大规模知识图谱表示学习

（申请清华大学工学学士学位论文）

培 养 单 位：计 算 机 科 学 与 技 术 系
学　　科：计 算 机 科 学 与 技 术
研　究　生：韩　旭
指 导 教 师：刘　知　远　助　理　教　授

二〇一七年五月

Large-scale Knowledge Graph Representation Learning

Thesis Submitted to

Tsinghua University

in partial fulfillment of the requirement

for the professional degree of

Bachelor of Engineering

by

Han Xu

(Computer Science and Technology)

Thesis Supervisor : Assistant Professor Liu Zhiyuan

May, 2017

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：

清华大学拥有在著作权法规定范围内学位论文的使用权，其中包括：（1）已获学位的研究生必须按学校规定提交学位论文，学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（2）为教学和科研目的，学校可以将公开的学位论文作为资料在图书馆、资料室等场所供校内师生阅读，或在校园网上供校内师生浏览部分内容。

本人保证遵守上述规定。

（保密的论文在解密后应遵守此规定）

作者签名：_____　　导师签名：_____

日　　期：_____　　日　　期：_____

摘 要

近年来，在人工智能及数据挖掘的部分领域中，为了抽象现实世界的知识并以一种统一载体进行结构化存储，学术界和工业界提出了知识图谱，并在此基础上进行了广泛研究。知识图谱中蕴含的丰富结构化信息对许多任务有很好的辅助效果，如问答系统、网络搜索，逻辑推演等。在广泛发挥作用的同时，知识图谱也存在部分问题亟需解决：第一，图谱虽然总量巨大但仍有更大一批知识存在缺失，需要进行深入的填补；第二，如何将图谱的结构化信息融入到当下松散的特征模型中去，需要有效的融合算法；第三，知识图谱巨大的规模对算法时间复杂度要求较高，需要高效的模型能够在有限的时间内对图谱进行操作。而解决这些问题的核心是能够高效、准确地将图谱表示成计算机能够理解的数字抽象。本文针对大规模知识图谱表示学习提出了一个训练框架，其特点主要集中在以下几点：第一，通过底层优化以及图谱划分，将以往的图谱模型转化成多线程训练模型，未来可以衍生为分布式的训练方法；第二，提出加权点采样算法和位移负例采样算法来取代原有的边采样及负例采样，从而能够对图中不同的实体和事实赋予不同的注意，缓解幂率分布带来的长尾影响，以及尽可能的合并运算以便进行加速；第三，采用了平行结合的方式与文本神经网络模型结合，对比于传统的串行结合方式，能够更快的引入文本信息来丰富图谱内容并进行信息融合。相关的实验表明，本文提出的框架能够在效果不降低的情况下极快的加速图谱学习，并和文本模型有很好的融合效果使得融合之后两者都有显著效果提升。我们已经开源了部分代码[①]以便于其他领域研究者使用。

关键词：大规模；知识图谱；表示学习；联合学习；并行加速

① https://github.com/thunlp/Fast-TransX

Abstract

In recent years, in some fields especially artificial intelligence and data mining, people organize structural knowledge about the world under the unified framework and construct various large-scale knowledge graphs for the knowledge storage. Because of rich structural information, knowledge graphs are playing an important role in many applications such as question answering, web search, logical inference, etc. However, there are also some problems need to be solved: (1) Most large-scale knowledge graphs are usually far from completion and need to be further extended. (2) We need effective methods to fuse knowledge information and existing text features so that we can incorporate knowledge graphs into practical applications. (3) the enormous scale of realistic knowledge graphs need efficient models to operate on the graphs within the limited time. The key to solving these problems is to learn large-scale knowledge graph representations. In this paper, we propose a framework to embed large-scale knowledge graphs and incorporate them into neural text models, which focused on the following points: (1) Based on the existing knowledge models, we change the underlying designs for acceleration. We also divide the overall knowledge graph into several parts and adopt these models for multi-threaded training. (2) We propose the weighted node-based sampling and offset-based negative sampling to replace original edge-based sampling and negative sampling algorithms. With the new sampling mechanism, we alleviate the long tail effect caused by power-law distribution, and merge arithmetic operations for acceleration. (3) We use a parallel training method to combine neural netwroks and knowledge graph embedding models so that text can be fastly fused with knowledge graph embeddings. The experimental results show that our framework can accelerate the existing knowledge modes dramatically without reducing the accuracy. At the same time, our method can effectively perform joint representation learning and obtain more informative knowledge and text representation, which significantly outperforms other baseline methods. Some resource codes have been released [①] so that researchers can easily adopt our framework for their own works.

Key words: Large-scale; Knowledge Graph; Representation Learning; Joint Learning; Parallel Acceleration

① https://github.com/thunlp/Fast-TransX

目 录

主要符号对照表

KG	知识图谱 (Knowledge Graph)
KGC	图谱填充 (Knowledge Graph Completion)
RE	关系抽取 (Relation Extraction)
E, e	知识图谱实体集合和具体实体，$e \in E$
R, r	知识图谱关系集合和具体关系，$r \in R$
T	知识图谱三元组集合，$(h, r, t) \in T$
G	知识图谱集合，定义为 $G = \{E, R, T\}$
D, s	文本集合和具体的文本句子，$s \in D$
V	文本集合 D 中的词汇集合
r_s	句子 S 的语义所对应的关系，$r_s \in R$
$\mathbf{h}, \mathbf{t}, \mathbf{r}, \mathbf{w}$	实体，关系和单词的向量，$\mathbf{h}, \mathbf{t}, \mathbf{r}, \mathbf{w} \in \mathbb{R}^{k_w}$ $(h, t \in E, r \in R, w \in V)$，其余加粗小写字母同样表示自身对应的向量
k_w	实体，关系和单词的向量维度

第 1 章　引言

1.1　研究背景

从古至今，信息的主要交流方式是基于语言的，人类知识的长期传承也是通过语言文字这个载体进行下去的。可以说，从人类早期的莎草纸、羊皮卷、竹简到之后的纸张，上面的文本内容成了知识千年以来突破时空的重要途径。而伴随着互联网在二十一世纪的蓬勃发展，信息的传递速度、传递带宽、一次载体能够传递的信息量都得到了极大提升。信息的增长趋势也从过去的线性级别增长变成了指数级别的增长，这意味着每天都有成千上万的信息涌入了网络之中。这些海量的数据一方面使得信息的来源变的空前丰富，但同时也使得我们对信息的把握、筛选遇到了巨大障碍。信息爆炸同时伴随噪音爆炸，在这样的环境下，从海量的嘈杂的文本中提取知识是不容易的。在这样的背景下，为了有效地获取知识，知识图谱（KG）的概念被提出并在学术界和工业界都受到了广泛的关注。

知识图谱（Knowledge Graph, KG），某些场景下也被称为知识库（Knowledge Base，KB），是一种将现实世界中人类的知识结构化之后形成的知识系统。在知识图谱中，大量的知识，诸如开放数据库和百科全书中的信息，通常以关系数据集合的形式被表达出来。而在关系数据集合中，基本事实被抽象为实体（Entity），而规则、逻辑、推理等关联性的信息则被抽象为实体间的关系（Relation）。若将实体对应于点，关系对应于边，则这些知识可以进一步以图的形式呈现，从而可以被计算机高效的使用，而这也是研究知识图谱的意义所在。这种将实体和抽象概念结构化成多关系数据库的模式也是近年来被大力提倡的，我们接触到的信息，尤其是文本信息突破了以往字符串线形构成的基本形式，可以以实体和关系构成的网状形式存在。

目前知识图谱已经作为人工智能领域的一项基础核心技术，被广泛引入到信息检索（Information Retrieval，IR）、问答系统（Question Answering，QA）、推荐系统（Recommender System，RS）等任务上。图谱中优质的结构化知识信息，能够指导我们的智能模型具备更深层的事物理解、更精准的任务查询以及可能的逻辑推理能力，从而在这些知识驱动应用中起着至关重要的作用。可以毫不夸张的说，正是由于这些结构化知识图谱的存在，我们建模实体以及实体之间的关系变的容易，让计算机能够理解知识、运用知识甚至于发掘创造知识的想法也逐渐具有了可行性。

而随着时间的积累和相关工作者长期的工作，结合机器自动标注、专家标注

和开放平台编辑校对等多种方法，现在已经构建出一些诸如图1.1 ① 中的高质量的大规模知识图谱，诸如 WordNet[1]，YAGO[2]，DBPedia[3]，Freebase[4]，Wikidata[5] 以及 Knowledge Vault[6]，并且被投入到部分相关研究场景中。截止到 Freebase 停止更新为止，Freebase 中收集了超过 2 亿个的实体，在其停止维护后这些信息正在被陆续迁移到 Knowledge Vault 和 Wikidata 中。经过维基社区的过滤和校对，截止目前，Wikidata 中也有超过 2600 万个高质量实体存在。与此同时，国内从事互联网领域尤其是和信息检索直接相关的企业也对知识图谱进行了投入，百度知心和搜狗知立方作为典型的中文知识图谱被构建出来并被使用到智能应用产品中进行知识驱动。

图 1.1　一些常用的大规模知识图谱[1–5]

知识图谱将具象事物与抽象概念表示为实体，将实体之间的联系表示为关系，并以（头实体，关系，尾实体）的形式表述知识。例如，“马克 · 吐温出生于佛罗里达州”在知识图谱中被表述为（马克·吐温，出生于，佛罗里达州）；“北京市下辖海淀区”在知识图谱中被表述为（北京市，区划管辖，海淀区）等等。其中马克·吐温、

① WordNet: http://wordnet.princeton.edu/
YAGO: http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/
DBPedia: http://wiki.dbpedia.org/
Freebase: https://developers.google.com/freebase/
Wikidata: https://www.wikidata.org/wiki/Wikidata:Main_Page/

佛罗里达州、北京市、海淀区即为实体，而出生于、区划管辖则是实体间的关系。一般来说，现在公开的知识图谱都是以这样的三元组 (triple fact) 的形式抽象知识，并采用类似于万维网联盟（W3C）发布采用的资源描述框架 (Resource Description Framework, RDF) 进行储存。

1.2　研究内容

1.3　相关工作

1.4　本文贡献

本文贡献总结如下:

（1）通过底层实现的优化以及对知识图谱的划分，将以往的图谱模型改造成基于多线程的训练模型，在未来可以进一步衍生为分布式的训练模型，从而能够高效的对大规模知识图谱进行学习。

（2）提出了加权点采样算法和位移负样例采样算法来取代原有的边采样及负例采样算法。能够在大规模图谱的幂率分布下对图中不同的实体和事实赋予不同的注意度，缓解幂率分布带来的长尾影响。二点采样的训练方式，可以在训练过程中尽可能的合并算术运算从而进一步加速。

（3）采用了平行结合的方式与文本神经网络模型融合，对比于传统的串行结合方式，这样的模型能够更快的引入文本信息来丰富图谱内容并进行信息融合。在此基础上我们提出了一套联合学习的方法，使得图谱和文本的信息融合是双向的，图谱和文本模型融合之后两者都有显著效果提升

（4）本文的相关工作提供了开源代码并对之前领域内的相关工作进行整理和总结，为之后相关领域需要使用知识图谱的后续研究打好基础，提供便利。

1.5　论文组织

本文在第二章中综述知识图谱的发展轨迹，并对已有的知识图谱表示学习模型进行介绍、分析，从而能够对这一任务的相关工作进行梳理和总结。第三章介绍过去几年中被提出的知识图谱和文本模型的结合方法及相关任务，其中重点综述神经网络引入后深度模型的发展和联合学习模型的发展。第四章介绍我们提出的在大规模知识图谱背景下进行表示学习的框架，以及在此框架下和文本部分进行并行融合的联合学习方法。第五章介绍实验所用的数据并给出量化的评测结果，验证本文提出框架的准确性、可靠性和运行效率。

第 2 章　基于并行的大规模知识图谱表示学习框架

2.1　简介

2.2　算法框架

在这一章节内容里，我们主要介绍我们设计出的开放式知识图谱嵌入结构 OpenKE。内容包括 OpenKE 在并行模式下实现的大规模知识图谱表示学习框架，以及已有的知识图谱模型在框架 OpenKE 下进行实现和融合的具体方式和实施细节。整体的模型骨架我们可以在图2.1中看到。在整个框架中，底层操作和上层模型是独立的，上层模型可以自由选择和适配，而底层结构则是我们进行了大量优化之后的并行架构。进一步讲，这样一个解耦合的框架使得一个新提出的模型也可以在无需过多关注底层的情况下得到高效实现，具有高自由度和高效率的特点，这些都将在之后的内容里详细铺开。当然，在介绍具体细节之前，我们先引入一些符号体系和重要概念。

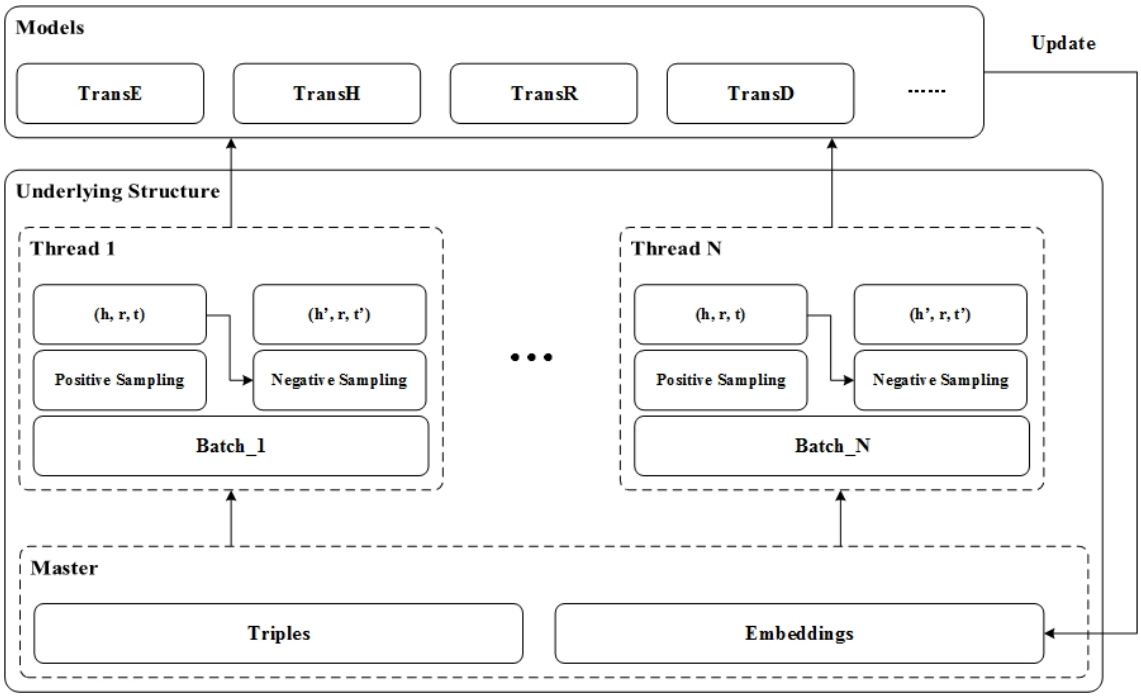

图 2.1　基于并行的大规模知识图谱表示学习框架 OpenKE 的结构示意图

2.2.1　符号体系和重要概念

我们将整个知识图谱定义为一个由实体集、关系集和事实三元组集合共同组成的大集合，即 $G = \{E, R, T\}$，这里 E、R 和 T 分别表示实体集合、关系集合和事实三元组集合。对于事实三元组集合中的任意事实 $(h, r, t) \in T$，这个三元组表明头实体 $h \in E$ 和尾实体 $t \in E$ 之间存在一个逻辑上的关联 $r \in R$。由于表示学习会将实体和关系都嵌入到连续空间中去并用对应的向量来表示他们的语义信息，所以对于任意的实体或者关系 $h, t \in E$ 或 $r \in R$，我们都用它们的加粗字母 $\mathbf{h}, \mathbf{t}, \mathbf{r}$ 来表示它们的向量，这里的向量也可以称为嵌入、嵌入向量、表示、嵌入表示等。

2.2.2　知识图谱表示学习模型

对于知识图谱表示学习，模型需要做的就是将实体和关系嵌入到连续空间中，从而利用空间向量来表达它们之间存在的语义关联。在这里，我们先以一个统一的数学视角来归纳这些模型方法，从而方便之后在底层上进行统一的实现。然后我们接着给出在具体实现时，以 TransE 为代表的平移模型各自之间的不同之处。

在知识图谱表示学习模型中，通常先定义一个能量函数来衡量事实三元组的合理程度。更准确的说，对于任意一个给定的三元组 (h, r, t)，模型会为之定义一个能量函数 $S(h, r, t)$。如果三元组是合理的，比如 $(h, r, t) \in T$，此时的能量函数将返回一个较低的值。相反，如果三元组是不成立的，那么能量函数则会返回一个较高的值。这个函数通常情况下和 h, r, t 在空间上的距离具有相关性，换句话说，存在关联的实体与关系在空间上也是相近的。

对于包括 TransE 及其一系列的拓展模型在内的基于平移的知识图谱表示模型，会定义一个潜在的向量表示 r_{ht} 来具象化头实体和尾实体 (h, t) 之间的潜在关系。这些模型之所以叫做基于平移的表示模型，是因为它们都遵循 ‘‘关系在空间中是实体向量间的平移变换’’ 这个基本假设。在这个假设下，如果三元组成立，那么会有潜在向量 $\mathbf{r}_{ht}$ 和显式的关系向量 $\mathbf{r}$ 在空间上极为接近。换句话说，我们获取头实体和尾实体之间关系的过程，就是一个寻找与潜在关系向量 $\mathbf{r}_{ht}$ 在空间上最近的关系向量的过程，由此我们将能量函数定义为两者的空间距离：

$$S(h, r, t) = \|\mathbf{r} - \mathbf{r}_{ht}\|_{L1/L2} \tag{2-1}$$

这里，能量函数 $S(h, r, t)$ 可以用 L1 距离衡量，也可以用 L2 距离衡量。在能量函数的基础上，我们可以进一步得到一个基于边界值优化的损失函数来作为我们的

训练目标，并有如下公式：

$$\mathcal{L}(G) = \sum_{(h,r,t)\in T} \sum_{(h',r,t')\in T'} \left[\gamma + S(h,r,t) - S(h',r,t')\right]_+ \tag{2-2}$$

这里，$[x]_+$ 是这样一个函数，如果 x 是正数的话，那么返回值就是 x，反之返回值为 0。$\gamma > 0$ 是一个边界值，用来约束损失函数的训练。$S(h,r,t)$ 是正例三元组的能量函数得分，而 $S(h',r,t')$ 是负例三元组的能量函数得分，T' 是负例三元组的集合。这样的损失函数表明，我们希望正确的三元组和负例三元组的能量函数尽可能拉开差距，但又不能使训练过拟合，所以选定一个边界值并在其范围内进行最大化区分度的训练，这也是这些模型一脉相承的设计思路。而对于负例三元组集合 T' 有：

$$T' = \{(h',r,t)\} \cup \{(h,r,t')\}, \tag{2-3}$$

负例三元组集合是通过将正例三元组 $(h,r,t) \in T$ 中的实体替换成其他实体集合中的实体 $h',t' \in E$ 来构造的。

在有了上述统一的损失函数和训练方法之后，我们可以基于几乎相同的边界值训练模式在我们的框架中来实现 TransE[7]、TransH[8]、TransR[9]、TransD[10] 等诸多模型。这些模型主要的区别在于它们潜在关系向量 $\mathbf{r}_{ht}$ 计算方式的不同，其余部分和上述模型归纳一致。对于这些模型之间的区别和变化，我们接下来将详细罗列并介绍。

TransE. 在 TransE 中，对于每个三元组 (h,r,t)，我们定义出以下的潜在关系向量 $\mathbf{r}_{ht}$ 和能量函数 $S(h,r,t)$：

$$\begin{aligned} \mathbf{r}_{ht} &= \mathbf{t} - \mathbf{h}, \\ S(h,r,t) &= \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{L1/L2} \end{aligned} \tag{2-4}$$

由此可以看出，所谓的平移假设，实际上就是希望空间上满足 $\mathbf{h}_r + \mathbf{r} \approx \mathbf{t}_r$。

TransH. 在 TransH 中，对于每个给定的实体，其在不同的关系环境下具有不同的嵌入表示。对于每个三元组 (h,r,t)，我们定义出以下的潜在关系向量 $\mathbf{r}_{ht}$ 和能量函数 $S(h,r,t)$：

$$\mathbf{h}_r = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h} \mathbf{w}_r, \tag{2-5}$$

$$\begin{aligned}
\mathbf{t}_r &= \mathbf{t} - \mathbf{w}_r^\top \mathbf{t} \mathbf{w}_r, \\
\mathbf{r}_{ht} &= \mathbf{t}_r - \mathbf{h}_r, \\
S(h,r,t) &= \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_{L1/L2}
\end{aligned}$$

这里 $\mathbf{w}_r$ 是一个归一化之后的向量，用来作为关系 r 所在平面的法向量。

TransR. 在 TransR 中，实体和关系是在不同的空间中被学习得到的，并且实体之间的平移性质是在关系向量所在的空间中进行的。对于每个三元组 (h,r,t)，我们定义出以下的潜在关系向量 $\mathbf{r}_{ht}$ 和能量函数 $S(h,r,t)$：

$$\begin{aligned}
\mathbf{h}_r &= \mathbf{M}_r \mathbf{h}, \\
\mathbf{t}_r &= \mathbf{M}_r \mathbf{t}, \\
\mathbf{r}_{ht} &= \mathbf{t}_r - \mathbf{h}_r, \\
S(h,r,t) &= \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_{L1/L2}
\end{aligned} \tag{2-6}$$

这里 $\mathbf{M}_r$ 是关系 r 的映射矩阵，用来将实体从实体空间映射到关系空间中，以方便平移性质的实现。

TransD. 在 TransD 中，实体和关系同样是在不同的空间中被学习得到的。TransD 和 TransR 是十分相似的，但是其采用了动态的映射矩阵来进行映射。对于每个三元组 (h,r,t)，我们定义出以下的潜在关系向量 $\mathbf{r}_{ht}$ 和能量函数 $S(h,r,t)$：

$$\begin{aligned}
\mathbf{h}_r &= \mathbf{M}_{rh} \mathbf{h}, \\
\mathbf{M}_{rh} &= \mathbf{r}_p \mathbf{h}_p^\top + \mathbf{I}, \\
\mathbf{t}_r &= \mathbf{M}_{rt} \mathbf{t}, \\
\mathbf{M}_{th} &= \mathbf{r}_p \mathbf{t}_p^\top + \mathbf{I}, \\
\mathbf{r}_{ht} &= \mathbf{t}_r - \mathbf{h}_r, \\
S(h,r,t) &= \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_{L1/L2}
\end{aligned} \tag{2-7}$$

这里 $\mathbf{r}_p$ 和 $\mathbf{h}_p, \mathbf{t}_p$ 都是用来生成映射矩阵 $\mathbf{M}_{rh}$ 和 $\mathbf{M}_{rt}$ 的映射向量。通过这样的映射方式，矩阵运算变为了向量运算，在保持了一定映射性质的情况下加快了整体训练速度。

Algorithm 1 并行学习伪代码

输入： 实体和关系集合 E, R，
训练三元组 $T = \{(h, r, t)\}$，
边界值 γ，嵌入空间维度，
训练轮数 $epoches$，线程数 $threads$，一批次训练数据量 $batches$。

初始化
$\mathbf{e} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$，对于任意 $e \in E$；
$\mathbf{r} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$，对于任意 $r \in R$。
初始化
其他参数，比如：
TransH 的 $\mathbf{w}_r$；
TransR 的 $\mathbf{M}_r$；
TransD 的 $\mathbf{r}_p, \mathbf{h}_p, \mathbf{t}_p$。
for $i \leftarrow 1$ to $epoches$ **do**
　在每一个线程中:
　for $j \leftarrow 1$ to $batches/threads$ **do**
　　采样正例 (h, r, t)
　　采样负例 (h', r, t')
　　if $\gamma + S(h, r, t) - S(h', r, t') > 0$ **then**
　　更新梯度 $\nabla[\gamma + S(h, r, t) - S(h', r, t')]_+$
　　end if
　end for
end for
返回训练得到的实体嵌入和关系嵌入

2.2.3　并行结构

在统一了各个知识图谱表示学习模型的形式之后，我们可以发现，这些模型除了上层的能量函数计算方法略有差距外，整体的底层结构，包括采样算法在内都是一致的。所以我们的框架在这些已有模型的底层基础之上，修改了部分结构以便于进行算法加速。其中极为重要的一点就是将知识图谱上的数据划分为若干部分训练三元组集合，并将模型改造成多线程形式来处理每一部分的集合。这样一个并行的学习方法在算法1中给出了伪代码。

我们的并行学习结构采用的是基于数据并行的多线程机制，并且以此为基础

Algorithm 2 基于位移的负例采样算法

预处理：　将所有实体 $e \in E$ 映射为连续整数编码 $num_e \in \{0, ..., \|E\| - 1\}$。

输入：　需要替换的三元组 (h, r, t)，

所有和 h, r 匹配的尾实体集合 $E_{hr} = \{t|(h, r, t) \in T\}$。

对于任意实体 $e \in E_{hr}$，我们用其整数编码来进行排序，其在 E_{hr} 中的排名为 $rank_e \in \{0, ..., \|E_{hr}\| - 1\}$。

1: $p \leftarrow \text{rand}(0, \|E\| - \|E_{hr}\|)$

2: $\hat{e} = \arg\max_{e \in E_{hr}} (num_e - rank_e)$，这里需满足 $num_e - rank_e \leq p$，$num_e - rank_e$ 具有偏序关系所以可用二分来解决。

3: **返回**生成的负例三元组 (h, r, t')，这里 t' 有 $num_{t'} = p + rank_{\hat{e}}$

实现了两种梯度更新的策略来训练模型。其中一种方法是通过无锁策略实现的，即所有的线程共享同一块内存空间，优化损失函数时可以实时地将导数反馈到内存中。因为没有加锁，所以在这个策略下，多个线程是可能出现竞争修改同一块内存的。所有线程共享统一的嵌入空间，直接更新嵌入而不进行同步操作，虽然这降低了梯度下降的质量，但整体上的速度提升效果非常明显。另一方面，我们还实现了一个中心同步梯度的方法。对于每个线程，它会计算其自身部分数据的梯度。当所有的线程都计算完毕后，中心会将整体的梯度进行加和，汇总后得到整体的梯度，然后这个结果将会被统一地反馈到实体和关系的嵌入向量或者其他参数上。

值得注意的是，同样的实体对或者实体关系对，可能在同一训练周期内被不同的线程同时计算到，这些重复运算是可以避免的，所以我们也合并了这些算术运算以便进一步加速我们的底层框架。

2.2.4　基于位移的负例采样算法

对于 TransE 及其一系列的拓展模型，它们的优化过程都是以最小化其基于边界值的损失函数而进行的，即最小化式2-2。从大量的实践中我们发现，负例三元组的选择对最后的模型效果有着至关重要的影响。在前文介绍的负例三元组生成模式下，负例三元组集合 T' 是通过随机使用 E 中实体替换正例三元组的头尾实体来构建的。由于很多关系并不是一对一的模式，这意味着这套替换机制有可能用另一个正确的实体替换了当前的实体，T' 中极有可能含有一些 T 中出现过的三元组。举个例子，替换机制将（美国，总统，奥巴马）替换为（美国，总统，克林顿），两者都是正确的三元组。在实际处理中，如果替换实体之后生成的三元组在 T 中出现，那么这个三元组将不会被用作负例来处理。因此，在原有的负例采样算法中我们会花费大量时间来检验采样出的三元组是否在 T 中出现。为了优化此处的

时间复杂度，我们提出了基于位移的负例采样算法来直接生成负例而无需进行任何检验。我们以替换尾实体为例，将该算法伪代码罗列在算法2中。

在算法操作过程中，对于所有实体，我们用从 0 开始的连续的整数定义它们的整数编码，并且以此排序来使得实体集合具有序列性质，之后我们的各项操作都可以建立在这些整数编码之上。在替换过程中，我们首先随机一个编码。接着，我们采取了这样一个采样思路——如果存在非候选项（即 E_{hr} 中不能用来替换的实体），其整数编码落在随机数的范围内，我们就在随机编码的基础上加对应数量的偏移，从而将所有的非候选项都错开，并得到一个负采样的三元组。直观上讲，这个方法就是将所有的实体排列在一条线上，其中不能替换的实体会将整个直线划分为若干个线段，如果随机落点落在某个线段内则可以直接返回，如果随机落点出现问题落在顶点上的话，我们就将落点偏移使得落点只会在线段之内，这样生成的三元组是一定不会在 T 中出现的。

2.3　实验设计与结果分析

我们选取了知识图谱上的链接预测任务来进行框架性能的评测。在这里，我们一方面会对我们的框架 OpenKE 进行各种性能上的测试，另一方面我们会将结果与一个已经被广泛使用的工具包 KB2E ①进行对比来体现 OpenKE 的高效性。KB2E 是 Lin 等人[9] 在论文发表后开源的工具包，其中实现了 TransE、TransH 以及其自身工作 TransR 在内的诸多知识图谱表示学习模型。KB2E 因为效果稳定而在很多工作中被使用，并且开源在 Github 上可以直接被获取。

在实验部分，所有的测试都是在 16GB 内存的单机上进行的，处理器为 Intel Core i7-6700K，拥有 4 个核心、8 个线程，处理器的基本频率为 4.00GHZ。

2.3.1　实验数据集

在实验中，我们选取了 FB15K 和 WN18 ②这两个数据集合来进行测试。数据集合 FB15K 和 WN18 是知识图谱表示学习模型的主要评测基准，并在过去的大量工作中被广泛使用。这两个数据集合都是从公开的大型知识图谱中经过采样得到的子集，其中 FB15K 的内容是从 Freebase 中抽取出的，WN18 的内容是从 WordNet 中抽取出的。我们在表2.1中详细罗列了 FB15K 和 WN18 的整体数据细节。

① https://github.com/thunlp/KB2E

② https://everest.hds.utc.fr/doku.php?id=en:transe

表 2.1　FB15K 和 WN18 的数据细节

Dataset	Relation	Entity	Train	Valid	Test
FB15K	1,345	14,951	483,142	50,000	59,071
WN18	18	40,943	141,442	5,000	5,000

2.3.2　实验与模型参数设置

为了与之前的模型以及对应论文中公布的实验结果能够合理地进行对比，我们在实验参数的设定上遵循了之前知识图谱表示学习模型通用的办法。经验上，我们选择了边界值 $\gamma = 4$ 来训练 WN18 下的各个模型，而对于 FB15K 我们则用 $\gamma = 1$ 作为训练的边界值。在之前一些工作的实验中，TransE、TransH 以及 TransR 在 50 维上表现明显，TransD 在 100 维度上表现明显，所以我们将 TransE、TransH 以及 TransR 的嵌入向量维度设置为 50，而将 TransD 的嵌入向量维度设置为 100。无论是 FB15K 还是 WN18 我们均训练所有的模型 1000 轮以控制拟合的程度相对一致，且一轮训练会将数据划分为 100 批来进行处理。

2.3.3　实验评估方式

在 TransE[7] 被提出时，链接预测就作为重要的评测方式来度量模型的嵌入表达能力。其评测方式为：给定一个头实体和关系的组合 $(h, r, ?)$ 来预测对应的尾实体，或者给定一个尾实体和关系的组合 $(?, r, t)$ 来预测对应的头实体，预测的依据则是模型根据知识图谱结构学到的嵌入表示。在具体实施细节上，对于任意给定的测试三元组 (h, r, t)，我们枚举所有的实体来替换头实体 h 以及尾实体 t，并使用式2-1计算能量得分，之后按照升序排列。根据我们归纳出的模型数学性质，(h, r, t) 如果成立的话，其得分应当比替换后的三元组得分要低，即 h, t 的隐关系向量 $\mathbf{r}_{ht}$ 和 r 的向量在空间上是最接近的。这里我们仍然举个例子来说明，如补全三元组（*?*，著作，威尼斯商人）时，意味着我们在回答 ‘‘《威尼斯商人》是谁的著作？’’ 这个问题，这时我们要把所有的实体都尝试一下并给出对应的得分，如果模型合理的话，莎士比亚被带入时会得到最优的评分。

和之前提到的相关工作一样，我们使用正确实体能量得分排在前十的比例来衡量预测质量，我们将这个结果称为 10 预测命中率（Hits@10）。因为我们需要的是模型整体的预测效能，并且从实践中来看 1 命中很难做到，也不是很好对模型质量进行细粒度区分，所以过去的模型和我们都选择了 10 预测命中率来进行实验。此外，我们也报告了正确实体的平均排序名次（Mean Rank)，这个指标结果一定程度上直观体现了整体向量的学习质量。

我们之前也提到过，一个通过替换实体得到的新三元组很有可能是图谱中存

在的正例三元组，该三元组其实不应当被算作是错误而存在。如需补全三元组（美国, 总统, ?）时，意味着我们在回答 ‘‘美国总统是谁? ’’。如果三元组本身是（美国，总统，奥巴马），但将奥巴马替换成华盛顿后有一个更好的评分，这并不能看作是一个失败的预测，甚至是模型低能的体现，毕竟（美国，总统，华盛顿）也是成立的。所以在替换完实体之后，在排名之前，我们将这些正确的三元组过滤掉，然后进行之前的 Hits@10 和 Mean Rank 的指标评估，这个结果我们称之为 “Filter’’，而没有经过过滤操作的结果我们称之为 “Raw’’。

在训练过程中，当替换三元组的实体时，我们遵循 Wang[8] 提出的两种方法，即 “unif” 和 “bern’’。“bern” 采用不同概率来替换头尾实体，对于非一对一的关系，我们会对实体可能性较多的部分给予更多的关注和更大的概率来生成负例。而过去一贯等概率替换头或者尾实体的方法被命名为 “unif’’。使用不同的决策会导致不同的实验结果，对于所有的模型我们同时使用 “unif” 和 “bern” 两种决策来训练模型，并且在各项指标里选择较优的结果来代表模型的泛化能力。

2.3.4　实验结果与分析

对于涉及到的模型的实验结果，我们直接引用了对应论文里汇报的数据，并且记录为其对应模型名。在我们的框架 OpenKE 下实现的各个模型，其实验结果被命名为模型名加 “OpenKE’’ 后缀，比如 TransE(OpenKE)。由于我们会与工具包 KB2E 中实现的模型来进行效率对比，所以我们引入了 KB2E 中的 TransE，命名为 TransE(KB2E)。

所有的实验结果都罗列在表2.2之中。除此以外，我们还报告了在不同的线程设定之下，损失函数的下降曲线，不同模型在不同线程设置下的加速比，这些分别体现在图2.2和图2.3中。

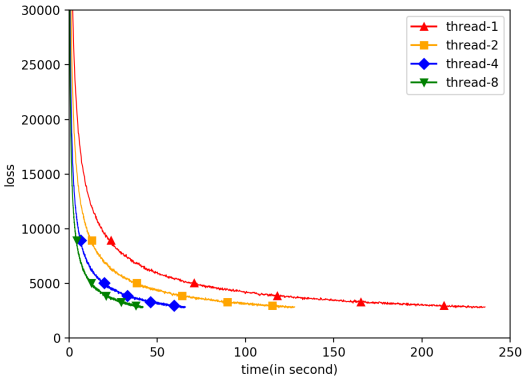

图 2.2　在不同线程设定下，TransE 在 FB15K 上的损失函数走向

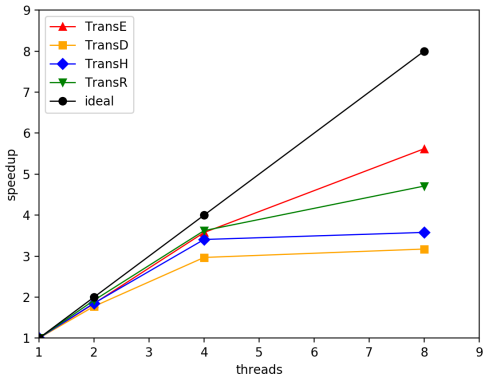

图 2.3　在不同线程设定下，OpenKE 下不同模型的加速比曲线

表 2.2　链接预测的结果

Data Sets	WN18					FB15K				
Metric	Mean Rank		Hits@10(%)		time(s)	Mean Rank		Hits@10(%)		time(s)
	Raw	Filter	Raw	Filter		Raw	Filter	Raw	Filter	
TransE(KB2E)	251	239	78.9	89.8	1674	210	82	41.9	61.3	3587
TransE(OpenKE)	273	261	71.5	83.3	12	205	69	43.8	63.5	42
TransE	263	251	75.4	89.2	-	243	125	34.9	47.1	-
TransH(OpenKE)	285	272	79.8	92.5	121	202	67	43.7	63.0	178
TransH	318	303	75.4	86.7	-	221	84	42.5	58.5	-
TransR(OpenKE)	284	271	81.0	94.6	296	196	73	48.8	69.8	1572
TransR	232	219	78.3	91.7	-	198	77	48.2	68.7	-
TransD(OpenKE)	309	297	78.5	91.9	201	236	95	49.9	75.2	231
TransD	224	212	79.6	92.2	-	194	91	53.4	77.3	-

从这些结果我们可以发现：

(1) KB2E[9] 是一个对之前知识图谱表示学习模型非常有效的实现，其评测结果比论文原始结果要高出许多。而和 KB2E 相比，OpenKE 下学到的模型在时间上要短的多，并且效果也非常接近，这意味着 OpenKE 极大的优化了训练速率并且没有影响模型性能。从整体来看，TransE 在 OpenKE 上的实现比其在 KB2E 上的实现加速了 85 倍。这么大的加速比一方面是因为 KB2E 是一个底层单线程的框架，而我们的 OpenKE 却是基于数据并行的并行框架。但是在 8 线程的处理器上，即使是理想情况，即不计通信同步时间，多线程加速比的极限也不会高于 8。所以，我们 85 倍的加速除了有多线程的贡献外，另一个重要的因素就是我们基于位移的负例采样算法和底层算术运算合并带来的积极作用。实际上，这些非算法结构上的优化对实际耗时的减少至关重要。

(2) 通过链接预测的结果，可以发现在我们的框架 OpenKE 下训练的模型，基本取得了与原始报告数据非常接近的准确度，并且在其中部分模型上，在 OpenKE 下实现可以获得略高的精度，这些现象和我们的预期是相符合的。因为我们的数据并行机制对在 OpenKE 下实现的模型没有影响，尤其是不会改变这些模型的数学性质，直观上讲我们的框架将一批数据分给若干个线程处理，而每个线程的处理方式和单线程是一致的。与其他模型相比，TransR 需要更多的时间来学习知识图谱的嵌入表示，这主要是 TransR 需要将实体通过映射矩阵投影到关系空间中，而矩阵运算其实是一个计算瓶颈，并且在 CPU 上很难解决。为了减轻矩阵运算带来的计算瓶颈，我们还在相同底层结构的框架上给出了 GPU 版本的模型实现。因为其模型性质和 CPU 版本是一致的，所以我们只是给出 GPU 版本的链接而不在实验中进行度量，链接附在本章节引言部分处。

(3) 伴随线程数的不断增多，损失函数下降需要的时间以及各模型和单线程相比的加速比都有了变化。从损失函数的下降曲线以及实际加速比曲线可以看到，多线程框架带来的优化以及耗时的下降非常的显著。当线程数小于 4 时，加速比与线程数几乎成正比，也基本接近理想的加速情况。当使用超过 4 个线程时，由于线程调度中的通信和同步，并行能力受到很大影响，加速比的上升速度开始逐渐缓慢甚至保持不变。这其中的影响因素主要来自于我们所采用的处理器。我们的处理器虽然有 8 个线程，却只有 4 个计算核心，这意味着在使用超过 4 个线程时，每个核心将需要负载至少两个的工作线程，线程的切片和通信过频会导致效率下降，所以出现了两个图表后半段加速比停滞的情况。实际上，如果采用更适合并发的处理器，这个停滞的现象将会在更多的线程被启用时才出现，而不是仅仅超过 4 个线程就接近瓶颈，这也启发我们在当前工作的基础上继续采用分布式而非单卡的框架来进行加速。

总的来说，各项评估结果表明，我们的框架成功解决了之前模型实现存在的巨大耗时问题，从而使得这些已经被提出的算法能够真正地对大规模的知识图谱进行表示学习。我们整合了这些模型底层的共通之处，使得新算法可以在不考虑底层繁琐细节的情况下也能得到高效实现。事实上，基于我们 OpenKE 的 TransE 只需耗时 18 个小时就可以训练整个 Wikidata 达 10000 轮左右，并可以获得一个稳定的嵌入表示。我们在引言部分介绍过，wikidata 是一个拥有超过 2500 万实体的巨大知识图谱。这些预先学习好的嵌入表示我们将其公开在网络 [①] 上供直接使用。

2.4　本章小结

对于真正的大规模知识图谱表示学习问题，我们提出了一个有效的训练框架 OpenKE 以便在现有模型基础上进行改进，从而能够解决我们的需求。与此同时，我们也基于该框架提供了已训练完成的大规模知识图谱嵌入向量，使得部分应用可以直接使用而无需再去耗时训练向量。我们的框架采用了基于数据并行机制的并行学习方式，从而能够取得数倍的速度提升。除此以外，我们还提出了基于位移的负例采样算法以及部分基础算术运算合并来进一步加速训练。在实验部分，链接预测上的实验结果表明，在通用的评测数据上我们的底层设计可以帮助现有模型有效地提高效率。而各个模型也在不剧烈影响精度的情况下显著缩短了训练的时间。目前在我们 OpenKE 架构下，部分模型得到了高效复现并且已经能够在真实的大规模知识图谱上进行训练。在未来，我们还将探索并尝试在 OpenKE 的框架下实现更多的知识图表示学习模型。此外，在现有基于多线程的并行学习模式

① http://openke.thunlp.org/

之外，我们还将尝试构建分布式架构来进一步解决规模和时耗问题。在我们的工作中，我们也基于 TensorFlow 来为 OpenKE 实现了一个相对简单的 GPU 版本，在我们未来的工作中，使用高效的 GPU 底层框架也是一个十分有意义的方向。

第 3 章　基于并行的知识图谱与文本模型联合学习框架

3.1　算法框架

在这一章节的内容里，论文主要介绍我们提出的基于并行的知识图谱与文本模型联合学习框架。内容包括以下几点：（1）联合框架下知识图谱与文本表示的统一形式；（2）知识图谱表示学习模型；（2）文本关系表示学习模型；（3）基于知识的跨句注意力机制；（4）初始化及实现细节，整体的框架结构也可以在图3.1中看到。从框架结构图中可以发现，在整个框架中，我们进行了大量的嵌入表示融合工作。这些融合工作包含了知识图谱与文本表示在模型形式上的统一，词与实体、关系与文本关系在嵌入向量上的统一等等。得益于这些统一的归纳与抽象，我们可以通过使用统一空间学习嵌入表示的方式来进行联合学习。联合学习支持模型间的参数共享，从而使得原本分离的模型在合并后可以相互影响并一定程度的促进各自效果提升。在介绍具体细节之前，我们仍然先引入一些符号体系和重要概念。

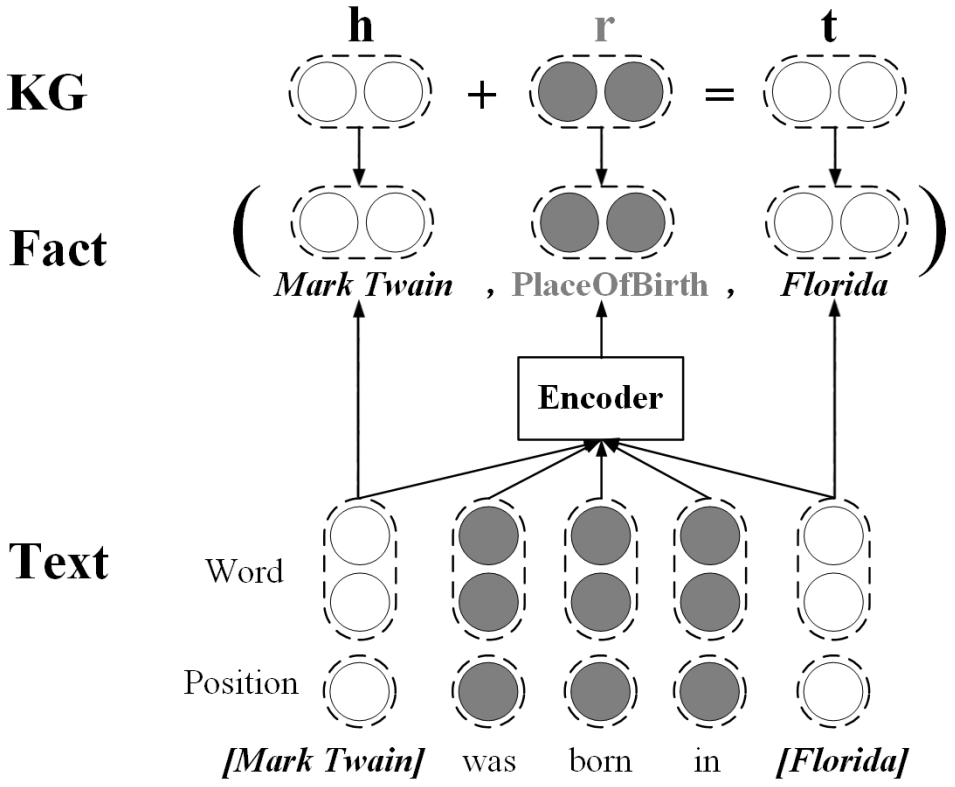

图 3.1　基于并行的知识图谱与文本模型联合学习框架

3.1.1　符号体系和重要概念

与大规模知识图谱表示学习框架一样，我们在这里同样将整个知识图谱定义为一个由实体集、关系集和事实三元组集合共同组成的大集合，即 $G = \{E, R, T\}$，这里 E、R 和 T 分别表示实体集合、关系集合和事实三元组集合。对于事实三元组集合中的任意事实 $(h, r, t) \in T$，这个三元组表明头实体 $h \in E$ 和尾实体 $t \in E$ 之间存在一个逻辑上的关联 $r \in R$。

和知识图谱 G 相对应的是文本，这里我们将文本数据定义为 D。D 是文本数据的集合，集合中的元素是大量的文本句子，这些句子构成的词汇表被定义为 V。在文本数据集合 D 中的任意一个句子 s，s 被定义为一个由若干词汇表 V 中单词构成的词语序列 $s = \{w_1, \ldots, w_n\}, w_i \in V$，$n$ 为句子的长度也是词语序列中单词的数量。在每个句子中有两个标注出的实体，且这个句子本身的文本内容可以叙述实体间的潜在语义关联 $r_s \in R$。对于文本实体和语义关系的具体标注方法将会在章节**??**处被介绍。

由于表示学习会将实体和关系都嵌入到连续空间中去并用对应的向量来表示他们的语义信息，所以对于任意的实体或者关系 $h, t \in E$ 或 $r \in R$，我们都用它们的加粗字母 $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^{k_w}$ 来表示它们的向量，这里的向量也可以称为嵌入、嵌入向量、表示、嵌入表示等。对于任意单词 $w \in V$，我们同样用加粗字母 $\mathbf{w} \in \mathbb{R}^{k_w}$ 来表示其向量。k_w 是这些单词、实体与关系的嵌入表示维度。

3.1.2　联合学习的整体模式

对于整个联合学习框架，我们希望框架可以支持下面的模型能够在一个统一的连续空间内同时学到实体、关系以及单词的嵌入表示，并通过这样的联合约束使得特征可以在知识图谱和文本间进行共享和传递。我们将所有的嵌入表示以及模型中涉及的变量都定义为模型参数，并用符号 $\theta = \{\theta_E, \theta_R, \theta_V\}$ 来表示，其中 $\theta_E, \theta_R, \theta_V$ 分别是实体、关系、单词的嵌入向量。如果将我们对框架的性能要求形式化的话，模型需要做的就是找到一组最优的参数 $\hat{\theta}$ 满足

$$\hat{\theta} = \arg\max_{\theta} P(G, D|\theta) \tag{3-1}$$

这里 $\theta_E, \theta_R, \theta_V$ 是之前定义的嵌入。$P(G, D|\theta)$ 是一个定义出的条件概率，用来刻画在给定实体、关系与单词的嵌入 θ 的情况下，嵌入对图谱与文本的拟合能力。直观点讲，模型的任务就是找到最好的嵌入表示能够最大程度的拟合给定的知识图

谱结构以及文本语义信息。而条件概率 $P(G, D|\theta)$ 又可以进一步被分解为

$$P(G, D|\theta) = P(G|\theta_E, \theta_R)P(D|\theta_V) \tag{3-2}$$

$P(G|\theta_E, \theta_R)$ 被用来从知识图谱 G 中学习结构特征，并得到实体和关系的嵌入表示。这个式子的物理意义就是希望模型能够最大限度的让知识图谱 G 中的事实概率变大，关于此部分的详细内容将会在章节3.1.3中展开。

$P(D|\theta_V)$ 被用来从文本信息 D 中学习文本特征，并得到单词与语义关系的嵌入表示。这个式子的物理意义就是希望模型能够最大限度的让 D 中句子的语义信息与其描述的语义关系相对应，关于此部分的详细内容将会在章节3.1.4中展开。

我们将知识图谱的概率定义为其包含的事实的概率，将文本的概率定义为语义信息与语义关系对应的概率，并对原概率式进行变换，得到

$$P(G|\theta_E, \theta_R) = \prod_{(h,r,t)\in G} P((h, r, t)|\theta_E, \theta_R), \tag{3-3}$$

$$P(D|\theta_V) = \prod_{s\in D} P((s, r_s)|\theta_V) \tag{3-4}$$

这里 $P((h, r, t)|\theta_E, \theta_R)$ 定义了知识图谱 G 中事实三元组在已知实体关系嵌入下的条件概率，而 $P((s, r_s)|\theta_V)$ 则定义了在已知单词嵌入的情况下，D 中句子 s 能准确描述语义关系 r_s 的条件概率。

3.1.3　知识图谱表示学习模型

对于知识图谱表示学习，我们在之前的工作中也进行了详细的叙述，其主要任务就是通过将实体和关系表示为空间中的嵌入向量从而能够抓取语义关联。在章节里我们已经将这个任务落实到对事实三元组条件概率进行优化的目标上。和 Lin[11] 一致，我们将优化条件概率 $P((h, r, t)|\theta_E, \theta_R)$ 转化为优化 $P(h|(r, t), \theta_E, \theta_R)$、$P(t|(h, r), \theta_E, \theta_R)$ 以及 $P(r|(h, t), \theta_E, \theta_R)$。

对于每一个知识图谱 G 中的实体对 (h, t)，我们定义出一个潜在关系向量 $\mathbf{r}_{ht}$ 来表达实体向量 $\mathbf{h}$ 到实体向量 $\mathbf{t}$ 之间的变换，具体形式如下：

$$\mathbf{r}_{ht} = \mathbf{t} - \mathbf{h} \tag{3-5}$$

与此同时，对于知识图谱 G 中的任意三元组 $(h, r, t) \in T$，对应存在一个显式的关

系 r 来描述 h 与 t 的关系。所以我们可以将三元组的能量函数定义为

$$\begin{aligned} f_r(h,t) &= b - \|\mathbf{r}_{ht} - \mathbf{r}\| \\ &= b - \|(\mathbf{t}-\mathbf{h}) - \mathbf{r}\| \end{aligned} \tag{3-6}$$

这里，b 是一个常数偏移量，通常在 7 左右。这个式子表明，我们期望三元组集合 T 中的任意三元组 (h,r,t) 都有 $\mathbf{h}+\mathbf{r}\approx\mathbf{t}$。

基于这个能量函数，我们以 $P(h|(r,t),\theta_E,\theta_R)$ 为例来形式化的给出 T 中三元组的条件概率：

$$P(h|(r,t),\theta_E,\theta_R) = \frac{\exp(f_r(h,t))}{\sum_{h'\in E}\exp(f_r(h',t))} \tag{3-7}$$

以此类推，我们按照同样的形式可以定义 $P(t|(h,r),\theta_E,\theta_R)$ 和 $P(r|(h,t),\theta_E,\theta_R)$。实际上，无论是理念还是实际模型，这个条件概率的表达任务和 TransE 是一致的，只是不再是基于边界值优化而是基于条件概率优化。所以，我们将这个模型命名为 Prob-TransE。

为了体现我们联合学习的模式可以适应多种知识图谱的表示学习模型，我们也引入了 TransD 来对知识图谱中三元组进行编码和嵌入，具体形式如下：

$$\begin{aligned} \mathbf{r}_{ht} &= \mathbf{t}_r - \mathbf{h}_r, \\ \mathbf{h}_r &= \mathbf{M}_{rh}\mathbf{h}, \\ \mathbf{t}_r &= \mathbf{M}_{rt}\mathbf{t}, \\ \mathbf{M}_{rh} &= \mathbf{r}_p\mathbf{h}_p^\top + \mathbf{I}^{k_r\times k_w}, \\ \mathbf{M}_{th} &= \mathbf{r}_p\mathbf{t}_p^\top + \mathbf{I}^{k_r\times k_w} \end{aligned} \tag{3-8}$$

这里 $\mathbf{r}_p\in\mathbb{R}^{k_r}$ 和 $\mathbf{h}_p,\mathbf{t}_p\in\mathbb{R}^{k_w}$ 都是用来进行映射的工作向量。出于实验上的简化，关系嵌入维度 k_r 和实体嵌入维度 k_w 在我们的框架下被默认为是一样的值，在实际操作中往往也是采用了这样的设定。类似于 Prob-TransE，我们将基于 TransD 进行条件概率优化的知识图谱表示学习模型命名为 Prob-TransD。

3.1.4　文本关系表示学习模型

给定一个包含两个实体的句子，句子中的词以及句子本身的语义信息很大程度上可以揭开实体间的关系，比如 ‘‘马克吐温出生于佛罗里达州’’ 直接表明了马

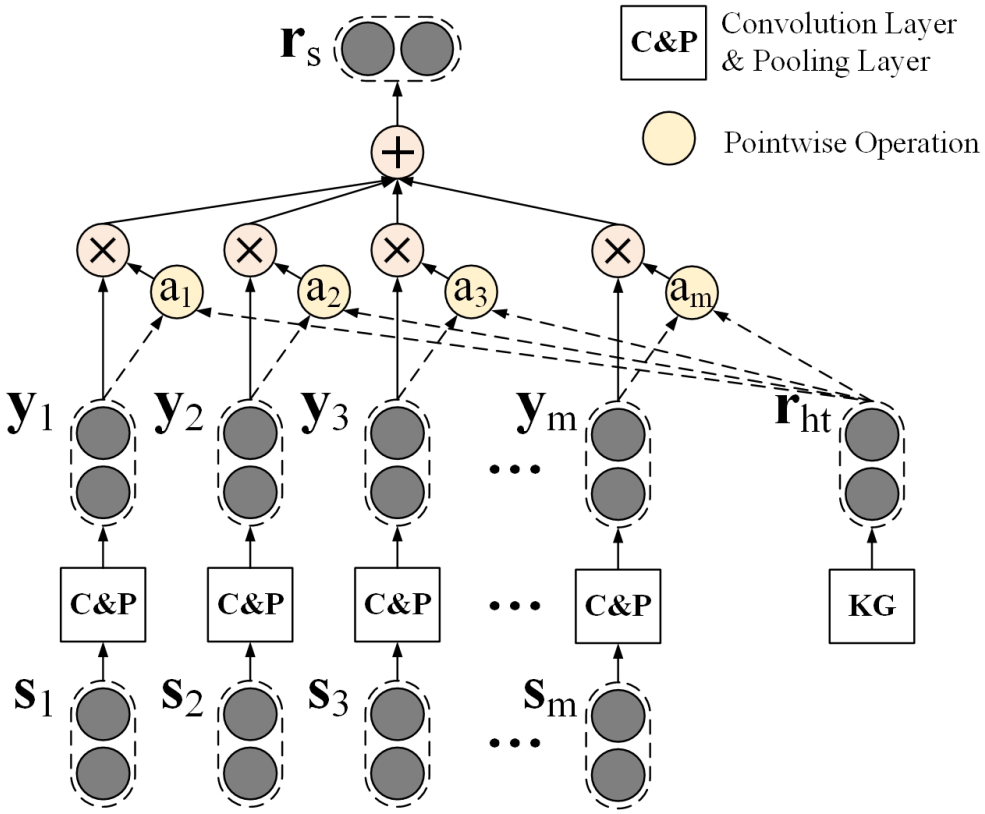

图 3.2　基于知识注意力机制的卷积神经网络模型

克吐温和佛罗里达州是人与籍贯的关系。Zeng、Toutanova 以及 Lin[12–14]，他们在各自的工作中都开始尝试使用神经网络来挖掘这样的语义信息，并且将语义信息描述的关系嵌入到低维空间中用来进行关系抽取。和他们的工作[12–14] 相似，我们也采用了卷积神经网络 CNN 对文本关系进行表示学习。

图3.2描述了我们卷积神经网络的整个结构，也是文本关系表示学习模型的整个结构。对于任意一个标注了实体对 (h,t) 的句子 s，如果实体对之间的关系为 r_s 的话，神经网络结构会以句子 s 的词语序列向量 $\mathbf{s} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ 作为输入。输入的句子向量在通过卷积神经网络中卷积与池化两层操作后，输出一个文本意义描述的关系向量 $\mathbf{y}$。由于存在多个句子标注了相同的实体，我们设置了一个基于知识图谱的注意力机制，用来在这些句子输出的基础上进行加权合并，然后得到一个全局的文本关系表示 $\mathbf{r}_s$。对于句子的语义信息能在多大程度上描述文本关系，在经过一层多项逻辑斯特回归之后，我们用以下的能量函数来刻画：

$$\mathbf{o} = \mathbf{M}\mathbf{r}_s, \tag{3-9}$$

这里 $\mathbf{M} \in \mathbb{R}^{\|R\| \times k_c}$ 是一个关系表示矩阵用来求得 $\mathbf{r}_s$ 在不同的关系上的能量评分，

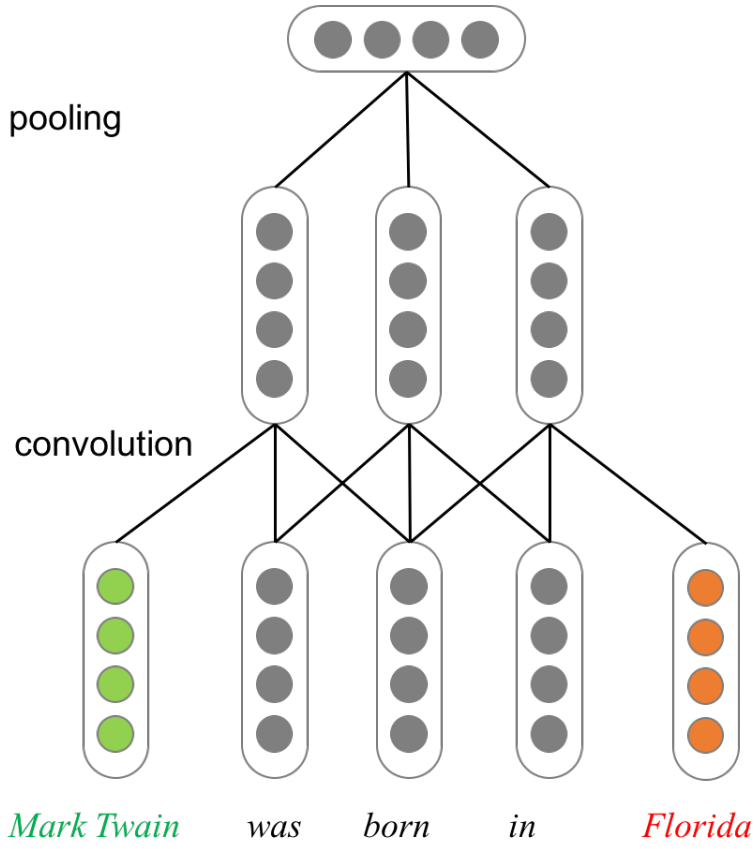

图 3.3　卷积层和池化层的结构示意图

k_c 是隐层向量的维度，最后我们可以定义一个条件概率 $P((s, r_s)|\theta_V)$:

$$P((s, r_s)|\theta_V) = \frac{\exp(\mathbf{o}_{r_s})}{\sum_{r \in R} \exp(\mathbf{o}_r)} \tag{3-10}$$

我们的文本关系表示模型就是由这几个部分构成的，包括输入层、卷积层、池化层、基于知识的注意力机制以及刚才介绍的分类层，相关细节将在下面一一展开。

3.1.4.1　输入层

We simply concatenate textual word embeddings and word position embeddings to build the input for CNN:

$$\mathbf{s} = \{[\mathbf{w}_1; \mathbf{p}_1], \ldots, [\mathbf{w}_n; \mathbf{p}_n]\}. \tag{3-11}$$

给定一个含有 n 个单词的句子 s，$s = \{w_1, \ldots, w_n\}$，输入层的功能就是将 s 中的所有单词转化成对应的输入词向量 $\mathbf{s} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$。对于给定句子 s 中的任意一

个单词 x_i，其输入向量 $\mathbf{x}_i$ 由两个实向量构成，一个是它的文本词向量 $\mathbf{w}_i$，另一个是它的位置向量 $\mathbf{p}_i$。

文本词向量可以将词的语义信息编码到低维空间中，并且通常会通过大量文本预先训练获得。大量的实验表明，以预先训练好的词向量作为网络输入可以有效提升神经网络的模型效果。在我们的工作中，词向量通过 Skip-Gram[15] 在大规模文本语料上提前训练获得。

位置向量的概念首次被提出是在 Zeng[12] 的工作中。位置向量是一个可以表明给定单词与句子标注实体之间相隔距离的特征。举个例子，“马克吐温出生于佛罗里达州’’ 中，马克吐温与佛罗里达州是标注出的实体，出离马克吐温和佛罗里达州的距离分别为 1 和 −3。我们会将这些距离映射到维度 k_p 的连续空间上。对于句子 s 中给定的单词 w_i，它的位置向量为 $\mathbf{p}_i = [\mathbf{p}_i^h, \mathbf{p}_i^t], \mathbf{p}_i^h, \mathbf{p}_i^t \in \mathbb{R}^{k_p}$，其中 $\mathbf{p}_i^h$ 和 $\mathbf{p}_i^t$ 分别是到头实体以及尾实体之间的距离向量。

接着我们合并词向量 $\mathbf{w}_i \in \mathbb{R}^{k_w}$ 与位置向量 $\mathbf{p}_i \in \mathbb{R}^{k_p \times 2}$ 得到最终的输入向量 $\mathbf{x}_i \in \mathbb{R}^{k_i} (k_i = k_w + k_p \times 2)$，从而卷积神经网络的输入为：

$$\begin{aligned} \mathbf{s} &= \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \\ &= \{[\mathbf{w}_1; \mathbf{p}_1], \ldots, [\mathbf{w}_n; \mathbf{p}_n]\}. \end{aligned} \tag{3-12}$$

3.1.5　卷积层

卷积层将输入层的输出 $\mathbf{s}$ 作为该层的输入，通过卷积层内的操作后导出为隐层向量 $\mathbf{h}$。在卷积层中，我们在输入的序列向量 $\mathbf{s}$ 上滑动一个尺寸为 m 的窗口。在每次窗口滑动中，我们可以采样得到一个局部的组合向量 $\hat{\mathbf{x}}_i$：

$$\hat{\mathbf{x}}_i = [\mathbf{x}_{i-\frac{m-1}{2}}; \ldots; \mathbf{x}_i; \ldots; \mathbf{x}_{i+\frac{m-1}{2}}], \tag{3-13}$$

这个组合向量就是通过将输入序列向量 $\mathbf{s}$ 在窗口中以 $\mathbf{x}_i$ 为中心的 m 个向量组合而成。然后，我们将这个组合向量 $\hat{\mathbf{x}}_i$ 进行线性变换以及激活从而得到隐层函数 $\mathbf{h}_i$

$$\mathbf{h}_i = \tanh(\mathbf{W}\hat{\mathbf{x}}_i + \mathbf{b}), \tag{3-14}$$

这里，$\mathbf{W} \in \mathbb{R}^{k_c \times m k_i}$ 是卷积层的卷积核矩阵，$\mathbf{b} \in \mathbb{R}^{k_c}$ 是卷积层的偏移向量，k_c 是隐层向量 $\mathbf{h}_i$ 的维度。

3.1.6　池化层

在池化层中，一个最大池化操作在隐层向量 $\mathbf{h}_1, \ldots, \mathbf{h}_n$ 上被实施来获得最后的输出向量 $\mathbf{y} \in \mathbb{R}^{k_c}$，具体的过程如下：

$$\mathbf{y}_j = \max\{\mathbf{h}_{1,j}, \ldots, \mathbf{h}_{n,j}\}, \tag{3-15}$$

这里，$\mathbf{y}_j$ 是输出向量 $\mathbf{y}$ 的第 j 维的值，$\mathbf{h}_{i,j}$ 是第 i 个隐向量 $\mathbf{h}_i$ 的第 j 维的值。池化层的主要作用在于对全局的特征进行汇总。在卷积层中，卷积实际上是对局部的语义的特征提取，但一个句子的语义实际上不是局部的而是一个全局的，池化的作用正是在每个局部采样中的每个维度上选取一个信号最为强烈的采样，从而最后能够汇总得到全局的语义特征，这是至关重要的一个步骤。

3.1.7　基于知识的跨句注意力机制

对于知识图谱中任意一个三元组 $(h, r, t) \in T$，实际上可能存在若干个句子会包含这个三元组中的实体对 (h, t)，并且这些句子的语义预示着实体对具有关系 r。经过输入层、卷积层、池化层，这些句子已经有了输出向量 $\mathbf{y}_1, \ldots, \mathbf{y}_m$，$m$ 为包含这些实体对的句子数量。对于这些句子，我们认为其中的某些句子对最后的文本关系表示学习会更具有贡献性，所以我们需要一个机制来选取这些重要的句子且规避噪音。

在这里我们通过引入知识图谱中的潜在关系向量 $\mathbf{r}_{ht} \in \mathbb{R}^{k_w}$ 来神经网络进行一个基于知识的注意力机制，从而强化重要句子的影响，具体形式如下：

$$\begin{aligned} \mathbf{e}_j &= \tanh(\mathbf{W}_s \mathbf{y}_j + \mathbf{b}_s), \\ a_j &= \frac{\exp(\mathbf{r}_{ht} \cdot \mathbf{e}_j)}{\sum_{k=1}^{m} \exp(\mathbf{r}_{ht} \cdot \mathbf{e}_k)}, \\ \mathbf{r}_s &= \sum_{j=1}^{m} a_j \mathbf{y}_j, \end{aligned} \tag{3-16}$$

这里，$\mathbf{W}_s \in \mathbb{R}^{k_w \times k_c}$ 是一个权重矩阵用来将前几层的输出向量转换到图谱空间中，$\mathbf{b}_s \in \mathbb{R}^{k_w}$ 则是线性变换的偏移向量。a_j 是第 j 个句子输出向量 $\mathbf{y}_j$ 在整个注意力机制结算后得到的权重评价。我们通过每个句子输出向量的权重来对这些向量进行加权求和从而得到一个全局的文本关系嵌入表示 $\mathbf{r}_s$。有了全局的嵌入向量后，我们可以将向量 $\mathbf{r}_s$ 带入式3-9以及式3-10中进行分类。

3.1.8　初始化及实现细节

在这一部分，我们主要介绍我们的模型在具体训练以及优化过程中的一些细节操作。对于我们以条件概率式3-2为形式的任务目标，我们定义了一个对数似然函数来作为我们的优化目标，

$$\begin{aligned}\mathcal{L}_\theta(G, D) &= \log P(G, D|\theta) + \lambda\|\theta\|_2 \\ &= \log P(G|\theta_E, \theta_R) + \log P(D|\theta_V) \\ &+ \lambda\|\theta\|_2\end{aligned} \tag{3-17}$$

这里，λ 是一个超参，$\|\theta\|_2$ 是一个 L_2 距离的约束条件。我们的所有模型，包括 Prob-TransE 和 Prob-TransD 以及 CNN 都是通过随机梯度下降（stochastic gradient descent，SGD）算法来进行优化。值得注意的是，我们的损失函数的梯度会被传递到输入层的词向量上，因为这样才能将知识图谱的特征嵌入到词向量中。我们实现了一个多线程同步训练的模式来进行图谱和文本两方面的模型同时训练。每个线程控制一个模型以及所使用的训练数据。多个模型在内存上共用词、实体以及关系的嵌入表示，且不加锁，梯度直接反馈到向量上。

插图索引

表格索引

公式索引

参考文献

[1] Miller G A. Wordnet: a lexical database for english[J]. Communications of the ACM, 1995.

[2] Hoffart J, Suchanek F M, Berberich K, et al. Yago2: A spatially and temporally enhanced knowledge base from wikipedia[J]. Proceedings of Artificial Intelligence, 2013.

[3] Auer S, Bizer C, Kobilarov G, et al. Dbpedia: A nucleus for a web of open data[M]//Proceedings of the semantic web. [S.l.: s.n.], 2007

[4] Bollacker K, Evans C, Paritosh P, et al. Freebase: a collaboratively created graph database for structuring human knowledge[C]//Proceedings of KDD. [S.l.: s.n.], 2008.

[5] Vrandečić D, Krötzsch M. Wikidata: a free collaborative knowledgebase[J]. Communications of the ACM, 2014.

[6] Dong X, Gabrilovich E, Heitz G, et al. Knowledge vault: A web-scale approach to probabilistic knowledge fusion[C]//Proceedings of SIGKDD. [S.l.: s.n.], 2014.

[7] Bordes A, Usunier N, Garcia-Duran A, et al. Translating embeddings for modeling multi-relational data[C]//Proceedings of NIPS. [S.l.: s.n.], 2013.

[8] Wang Z, Zhang J, Feng J, et al. Knowledge graph embedding by translating on hyperplanes [C]//Proceedings of AAAI. [S.l.: s.n.], 2014.

[9] Lin Y, Liu Z, Sun M, et al. Learning entity and relation embeddings for knowledge graph completion[C]//Proceedings of AAAI. [S.l.: s.n.], 2015.

[10] Ji G, He S, Xu L, et al. Knowledge graph embedding via dynamic mapping matrix[C]// Proceedings of ACL. [S.l.: s.n.], 2015.

[11] Lin Y, Liu Z, Sun M. Knowledge representation learning with entities, attributes and relations [J]. Proceedings of IJCAI, 2016.

[12] Zeng D, Liu K, Lai S, et al. Relation classification via convolutional deep neural network[C]// Proceedings of COLING. [S.l.: s.n.], 2014.

[13] Toutanova K, Chen D, Pantel P, et al. Representing text for joint embedding of text and knowledge bases.[C]//Proceedings of EMNLP. [S.l.: s.n.], 2015.

[14] Lin Y, Shen S, Liu Z, et al. Neural relation extraction with selective attention over instances [C]//Proceedings of ACL. [S.l.: s.n.], 2016.

[15] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. Proceedings of ICLR, 2013.

致 谢

衷心感谢我的指导老师刘知远助理教授对我的精心指导。在本科的四年里，他为我创造了良好的学术氛围、优越的科研环境以及优质的学术交流机会。他循循善诱，引导我在自然语言处理领域不断探索，言传身教使我深受启发且受益良多。

感谢清华大学自然语言处理组林衍凯、谢若冰学长为代表的诸多前辈在科研工作中给予我的交流、探讨和帮助，尤其是在进组之初的答疑解惑，确定研究方向之后的论证建议，投稿论文时的严格把关，让我在学术道路上不再孤单。

感谢清华大学计算机科学与技术系的诸多老师们多年来的悉心教导，在专业课程上给予我充分锻炼，在课程外给予我充分拓展，使我能够对专业有着全局性的视野和把握，也让我在这四年中个人能力得到了巨大锻炼。

感谢大学期间我的朋友们给我的关心和陪伴，无论是班级同学、思源十三期的同学还是实验室同届的同学，和大家在一起的日子我十分的快乐和满足。无论是远赴海外的吴佳炜、徐磊、曾文远同学，还是和我继续留在实验室的郭志芃同学，祝大家都将有一个美好的人生，有缘再聚。

最后我要感谢我的家人。在我走来的一路上，你们一直是我的坚固堡垒和强力后盾，在后方给予我无限的支持，让我能够无所畏惧的向前冲刺。在参与信息竞赛的道路上，尤其是低谷之时，你们始终给我鼓励与支持，让我有无限勇气来坚持到底。对于即将参与高考的表弟和堂妹，我也衷心祝愿你们取得优异成绩，努力把握自己的人生。

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____ 日　期：_____

综合论文训练记录表

学生姓名		学号		班级	
论文题目					
主要内容以及进度安排	指导教师签字： 考核组组长签字： 年　　月　　日				
中期考核意见	考核组组长签字： 年　　月　　日				

指导教师评语	指导教师签字：_____ 年　　月　　日
评阅教师评语	评阅教师签字：_____ 年　　月　　日
答辩小组评语	答辩小组组长签字：_____ 年　　月　　日

总成绩：_____

教学负责人签字：_____

年　　月　　日