — scikit-learn 1.0.1 documentation." https://scikit-learn.org/stable/modules/clustering.html.

[27] L. H. Tran and L. H. Tran, "Mobility patterns based clustering: A novel approach," *International Journal of Machine Learning and Computing*, vol. 8, no. 4, 2018.

[28] D. Combe, C. Largeron, M. Géry, and E. Egyed-Zsigmond, "I-louvain: An attributed graph clustering method," in *International Symposium on Intelligent Data Analysis*, pp. 181–192, Springer, 2015.

[29] B. B. Traore, B. Kamsu-Foguem, and F. Tangara, "Deep convolution neural network for image recognition," *Ecological Informatics*, vol. 48, pp. 257–268, 2018.

[30] Y. Qian and P. C. Woodland, "Very deep convolutional neural networks for robust speech recognition," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 481–488, IEEE, 2016.

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

[32] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," *arXiv preprint arXiv:1810.05997*, 2018.

Chapter 2

Weighted Un-Normalized Hypergraph Laplacian Eigenmaps

2.1 Introduction

Recently, our capability to collect and store data has far exceeded our capability to analyze it. In problems such as face recognition and biological network inference problem using gene expression data, we are given a large dataset in which each observation contains a large number of variables. This number of variables is called the dimension of each observation. In this given setting, the data lies in a high-dimensional space. For example, the 256*256 image has 65536 dimensions if we treat each pixel as one variable/feature/attribute. It's hard for human beings to visualize and understand these high dimensional data because of limited computing resources. Moreover, it turns out that not all variables are needed for understanding the primary phenomenon. In the other words, there is a high degree of redundancy in the data they represent. Hence, the structure and the content of the data may be captured by a lesser set of variables. There are also may be too much noise in the data. Hence there is a need to reduce the dimensionality of the data (i.e., reduce the noise of the data) before we apply the clustering (i.e., un-supervised learning) methods and classification (i.e., semi-supervised learning and supervised learning) methods to the dataset. In the other words, we can build the more effective data analysis tools. Those are why we need to develop the dimensional reduction methods.

In our literature review, many dimensional reduction methods have been successfully developed and applied to various applications such as speech recognition, face recognition, and biological network inference problem using gene expression data, to name a few. To the best of my knowledge, there are two classes of dimensional reduction methods which

are the linear and the non-linear techniques [1]. Linear dimensional reduction methods assume that the data lies on or close to linear subspace of the high-dimensional ambient space. Linear dimensional reduction methods have been developed and used for a long time. For example, Principal Component Analysis (i.e., PCA) was developed in 1901 and is still the most widely used dimensional reduction methods nowadays. For instance, the PCA technique is employed in and successfully applied to speech recognition research field [2], face recognition research field [3], and biological network inference research field [4]. In the other hand, non-linear dimensional reduction methods make no assumption about the linearity and are designed to recognize complex non-linear manifolds as well as linear ones. Recently, many researchers have focused on developing various non-linear dimensional reduction methods such as Kernel PCA [5], Isomap [6], Local Linear Embedding [7], and Laplacian Eigenmaps [8].

However, in many real-world applications, representing the dataset as un-directed graph, used in Laplacian Eigenmaps and Local Linear Embedding methods, is not complete. Approximating complex relationship as pairwise will lead to the loss of information. Let us consider classifying a set of genes into different gene functions. From [9], we may construct an un-directed graph in which the vertices represent the genes and two genes are connected by an edge if these two genes show a similar pattern of expression (i.e., the gene expression data is used as the datasets in [9]). Any two genes connected by an edge tend to have similar functions. However, assuming the pairwise relationship between genes is not complete, the information a group of genes that show very similar patterns of expression and tend to have similar functions [10] (i.e., the functional modules) is missed. The natural way overcoming the information loss is to represent the gene expression data as the hypergraph [10]. A hypergraph is a graph in which an edge (i.e., a hyper-edge) can connect more than two vertices. However, representing the dataset as the hypergraph will not lead to the perfection. The number of hyper-edges may be large; hence this will lead to high time complexity of the clustering methods or the classification methods when we try to apply the clustering/classification methods to this hypergraph dataset. Thus, there exists a need to develop the dimensional reduction methods for the hypergraph datasets. In [11, 12], the symmetric normalized hypergraph Laplacian Eigenmaps has been developed and successfully applied to zoo dataset. To the best of my knowledge, the random walk and un-normalized hypergraph Laplacian Eigenmaps have not yet been developed and applied to any practical applications. In this chapter, we will develop the un-normalized hypergraph Laplacian Eigenmaps and apply this method combined with kernel ridge regression method to the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset.

Moreover, in the developed un-normalized hypergraph Laplacian Eigenmaps algorithm, we assume that the weights of all hyper-edges are equal to 1. This is not true at all in practical applications. In the other words, some hyper-edges may be more important than other hyper-edges and thus will have the weights that are larger than the weights of other hyper-edges. Thus, in this chapter, we will also develop the weighted un-normalized hypergraph Laplacian Eigenmaps and apply this method, combined with kernel ridge regression method, to the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset.

We will organize this chapter as follows: Section 2.2 will introduce the definitions of the three hypergraph Laplacians. Section 2.3 will present the un-normalized hypergraph Laplacian Eigenmaps algorithm in detail. Section 2.4 will present the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm in detail. Section 2.5 will show the experimental results of the un-normalized hypergraph Laplacian Eigenmaps algorithm and the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm combined with the kernel ridge regression method applied to the zoo dataset available from UCI repository and the tiny version of the 20 newsgroups dataset. Section 2.6 will conclude this chapter and the future direction of researches will be discussed.

2.2 Preliminary notations and definitions

Given a hypergraph $G = (V, E)$, where V is the set of vertices and E is the set of hyper-edges. Each hyper-edge $e \in E$ is the subset of V. Please note that the cardinality of e is greater than or equal two. In the other words, $|e| \geq 2$, for every $e \in E$. Let $w(e)$ be the weight of the hyper-edge e. Then W will be the $R^{|E|*|E|}$ diagonal matrix containing the weights of all hyper-edges in its diagonal entries.

The incidence matrix H of G is a $R^{|V|*|E|}$ matrix that can be defined as follows

$$h(v,e) = \begin{cases} 1 \; if \; vertex \; v \; belongs \; to \; hyperedge \; e \\ 0 \; otherwise \end{cases} \tag{2.1}$$

From the above definition, we can define the degree of vertex v and the degree of hyper-edge e as follows

$$d(v) = \sum_{e \in E} w(e) * h(v,e) \tag{2.2}$$

$$d(e) = \sum_{v \in V} h(v,e) \tag{2.3}$$

Let D_v *and* D_e be two diagonal matrices containing the degrees of vertices and the degrees of hyper-edges in their diagonal entries respectively. Please note that D_v is the $R^{|V|*|V|}$ matrix and D_e is the $R^{|E|*|E|}$ matrix.

Please note that, we assume that the weight of each hyper-edge is 1.

The un-normalized hypergraph Laplacian [10,11,12] is defined as follows

$$L = D_v - HWD_e^{-1}H^T \tag{2.4}$$

The symmetric normalized hypergraph Laplacian [11,12] is defined as follows

$$L_{sym} = I - D_v^{-\frac{1}{2}}HWD_e^{-1}H^TD_v^{-\frac{1}{2}} \tag{2.5}$$

The random walk hypergraph Laplacian [10,11,12] is defined as follows

$$L_{rw} = I - D_v^{-1}HWD_e^{-1}H^T \tag{2.6}$$

2.3 Un-normalized hypergraph Laplacian Eigenmaps algorithm

Suppose that we are given the hypergraph. In the other words, we know the topology of the hypergraph. What we want to do is to transform each node of the hypergraph to a numerical vector utilizing the topology of the hypergraph. The un-normalized hypergraph Laplacian Eigenmaps algorithm will exactly map each node of the hypergraph to numerical vector.

Un-normalized hypergraph Laplacian Eigenmaps algorithm

In this part, we will give the brief overview of the un-normalized hypergraph Laplacian Eigenmaps algorithm. The outline of this algorithm is as follows

1. Construct D_v *and* D_e from the incidence matrix H of G
2. Compute the un-normalized hypergraph Laplacian $L = D_v - HWD_e^{-1}H^T$
3. Compute all eigenvalues and eigenvectors of L and sort all eigenvalues and their corresponding eigenvector in ascending order. Pick the first k eigenvectors $v_2, v_3, \ldots, v_{k+1}$ of L in the sorted list. k can be determined in the following two ways:
   (a) k is the number such that $\frac{\lambda_{k+2}}{\lambda_{k+1}}$ is largest for all $2 \leq k \leq |V|$
   (b) k is the number such that $\lambda_{k+2} - \lambda_{k+1}$ is largest for all $2 \leq k \leq |V|$

4. Let $U \in R^{|V|*k}$ be the matrix containing the vectors $v_2, v_3, \ldots, v_{k+1}$ as columns and U is the final result

2.4 Weighted un-normalized hypergraph Laplacian Eigenmaps algorithm

In the above un-normalized hypergraph Laplacian Eigenmaps algorithm, we assume that the weights of all hyper-edges are equal to 1. This is not true at all in practical applications. In the other words, some hyper-edges may be more important than other hyper-edges and thus will have the weights that are larger than the weights of other hyper-edges.

Hence in order to assign weights to hyper-edges of the hypergraph, to transform the nodes of the hypergraph to numerical vectors, and to improve the accuracy of the classification algorithms of hypergraphs, we would like to solve the following minimization problems

$$argmin_{f,w} \frac{1}{2} \sum_{e \in E} \sum_{\{u,v\} \subseteq E} \frac{w(e)}{d(e)} (f(u) - f(v))^2 + \alpha ||f - y||^2 + \beta ||w||^2 \; such \; that \; 1_{|E|}^T w = 1 \tag{2.7}$$

Please note that w is the vector containing the weights of all the hyper-edges of the hypergraph, $f \in R^{|V|}$ is the final ranking vector (i.e., the output vector) that is used for the classification of the nodes of the hypergraph with some threshold value.

Moreover, we know that

$$\frac{1}{2} \sum_{e \in E} \sum_{\{u,v\} \subseteq E} \frac{w(e)}{d(e)} (f(u) - f(v))^2 = f^T L f \tag{2.8}$$

The proof of (2.8) can be found in [10].

Thus, we need to solve the following optimization problem

$$argmin_{f,w} f^T L f + \alpha ||f - y||^2 + \beta ||w||^2 \; such \; that \; 1_{|E|}^T w = 1 \tag{2.9}$$

With a fixed w, we can optimize f like the following

$$argmin_f f^T L f + \alpha ||f - y||^2 = argmin_f f^T L f + \alpha (f - y)^T (f - y))$$

In the other words, we need to solve

$$\frac{\partial(f^T Lf + \alpha (f-y)^T (f-y))}{\partial f} = 0$$

This will lead to

$$Lf + \alpha (f-y) = 0$$

$$(L+\alpha I) f = \alpha y$$

Hence the solution f^* of the above equations is

$$f^* = \alpha (L+\alpha I)^{-1} y$$

With a fixed f, we can optimize w like the following

$$argmin_w f^T Lf + \beta ||w||^2 \; such \; that \; 1_{|E|}^T w = 1$$

The Lagrangian function of the above optimization problem is

$$\begin{aligned} \delta (w, \gamma) &= f^T Lf + \beta w^T w + \gamma (1_{|E|}^T w - 1) \\ &= f^T \left(D_v - HWD_e^{-1} H^T\right) f + \beta w^T w + \gamma (1_{|E|}^T w - 1) \end{aligned}$$

Next, we need to solve

$$\frac{\partial (f^T \left(D_v - HWD_e^{-1} H^T\right) f)}{\partial w_i} + 2\beta w_i + \gamma = 0$$

This will lead to

$$w_i = \frac{1}{2\beta}[-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i} - \gamma]$$

Moreover, we know that

$$1_{|E|}^T w = 1$$

In the other words, we have

$$\sum_i \frac{1}{2\beta}\left[-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i} - \gamma\right] = 1$$

$$\frac{1}{2\beta}\sum_i \left[-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i} - \gamma\right] = 1$$

$$2\beta = \sum_i (-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i}) - |E|\,\gamma$$

$$\gamma = \frac{1}{|E|}[\sum_i (-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i}) - 2\beta]$$

In general, we have

$$\gamma = \frac{1}{|E|}[\sum_i (-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i}) - 2\beta]$$

$$w_i = \frac{1}{2\beta}[-\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i} - \gamma]$$

Now, we need to compute $\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i}$.
We have

$$\frac{\partial\left(f^T\left(D_v - HWD_e^{-1}H^T\right)f\right)}{\partial w_i} = f^T\frac{\partial(D_v)}{\partial w_i}f - f^T H\frac{\partial W}{\partial w_i}D_e^{-1}H^T f$$

$$= f^T\left(diag\left(H\left(:,i\right)\right) - D_e^{-1}\left(i,i\right)H\left(:,i\right)H\left(:,i\right)^T\right)f$$

Thus, finally, we have that

$$\gamma = \frac{1}{|E|}[\sum_i\left(-f^T\left(diag\left(H\left(:,i\right)\right) - D_e^{-1}\left(i,i\right)H\left(:,i\right)H\left(:,i\right)^T\right)f\right) - 2\beta] \tag{2.10}$$

$$w_i = \frac{1}{2\beta}[-f^T\left(diag\left(H\left(:,i\right)\right) - D_e^{-1}\left(i,i\right)H\left(:,i\right)H\left(:,i\right)^T\right)f - \gamma] \tag{2.11}$$

Weighted un-normalized hypergraph Laplacian Eigenmap algorithm

In this part, we will give the brief overview of the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm. The outline of this algorithm is as follows

1. Construct D_v and D_e from the incidence matrix H and matrix W of G (initially, W is the identity matrix)
2. Compute the un-normalized hypergraph Laplacian $L = D_v - HWD_e^{-1}H^T$
3. Compute $f = \alpha(L+\alpha I)^{-1}y$
4. Compute the weight matrix W

$$\gamma = \frac{1}{|E|}[\sum_i\left(-f^T\left(diag\left(H\left(:,i\right)\right) - D_e^{-1}\left(i,i\right)H\left(:,i\right)H\left(:,i\right)^T\right)f\right) - 2\beta]$$

$$w_i = \frac{1}{2\beta}[-f^T\left(diag\left(H\left(:,i\right)\right) - D_e^{-1}\left(i,i\right)H\left(:,i\right)H\left(:,i\right)^T\right)f - \gamma]$$

5. Update the matrix D_v and W
6. Repeat step 2-step 5 until convergence and get the final ranking vector f.
7. Compute all eigenvalues and eigenvectors of the "updated" L and sort all eigenvalues and their corresponding eigenvector in ascending order. Pick the first k eigenvectors $v_2, v_3, \ldots, v_{k+1}$ of L in the sorted list. k can be determined in the following two ways:

(a) k is the number such that $\frac{\lambda_{k+2}}{\lambda_{k+1}}$ is largest for all $2 \leq k \leq |V|$

(b) k is the number such that $\lambda_{k+2} - \lambda_{k+1}$ is largest for all $2 \leq k \leq |V|$

8. Let $U \in R^{|V|*k}$ be the matrix containing the vectors $v_2, v_3, \ldots, v_{k+1}$ as columns and U is the final result

Clearly, from the above algorithm, we recognize that the weighted un-normalized hyper-graph Laplacian Eigenmaps algorithm contains the weighted hyper-graph based semi-supervised learning method starting from step 1 to step 6.

2.5 Experiments and Results

In this chapter, we used the zoo data set and the tiny version of 20 newsgroups dataset which can be obtained from UCI repository and from http://www.cs.nyu.edu/~roweis/data.html respectively.

The zoo data set contains 101 animals with 17 attributes. The attributes include hair, feathers, eggs, milk, etc. The animals have been classified into 7 different classes. In this dataset, each attribute is the hyper-edge.

The tiny version of the 20 newsgroups dataset contains the binary occurrence data for 100 words across 16242 postings. However, we just choose small subset of this tiny dataset containing 4000 postings in order to test our algorithms. In this dataset, each word is the hyper-edge.

In this section, we experiment with the above proposed un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method [13], the hyper-graph based semi-supervised learning method [10–12], and the weighted hyper-graph based semi-supervised learning method applied directly to the zoo dataset and the tiny version of 20 newsgroups dataset in terms of accuracy performance measure. The accuracy performance measure Q is given as follows

$$Q = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative}$$

All experiments were implemented in Matlab 6.5 on virtual machine. The accuracy performance measures of the four above proposed methods are given in the following table 2.1 and table 2.2.

The following figure 2.1 shows the **accuracies** of the four proposed methods which are the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized

Table 2.1 **Accuracies** of the four proposed methods which are the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the **zoo dataset**

Accuracy (%)			
Un-normalized hypergraph Laplacian Eigenmaps + Kernel ridge regression method	Weighted un-normalized hypergraph Laplacian Eigenmaps + Kernel ridge regression method	Hyper-graph based semi-supervised learning method	Weighted hyper-graph based semi-supervised learning method
95.77	95.77	90.48	98.64

Table 2.2 **Accuracies** of the four proposed methods which are the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the **20 newsgroups dataset**

Accuracy (%)			
Un-normalized hypergraph Laplacian Eigenmaps + Kernel ridge regression method	Weighted un-normalized hypergraph Laplacian Eigenmaps + Kernel ridge regression method	Hyper-graph based semi-supervised learning method	Weighted hyper-graph based semi-supervised learning method
86.92	85.50	86.05	87.18

hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the **zoo dataset**:

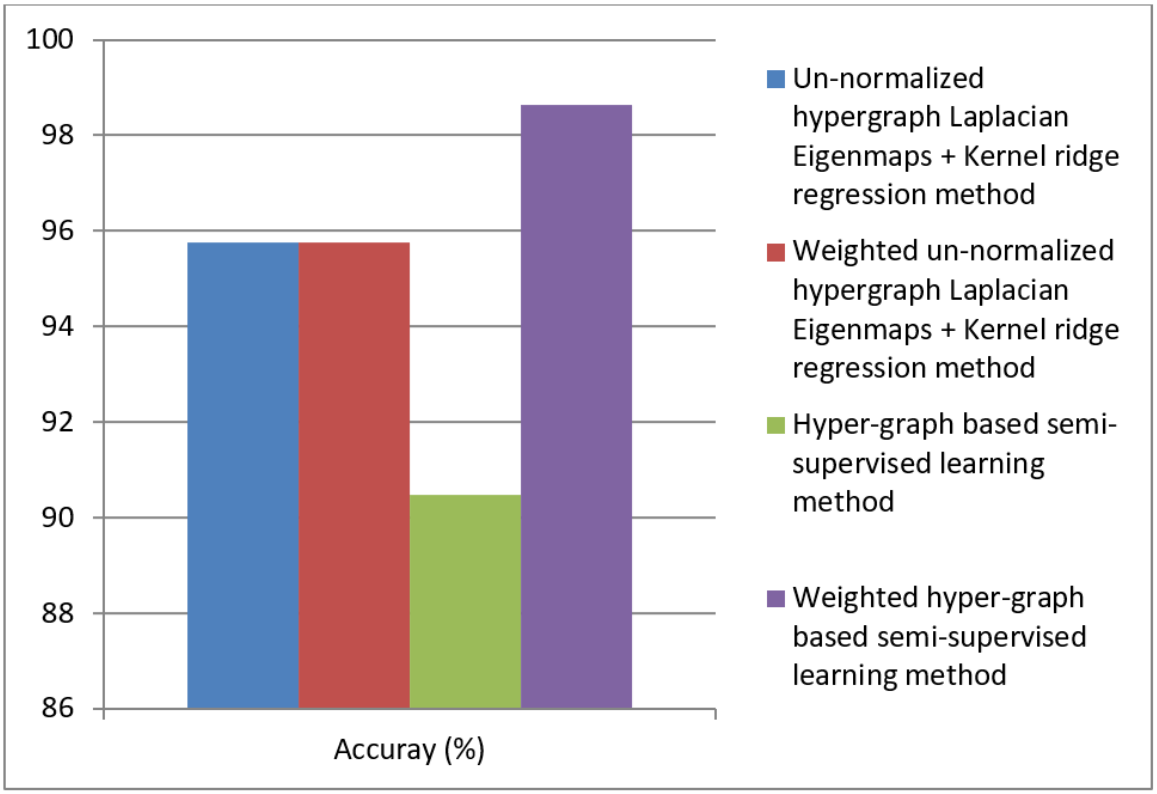

Fig. 2.1 The **accuracies** of the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the **zoo dataset**

The following figure 2.2 shows the accuracies of the four proposed methods which are the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the 20 newsgroups dataset:

From the above figures, we easily recognized that the weighted hyper-graph semi-supervised learning method achieves the highest accuracy performance measures since the solution vector of the weighted hyper-graph semi-supervised learning method is directly obtained from the optimization problem (2.7) which is used to maximize the accuracy of the hypergraph semi-supervised learning method. In the other hands, the weighted un-normalized hypergraph Laplacian Eigenmaps do not always perform better the un-normalized hypergraph

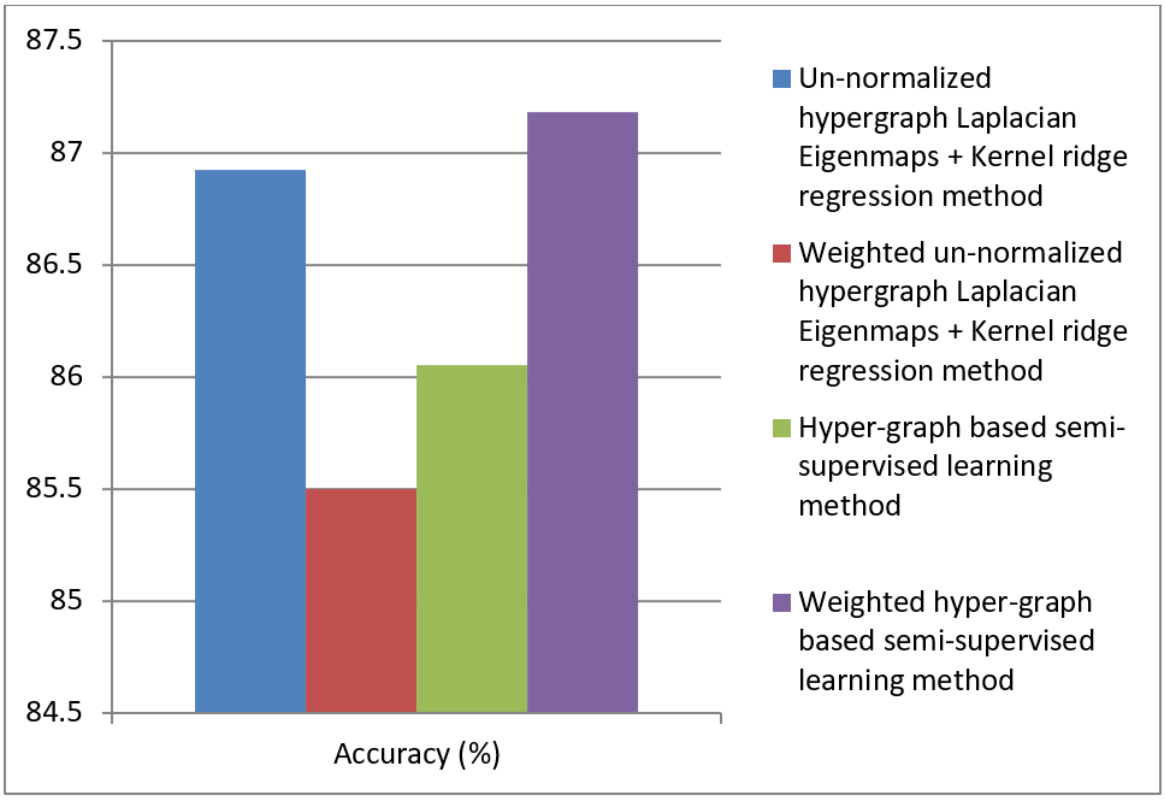

Fig. 2.2 The **accuracies** of the un-normalized hypergraph Laplacian Eigenmaps and the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the hyper-graph based semi-supervised learning method, and the weighted hyper-graph based semi-supervised learning method for the **20 newsgroups dataset**

Laplacian Eigenmaps algorithm since these two algorithms are combined with the kernel ridge regression algorithm, which is not the best supervised classification algorithm (especially for the feature vectors which are results of the weighted un-normalized hypergraph Laplacian Eigenmaps and un-normalized hypergraph Laplacian Eigenmaps) for the scope of this chapter.

2.6 Conclusions

In this chapter, we have proposed the detailed algorithms of the un-normalized hypergraph Laplacian Eigenmaps, the weighted un-normalized hypergraph Laplacian Eigenmaps applying to the zoo dataset and the tiny version of 20 newsgroups dataset. Interestingly, experiments show that the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm do not always perform better the un-normalized hypergraph Laplacian Eigenmaps algorithm. However, the weighted hypergraph semi-supervised learning method do always

perform better than the un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, the weighted un-normalized hypergraph Laplacian Eigenmaps combined with the kernel ridge regression method, and the hypergraph semi-supervised learning method.

In the future, the un-normalized hypergraph Laplacian Eigenmaps, the weighted un-normalized hypergraph Laplacian Eigenmaps, and the weighted hypergraph semi-supervised learning method will be applied to larger hypergraphs such as web hypergraph (to detect spam or not) and will be implemented in Python and MapReduce.

To the best of my knowledge, the un-normalized hypergraph p-Laplacian Eigenmap has not yet been developed. This method is worth investigated because of its difficult nature and its close connection to partial differential equation on hypergraph field.

References

[1] L. Van Der Maaten, E. Postma, J. Van den Herik, *et al.*, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, no. 66-71, p. 13, 2009.

[2] H. Trang, T. H. Loc, and H. B. H. Nam, "Proposed combination of pca and mfcc feature extraction in speech recognition system," in *2014 International Conference on Advanced Technologies for Communications (ATC 2014)*, pp. 697–702, IEEE, 2014.

[3] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*, pp. 586–587, IEEE Computer Society, 1991.

[4] J.-P. Vert and Y. Yamanishi, "Supervised graph inference," in *Advances in neural information processing systems*, pp. 1433–1440, 2005.

[5] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[6] 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.

[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] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[9] L. Tran, "Application of three graph laplacian based semi-supervised learning methods to protein function prediction problem," *arXiv preprint arXiv:1211.4289*, 2012.

[10] L. Tran, "Hypergraph and protein function prediction with gene expression data," *arXiv preprint arXiv:1212.0388*, 2012.

[11] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," *Advances in neural information processing systems*, vol. 19, pp. 1601–1608, 2006.

[12] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," 2005.

[13] H. Trang and L. Tran, "Kernel ridge regression method applied to speech recognition problem: A novel approach," in *2014 International Conference on Advanced Technologies for Communications (ATC 2014)*, pp. 172–174, IEEE, 2014.

Chapter 3

Un-Normalized Hypergraph P-Laplacian Based Semi-Supervised Learning Methods

3.1 Introduction

To classify the samples is the important problem in machine learning research area. Identifying the class of samples by human effort is very expensive and hard. Hence a lot of computational methods have been proposed to infer the classes of the samples.

To predict the class of the sample, machine learning methods such as the k-nearest neighbors [1, 2], Artificial Neural Networks [3, 4], Support Vector Machine [5, 6], the un-normalized graph Laplacian based semi-supervised learning method [7–13], or the symmetric normalized and random walk graph Laplacian based semi-supervised learning methods [14, 15] can be employed infer the classes of un-labeled samples. While the k-nearest neighbors, the Artificial Neural Networks, and the Support Vector Machine are supervised learning methods, the un-normalized, random walk, and symmetric normalized graph Laplacian based semi-supervised learning methods are graph based semi-supervised learning methods.

The un-normalized, symmetric normalized, and random walk graph Laplacian based semi-supervised learning methods are developed based on the assumption that the labels of two adjacent samples in the network are likely to be the same [7, 8, 14, 15, 9]. Hence this assumption can be interpreted as pair of samples showing a similar pattern and thus sharing edge in the network tends to have similar classes/labels.

In detail, in 2002, Xiaojin Zhu et al. developed the random walk graph Laplacian based semi-supervised learning method and applied this method to various classical applications such as digit recognition [14]. In 2004, Dengyong Zhou et al. developed the symmetric normalized graph Laplacian based semi-supervised learning method and applied this method to various classical applications such as digit recognition and text classification [15]. In 2005 and 2009, Koji Tsuda et al. developed the un-normalized graph Laplacian based semi-supervised learning method and applied this method successfully to the practical bio-informatics problem which is the protein function prediction problem.

In [16–19], the tabular dataset is employed for classification problem. However, assuming the pairwise relationship between samples/feature vectors is not complete; the information a group of samples that shows very similar pattern and tends to have similar classes/labels [16–19] is missed. The natural way overcoming the information loss of the above assumption is to represent the tabular dataset as the hypergraph [16–19]. A hypergraph is a graph in which an edge (i.e., a hyper-edge) can connect more than two vertices. In [16, 17], Dengyong Zhou et al. developed the symmetric normalized hypergraph Laplacian based semi-supervised learning method and successfully applied this method to various applications such as digit recognition and text classification. In [18, 19], the un-normalized and random walk hypergraph Laplacian based semi-supervised learning methods, which are the two variants of the method developed by Dengyong Zhou, have been developed and successfully applied to protein function prediction and speech recognition problems by Loc Tran in 2014. In general, these three hypergraph Laplacian based semi-supervised learning methods successfully outperform the un-normalized, symmetric normalized, and random walk graph Laplacian based semi-supervised learning methods in classification problem and are completely studied.

To the best of our knowledge, the un-normalized hypergraph p-Laplacian based semi-supervised learning methods have not yet been developed and obviously have not been applied to any practical applications. This method is worth investigated because its nature is difficult and because of its close connection to partial differential equation on hypergraph field. Specifically, in this chapter, the un-normalized hypergraph p-Laplacian based semi-supervised learning methods will be developed based on the un-normalized hypergraph p-Laplacian operator definition such as the curvature operator of hypergraph (i.e., the un-normalized hypergraph 1-Laplacian operator). Then these un-normalized hypergraph p-Laplacian based semi-supervised learning methods will be applied to the zoo dataset and the tiny version of 20 newsgroups dataset. Please note that the un-normalized hypergraph p-Laplacian based semi-supervised learning method is the generalization of the un-normalized hypergraph Laplacian based semi-supervised learning method.

We will organize the chapter as follows: Section 3.2 will introduce the preliminary notations and definitions used in this chapter. Section 3.3 will introduce the definitions of the gradient and divergence operators of hypergraph. Section 3.4 will introduce the definition of Laplace operator of hypergraph and its properties. Section 3.5 will introduce the definition of the curvature operator of hypergraph and its properties. Section 3.6 will introduce the definition of the p-Laplace operator of hypergraph and its properties. Section 3.7 will show how to derive the algorithm of the un-normalized hypergraph p-Laplacian based semi-supervised learning method from regularization framework. In section 3.8, we will compare the accuracy performance measures of the un-normalized hypergraph Laplacian based semi-supervised learning algorithm (i.e., the current state of art method) and the un-normalized hypergraph p-Laplacian based semi-supervised learning algorithms. Section 3.9 will conclude this chapter and the future direction of researches utilizing discrete operator of hypergraph will be discussed.

3.2 Preliminary notations and definitions

Given a hypergraph $G = (V, E)$, where V is the set of vertices and E is the set of hyper-edges. Each hyper-edge $e \in E$ is the subset of V. Please note that the cardinality of e is greater than or equal two. In the other words, $|e| \geq 2$, for every $e \in E$. Let $w(e)$ be the weight of the hyper-edge e. Then W will be the $R^{|E|*|E|}$ diagonal matrix containing the weights of all hyper-edges in its diagonal entries.

The incidence matrix H of G is a $R^{|V|*|E|}$ matrix that can be defined as follows

$$h(v,e) = \begin{cases} 1, & \text{if vertex } v \text{ belongs to hyperedge } e \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

From the above definition, we can define the degree of vertex v and the degree of hyper-edge e as follows

$$d(v) = \sum_{e \in E} w(e) * h(v,e) \tag{3.2}$$

$$d(e) = \sum_{v \in V} h(v,e) \tag{3.3}$$

Let D_v and D_e be two diagonal matrices containing the degrees of vertices and the degrees of hyper-edges in their diagonal entries respectively. Please note that D_v is the $R^{|V|*|V|}$ matrix and D_e is the $R^{|E|*|E|}$ matrix.

Please note that, we assume that the weight of each hyper-edge is 1.

The inner product on the function space R^V is

$$< f, g >_V = \sum_{u \in V} f_u g_u \tag{3.4}$$

Also define an inner product on the space of functions R^E on the edges

$$< F, G >_E = \sum_e \sum_{(u,v) \subset e} F_{uv} G_{uv} \tag{3.5}$$

Here let $H(V) = (R^V, < ., . >_V)$ and $H(E) = (R^E, < ., . >_E)$ be the Hilbert space real-valued functions defined on the vertices of the hypergraph G and the Hilbert space of real-valued functions defined in the edges of G respectively.

3.3 Gradient and Divergence Operators

We define the gradient operator $d : H(V) \to H(E)$ to be (for each hyper-edge e)

$$(df)_{uv} = \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (f_v - f_u) \tag{3.6}$$

where $f : V \to R$ be a function of $H(V)$.

We define the divergence operator $div : H(E) \to H(V)$ to be

$$< df, F >_{H(E)} = < f, -divF >_{H(V)} \tag{3.7}$$

where $f \in H(V), F \in H(E)$

Next, we need to prove that

$$(divF)_v = \sum_{e \in E} \sum_u \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (F_{vu} - F_{uv})$$

Proof:

$$< df, F >= \sum_e \sum_{(u,v) \subset e} df_{uv} F_{uv}$$

$$= \sum_{e} \sum_{(u,v)\subset e} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} f_v F_{uv} - \sum_{e} \sum_{(u,v)\subset e} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} f_u F_{uv}$$

$$= \sum_{k\in V} \sum_{e} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(k,e)} f_k F_{uk} - \sum_{k\in V} \sum_{e} \sum_{v} \sqrt{\frac{w(e)}{d(e)} h(k,e)\, h(v,e)} f_k F_{kv}$$

$$= \sum_{k\in V} \sum_{e} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(k,e)} f_k F_{uk} - \sum_{k\in V} \sum_{e} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(k,e)\, h(u,e)} f_k F_{ku}$$

$$= \sum_{k\in V} f_k \sum_{e\in E} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(k,e)} (F_{uk} - F_{ku})$$

Thus, we have

$$(divF)_v = \sum_{e\in E} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} (F_{vu} - F_{uv}) \tag{3.8}$$

3.4 Laplace operator

We define the Laplace operator $\Delta : H(V) \to H(V)$ to be

$$\Delta f = -\frac{1}{2} div(df) \tag{3.9}$$

Next, we compute

$$(\Delta f)_v = \frac{1}{2} \sum_{e\in E} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} (df_{uv} - df_{vu})$$

$$= \frac{1}{2} \sum_{e\in E} \sum_{u} \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} \left(\sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} (f_v - f_u) - \sqrt{\frac{w(e)}{d(e)} h(u,e)\, h(v,e)} (f_u - f_v) \right)$$

$$= \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,(f_v - f_u)$$

$$= \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_v - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u$$

$$= f_v\sum_{e\in E}\frac{w(e)}{d(e)}h(v,e)\sum_{u}h(u,e) - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u$$

$$= f_v\sum_{e\in E}\frac{w(e)}{d(e)}h(v,e)\,d(e) - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u$$

$$= f_v\sum_{e\in E}w(e)\,h(v,e) - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u$$

$$= f_v d(v) - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u$$

Thus, we have

$$(\Delta f)_v = f_v d(v) - \sum_{e\in E}\sum_{u}\frac{w(e)}{d(e)}h(u,e)\,h(v,e)\,f_u \tag{3.10}$$

The hypergraph Laplacian is a linear operator. Furthermore, the hypergraph Laplacian is self-adjoint and positive semi-definite.

Let $S_2(f) = \frac{1}{2}\sum_i ||d_i f||^2$, we have the following theorem 1

Theorem 1:

$$D_f S_2 = \Delta f \tag{3.11}$$

3.5 Curvature operator

We define the curvature operator $\kappa : H(V) \to H(V)$ to be

$$\kappa f = -\frac{1}{2} div(\frac{df}{||df||}) \tag{3.12}$$

Next, we compute

$$(\kappa f)_v = \frac{1}{2}\sum_{e\in E}\sum_u \sqrt{\frac{w(e)}{d(e)}h(u,e)h(v,e)}((\frac{df}{||df||})_{uv} - (\frac{df}{||df||})_{vu})$$

$$= \frac{1}{2}\sum_{e\in E}\sum \sqrt{\frac{w(e)}{d(e)}h(u,e)h(v,e)}(\frac{1}{||d_u f||}\sqrt{\frac{w(e)}{d(e)}h(u,e)h(v,e)}(f_v - f_u) - \frac{1}{||d_v f||}\sqrt{\frac{w(e)}{d(e)}h(u,e)h(v,e)}(f_u - f_v))$$

$$= \frac{1}{2}\sum_{e\in E}\sum_u \frac{w(e)}{d(e)}h(u,e)h(v,e)(\frac{1}{||d_u f||} + \frac{1}{||d_v f||})(f_v - f_u)$$

Thus, we have

$$(\kappa f)_v = \frac{1}{2}\sum_{e\in E}\sum_u \frac{w(e)}{d(e)}h(u,e)h(v,e)(\frac{1}{||d_u f||} + \frac{1}{||d_v f||})(f_v - f_u) \tag{3.13}$$

From the above formula, we have

$$d_u f = ((df)_{uv} : (u,v) \subset e, \forall e)^T \tag{3.14}$$

The local variation of f at u is defined to be

$$||d_u f|| = \sqrt{\sum_e \sum_{(u,v)\subset e} (df)^2_{uv}} = \sqrt{\sum_e \sum_{(u,v)\subset e} \frac{w(e)}{d(e)}h(u,e)h(v,e)(f_v - f_u)^2} \tag{3.15}$$

To avoid the zero denominators in (3.13), the local variation of f at u is defined to be

$$||d_u f|| = \sqrt{\sum_e \sum_{(u,v)\subset e} (df)^2_{uv} + \varepsilon} \tag{3.16}$$

where $\varepsilon = 10^{-10}$.

The hypergraph curvature is a non-linear operator.
Let $S_1(f) = \sum_u ||d_u f||$, we have the following theorem 2
Theorem 2:

$$D_f S_1 = \kappa f \tag{3.17}$$

3.6 p-Laplace operator

We define the p-Laplace operator $\Delta_p : H(V) \to H(V)$ to be

$$\Delta_p f = -\frac{1}{2} div(||df||^{p-2} df) \tag{3.18}$$

Clearly, $\Delta_1 = \kappa$ and $\Delta_2 = \Delta$. Next, we compute

$$(\Delta_p f)_v == \frac{1}{2} \sum_{e\in E} \sum_u \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (||df||^{p-2} df_{uv} - ||df||^{p-2} df_{vu})$$

$$= \frac{1}{2} \sum_{e\in E} \sum_u \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (||d_u f||^{p-2} \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (f_v - f_u)$$

$$-||d_v f||^{p-2} \sqrt{\frac{w(e)}{d(e)} h(u,e) h(v,e)} (f_u - f_v))$$

$$\frac{1}{2} \sum_{e\in E} \sum_u \frac{w(e)}{d(e)} h(u,e) h(v,e) (||d_u f||^{p-2} + ||d_v f||^{p-2}) (f_v - f_u)$$

Thus, we have

$$(\Delta_p f)_v = \frac{1}{2} \sum_{e\in E} \sum_u \frac{w(e)}{d(e)} h(u,e) h(v,e) (||d_u f||^{p-2} + ||d_v f||^{p-2}) (f_v - f_u) \tag{3.19}$$

Let $S_p(f) = \frac{1}{p} \sum_i ||d_i f||^p$, we have the following theorem 3
Theorem 3:

$$D_f S_p = \Delta_p f \tag{3.20}$$

3.7 Discrete regularization on hypergraphs and classification problems

Given a hypergraph $G = (V, E)$, where V is the set of vertices and E is the set of hyper-edges. Each hyper-edge $e \in E$ is the subset of V. Let y denote the initial function in $H(V)$. y_i can be defined as follows

$$y_i = \begin{cases} 1, & \text{if sample } i \text{ belongs to the class} \\ -1, & \text{if sample } i \text{ does not belong to the class} \\ 0, & \text{otherwise} \end{cases}$$

Our goal is to look for an estimated function f in $H(V)$ such that f is not only smooth on G but also close enough to an initial function y. Then each sample i is classified as $sign(f_i)$. This concept can be formulated as the following optimization problem

$$argmin_{f \in H(V)} S_p(f) + \frac{\mu}{2} ||f - y||^2 \tag{3.21}$$

The first term in (3.21) is the smoothness term. The second term is the fitting term. A positive parameter μ captures the trade-off between these two competing terms.

3.7.1 2-smoothness

When $p = 2$, the optimization problem (3.21) is

$$argmin_{f \in H(V)} \frac{1}{2} \sum_i ||d_i f||^2 + \frac{\mu}{2} ||f - y||^2 \tag{3.22}$$

By theorem 1, we have

Theorem 4: The solution of (3.22) satisfies

$$\Delta f + \mu(f - y) = 0 \tag{3.23}$$

Since Δ is a linear operator, the closed form solution of (3.23) is

$$f = \mu(\Delta + \mu I)^{-1} y \tag{3.24}$$

Where I is the identity operator and $\Delta = D_v - HWD_e^{-1}H^T$. (3.24) is the algorithm proposed by [18, 19].

3.7.2 1-smoothness

When $p = 1$, the optimization problem (3.21) is

$$argmin_{f \in H(V)} \sum_i ||d_i f|| + \frac{\mu}{2}||f - y||^2 \tag{3.25}$$

By theorem 2, we have

Theorem 5: The solution of (3.25) satisfies

$$\kappa f + \mu\,(f - y) = 0 \tag{3.26}$$

The curvature κ is a non-linear operator; hence we do not have the closed form solution of equation (3.25). Thus, we have to construct iterative algorithm to obtain the solution. From (3.26), we have

$$\frac{1}{2}\sum_{e\in E}\sum_u \frac{w(e)}{d(e)} h(u,e)\,h(v,e)\,(\frac{1}{||d_u f||} + \frac{1}{||d_v f||})\,(f_v - f_u) + \mu\,(f_v - y_v) = 0 \tag{3.27}$$

Define the function $m : E \to R$ by

$$m_{uv} = \frac{1}{2}\sum_{e\in E} \frac{w(e)}{d(e)} h(u,e)\,h(v,e)\,(\frac{1}{||d_u f||} + \frac{1}{||d_v f||}) \tag{3.28}$$

Then equation (3.27) which is

$$\sum_u m_{uv}(f_v - f_u) + \mu\,(f_v - y_v) = 0$$

can be transformed into

$$(\sum_u m_{uv} + \mu) f_v = \sum_u m_{uv} f_u + \mu y_v \tag{3.29}$$

Define the function $p : E \to R$ by

$$p_{uv} = \begin{cases} \frac{m_{uv}}{\sum_u m_{uv}+\mu}, & \text{if } u \neq v \\ \frac{\mu}{\sum_u m_{uv}+\mu}, & \text{if } u = v \end{cases} \tag{3.30}$$

Then

$$f_v = \sum_u p_{uv} f_u + p_{vv} y_v \tag{3.31}$$

Thus, we can consider the iteration

$$f_v^{(t+1)} = \sum_u p_{uv}^{(t)} f_u^{(t)} + p_{vv}^{(t)} y_v, \forall v \in V$$

to obtain the solution of (3.25).

3.7.3 p-smoothness

For any number p, the optimization problem (3.21) is

$$argmin_{f \in H(V)} \frac{1}{p} \sum_i ||d_i f||^p + \frac{\mu}{2} ||f - y||^2 \tag{3.32}$$

By theorem 3, we have

Theorem 6: The solution of (3.32) satisfies

$$\Delta_p f + \mu (f - y) = 0 \tag{3.33}$$

The $p-Laplace$ operator is a non-linear operator; hence we do not have the closed form solution of equation (3.33). Thus, we have to construct iterative algorithm to obtain the solution. From (3.33), we have

$$\frac{1}{2} \sum_{e \in E} \sum_u \frac{w(e)}{d(e)} h(u,e) h(v,e) \left(||d_u f||^{p-2} + ||d_v f||^{p-2} \right) (f_v - f_u) + \mu (f_v - y_v) = 0$$

Define the function $m : E \to R$ by

$$m_{uv} = \frac{1}{2} \sum_{e \in E} \sum_u \frac{w(e)}{d(e)} h(u,e) h(v,e) \left(||d_u f||^{p-2} + ||d_v f||^{p-2} \right)$$

Then equation (3.32) which is

$$\sum_u m_{uv}(f_v - f_u) + \mu\,(f_v - y_v) = 0$$

can be transformed into

$$(\sum_u m_{uv} + \mu) f_v = \sum_u m_{uv} f_u + \mu y_v \tag{3.34}$$

Define the function $p : E \to R$ by

$$p_{uv} = \begin{cases} \frac{m_{uv}}{\sum_u m_{uv}+\mu}, & \text{if } u \neq v \\ \frac{\mu}{\sum_u m_{uv}+\mu}, & \text{if } u = v \end{cases} \tag{3.35}$$

Then

$$f_v = \sum_u p_{uv} f_u + p_{vv} y_v \tag{3.36}$$

Thus, we can consider the iteration

$$f_v^{(t+1)} = \sum_u p_{uv}^{(t)} f_u^{(t)} + p_{vv}^{(t)} y_v, \forall v \in V \tag{3.37}$$

to obtain the solution of (3.32).

3.8 Experiments and results

Datasets

In this chapter, we used the zoo dataset and the tiny version of 20 newsgroups dataset which can be obtained from UCI repository and from http://www.cs.nyu.edu/~roweis/data.html respectively.

The zoo data set contains 101 animals with 17 attributes. The attributes include hair, feathers, eggs, milk, etc. The animals have been classified into 7 different classes. In this dataset, each attribute is the hyper-edge.

The tiny version of the 20 newsgroups dataset contains the binary occurrence data for 100 words across 16242 postings. However, we just choose small subset of this tiny dataset containing 200 postings in order to test our algorithms. In this dataset, each word is the hyper-edge.

Thus, these two datasets are themselves the hypergraphs and we don't need to preprocess these two datasets. These two input datasets are very similar to the following figure 3.1 of [16, 17].

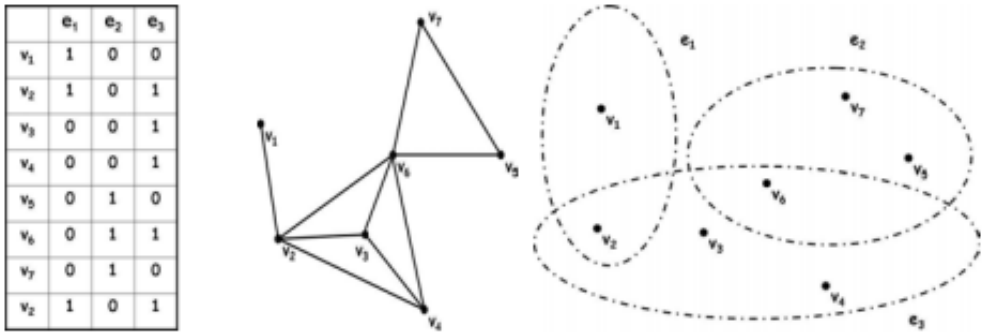

Fig. 3.1 Hypergraph example with 8 vertices and 3 hyper-edges [12,13]

Experiments and Results

In this section, we experiment with the above proposed un-normalized hypergraph p-Laplacian based semi-supervised learning methods with $p = 3, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0$ and the current state of the art method (i.e. the un-normalized hypergraph Laplacian based semi-supervised learning method $p = 2$) applied directly to the zoo dataset and the tiny version of 20 newsgroups dataset in terms of accuracy performance measure. The accuracy performance measure Q is given as follows

$$Q = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative}$$

All experiments were implemented in MATLAB 6.5 on virtual machine. The accuracy performance measures of the above proposed methods are given in the following table 3.1 and table 3.2.

The following figure 3.2 shows the accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method $p = 2$ for the zoo dataset:

The following figure 3.3 shows the accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method $p = 2$ for the tiny version of the 20 newsgroups dataset:

Table 3.1 Accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method $p = 2$ for the zoo dataset

p values	Accuracy performance measures (%)
2	85.71
3	88.80
3.1	90.48
3.2	91.04
3.3	91.60
3.4	92.16
3.5	92.44
3.6	92.44
3.7	92.16
3.8	90.20
3.9	89.64
4.0	88.52

Table 3.2 Accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method p=2 for the tiny version of 20 newsgroups dataset

p values	Accuracy performance measures (%)
2	75.50
3	78.25
3.1	80.00
3.2	81.25
3.3	81.75
3.4	82.25
3.5	82.00
3.6	82.50
3.7	83.50
3.8	83.75
3.9	82.75
4.0	82.75

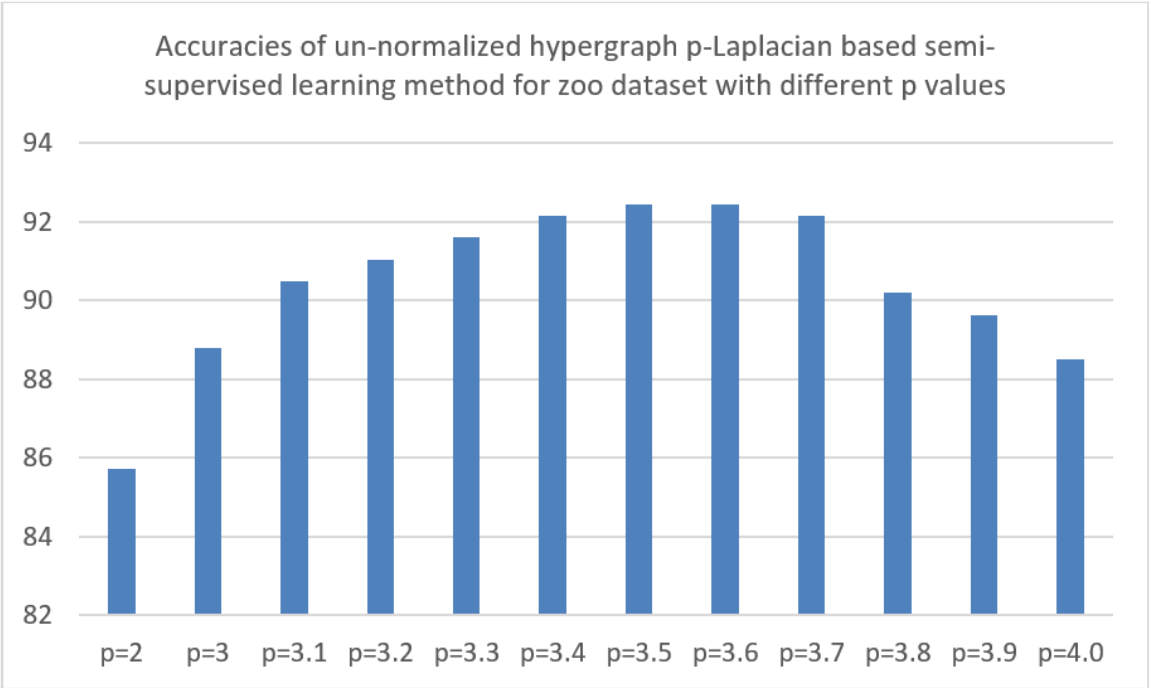

Fig. 3.2 The accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method p=2 for the zoo dataset

From the above tables, we easily recognized that the un-normalized hypergraph p-Laplacian based semi-supervised learning methods outperform the current state of art method. The results from the above tables show that the un-normalized hypergraph p-Laplacian based semi-supervised learning methods are at least as good as the current state of the art method ($p = 2$) but often lead to significant better classification accuracy performance measures.

3.9 Conclusions

In this chapter, we have proposed the detailed algorithms of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods applied to the zoo dataset and the tiny version of 20 newsgroups dataset. Interestingly, experiments show that the un-normalized hypergraph p-Laplacian based semi-supervised learning methods are at least as good as the un-normalized hypergraph Laplacian based semi-supervised learning method (the current state of the art method $p = 2$) but often lead to significant better classification accuracy performance measures.

To the best of our knowledge, the un-normalized hypergraph p-Laplacian Eigenmaps and the un-normalized hypergraph p-Laplacian clustering methods have not yet been developed.

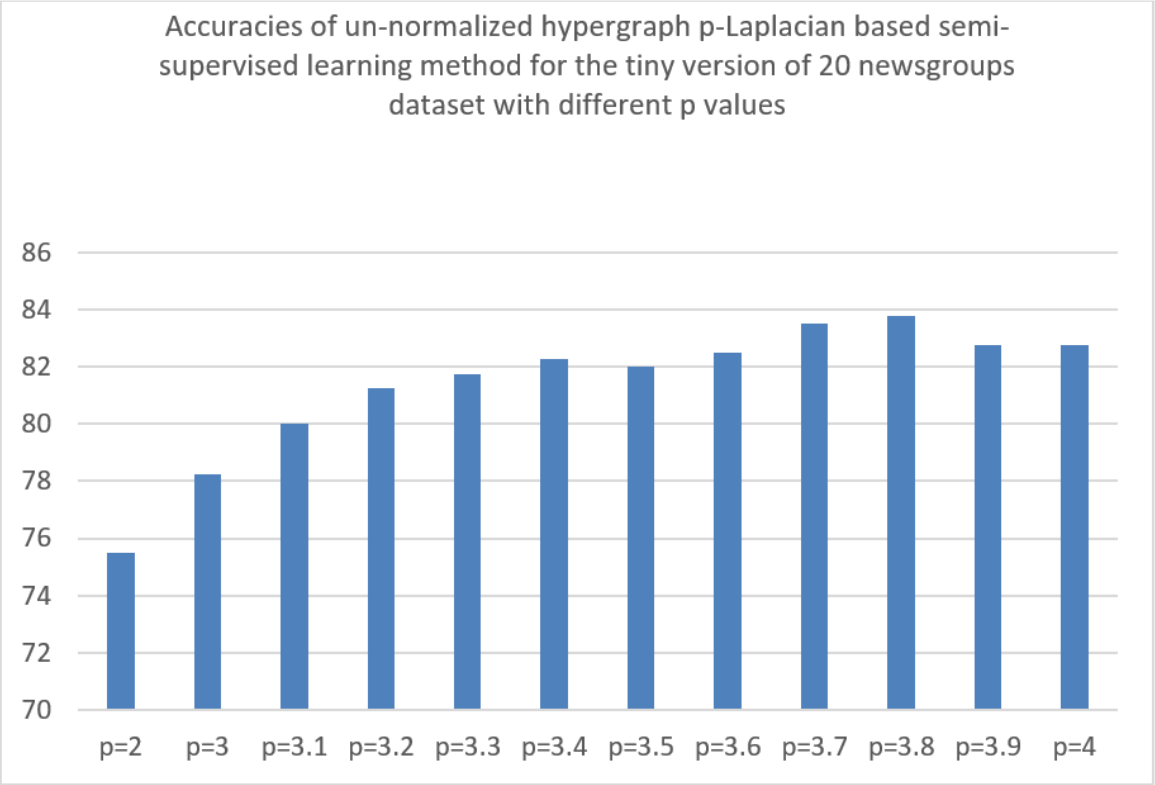

Fig. 3.3 The accuracies of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods and the current state of the art hypergraph Laplacian based semi-supervised learning method p=2 for the tiny version of the 20 newsgroups dataset

These methods are worth investigated because of their difficult nature and their close connection to partial differential equation on hypergraph field. In the future, we will develop the un-normalized hypergraph p-Laplacian Eigenmaps and the un-normalized hypergraph p-Laplacian clustering methods by constructing the hypergraph p-Laplacian matrix.Then from this hypergraph p-Laplacian matrix, we can also construct another version of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods.

References

[1] J. Labiak and K. Livescu, "Nearest neighbors with learned distances for phonetic frame classification," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[2] B. Schwikowski, P. Uetz, and S. Fields, "A network of protein–protein interactions in yeast," *Nature biotechnology*, vol. 18, no. 12, pp. 1257–1261, 2000.

[3] A. Marshall, "Artificial neural network for speech recognition," *2nd Annual Student Research Showcase*, 2005.

[4] L. Shi, Y. Cho, and A. Zhang, "Prediction of protein function from connectivity of protein interaction networks," *International Journal of Computational Bioscience*, vol. 1, no. 1, pp. 210–1009, 2010.

[5] A. Ganapathiraju, *Support vector machines for speech recognition*. Mississippi State University, 2002.

[6] G. R. Lanckriet, M. Deng, N. Cristianini, M. I. Jordan, and W. S. Noble, "Kernel-based data fusion and its application to protein function prediction in yeast," in *Biocomputing 2004*, pp. 300–311, World Scientific, 2003.

[7] K. Tsuda, H. Shin, and B. Schölkopf, "Fast protein classification with multiple networks," *Bioinformatics*, vol. 21, no. suppl_2, pp. ii59–ii65, 2005.

[8] H. Shin, K. Tsuda, and B. Schölkopf, "Protein functional class prediction with a combined graph," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3284–3292, 2009.

[9] L. Tran, "Application of three graph laplacian based semi-supervised learning methods to protein function prediction problem," *arXiv preprint arXiv:1211.4289*, 2012.

[10] H. Trang and L. H. Tran, "Graph based semi-supervised learning methods applied to speech recognition problem," in *International Conference on Nature of Computation and Communication*, pp. 264–273, Springer, 2014.

[11] L. H. Tran and L. H. Tran, "Un-normalized graph p-laplacian semi-supervised learning method applied to cancer classification problem," *Journal of Automation and Control Engineering Vol*, vol. 3, no. 1, 2015.

[12] L. Tran and L. Tran, "The un-normalized graph p-laplacian based semi-supervised learning method and speech recognition problem," *International Journal of Advances in Soft Computing & Its Applications*, vol. 9, no. 1, 2017.

[13] L. Tran, "The un-normalized graph p-laplacian based semi-supervised learning method and protein function prediction problem," in *Knowledge and Systems Engineering*, pp. 23–35, Springer, 2014.

[14] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," 2002.

[15] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in neural information processing systems*, pp. 321–328, 2004.

[16] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," 2005.

[17] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," *Advances in neural information processing systems*, vol. 19, pp. 1601–1608, 2006.

[18] L. H. Tran, T. Hoang, and B. H. N. Huynh, "Hypergraph based semi-supervised learning algorithms applied to speech recognition problem: a novel approach," *Journal of Science and Technology, Vietnam Academy of Science and Technology, ISSN: 0866-708X*, 2014.

[19] L. Tran, "Hypergraph and protein function prediction with gene expression data," *arXiv preprint arXiv:1212.0388*, 2012.

Chapter 4

Hypergraph Convolutional Neural Network Based Clustering Technique

4.1 Introduction

Clustering is the very important problem in machine learning and deep learning research areas. It's a technique that separates data points/samples into groups/clusters such that data points/samples are more similar to other data points/samples in the same groups/clusters than those in the other groups/clusters. Its applications are huge such as mobility pattern clustering [1], text clustering [2–5], customer segmentation [6–9], etc.

Let's discuss deeply about the motivations of one specific clustering problem which is the text clustering problem of our company. In details, in this clustering problem, we would like to partition Facebook users into their appropriate groups/clusters (i.e., depending on the contents that they are talking about). For example, there are groups of Facebook users talking about games, there are groups of Facebook users talking about music, there are groups of Facebook users talking about religions, etc.

The applications of this text clustering problem are huge such as:

- Our influencers/streamers can create the "appropriate" contents for their fans/users.
- This text clustering problem will lead to the implementation of the recommendation/matching systems (for example, fans/influencers, fans/contents, etc.). We can employ the bi-partite graph to implement these recommendation/matching systems (i.e. bi-partite graph matching problem).

- FAKE fans detection or anomaly/abnormal detection: for example, the fans/users belong to the group game, but they do not comment/feedback appropriately (i.e. they always talk about music), ...

- etc.

There are various clustering techniques, available from Python sklearn package [10] such as k-means [1], hierarchical clustering technique, affinity propagation, etc., that can be employed to solve these clustering problems. However, please note that these techniques can ONLY be applied to feature datasets.

However, in this chapter, we will just employ k-means clustering technique as **the "vanilla" or baseline technique** because of three main reasons:

- It's very simple to implement.

- It scales to large datasets.

- It guarantees to converge to final solutions.

Moreover, there are other clustering techniques (i.e. belong to different class of clustering techniques) that can also be employed to solve the clustering problem but can ONLY be applied to network datasets such as spectral clustering [1, 11–13], maximum modularity approach [14–20], etc.

The weakness of these two classes of clustering techniques is obviously that they can ONLY be applied to one type of dataset. This will lead to the loss of information. Now, let's consider the case that we have both types of datasets such as the feature dataset and the network dataset. Assume that the samples in both datasets are the same, how can we apply the clustering technique to both datasets?

In this chapter, we will develop the novel clustering method that utilize both the feature dataset and the network dataset.

In details, initially, we will employ the Graph Auto-Encoders proposed by Thomas Kipf [21–25] to transform both the network dataset and the feature dataset from the high dimensional space to low dimensional space. Finally, the k-means clustering technique will be applied to the transformed dataset. This novel clustering technique will be called the graph convolutional neural network based clustering technique.

There are some main advantages of this novel clustering technique such as:

- The memory/space required to store the data is lowered. This will lead to the low space complexity.

- The low space complexity will lead to less computational and training time of the classification/clustering techniques (i.e. the low time complexity).
- The noises and the redundant features are removed in the feature dataset and the network dataset. This will lead to the high performance of the clustering technique. This claim will be shown in the Experiments and Results section
- Both feature dataset and network dataset are utilized. This will lead to no information loss. However, there is one weakness associated with the graph convolutional neural network based clustering technique.

In other words, assuming the pairwise relationships among the objects/entities/samples in this graph representation is not complete. Let's consider the case that we would like to partition/segment a set of articles into different topics [26, 27].

Initially, we employ the graph data structure to represent this dataset. The vertices of the graph are the articles. Two articles are connected by an edge (i.e., the pairwise relationship) if there is at least one author in common. Finally, we can apply clustering technique to this graph to partition/segment the vertices into groups/clusters.

Obviously, we easily see that, in this graph data structure, we **ignore** the information whether **one specific author is the author of three or more articles (i.e., the co-occurrence relationship or the high order relationship).**

This will lead to **the loss of information**. In other words, this will lead to the low performance (i.e., the low accuracy) of the clustering technique.

In order to overcome this difficulty, we try to employ the hypergraph data structure to represent for the above relational dataset. In details, in this hypergraph data structure, the articles are the vertices and the authors are the hyper-edges. This hyper-edge can connect more than two vertices (i.e., articles). Please note that if the feature dataset is only given, we need to construct the hypergraph from the feature dataset. We will discuss more about how to construct the hypergraph from the feature vectors in section 4.3.

In details, initially, we will develop the Hypergraph Auto-Encoders to transform both the hypergraph dataset (i.e., constructed from the feature dataset) and the feature dataset from the high dimensional space to low dimensional space. Finally, the k-means clustering technique will be applied to the transformed dataset. This novel clustering technique will be called the hypergraph convolutional neural network based clustering technique.

Last but not least, please note that these clustering techniques are un-supervised learning techniques. Hence, in this chapter, we do not need labeled datasets.

We will organize the chapter as follow: Section 4.2 will define the problem and will present the novel graph convolutional neural network based clustering technique. Section 4.3

will present the novel hypergraph convolutional neural network based clustering technique. Section 4.4 will describe the two datasets that will be used in this chapter which are the Citeseer dataset and the Cora dataset. Then, section 4.5 will compare the performance the hypergraph convolutional neural network based clustering technique and the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, and the spectral clustering technique testing on these two Citeseer and Cora datasets. Section 4.6 will conclude this chapter and the future directions of researches will be discussed.

4.2 Graph convolutional neural network based clustering technique

4.2.1 Problem formulation

Suppose that we are given a set of samples $x_1, x_2, \ldots, x_n$ where n is the total number of samples and the pre-defined number of clusters k.

In details, we are given the adjacency matrix $A \in R^{n*n}$ where

$$A_{ij} = \begin{cases} 1, & \text{if vertex i is connected to vertex } j \\ 0, & \text{otherwise} \end{cases}$$

Our objective is to output the clusters/groups $A_1, A_2, \ldots, A_k$ where
$A_i = \{j | 1 \leq j \leq n \; and \; j \; belongs \; to \; cluster \; i\}$.

4.2.2 Adjacency matrix is not provided

Suppose that we are given the feature matrix $X \in R^{n*L_1}$ but not the adjacency matrix $A \in R^{n*n}$.

In this case, we can construct the similarity graph from these feature vectors by using the k-nearest neighbor graph.

In other words, sample i is connected with sample j by an edge (no direction: un-directed graph) if sample i is among the k nearest neighbors of sample j or sample j is among the k nearest neighbor of sample i.

We will discuss more about how to construct the similarity graph from the feature vectors in section 4. Finally, we obtain the adjacency matrix A representing for the similarity graph.

$$A_{ij} = \begin{cases} 1, & \text{if sample } i \text{ is connected to sample } j \\ 0, & \text{if sample } i \text{ is not connected to sample } j \end{cases}$$

Please note that this phase is required for spectral clustering technique and graph convolutional neural network based clustering techniques if only the feature matrix is provided.

4.2.3 Graph convolutional neural network based clustering technique

Currently, we have the set of feature vectors $x_1, x_2, \ldots, x_n$ and the adjacency matrix A representing the relationships among the samples in the dataset.

Please note that $x_i \in R^{1*L_1},\ 1 \leq i \leq n$ and $A \in R^{n*n}$.

Let $\hat{A} = A + I$, where I is the identity matrix.

Let $\hat{D}$ be the diagonal degree matrix of $\hat{A}$. In other words,$\hat{D}_{ii} = \sum_j \hat{A}_{ij}$.

The final output (i.e. the embedding matrix) Z of the graph convolutional neural network can be defined as follows

$$Z = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}} ReLU\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}} X \theta_1\right)\theta_2$$

Please note that X is the input feature matrix and $X \in R^{n*L_1}$. $\theta_1 \in R^{L_1*L_2}$and $\theta_2 \in R^{L_2*D}$ are two parameter matrices that are needed to be learned during the training process.

Please note that D is the dimensions of the embedding matrix Z.

Next, we need to reconstruct the adjacency matrix A from Z. We will obtain the reconstruction $A\prime$ representing the similarity graph by using the following formula

$$A' = sigmoid(ZZ^T)$$

ReLU operation can also be called Rectified Linear Unit. It is defined as follow

$$ReLU\,(x) = \max(0, x)$$

A *sigmoid* function can be defined as follow

$$sigmoid\,(x) = \frac{1}{1+e^{-x}}$$

For this graph auto-encoder model, we need to evaluate the cross-entropy error over all samples in the dataset:

$$L = -\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}\ln\left(sigmoid\left(z_i z_j^T\right)\right) + (1-A_{ij})\ln\left(1-sigmoid\left(z_i z_j^T\right)\right)$$

Please note that z_i is the row i vector of the embedding matrix Z and $z_i \in R^{1*D}$.

The two parameter matrices $\theta_1 \in R^{L_1*L_2}$ and $\theta_2 \in R^{L_2*D}$ are trained by using the gradient descent method.

Finally, partition the samples $(z_i)_{i=1,\ldots,n}$ in R^{1*D} with the k-means algorithm into k clusters/groups.

4.2.4 Discussions about graph convolutional neural network based clustering technique

From the above section 4.2.3, unlike other clustering techniques such as k-means, we easily see that this proposed clustering technique (i.e., the graph convolutional neural network based clustering technique) utilizes both the feature dataset and the network dataset. This is the very strong argument of this proposed technique. Since no information are lost, the performance of this novel clustering technique is expected to be higher than the performances of the other classic clustering techniques such as k-means. This claim will be shown in the Experiments and Results section. However, there is one major weakness of this proposed clustering technique. When new samples arrive, we cannot predict which clusters these samples belong to (unlike k-means clustering technique). In other words, we have to update the adjacency matrix, we have to re-train our graph auto-encoder, etc. Thus, this technique can only be considered as the offline clustering technique although its performance is a lot higher than the performance of the other online clustering techniques.

4.3 Hypergraph convolutional neural network based clustering technique

4.3.1 Problem formulation

Suppose that we are given a set of samples $x_1, x_2, \ldots, x_n$ where n is the total number of samples and the pre-defined number of clusters k.

In details, we are given the incidence matrix $H \in R^{n*n}$ where

$$H_{ij} = \begin{cases} 1, & \text{if vertex } i \text{ belongs to hyperedge } j \\ 0, & \text{otherwise} \end{cases}$$

and the feature matrix $X \in R^{n*L_1}$ where L_1 is the dimensions of the feature vectors.

Our objective is to output the clusters/groups $A_1, A_2, \ldots, A_k$ where
$A_i = \{j | 1 \leq j \leq n \; and \; j \; belongs \; to \; cluster \; i\}$.

4.3.2 Incidence matrix of the hypergraph is not provided

Suppose that we are given the feature matrix $X \in R^{n*L_1}$ but not the incidence matrix $H \in R^{n*n}$.

In this case, we can construct the incidence matrix H from these feature vectors by using the k-nearest neighbor graph.

In other words, sample i belongs to hyperedge j if sample i is among the k nearest neighbors of sample j or sample j is among the k nearest neighbor of sample i. In this chapter, k is set to be 5.

Finally, we obtain the incidence matrix H representing for the hypergraph

$$H_{ij} = \begin{cases} 1, & \text{if sample } i \text{ is connected to sample } j \\ 0, & \text{if sample } i \text{ is not connected to sample } j \end{cases}$$

Please note that this phase is required for hypergraph convolutional neural network based clustering techniques if only the feature matrix is provided.

4.3.3 Hypergraph convolutional neural network based clustering technique

Currently, we have the set of feature vectors $x_1, x_2, \ldots, x_n$ and the incidence matrix H of the hypergraph.

Please note that $x_i \in R^{1*L_1},\ 1 \leq i \leq n$ and $H \in R^{n*n}$.

Let $w(e)$ be the weight of the hyper-edge e. Then W will be the R^{n*n} diagonal matrix containing the weights of all hyper-edges in its diagonal entries.

From the incidence matrix H and the weight matrix W, we can define the degree of vertex v and the degree of hyper-edge e as follows

$$d(v) = \sum_{e \in E} w(e) * h(v,e)$$

$$d(e) = \sum_{v \in V} h(v,e)$$

Let D_v and D_e be two diagonal matrices containing the degrees of vertices and the degrees of hyper-edges in their diagonal entries respectively. Please note that D_v is the R^{n*n} matrix and D_e is the R^{n*n} matrix.

The final output (i.e., the embedding matrix) Z of the hypergraph convolutional neural network can be defined as follows

$$Z = D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} ReLU\left(D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} X \theta_1\right) \theta_2$$

Please note that X is the input feature matrix and $X \in R^{n*L_1}$.

$\theta_1 \in R^{L_1*L_2}$ and $\theta_2 \in R^{L_2*D}$ are two parameter matrices that are needed to be learned during the training process.

Please note that D is the dimensions of the embedding matrix Z.

Next, we need to reconstruct the incidence matrix H from Z. We will obtain the reconstruction $H\prime$ representing the hypergraph by using the following formula

$$H' = sigmoid(ZZ^T)$$

ReLU operation can also be called Rectified Linear Unit. It is defined as follow

$$ReLU\,(x) = \max(0, x)$$

A *sigmoid* function can be defined as follow

$$sigmoid\,(x) = \frac{1}{1+e^{-x}}$$

For this hypergraph auto-encoder model, we need to evaluate the cross-entropy error over all samples in the dataset:

$$L = -\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} H_{ij}\ln\left(sigmoid\left(z_i z_j^T\right)\right) + (1-H_{ij})\ln\left(1-sigmoid\left(z_i z_j^T\right)\right)$$

Please note that z_i is the row i vector of the embedding matrix Z and $z_i \in R^{1*D}$.

The two parameter matrices $\theta_1 \in R^{L_1*L_2}$ and $\theta_2 \in R^{L_2*D}$ are trained by using the gradient descent method.

Finally, partition the samples $(z_i)_{i=1,\ldots,n}$ in R^{1*D} with the k-means algorithm into k clusters/groups.

4.3.4 Discussions about hypergraph convolutional neural network based clustering technique

From the above section 4.3.3, unlike the graph convolutional neural network based clustering techniques, instead of employing the pairwise relationships among objects/entities/samples, we easily see that this hypergraph convolutional neural network based clustering technique employs the high order relationship among objects/entities/samples. This will lead to no loss of information. Hence the performance of this hypergraph convolutional neural network based clustering technique is expected to be higher than the performance of the graph convolutional neural network based clustering technique. This claim will be shown in the Experiments and Results section. However, like the graph convolutional neural network based clustering technique, this hypergraph convolutional neural network based clustering technique is the offline clustering technique. In other words, when new samples arrive, we have to update the incidence matrix of the hypergraph, we have to re-train our hypergraph auto-encoder, etc.

4.4 Experiments and Results

4.4.1 Dataset descriptions

In this chapter, we will use two datasets which are the Citeseer dataset and the Cora dataset to test our novel clustering technique (i.e. the graph convolutional neural network based clustering technique).

Cora: This dataset consists of 2,708 scientific publications classified into one of seven classes which are Case_Based, Genetic_Algorithms, Neural_Networks, Probabilistic_Methods, Reinforce_Learning, Rule_Learning, and Theory. The citation network consists of 5,429 links. In other words, this Cora citation network contains 2,708 nodes (i.e., scientific publications) and 5,429 edges (i.e., citation links). Each publication in this Cora dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary contains 1,433 unique words. In other words, we have the $R^{2708*1433}$ feature matrix.

Citeseer: This dataset consists of 3,312 scientific publications classified into one of six classes which are Agents, AI, DB, IR, ML, and HCI. The citation network consists of 4,732 links. In other words, this Citeseer citation network contains 3,312 nodes (i.e., scientific publications) and 4,732 edges (i.e., citation links). Each publication in the Citeseer dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary contains 3,703 unique words. In other words, we have the $R^{3312*3703}$ feature matrix.

Please note that **in the case that the adjacency matrices are not given**, we need to construct the similarity graph from the feature vectors of the datasets like the following ways:

- The ε-neighborhood graph: Connect all the samples whose pairwise distances are smaller than ε.
- k-nearest neighbor graph: Sample i is connected with sample j by an edge (no direction: un-directed graph) if sample i is among the k nearest neighbors of sample j or sample j is among the k nearest neighbor of sample i.
- The fully connected graph: All samples are connected.

In this chapter, the k-nearest neighbor graph is employed to construct the similarity graph from the feature vectors of the Cora dataset and the Citeseer dataset. Please note that k is chosen to be 5.

Last but not least, the way showing how to construct the hypergraph (i.e., the incidence matrix H) from the feature vectors are discussed in detailed in section 4.3.

Please note that D, the dimensions of the embedding matrix Z, is set to be 16.

4.4.2 Experimental results

In this section, we will try to compare the performance of the hypergraph convolutional neural network based clustering technique with the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, the spectral clustering technique for feature datasets, the spectral clustering technique for network datasets. We test our models (Python code) on Google Colab with NVIDIA Tesla K80 GPU and 12 GB RAM.

There are three performances of clustering techniques that we are going to employ in this chapter which are:

- Silhouette coefficient.
- Davies-Bouldin score.
- Calinski-Harabasz score.

The Silhouette coefficient is defined for each sample and is composed of two score:

- a: The mean distance between the sample and all other points/samples in the same groups/clusters.
- b: The mean distance between the sample and all other points/samples in the next nearest cluster.

Then the Silhouette coefficient s for a single sample can be computed as follow

$$s = \frac{b-a}{\max(a,b)}$$

The Silhouette coefficient of the dataset is the mean of the Silhouette coefficient for all samples.

From the above formula, we easily recognize that **the higher the Silhouette coefficient, the better the clustering results.**

The Davies-Bouldin score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score.

Table 4.1 Citeseer dataset: Comparison of graph convolutional neural network based clustering technique with the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix.

	Silhouette coefficient	Davies-Bouldin score	Calinski-Harabasz score
The hypergraph convolutional neural network based clustering technique	**0.5034**	**0.4786**	**40369.4615**
The graph convolutional neural network based clustering technique	0.1203	2.0923	231.2733
The k-means clustering technique	0.0007	7.3090	15.2688
The spectral clustering technique for feature vectors	0.0047	5.9004	4.1290
The spectral clustering technique for adjacency matrix	-0.0081	14.8230	2.1392

The minimum score is zero. From the above definition, we easily see that **the lower the Davies-Bouldin score, the better the clustering results.**

The Calinski-Harabasz score is also known as the Variance Ratio Criterion. This score is defined as the ratio of the sum of **between-clusters dispersion** and of **inter-cluster dispersion** for all clusters. From the definition, we easily see that **the higher the Calinski-Harabasz score, the better the clustering results.**

For the Citeseer dataset, the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix are given in the following table 4.1:

For the Cora dataset, the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, the spectral clustering technique

Table 4.2 Citeseer dataset: Comparison of graph convolutional neural network based clustering technique with the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix.

	Silhouette coefficient	Davies-Bouldin score	Calinski-Harabasz score
The hypergraph convolutional neural network based clustering technique	**0.3349**	**1.0471**	**1288.7860**
The graph convolutional neural network based clustering technique	0.1963	1.7312	322.3034
The k-means clustering technique	0.0149	5.5878	21.5432
The spectral clustering technique for feature vectors	-0.0196	6.6584	20.4136
The spectral clustering technique for adjacency matrix	-0.0465	8.9685	2.0282

for feature vectors, the spectral clustering technique for adjacency matrix are given in the following table 4.2:

4.4.3 Discussions

From the above table 4.1 and table 4.2, we easily recognize that the graph convolutional neural network based clustering technique is significantly better than the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix since graph convolutional neural network based clustering technique utilize both the information from the feature vectors and the adjacency matrix of the dataset and noises and redundant features in the dataset (i.e., both in the feature vectors and the adjacency matrix) are removed.

Moreover, the hypergraph convolutional neural network based clustering technique is better than the graph convolutional neural network based clustering technique since the hypergraph data structure employs the high order relationships among the samples/entities/objects. This will lead to no loss of information.

However, when new samples arrive, for the (hyper)-graph convolutional neural network based clustering techniques, we cannot predict which clusters these samples belong to. In other words, we have to update the adjacency matrix or the incidence matrix, re-train our (hyper)-graph auto-encoder, re-compute the embedding matrix, and apply k-means clustering technique to this embedding matrix again to get the final clustering result. Hence the (hyper)-graph convolutional neural network based clustering technique can only be considered as the offline clustering technique.

Last but not least, please note that when new samples/data points arrive, we need to re-train our model. For industrial projects, this novel model can be updated/re-trained twice per month or once per month.

4.5 Conclusions

In this chapter, our contributions are four folds:

- Develop the novel graph convolutional neural network based clustering technique.
- Develop the novel hypergraph convolutional neural network based clustering technique.
- Apply these novel clustering techniques to two Citeseer and Cora datasets.
- Compare the performance of the hypergraph convolutional neural network based clustering technique with the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix.

In this chapter, we propose and employ the (hyper)-graph convolutional neural network based clustering technique to solve the clustering problem. To the best of our knowledge, this work is not complete. There are various types of (hyper)-graph convolutional neural networks. In the future, we will employ the (hyper)-graph convolutional neural network with/without attention [28, 29] to solve this clustering problem.

References

[1] L. H. Tran and L. H. Tran, "Mobility patterns based clustering: A novel approach," *International Journal of Machine Learning and Computing*, vol. 8, no. 4, 2018.

[2] J. Yi, Y. Zhang, X. Zhao, and J. Wan, "A novel text clustering approach using deep-learning vocabulary network," *Mathematical Problems in Engineering*, vol. 2017, 2017.

[3] A. Hadifar, L. Sterckx, T. Demeester, and C. Develder, "A self-training approach for short text clustering," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pp. 194–199, 2019.

[4] Y.-l. Liu and X.-j. Wen, "A short text clustering method based on deep neural network model," *Journal of Computers*, vol. 29, no. 6, pp. 90–95, 2018.

[5] Z. Dai, K. Li, H. Li, and X. Li, "An unsupervised learning short text clustering method," in *Journal of Physics: Conference Series*, vol. 1650, p. 032090, IOP Publishing, 2020.

[6] C. Marcus, "A practical yet meaningful approach to customer segmentation," *Journal of consumer marketing*, 1998.

[7] K. K. Tsiptsis and A. Chorianopoulos, *Data mining techniques in CRM: inside customer segmentation*. John Wiley & Sons, 2011.

[8] J. Wu and Z. Lin, "Research on customer segmentation model by clustering," in *Proceedings of the 7th international conference on Electronic commerce*, pp. 316–318, 2005.

[9] J.-J. Jonker, N. Piersma, and D. Van den Poel, "Joint optimization of customer segmentation and marketing policy to maximize long-term profitability," *Expert Systems with Applications*, vol. 27, no. 2, pp. 159–168, 2004.

[10] "2.3. clustering — scikit-learn 1.0.1 documentation." https://scikit-learn.org/stable/modules/clustering.html.

[11] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[12] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, pp. 849–856, 2002.

[13] L. H. Tran, L. H. Tran, and H. Trang, "Un-normlized and random walk hypergraph laplacian un-supervised learning," in *International Conference on Nature of Computation and Communication*, pp. 254–263, Springer, 2014.

[14] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Generalized louvain method for community detection in large networks," in *2011 11th international conference on intelligent systems design and applications*, pp. 88–93, IEEE, 2011.

[15] D. Combe, C. Largeron, M. Géry, and E. Egyed-Zsigmond, "I-louvain: An attributed graph clustering method," in *International Symposium on Intelligent Data Analysis*, pp. 181–192, Springer, 2015.

[16] P. Held, B. Krause, and R. Kruse, "Dynamic clustering in social networks using louvain and infomap method," in *2016 Third European Network Intelligence Conference (ENIC)*, pp. 61–68, IEEE, 2016.

[17] D. L. Sánchez, J. Revuelta, F. De la Prieta, A. B. Gil-González, and C. Dang, "Twitter user clustering based on their preferences and the louvain algorithm," in *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pp. 349–356, Springer, 2016.

[18] S. Wang and R. Koopman, "Clustering articles based on semantic similarity," *Scientometrics*, vol. 111, no. 2, pp. 1017–1031, 2017.

[19] S. Ghosh, M. Halappanavar, A. Tumeo, A. Kalyanaraman, H. Lu, D. Chavarria-Miranda, A. Khan, and A. Gebremedhin, "Distributed louvain algorithm for graph community detection," in *2018 IEEE international parallel and distributed processing symposium (IPDPS)*, pp. 885–895, IEEE, 2018.

[20] S. Emmons, S. Kobourov, M. Gallant, and K. Börner, "Analysis of network clustering algorithms and cluster quality metrics at scale," *PloS one*, vol. 11, no. 7, p. e0159161, 2016.

[21] S. Bhowmick and S. Srinivasan, "A template for parallelizing the louvain method for modularity maximization," in *Dynamics On and Of Complex Networks, Volume 2*, pp. 111–124, Springer, 2013.

[22] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," *arXiv preprint arXiv:1802.04407*, 2018.

[23] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang, "Mgae: Marginalized graph autoencoder for graph clustering," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 889–898, 2017.

[24] S. Fan, X. Wang, C. Shi, E. Lu, K. Lin, and B. Wang, "One2multi graph autoencoder for multi-view graph clustering," in *Proceedings of The Web Conference 2020*, pp. 3070–3076, 2020.

[25] H. L. EIGENMAPS, "Weighted un-normalized hypergraph laplacian eigenmaps for classification problems," *Int. J. Advance Soft Compu. Appl*, vol. 10, no. 3, 2018.

[26] N. T. V. Dang, L. Tran, and L. Tran, "Noise-robust classification with hypergraph neural network," *arXiv preprint arXiv:2102.01934*, 2021.

[27] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," 2005.

[28] L. H. Tran and L. H. Tran, "Applications of (sparse)-pca and laplacian eigenmaps to biological network inference problem using gene expression data," *Int. J. Advance Soft Compu. Appl*, vol. 9, no. 2, 2017.

[29] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

Chapter 5

Noise-Robust Classification System With Hypergraph Neural Network

5.1 Introduction

During the last decade, the deep convolution neural network can be considered the current state of the art method for various classification tasks such as image recognition [1], speech recognition [2], to name a few. Recently, to deal with irregular data structures, data scientists have gained much interests in graph convolution neural network method such as [3]. In this method, the pairwise relationships between objects (samples) are used. In the other words, in this graph data structure, the edge of the graph can connect only two vertices. Easily, we see that assuming the pairwise relationship between objects (samples) is not complete. Moreover, to the best of our knowledge, based on our literature review work, the graph convolution neural network has not yet been applied to solve the noisy label learning problem. This is considered the very "hard" and very important problem. In the other words, it's very easy to make error(s) when data annotator labels data.

To overcome the information loss of the "pairwise relationship between objects" assumption of graph data structure, [4, 5] have recently proposed the hypergraph neural network approach. In this hypergraph data structure, an edge (hyperedge) can connect more than two vertices. In the other words, the hyperedge is the subset of the set of vertices of the hypergraph. Recently, this hypergraph neural network method has just been employed to solve classification tasks [4, 5] and outperforms the graph neural network and can be considered the current state of the art method of semi-supervised learning approach. Obviously, this method has also not been utilized to solve the noisy label learning problem, to the best of our knowledge.

Inspired from the idea combining the PageRank algorithm with the graph convolution neural network in [6], in this chapter, we propose the novel version of hypergraph neural network method combining the classic hypergraph based semi-supervised learning method [7, 8] with the hypergraph neural network method [4, 5].

In this chapter, our contributions are three-folds:

- in order to reduce the runtime constructing the graph and the hypergraph from the image datasets, we apply the dimensional reduction technique PCA to the image datasets.
- propose the novel version of hypergraph neural network method combining the classic hypergraph based semi-supervised learning method with the hypergraph neural network method
- compare the accuracy performance measures of the classic graph based semi-supervised learning problem, the classic hypergraph based semi-supervised learning problem, the graph neural network method, the hypergraph neural network method, and our proposed hypergraph neural network method when we apply these five methods to solve the noisy label learning problem.

We will organize the chapter as follows: Section 5.2 will discuss the related work. Section 5.3 will introduce the novel version of hypergraph neural network method. Section 5.4 will describe the datasets and present the experimental results. Section 5.5 will conclude this chapter and the future direction of researches will be discussed.

5.2 Related work

Learning with label noise gains much interests since label noise may lead to many undesirable concerns such as the decrease in learning performance. The current studies associated to this problem can be assembled into three main groups [9]:

- Robust model approach
- Data filtering approach
- Inherently noise-tolerant learning approach

5.2.1 Robust model approach

This approach empirically studies the robustness property of various classical classification algorithms such as Naïve Bayes probabilistic classifier, C4.5 decision tree, the SMO support vector machine [9], to name a few. Experimental results show that the Naïve Bayes probabilistic classifier and the random forest ensemble classifier are the most robust classical classification systems against noise label [9].

However, the weakness of these classical classification is very clear. First, this approach is inactive. Instead of refining and changing the classical classification algorithms, they only explore the robustness property of commonly used classification systems. Second, the most robust classification system is effective only when the percentage of label noise is minor. The performance of the most robust classification system drops significantly when the percentage of label noise is huge.

Recently, the deep neural networks (i.e., the modern classification systems) have been established and well developed. For example, the deep convolution neural network can be considered the current state of the art and the best classification system for image recognition problem [1]. However, [10–12] showed that a deep neural network with huge enough capacity can memorize the training set labels even when they are randomly made. Hence, they are mostly vulnerable to the label noise. Similar to classical classification systems, label noise can cause overfitting and significantly drop the deep neural networks' performance.

However, [12] observed that when the learning rate is high, deep neural networks may preserve quite extraordinary accuracy. In the other words, the influence of the label noise is not important. This observation was employed in [12] to preserve an approximation of the labels using the running average of deep neural network's forecasts with a high learning rate. These approximations then can be utilized as the control (or supervision) signals to train the deep neural network.

Inspired by the work of [10–12] and [13, 3, 4, 6], in this chapter, we will develop the graph and hypergraph convolution neural network methods and apply these methods to solve the "noisy label learning" problem (using the image datasets such as MNIST, USPS, and FASHION MNIST). To the best of our knowledge, this work has not been investigated and developed.

5.2.2 Data Filtering Approach

In this approach, the samples with noisy labels are distinguished and fixed before the training process. The ruined labels can be merely eliminated or relabeled at the very beginning.

One idea is to employ the forecasts from the classification system (for e.g., the SVM system) to detect mislabeled samples. The class of these methods is called the classification filtering system. However, filtering all the samples that are misclassified by the classification filtering system is too inflexible and risky since the classification filtering system learned from data with noisy labels might not always be accurate.

$k-nn$ classification systems is related to another class of methods. $k-nn$ classification systems are shown that they are very vulnerable to label noise. Hence some $k-nn$ methods target at improving or changing the rules of $k-nn$ algorithms to recognize noisy labels and then eliminate or relabel the associated samples. However, the methods associated to this class are heuristics and might not be efficient close the classification boundary. Moreover, the choice of k (i.e., the number of neighbors) might affect the performance of these methods considerably.

Some data filtering methods depend on some thresholds. These methods compute the score for each sample by using some measures and eliminate the samples that are above the definite threshold. Obviously, these methods are similar to outlier or anomaly or abnormal detection methods which are very hard to solve. Moreover, it's very difficult to differentiate the accurate exemptions from the mislabeled samples.

Last but not least, for the data filtering approach, this approach tends to eliminate a large amount of samples, which may contain key information for classification. Easily, we can also recognize that we can employ the forecasts from deep neural networks to detect mislabeled samples.

5.2.3 Inherently noise-tolerant learning approach

In this approach, there are two ways to attack the "noisy label learning problem":

- First, this way will model the noisy label before or during the training phase to take the noisy label into attention. This model holds the information of the noise and can be inserted into the classification system [14–18]. However, in order to build the model for noisy label, these methods require supplementary parameters, that might increase the time complexity and the model complexity. We know that high model complexity occasionally leads to over-fitting.

- Second, this way proposes the robust loss function for the noise-tolerant model. For example, [19] explored the robustness of various loss functions such as mean squared loss, mean absolute loss, and cross-entropy loss. [20] combined the benefits of the mean absolute loss and the cross-entropy loss to attain the improved loss function.

5.3 Hypergraph Neural Network

5.3.1 Problem Formulations

In this chapter, we would like to solve the "noisy label learning" problem [10, 21]. This problem can also be called the "label distribution learning" problem [22], to name a few.

In this problem, let $X_{train} = x_1, x_2, \ldots, x_l$ be the training set, where $x_i \in R^m . x_i$ can also be called the feature vector i or instance i or sample i of the training set with $1 \leq i \leq l$. Let$Y_L = y_1, y_2, \ldots, y_C$ be the complete set of labels where C is the number of classes in the dataset.

For each sample x_i, there is a label distribution $d = d_{x_i}^{y_1}, d_{x_i}^{y_2}, \ldots, d_{x_i}^{y_C} \in R^C$. Please note that $d_{x_i}^{y_c}$ is the probability that the sample x_i belongs to the class c.

From the above definition, we know that $0 \leq d_{x_i}^{y_c} \leq 1$ and $\sum_c d_{x_i}^{y_c} = 1$.

The objective of the noisy label learning problem is to learn a mapping function $g : x \rightarrow d$ between the sample x and its corresponding label distribution function d.

In the other words, the goal of noisy label learning is to learn the conditional probability mass function $p(y|x)$, where $y \in Y_L,\ x \in X_{train}$.

Assume that $p(y|x)$ is the parametric model $p(y|x, \theta)$, where θ is the parameter vector. Given the training set X_{train}, we need to find (i.e., solve for) θ that can generate the distribution similar to d [23].

5.3.2 Preliminary notations and definitions

Given a hypergraph $G = (V, E)$, where V is the set of vertices and E is the set of hyper-edges. Each hyper-edge $e \in E$ is the subset of V. Please note that the cardinality of e is greater than or equal two. In the other words, $|e| \geq 2$, for every $e \in E$. Let $w(e)$ be the weight of the hyper-edge e. Then W will be the $R^{|E|*|E|}$ diagonal matrix containing the weights of all hyper-edges in its diagonal entries.

The incidence matrix H of G is a $R^{|V|*|E|}$ matrix that can be defined as follows

$$h(v, e) = \begin{cases} 1, & \text{if vertex } v \text{ belongs to hyperedge } e \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

The example of the hypergraphs is illustrated in Figure 5.1.

The incidence matrix H of the hypergraph in the above Figure 5.1 can be expressed as follows

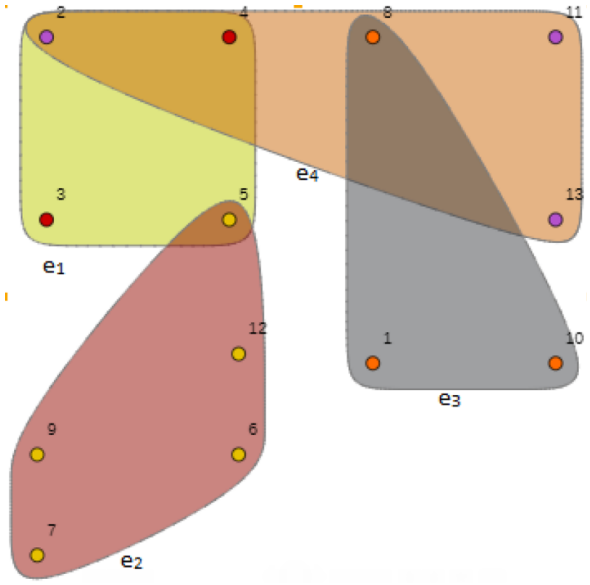

Fig. 5.1 Hypergraph examples with 13 vertices and 4 hyperedges.

$$H = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From the above definition, we can define the degree of vertex v and the degree of hyper-edge e as follows

$$d(v) = \sum_{e \in E} w(e) * h(v,e) \tag{5.2}$$

$$d(e) = \sum_{v \in V} h(v,e) \tag{5.3}$$

Let D_v *and* D_e be two diagonal matrices containing the degrees of vertices and the degrees of hyper-edges in their diagonal entries respectively. Please note that D_v is the $R^{|V|*|V|}$ matrix and D_e is the $R^{|E|*|E|}$ matrix.

From the above definitions, [7, 8] define the symmetric normalized hypergraph Laplacian as follows

$$L_{sym} = I - D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} \tag{5.4}$$

Moreover, [7, 8] define the random walk hypergraph Laplacian as follows

$$L_{rw} = I - D_v^{-1} H W D_e^{-1} H^T \tag{5.5}$$

Please note that the two terms $D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}}$ and $D_v^{-1} H W D_e^{-1} H^T$ in the symmetric normalized hypergraph Laplacian and the random walk hypergraph Laplacian respectively will be used in our proposed hypergraph neural network method.

5.3.3 Hypergraph based semi-supervised learning problem

Given a set of images $x_1, \ldots, x_l, x_{l+1}, \ldots, x_{l+u}$ where $n = |V| = l + u$ is the total number of images (i.e., vertices) in the hypergraph $G = (V, E)$.

The method constructing the incidence matrix H from the image dataset will be specified clearly in the Experiments and Results section.

Define C be the number of classes. Please note that $x_1, \ldots, x_l$ is the set of all labeled points and $x_{l+1}, \ldots, x_{l+u}$ is the set of all un-labeled points.

Let $Y \in R^{|V|*C}$ the initial label matrix for n images in the hypergraph G be defined as follows

$$Y_{ij} = \begin{cases} 1, & \text{if } x_i \in \textit{ class } j \text{ and } 1 \leq i \leq l \\ -1, & \text{if } x_i \notin \textit{ class } j \text{ and } 1 \leq i \leq l \\ 0, & \text{if } l+1 \leq i \leq n \end{cases} \tag{5.6}$$

Let the matrix $F \in R^{|V|*C}$ be the estimated label matrix for the set of images $x_1, \ldots, x_l, x_{l+1}, \ldots, x_{l+u}$, where the point x_i is labeled as $sign(F_{ij})$ for each $class j (1 \leq j \leq C)$

Our objective is to predict the labels of the un-labeled points $x_{l+1}, \ldots, x_{l+u}$. In the other words, we need to compute the final solution matrix F.

From [7, 8], the closed form solution of the hypergraph based semi-supervised learning method can be computed as follows

$$F = (1-\alpha)(I - \alpha D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}})^{-1} Y \tag{5.7}$$

where α is the parameter.

5.3.4 Hypergraph neural network

From [3, 4, 6], the output of the hypergraph neural network can be defined and computed as follows

$$Z = softmax(D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} ReLU(D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} X\theta^1)\theta^2) \tag{5.8}$$

or

$$Z = softmax(D_v^{-1} H W D_e^{-1} H^T ReLU(D_v^{-1} H W D_e^{-1} H^T X\theta^1)\theta^2) \tag{5.9}$$

Please note that $X \in R^{n*L_1}$ is the feature matrix (i.e., the image dataset). $\theta^1 \in R^{L_1*L_2}$ and $\theta^2 \in R^{L_2*C}$ are two parameter matrices that are needed to be learned during the training process.

5.3.5 Our proposed hypergraph neural network

Inspired by the work in [6] for graph neural network, we try to apply this work (idea) to develop our novel version for hypergraph neural network method.

In this novel version, we initially compute the final solution matrix F of the classic hypergraph based semi-supervised learning method as in equation (5.7)

$$F = (1-\alpha)(I - \alpha D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}})^{-1} Y$$

In this first step, we overcome the difficulty which is "only neighbors in the two-hop neighborhood are considered" of the graph neural network method and hypergraph neural network method [3, 4, 6]. The time complexity of this computation is still low since we employ the sparse matrix computation techniques, for examples, the conjugate gradient method, to solve the above sparse linear system of equations.

Finally, we compute the final output Z of the hypergraph neural network as in equation (5.8) with the input feature matrix X replaced by F.

$$Z = softmax(D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} ReLU(D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}} F \theta^1) \theta^2)$$

Simply speaking, our novel method is the combination of the classic hypergraph based semi-supervised learning method and the hypergraph neural network method.

The two main differences of our proposed method with the method proposed in [6] are:

- Initially, the input feature matrix X does not need to go through a neural network.
- At our final step, we compute the output Z of hypergraph neural network similar to the formula proposed by [3]. Please note that [3] proposed the formula to compute the output of the graph neural network. In the other words, in the method proposed in [6], the final solution matrix F of the hypergraph based semi-supervised learning method is just needed to go through only one layer of the neural network which is the softmax layer.

5.4 Experimental Results

In this section, we will apply the classic graph based semi-supervised learning method [24], the classic hypergraph based semi-supervised learning method, graph neural network method, hypergraph neural network method, and our proposed hypergraph neural network method to solve the noisy label learning problem. In the other words, we will test the noise robustness of these five methods. The three image datasets that we will use in the experiments are the MNIST dataset, the USPS dataset, and the FASION MNIST dataset.

5.4.1 Datasets

MNIST: This image dataset is the dataset containing the handwritten images from '0' to '9'. There are 70,000 images in the dataset. There are 60,000 images in the training set and 10,000 images in the testing set. Obviously, the number of classes in this MNIST image dataset is 10. Each image in the dataset is the 28-by-28 matrix (gray scale image). Our first task in the preprocessing step is to convert this gray scale image to 1-by-784 vector. We achieve this task by concatenating every row of the gray scale image to a "long" row vector. In the other words, we have the $R^{70,000*784}$ feature matrix.

USPS: This image dataset is also the handwritten image dataset from '0' to '9'. However, in this dataset, there are just 9,298 images in the dataset. There are 7,291 images in the training set and 2,007 images in the testing set. The number of classes in this USPS dataset

is 10. Each image in the dataset is the 16-by-16 matrix (gray scale image). We concatenate every row of the gray scale image (in this USPS dataset) to the 1-by-256 "long" row vector. Thus, finally, we have the $R^{9,298*256}$ feature matrix.

FASHION MNSIT: This image dataset is the dataset containing images of shoes, clothes, caps, etc. There are 70,000 images in the dataset. There 60,000 images in the training set and 10,000 images in the testing set. The number of classes in this FASHION MNIST image dataset is 10. Each image in the dataset is the 28-by-28 matrix (gray scale image). We concatenate every row of the gray scale image to the 1-by-784 "long" row vector. In the other words, we have the $R^{70,000*784}$ feature matrix. This FASHION MNIST image dataset is considered the hardest image dataset to test in our experiments.

5.4.2 Experiments and Results

In order to reduce the noise and redundant features in the input feature matrices and in order to reduce the time constructing the graphs and hypergraphs from the three image datasets, we apply the dimensional reduction PCA technique to the three input feature matrices. Finally, the MNIST dataset is transformed to the $R^{70,000*50}$ matrix. The USPS dataset is transformed to $R^{9,298*50}$ matrix. The FASHION MNIST dataset is transformed to $R^{70,000*300}$ matrix.

The way constructing the graphs from the three image datasets can be found in [24, 25]. Next, we will discuss how to construct the incidence matrix H of the hypergraphs from the three image datasets. Please note that the number of hyperedges in the hypergraph is equal to the number of images in the dataset [26]. The image i belongs to hyperedge j if image i is among the k-nearest neighbor of image j or image j is among the k-nearest neighbor of image i. In this chapter, k is chosen to be 5.

Finally, from the computed H, we can compute the two terms $D_v^{-\frac{1}{2}}HWD_e^{-1}H^TD_v^{-\frac{1}{2}}$ and $D_v^{-1}HWD_e^{-1}H^T$ in the symmetric normalized hypergraph Laplacian and the random walk hypergraph Laplacian used in the classic hypergraph based semi-supervised learning method, the hypergraph neural network method, and our proposed hypergraph neural network method.

The two main differences of our methods with other semi-supervised learning methods [27, 28] solving the image classification problem with noisy labels are:

- Other semi-supervised learning methods just used the subsets of the MNIST and the USPS datasets. For example, in [27], the authors just used 10,000 images from MNIST to evaluate their methods. In the other hand, our methods use the complete MNIST, USPS, and FASHION MNIST image datasets.

- Our methods apply directly the PCA technique to the feature matrices of the three image datasets in order to reduce the time constructing the graphs and the hypergraphs

Table 5.1 MNIST dataset: Comparison of our five methods with various noise levels. The classification accuracy (%) is reported

Noise level	0%	15%	30%	45%
Graph based semi-supervised learning	97.70	93.33	80.94	57.45
Hypergraph based semi-supervised learning	97.65	97.56	97.54	84.49
Graph neural network	97.39	97.31	97.11	86.15
Hypergraph neural network (current state of the art semi-supervised learning method)	97.52	97.40	**97.34**	87.07
Proposed hypergraph neural network	**97.72**	**97.69**	97.30	**91.65**

of the three image datasets. To the best of our knowledge, this work has not been done before. The experimental results show that if we do not apply the PCA technique to the feature matrices of the three image dataset, the hypergraph neural network significantly outperforms the graph neural network. If we apply the PCA technique to the feature matrices of the three image datasets, the hypergraph neural network method does not significantly outperform the graph neural network method; however, both the hypergraph neural network and the graph neural network with PCA are better than the graph and hypergraph neural network without using PCA technique. These claims will be clarified in Table 5.1, 5.2, 5.3, 5.4

In general, in this chapter, what we want to achieve is clear: we would like to prove that the hypergraph neural networks (the current state of the art semi-supervised learning method and our proposed method) are at least as good as the graph neural network but sometimes lead to better accuracy performance measures. We run our five methods (Python code) on Google Colab with NVIDIA Tesla K80 GPU and 12 GB RAM. The following tables 5.1 (figure 5.2), 5.2 (figure 5.3), 5.3 (figure 5.4), and 5.4 show the experimental results of our five methods.

From the above three tables, we easily recognize initially and directly that when the noise level increases, our proposed hypergraph neural network outperforms the other methods.

Table 5.2 USPS dataset: Comparison of our five methods with various noise levels. The classification accuracy (%) is reported

Noise level	0%	15%	30%	45%
Graph based semi-supervised learning	**95.06**	**94.96**	92.82	66.26
Hypergraph based semi-supervised learning	**95.06**	94.91	**94.71**	74.58
Graph neural network	94.66	94.42	93.12	70.00
Hypergraph neural network (current state of the art semi-supervised learning method)	94.76	94.71	93.82	74.48
Proposed hypergraph neural network	**95.06**	94.81	94.37	**82.51**

Table 5.3 FASHION MNIST dataset: Comparison of our five methods with various noise levels. The classification accuracy (%) is reported

Noise level	0%	15%	30%	45%
Graph based semi-supervised learning	86.13	85.75	84.02	63.61
Hypergraph based semi-supervised learning	84.88	84.69	83.43	69.29
Graph neural network	**86.76**	**86.35**	85.17	67.79
Hypergraph neural network (current state of the art semi-supervised learning method)	86.41	86.30	**85.47**	70.09
Proposed hypergraph neural network	86.14	85.91	85.02	**75.89**

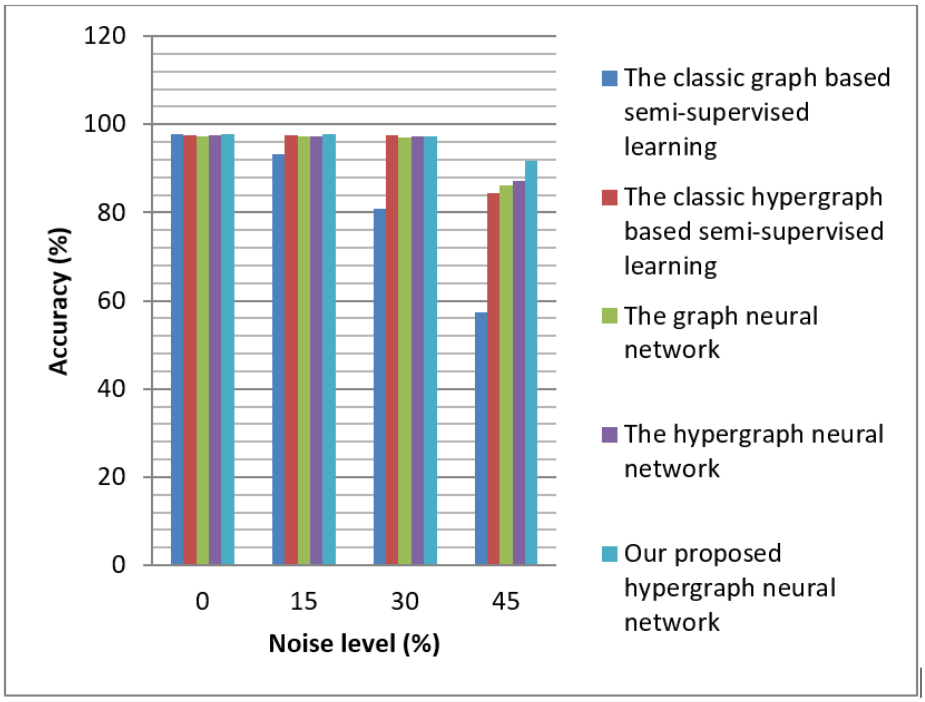

Fig. 5.2 MNIST dataset: Comparison of our five methods with various noise levels.

Table 5.4 FASHION MNIST dataset: Comparison of the hypergraph neural network method (the current state of art semi-supervised learning method) and the graph neural network method

Noise level	0%
Graph neural network (without PCA)	79.98
Hypergraph neural network (without PCA)	85.09
Graph neural network (with PCA)	**86.76**
Hypergraph neural network (with PCA)	86.41

Second, from the experimental results, we see that the hypergraph neural network methods (both the current state of the art semi-supervised learning method and our proposed method) are at least as good as the graph neural network proposed by Thomas Kipf but sometimes lead to better accuracy performance measures. Finally, we can also easily see that the classic graph based semi-supervised learning method performs worst when the noise level increases.

Last but not least, in the FASHION MNIST dataset, we would like to show that if we do not apply the PCA technique to the feature matrix of the FASHION MNIST image dataset, the hypergraph neural network significantly outperforms the graph neural network. This claim is shown in the following table 5.4.

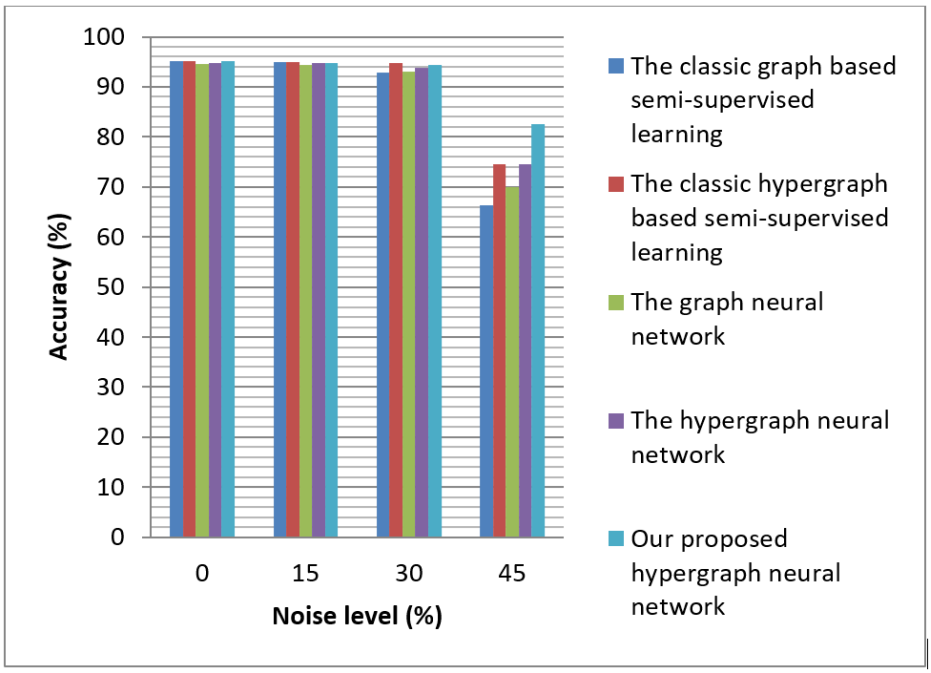

Fig. 5.3 USPS dataset: Comparison of our five methods with various noise levels.

5.5 Conclusions

In this chapter, we have proposed the novel hypergraph neural network method. Our contributions are:

- reduce the time constructing the graph and the hypergraph by initially applying the PCA technique to the image dataset.
- our novel hypergraph neural network method is in fact the combination of the classic hypergraph based semi-supervised learning method and the hypergraph neural network proposed by [4] (i.e., the current state of the art semi-supervised learning method).

The experimental results show that our proposed hypergraph neural network outperforms other semi-supervised learning methods as the noise level in the training set increases. In the other words, our proposed approach is quite robust to noise labels to some extent. Moreover, the hypergraph neural networks (the current state of the art method of semi-supervised learning approach and our proposed method) are at least as good as the graph neural network proposed by Thomas Kipf, but sometimes lead to better accuracies.

Last but not least, in the future work, we will combine the hypergraph p-Laplacian based semi-supervised learning method with the current state of the art semi-supervised learning method (i.e., the hypergraph neural network proposed by [4]) to form a novel hypergraph

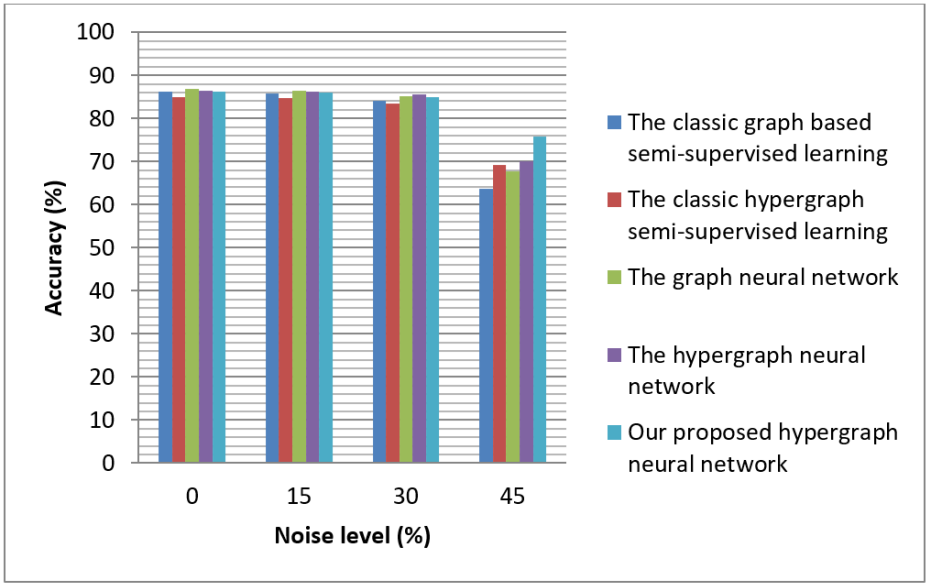

Fig. 5.4 FASHION MNIST dataset: Comparison of our five methods with various noise levels.

neural network method. Finally, we can apply this novel method to various classification tasks such as protein function prediction, cancer classification, and speech recognition, to name a few.

References

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

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

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

[4] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3558–3565, 2019.

[5] S. Bai, F. Zhang, and P. H. Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognition*, vol. 110, p. 107637, 2021.

[6] J. Klicpera, A. Bojchevski, and S. Günnemann, "Personalized embedding propagation: Combining neural networks on graphs with personalized pagerank," *CoRR, abs/1810.05997*, 2018.

[7] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," *Advances in neural information processing systems*, vol. 19, pp. 1601–1608, 2006.

[8] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," 2005.

[9] Z. Zhao, *Classification in the presence of heavy label noise: A Markov chain sampling framework*. PhD thesis, Applied Sciences: School of Computing Science, 2017.

[10] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7017–7025, 2019.

[11] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[12] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560, 2018.

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

[14] J. Bootkrajang and A. Kabán, "Label-noise robust logistic regression and its applications," in *Joint European conference on machine learning and knowledge discovery in databases*, pp. 143–158, Springer, 2012.

[15] J. Bootkrajang, *Supervised learning with random labelling errors*. PhD thesis, University of Birmingham, 2013.

[16] A. J. Bekker and J. Goldberger, "Training deep neural-networks based on unreliable labels," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2682–2686, IEEE, 2016.

[17] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," 2016.

[18] A. J. Bekker, M. Chorev, L. Carmel, and J. Goldberger, "A deep neural network witharestricted noisy channel for identification of functional introns," in *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2017.

[19] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[20] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[21] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2013.

[22] W. Shen, K. Zhao, Y. Guo, and A. Yuille, "Label distribution learning forests," *arXiv preprint arXiv:1702.06086*, 2017.

[23] X. Geng, "Label distribution learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1734–1748, 2016.

[24] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in neural information processing systems*, pp. 321–328, 2004.

[25] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[26] Y. Huang, Q. Liu, S. Zhang, and D. N. Metaxas, "Image retrieval via probabilistic hypergraph ranking," in *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3376–3383, IEEE, 2010.

[27] B. Jiang, Z. Zhang, D. Lin, J. Tang, and B. Luo, "Semi-supervised learning with graph learning-convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11313–11320, 2019.

[28] Z. Lu and L. Wang, "Noise-robust semi-supervised learning via fast sparse coding," *Pattern Recognition*, vol. 48, no. 2, pp. 605–612, 2015.

Chapter 6

Directed Hypergraph Neural Network

6.1 Introduction

In recent years, deep convolution neural networks have gained much interests from data scientists and have utilized to solve many classification tasks such as image recognition [1] and speech recognition [2], to name a few. In order to deal with irregular data structure, graph convolution neural networks have been developed by a lot of data scientists such as Thomas Kipf [3]. There are two classes of graph convolution neural network. The first class of graph convolution neural network is the spatial based approach. This spatial based approach implements the convolution on the graph by accumulating information of the neighbor nodes. The second class of graph convolution neural network is the spectral based approach. This spectral based approach implements a variant of graph convolution neural network based on different graph Laplacians. Easily, we recognize that the time complexity of spectral based approach is much higher than the time complexity of spatial based approach; however, the accuracy of the spectral based approach is higher than the accuracy of the spatial based approach. In this graph data structure, we easily see that the edge carries no information about the direction. Moreover, in this graph data structure, the edge can connect only two vertices. In the other words, data scientists have concentrated primarily on developing deep neural network method for un-directed graph.

In order to overcome these two information losses which are "the edge carry no information about the direction" and "the edge can connect only two vertices" of the graph data structure which "can" affect the performance of the "node clustering task" or the "node classification task", we employ the directed hypergraph data structure [4, 5] and develop the deep neural network method based on this directed hypergraph data structure. This method is called the spectral directed hypergraph neural network method. The development of this directed hypergraph neural network is considered the very hard task and the novel work.

First, we need to define the transition probability matrix of the random walk on the directed hypergraph. Then, we need to compute the PageRank vector of the directed hypergraph. Finally, we can compute the directed hypergraph Laplacian. From the directed hypergraph Laplacian, we can start developing the spectral directed hypergraph neural network and apply this novel method to solve the classification task. The two citation datasets that are used in the classification task are the Cora and the Citeseer datasets. To the best of our knowledge, this research work has not been developed before.

In this chapter, our contributions are three folds:

- Develop the novel directed hypergraph semi-supervised learning method.
- Develop the novel directed hypergraph neural network.
- The accuracies of the classic directed graph semi-supervised learning method (which is the baseline method), the novel directed hypergraph semi-supervised learning method, the novel directed hypergraph neural network are computed and compared.

We will organize the chapter as follows: Section 6.2 will present the preliminary notations and definitions. Section 6.3 will introduce the novel directed hypergraph semi-supervised learning method. Section 6.4 will present the directed hypergraph neural network. Section 6.5 will describe the datasets and present the experimental results. Section 6.6 will conclude this chapter and the future direction of researches will be discussed.

6.2 Preliminary notations and definitions

Given the directed hypergraph $H = (V,E)$ where V is the set of vertices and E is the set of hyper-arcs. Each hyper-arc $e \in E$ is written as $e = (e^{Tail}, e^{Head})$. The vertices of e are denoted by $\mathbf{e} = e^{Tail} \cup e^{Head}$.$e^{Tail}$ is called the tail of the hyper-arc e and e^{Head} is called the head of the hyper-arc e. Please note that $e^{Tail} \neq \emptyset$, $e^{Head} \neq \emptyset$, $e^{Tail} \cap e^{Head} = \emptyset$.

The directed hypergraph $H = (V,E)$ can be represented by two incidence matrices H^{Tail} and H^{Head}.

These two incidence matrices H^{Tail} and H^{Head} can be defined as follows

$$h^{Tail}\left(v, e^{Tail}\right) = \begin{cases} 1, & \text{if } v \in e^{Head} \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

$$h^{Head}\left(v, e^{Head}\right) = \begin{cases} 1, & \text{if } v \in e^{Head} \\ 0, & \text{otherwise} \end{cases} \tag{6.2}$$

The example of the directed hypergraph is illustrated in Figure 6.1:

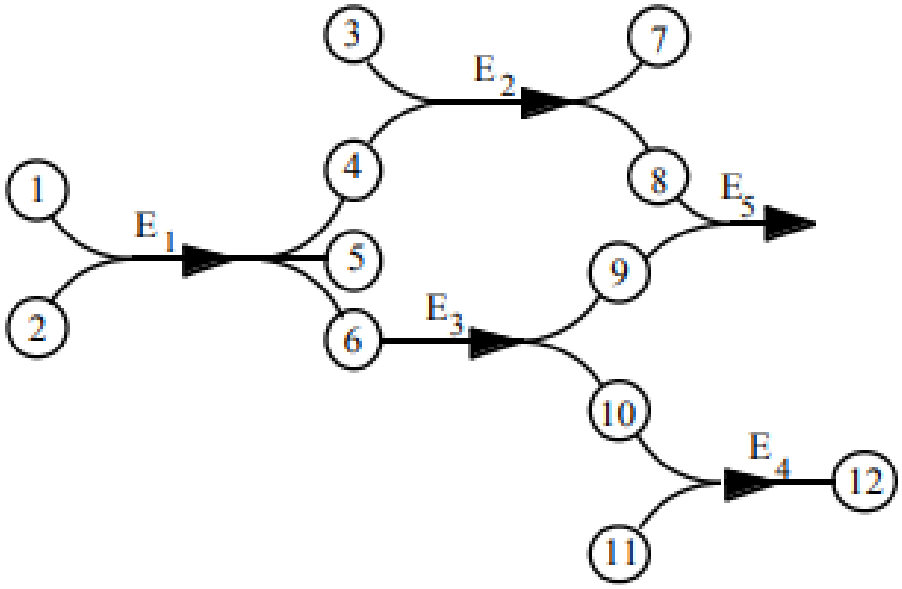

Fig. 6.1 Directed hypergraph example with 12 vertices and 5 hyper-arcs [4].

Let $w(e)$ be the weight of the hyper-arc e. Let W be the diagonal matrix containing the weights of hyper-arcs in its diagonal entries.

From the above definitions, we can define the tail and head degrees of the vertex v and the tail and head degrees of the hyper-arc e as follows

$$d^{Tail}(v) = \sum_{e \in E} w(e) h^{Tail}\left(v, e^{Tail}\right) \tag{6.3}$$

$$d^{Head}(v) = \sum_{e \in E} w(e) h^{Head}\left(v, e^{Head}\right) \tag{6.4}$$

$$d^{Tail}(e) = \sum_{v \in V} h^{Tail}\left(v, e^{Tail}\right) \tag{6.5}$$

$$d^{Head}(e) = \sum_{v \in V} h^{Head}\left(v, e^{Head}\right) \tag{6.6}$$

Let $D_v^{Tail}, D_v^{Head}, D_e^{Tail}$, and D_e^{Head} be four diagonal matrices containing the tail and head degrees of vertices and the tail and head degrees of hyper-arcs in their diagonal entries respectively. Please note that D_v^{Head} and D_v^{Tail} are the $R^{|V|*|V|}$ matrices and D_e^{Head} and D_e^{Tail} are the $R^{|E|*|E|}$ matrices.

From [6], we know that the transition probability of the random walk on directed hypergraph can be defined as follows

$$p(u,v) = \sum_{e \in E} w(e) \frac{h^{Tail}(u, e^{Tail})}{d^{Tail}(u)} \frac{h^{Head}(v, e^{Head})}{d^{Head}(e)} \tag{6.7}$$

From the above definition, the transition probability matrix P of the random walk on the directed hypergraph can be defined in the matrix form as follows

$$P = D_v^{Tail\,-1} H^{Tail} W D_e^{Head\,-1} H^{Head\,T} \tag{6.8}$$

The PageRank vector π of the directed hypergraph is the solution of the following equation

$$\pi^T = \pi^T P \tag{6.9}$$

Moreover, we know that the above equation can easily be solved by the Power method.

Next, we give two novel definitions of the directed hypergraph Laplacian which are un-normalized directed hypergraph Laplacian and symmetric normalized directed hypergraph Laplacian.

Let S be the diagonal matrix containing all elements of PageRank vector π of the directed hypergraph in its diagonal entries.

The un-normalized directed hypergraph Laplacian can be defined as follows

$$L = S - \frac{S D_v^{Tail\,-1} H^{Tail} W D_e^{Head\,-1} H^{Head\,T} + \left(D_v^{Tail\,-1} H^{Tail} W D_e^{Head\,-1} H^{Head\,T}\right)^T S}{2} \tag{6.10}$$

The symmetric normalized directed hypergraph Laplacian can be defined as follows

$$L_{sym} = I - \frac{S^{\frac{1}{2}} D_v^{Tail\,-1} H^{Tail} W D_e^{Head\,-1} H^{Head\,T} S^{-\frac{1}{2}} + S^{-\frac{1}{2}} \left(D_v^{Tail\,-1} H^{Tail} W D_e^{Head\,-1} H^{Head\,T}\right)^T S^{\frac{1}{2}}}{2} \tag{6.11}$$

From these two above definitions, we can develop the directed hypergraph Laplacian based semi-supervised learning algorithms and the directed hypergraph neural network.

6.3 Directed hypergraph semi-supervised learning

Given a set of samples $x_1,\ldots,x_l,x_{l+1},\ldots,x_{l+u}$ where $n=|V|=l+u$ is the total number of samples (i.e., vertices) in the directed hypergraph $H=(V,E)$.

The method constructing the incidence matrix H^{Tail} and H^{Head} from the datsets will be specified clearly in the Experiments and Results section (i.e., Section 6.5).

Define C be the number of classes. Please note that $x_1,\ldots,x_l$ is the set of all labeled points and $x_{l+1},\ldots,x_{l+u}$ is the set of all un-labeled points.

Let $Y\in R^{|V|*C}$ the initial label matrix for n samples in the directed hypergraph H be defined as follows

$$Y_{ij}=\begin{cases}1, & \text{if } x_i\in \textit{ class } j \text{ and } 1\leq i\leq l\\ -1, & \text{if } x_i\notin \textit{ class } j \text{ and } 1\leq i\leq l\\ 0, & \text{if } l+1\leq i\leq n\end{cases} \tag{6.12}$$

Let the matrix $F\in R^{|V|*C}$ be the estimated label matrix for the set of samples $x_1,\ldots,x_l,x_{l+1},\ldots,x_{l+u}$, where the point x_i is labeled as $sign(F_{ij})$ for each class $j(1\leq j\leq C)$

Our objective is to predict the labels of the un-labeled points $x_{l+1},\ldots,x_{l+u}$. In the other words, we need to compute the final solution matrix F.

In short, we would like to solve the following optimization problem

$$E(f)=\frac{1}{2}\sum_{(u,v)\subseteq E}\pi(u)\sum_{e\in E}w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\left(\frac{f(u)}{\sqrt{\pi(u)}}-\frac{f(v)}{\sqrt{\pi(v)}}\right)^2+\mu||f-y||^2 \tag{6.13}$$

We know that

$$\frac{1}{2}\sum_{(u,v)\subseteq E}\pi(u)\sum_{e\in E}w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\left(\frac{f(u)}{\sqrt{\pi(u)}}-\frac{f(v)}{\sqrt{\pi(v)}}\right)^2$$

$$=\frac{1}{2}\sum_{e\in E}\frac{1}{2}\sum_{v\in V}\{\sum_{u\to v}\pi(u)w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\left(\frac{f(u)}{\sqrt{\pi(u)}}-\frac{f(v)}{\sqrt{\pi(v)}}\right)^2$$

$$+\sum_{u\leftarrow v}\pi(v)w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\left(\frac{f(v)}{\sqrt{\pi(v)}}-\frac{f(u)}{\sqrt{\pi(u)}}\right)^2\}$$

$$= \frac{1}{2}\sum_{e\in E}\{\frac{1}{2}\sum_{v\in V}\{\sum_{u\to v} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}f^2(u)$$

$$+\sum_{u\to v} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(u)}{\pi(v)}f^2(v)$$

$$-2\sum_{u\to v}\pi(u)w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{1}{\sqrt{\pi(u)}\sqrt{\pi(v)}}f(u)f(v)\}$$

$$+\frac{1}{2}\sum_{v\in V}\{\sum_{u\leftarrow v} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}f^2(v)$$

$$+\sum_{u\leftarrow v} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(v)}{\pi(u)}f^2(u)$$

$$-2\sum_{u\to v}\pi(v)w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{1}{\sqrt{\pi(u)}\sqrt{\pi(v)}}f(u)f(v)\}\}$$

Moreover, we know that

$$\sum_{e\in E}\sum_{(u,v)\subseteq e} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}f^2(u)$$

$$=\sum_{u\in V}\sum_{v\leftarrow u}\sum_{e\in E} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}f^2(u)$$

$$=\sum_{u\in V}\left(\sum_{v\leftarrow u} p(u,v)\right)f^2(u)=\sum_{u\in V} f^2(u)=\sum_{v\in V} f^2(v)$$

$$\sum_{e\in E}\sum_{(u,v)\subseteq e} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(u)}{\pi(v)}f^2(v)$$

$$=\sum_{v\in V}\sum_{u\to v}\sum_{e\in E} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(u)}{\pi(v)}f^2(v)$$

$$=\sum_{v\in V}\left(\sum_{u\to v}\frac{p(u,v)\pi(u)}{\pi(v)}\right)f^2(v)=\sum_{v\in V} f^2(v)$$

$$\sum_{e\in E}\sum_{(u,v)\subseteq e}\pi(u)w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{1}{\sqrt{\pi(u)}\sqrt{\pi(v)}}f(u)f(v)$$

$$=\sum_{v\in V}\sum_{u\to v}\sum_{e\in E} w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{\sqrt{\pi(u)}}{\sqrt{\pi(v)}}f(u)f(v)$$

$$\sum_{e\in E}\sum_{(v,u)\subseteq e} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}f^2(v)$$

$$=\sum_{v\in V}\sum_{u\leftarrow v}\sum_{e\in E} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}f^2(v)$$

$$=\sum_{v\in V}(\sum_{u\leftarrow v} p(v,u))f^2(v)=\sum_{v\in V} f^2(v)$$

$$\sum_{e\in E}\sum_{(v,u)\subseteq e} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(v)}{\pi(u)}f^2(u)$$

$$=\sum_{u\in V}\sum_{v\rightarrow u}\sum_{e\in E} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{\pi(v)}{\pi(u)}f^2(u)$$

$$=\sum_{u\in V}\left(\sum_{v\rightarrow u}\frac{p(v,u)\pi(v)}{\pi(u)}\right)f^2(u)=\sum_{v\in V} f^2(v)$$

$$\sum_{e\in E}\sum_{(v,u)\subseteq e}\pi(v)w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{1}{\sqrt{\pi(u)}\sqrt{\pi(v)}}f(u)f(v)$$

$$=\sum_{v\in V}\sum_{u\leftarrow v}\sum_{e\in E} w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{\sqrt{\pi(v)}}{\sqrt{\pi(u)}}f(u)f(v)$$

Thus, we can conclude that

$$\frac{1}{2}\sum_{(u,v)\subseteq E}\pi(u)\sum_{e\in E}w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\left(\frac{f(u)}{\sqrt{\pi(u)}}-\frac{f(v)}{\sqrt{\pi(v)}}\right)^2$$

$$=\sum_{v\in V}f^2(v)-\frac{1}{2}\left(\sum_{u\to v}\sum_{e\in E}w(e)\frac{h^{Tail}\left(u,e^{Tail}\right)}{d^{Tail}(u)}\frac{h^{Head}\left(v,e^{Head}\right)}{d^{Head}(e)}\frac{\sqrt{\pi(u)}}{\sqrt{\pi(v)}}f(u)f(v)\right.$$

$$\left.+\sum_{u\leftarrow v}\sum_{e\in E}w(e)\frac{h^{Tail}\left(v,e^{Tail}\right)}{d^{Tail}(v)}\frac{h^{Head}\left(u,e^{Head}\right)}{d^{Head}(e)}\frac{\sqrt{\pi(v)}}{\sqrt{\pi(u)}}f(u)f(v)\right)$$

Finally, in general, the closed form solution of the directed hypergraph based semi-supervised learning method can be computed as follows

$$F=(1-\alpha)(I-\alpha\frac{S^{1/2}D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T}S^{-1/2}+S^{-1/2}(D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T})^TS^{1/2}}{2})^{-1}Y \tag{6.14}$$

where α is the parameter.

6.4 Directed hypergraph neural network

Let

$$T=\frac{S^{1/2}D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T}S^{-1/2}+S^{-1/2}(D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T})^TS^{1/2}}{2} \tag{6.15}$$

From [3, 7, 8], the output of the directed hypergraph neural network can be defined and computed as follows

$$Z=softmax(TReLU(TX\theta^1)\theta^2) \tag{6.16}$$

Please note that $X\in R^{n*L_1}$ is the feature matrix (i.e., the image dataset). $\theta^1\in R^{L_1*L_2}$ and $\theta^2\in R^{L_2*C}$ are two parameter matrices that are needed to be learned during the training process.

We can easily recognize that instead of adding a self-loop to each node in renormalization phase as in [3], we directly use the term $\frac{S^{1/2}D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T}S^{-1/2}+S^{-1/2}(D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T})^TS^{1/2}}{2}$ (i.e. T) in the directed hypergraph Laplacian to compute the output Z of the directed hypergraph neural network. Obviously, we see that T has the eigenvalues in the range $[-1,1]$.

6.5 Experiments and Results

In this section, we will apply the classic directed graph based semi-supervised learning method [9], the novel directed hypergraph based semi-supervised learning method, the novel directed hypergraph neural network method to solve the classification problem. The two citation datasets that we will use in the experiments are the Cora dataset and the Citeseer dataset [10].

6.5.1 Datasets

Cora: This dataset consists of 2,708 scientific publications classified into one of seven classes which are Case_Based, Genetic_Algorithms, Neural_Networks, Probabilistic_Methods, Reinforce_Learning, Rule_Learning, Theory. The citation network consists of 5,429 links. In the other words, this Cora citation network contains 2,708 nodes (i.e., scientific publications) and 5,429 edges (i.e., citation links). For training, we use 20 samples per class. In the other words, there are 140 samples in the training set. Each publication in this Cora dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary contains 1,433 unique words. In the other words, we have the $R^{2,708*1,433}$ feature matrix.

Citeseer: This dataset consists of 3,312 scientific publications classified into one of six classes which are Agents, AI, DB, IR, ML, and HCI. The citation network consists of 4,732 links. In the other words, this citeseer citation network contains 3,312 nodes (i.e., scientific publications) and 4,732 edges (i.e., citation links). For training, we use 70 samples per class. In the other words, there are 420 samples in the training set. Each publication in the citeseer dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary contains 3,703 unique words. In the other words, we have the $R^{3,312*3,703}$ feature matrix.

6.5.2 Experiments and Results

In this section, initially, we will show how to construct the directed graph from the citation network. If there is a link from node i to node j, we will construct a link from node j to node i. In the other words, there are two links (with directions) between the cited scientific publication and the citing publication. Not only the cited scientific publication influences the citing publication, but the citing scientific publication also influences the cited publication.

Next, we will discuss how to construct a directed hypergraph. First, please note that the number of hyper-arcs in the hypergraph is equal to the number of scientific publications in the dataset. There are two classes of directed hypergraph that we will construct (from the directed graph describing above) in this chapter. The first class is the F-directed hypergraph. The F-directed hypergraph is the directed hypergraph whose hyperarcs are F-arcs. The F-arc is the hyper-arc whose the tail has only one node. The second class is the B-directed hypergraph. The B-directed hypergraph is the directed hypergraph whose hyperarcs are B-arcs. The B-arc is the hyper-arc whose the head has only one node. After constructing the F-directed hypergraph or the B-directed hypergraph, we will construct the two incidence matrices H^{Tail} and H^{Head} of the directed hypergraph. From the computed H^{Tail} and H^{Head}, we can compute the term $\frac{S^{1/2}{D_v^{Tail}}^{-1}H^{Tail}W{D_e^{Head}}^{-1}H^{Head^T}S^{-1/2}+S^{-1/2}(D_v^{Tail^{-1}}H^{Tail}WD_e^{Head^{-1}}H^{Head^T})^TS^{1/2}}{2}$ in the symmetric normalized directed hypergraph Laplacian that will be used in the novel directed hypergraph based semi-supervised learning method and the novel directed hypergraph neural network method.

The example of the B-arc and F-arc are illustrated in Figure 6.2:

Fig. 6.2 B-arc (a) and F-arc (b) examples [4].

We run our six methods which are the classic directed graph based semi-supervised learning method, the novel F-directed hypergraph based semi-supervised learning method, the B-directed hypergraph based semi-supervised learning method, the directed graph neural network method, the F-directed hypergraph neural network method, the B-directed hypergraph neural network method (Python code) on Google Colab with NVIDIA Tesla K80 GPU

Table 6.1 Cora dataset: Comparison of our six methods. The classification accuracy (%) is reported

Methods	Accuracy (%)
Directed graph based semi-supervised learning	67.25
F-directed hypergraph based semi-supervised learning	67.25
B-directed hypergraph based semi-supervised learning method	67.25
Directed graph neural network method	81.42
F-directed hypergraph neural network method	**81.93**
B-directed hypergraph neural network method	81.85

and 12 GB RAM. The following table 6.1 (figure 6.3) and table 6.2 (figure 6.4) show the experimental results of our six methods.

From the above tables and figures, we easilly see that the directed graph neural network method and the directed hypergraph neural network method significantly are better than the classic directed graph semi-supervised learning method and the novel directed hypergraph semi-supervised learning method. Moreover, the F-directed hypergraph neural network method and the B-directed hypergraph neural network method are slightly better than the directed graph neural network method. In general, for these two Cora dataset and Citeseer dataset, the F-directed hypergraph neural network method reaches the highest accuracy performance measures.

Last but not least, interestingly, the novel directed hypergraph based semi-supervised learning method does not outperform the classic directed graph based semi-supervised learning method. We think that the way constructing the directed hypergraph from the

Table 6.2 Cora dataset: Comparison of our six methods. The classification accuracy (%) is reported

Methods	Accuracy (%)
Directed graph based semi-supervised learning	48.23
F-directed hypergraph based semi-supervised learning	48.23
B-directed hypergraph based semi-supervised learning method	48.23
Directed graph neural network method	69.84
F-directed hypergraph neural network method	**70.53**
B-directed hypergraph neural network method	70.43

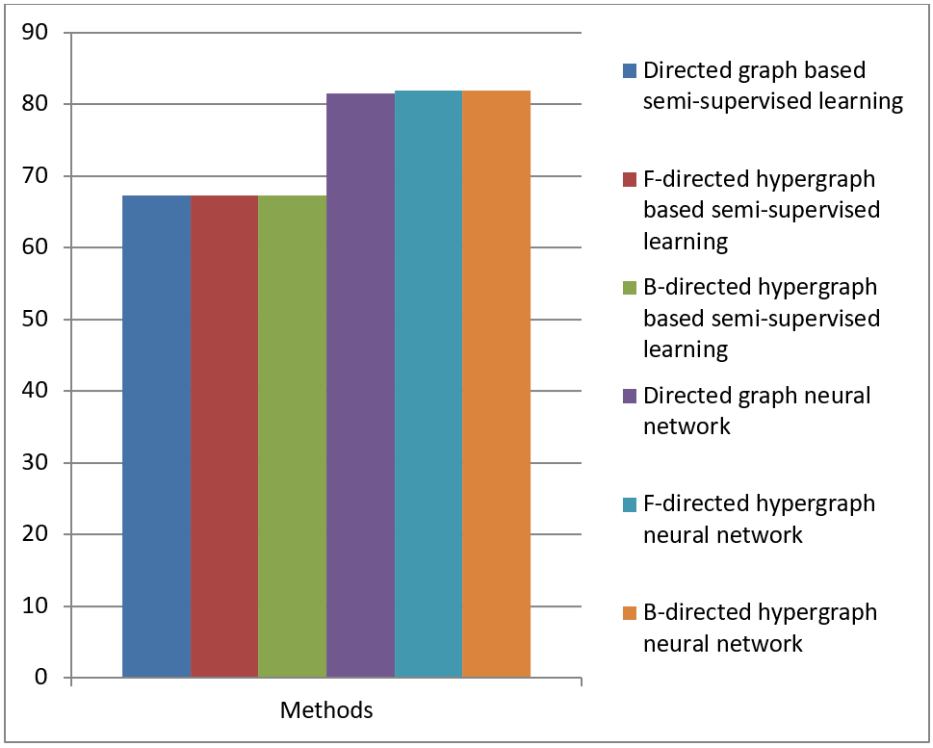

Fig. 6.3 Cora dataset: Comparison of our six methods

directed graph is not good enough. In the future, we will investigate more carefully how to construct the directed hypergraph from the directed graph.

6.6 Conclusions

In this chapter, we have successfully developed the novel directed hypergraph based semi-supervised learning method and the novel directed hypergraph neural network method. Experimental results show that the directed hypergraph neural network method significantly outperforms the novel directed hypergraph based semi-supervised learning method. Moreover, the F-directed hypergraph neural network method achieves the highest accuracy performance measures for the two datasets: Cora and Citeseer among other directed (hyper)-graph based methods.

Last but not least, in the future work, we will combine the directed hypergraph p-Laplacian based semi-supervised learning method with the directed hypergraph neural network to construct a novel directed hypergraph neural network method. Finally, we can apply this novel method to various datasets such Cora, Citeseer, and PubMed, to name a few.

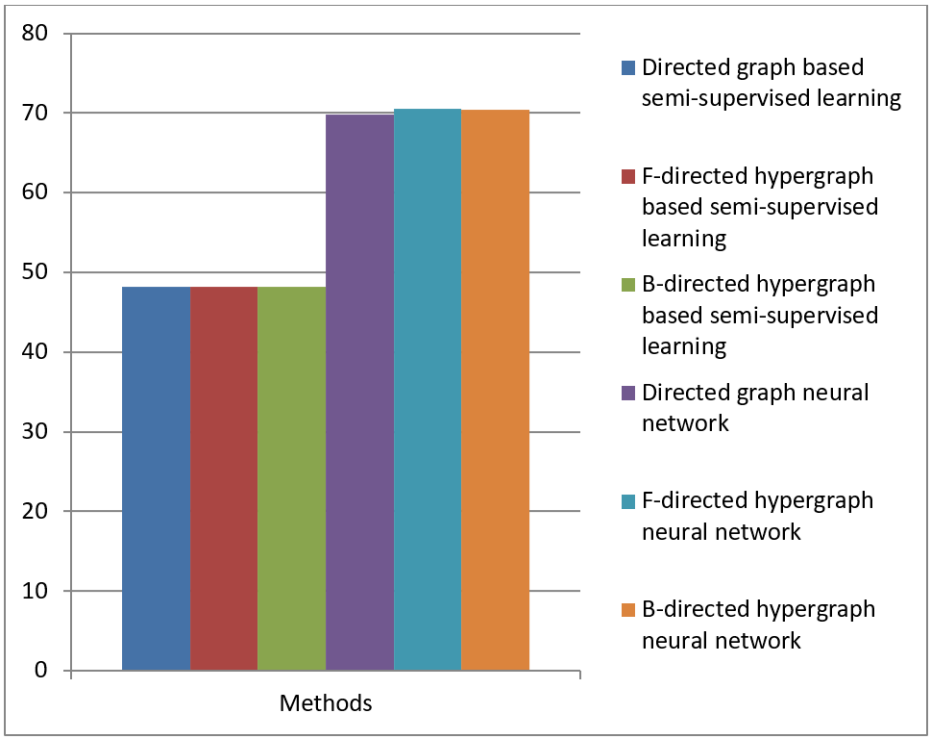

Fig. 6.4 Citeseer dataset: Comparison of our six methods

References

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

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

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

[4] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen, "Directed hypergraphs and applications," *Discrete applied mathematics*, vol. 42, no. 2-3, pp. 177–201, 1993.

[5] G. Gallo and M. G. Scutella, "Directed hypergraphs as a modelling paradigm," *Rivista di matematica per le scienze economiche e sociali*, vol. 21, no. 1-2, pp. 97–123, 1998.

[6] A. Ducournau and A. Bretto, "Random walks in directed hypergraphs and application to semi-supervised image segmentation," *Computer Vision and Image Understanding*, vol. 120, pp. 91–102, 2014.

[7] S. Bai, F. Zhang, and P. H. Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognition*, vol. 110, p. 107637, 2021.

[8] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3558–3565, 2019.

[9] D. Zhou, J. Huang, and B. Schölkopf, "Learning from labeled and unlabeled data on a directed graph," in *Proceedings of the 22nd international conference on Machine learning*, pp. 1036–1043, 2005.

[10] L. Getoor, "Link-based classification," in *Advanced methods for knowledge discovery from complex data*, pp. 189–207, Springer, 2005.

Chapter 7

Conclusions

In 2007, Ulrike von Luxburg has shown (in detail) that there are three main types of graph Laplacians which are:

- Un-normalized graph Laplacian
- Random walk graph Laplacian
- Symmetric normalized graph Laplacian

This leads to three spectral clustering algorithms that have proposed clearly in Ulrike von Luxburg's paper.

Hypergraph is the generalization of the graph. Similar to graph, we define three main types of hypergraph Laplacians (in chapter 1) which are:

- Un-normalized hypergraph Laplacian
- Random walk hypergraph Laplacian
- Symmetric normalized hypergraph Laplacian

In chapter 2, we proposed the detailed algorithms of the un-normalized hypergraph Laplacian Eigenmaps, the weighted un-normalized hypergraph Laplacian Eigenmaps applying to the zoo dataset and the tiny version of 20 newsgroups dataset. Interestingly, experiments show that the weighted un-normalized hypergraph Laplacian Eigenmaps algorithm is at least as good as the un-normalized hypergraph Laplacian Eigenmaps algorithm but sometimes leads to better accuracy performance.

In chapter 3, we proposed the detailed algorithms of the un-normalized hypergraph p-Laplacian based semi-supervised learning methods applied to the zoo dataset and the tiny

version of 20 newsgroups dataset. Interestingly, experiments show that the un-normalized hypergraph p-Laplacian based semi-supervised learning methods are at least as good as the un-normalized hypergraph Laplacian based semi-supervised learning method (the current state of the art method $p = 2$) but often lead to significant better classification accuracy performance measures.

In chapter 4, our contributions are four folds:

- Develop the novel graph convolutional neural network based clustering technique.
- Develop the novel hypergraph convolutional neural network based clustering technique.
- Apply these novel clustering techniques to two Citeseer and Cora datasets.
- Compare the performance of the hypergraph convolutional neural network based clustering technique with the performances of the graph convolutional neural network based clustering technique, the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix.

In this chapter, we proposed the (hyper)-graph convolutional neural network based clustering technique to solve the clustering problem. The graph convolutional neural network based clustering technique is significantly better than the k-means clustering technique, the spectral clustering technique for feature vectors, the spectral clustering technique for adjacency matrix since graph convolutional neural network based clustering technique utilize both the information from the feature vectors and the adjacency matrix of the dataset and noises and redundant features in the dataset (i.e., both in the feature vectors and the adjacency matrix) are removed. Moreover, the hypergraph convolutional neural network based clustering technique is better than the graph convolutional neural network based clustering technique since the hypergraph data structure employs the high order relationships among the samples/entities/objects. This will lead to no loss of information.

In chapter 5, we proposed the novel hypergraph neural network method. Our contributions are:

- reduce the time constructing the graph and the hypergraph by initially applying the PCA technique to the image dataset.
- our novel hypergraph neural network method is in fact the combination of the classic hypergraph based semi-supervised learning method and the hypergraph neural network proposed by Yifan Feng (i.e., the current state of the art semi-supervised learning method).

The experimental results show that our proposed hypergraph neural network outperforms other semi-supervised learning methods as the noise level in the training set increases. In the other words, our proposed approach is quite robust to noise labels to some extent. Moreover, the hypergraph neural networks (the current state of the art method of semi-supervised learning approach and our proposed method) are at least as good as the graph neural network proposed by Thomas Kipf, but sometimes lead to better accuracies.

In chapter 6, we have successfully developed the novel directed hypergraph based semi-supervised learning method and the novel directed hypergraph neural network method. Experimental results show that the directed hypergraph neural network method significantly outperforms the novel directed hypergraph based semi-supervised learning method. Moreover, the F-directed hypergraph neural network method achieves the highest accuracy performance measures for the two datasets: Cora and Citeseer among other directed (hyper)-graph based methods.

In general, in this thesis, I developed the novel methods (i.e., novelty property) to solve various machine learning/deep learning problems for hypergraph and directed hypergraph data structure.

In specific, we solved three main problems in machine learning/deep learning research field such as:

- Representational learning/Dimensional reduction/...
- Clustering
- Classification

RÉSUMÉ

Le graphe des Laplaciens a été largement utilisé dans la réduction dimensionnelle méthodes, méthodes de regroupement et méthodes d'apprentissage semi-supervisé depuis les années 2000. Les candidatures de ces méthodes sont énormes comme im- recherche d'âge, séparation de la parole, et la prédiction de la fonction des protéines tion. Comment- jamais, dans de nombreuses applications du monde réel, représentant l'ensemble de données sous forme de graphique n'est pas complet. Com-relation plex comme par paire conduira à la perte d'informations. Le naturel moyen de surmonter la perte d'informations est de représenter l'ensemble de données comme l'hy- pergraphe. Dans cette thèse, la dimension méthodes de réduction fonctionnelle, clustering méthodes et méthodes de classification pour la structure de données hypergraphiques être développé. Ce travail comprend la méthode classique d'apprentissage automatique ods et l'apprentissage profond moderne méthodes de structure de données hypergraphiques ture.

MOTS CLÉS

graphe, hypergraphe, laplacien, méthode d'apprentissage semi-supervisé, clustering, plongements, réseau de neurones.

ABSTRACT

The graph Laplacians has been widely used in dimensional reduction methods, clustering methods, and semi-supervised learning methods (i.e., Laplacian Eigenmaps, spectral clustering, and graph-based semi-supervised learning) since 2000s (i.e., the classical machine learning method). The applications of these methods are huge such as image retrieval (Laplacian Eigenmaps), speech separation (spectral clustering), and protein function prediction (graph Laplacian based semi-supervised learning method). However, in many real-world applications, representing the dataset as the graph is not complete. Approximating complex relationship as pairwise will lead to the loss of information. The natural way overcoming the information loss is to represent the dataset as the hypergraph. In this thesis, the dimensional reduction methods, clustering methods, and classification methods for hypergraph data structure (i.e., utilizing the hypergraph Laplacian) will be developed. This work includes the classic machine learning methods and the modern deep learning methods for hypergraph data structure.

KEYWORDS

graph, hypergraph, Laplacian, semi-supervised learning method, clustering, embeddings, neural network.