南京大学本科生毕业论文 (设计、作品) 中文摘要

题目：图嵌入的两种实现方法研究
院系：匡亚明学院
专业：计算机科学与技术
本科生姓名：李凯旭
指导老师（姓名、职称）：顾庆 教授
摘要：

在分类问题中，图嵌入方法在针对图数据的机器学习过程中是一种有效的数据处理方式。大型图结构进行嵌入的目的在于学习一个隐藏层表示，将原本的高维特征向量映射到一个较低的维度，同时保证在嵌入过程中节点的内在属性和图拓扑结构尽可能的被保留下来。在图嵌入方法中，出现了两种类型的方法：一是以矩阵运算为基础的图卷积网络方法，二是以自然语言处理（尤其是 skip-gram 算法）为基础的随机游走方法。

这篇文章中首先介绍这两种方法的处理过程，并描述自己在图嵌入分类应用中两种方法的实现模型。对于图卷积网络方法，我们采用堆叠两层一阶邻居聚合函数的方法。对于随机游走方法，我们采用朴素的 DeepWalk 方法。我们在文献引用网络（cora）和社交网络（blogcatalog，Flickr）上测试了两种方法，证明了两种嵌入方法在图嵌入领域捕捉图拓扑结构和节点特征的能力。同时，我们也测试了图卷积网络方法在微小扰动下的表现，并提出了改进建议。

这两种方法均有可扩展的能力，较高的准确性与并发的可能性。这些特征使得这些方法在更广的现实世界进行应用，例如节点分类，蛋白质结构预测，边属性预测等。

关键词： 图嵌入；隐藏层表示；深度学习

南京大学本科生毕业论文 (设计、作品) 英文摘要

THESIS: Research on the implementation of two methods of graph embedding
DEPARTMENT: Department of Computer Science and Technology
SPECIALIZATION: Kuang Yaming Honors School
UNDERGRADUATE: Li Kaixu
MENTOR: Professor Gu Qing
ABSTRACT:

The graph embedding have proved an efficient way to process raw data in machine learning on graph-structured data in the classification problem. The aim of graph embedding on large-scale graphs is to learn a latent representation for each node in the graph, mapping the initial high-dimensional feature vectors to a relative low-dimensional feature vectors. At the same time, we want to maintain the initial inherent features and the topology of the graph. As for the approaches for embedding, most of them fall into two categories. The first one is based on the explicit graph Laplacian regulation on adjacent matrix, using graph convolutional networks(GCN). The other one is inspired by the skip-gram model in the nature language processing, using random walk methods.

We first introduce the theoretical background and the concise process of the two methods. Then, we present our model based on the two ways when we try to solve the classification problem. In GCN, we apply a two layer-wise convolutional network, leveraging aggregation functions on 1-th neighbourhood. In randomwalk, we use the simple DeepWalk. We experiment the two different ways on citation network(cora) and social network(blogcatalog, Flickr) and confirm the ability of them to capture the features and topology of a graph. Moreover, we test the performance of GCN method when faced with small noise and propose a method to improve it.

Both methods can be scalable, highly accurate and parrellizable. These characteristics make the methods able to put into application in the broaden real world, such as node classification, protein function prediction and link prediction.

KEY WORDS: Graph Embedding, latent representation, Deep learning

目　　录

1 绪论

1.1 研究背景

在现实研究中，图结构是一种通用的数据建模方法。例如研究文献引用、社交关系、蛋白质分子结构等问题中，图都可以比较方便的构建数据模型。分析这些网络、提取网络中潜藏的信息在研究显示问题中有很大的意义。因此，对于图的研究一直是一个比较热门的领域，对于图特征的分析在现实应用中也十分重要。

尽管分析图结构是十分重要的，但是大多数图结构分析方法的计算复杂度和空间消耗有比较高。原因在于这些原始数据本身相关信息的存储消耗就已经比较大。而节点之间的关系也比较复杂，往往需要使用稀疏高维邻接矩阵表示，通用的机器学习方法难以直接对原始数据进行处理。研究者在如何加速图数据处理也提出了一些方法，例如分布式图数据结构 (GraphX[1] 等) 以及更加高效的图存储方法。

针对这一问题，除了在图结构本身的存储和处理上进行加速，有研究者提出使用图嵌入这一高效的方法。图嵌入的目的在于尽可能维持原图像拓扑结构和其他潜在特性的基础上，将原始高维稀疏向量映射为低维稠密向量。根据现实图问题的要求可以进行不同的嵌入方式。在大型图结构上进行图嵌入，从而进行预测的方法已经被证实是十分有效的。[2]

一个典型的例子是在空手道俱乐部分类问题（karate graph）进行二维图嵌入。[3] 效果如下：

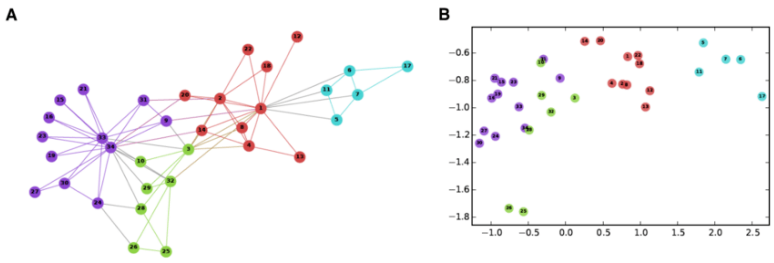

图 1-1: 在空手道俱乐部问题图像 (karate graph) 上进行二维图嵌入

在空手道俱乐部分类问题中，总计有四个不同类别，用四种不同颜色表示。每个节点代表一个俱乐部成员，节点之间的连接边表示社交关系。可以看到，原图像是一个比较复杂的拓扑图结构。在进行图嵌入映射至二维空间后，

节点直接在二维空间上的距离代表节点之间的相似度。可以看到，不同的节点在空间上根据连接关系互相靠近或远离，节点根据自身类别进行汇聚，原先复杂的分类边界在进行图嵌入之后变得清晰。

进行图嵌入的有下面几点优势[4]：

1. 大部分机器学习算法针对低维向量，不适合直接在图上运行。进行图嵌入扩展了现有机器学习算法应用。

2. 在维持图拓扑结构和隐藏信息的情况下压缩数据。直接使用邻接矩阵等表示方式直观上较为方便，但是消耗了大量存储空间的情况下并没有保存与之相配的信息。图嵌入节约了存储空间，同时信息损失在可接受范围内。

3. 处理图嵌入得到的向量比直接处理图数据要方便很多。

4. 图嵌入的可视化效果较好，可以方便发现图数据的一些潜在特性。

图嵌入的结果可以直接用于解决问题。[5] 例如节点分类问题中，图节点依据各自的属性、标签和拓扑结构确定自己在图嵌入后的相对位置。具有相同标签的节点在嵌入过程中相互靠近，不同标签的节点在嵌入中自发的相互远离。对嵌入后得到的图进行线性分割可以快速处理节点分类问题。

图嵌入也可以在其他算法中作为初始数据的预处理方法。[6] 例如商品推荐算法中，首先需要研究用户选择商品的倾向。依据用户的购物历史，可以按照用户在某一时间段选择商品的顺序构建有向图。但是对于大型企业而言，用户数量和每个用户平均购物数量都比较打，直接对图数据操作是不现实的。在这样的情况下可以使用图嵌入预先处理数据，方便后续的研究工作。

虽然图嵌入有诸多的优势，但是实现的过程中往往面临许多困难[4]：

1. 属性选择。现实图模型往往有海量的节点属性。图嵌入尽管尽可能保留的图属性和拓扑结构，但是仍然会丧失一些属性。选择保留哪些必要属性十分困难。在面临不同问题时，同一属性的重要程度也不一样。属性选择的好坏直接影响到图嵌入的效果。

2. 可扩展性。大多数真实网络十分庞大，包含了大量节点和连接边。为了减少算法运行次数从而减少开销，图嵌入技术需要有好的可扩展性。

3. 嵌入维数选择。嵌入维数过低会导致大量信息丢失，嵌入维数过高则会导致嵌入过程的空间时间复杂度过高，造成冗余，选择一个最佳嵌入维度十分困难。

1.2 相关工作

在[7]中提出，可以根据图嵌入的问题形式与嵌入工具，对现有的图嵌入方法进行分类。

首先是问题形式，根据输入形式与输出形式可以进行细分类。

根据输入可以分为：

同构图 节点的实际含义相同，边的实际含义相同
异构图 节点或边至少有一个有大于等于两种实际含义
含辅助信息的图 除了结构信息外还包含了其他信息，比如节点标签，节点特征，信息传播路径等。
无关信息构建图 图中的边由节点之间是否相似进行决定，往往是人为标记的，可能并没有实际的物理意义。

根据输出可以分为：

节点嵌入 将节点的特征从高维映射至低维
边嵌入 将边特征从高维映射至低维
混合嵌入 节点和边特征均需要从高维映射至低维
全图嵌入 将整个图映射为一个低维向量

下图演示了针对一个简单演示图，不同目标的映射结果。

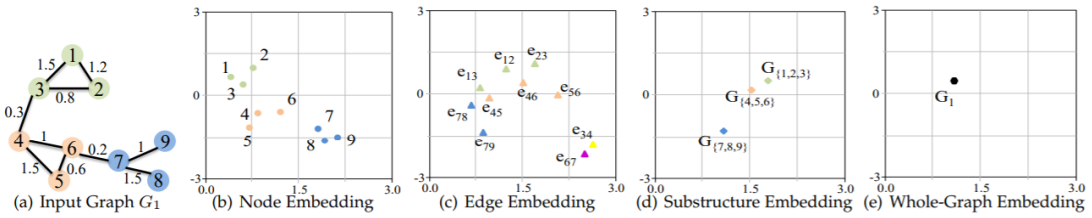

图 1-2: 一个演示

在图嵌入中，在含辅助信息的图中进行节点嵌入是研究比较深入的（这种同一般满足同构的要求，而且还拥有节点属性、节点特征）。因此，后面对研究进展的进展也主要根据这种图类型进行阐述。

根据给定的图特征，在图嵌入方法中，有几种比较主流的方法：

1. 基于矩阵分解的方法（早期）
2. 基于图卷积网络（Graph Convolutional Network）的方法
3. 基于自然语言处理（尤其是 skip-gram 算法）的随机游走方法

矩阵分解方法是图嵌入早期比较常用的方法。矩阵分解方法使用矩阵形式表示图的属性（例如节点的相似性，节点属性等），通过对矩阵的分解获得图的嵌入形式。分为图拉普拉斯分解和节点的相似性矩阵分解。

在矩阵分解方法中认为，矩阵代表了节点之间的相似度，因此相邻节点应该尽可能接近。换言之，节点之间的近似值越大，嵌入结果应该越近似，他们之间的差值就应该有比较大的惩罚值。因此可以视为下面的最优解问题：

$$y^* = arg\min \sum_{i \neq j} (y_i - y_j)^2 W_{ij} = arg\min y^T L y \tag{1-1}$$

其中，W_{ij} 是定义中节点 v_i 与节点 v_j 的相似度，$L = D - W$ 是图的拉普拉斯矩阵，D 为原矩阵的度矩阵，即 $D = \sum_{j \neq i}^{W_{ij}}$。在实际使用时，一般定义 $y^D L y = 1$，因此，优化问题被进一步优化，可以修改为：

$$y^* = arg \min_{y^T D y = 1} y^T L y = arg\min \frac{y^T L y}{y^T D y} = arg\max \frac{y^T W y}{y^T D y} \tag{1-2}$$

因为实际上是求 $\lambda_{\max}$，使得 $Wy = \lambda Dy$，因此最大值应该与矩阵最大特征值相关。这一范畴内的不同方法的区别在于使用的相似度矩阵不同。

因为这个方法所有的节点都需要参与计算，所以这个方法的可扩展性相对比较差。现实使用时需要额外定义线性函数对原嵌入结果进行扩展。这里不再进行详细描述。

另一种近似度矩阵本质与拉普拉斯矩阵分解差别不大，不同之处在于这种方法认为拉普拉斯矩阵不能反应矩阵的特征。在比较早期的 MDS 方法[8] 使用的是特征矩阵 X_i 与 X_j 之间的欧几里得距离进行优化。相对比较近期的方法如[9] 则是主要使用回归模型来保持全图的拓扑结构。

第二种方法是的图卷积网络方法（Graph Convolutional Network）。图卷积网络方法依然沿用了矩阵分解中用近似度矩阵表征节点近似度的方法。图卷积网络方法中近似度矩阵使用邻接矩阵。因此最优问题可以修改为：

$$y^* = arg\min \sum_{i \neq j} (y_i - y_j)^2 A_{ij} = arg\min y^T L y \tag{1-3}$$

在图卷积网络方法最早的 GCN 模型中，Thomas N. Kipf 和 Max Welling 在针对节点分类问题时首先提出[10]，参考二维图像卷积，提出堆叠 K 层卷积层网络，每层卷积层视为一个图在频域上的一次卷积[11]。由于频域卷积本身计算复杂，经过大量数学计算的简化后，可以修改为空间域汇集一阶邻居信息的方法，通过循环训练使得每个节点获得全图的拓扑信息。在[12] 中也采取了类似的

空间域卷积方法。

每一个卷积层的传播规则定义如下（H 代表所有节点汇总的特征矩阵）：

$$\begin{aligned} H^{i+1} &= f(H^i, A) \\ &= \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^i W^i) \end{aligned} \tag{1-4}$$

通过多层传播训练，每一个节点逐步获得全图信息，特征类似的节点自发进行聚合。

GCN 方法的主要思想在于堆叠 K 层网络，每层网络针对每个节点的邻居域进行特征汇聚。通过训练隐藏层参数调整邻居域节点在特征汇集中的权重来实现节点嵌入。直观而言，GCN 的模型如下：

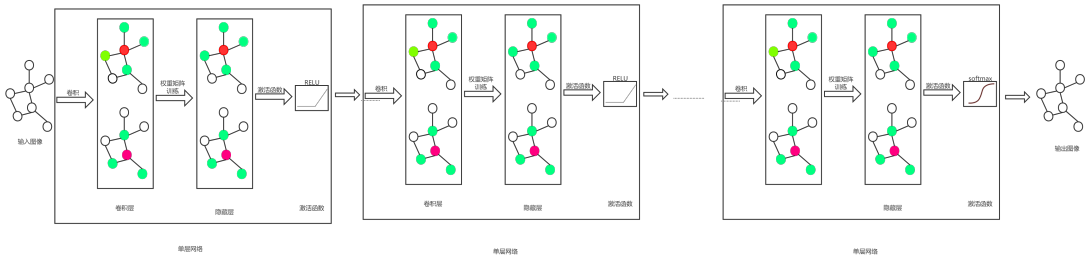

图 1-3: GCN 模型

在后续研究中，例如，斯坦福大学 William L. Hamilton 等人提出，普通的卷积方法过于简单，改用聚合函数[6]，实现无监督图卷积网络的方法 Gragph-Sage。相比于普通的卷积层，GraphSage 调整的参数不再是权重矩阵，而是训练每一卷积层的聚合函数，本质上依然可以视为隐藏层权重的调整。GraphSage 相较于 GCN 方法有更强的可扩展性。

国内也对图卷积网络方法进行了探索。复旦大学 Shengzhong Zhang 等人从连通图分割这一 NP-难问题获得启发[13]，提出在训练过程中，使用多层滤波模板代替正样本选择。针对负样本训练 K 层隐藏层权重矩阵，参考连通图分割问题建立损失函数。由于正样本选择是图嵌入的难点，这一尝试扩展了图卷积网络方法在无监督学习中的应用。

最后一个思考方向来自于机器学习在自然语言处理领域的成功。自然语言处理另领域中，word2vec[14] 是一个比较著名的自然语言处理模型。对于 word2vec，比较重要的两个模型是 skip-gram 模型和 CBOW 模型。其中，skip-gram[15] 模型对于随机游走的图嵌入方法是主要的启发。skip-gram 算法目的在于尽可能扩大大小为 m 的窗口内单词的共现概率。skip-gram 模型没有采用常规通过上下文推测词语的方法。在自然语言处理模型 skip-gram 中，给定语料

库和语句，根据窗口大小 m 提取训练样本，计算下面的概率：

$$Pr(\{v_{i-m},...,v_{i+m}\} \setminus v_i \mid f(v_i)) \tag{1-5}$$

根据概率计算得到图的节点嵌入结果。

考虑到稀疏图和文本的相似性，Bryan Perozzi、Rami Al-Rfou 和 Steven Skiena 提出将 skip-gram 的方法应用到图嵌入中的 DeepWalk 方法。[6] 考虑到 skip-gram 模型本身针对的一维的文本，而图本身并不是一维结构，他们提出在图中进行随机游走的挑选合适的路径。每一条被选择的路径均可以视为一条语句，将所有的节点视为语料库训练 skip-gram 模型，使用该模型的输出对图结构进行嵌入。考虑到图节点本身无法参与计算，作者为每个节点定义 d 维（d 远小于节点数量 V ）向量参与训练。DeepWalk 在实际应用中采用下面的最大化问题：

$$minimize_f - \log Pr(\{v_{i-m},...,v_{i+m}\} \setminus v_i \mid f(v_i)) \tag{1-6}$$

其中，m 是控制随机路径长度的窗口大小。

随机路径方法的本质是将图中所有的节点视为语料库，根据某些方法从图中选择出合适的随机路径作为语句，用 word2vec 的方法对语句进行训练，最终得到一个合适的嵌入结果。DeepWalk 的模型（skip-gram，LSTM 聚合函数[16]）为后面类似的方法提供了思考。后面有相当一部分方法沿用了 DeepWalk 的结构，但是修改了随机路径的选择方法（[17]，[18]）或者近似性的定义（[19]，[20]，[21]）。其中，LTSM 使用比较多的原因是在某些图中可能需要嵌入的不只是一个独立的节点，而是一个固定长度的向量。例如，[22] 中问题节点和回答节点需要被视为一个整体，这时 LSTM 可以发挥作用。

Deepwalk 的方法思路直观如下：

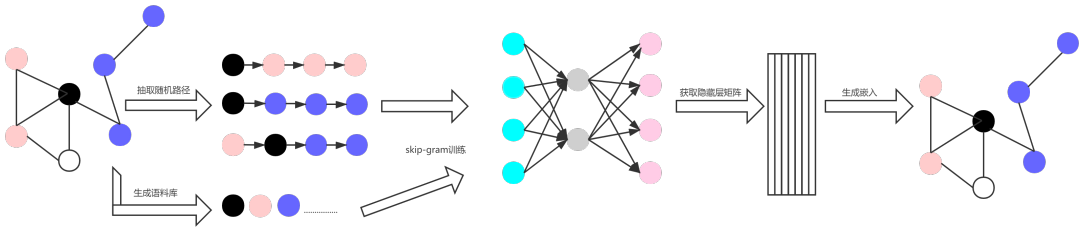

图 1-4: DeepWalk 思路图

后续研究修改路径的一个典型是[23]。文章提出 DeepWalk 完全随机的路径选择不够合理。他们参考深度优先搜索和宽度优先搜索，提出 DeepWalk 的完全随机的路径选择过程可进一步优化。参考 BFS 和 DFS 问题，通过添加变量路径回退概率 p、路径前进概率 q，调整每次随机游走路径的选择倾向，尽可能用不同的方式遍历整个图。依据这样的思考，文章提出使用概率计算每一次跳

跃的概率 $Pr(t,x) = weight(t,x) \cdot w_v x$，其中，$weight(t,x)$ 定义为：

$$\beta_t(s) = \begin{cases} \frac{1}{p} & if\ d_{tx} = 0 \\ 1 & if\ d_{tx} = 1 \\ \frac{1}{q} & if\ d_{tx} = 2 \end{cases} \tag{1-7}$$

近似性定义修改的一个典型例子是[24]。研究者提出传统的随机游走方法仅针对了同质化的网络，难以处理节点属性和边含义各异的异质网络。在同质化网络中，每一条连接边都具有相同的抽象含义，每一个节点属性和标签的维数一致。而在现实情况中这一点往往难以保证。例如引用网络中，文章可能与作者、研究机构、发布文章等各异的节点发生联系，这几个关系在图中均可以用连接边表示，但是这些连接边代表室现实的含义并不完全一致，也会导致节点和边的属性并不是同一维度。研究者提出针对异质化网络，将原先的 skip-gram 方法调整为针对异质化网络的 heterogeneous skip-gram 语言处理方法，在此基础上进一步提出 metapath2vec 方法和 metapath2vec++ 方法进行节点嵌入，扩展了随机游走方法在异质化网络上的应用。

1.3 本文主要工作

本文旨在对图嵌入问题的方法进行整理和一定的优化。主要工作如下：

1. 对图嵌入问题进行规范的定义

2. 对图卷积网络方法和随机游走方法过程进行详细解释，并做出一定程度的改进

3. 在标准公开的数据集上实践图卷积网络方法和随机游走方法

4. 汇总目前工作仍然存在的局限性，提出未来工作的方向

1.4 本文结构

本文后续章节的结构如下：

第一章：绪论。简要说明了图嵌入问题的现实背景和研究进展，概括本文的工作。

第二章：图嵌入问题中一些基本概念定义，图嵌入问题定义及图嵌入问题实例

第三章：图卷积网络方法和随机游走方法原理、算法及优化

第四章：实验过程。介绍了实验进行的配置环境，使用的算法，图嵌入的结果，探究图嵌入方法在噪声情况下的表现并提出解决方案。

第五章：总结和讨论。总结全文内容，提出目前方法仍然存在的问题，提出未来的研究方向。

2 图嵌入问题定义

2.1 基本定义

首先，介绍图嵌入问题的一些基本定义。

Definition 1 一个图 G 是一个偶对 (V, E)，其中，$v \in V$ 是图中的一个节点，$e \in E$ 是图中的一条边。G 中有节点类型映射函数 $f_v : V \to T^v$ 和边类型映射函数 $f_e : E \to T^e$

T^v 和 T^e 分别表示图中的节点类型集合与边类型集合，图 G 中的每一个节点 $v \in V$ 属于 T^v 中的一个特定元素类型。对于边同样如此。

Definition 2 同构图 $G_{homo} = (V, E)$ 是一个图，满足 $|T^v| = |T^e| = 1$。图 G 中的所有节点均属于一个类别，所有边均属于另一个类别。

简单理解是，同构图中所有节点、所有边的物理意义相同。例如在引文网络中，所有的节点均表示一篇文章，所有边均表示引用关系。

Definition 3 异构图 $G_{heter} = (V, E)$ 是一个图，满足 $|T^v| > 1$ 或 $|T^e| > 1$。

异构图的特点是，他的边类型或节点类型可能不止有一类。再次以引文网络为例，图中的节点的物理含义可能有文章、作者、出版社、机构等多种类型，边的物理含义也可能有多种物理含义。一个异构图的例子如下：

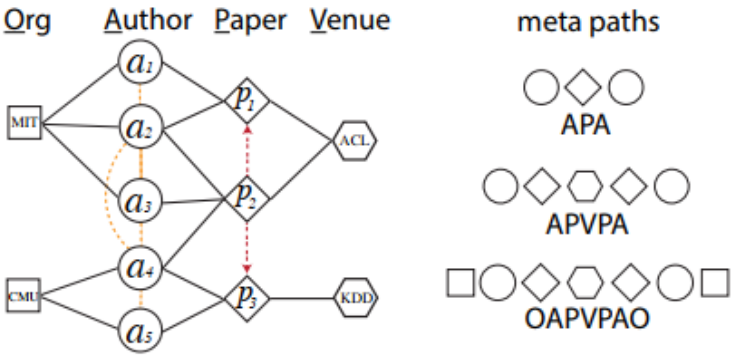

图 2-1: 一个引文网络异构图的示例

可以看到，这个图中的节点类型超过了 1，是一个异构图。

Definition 4 节点 v_i 与节点 v_j 的一阶相似性是边 e_{ij} 的权重，即 W_{ij}

由于我们针对同构图，$W_{ij} = A_{ij}$，因此我们的一阶相似性取决于节点之间的连接情况。可以定义 $s_i^1 = A_{i1}, A_{i2}, ..., A_{in}$。从而我们定义图的二阶相似：

Definition 5 节点 v_i 与 v_j 的二阶相似性是节点 v_i 与节点 v_j 的邻域相似性。即 $s_{v_i}^1$、$s_{v_j}^1$ 的相似性。

类似的可以定义更加高阶的相似性。

2.2 图嵌入问题定义

这里给出图嵌入问题的定义：

Problem 1 给定图 G = (V, E) 和预先确定的嵌入维度 d($d \ll |V|$)，图嵌入问题是将图 G 映射至 d 维空间，同时尽可能保持图的属性。图的属性可以由一维相似性或更高阶相似性量化。

需要说明的是，这里的目标是“将图 G 映射至 d 为空间”，但是评价结果的好坏需要用“图的属性是否被很好的保持”来判断。一般情况下，评判图属性保持的优劣使用标签预测的准确率。因此在实际算法中，往往在输出时需要额外添加一层到标签的隐藏层。实验中的表面输出是预测标签的结果。但是实际需要的是这一用于转换的隐藏层的 d 维向量表示。这种方法被一些研究者称为“伪任务”。[14]

参考前文针对不同目标的映射结果，我们需要的结果是第一种对节点进行嵌入输出：

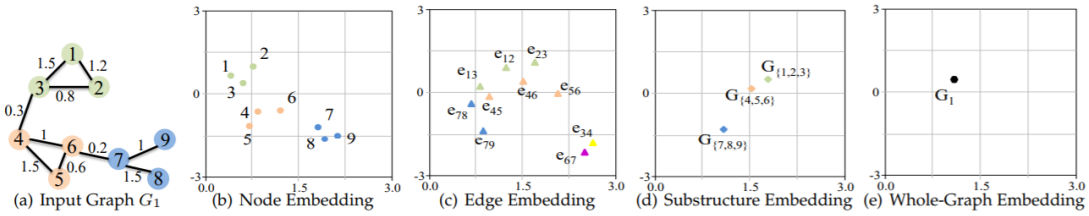

图 2-2: 一个演示

这里拟定采用两种误差计算方法：

平方差 $Loss = L_0 + \lambda L_{reg} = L_0 + \sum_{i,j} A_{ij} \|f(x_i) - f(x_j)\|^2$

交叉熵误差 $Loss = -\sum_{l\in Y_l}\sum_{f=1}^{F} Y_{lf}\ln Z_{lf}$

其中，平方误差之前有介绍过，是在图拉普拉斯方法和 GCN 方法中主要采用的误差计算方式。交叉熵误差则是一种仅与标签相关的误差，尽可能只关注可能性最大的标签值，在分类预测问题中表现更好。

2.3 图嵌入问题实例

以在空手道俱乐部分类问题（karate graph）进行二维图嵌入为例。[3] 左侧是问题的输入图 $G = (V, E)$。输入嵌入前节点数 n，所有节点的特征矩阵 F，实际标签 L，邻接矩阵 A。根据这些信息可以构建左侧的拓扑图。

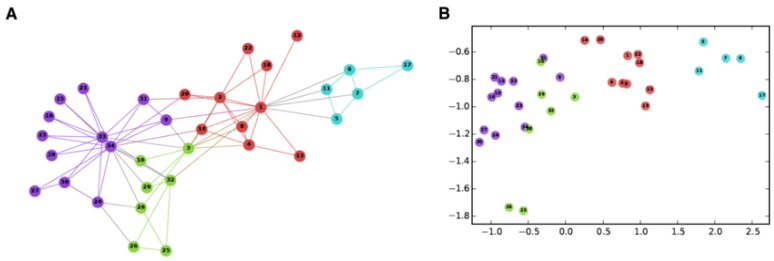

图 2-3: 在空手道俱乐部分类问题（karate graph）上进行二维图嵌入

假设嵌入后所有节点的特征矩阵为 f，根据 f 预测的标签为 L2。那么图嵌入的目标如下：

$$\begin{aligned} arg\min Loss &= arg\min(L_0 + \lambda L_{reg}) \\ &= arg\min(\sum_{i=0}^{n}(L2_i - L)^2 + \lambda\cdot\sum_{i,j}A_{ij}\|f(x_i) - f(x_j)\|^2) \end{aligned} \tag{2-1}$$

其中，L_0 表示所有节点的标签误差，一般情况下只包含训练用节点。L_{reg} 表示嵌入后根据邻接情况得到的相似性损失，λ 是自己选定的损失权重。也就是说，误差计算方式需要兼顾标签值预测的准确性和嵌入后的相似性损失两个部分。在实际算法中，可以通过调整嵌入损失的权重来判断嵌入的准确性。

图嵌入的结果则如右侧所示，是一个相对低维的向量嵌入结果，即嵌入后的特征矩阵 f。由于我们只进行节点嵌入，连接边是同一属性的，因此连接边的嵌入结果仍然是邻接矩阵 A。A 在实际应用中可以用稀疏矩阵表示。

3 图嵌入方法分析

前文已经说明，图嵌入有图卷积网络方法和基于自然语言处理的随机游走方法两种主流方法，这里分别介绍两种方法的细节。

3.1 图卷积网络方法

图卷积网络方法认为，学习一个满足要求的图嵌入映射 H，对于图中的每一个节点 v 而言，相当于学习一个函数 f，f 将每一个节点的特征矩阵映射到一个更低的维度。由于之前提到过，学习图节点的标签只是一个“伪任务”，最后我们需要的嵌入结果维数并不一定就是标签的维数，因此一个合理的思考是，通过堆叠多层卷积网络，逐步对图中的特征矩阵进行嵌入降维。最终使用模型进行嵌入时，我们只需要舍弃最终的输出层，保留倒数第二层之前的隐藏层即可。

所以我们需要的是，定义 K 个卷积层，通过堆叠这 K 个卷积层并进行循环来进行图嵌入训练。因此核心就是如何定义每个卷积层的嵌入函数 f 来进行图嵌入。

3.1.1 图卷积的定义

卷积是数学中的概念，简而言之，对于函数 f 和函数 g 而言，卷积计算按照如下的定义：

$$(f * g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dx \tag{3-1}$$

当函数 f 与函数 g 为离散函数，且长度有限时，定义如下：

$$(f * g)[n] = \sum_{m=-M}^{M} f(n-m)g(m) \tag{3-2}$$

在二维图像处理中，往往可以使用卷积的方法进行图像特征的提取。在二维图像中，针对图像中一个特定位置 (x, y) 二维离散的空间域卷积可以定义为：

$$filter2d[x][y] = \sum_{i=x-k}^{x+k} \sum_{j=y-k}^{j+k} pic[i][j] \cdot w[i][j] \tag{3-3}$$

可以看到，这个空间卷积核的本质就是针对图像中的每一个像素 (x, y)，定义该元素周围从 $(x-k, y-k)$ 到 $(x+k, y+k)$ 的矩形像素框。对矩形框中的每一个像素，为他赋予对应的权重，将原像素与权重的乘积之和重新赋予到该像素的位置。

但是，拓扑图的特殊之处在于，因为每个节点周围的邻居节点并不是固定的，所以无法像上面的步骤，通过指定一个特定大小的卷积核就可以进行卷积操作，也不能通过直接指定一个坐标提取出节点位置。但是，考虑到卷积的本质就是指定一个节点的 k 阶邻居并对他们赋予一定的权重，对特征矩阵与权重矩阵点乘的结果进行处理，我们可以这样定义普通拓扑图的空域卷积：

$$filter_{graph} = \sum_{i=0}^{|N(v)|} \sum_{u \in N(v)} feature(u) \cdot w_i \tag{3-4}$$

本质上可以看成对节点 v 定义一个邻居的范围 $N(v)$，假设需要的 k 阶邻居，只需要定义 $N(V) = \{u | dis(u, v) \leq k\}$ 就可以了，dis 函数表示节点之间最短路径的长度。

同样的，也存在谱域的图卷积定义，在[10] 中有详细的数学推导，这里由于不进行使用，所以不做讨论。

3.1.2 卷积层生成

在这里，我们描述图卷积网络方法如何生成卷积层（换而言之，前向传播算法）。在介绍图卷积网络方法的理论时，首先假设，下面提到的模型已经训练完成，并且参数固定。

在图嵌入中过程中，对于每个节点而言，经过一层卷积层可以认为是经过了一个函数 f，对于每个节点 v，$f(v)$ 就是图嵌入的结果。显而易见，对于一个图而言，可输入的参数有矩阵的特征矩阵 F，标签矩阵 L 以及邻接矩阵 A。因此，嵌入函数 f 应该是一个 $f : f(F, A) \rightarrow L$ 的函数，可以理解为全图的特征矩阵与连接情况的聚合。

比较容易理解的是，如果图中的两个节点有边相连，那么他们经过聚合后属于同一个类别的概率应该要高于没有直接相邻的节点。以论文引用网络为例，如果 A 节点与 B 节点有一条有向边，那么代表 A 节点引用了 B 节点，那么 A 与 B 很可能会落在同一个论文分类中。因此，A 与 B 的特征矩阵经过聚合后应该也是比较接近的。

因此，一个合理的想法是，在每次卷积层训练时，每个节点仅考虑 K 阶邻居的特征。由于训练的参数是固定的，每一个节点 v 可以通过下面的算式来得到聚合后自己的特征：

$$
\begin{aligned}
h_u &\leftarrow w1_u \cdot f_u \forall u \in N(v) \\
output_v &\leftarrow w2_v \cdot AGGREGATE(h_u, \forall u \in N(v))
\end{aligned}
\tag{3-5}
$$

其中，$N(v)$ 代表节点 v 的邻居（为了表示的简便，我们假设每个节点也是自己的邻居），f 表示各个节点对应的特征。这个过程假定已经知道了聚合过程中，每个邻居节点的特征聚合后的权重 w1。因此首先给每个邻居节点分配权重 w1 后，加入聚合函数 AGGREGATE 整合每个节点的邻居特征，并最终通过 w2 矩阵将原先得到的特征进行图嵌入的降维，得到嵌入之后的特征矩阵。

连续两次权重矩阵计算可以通过矩阵乘法的规则进行简化，得到这样的前向传播规则：

$$
output_v \leftarrow \sigma(w_v \cdot AGGREGATE(h_u, \forall u \in N(v))) \tag{3-6}
$$

因此我们可以得到这样前向传播的算法：

算法 3.1 图卷积网络嵌入方法层与层之间的前向传播部分

输入: 图 G = (V, E)；特征矩阵 $F = \{x_v, \forall v \in V\}$；已知节点的 label 矩阵 $L = \{y_v, somev \in V\}$；卷积层层数 K；权重矩阵 $W^K, \forall k \in \{1, ..., K\}$；非线性激活函数 σ；邻居函数 $N : v \rightarrow V$；聚合函数 AGGREGATE

输出: output 所有节点的标记标签矩阵 z_v

$h_0 \leftarrow x_v$
for k from 1 to K **do**
  for $v \in V$ **do**
    $h^k_{N(v)} \leftarrow AGGREGATE(\{h^{k-1}_u\} \forall u \in N(v))$
    $h^k_v \leftarrow \sigma(h^k_{N(v)} \cdot W^k)$
  end for
end for
return $z_v \leftarrow h^K_v$ for all $v \in V$

上述算法描述了对于一个图 G 如何定义卷积层的传播规则。最外层的 K 层循环表示总计有 K 层卷积层，k 表示当前步骤所在的卷积层序号，h^k 表示一个节点当前的特征。首先，对于每一个节点 v，通过聚合函数 AGGREGATE 聚合当前节点所有邻居的特征，并用 $h^k_{N(v)}$ 表示（这里将 v 自身也视为 v 的邻居），得到的聚合特征参与矩阵的训练。每一个的输出矩阵对应下一层的输入矩阵。这样，训练时矩阵可以逐步进行维度转化。在第 K 层，输出的特征矩阵正好对应的预测每个节点对应的标签矩阵。在实际操作中，节点对应的标签用独热编码 one-hot 进行计算，因此每个节点的输出特征应该与类别总数一致，矩阵内每一个数据代表预测中节点落在此类别的概率。

通过这样的算法，在每一个卷积层，每个节点通过聚合邻居节点的特征得

到当前层自身的节点特征。通过在训练参数的过程中不断地进行循环，每个节点都可以逐步获得其他不在邻居节点范围内的其他节点的节点特征。通过这种扩散方式，每个节点都可以获得全图的节点特征和拓扑信息。

在最初的 GCN 算法中，每一个卷积层的传播规则定义是如下（H 代表所有节点汇总的特征矩阵）：

$$\begin{aligned} H^{i+1} &= f(H^i, A) \\ &= \sigma(\hat{D}^{\frac{-1}{2}}\hat{A}\hat{D}^{\frac{-1}{2}}H^iW^i) \end{aligned} \tag{3-7}$$

在不考虑归一化问题的情况下，实际传播规则即为：

$$\begin{aligned} H^{i+1} &= f(H^i, A) \\ &= \sigma(\hat{A}H^iW^i) \end{aligned} \tag{3-8}$$

对于每一个节点来说，相当于对所有的一阶邻居特征进行求和，本质上是定义邻居为一阶邻居，聚合函数为求和函数的情况。归一化的过程是为了防止训练过程中矩阵的膨胀。

可以看到的是，图卷积网络方法是可以设置为小样本（mini-batch）的归纳式算法。在进行邻居节点的特征聚合时，可以通过采样的方法，定义选择邻居的上限 T，随机的挑选每个节点的至多 T 个邻居，聚合他们的特征，可以生成这样的算法：

算法 3.2 图卷积网络嵌入方法层间前向传播部分小样本版

输入： 图 G = (V, E)；特征矩阵 $F = \{x_v, \forall v \in V\}$；已知节点的标签矩阵 $L = \{y_v, somev \in V\}$；卷积层层数 K；权重矩阵 $W^K, \forall k \in \{1, ..., K\}$；非线性激活函数 σ；邻居函数 $N : v \to V$；聚合函数 AGGREGATE

输出： output 所有节点的标记标签矩阵 z_v

$h_0 \leftarrow x_v$
for k from 1 to K **do**
  for $v \in V$ **do**
    $total^k_{N(v)} = list(h^{k-1}_u, \forall u \in N(v))$
    $sample^k_{N(V)} = RandomPermutation(list(h^{k-1}_u, \forall u \in N(v)))$
    $h^k_{N(v)} \leftarrow AGGREGATE(\{h^{k-1}_u\}\ \forall u \in sample^k_{N(v)})$
    $h^k_v \leftarrow \sigma(h^k_{N(v)} \cdot W^k)$
  end for
end for
return $z_v \leftarrow h^K_v$ for all $v \in V$

3.1.3 损失函数

对于半监督的分类问题，我们使用损失函数：

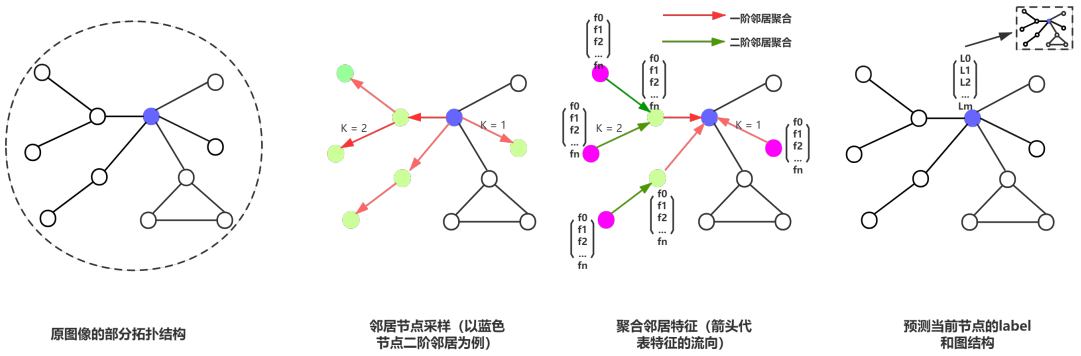

图 3-1: 图卷积网络方法取样和聚合的直观描述

$$Loss = -\sum_{l \in Y_l} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

其中，Y_l 表示所有已知标签的节点集合。Y_lf 表示节点在实验中的实际 label。Z_lf 表示预测的标签在对应位置上的概率值。

3.1.4 聚合函数

由于节点本身是没有确定顺序的，因此聚合函数需要作用在一列无序矩阵上。理想情况下聚合函数是对称的（结果与输入顺序无关），同时在可训练的要求下要有比较好的聚合能力。我们尝试了以下几种聚合方法：

均值聚合 我们首先尝试的是均值聚合。这里我们取出一个邻居的所有特征矩阵的平均值。均值聚合相对比较接近最初 GCN 方法。在这里，聚合函数如下：$h_v^k \leftarrow \sigma(W \cdot MEAN(h_u^{k-1}) \ \forall u \in N(v))$

最大/最小值聚合 类似的方法是采用最大值和最小值聚合。这里我们取出一个邻居的所有特征矩阵的最大/最小值。在这里，聚合函数如下：$h_v^k \leftarrow \sigma(W \cdot MAX(h_u^{k-1}) \ \forall u \in N(v))$

采样聚合 在保证正确率的情况下，可以尝试采用聚合部分邻居节点的方法，聚合函数同上。

3.1.5 有关干扰问题的探讨

以最初的图卷积方法为例，GCN 方法对于干扰的存在十分敏感。这里主要考虑特征矩阵受到干扰的情况。不考虑标签矩阵和邻接矩阵受到影响的原因是：标签是使用 one-hot 独热编码进行处理的，如果有波动应该在编码的时候就可以发现；而邻接矩阵只有在图的拓扑结构发生变化时才会改动，而且矩阵内的元素非 0 即 1，一般情况下应该是受到了恶意攻击，不太可能是自然存在的波动。在[25] 中尝试模拟了 GCN 网络遇到针对边进行攻击的情况，有提出自己 Pro-GNN 的解决方案。然而，由于最终节点嵌入的特征矩阵本身就应该是一

个浮点类型的矩阵，在不知道输入数据的具体内容时是输入数据是否受到污染确实是不容易进行判断的，所以这里主要探讨对于节点特征的干扰。

在文章实际操作实验中，用初始的 GCN 网络针对 cora 数据集进行轮次数 epoch=200 的训练时，如果针对特征矩阵输入一个 0 0.05 的均匀干扰，GCN 网络进行判断的精度直接从 80% 降低到了 76% 左右。cora 的特征矩阵是 1000 多维的一个非 0 即 1 的特征矩阵，0.05 的按照道理来说而言是很小的误差，然而对于 GCN 的预测结果直接产生了很大的影响。因此，思考如何处理 GCN 的抗干扰能力是十分有必要的。

这里我们重新回顾 GCN 的方法，对于 GCN 而言，传播函数如下：

$$H^{k+1} = \sigma\hat{A}'H^kW^k \quad = \sigma(D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}H^kW^k) \tag{3-9}$$

为了简单起见，用 $H^{k+1} = \hat{A}\hat{H}W$ 来表示。如果需要包含噪声的话，我们可以对原来的公式进行修改，那么此时，有：

$$\begin{aligned} H^{k+1} &= \hat{A}\hat{H}W \\ &= \hat{A}(H + noise)W \\ &= \hat{A}HW + \hat{A} \cdot noise \cdot W \end{aligned} \tag{3-10}$$

也就是说，对于每一层而言，相当于多加了一个 $\hat{A} \cdot noise \cdot W$ 的分布。很显然的是 $\hat{A}$ 是一个基于邻接矩阵的定值，这不影响噪声本来的分布模式。一个简单的想法是，如果我们认为每层的 W 是训练好的参数，那么额外出现的分布应该与噪声本身的分布一致。所以我们考虑在每一层训练权重矩阵时，额外添加一个参数 t 用于表征噪声的特征。由于噪声本身是一个分布，不能直接参与训练，所以我们训练噪声的参数 t，在每次训练计算损失的时候根据参数 t 随机生成一个训练用噪声参与训练，从理想情况下就应该可以尽可能的平滑噪声对训练的影响。

因此我们可以写出这样的传播规则：

算法 3.3 图卷积网络嵌入方法（排除噪声）

输入: 图 G = (V, E)；特征矩阵 $F = \{x_v, \forall v \in V\}$；已知节点的标签矩阵 $L = \{y_v, somev \in V\}$；卷积层层数 K；权重矩阵 $W^K, \forall k \in \{1, ..., K\}$；非线性激活函数 σ；邻居函数 $N : v \rightarrow V$；噪声参数 t；已知噪声分布 noise；聚合函数 AGGREGATE

输出: output 所有节点的标记标签矩阵 z_v

$h_0 \leftarrow x_v$
for k from 1 to K **do**
  for $v \in V$ **do**
    $h^k_{N(v)} \leftarrow AGGREGATE(\{h^{k-1}_u\} \forall u \in N(v))$
    $h^k_v \leftarrow \sigma(h^k_{N(v)} \cdot W^k)$
    $MyNoise \leftarrow t \cdot noise$
    $NormalizedNoise \leftarrow \sigma(\hat{A}MyNoise \cdot W^k)$
    $output^k \leftarrow CONCAT(h^k_v, NormalizedNoise)$
  end for
end for
return $z_v \leftarrow output^K$ for all $v \in V$

这个方法虽然有一定的效果（后面实验会进行说明），但是提升不是非常大，而且比较要求对噪声的了解，当参与训练的噪声分布与实际不一致的时候反而会降低表现。

在 GCN 原作者 Thomas N. Kipf 等人对 GCN 后续的改进中[?] 也在注意到了 GCN 对于微小扰动敏感的问题，提出需要设计一个更加鲁棒的 GCN 模型。这个模型的困难之处在于既要保证 GCN 原本的高效性，也需要避免邻居聚合中的错误传递。在他们提出的 RGCN 模型中，他们引入的是高斯分布的干扰。

他们的思考方向，当噪声是高斯分布时，每个节点由于都是周围邻居的加权特征聚合，应当也满足高斯分布，且均值与方差和每个随机变量是相关的。因此在训练时，将原先的特征矩阵 H 转化为 $f(H, A)$，每个节点的训练特征从平常的离散值修改为基于离散值的高斯分布，在训练时通过生成随机数的方法来进行改进。

这个也是一个合理的思考方向，可惜的是在 RGCN 论文的试验表现中，没有很显著的准确性提升。本文在后面使用的基于噪声进行训练产生的结果也不尽如人意，所以目前仍然是一个值得思考的方向。

3.2 随机游走方法

在随机游走方法中同样认为，如果两个节点之间有连接边，那么他们的嵌入结果应该尽可能的接近。与之前的图卷积网络方法不同之处在于，图卷积网络方法通过聚合函数或者卷积来获取周围邻居的特征信息，而随机游走方法则

是在图中随机的抽取路径，对抽取的路径进行训练。

在对 Youtube 社交网络随机抽取路径结果与 Wikipedia 文章的语料库分布对比后发现，他们的频率分布是比较接近的。而在处理一维文本时，已经有一个可用的训练模型 skip-gram。如果将图中的所有节点视为语料库，在图中随机抽取路径作为可以进行分析的语句，利用 skip-gram 进行训练，那么就可以做到嵌入的效果。

随机游走方法的步骤是：首先，提取图中的节点，对每一个节点生成一定数量的随机路径，将随机路径生成的结果作为输入建立神经网络，训练隐藏层的权重，最终转化为低维向量。

3.2.1 随机路径

我们定义一个从节点 v_i 产生的随机路径为 $Path_{v_i}$。产生的路径可以描述为 $Path_{v_i} = \{N_{v_i}^1, N_{v_i}^2, ..., N_{v_i}^k\}$，其中，$N_{v_i}^i$ 表示第 i 次选择的节点，要求是 $N_{v_i}^i$ 是 $N_{v_i}^{i-1}$ 的一阶邻居。

这样对随机路径进行取样，除了充分利用节点周围的图拓扑信息外，还有下面几点好处：

支持并发 选择随机路径的过程可以是并发的，可以同时选取多条路径并进行训练，效率可以大幅提升。

可扩展 在新节点加入图中时，只需要针对新节点周围进行合适的路径选择就可以快速捕捉加入新节点后图结构的变化，不需要重新计算整个图。

细节敏感 在图结构发生微小变化时，整个图的拓扑结构可能变化不大，但是对于其中一些随机路径产生的影响就会比较明显。

3.2.2 单条路径训练模型

在自然语言处理中，语句建模的目的是在给定语料库的情况下，尽可能准确的预测一些特定语句出现的概率。更加规范的表述是，给定一个词序列 $W_1^n = (w_0, w_1, .., w_n)$，$w_i \in V$（$V$ 是语料库），我们需要在训练中最大化 $Pr(w_n|w_0, w_1, ..., w_{n-1})$。即，给定一条语句和一个训练位置 x，尽可能增大位置 x 的实际词出现的概率。

在这里，我们仿照自然语言处理定义单语句训练的目标。在给定训练节点 v_i 和训练路径 $v_1, v_2, ..., v_i, v_{i-1}$ 的情况下，我们需要最大化下面的概率：

$$Pr(v_i \mid (v_1, v_2, ..., v_{i-1})) \tag{3-11}$$

由于我们需要学习一个隐藏层表示，而且节点本身是不适合直接计算的，引入一个映射函数 $f: v \in V \rightarrow R^{|V| \times d}$。对于每一个节点 v 而言，$f(v)$ 代表了它的隐藏层表示（也就是后面需要训练的参数）。因此问题转化为预测下面的概率：

$$Pr(v_i \mid (f(v_1), f(v_2), ..., f(v_{i-1}))) \tag{3-12}$$

由于这个概率在选择路径的长度增加或产生变化时会非常难以计算，因此难以直接使用。幸运的是，在自然语言处理中，skip-gram 算法已经解决了这个问题。这里先直接应用 skip-gram 的结论，后续进行详细解释。

实际应用中，放弃上述的概率计算，转而计算下面的最大化问题：

$$minimize_f - \log Pr(\{v_{i-m}, ..., v_{i+m}\} \setminus v_i \mid f(v_i)) \tag{3-13}$$

使用这种方式计算可以加速训练，并且从依赖上下文预测转为通过已知词推测上下文，更好地体现随机路径体现的“邻居”概念。所以可以得到下面的算法：

算法 3.4 随机路径图嵌入方法

输入： 图 G = (V, E)；窗口大小 w；嵌入维数 d；单节点进行的游走次数 n；游走长度 t **输出：** output 所有节点的标记嵌入后的矩阵表示 $f \in R^{|V| \times d}$

初始化 f 矩阵 $from\ Uniform^{|V| \times d}$
for i = 0 to n **do**
  $O = RandomPermutation(V)$
  for $v_i \in O$ **do**
    $W_{v_i} = RandomWalk(G, v_i, t)$
    $SkipGram(f, W_{v_i}, w)$
  end for
end for
return f

这个算法的过程非常清晰。首先用均匀分布初始化 f 矩阵。之后，根据给定的游走次数进行 n 次循环。每次循环首先打乱节点的排序，对每一个含有 label 的节点取一次长度为 t 的随机路径，并进行 $skip-gram$ 训练。最终输出隐藏层矩阵 f。

在 Deepwalk 中使用的就是这样的随机取路径算法。在 Node2vec 中，参考 BFS 理论和 DFS 理论，这里额外定义了前进概率 p 和回退概率 q。这样的优势是可以通过调整参数 p 和 q 来控制游走路径的向外扩展的倾向。具体的分析如下：

原先每次随机路径选择相当于根据路径长度进行 k 次 1 阶跳跃，这里修改为定义 2 阶跳跃。假设随机游走第一次选的边为 (t, v)，现在处于节点 v 的

位置，现在需要决定下一跳进入哪一个节点。这里使用概率计算 $Pr(t,x) = weight(t,x) \cdot w_v x$，其中，$weight(t,x)$ 定义为：

$$\beta_t(s) = \begin{cases} \frac{1}{p} & if\ d_{tx} = 0 \\ 1 & if\ d_{tx} = 1 \\ \frac{1}{q} & if\ d_{tx} = 2 \end{cases} \tag{3-14}$$

3.2.3 skip-gram 算法

skip-gram 算法的结构非常简单，下面是当时提出时 skip-gram 算法的简易架构（另一个是 CBOW，就是根据上下文推导当前词语的）：

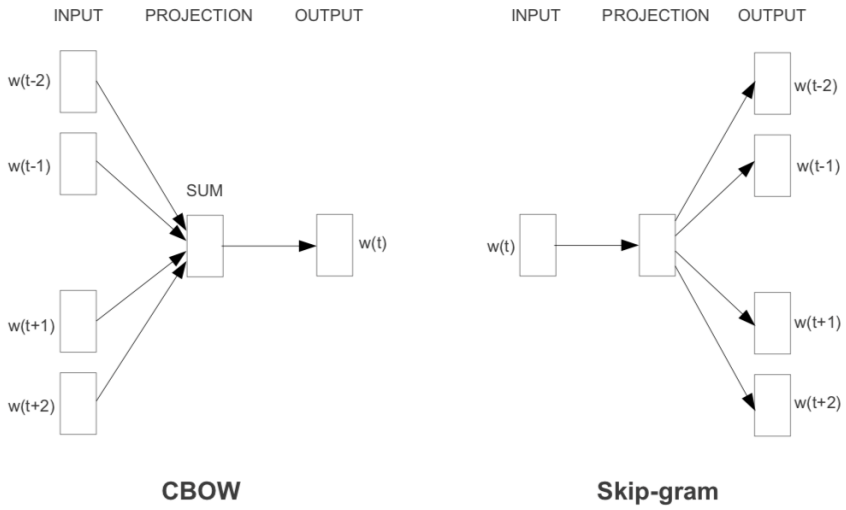

图 3-2: skip-gramy 与 cbow 结构的对比

skip-gram 的训练过程是：

1. 选择句子中的一个输入词 input

2. 定义 skip-window 参数，这个参数决定了 input 向前和向后选取词的数量

3. 取出对应的 input 和 output 对，训练神经网络

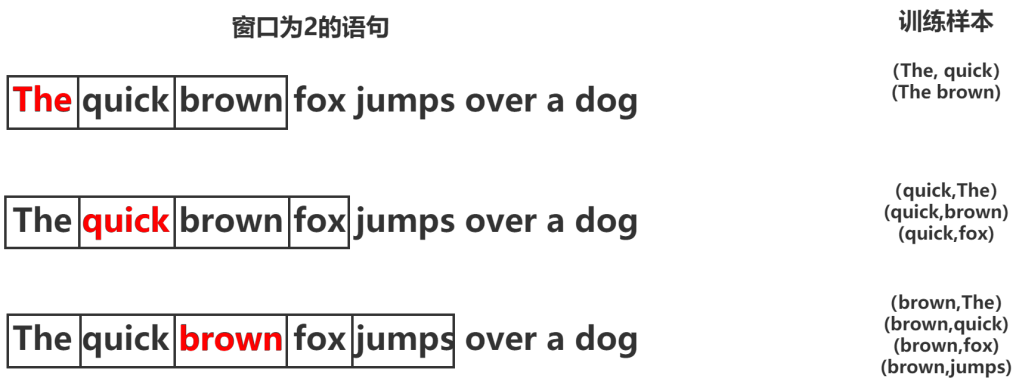

图 3-3: skip-gram 训练的一个实例

上图是 skip-gram 训练的一个实例。我们可以定义一个 m 的窗口大小。对于句子中的每一个词，都根据窗口大小 m 提取训练样本。然后计算下面的概率：

$$Pr(\{v_{i-m},...,v_{i+m}\} \setminus v_i \mid f(v_i)) \tag{3-15}$$

因为需要尽可能增大句子中的词对出现的概率，因此需要尽可能增大训练样本的可能性，从而准确的预测上面的概率。

实际应用中，考虑到词（节点）之间的独立性，概率计算可以做这样的转换：

$$Pr(\{v_{i-m},...,v_{i+m}\} \setminus v_i \mid f(v_i)) = \prod_{j=i-w,j\neq i}^{j=i+w} Pr(v_j|f(v_i)) \tag{3-16}$$

因此可以构建这样的 skip-gram 网络结构：

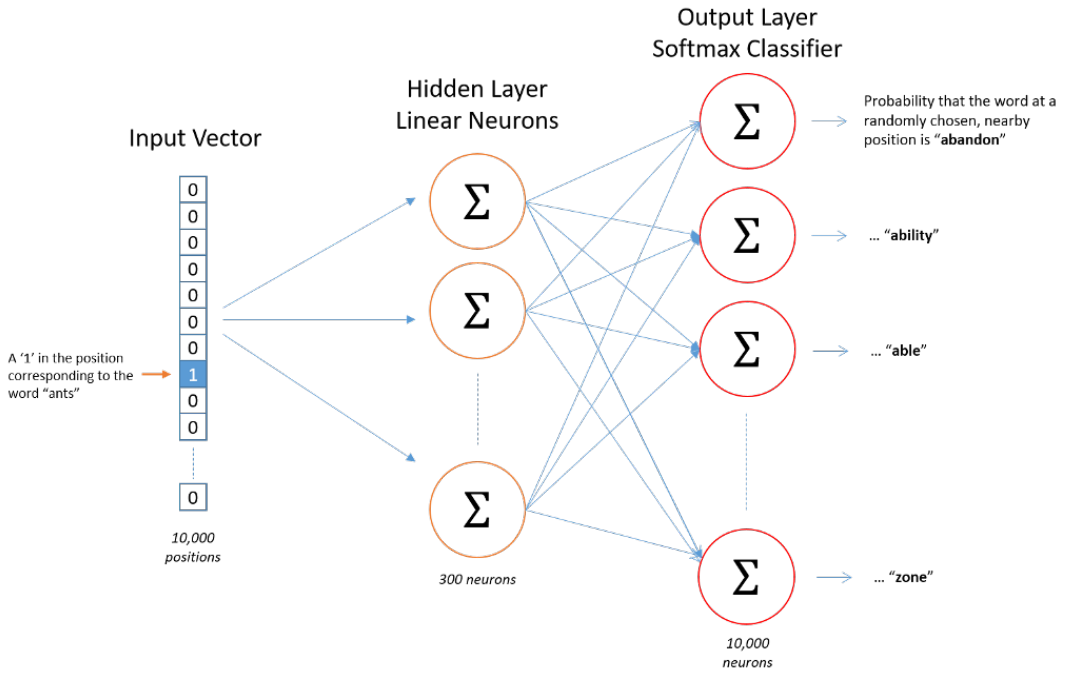

图 3-4: skip-gram 的模型

可以看到，语料库中所有的语句使用 one-hot 编码，之后映射到一个相较于原图节点数较低维度的神经网络上，最终输出时映射回到原先语料库的维度，通过训练输出和路径中提取出的节点对之间的损失来进行嵌入。中间层映射得到的就是词嵌入后的结果。

4 实验过程

4.1 数据集

主要使用的数据集有以下三个：cora，BlogCatalog，Flickr。

4.1.1 cora 数据集

cora 数据集是一个引文网络数据集。节点数 2708。每篇论文都由一个 1433 维的词向量表示，边数 5429，总计标签数为 8。这是一个相对较小的网络

4.1.2 BlogCatalog 数据集

BlogCatalog 数据集是一个社会关系网络数据集，由博主和他的社会关系（好友）组成。节点数为 5196。每个节点有 8189 维特征，边条数为 171743，标签数为 39。[26]

4.1.3 Flickr 数据集

Flickr 是一个来自 Flickr 的图片数据集。他的 feature 是对于图片内容的描述，label 为所属类别。总计节点数 7575，每个节点有 12047 维特征，边数 239738，总类别 195.[26]

4.2 实现细节

实验过程在 pytorch 框架上进行。实验计算机为 1.8GHz 英特尔 i7-8565CPU，8G RAM 的环境下运行。

在 GCN 方法中，使用两层 GCN 网络，中间隐藏层在未指定大小的情况下默认为 16 个 uint，使用的聚合函数包括普通 GCN 网络卷积、最大/最小值聚合（MAX,MIN）以及均值聚合 (MEAN)。图网络结构如下：

同时，针对均值函数方法试验了针对节点随机选取邻居的方法（MEAN-SAMPLING）。这里，当节点邻居数超过 6 时，随机的选择其中的 6 个邻居节

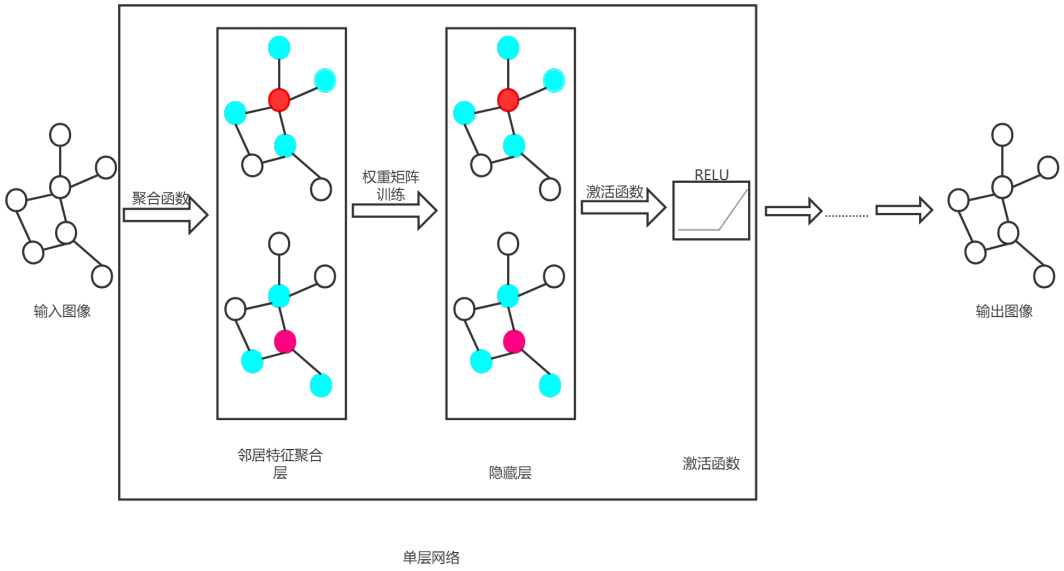

图 4-1: 本文使用的 GCN 网络模型

点，进行均值聚合。这种方法可以视为一种 mini-batch 的方法，在实际应用中可以加速特征提取。

选择两层 GCN 基于如下原因：

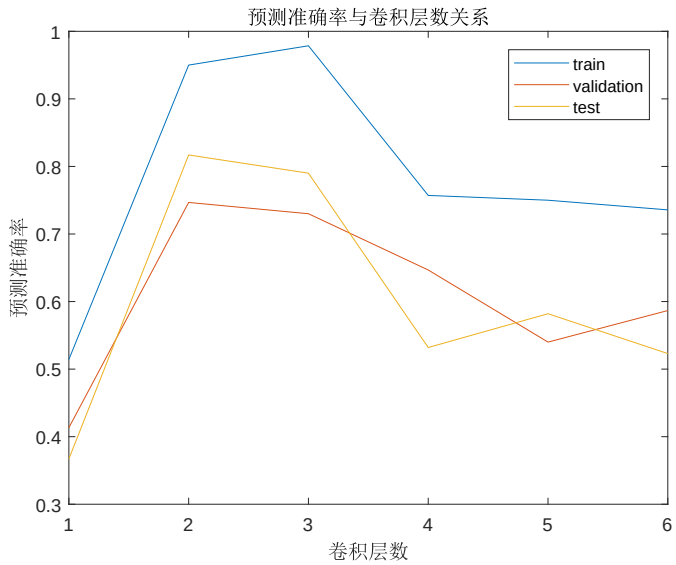

图 4-2: GCN 网络模型的准确率表现随层数变化的关系

上图是 GCN 原本的方法在 cora 数据集上进行轮次数 epoch=400，层数从 1 直至 6 的训练时，训练集、验证集和测试集的结果。从图标上可以看到，随着卷积层数量的增加，训练的效果呈现先增后减的趋势。其中，当 GCN 卷积层为 2 层和 3 层时效果最好，当卷积层数超过 4 层的时候卷积效果急剧下降，最后的准确率几乎相当于不进行训练的效果。从 GCN 卷积的原理也可以理解。

由于每次训练相当于通过等同于层数的邻居距离进行训练，当选择的邻居范围过大的时候，数据就会被过分的平滑，最终导致很差的训练效果。因此，考虑到预测准确性和复杂度的要求这里选择只使用 2 层的卷积进行比较。

在随机游走方法中，使用最简易的随机路径选择方法，默认循环次数 10，默认路径长度为 40，嵌入维数 16。

GCN 方法，训练过程中，cora 数据集挑选 140 个数据进行训练，其他两个数据集采用前 5% 的数据训练。验证过程 cora 数据集使用 500 个数据，其他的数据集采用前 10%。验证数据不参与训练

在实验噪声影响时，尝试无噪声，均匀噪声，标准正态噪声三种情况，噪声权重为 0.05。

4.3 训练结果

我们根据之前介绍的聚合函数选择，使用了最大值、最小值、均值和原先 GCN 论文中所使用的聚合函数，在 cora 数据集上训练得到的详细结果如下：

训练轮次数	1	50	100	150	200
MAX	14/10	52/47	83/68	87/67	90/74
MIN	7/6	30/31	37/37	45/44	99/78
MEAN	7/8	54/50	87/76	90/79	98/80
MEAN-sampling	15/16	41/43	66/59	79/71	87/74
Original-GCN	7/16	55/50	85/68	92/69	93/73

表 4-1: 隐藏层训练过程，训练集准确率/验证集准确率（按照百分比）

实际噪声	无	均匀	正态
无	82.20	不稳	82.60
均匀	76.00	不稳	79.10
正态	81.20	不稳	83.90

表 4-2: 考虑噪声隐藏层训练结果测试集准确率（按照百分比）

之所以标记不稳定，是因为在实验中发现对于均匀分布，他的初始值对于结果的影响很大。如果初始值与实际的权重比较相近，训练结果就会比较好。如果初始值与实际值差别较大（尤其是如果超过了实际值），则难以收敛到实际值，甚至可能只有 20 30 的准确率。相对来说正态分布的表现就比较好一些，但是对于结果也是只有很小的提升，并不足以完全替代初始方法。

聚合函数	损失	准确率
MAX	1.9040	71.50
MIN	1.6917	36.80
MEAN	0.7139	83.90
MEAN-sampling	0.9830	78.90
Original-GCN	0.8070	80.70
deepwalk	1.0020	72.00

表 4-3: 隐藏层训练结果的损失和测试集准确率（按照百分比）

GCN 方法训练的过程和结果如上表所示。可以看到，训练 200 轮次之后，几个不同的聚合函数结果在准确性上有一定的差距，但除了 MIN 的聚合方法，其他方法是相对不是特别大。在损失计算上略有差别。经过比较之后发现，MIN 函数的效果最差，MAX 效果相对可接受，MEAN 和 GCN 原本的方法效果比较好。

在 Blogcatalog 上的训练结果

轮次数	1	50	100	150	200
MAX	17/19	68/62	79/66	80/72	86/72
MEAN	18/19	70/63	80/67	84/65	87/67
MEAN-sampling	15/16	41/43	66/59	79/71	87/74
Original-GCN	18/19	70/60	78/	80/68	88/70

表 4-4: 隐藏层训练过程，训练集准确率/验证集准确率（按照百分比）

聚合函数	损失	准确率
MAX	0.9942	69.50
MEAN	0.9326	69.69
MEAN-sampling	1.0830	65.90
Original-GCN	0.8134	70.89
deepwalk	1.0402	63.03

表 4-5: 隐藏层训练结果的损失和测试集准确率（按照百分比）

在 Flickr 上的训练结果

epoch	1	50	100	150	200
MAX	8/9	66/40	75/43	78/46	86/52
MEAN	8/9	68/43	78/45	80/50	87/51
Original-GCN	9/10	69/46	80/48	80/52	89/52

表 4-6: 隐藏层训练过程，训练集准确率/验证集准确率（按照百分比）

聚合函数	损失	准确率
MAX	1.5842	52.61
MEAN	1.4626	53.20
Original-GCN	1.3763	55.61

表 4-7: 隐藏层训练结果的损失和测试集准确率（按照百分比）

可以看到，在 Flickr 数据集上几个方法的表现都不是很好。主要原始应该是 Flickr 数据集的类别标签过多，容易出现误判的现象。这个问题说明 GCN 方法的参数对于类别标签过多时难以进行处理。

可以理解的是，由于 MEAN 和 original-GCN 的方法都是倾向于对所有邻居节点进行均值，所以在一般情况（也就是节点连接边的分布比较均匀的情况下）表现比较好。MIN 和 MAX 应该作为在特殊节点情况下进行特殊的判别（比如两个相邻节点的度差距比较大，或者两个在聚合块边缘的相邻节点），这样效果会好一些。

由于 MEAN 方法在训练的效果中表现较好，因此这里对 MEAN 聚合函数采取邻居取样的尝试（MEAN-sampling 一栏）。在 cora 数据集中邻居节点超过 6 时随机选择 6 个邻居进行聚合，在 BlogCatalog 中选择 100。由于 Flickr 数据集本身表现不佳，不进行采样。可以看到，预测的精度略有下降，但是处于可接受范围之内。这样的尝试证实对每个节点的邻居采样聚合在现实操作中是可行的。

5 目前的局限和未来工作

这里，我们描述当前图卷积网络方法的一些局限性以及未来工作将如何克服这些问题：

邻居选择 在当前的邻居特征聚合中，主要还是采用全体一阶邻居的方式，消耗随着图节点的扩展会逐步增大。而 mini-bat 方法中，邻居的选择依然是以均匀随机为主，在某些特定节点上可能效果不佳。这可以通过更加巧妙的聚合函数进行优化。

有向图和边特征 目前 GCN 方法虽然允许有向图输入，但是处理时是采用转化为无向图的方法。并且 GCN 是针对节点特征聚合的方法，并不天生具备边属性预测的能力。在未来将边转化为中间节点再训练可能解决优化这一问题。

聚合函数 目前允许的聚合函数只有 MIN,MAX,MEAN,GCN 这几种方法，但是也有可能在某些途中，自身特征的重要性远远高于或低于节点邻居的特征。因此需要因为参数 λ 来调整自身特征在训练中的比重。

异质图 目前允许输入的图需要时同质的，也就是说节点与节点之间的 feature 矩阵与 label 矩阵，虽然数值不同，但是维数需要保持一致。这一点在异质网络上是难以实现的。对于随机游走方法，调整随机路径游走的方式可以模拟异质网络。但是对于 GCN 方法而言，目前还没有一个针对异质网络比较好的方法。

噪声干扰 目前对于噪声干扰的探讨还处于一个比较初级的阶段，对于如何处理噪声仍然没有一个比较准确有效的方法，所以以后还是需要进行更加深入的研究。

6 总结

我们介绍了图嵌入方法中的两种主流方法：图卷积网络方法和随机游走方法。我们详细解释了两种方法的来源、原理和具体算法。在实验部分，我们实现了半监督的图卷积网络图嵌入方法和随机游走方法，并针对图卷积网络嵌入方法做出了一定程度的优化。在公开数据集上的表现证实了这两种方法在图嵌入中提取节点特征和图拓扑结构的能力，以及较好的效率。最后我们提出了目前方法仍然存在的问题和改进可能。

参考文献

[1] WEI X, XU L, CAO B, et al. Cross View Link Prediction by Learning Noise-Resilient Representation Consensus[C/OL] // WWW '17 : Proceedings of the 26th International Conference on World Wide Web. Republic and Canton of Geneva, CHE : International World Wide Web Conferences Steering Committee, 2017 : 1611 – 1619.
https://doi.org/10.1145/3038912.3052575.

[2] CAO S, LU W, XU Q. Grarep: Learning graph representations with global structural information[C] // Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. 2015 : 891 – 900.

[3] HAMILTON W, YING R, LESKOVEC J. Representation Learning on Graphs: Methods and Applications[J], 2017.

[4] GOYAL P, FERRARA E. Graph embedding techniques, applications, and performance: A survey[J/OL]. Knowledge-Based Systems, 2018, 151 : 78 – 94.
https://www.sciencedirect.com/science/article/pii/S0950705118301540.

[5] G.E.HINTON. Learning distributed representations of concepts[C] // Proceedings of the eighth annual conference of the cognitive science society. 1986 : 1 – 12.

[6] PEROZZI B, AL-RFOU R, SKIENA S. DeepWalk: Online Learning of Social Representations[C/OL] // KDD '14 : Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA : ACM, 2014 : 701 – 710.
http://doi.acm.org/10.1145/2623330.2623732.

[7] CAI H, ZHENG V W, CHANG K C-C. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications[J/OL]. IEEE Transactions on Knowledge and Data Engineering, 2018, 30(9) : 1616 – 1637.
http://dx.doi.org/10.1109/TKDE.2018.2807452.

[8] HOFMANN T, BUHMANN J. Multidimensional scaling and data clustering[J]. Advances in neural information processing systems, 1995 : 459 – 466.

[9] YANG Y, NIE F, XIANG S, et al. Local and global regressive mapping for manifold learning with out-of-sample extrapolation[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 24. 2010.

[10] KIPF T N, WELLING M. Semi-Supervised Classification with Graph Convolutional Networks[C] // International Conference on Learning Representations (ICLR). 2017.

[11] BRUNA J, ZAREMBA W, SZLAM A, et al. Spectral networks and locally connected networks on graphs[J]. arXiv preprint arXiv:1312.6203, 2013.

[12] DEFFERRARD M, BRESSON X, VANDERGHEYNST P. Convolutional neural networks on graphs with fast localized spectral filtering[J]. arXiv preprint arXiv:1606.09375, 2016.

[13] ZHANG S, HUANG Z, ZHOU H, et al. SCE: Scalable Network Embedding from Sparsest Cut[C/OL] // KDD '20 : Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA : Association for Computing Machinery, 2020 : 257 – 265. https://doi.org/10.1145/3394486.3403068.

[14] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed Representations of Words and Phrases and their Compositionality[G/OL] // NIPS. [S.l.] : Curran Associates, Inc., 2013 : 3111 – 3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[15] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[16] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8) : 1735 – 1780.

[17] JIN Z, LIU R, LI Q, et al. Predicting user's multi-interests with network embedding in health-related topics[C] // 2016 International joint conference on neural networks (IJCNN). 2016 : 2568 – 2575.

[18] YANG Z, COHEN W, SALAKHUDINOV R. Revisiting semi-supervised learning with graph embeddings[C] // International conference on machine learning. 2016 : 40 – 48.

[19] ZHANG H, SHANG X, LUAN H, et al. Learning from collective intelligence: Feature learning using social images and tags[J]. ACM transactions on multimedia computing, communications, and applications (TOMM), 2016, 13(1) : 1 – 23.

[20] LI J, ZHU J, ZHANG B. Discriminative deep random walk for network classification[C] // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2016 : 1004 – 1013.

[21] YANG Z, TANG J, COHEN W. Multi-modal Bayesian embeddings for learning social knowledge graphs[J]. arXiv preprint arXiv:1508.00715, 2015.

[22] FANG H, WU F, ZHAO Z, et al. Community-based question answering via heterogeneous social network learning[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 30. 2016.

[23] GROVER A, LESKOVEC J. node2vec: Scalable Feature Learning for Networks[C] // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

[24] DONG Y, CHAWLA N V, SWAMI A. Metapath2vec: Scalable Representation Learning for Heterogeneous Networks[C/OL] // KDD '17 : Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA : Association for Computing Machinery, 2017 : 135 – 144.
https://doi.org/10.1145/3097983.3098036.

[25] JIN W, MA Y, LIU X, et al. Graph Structure Learning for Robust Graph Neural Networks[C] // 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2020. 2020 : 66 – 74.

[26] ANON. Exploring Expert Cognition for Attributed Network Embedding[C]. 2018 : 270 – 278.

致　　谢

感谢论文的指导老师顾庆教授和李正亮博士，在论文过程中的材料收集、实验设计、实验准备到现在的论文完成，都给予了极大的帮助和鼓励。在论文的准备过程中，两位老师严谨的学术态度和及时督促激励我不断追求进步，不断学习新的知识。

感谢我的舍友。尽管他们与我并不是同一个专业，但是在我进行毕业论文编写遇到困难时，不厌其烦倾听我的困难，并给出他们的意见，为我编写毕设提供了极大的动力。

感谢匡亚明学院，在平时给我们极大的自由发展空间，在毕设的各个环节及时提醒进度，帮助学生进行毕设的规划和指导。

感谢我的家人，在我大学四年给予我关怀和支持，在顺利完成学业的同时考虑未来的发展方向。