北京大学

硕士研究生学位论文

题目：面向知识图谱的图神经网络研究

姓　　名：陈语嫣

学　　号：1701214271

院　　系：前沿交叉学科研究院

专　　业：数据科学专业

研究方向：计算机科学与技术方向

导　　师：邹磊教授

二〇二〇年三月

版权声明

摘要

随着信息科学的飞速发展，电子化数据的数量呈指数型增长，迎来了大数据时代。从海量数据中提取用户所需要的信息，是互联网提供的重要功能之一，搜索引擎为用户提供了这一服务。然而，互联网上信息冗杂，且大部分为非结构化数据，这为快速而准确地查找信息带来了极大的挑战。

为了提高搜索的质量，谷歌在2012年提出了知识图谱，致力于为用户提供具有完整知识体系的搜索结果。知识图谱能提供详细的关于主题的结构化信息，用户能够直接解决查询的问题，而不必进入其他网站自己汇总信息。自提出以来，知识图谱的数量和规模都在不断增长，并在实际应用中发挥了重要作用。知识图谱是以图的形式储存的，因此这些应用背后的算法都涉及对图的分析和计算，图算法是知识图谱领域的重要基础。

另一方面，随着近十年来深度学习领域的进步与突破，人工智能与机器学习相关的话题越来越受到人们的关注。从而也就很自然地想到，是否能将深度学习应用到图分析任务中。这一研究方向称为图神经网络，并在近几年取得了极大的研究进展。

本文对面向知识图谱的神经网络进行深入研究。首先对现有的图神经网络模型进行分析，通过实验对几个著名模型进行评估，总结模型效果及特性，并对注意力机制和数据划分的影响进行深入分析。而后，对于知识图谱的实体分类问题，研究基于图神经网络的解决方案。指出了现有模型在这一问题上存在的不足，并提出一个全新的模型—门控关系图神经网络。门控关系图神经网络借用TransE的思想整合边和节点的信息，并引入门控机制控制节点的状态更新。通过在DB10K和FB15K数据集上的实验验证了这一模型的有效性。

关键词：图神经网络，知识图谱，实体分类，门控机制

Research Of Graph Neural Networks Towards Knowledge Graphs

Yuyan Chen (Data Science)
Directed by Prof. Lei Zou

ABSTRACT

With the rapid development of information technology, the amount of electronic data is growing exponentially, and here comes the big data era. Retrieving the information the users need is one of the important functions of internet, and this service is provided by search engine. However, the redundancy and miscellaneous of information, most of which is non-structral, brings tremendous challenge for finding information fast and accurately.

To improve the quality of search, Google proposed the concept of Knowledge Graph in 2012, which was used to provide search results with complete knowledge system for users. Knowledge graphs can provide detailed structral information about topics, and users can solve the problem directly, without going into other websites and gathering information by themselves. The amount and size of knowledge graphs keeps growing, and these knowledge graphs is of great importance in many real-world applications. Knowledge graphs are stored in the form of graph, as a result, the algorithms behind those applications involve graph analysis and graph computation. In other words, graph algorithm is the foundation of Knowledge Graph related problems.

On the other hand, with the advancement and breakthrough in deep learning field in recent decade, artificial intelligence and machine learning related topic are getting more and more attention and interest. It is normal to consider if deep learning can be applied to graph analysis tasks. This research direction is called Graph Neural Networks, which has made great progress in recent years.

This paper performs in-deep research into graph neural networks towards knowledge graphs. Firstly, we analyze current graph neural network models, and experiments are used to evaluate several representative models, whose efficacy and properties are then concluded. Afterwards, for the entity classification problem in knowledge graphs, we work on the solution based on graph neural networks. Current models are indicated to be insufficient. We proposed

a brand new model — Gated Relational Graph Neural Network. Our method borrows the idea of TransE to integrate the features of edges and nodes. Besides, gate mechanism is introduced to control the update of nodes' states. Experiments on DB10K and FB15K prove the effectiveness of our method.

KEYWORDS: Graph Neural Networks, Knowledge Graphs, Entity Classification, Gate Mechanism

目录

第一章　引言

随着信息科学的飞速发展，电子化数据的数量呈指数型增长，迎来了大数据时代。IBM提出了大数据的5V特点：大量（Volume）、高速（Velocity）、多样（Variety）、低价值密度（Value）和真实性（Veracity）。从这样的海量数据中提取用户所需要的信息，是互联网提供的重要功能之一，搜索引擎为用户提供了这一服务。然而，互联网上信息冗杂，包括结构化数据、半结构化数据及非结构化数据，且非结构化数据越来越成为主要部分，这为快速而准确地查找信息带来了极大的挑战。

为了提高搜索的质量，谷歌在2012年提出了知识图谱（Knowledge Graph），也称为知识库，致力于为用户提供具有完整知识体系的搜索结果。知识图谱能提供详细的关于主题的结构化信息，用户能够直接解决查询的问题，而不必进入其他网站自己汇总信息。

自提出以来，知识图谱的数量和规模都在不断增长，例如Yago[15]、DBpedia[20]、Freebase[3]和NELL[6]等，这些知识图谱可达几十上百GB，包含几百万个实体、数亿条边，并被用于解决各种各样的实际问题，例如语义解析[2]、命名实体消歧[9]、智能问答[38]等等。知识图谱是以图的形式储存的，因此这些应用背后的算法都涉及对图的分析和计算，图算法是知识图谱领域的重要基础。

另一方面，随着近十年来深度学习领域的进步与突破，人工智能与机器学习相关的话题越来越受到人们的关注。从而也就很自然地想到，是否能将深度学习应用到图分析任务中。这一研究方向称为图神经网络（Graph Neural Networks, GNN），并在近几年取得了极大的研究进展。

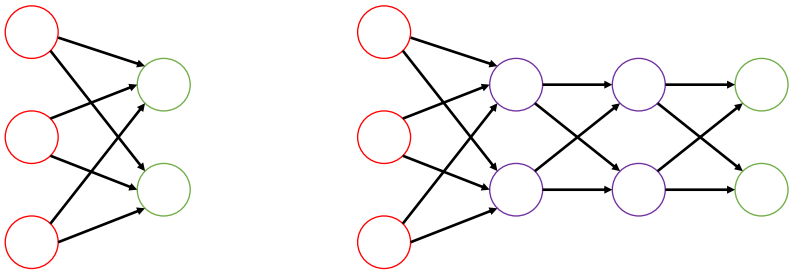
在本章中，首先介绍图神经网络相关研究，然后介绍知识图谱和知识图谱所涉及的分析任务及方法，并分析图神经网络方法在知识图谱问题上的研究现状。

1.1　图神经网络相关研究

1.1.1　神经网络概述

在计算机科学领域，所说的神经网络一般是指人工神经网络中的前馈神经网络，由一个输入层、一个输出层和若干个隐藏层组成。简单的神经网络结构如图1.1所示，图中的圆圈也称为神经元，网络中神经元的数量对应数据特征的维数。

感知器（Perceptron）仅由输入层和输出层组成。输入层传输数据、对输入进行计算，也称为计算层。感知器只适用于线性分类任务，例如逻辑回归模型。



(a) 感知机　　(b) 多层感知机

图 1.1　简单神经网络的模型结构

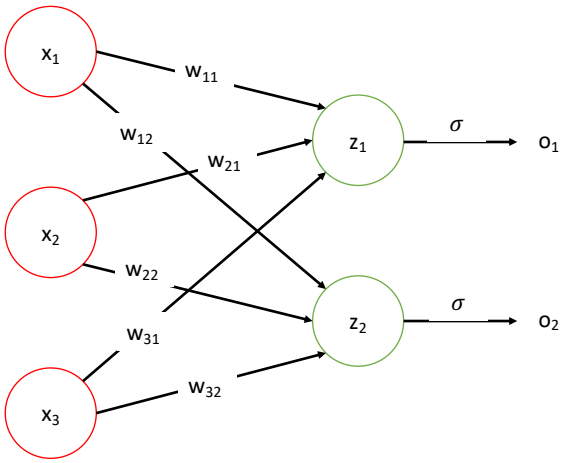


图 1.2　感知器

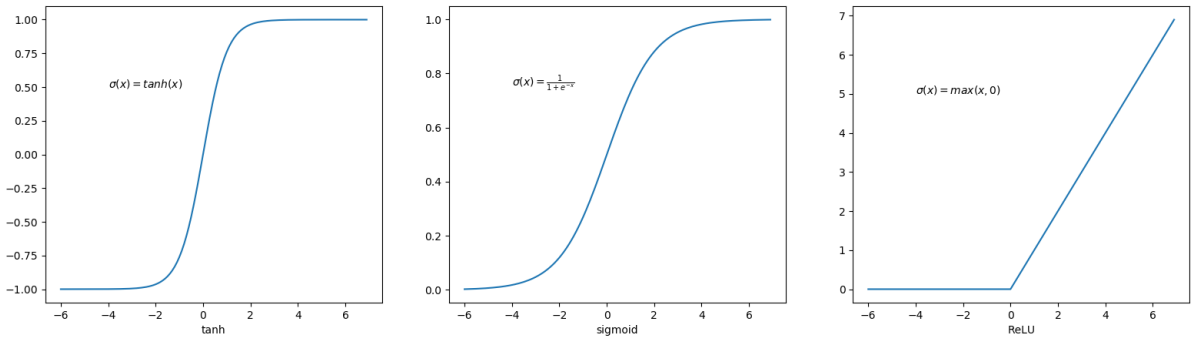


图 1.3　常见激活函数

在图1.2所示的感知器中，输入数据是a_1、a_2和a_3，输出数据是z_1和z_2，w是由网络训练的权重，变量之间存在如下关系：

$$\begin{aligned} z_1 &= \sigma(a_1 * w_{11} + a_2 * w_{21} + a_3 * w_{31}) \\ z_2 &= \sigma(a_1 * w_{12} + a_2 * w_{22} + a_3 * w_{32}) \end{aligned} \tag{1.1}$$

其中，$\sigma(\cdot)$为激活函数，也称为非线性函数，常用的选择包括tanh、sigmoid和ReLU等，函数图像如图1.3所示。直观上激活函数的作用是模拟生物神经元的激活，在数学层面上为模型引入非线性，从而提高模型的能力。

感知器是线性的，无法解决异或问题。多层感知器（Multi-Layer Perception）在感知器的基础上增加了若干个中间层，取得了非常好的非线性分类效果。中间层也称作隐藏层，其参数矩阵使数据的原始坐标由线性不可分转化为线性可分，这是实线多层感知器非线性分类的关键。理论证明，单个隐藏层的多层感知器可以逼近任意连续函数。

神经网络的训练目标是使损失函数最小化。损失函数根据任务的性质进行选择，例如，回归问题常采用最小二乘最为损失函数，分类问题常采用交叉熵作为损失函数。

神经网络需要通过多次迭代逼近模型的最优解，传统方法的运算效率低下，一度成为神经网络发展的重大阻碍。反向传播（back-propagation, BP）算法的提出解决了这一问题，它基于链式法则，将参数的梯度从输出层向前传递，而后对参数进行更新。

训练神经网络每一轮迭代可以分为三步：前向传播、反向传播、更新参数。前向传播根据当前的参数计算预测值，反向传播根据损失函数计算各参数梯度，而后用梯度下降法，根据给定的学习速率和刚才计算得到的参数梯度对参数进行更新。

1.1.2 卷积神经网络

卷积神经网络（Convolutional Neural Networks, CNN）是最常用的深度学习模型之一，具有捕获多尺度局部空间特征的能力。CNN的网络结构与多层感知器相似，只是在隐藏层引入了卷积、池化等操作。

卷积运算最早运用在信号处理领域，表示系统对输入信号的处理过程，实质就是对信号进行滤波，具有平滑信号的作用。从数学上来说，卷积是利用f和g两个函数生成第三个函数，记作$(f*g)(n)$，连续和离散的定义分别为:

$$(f*g)(n)=\int_{-\infty}^{\infty}f(\tau)g(n-\tau)d\tau \tag{1.2}$$

和

$$(f*g)(n)=\sum_{\tau}f(\tau)g(n-\tau) \tag{1.3}$$

在神经网络中，以图像处理为例，图像中有高频信号和低频信号，很多任务的目标是消除高频信号所代表的噪点的影响，学习到图片真正的内容，就可以利用卷积来实现。一张黑白图像可以由一个二维矩阵表示，矩阵上的元素代表图片对应像素上的颜色，如果是彩色图片，可以用另一个维度分别表示不同颜色的值再进行叠加，例如RGB表示法就可以用一个$h\times w\times 3$的三维矩阵表示一张高度为h、宽度为w的图片，在最后一个维度上，分别表示储存的是红色（Red, R）、绿色（Green, G）和蓝色（Blue, B）的亮度信息。假设有一张图以二维矩阵的形式储存$X\in R^{h\times w}$，卷积核为$G\in R^{k\times k}$，k一般是一个奇数，在这里我们取$k=3$，将中心元素的下标记作0，将G表示为

$$\begin{bmatrix}a_{-1,-1} & a_{-1,0} & a_{-1,1}\\ a_{0,-1} & a_{0,0} & a_{0,1}\\ a_{1,-1} & a_{1,0} & a_{1,1}\end{bmatrix}$$

用G对X进行卷积操作，得到的新的图像矩阵Y如下计算：

$$Y_{m,n}=\sum_{i=\lfloor k/2\rfloor}^{\lfloor k/2\rfloor}\sum_{j=\lfloor k/2\rfloor}^{\lfloor k/2\rfloor}X_{m-i,n-j}a_{i,j} \tag{1.4}$$

在本例中，有：

$$\begin{aligned}Y_{m,n}=X_{m-1,n-1}*a_{1,1}+X_{m-1,n}*a_{1,0}+X_{m-1,n+1}*a_{1,-1}+\\ X_{m,n-1}*a_{0,1}+X_{m,n}*a_{0,0}+X_{m,n+1}*a_{0,-1}+\\ X_{m+1,n-1}*a_{-1,1}+X_{m+1,n}*a_{-1,0}+X_{m+1,n+1}*a_{-1,-1}\end{aligned} \tag{1.5}$$

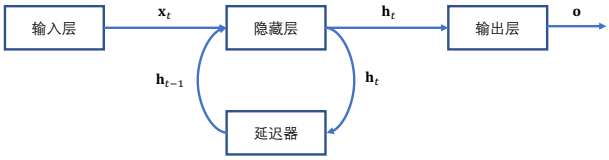


图 1.4　RNN网络结构

不同的卷积核可以提取图像中的不同特征。在神经网络中，卷积核可以由网络训练得到。除了卷积之外卷积神经网络还有一个重要操作是池化（pooling），使用一个固定大小的窗口在输入矩阵上滑动，将窗口中的值整合输出，即对指定区域内的数值计算平均值或最大值。池化的本质是采样，具有降低数据量、增加鲁棒性的作用。

卷积神经网络由具有局部连接和共享权重的特质。局部连接指的是，在卷积和池化操作中，对每一个数据点的处理只和周围一定范围内的数据有关，这一范围取决于卷积核或池化窗口的大小。共享权重则是指卷积核在整个输入图像上共享，即用同一个卷积核对图片不同区域的特征做卷积。这两个特质降低了网络的参数数量，从而降低了网络的训练难度。

1.1.3　循环神经网络

前馈神经网络中的输入是相互独立的，输入的顺序并不影响当前的输出结果。但在现实中，某些任务需要依赖过去的信息，例如在翻译任务中，当前单词可以依赖上下文拥有不同的含义，或是视频、音频相关的处理任务，都包含时间序列相关的信息。而前馈神经网络并不具备处理时序数据的能力。

循环神经网络（Recurrent Neural Network, RNN）是一种具有短期记忆功能的神经网络。网络结构带有环路神经元可以接收自身的信息，如图1.4所示，以如下公式更新隐藏层的特征：

$$\mathbf{h_t} = \mathbf{f}(\mathbf{h_{t-1}}, \mathbf{x_t}) \tag{1.6}$$

其中，下标t表示不同时刻，$\mathbf{x_t}$表示t时刻的输入信息$\mathbf{h_t}$表示t时刻的隐状态，$\mathbf{h_0} = \mathbf{0}$，即t时刻的隐状态不仅仅是由t时刻的输入决定的，还与t-1时刻的隐状态相关。

循环神经网络记忆的容量是有限的，在对之前的信息进行记忆并加以利用时，可能会包含大量无关信息。针对这一问题，一个有效的方法是引入注意力机制（attention mechanism）。注意力机制模拟人脑认知功能中的注意力，对某些信息给予更多的关注，对大量周边的无关信息进行过滤。注意力机制赋予了模型区别辨别的能力，例如，在翻译任务中，注意力机制可以解释输入句和输出句的对齐关系；在图像标注任务中，可以解释图片不同区域对输出文本的影响程度。

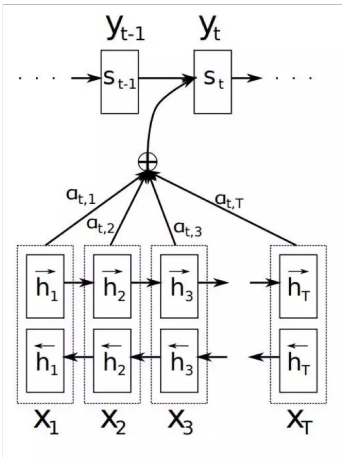


图 1.5　注意力机制

注意力机制基于循环神经网络的Seq2Seq模型。Seq2Seq是一个端到端的机器翻译模型，解决的主要问题是将变长的输入X映射到变长的输出Y。它包含一个编码器（encoder）和一个解码器（decoder）的，其中，编码器把输入X编码成因定长度的隐向量Z，解码器基于隐向量Z解码出目标输出Y。这一模型对输入中的每个单词赋予了相同的权重，缺乏区分度。

这一问题的解决方法就是注意力机制，如图1.5所示。图中，s_i是解码器中t时刻的隐状态，h_j是编码器中对应第j个输入单词的隐状态，α_{ij}表示编码器中第j个词与解码器中第i个词的权值，表示影响程度。并且有：

$$\begin{aligned} \alpha_{ij} &= \frac{exp(e_{ij})}{\sum_{k=1}^{T} exp(e_{ij})} \\ e_{ij} &= a(s_{i-1}, h_j) \end{aligned} \tag{1.7}$$

可以看到，α_{ij}是一个softmax函数输出，使相关性系数的和为1。e_{ij}是注意力打分函数，由第$i-1$个解码隐状态和第j个编码隐状态计算得到，计算方式并不是唯一的，不同的方法代表不同的注意力模型。

除了基本的注意力模式之外，还有许多变体。例如，键值对注意力利用“键”生成注意力分布，再利用”值“生成选择的输入信息；多头注意力利用多个查询平行地从输入中选择信息；结构化注意力针对输入信息本身具有层次结构的情况，对于不同粒度进行区别处理，等等。

循环神经网络存在梯度消失和梯度爆炸问题。为了解决这一问题，可以引入门控循环单元（gated recurrent unit, GRU），利用门控机制来控制输入和记忆对当前时刻隐状态的影响。GRU包含一个更新门（update gate）和一个重置门（reset gate），更新门控制记忆的传递，重置门控制当前输入与之前记忆的结合，如图1.6所示。

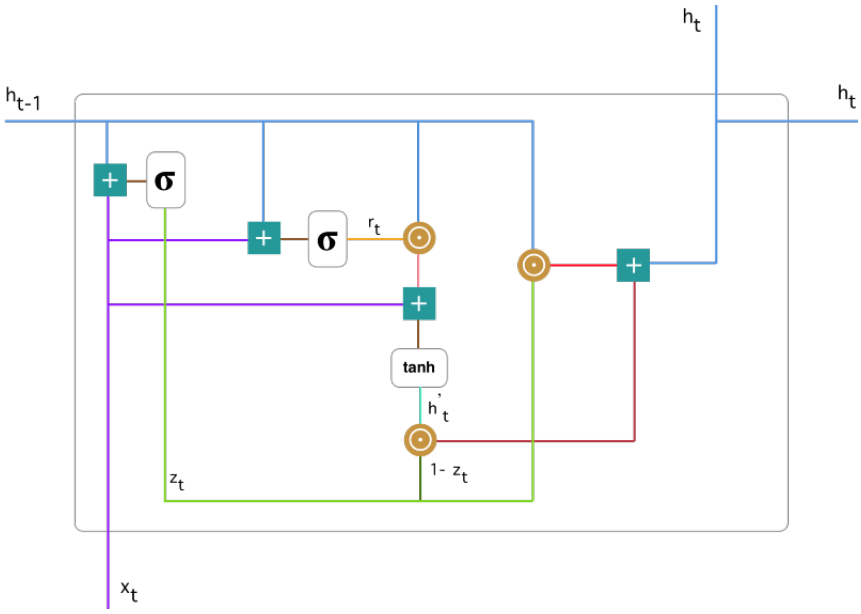


图 1.6　门控循环单元结构

在t时刻，首先计算更新门z_t：

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}) \tag{1.8}$$

其中x_t为t时刻的输入向量，h_{t-1}表示前一个时刻保存的信息，决定要将多少过去的信息用在当前的计算中。而后，计算重置门

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}) \tag{1.9}$$

重置门与更新门的计算表达式相同，再利用重置门计算新的记忆内容

$$h'_t = tanh(Wx_t + r_{tt-1}) \tag{1.10}$$

其中⊙表示哈达玛积。最后，网络用更新门的结果计算当前隐状态

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t \tag{1.11}$$

1.1.4　从卷积神经网络到图神经网络

卷积神经网络在诸如图像分类[19]，机器翻译[1]等任务中均取得了出色的效果。值得一提的是，上述任务所涉及的数据都是网格状的，例如图像、文本和音频。因此，当CNN最初被提出来解决计算机视觉中的问题并取得巨大成功时，研究者们便能够简单地将其迁移到自然语言处理上。但是，许多现实世界的数据格式都不像图像或序列。比如由节点和边组成的图，就位于不规则的非欧氏域中，传统的离散卷积在拓扑图上无法保持平移一致性，换句话说，图上各个顶点的度数（邻居节点数量）是不同的，

不能用一个统一尺寸的卷积核进行运算。因此很难将CNN推广到图当中。

CNN是图神经网络的灵感之一，另一个灵感来源于图嵌入（Graph Embedding）[39]，将图中的节点、边或子图表示为连续低维向量。图嵌入与图神经网络的主要区别在于：图神经网络是有监督或半监督学习，需要根据特定的图分析任务决定模型的损失函数；而图嵌入学习表示向量是无监督的过程，不针对特定的任务，在学习到特征向量之后，再用于上游任务的训练。但在某种角度来看，图神经网络也可以看作一种图表示学习的方法，在模型输出层的前一层的隐向量，即为学习到的节点表示。

经典的图嵌入算法包括DeepWalk[26]，node2vec[12]，LINE[29]等。DeepWalk[26]利用了词嵌入的思想，将图中的路径看作句子，节点看作组成句子的单词，而图中的路径则通过截断随机游走（truncated random walk）得到，即预先设定序列长度，从图上某一顶点开始，以一定概率向节点的邻居移动，并不断重复这个过程，直到路径达到设定的长度。而后，在得到的路径上应用skip-gram模型[24]对节点的向量表征进行更新。skip-gram模型在词嵌入中基于某一个单词预测上下文，因此在图嵌入中能够表征节点的邻居信息。node2vec[12]与deepwalk类似，不同点在于随机游走的策略。node2vec额外引入了两个参数p和q来控制随机游走，p控制重复访问前一节点的概率，q控制游走的方向，即权衡深度优先和广度优先，而DeepWalk的随机游走是深度优先的。LINE[29]重新定义了图中顶点的相似度，包括一阶相似度和二阶相似度。其中，一阶相似度是顶点之间的边的权重，而不相邻的顶点一阶相似度为0；二阶相似度表示顶点的邻居相似度，如果两个顶点没有共同的邻居节点，则其二阶相似度为0。

图神经网络最早由Gori等人[11]和Scarselli等人[27]提出，其主要思想是利用图的拓扑结构以及节点（和边）的特征。更具体地说，GNN利用当前节点的邻居节点、边、以及它自己在上一层中的隐状态来更新当前的隐状态，可以如下表示：

$$\mathbf{h}_v{}^{(t+1)}=f(\mathbf{h}_v{}^{(t)},\mathbf{h}_{ne[v]}{}^{(t)},\mathbf{h}_{co[v]}{}^{(t)}) \tag{1.12}$$

其中，$\mathbf{h}_v^{(t)}$是节点v在时刻t的隐状态向量，$\mathbf{h}_{ne[v]}^{(t)}$是v的邻居节点的表示，$\mathbf{h}_{co[v]}^{(t)}$是与v相连的边的表示。

近几年的研究提出了许多GNN的变体，并且取得了很好的效果。

粗略来说，这些模型主要在任务、传播过程或训练过程方面有所不同。图分析涉及的任务大致包括节点分类、链接预测、图分类和聚类等，同时也需要考虑到图的类型，比如节点是否同质，边是否有方向、权重或者标签。其中，在有标签节点和无向边组成的图上进行节点分类是最经典的问题之一。在传播过程中，主要区别来自于模型采用不同的聚合函数，例如卷积聚合函数[13,18]，注意力聚合函数[30,32]和门控聚合函数[22]。在训练方法上，变体主要包括邻居采样[13]，感受野控制[7]和提升方法

（boosting）[21]。

需要重点提到的一个图神经网络模型是图卷积神经网络（Graph Convolutional Network, GCN）[18]，这是一项突破性的工作，由谱图卷积的一阶近似简化得到，计算效率和可扩展性都大大提高，以其优秀的整合邻居节点信息的能力，在节点分类任务上取得了很好的效果，并为很多后续的图神经网络模型提供了灵感。本文的工作之一就是对包含GCN在内的几个具有代表性的图神经网络进行实验评估和分析。

1.2　知识图谱相关研究

1.2.1　知识图谱简介

知识图谱的本质是一个多关系图，实体即为图中的节点，关系是图中的边。在知识图谱中，知识以三元组的形式表示，一条三元组记为(s, p, o)，其中s称为主语或头实体，p称为谓词，也就是关系，o称为宾语，宾语可以是实体或者字面值常量，当宾语是另一个实体时，也称作尾实体，三元组有时也用(h, r, t)标记。例如，三元组$(China, Capital, Beijing)$可以表述“中国的首都是北京”这么一条知识；在这条三元组中，China是主语/头实体，Capital是谓词/关系，Beijing是宾语/尾实体。

随着知识图谱越来越广泛的应用，也出现了一些问题，根据范围可分为KG内的和KG外的。KG内的问题主要包括三元组分类、链接预测、实体分类和实体消歧，KG外的问题涉及关系抽取和智能问答等。在本文中，重点关注知识图谱的实体分类问题，这一问题也可以看作是特定关系的知识图谱补全。大多数情况下，知识图谱中的实体拥有多种类型，由谓词为isA的三元组表示。而实体的类别信息很可能是不完整的。利用已有信息学习实体类别是完善知识库的重要手段，具体方法包括图的表征学习和图神经网络等。

1.2.2　知识表示学习

知识图谱上的图嵌入也称为知识表示学习（Knowledge Representation Learning），大致可以分为两类：语义匹配模型和基于翻译的模型。

语义匹配模型利用基于相似性的评分函数，例如：RESCAL[25]将实体表示为一维向量$\mathbf{h}$和$\mathbf{t}$，将关系表示成二维矩阵$\mathbf{M_r}$，用函数$f_r(h,t) = \mathbf{h}^\top \mathbf{M_r} \mathbf{t}$计算三元组$(h, r, t)$的得分，表示这一三元组表示的知识的合理性，可以作为链接预测或三元组分类的依据；DistMult[36]在RESCAL的基础上，限定关系的表示$\mathbf{M_r}$为对角矩阵，简化了计算，但同时也弱化了处理复杂关系的能力；HolE将实体和关系都表示为一维向量，并利用循环相关运算定义打分函数$f_r(h,t) = \sum_{i=0}^{d-1} [\mathbf{r}]_i \sum_{k=0}^{d-1} [\mathbf{h}]_k \cdot [\mathbf{t}]_{(k+i) \bmod d}$，减少参数数量的同时维

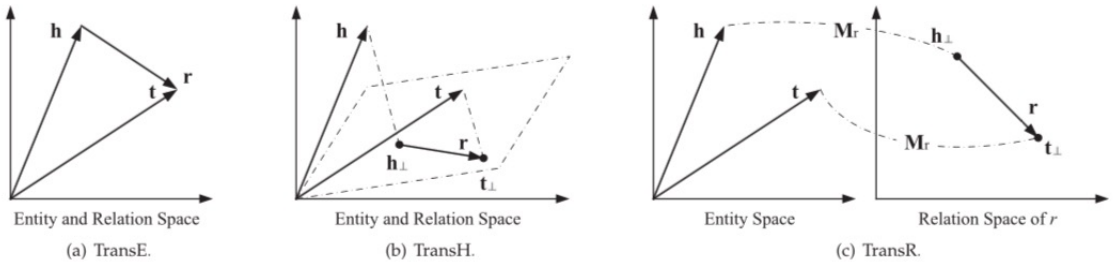


图 1.7　TransE，TransR和TransH图示

持了RESCAL的表达能力；ComplEx模型[31]将DistMult从实数空间扩展到复数空间，打分函数定义为$f_r(h,t) = \mathrm{Re}(\mathbf{h}^\top diag(\mathbf{r})\bar{\mathbf{t}})$。

基于翻译的模型将三元组中的关系看作是头实体到尾实体的翻译。最著名的模型是TransE[4]，它将实体和关系表示为同一空间中的向量，采用能量函数$\mathbf{s}+\mathbf{p}\approx\mathbf{o}$。也就是说，当三元组$(s,p,o)$存在时，尾实体的表示$\mathbf{o}$应该是离$\mathbf{s}+\mathbf{p}$最近的邻居。TransE的损失函数可以如下表示：

$$\mathcal{L} = \sum_{(s,p,o)\in S} \sum_{(s',p,o')\in S'_{(s,p,o)}} \max(0, \gamma + d(\mathbf{s}+\mathbf{p},\mathbf{o}) - d(\mathbf{s}'+\mathbf{p},\mathbf{o}')) \tag{1.13}$$

其中，S是训练三元组集，$S'_{(s,p,o)} = \{(s',p,o)|s'\in\mathcal{V}\}\cup\{(s,p,o')|o'\in\mathcal{V}\}$是负样本集合，通过替换正样本的头实体或尾实体得到。

虽然TransE模型非常简洁，它在知识图谱补全问题上取得了非常好的效果。TransE的扩展模型旨在更好地对1对N、N对1和N对N关系进行建模，例如TransH[33]将实体表示为向量，把关系表示为超平面上的向量，然后用实体在超平面上的投影计算能量函数；TransR[23]和TransH类似，TransH引入了超平面，而TransR引入了新的向量空间；TransD[16]将TransR中的投影矩阵分解为两个向量的积，从而减少了参数数量、简化了计算。

1.3　图神经网络与知识图谱

图神经网络方法也可以用于知识图谱相关任务。以GCN为例，虽然GCN只能处理只能处理包含带特征节点和带权无向边的简单图，显然，知识图谱并不在此范围内，但我们可以用一些预处理方案来解决这一问题。例如，将三元组(s,p,o)拆分为(s,p_1)和(o,p_2)[37]。具体来说，每个实体都被看作是图中的一个节点，而每个关系被拆为图中的两个节点，然后添加两条边，分别连接头实体和第一个关系节点、尾实体和第二个关系节点。从而，我们得到一个异质性的二部图，可以利用GCN进行计算。

也有一些研究工作提出了可以直接处理知识图谱的图神经网络，例如R-GCN[28]基于GCN框架对关系型数据建立模型。其基本思想是对不同的关系类型训练不同的权重

矩阵，卷积函数如下：

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\mathbf{W}_r^{(l)}\mathbf{h}_j^{(l)} + \mathbf{W}_0^{(l)}\mathbf{h}_i^{(l)}\right) \tag{1.14}$$

在一个有n个关系的知识图谱中，R-GCN需要在每一层神经网络训练n个权重矩阵。虽然R-GCN引入了两种策略—基础分解和块对角线分解—来正则化权重矩阵，以处理高度多关系型数据，但这一模型能够处理的关系数量仍然非常有限。另一方面，R-GCN也只是将不同类型的关系区分开来，并没有考虑关系本身所包含的信息。

1.4　本文主要贡献

本文首先回顾了几个经典的图神经网络模型，并通过实验比较了这些模型各自的优势和机制，包括注意力机制的作用、数据划分方式的影响等。然后，本文分析了用图卷积神经网络对知识图谱进行实体分类的预处理方法和缺陷，进而提出一种新的图神经网络模型，在知识图谱的实体分类问题上提升了准确率。对图神经网络的实验分析对就本文第二章内容，针对知识图谱的网络模型对应本文第三章内容。

在第二章中，我们对几个著名的图神经网络模型进行了充分的实验，包括图卷积神经网络、图注意力网络（Graph Attention Network, GAT）、基于注意力的图神经网络（Attention-based Graph Neural Network, AGNN）、共同训练和自我训练GCN（Co-Training and Self-Training GCN）、随机GCN（Stochastic GCN）。发现它们的性能在不同情境下有所不同。在大多数情况下，GAT的表现优于GCN，Co-Training系列方法可能在弱监督场景中用处较大，随机GCN减少了大量的训练时间，同时实现了与GCN相似的准确性。我们深入分析了注意力机制，包括GAT和AGNN之间的区别以及它们的工作原理。事实上，我们发现GAT的提升可能主要来自其多头机制（multi-head mechanism），而不是注意力本身。我们研究了测试准确率和数据划分之间的相关性，发现准确率与训练集中“一致节点”的比例有关。我们新增了一项实验来证明这一统计结果，这一发现对于实际应用中的数据标记策略提供了一定的指导意义。这部分实验和分析也为第三章中的模型提供了灵感。

在第三章中，我们首先分析GCN模型在处理知识图谱时存在的缺陷。而后提出一种利用GCN处理知识图谱的新方案，取得了更好的效果。更进一步，我们为知识图谱提出一种新的图神经网络模型—门控关系图神经网络（Gated Relational Graph Neural Network, GRGNN），这一模型能够处理知识图中的各种关系。大型知识图谱可能包含成千上万种关系，而之前并没有哪一种图神经网络模型可以处理如此丰富的边信息。 GRGNN尝试以一种简单而有效的方式整合实体和关系的特征。知识图谱是有向图，因此我们在GRGNN中考虑了边的方向。此外，我们采用门控机制来整合当前节点

及其邻居的特征。各种特征都可以用作模型的输入，这使GRGNN更加灵活。实验结果表明，GRGNN在知识图谱实体分类任务上优于其他方法。我们在FB15K和DB10K两个数据集上评估了GRGNN的效果。我们进一步分析网络深度（即模型层数）的影响。与GCN相似，随着模型深度的增加，GRGNN的性能也会下降。下降速率根据初始特征而有所不同，这意味着某些特征对模型深度更不敏感，而且使用此类特征的GRGNN不会有太多恶化，因为门机制可以避免节点嵌入的过度平滑。

第二章　图神经网络分析

2.1　符号

一个包含N个节点的简单图可以用一个$N \times N$的邻接矩阵$\mathbf{A}$来描述，$\mathbf{A}_{ij}$表示第i个点到第j个点的有向边的权重，$\mathbf{A}_{ij}$为0则表示从节点i到节点j没有边。在不带边权的无向图中，$\mathbf{A}_{ij}$是由0和1组成的对称矩阵，有$\mathbf{A}_{ij} = \mathbf{A}_{ji}$。对角矩阵$\mathbf{D} = \mathrm{diag}(d_1, d_2, ..., d_n)$是图的度数矩阵，只有对角线上的元素拥有非0值，其中$d_i = \sum_j \mathbf{A}_{ij}$。$\mathcal{N}(i)$表示节点$i$的邻居节点组成的集合，有$d_i = |\mathcal{N}(i)|$。图上的拉普拉斯矩阵一般定义为$\mathbf{L} = \mathbf{D} - \mathbf{A}$，有时也采用归一化的拉普拉斯矩阵$\mathbf{L} = \mathbf{I_N} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$。此外，在原图的基础上，为每个节点加上自环，所得图的邻接矩阵为$\tilde{\mathbf{A}}$，即$\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$，$\tilde{\mathbf{D}}_{i,i} = \sum_j \tilde{\mathbf{A}}_{ij}$。

在图神经网络中，$\mathbf{X}$表示节点的初始特征，即网络的输入；$\mathbf{h}_i^{(l)}$表示节点i在第l层的隐状态，$\mathbf{H}^{(l)} = [\mathbf{h}_1^{(l)}, \mathbf{h}_2^{(l)}, ..., \mathbf{h}_N^{(l)}]$是由节点隐状态组成的矩阵，可以直接用于计算，$\mathbf{W}$是网络训练的权重矩阵，$\sigma$是网络中的非线性激活函数。$\mathbf{O}$表示输出层矩阵，$\mathbf{o}_i$表示节点$i$对应的输出。

2.2　模型

在本节中，我们对几个具有代表性的图神经网络模型进行实验评估和分析，分别是GCN，GAT，AGNN，Stochastic GCN，Co-Training GCN和Self-Training GCN。在这些模型中，GAT和AGNN以不同的方式引入了注意力机制；Stochastic GCN提出了一种随机近似算法，极大程度上加快了GCN的训练速度；Co-Training GCN和Self-Training GCN设法在标记数据有限的情况下扩大训练集。

我们对这些方法进行评估的动机基于以下原因。首先，并非所有模型都在相同的数据集上进行实验。例如，只有GCN和Stochastic GCN使用了NELL数据集进行实验，其他模型在该数据集上的表现尚不清楚。其次，不同模型的实验设置有所不同，例如，Co-Training系列方法报告了不同标签率下的测试准确度。因此，虽然这些模型在原始论文中都报告了实验结果，但仍然很难确定它们的相对优劣和特性。第三，有必要对这些模型的一些关键组成部分做进一步分析，例如注意力机制和用于知识图谱的预处理方案。

我们将图上的节点分类问题当作半监督学习问题，也就是说，训练时已知的信息包括除节点类别之外的完整图结构，以及所有节点和边的特征，但是只有一小部分节点类型是已知的。

2.2.1 图卷积神经网络

定义在傅立叶域上的图卷积操作，用图拉普拉斯矩阵$\mathbf{L}$代替傅立叶变换中的拉普拉斯算子，然后求特征值，即对$\mathbf{L}$进行矩阵分解，$\mathbf{L}=\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathbf{T}}$，其中$\mathbf{\Lambda}$即为拉普拉斯矩阵的特征值组成的对角矩阵，$\mathbf{U}$为特征向量组成的矩阵，图上的卷积就定义为：

$$\mathbf{g}_\theta \star \mathbf{x} = \mathbf{U}\mathbf{g}_\theta(\mathbf{\Lambda})\mathbf{U}^{\mathbf{T}}\mathbf{x} \tag{2.1}$$

其中$\mathbf{g}_\theta$为卷积核，是$\mathbf{\Lambda}$的函数。这一方法需要对拉普拉斯矩阵做特征分解，计算非常困难，因而有人提出可以用切比雪夫多项式的K阶展开作为$\mathbf{g}_\theta$的近似[14]：

$$\mathbf{g}_{\theta'} \star \mathbf{x} \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\mathbf{L}})\mathbf{x} \tag{2.2}$$

其中$\tilde{\mathbf{L}} = \frac{2}{\lambda_{max}}\mathbf{L} - \mathbf{I_N}$。K阶展开意味着在当前节点的K步邻居范围内进行卷积。进一步近似$\lambda_{max} \approx 2$，上述表达式可简化为

$$\mathbf{g}_{\theta'} \star \mathbf{x} \approx \theta'_0\mathbf{x} + \theta'_1(\mathbf{L} - \mathbf{I_N})\mathbf{x} = \theta'_0\mathbf{x} + \theta'_1\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{x} \tag{2.3}$$

θ'_0和θ'_1是两个自由参数，取$\theta = \theta'_0 = -\theta'_1$，得到

$$\mathbf{g}_{\theta'} \star \mathbf{x} \approx \theta(\mathbf{I_N} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{x} \tag{2.4}$$

先前假定$\lambda_{max} \approx 2$，因此，现在$\mathbf{I_N} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$的特征值均在$[0,2]$之间，反复迭代可能出现梯度消失或梯度爆炸问题。为了避免这一问题，进一步将$\mathbf{I_N} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$替换为$\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$，得到图卷积神经网络的更新公式如下：

$$\mathbf{H}^{(t+1)} = \sigma(\mathbf{P}\mathbf{H}^{(t)}\mathbf{W}^{(t)}), \quad \text{with } \mathbf{P} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}} \tag{2.5}$$

其中，$\mathbf{H}^{(\mathbf{0})}$是节点的初始特征。

一个用于节点分类问题的两层图卷积神经网络形式如下：

$$\mathbf{O} = \text{softmax}(\mathbf{P}\text{ReLU}(\mathbf{P}\mathbf{X}\mathbf{W}^{(0)})\mathbf{W}^{(1)}) \tag{2.6}$$

第一层采用ReLU作为激活函数，第二层的输出维度与要分类的类别数量相同，softmax的结果作为每一类的概率。

2.2.2 图注意力网络

GAT由若干图注意力层（graph attention layer）组成。图注意力层首先将一个权重矩阵应用于节点特征，完成节点隐状态的维度变换，然后使用共享注意机制来计算每两个连接节点之间的注意力系数，后接用于非线性的LeakyReLU函数和用于归一化

的softmax函数。另外，模型采用多头注意力机制来保持学习过程稳定，即训练K个独立的注意力，然后对输出向量进行串联或平均。在GAT中，输出层为了保持向量维数与分类数相同，对多头注意力采取平均算法，非输出层将多头注意力得到的向量进行串联，作为下一层的输入。本文实验中使用的双层GAT可表示如下：

$$\alpha_{ij}^{(l)} = \text{softmax}_j(\text{LeakyReLU}([\mathbf{h}_i^{(l)}\mathbf{W}||\mathbf{h}_j^{(l)}\mathbf{W}]\mathbf{a}^{(l)})) \tag{2.7}$$

其中，$\mathbf{h}_i^{(l)} \in \mathbb{R}^{d^{(l)}}$是节点$i$在第$l$层的隐状态，$\mathbf{a^{(l)}} \in \mathbb{R}^{2d^{(l)}\times 1}$由网络训练得到。通过$\mathbf{a}$与节点的隐状态计算得到$\alpha_{ij}$，即为节点$i$到节点$j$的注意力系数，而后通过ELU函数激活，再经由softmax函数进行归一化：

$$\mathbf{h}_i^{(1)} = \Big\|_{k=1}^{K} \text{ELU}(\sum\nolimits_{j\in\tilde{\mathcal{N}}_i} \alpha_{ij}^k \mathbf{x}_j \mathbf{W}^{(0)}) \tag{2.8}$$

$$\mathbf{o}_i = \text{softmax}(\frac{1}{K}\sum_{k=1}^{K}\sum\nolimits_{j\in\tilde{\mathcal{N}}_i} \alpha_{ij}^k \mathbf{h}_j{}^{(1)}\mathbf{W}^{(1)}) \tag{2.9}$$

ELU函数如下：

$$ELU(x) = \begin{cases} x & \text{if } x \geqslant 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases} \tag{2.10}$$

2.2.3 基于注意力的图神经网络

AGNN基于向量的余弦相似度计算注意力系数，即特征越接近、相似程度越高的相邻节点，注意力系数越大，在传播矩阵中也拥有更大的比重。AGNN由一个词嵌入层（word-embedding layer）、若干带注意力的传播层、和一个输出层组成。词嵌入层即一个使用ReLU激活函数的全连接层，对节点的特征进行维度转化：

$$\mathbf{H}^{(1)} = \text{ReLU}(\mathbf{X}\mathbf{W}^{(0)}) \tag{2.11}$$

在带注意力的传播层中，每层只需要训练一个参数β，并通过节点隐状态的余弦相似度计算注意力系数α_{ij}，同样利用softmax函数进行归一化：

$$\alpha_{ij}^{(l)} = \text{softmax}_j(\beta^{(l)}\cos(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)})) \tag{2.12}$$

$$\mathbf{h}_i^{(l+1)} = \sum\nolimits_{j\in\tilde{\mathcal{N}}_i} \alpha_{ij}^{(l)} \mathbf{h}_j^{(l)} \tag{2.13}$$

输出层和词嵌入层类似，将节点的隐状态维度转换为分类数量：

$$\mathbf{O} = \text{softmax}(\mathbf{H}^{(l)}\mathbf{W}^{(1)}) \tag{2.14}$$

2.2.4 随机GCN

GCN模型采用随机梯度下降法进行训练，但由于模型中节点的感受野（receptive field）会随着模型深度指数型增长，换句话说，在两层GCN中，每个节点的输出与其两步邻居之内的节点有关，大大影响了随机梯度下降的训练效率。可以采用邻居抽样的方法对感受野进行控制，但随机采样可能导致分类准确率下降。Stochastic GCN的主要贡献在于它提出了GCN迭代的近似，节省了大量计算，并且理论上证明了可以收敛到GCN的局部最优点。

Stochastic GCN比较了四种模型：精确模型、邻居抽样（neighbor sampling, NS）模型、控制变量（control variate, CV）模型和控制dropout变量（control variate for dropout, CVD）模型。精确模型即为GCN模型；NS模型由Hamilton等人[13]提出，在模型的第l层随机为每个节点抽取$D^{(l)}$个邻居节点，然后如下卷积：

$$\mathbf{H}^{(l+1)} = \sigma((\hat{\mathbf{P}}^{(l)}\mathbf{H}^{(l)}\mathbf{W}^{(l)}) \tag{2.15}$$

其中$\hat{\mathbf{P}}^{(l)}$是带邻居抽样的传播矩阵，具体而言，

$$\hat{\mathbf{P}}_{ij}^{(l)} = \begin{cases} \frac{d_i}{D^{(l)}}\mathbf{P}_{ij} & \text{if } j \in \hat{\mathcal{N}}(i) \\ 0 & \text{otherwise} \end{cases} \tag{2.16}$$

其中，$\hat{\mathcal{N}}(i)$是$\tilde{\mathcal{N}}_i$的子集，在每次迭代中随机抽取得到。$D^{(l)}$是预先设定的抽样大小，即在网络的第l层，为节点i抽取$D^{(l)}$个邻居节点组成集合$\hat{\mathcal{N}}(i)$进行卷积运算，因此有$\left|\hat{\mathcal{N}}(i)\right| = D^{(l)}$。

但是NS模型存在两种随机性，一种由随机的邻居抽样导致，另一种由神经网络中的随机dropout导致。CV模型在NS的基础上加上一个控制言火日王，对邻居抽样的随机性进行估计以减小方差，其单层网络可以表示如下：

$$\mathbf{H}^{(l+1)} = \sigma((\hat{\mathbf{P}}^{(l)}(\mathbf{H}^{(l)} - \bar{\mathbf{H}}^{(l)}) + \mathbf{P}\bar{\mathbf{H}}^{(l)})\mathbf{W}^{(l)}) \tag{2.17}$$

其中，$\bar{\mathbf{H}}^{(l)}$是历史激活，需要在网络中额外维护。

为了处理dropout带来的随机性，文章提出了另一估计器CVD。令$\text{dropout}_p(\mathbf{X}) = \mathbf{M} \circ \mathbf{X}$为dropout操作，$E_M$为dropout的期望，$\mathbf{M}$服从伯努力分布，$\circ$为逐元素乘。CVD储存$\mu^{(l)} := E_M[\mathbf{H}^{(l)}]$以替代$\mathbf{H}^{(l)}$，并由此得到历史平均激活$\bar{\mu}^{(l)}$。CVD模型可以如下表示：

$$\mathbf{H}^{(l+1)} = \sigma(((\frac{1}{D^{(l)}}\mathbf{D})^{-\frac{1}{2}}\hat{\mathbf{P}}^{(l)}(\mathbf{H}^{(l)} - \mu^{(l)}) + \hat{\mathbf{P}}^{(l)}(\mu^{(l)} - \bar{\mu}^{(l)}) + \mathbf{P}\bar{\mu}^{(l)})\mathbf{W}^{(l)}) \tag{2.18}$$

理论分析证明，CV和CVD模型中，来自邻居采样的方差随着训练的进行会趋向于0，CVD模型可以很好地处理dropout导致的方差，从而能够收敛到GCN模型的局部最优

点，达到与GCN类似的效果。

2.2.5　协同训练GCN和自训练GCN

Co-Training GCN和Self-Training GCN实际上是用于弱监督环境中的预训练技术，即在有标签数据非常少的情况下提升GCN的训练效果。Co-Training GCN使用随机游走模型来扩大训练集，而Self-Training使用GCN模型获取更多的训练数据。

首先，要确定需要多少带标签节点。作者建议由公式$\bar{d}^{\tau} \times \eta \approx N$来计算$\eta$，$\bar{d}$是图的平均度数，$\tau$是用于训练分类的GCN模型的网络层数，由此计算得到的η应该是GCN传播标签以覆盖整个图所需的标签数量。然后利用随机游走模型或GCN模型，将每个类的最可信的节点添加到训练集中，使训练集扩大到预期的规模。

Algorithm 1: ParWalk模型与GCN联合训练

Input: 图拉普拉斯矩阵$\mathbf{L}$，$\mathbf{\Lambda}$，α，
每一类的带标签节点集合S_k，每一类额外需要的标记节点数量t

Output: 扩展后的训练集S'_k

```
P := (L+αΛ)^{-1}
for each class k do
    p := Σ_{j∈S_k} P_{:,j}
    sort p in descending order
    S'_k := S_k ∪ {top t nodes in p}
end
```

在Co-Training方法中，采用ParWalk[34]作为随机游走模型进行训练，详见算法1，首先计算归一化的吸收概率矩阵$\mathbf{P}$，$\mathbf{P}_ij$表示从节点i出发进行随机游走，被吸收到节点j的概率，同时也表示节点i和节点j属于同一类别的概率。然后对于每一类别，计算$\mathbf{p}$，$\mathbf{p}$是一个N维向量，$\mathbf{p}_i$表示节点i属于类别k的概率，然后选取$\mathbf{p}$最大的t个数对应的节点，将其加入到训练集中。

Algorithm 2: GCN自训练扩展训练集

Input: 节点特征$\mathbf{X}$，传播矩阵$\mathbf{P}$，每一类的带标签节点集合S_k，每一类额外需要的标记节点数量t

Output: 扩展后的训练集S'_k

```
Z := GCN(X)
for each class k do
    sort Z_{:,k} in descending order
    S'_k := S_k ∪ {top t nodes in p}
end
```

表 2.1 数据集统计信息

数据集	类型	节点数	边数	类别数	特征
Cora	引文网络	2,708	5,429	7	词袋
Citeseer	引文网络	3,327	4,732	6	词袋
Pubmed	引文网络	19,717	44,338	3	TF-IDF
NELL	知识图谱	65,755	266,144	105	独热编码
NELL-10K	知识图谱	11,460	49,780	105	独热编码 encoding

Co-Training GCN和Self-Training GCN还可以结合使用。将Co-Training中最可信的节点和Self-Training中的节点的并集添加到训练集中即为“并（Union）”方法，这一方法可以获得更多的训练数据。还可以添加来自Co-Training和Self-Training的共同节点，由此扩展得到的训练集应该会更准确，称为“交（Intersection）”方法。

2.3 方法

2.3.1 数据集

我们使用四个数据集来测试这些模型。Cora，Citeseer和Pubmed是GCN论文[18]中使用的引文网络，而NELL是Stochastic GCN[7]文章中使用的知识图谱，与GCN一文中使用的NELL在数据划分和节点特征上略有不同。在引文网络中，节点是带有词袋或TF-IDF特征的论文，边是论文之间的引用关系，在实验中被视为无向图。词袋特征统计论文中各个单词出现的频次并以此组成特征向量，TF-IDF计算文章中单词的区分能力，其中TF为词频，IDF为逆向文件频率，TF-IDF倾向于过滤掉常见词并保留重要的词语。NELL是由三元组组成的知识图谱。由于GCN无法处理边上的信息，因此需要对知识图谱进行预处理，每一关系被分为两个节点，即将$\Phi s, p, o\Psi$转换为$\Phi s, p_1\Psi$和$\Phi o, p_2\Psi$[37]。处理后，我们得到一个异构二部图。表2.1汇总了有关数据集的更多统计信息。由于GAT在每层中存储不同的传播矩阵，因此它需要与节点数的平方成比例的内存空间，而预处理过后的NELL数据集包含65755个节点，很难直接运行GAT算法。为了解决这个问题，我们抽取了一个由11460个节点组成的NELL子图，称为NELL-10K。实验表示，NELL与NELL-10K在GCN中的分类准确性相近，但NELL-10K的训练时间减少了90% 以上。我们使用NELL-10K作为知识图谱数据集进行实验。

2.3.2 实验设置

对于要测试的模型，除非另作说明，我们采用原始文献中的设置和超参数，详见表2.2。首先，我们根据给定的数据划分测试分类准确率，每个类包含20个有标签数据，验证集为500个节点，测试集有1000个节点。然后我们改变训练集大小并报告相

表 2.2　实验参数

		Cora	Citeseer	Pubmed	NELL-10K
GCN	模型层数	2	2	2	2
	dropout	0.5	0.5	0.5	0.1
	学习速率	0.01	0.01	0.01	0.005
	隐藏单元	16	16	16	64
	L2正则化	$5\cdot10^{-4}$	$5\cdot10^{-4}$	$5\cdot10^{-4}$	$1\cdot10^{-5}$
GAT	模型层数	2	2	2	3
	K（多头机制）	[8,1]	[8,1]	[8,8]	[8,8,8]
	dropout	0.6	0.6	0.6	
	学习速率	0.005	0.005	0.005	
	隐藏单元	8	8	8	32
	L2正则化	5·10-4	5·10-4	0.001	
AGNN	模型层数	2	4	4	3
	dropout	0.5	0.5	0.5	
	学习速率	0.01	0.01	0.01	
	隐藏单元	16	16	16	32
	L2正则化	5·10-4	5·10-4	5·10-4	
Co-Training	模型层数	2	2	2	2
	dropout	0.5	0.5	0.5	0.1
	学习速率	0.01	0.01	0.01	0.005
	隐藏单元	16	16	16	64
	L2正则化	5·10-4	5·10-4	5·10-4	5·10-4
	α	10-6	10-6	10-6	10-6
	Λ	I	I	I	I
Stochastic GCN	模型层数	2	2	2	2
	dropout	0.5	0.5	0.5	0.1
	学习速率	0.01	0.01	0.01	0.005
	隐藏单元	16	16	16	64
	L2正则化	$5\cdot10^{-4}$	$5\cdot10^{-4}$	$5\cdot10^{-4}$	$1\cdot10^{-5}$

应的结果。

2.4　结果

2.4.1　在给定数据划分下的准确率

我们在所有模型上对来自GCN文中[18]的固定数据划分进行测试，以此评估各个模型的效果。在原始文献中，几乎所有的模型都已经对Cora、Citeseer和Pubmed进行了参数优化，但NELL数据集只在GCN和Stochasitic GCN文中进行了测试。因此，需要额外确定NELL数据集在其他模型上的参数设置：Co-Training系列方法采用GCN的网络结构，对其使用与GCN相同的超参数；对于GAT和AGNN，我们对其超参数进行网格搜索，GAT最好的结果对应的参数为3层网络，每层由8个注意力头组成，各计算32个特

表 2.3　在给定数据划分下的分类准确率（%)

	Cora	Citeseer	Pubmed	NELL-10K
GCN	81.5 ± 0.71	70.8 ± 0.89	79.0 ± 0.28	65.8 ± 1.48
GAT	83.8 ± 0.64	72.2 ± 0.40	77.7 ± 0.70	68.8 ± 1.76
AGNN	81.9 ± 0.34	70.4 ± 0.41	78.8 ± 0.62	68.5 ± 1.31
Co-Training	81.2 ± 0.51	69.2 ± 0.61	77.7 ± 0.24	51.3 ± 0.27
Self-Training	80.9 ± 0.65	70.5 ± 0.81	78.6 ± 0.40	62.6 ± 1.54
Union	81.8 ± 0.61	69.2 ± 0.47	78.8 ± 0.39	60.2 ± 1.21
Intersection	78.1 ± 0.58	72.1 ± 0.46	77.7 ± 0.62	65.5 ± 1.47
Stochastic - exact	81.4 ± 0.78	70.8 ± 0.67	78.3 ± 0.40	64.8 ± 1.14
Stochastic - NS	75.7 ± 1.59	68.0 ± 1.46	74.6 ± 1.53	54.7 ± 1.42
Stochastic - CV	81.0 ± 0.85	69.9 ± 0.82	77.6 ± 0.79	64.6 ± 1.03
Stochastic - CVD	80.5 ± 0.40	70.7 ± 0.94	78.9 ± 0.66	64.8 ± 1.25

表 2.4　Stochastic模型的训练时间（秒）

	Cora	Citeseer	Pubmed	NELL-10K
Stochastic - exact	4.52	4.12	6.46	2.51
Stochastic - NS	1.69	1.39	1.32	1.75
Stochastic - CV	3.02	2.51	2.08	2.89
Stochastic - CVD	2.90	2.66	2.68	3.15

征；AGNN对应的参数为3层网络，隐藏层大小为32个单元。我们在每个数据集上为每个模型运行10次算法，并以百分比形式报告分类准确率平均值和标准差，参见表2.3。

GAT在Cora，Citeseer和NELL-10K上的表现比GCN更好，其差距大于标准差。Co-Training和Self-Training的Intersection方法在Citeseer数据集上取得了较为明显的效果。AGNN在NELL-10K上实现了与GAT类似的性能。而在其他情况下，没有显著优于GCN的模型。 Stochastic GCN的代码实现与其他模型有所不同：GCN、AGNN、GAT等模型均采用full-batch方式进行训练，而Stochastic GCN实现了mini-batch训练，从而大大加快了训练速度。Stochastic GCN实现了四种模型，其中exact模型等同于GCN，NS模型基于GraphSAGE[13]实现，CV和CVD模型是文章中提出的控制变量模型，额外加入了估计器以减小NS模型的方差。他们的训练时间总结在表**??**中。结合表2.3和表2.4可以看出，虽然NS方法的训练时间很短，但是模型准确率远远不如GCN模型，CV和CVD模型的准确率与GCN相似，同时大大缩短了训练时间。

2.4.2　不同训练集大小下的准确率

Li等人[21]认为，GCN的缺点之一在于它需要大量额外的带标签数据作为验证集，而在实际中可能无法获得这么多的标记数据，因此提出了Co-Training和Self-Training方法，可以在有标签数据很少的情况下提升GCN的效果。针对这一问题，我们用实验

表 2.5　Cora数据集上不同训练集大小的准确率

	2	4	10	20	50
GCN	54.9 ± 8.59	63.0 ± 5.16	74.4 ± 1.63	79.9 ± 1.42	83.6 ± 0.73
Co-Training	58.6 ± 6.61	65.1 ± 6.33	72.7 ± 3.35	79.1 ± 1.71	83.1 ± 1.09
Self-Training	61.1 ± 7.06	68.7 ± 3.33	76.1 ± 1.38	79.9 ± 1.19	82.7 ± 1.05
LP	57.0 ± 8.18	60.4 ± 5.25	62.7 ± 3.56	67.8 ± 1.90	72.6 ± 1.79
Union	60.9 ± 6.40	68.5 ± 3.91	76.2 ± 1.68	80.5 ± 1.23	83.3 ± 1.04
Intersection	60.1 ± 6.69	68.9 ± 2.97	73.7 ± 1.53	79.2 ± 1.37	82.7 ± 0.77
GAT	63.4 ± 8.32	72.1 ± 3.12	77.4 ± 1.88	81.8 ± 1.31	85.1 ± 1.05
AGNN	54.3 ± 6.93	65.1 ± 3.83	75.5 ± 2.01	79.6 ± 1.67	83.8 ± 0.97

表 2.6　Cora数据集上不同训练集大小的准确率

	2	4	10	20	50	100
GCN	44.1±5.97	53.7±4.28	65.2±1.93	67.7±1.84	71.6±1.24	72.8±0.85
Co-Training	40.4±7.23	51.0±6.72	59.8±3.86	64.0±2.52	68.6±1.59	70.3±0.82
Self-Training	50.4±5.39	59.3±4.44	66.0±1.11	67.8±1.22	70.7±1.43	71.4±0.84
LP	36.7±6.06	38.8±4.62	43.0±2.16	44.5±2.79	49.5±2.49	54.1±1.78
Union	49.0±5.05	55.6±5.04	63.4±2.15	65.7±1.95	69.4±1.53	70.6±0.74
Intersection	49.0±7.38	60.5±7.69	68.2±1.17	69.8±1.43	71.4±1.48	72.0±6.53
GAT	49.7±6.57	59.1±4.11	67.2±1.98	69.3±1.63	72.4±1.17	74.1±1.17
AGNN	50.0±6.37	58.3±3.64	66.6±1.34	69.5±1.28	73.8±1.45	74.3±0.93

验证在不同大小的训练集下GCN、Co-Training、Self-Training、GAT和AGNN的效果。在Cora上，训练集是每个类2/4/10/20/50个节点，在Citeseer和Pubmed上每个类2/4/10/20/50/100个节点。仅在GAT上使用验证集防止过拟合，其他模型不使用验证集。此外，我们还测试了使用ParWalks方法进行标签传播（label propagation, LP）进行分类的准确率。我们对数据进行10次随机划分，并在所有模型中共用这些划分。模型的其他参数保持不变。结果参见表2.5、表2.6和表2.7。

表 2.7　Pubmed数据集上不同训练集大小的准确率

	2	4	10	20	50	100
GCN	63.3±7.78	67.9±4.99	75.4±1.79	78.8±1.87	82.2±1.83	84.3±0.87
Co-Training	62.9±8.87	67.4±6.18	75.2±5.79	77.6±3.07	81.7±1.27	83.8±1.12
Self-Training	66.7±9.14	69.4±5.10	75.7±2.11	77.6±2.49	81.5±1.33	82.7±0.78
LP	62.5±10.37	65.3±7.70	68.0±4.32	68.5±2.20	69.3±2.65	70.5±1.14
Union	65.8±8.80	68.9±5.29	77.1±2.09	78.9±2.04	82.4±0.95	84.0±0.90
Intersection	66.0±8.47	69.1±4.20	74.4±2.79	77.0±2.45	70.5±1.63	83.0±0.70
GAT	66.7±7.51	69.2±4.01	75.0±2.05	78.0±1.99	81.4±1.40	83.5±0.73
AGNN	63.4±8.85	67.9±4.66	74.4±2.66	77.4±1.32	80.1±1.77	81.8±1.49

在所有三个数据集上，仅当有标签节点很少时，Co-Training系列方法的效果优于GCN。而在Co-Training系列方法中，只有在Citeseer数据集上，每类标签数为2或者4时，Co-Trainnig方法优于GCN。大部分情况下，Co-Training的效果不如Self-Training。而当每一类别有超过20个有标签数据时，Co-Training和Self-Training方法相比GCN没有体现出优势。Co-Training方法的效果在很大程度上依赖于图拓扑结构的质量，正如表中所示，当LP方法表现不佳时，Co-Training的效果比GCN更差。例如在Citeseer中，图由几个连同分量组成，利用图结构进行标签传播可能受到阻碍，此时用Co-Training方法扩大的训练集可能含有较多错误标签，继而影响后续GCN的分类效果。

值得一提的是，我们发现在所有三个数据集上，有标签数据很少时，GAT的效果显著优于GCN，AGNN在Citeseer上的分类准确率也好于GCN。我们将在下一节中对GAT和AGNN这两个引入注意力机制的模型做进一步分析。

2.5 分析

2.5.1 注意力机制

GAT和AGNN这两个模型以不同的方式引入了注意力机制。GAT采用的是自注意力（self-attention），串联两个相邻节点的隐状态并点乘一个向量来获得系数，而AGNN计算相邻节点隐状态之间的余弦相似性，其隐藏的假设是相似的节点之间应该传递更多信息。

我们通过累加类之间的传播系数来计算类间相关性得分，即在类之间传递的信息总量，然后将其与$\alpha_{ij} = \frac{1}{|\widetilde{N}(i)|}$的常量注意力机制进行比较。

$$Message(c_2 \to c_1) = \frac{1}{|S_{c_1,c_2}|} \sum_{(i,j)\in S_{c_1,c_2}} \alpha_{ij} \tag{2.19}$$

$$Relevance_{Model}(c_2 \to c_1) = \frac{Message_{Model}(c_2 \to c_1) - Message_{constant}(c_2 \to c_1)}{Message_{constant}(c_2 \to c_1)} \tag{2.20}$$

其中$S_{c_1,c_2} = \{(i,j)|(i,j) \in \mathcal{E} \wedge Y_i = c_1 \wedge Y_j = c2\}$。

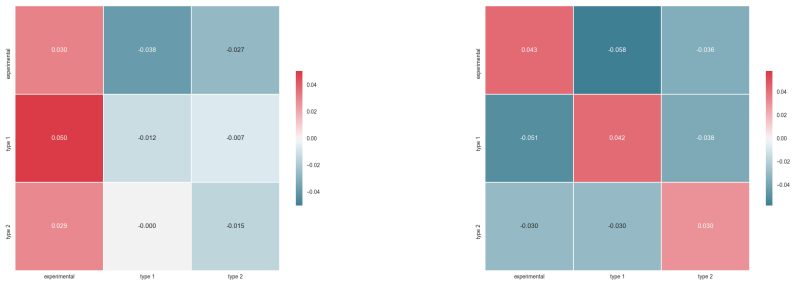
图2.1、2.2和2.3分别展现了Cora、Citeseer和Pubmed的注意力矩阵。从图中可以看出，GAT和AGNN学习到的注意力非常不同。在三个数据集上，AGNN的矩阵对角线上的值均为正数，表明AGNN对同一类别节点之间传递的信息施加更多权重，而GAT没有显示出这种趋势。实际上，GAT计算的注意力系数与常数注意力差距非常小。如果不使用多头机制，GAT的测试结果并没有超过GCN。另一方面，由于GAT和GCN的传播矩阵区别不大，我们推测GAT的提升可能主要来自其多头机制。

(a) GAT　　(b) AGNN

图 2.1　Cora数据集上GAT和AGNN模型的类间相关系数

(a) GAT　　(b) AGNN

图 2.2　Citeseer数据集上GAT和AGNN模型的类间相关系数



(a) GAT　　(b) AGNN

图 2.3　Pubmed数据集上GAT和AGNN模型的类间相关系数

表 2.8　Citeseer数据集上不同层数的GCN、GAT和AGNN模型表现

	GCN	GAT	AGNN
2层	70.8 ± 0.89	72.2 ± 0.48	70.6 ± 0.34
3层	64.6 ± 3.56	71.4 ± 0.97	70.1 ± 0.44
4层	50.0 ± 11.06	68.0 ± 1.99	70.4 ± 0.41

Li等人[21]证明了图卷积是拉普拉斯平滑的一种特殊形式。因此，随着图卷积层的数量增加，同一连通分量内的节点的特征趋向于收敛到同一值，这使得不同类型的节点难以区分。GCN在两个卷积层时实现了最佳性能，如果采用更深层模型，会面临特征收敛以及过拟合的问题。另一方面，注意力机制使得更深的模型成为可能。正如表2.2中所示，某些数据集上，GAT和AGNN在三层或四层网络时取得了最好的效果，突破了传统GCN模型两层网络的限制。注意力机制对图上的边赋予不同的权重，而其权重并不是固定不变的，在网络中是动态的、适应性的，从而可以在图中传递更多的有效信息，并且减少噪声的传播干扰。AGNN仅在每个传播层上训练一个参数，在一定程度上也避免了由参数过多引起的过拟合问题。我们测试了在Citeseer数据集上模型深度对GCN、GAT和AGNN的影响，结果如表2.8所示，GCN的性能随着模型加深而迅速下降，GAT显示缓慢下降且AGNN基本保持稳定。

2.5.2　数据划分

我们注意到，在给定的数据划分下，分类效果要好于随机数据划分下的结果。这一现象提示我们，是否会有一些数据划分特征会对训练效果产生影响。我们收集一些统计数据并使用线性回归以查看哪些变量与模型准确性相关。

对于节点i，在其邻居$\mathcal{N}(i)$中，如果有超过一半的节点与i所属类别相同，我们称i为“一致节点”。我们考虑训练集的1步和2步邻居数量以及其中一致节点的比率对训练效果的影响，使用随机数据划分执行100次GCN算法并使用回归分析来查看它们与测试准确率的相关性，结果见表**??**。可以看出，训练集中一致节点的比例起着重要作用。其背后的基本假设是，在卷积之后，不一致节点的特征与其真实类的相关性较小。换句话说，训练集中的一致节点越多，训练期间传播的噪声越少，学习的参数模型越好，并且将导致测试集上的更高准确率。

接下来，我们进一步验证训练集中一致节点的比便对模型分类效果的影响。删除数据中连接属于不同类型的两个节点的所有边，可以得到一个新的邻接矩阵，并在训练过程中使用此邻接矩阵。这样一来，训练集中的节点全都变为了一致节点。然后，在测试时，我们使用原始的未删除边的邻接矩阵，其他参数均与GCN模型相同，将这种方法称为M1，并比较这一方法与GCN的效果，结果见表2.10。可以看出M1方法显

表 2.9　模型准确率与数据划分变量的回归分析

	1-hop size		2-hop size		rate_0		rate_1		rate_2	
	t	*p*	*t*	*p*	*t*	*p*	*t*	*p*	*t*	*p*
Cora	.67	.503	.57	.567	**2.13**	**.036**	.061	.546	.41	.681
Citeseer	-1.18	.239	.15	.881	**2.82**	**.006**	1.70	.092	.37	.712
Pubmed	.50	.619	-.49	.627	**4.02**	**.000**	.91	.367	-.37	.714

[1] 1-hop size和2-hop size分别表示训练节点的1步和2步邻居包含的节点数量，
rate_0ϒ−Γfi$rate_1 rate_2 h1e2eE$−ΓΘt$e'R$tfip$h$:tG>W'Θt<′fip<ffffiffis''ΘN>W'4sΦæ8Ψ:0.05Ψp<h:$∗$KwΨ>

表 2.10　训练时采用无噪声邻接矩阵的结果

	Cora	Citeseer	Pubmed
GCN	79.9 ± 1.42	67.7 ± 1.84	78.8 ± 1.87
M1	**81.5 ± 1.29**	**70.3 ± 1.67**	**79.7 ± 2.63**

著提升了准确率。这一结果反过来支持先前的假设，即降低噪声有助于训练模型。这可能对实际应用中的数据标注具有一定的指导意义。虽然我们在训练时删除了所有连接不同类型节点的边，但实际上，只有带标签节点的两步邻居之内的边会对模型效果产生影响。另一方面，如果我们能够减少噪音或以某种方式让模型学习这些信息，它也可能对模型效果有所帮助。

第三章　门控关系图神经网络

给定一个知识图谱$\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$，其中$\mathcal{V}$是节点（实体）的集合，$\mathcal{R}$是关系集合，$\mathcal{E} \in \mathcal{V} \times \mathcal{R} \times \mathcal{V}$是边（三元组）集合。每个节点可以有零个、一个或者多个标签（即类别）。对于节点i，用$\mathcal{N}e_i$表示i的邻居节点，$\mathcal{N}r_i$表示i的邻居边，也就是说，$\forall e \in \mathcal{N}e_i$, $\forall r \in \mathcal{N}r_i$, $(i, e, r) \in \mathcal{E}$或者$(i, e, r) \in \mathcal{E}$。$Y_i$表示$i$的标签集。$e$表示实体，$r$表示关系，对应的粗体字母表示特征向量，将特征放在一起，即得到实体的特征矩阵$\mathbf{X_e} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$和关系的特征矩阵$\mathbf{X_r} = [\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_m]$，通常作为网络的输入。

3.1　GCN处理知识图谱的缺陷

正如前两章提到的，GCN只能处理简单无向图，无法处理边上的标签。因此，在用GCN对知识图谱进行节点分类时，需要先进行预处理。一种广泛使用的预处理方案是将(s, p, o)分解为(s, p_1)和(o, p_2)，如图3.1所示。

这一过程的一个重大缺点是它删除了头实体和尾实体之间的相连关系。在图3.1的例子中，原先有(*China*, capital, *Beijing*)和(*U.S.A*, capital, *Washington*)两条三元组，*China*和*U.S.A*都属于<country>类别，*Beijing*和*Washington*都属于<city>。在预处理之后，"capital"被拆分为$capital_1$和$capital_2$两个节点，其中$capital_1$与*China*和*U.S.A*分别相连，$capital_2$与*Beijing*和*Washin*连。通过2层GCN，*China*和*U.S.A*，*Beijing*和*Washington*可以相互卷积。其背后的假设是同一种关系的同一端实体倾向于具有相同的类型。因此，对<country>类型的节点进行分类是有帮助的，因为它们始终出现在关系“capital”的左端，并在“capital“右端出现<city>类型的节点。但是，有时这种方案会混淆不同类型的实体。考虑三元组(*Victor Hugo*, works_written, *Notre-Dame de Paris*)和(*Andrew Lloyd Webber*, works_written, *The Phantom of the Opera*)，如果我们知道*Notre-Dame de Paris*是一部小说而*The Phantom of the Opera*是一部戏剧，那么自然能将*Victor Hugo*归类为小说家、将*Andrew Lloyd Web-*

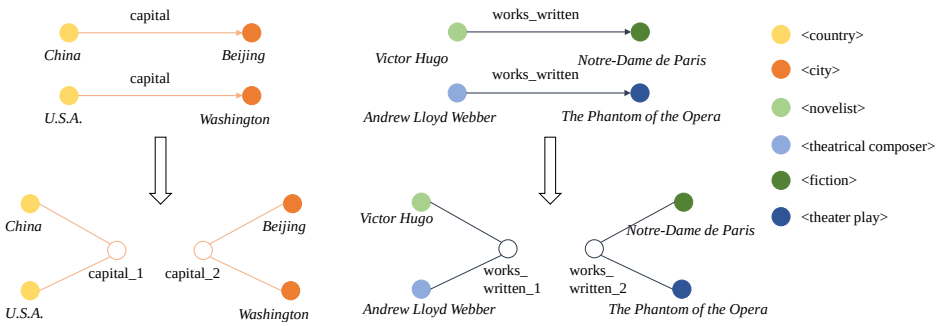


图 3.1　知识图谱预处理方案图示

表 3.1　两种预处理方案的数据统计结果和分类效果（F1分数）

	节点数	边数	特征（维数）	传播矩阵	F1
原始方法	17,641	308,006	one-hot (17,641)	$\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$	73.1
我们的新方法	14,951	260,184	relations (2,690)	$\rho(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}) + (1-\rho)\mathbf{I}_n$	**74.1**

*ber*归类为戏剧作曲家。但是在分割后的图中，仅凭借图中的信息，*Victor Hugo*和*Andrew Lloyd Webber*将变得无法区分。

我们提出一种在知识图谱上运行GCN的新方法：将与实体相连的关系作为特征，处理方式与词袋类似，s和o的关系被区分对待，在知识图谱$\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$中，为每一实体赋予2×$|\mathcal{R}|$维的特征$\mathbf{r}$，假定当前节点为$i$，$\mathbf{r}_{2*k}$为三元组$(i, r_k, \cdot)$出现的次数，$\mathbf{r}_{2*k+1}$为三元组$(, r_k, i)$出现的次数。然后忽略知识图谱上边的标签，直接运行GCN。为了更好地保留当前节点的特征，我们引入一个额外参数ρ，并将传播矩阵$\mathbf{P} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$替换为$\mathbf{P}=\rho(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}) + (1-\rho)\mathbf{I}_n$，其中$\rho$由网络训练。

我们使用FB15K[4]对这种新的预处理方案进行多标签分类的实验评估。FB15K是一个从大型知识图谱FreeBase[3]中抽取的数据集。采用Xie等人[35]提出的方法，我们从Freebase中提取FB15K中实体的所有类型信息，然后出现的类别按频率降序排序。除了几乎所有实体都有的类别“common/topic”之外，前50种类型被用于实验。

FB15K包含14951个实体和1345种关系。因此，我们的方法将为实体分配2690维特征。在原来的预处理方案中，拆分图后的矩阵是0/1矩阵，我们对其稍作改动，即，如果实体节点和关系节点之间存在多条边，我们在邻接矩阵中对应的边权设置为出现的次数。我们选择了144个标记节点为训练集，确保其中每个类的正例出现不少于10次。表3.1中汇总了一些统计数据和测试结果。

与原先的预处理方案相比，我们的新方法获得了更好的效果。事实上，GCN处理带边信息图的能力有限。最近有一些针对关系数据的节点分类的工作。例如，R-GCN[28]对每个关系上的传播采用不同的权重矩阵，RGAT[5]在R-GCN的基础上引入注意力机制。然而，在具有成千上万种关系的知识图谱中，很难训练如此大量的参数。知识图谱的GNN是一个仍需研究的领域。

3.2　模型结构

在本节中，我们将介绍门控关系图神经网络（GRGNN）的架构，然后通过与其他相关工作的比较来分析其优势。

总体来说，GRGNN整合节点的邻居点和邻居边的信息，然后用整合的信息来更新当前节点的隐状态。

述GRGNN的单层结构如图3.2所示。假设我们正在更新节点i的隐状态，三元组集

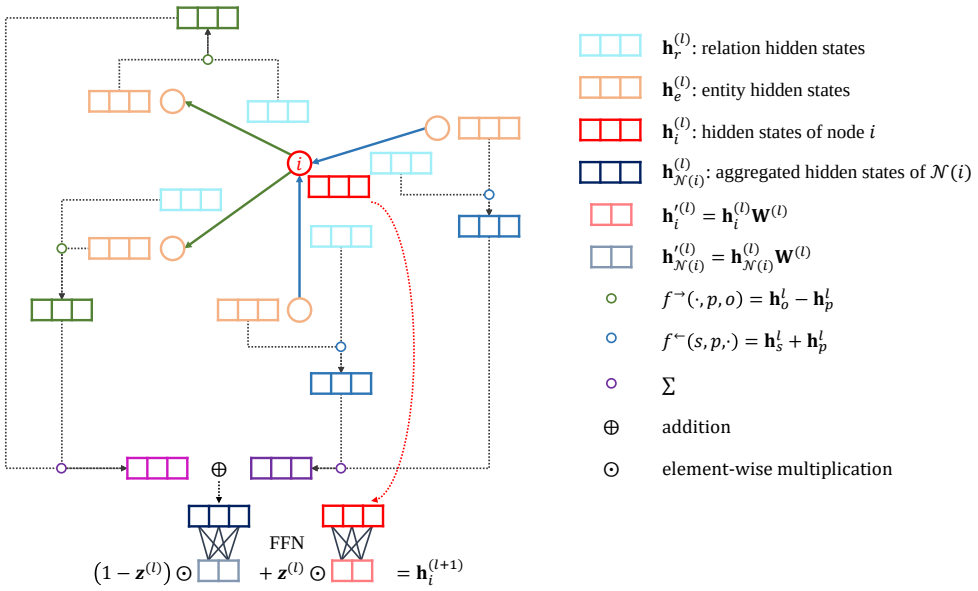


图 3.2　单层GRGNN网络示例

合T_1包含i作为头实体的三元组，T_2包含i作为尾实体的三元组。用公式表示，即为$T_1 = \{(s,p,o)|(s,p,o) \in \mathcal{E} \wedge s = i\}$，$T_2 = \{(s,p,o)|(s,p,o) \in \mathcal{E} \wedge o = i\}$。我们想要定义函数$f^{\rightarrow}(\cdot,p,o)$和$f^{\leftarrow}(s,p,\cdot)$来整合实体的状态和关系的状态。

对于三元组(s,p,o)，根据TransE的思想，尾实体的表示向量应该接近头实体加上关系的向量，因而我们可以将$\mathbf{s}+\mathbf{p}$整合到实体o的隐状态中，将$\mathbf{o}-\mathbf{p}$整合到实体s的隐状态中。于是我们的模型中，$f^{\rightarrow}(\cdot,p,o)$定义为$\mathbf{h}_o^{(l)} + \mathbf{h}_p^{(l)}$，$f^{\leftarrow}(s,p,\cdot)$定义为$\mathbf{h}_s^{(l)} - \mathbf{h}_p^{(l)}$。然后，我们将节点$i$的邻居状态进行加和，从而得到$i$的邻居聚合状态，如下：

$$\mathbf{h}_{\mathcal{N}(i)} = \frac{1}{|T_1|}\sum_{(\cdot,p,o)\in T_1}(\mathbf{h}_o - \mathbf{h}_p) + \frac{1}{|T_2|}\sum_{(s,p,\cdot)\in T_2}(\mathbf{h}_s + \mathbf{h}_t) \tag{3.1}$$

使用GCN进行节点分类时，通常会在原图的基础上为每一节点加上自环，以在卷积过程中保留当前节点的特征，我们在GRGNN中采用相似的方法整合节点本身和邻居的状态。GCN认为当前节点和邻居节点具有同等重要性，因此自环和边的权重均为1，这样的假设有待商榷。在此我们引进一个额外的参数ρ来权衡当前节点自身和其邻居的特征。

$$\mathbf{h}_i^{(l+1)} = \sigma(\rho^{(l)}\mathbf{h}_i^{(l)}\mathbf{W}^{(l)} + (1-\rho^{(l)})\mathbf{h}_{\mathcal{N}(i)}^{(l)}\mathbf{W}^{(l)}) \tag{3.2}$$

其中$\rho^{(l)} \in (0,1)$。$\rho^{(l)}$的值越大，节点自身就会拥有和邻居相比更大的权重。$\rho^{(l)}$趋向于0时，表示仅利用邻居信息而忽略节点本身的特征，同样，$\rho^{(l)}$趋向于1时，模型将趋向于仅利用原始节点特征、忽略图拓扑结构的MLP模型。为了简化计算，$\rho^{(l)}$在图中的所有节点共享，并且由网络通过梯度下降学习得到。

更进一步，我们可以将自环权重参数$\rho^{(l)}$扩展为门控机制[8]。即在更新当前节点时，用一个更新门$\mathbf{z} \in \mathbb{R}^d$（$d$是当前层的输出维数）来控制要从当前节点在前一层中的隐状态携带多少信息进入下一层。节点i的激活可以如下计算：

$$\mathbf{h}_i^{(l+1)} = \sigma(\mathbf{z}^{(l)} \odot (\mathbf{h}_i^{(l)}\mathbf{W}^{(\mathbf{l})}) + (1 - \mathbf{z}^{(l)}) \odot (\mathbf{h}_{N(i)}^{(l)}\mathbf{W}^{(\mathbf{l})})) \tag{3.3}$$

其中⊙表示哈达马乘积。

将若干层上述网络叠加在一起，我们就得到一个完整的GRGNN模型。对于多标签分类问题，GRGNN的最后一层的非线性函数采用sigmoid函数，即每个节点对于每一类别输出一个0到1之间的值，表示节点属于当前类别的可能性。

值得一提的是，虽然GRGNN借用了TransE的思想$\mathbf{h} + \mathbf{r} \approx \mathbf{t}$，但并不一定要使用TransE训练得到的图嵌入作为网络的输入特征。在无特征图中，采用随机向量或者独热编码作为初始特征，也可以用GRGNN取得较好的分类效果。然而，采用不同的特征可能会导致分类效果的不同，另外，不同特征对模型深度的敏感性也不同，更详细的实验结果将在下一节中展示。

3.3 实验

3.3.1 数据集

我们采用两个数据集—FB15K和DB10K—来评估GRGNN在实体多标签分类问题上的效果。数据集的统计信息如表3.2所示。

先前已经介绍过FB15K[4]数据集。除了FB15K之外，我们引入一个新的数据集DB10K，提取自DBpedia[20]。DBpedia是从Wikipedia提取的大规模知识库，其中包含超过4亿个事实，描述了370万个实体。具体来说，我们选取一个实体作为种子实体，从种子实体开始，使用广度优先搜索来生成所有要包含在输出子图中的实体。然后提取这些实体之间的边。我们选择出现次数不少于100次且小于$n/2$次的类别，其中n是提取的节点数。最后，DB10K由10000个实体和54991个三元组组成，包含49种类型和189种关系。

GCN模型需要先对知识图谱进行预处理，我们对原先的预处理方法[37]稍作修改。原来的方法中边的权重均为1，如果一个实体点和一个关系点间存在多条边，我们将边出现的次数作为权重。

3.3.2 设置

首先将二层网络组成的GRGNN与其他方法（包括GCN和MLP）进行比较。由于原始GCN在输出层使用*softmax*函数进行分类（每个节点只有一种类型），需要将非线

表 3.2　数据集统计信息

	节点数量	关系数量	三元组数量	节点（分割后）	边（分割后）	标签率
FB15K	14,951	1,345	592,213	17,641	308,006	0.083
DB10K	10,000	189	108,473	10,378	54,991	0.166

性更改为*logistic sigmoid*激活以进行多标签分类。评估指标为Micro F1分数。然后，我们分析不同层数的模型性能。

在为训练集拆分数据集时，我们遵循以下准则：每个类别的正样本出现至少100次。得到的数据中，FB15K的训练集包含1245个节点，DB10K的训练集包含1664个节点。额外抽取500个节点作为验证集。验证节点仅用于提前停止训练从而防止过拟合。

有很多特征可以用作GRGNN的输入，例如随机特征，独热编码和TransE训练得到的嵌入。除此以外，还可以将这些特征串连起来以提高表达能力。

我们在FB15K数据集上优化了架构超参数，然后将其重用于DB10K。对于GCN和MLP模型，dropout比率设置为0.1，L2正则化设为$1{\cdot}10^{-5}$，隐藏单元数设为128。对于GRGNN模型，由于尺寸和稀疏性的不同，参数会根据初始特征而变化。对于随机特征和TransE特征，我们使用以下超参数：$1{\cdot}10^{-5}$用于L2正则化，64用于隐藏单元数，并且不采用dropout；对于独热编码，dropout比率为0.5，L2正则化权重$1{\cdot}10^{-5}$，隐藏单元数为256；对于TransE和独热编码串连特征，dropout率为0.3，$1\cdot10^{-5}$的L2正则化和256个隐藏单元。

所有模型的权重矩阵都采用Glorot初始化[10]，采用初始学习率为0.01的Adam SGD优化器[17]最大程度地减少训练节点上的交叉熵损失。停止策略的patience为100，来避免过度拟合，也就是说，当验证集上的micro-F1分数持续下降100个epoch时，就认为优化已完成。然后，使用在验证集上表现最佳的参数对测试节点进行评估。

3.3.3　结果

结果总结在表3.3和表3.4中。报告的数字以百分比表示Micro F1分数。我们使用四种初始特征测试所有三个模型。TransE嵌入是用于实体和关系的100维向量，随机特征与TransE特征具有相同的维度，其值是从(−1fi1)中随机选择的，独热编码为图中的每个节点分配唯一特征，串连特征由TransE和one-hot拼接而成。对于GCN，对数据先进行预处理[37]，由于每个关系都分为两个节点，因此我们将$\mathbf{r}$和$-\mathbf{r}$分配给每对关系节点作为TransE特征。

结果表明，我们的方法在两个数据集上均明显优于GCN和MLP。对于FB15K，使用串连特征的GRGNN模型可获得最佳结果，因为对于DB10K，则是使用独热编码的GRGNN模型效果最好。

表 3.3　FB15K的多标签分类准确率

方法	TransE	Randomized	One-hot	Concatenated
GCN	74.76	73.13	75.69	73.91
MLP	78.12	-	-	77.98
GRGNN	76.88	73.28	77.89	**80.10**

表 3.4　DB10K的多标签分类准确率

方法	TransE	Randomized	One-hot	Concatenated
GCN	86.71	93.53	93.18	94.96
MLP	93.14	-	-	89.93
GRGNN	95.61	95.63	**97.07**	96.55

3.3.3.1　模型深度的影响

众所周知，在计算机视觉和其他传统AI领域中，神经网络可以堆叠数百层以获得更好的性能，因为更深的结构具有更多可以提高表达能力的参数。但是，在大多数情况下，GNN的层数不超过三层。Li等人21证明图卷积是拉普拉斯平滑的一种特殊形式。因此，随着图卷积层数的增加，连通分支内节点的特征趋于收敛到相同的值，这使得不同类型的节点难以区分。GCN通过两个卷积层获得最佳性能，并且在采用更深的模型时会出现收敛特征以及过度拟合的问题。

在这里，我们研究模型深度（层数）对分类性能的影响。我们报告多达8层模型的结果。仅在网络的第一层和最后一层采用Dropout，而L2正则化仅针对第一层进行计算。每层中隐藏单元的数量相同。超参数的选择与上一节相同。

FB15K上的实验如图3.3所示，阴影区域表示标准误差。测试了使用TransE特征和独热特征的GRGNN模型。在两到三层的情况下，模型的性能相差不大。但是，与TransE特征相比，在更深层次的模型中使用独热特征会导致F1得分低得多。此外，采用TransE特征的GRGNN对模型深度不敏感，正如图中所示，当层数增加时性能不会下降太多。在

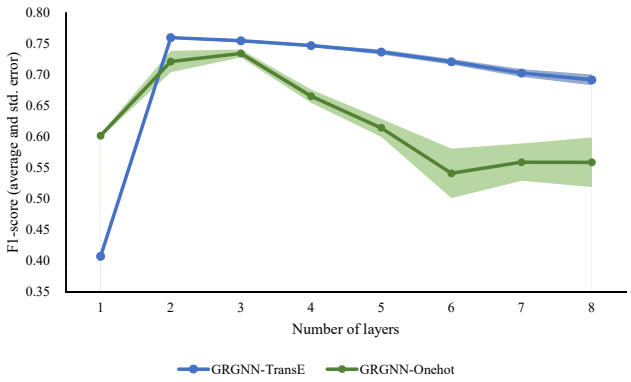


图 3.3　Influence of model depth (number of layers) on FB15K.

我们的模型中，这种现象可以部分由门控机制来解释。由于更新门控制由当前节点携带并由邻居更新的信息，因此不同节点的隐藏状态不一定会收敛相同的值。因此，这种机制和不敏感的特征可能为更深，更复杂的GNN提供机会。

第四章　结论和展望

本文对图神经网络及其在知识图谱实体分类问题的效果进行了深入研究，具体可以分为以下两个方面：

首先对几个图神经网络模型的性能进行了实验评估，包括GCN、GAT、AGNN、Stochastic GCN、Co-Training GCN和Self-Training GCN。在大部分情况下，GAT模型的准确率最高。Co-Training系列方法在有标签节点非常少时可以提升模型效果。Stochastic GCN中的CV和CVD模型减少了训练时间，同时保持了GCN的准确率。而后对注意力机制和数据划分做进一步分析，发现AGNN和GAT训练得到的注意力系数非常不同，AGNN更多地传递同类节点之间的信息，而GAT的注意力系数与原始GCN的传播矩阵区别不大，GAT的多头机制对于模型准确率提升起到了非常大的作用。此外，发现测试准确率与训练集中一致节点的比例显著相关，这对于数据标注具有一定的指导作用。

其次重点探讨了图神经网络模型如何解决知识图谱上的实体分类模型。GCN模型无法直接处理知识图谱中边上的信息，因此需要对数据进行预处理。传统的预处理方案具有一定的缺陷，提出一种新的预处理方法，然后再用GCN模型进行分类，准确率有所提升。为了更好地处理知识图谱中边上丰富的语义信息，提出一种新的图神经网络模型：GRGNN。GRGNN结合了TransE与GCN的思想，整合节点的邻居节点与邻居边的信息，并引入门控机制来控制当前节点特征的更新。在FB15K和DB10K数据集上的实验显示，GRGNN模型的分类效果好于GCN和TransE。未来的工作主要集中在以下几个方面：增强模型的可扩展性，当前模型能够处理的数据大小有限，而完整的知识图谱可能包括上百万节点；在对知识图谱进行实体分类时，结合自然语言处理方法对图谱中的文本信息加以利用；进一步改进模型，探索能够适应深层模型的机制。

参考文献

[1] [2] Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. "*Neural machine translation by jointly learning to align and translate*". In: *ICLR*. **2014**.

[2] Jonathan Berant, Andrew Chou, Roy Frostig *et al.* "*Semantic parsing on freebase from question-answer pairs*". In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. **2013**: 1533–1544.

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh *et al.* "*Freebase: a collaboratively created graph database for structuring human knowledge*". In: *SIGMOD*. **2008**: 1247–1250.

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran *et al.* "*Translating embeddings for modeling multi-relational data*". In: *NIPS*. **2013**: 2787–2795.

[5] Dan Busbridge, Dane Sherburn, Pietro Cavallo *et al.* "*Relational Graph Attention Networks*". *arXiv preprint arXiv:1904.05811*, **2019**.

[6] Andrew Carlson, Justin Betteridge, Bryan Kisiel *et al.* "*Toward an architecture for never-ending language learning*". In: *Twenty-Fourth AAAI Conference on Artificial Intelligence*. **2010**.

[7] Jianfei Chen, Jun Zhu and Le Song. "*Stochastic Training of Graph Convolutional Networks with Variance Reduction*". In: *ICML*. **2018**: 941–949.

[8] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre *et al.* "*Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation*". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. **2014**: 1724–1734.

[9] Silviu Cucerzan. "*Large-scale named entity disambiguation based on Wikipedia data*". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. **2007**: 708–716.

[10] Xavier Glorot and Yoshua Bengio. "*Understanding the difficulty of training deep feedforward neural networks*". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. **2010**: 249–256.

[11] Marco Gori, Gabriele Monfardini and Franco Scarselli. "*A new model for learning in graph domains*". In: *IJCNN*. **2005**: 729–734.

[12] Aditya Grover and Jure Leskovec. "*node2vec: Scalable feature learning for networks*". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. **2016**: 855–864.

[13] Will Hamilton, Zhitao Ying and Jure Leskovec. "*Inductive representation learning on large graphs*". In: *NIPS*. **2017**: 1024–1034.

[14] David K Hammond, Pierre Vandergheynst and Rémi Gribonval. "*Wavelets on graphs via spectral graph theory*". *Applied and Computational Harmonic Analysis*, **2011**, *30*(2): 129–150.

[15] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich *et al.* "*YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia*". *Artificial Intelligence*, **2013**, *194*: 28–61.

[16] Guoliang Ji, Shizhu He, Liheng Xu *et al.* "*Knowledge graph embedding via dynamic mapping matrix*". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. **2015**: 687–696.

[17] Diederik P Kingma and Jimmy Ba. "*Adam: A method for stochastic optimization*". *arXiv preprint arXiv:1412.6980*, **2014**.

[18] Thomas N. Kipf and Max Welling. "*Semi-Supervised Classification with Graph Convolutional Networks*". In: *ICLR*. **2017**.

[19] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. "*Imagenet classification with deep convolutional neural networks*". In: *NIPS*. **2012**: 1097–1105.

[20] Jens Lehmann, Robert Isele, Max Jakob *et al.* "*DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia*". *Semantic Web*, **2015**, *6*(2): 167–195.

[21] Qimai Li, Zhichao Han and Xiao-Ming Wu. "*Deeper insights into graph convolutional networks for semi-supervised learning*". In: *AAAI*. **2018**.

[22] Yujia Li, Daniel Tarlow, Marc Brockschmidt *et al.* "*Gated graph sequence neural networks*". *arXiv preprint arXiv:1511.05493*, **2015**.

[23] Yankai Lin, Zhiyuan Liu, Maosong Sun *et al.* "*Learning entity and relation embeddings for knowledge graph completion*". In: *Twenty-ninth AAAI conference on artificial intelligence*. **2015**.

[24] Tomas Mikolov, Kai Chen, Greg Corrado *et al.* "*Efficient estimation of word representations in vector space*". In: **2013**.

[25] Maximilian Nickel, Volker Tresp and Hans-Peter Kriegel. "*A three-way model for collective learning on multi-relational data*". In: *Proceedings of the 28th International Conference on International Conference on Machine Learning*. **2011**: 809–816.

[26] Bryan Perozzi, Rami Al-Rfou and Steven Skiena. "*Deepwalk: Online learning of social representations*". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. **2014**: 701–710.

[27] Franco Scarselli, Marco Gori, Ah Chung Tsoi *et al.* "*The graph neural network model*". *IEEE Transactions on Neural Networks*, **2009**, *20*(1): 61–80.

[28] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem *et al.* "*Modeling relational data with graph convolutional networks*". In: *European Semantic Web Conference*. **2018**: 593–607.

[29] Jian Tang, Meng Qu, Mingzhe Wang *et al.* "*Line: Large-scale information network embedding*". In: *Proceedings of the 24th international conference on world wide web*. **2015**: 1067–1077.

[30] Kiran K Thekumparampil, Chong Wang, Sewoong Oh *et al.* "*Attention-based graph neural network for semi-supervised learning*". *arXiv preprint arXiv:1803.03735*, **2018**.

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel *et al.* "*Complex embeddings for simple link prediction*". In: *International Conference on Machine Learning*. **2016**: 2071–2080.

[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova *et al.* "*Graph Attention Networks*". In: *ICLR*. **2018**.

[33] Zhen Wang, Jianwen Zhang, Jianlin Feng *et al.* "*Knowledge graph embedding by translating on hyperplanes*". In: *Twenty-Eighth AAAI conference on artificial intelligence*. **2014**.

[34] Xiao-Ming Wu, Zhenguo Li, Anthony M So *et al.* "*Learning with partially absorbing random walks*". In: *Advances in neural information processing systems*. **2012**: 3077–3085.

[35] Ruobing Xie, Zhiyuan Liu, Jia Jia *et al.* "*Representation learning of knowledge graphs with entity descriptions*". In: *AAAI*. **2016**.

[36] Bishan Yang, Wen-tau Yih, Xiaodong He *et al.* "*Embedding entities and relations for learning and inference in knowledge bases*". In: **2015**.

[37] Zhilin Yang, William Cohen and Ruslan Salakhudinov. "*Revisiting Semi-Supervised Learning with Graph Embeddings*". In: *International Conference on Machine Learning*. **2016**: 40–48.

[38] Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He *et al.* "*Semantic parsing via staged query graph generation: Question answering with knowledge base*". **2015**.

[39] Jie Zhou, Ganqu Cui, Zhengyan Zhang *et al.* "*Graph neural networks: A review of methods and applications*". *arXiv preprint arXiv:1812.08434*, **2018**.

[40] 作者。“标题”[J]。期刊，2014-04-01。

附录 A　附件

pkuthss 文档模版最常见问题：

`\cite`、`\parencite` 和 `\supercite` 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：**test-en**，[40]、[test-en, 40]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 `texdoc pkuthss`“问题及其解决”一章“上游宏包可能引起的问题”一节中关于 biber 的说明。

致谢

pkuthss 文档模版最常见问题：

`\cite`、`\parencite` 和 `\supercite` 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：**test-en**，[40]、[test-en, 40]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 `texdoc pkuthss`“问题及其解决”一章“上游宏包可能引起的问题”一节中关于 biber 的说明。

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：　　日期：　　年　　月　　日

学位论文使用授权说明

（必须装订在提交学校图书馆的印刷本）

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
- 因某种特殊原因须要延迟发布学位论文电子版，授权学校在 □ 一年 / □ 两年 / □ 三年以后在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名：　　导师签名：　　日期：　　年　　月　　日