DB4Trans:数据库内置知识图谱嵌入模型训练引擎

柳鹏凯 王 鑫 刘宝珠 蔡顺汀 李思卓

(天津大学智能与计算学部 天津 300350)

摘 要 知识图谱嵌入技术将知识图谱中的实体和关系嵌入到连续的向量空间中,在简化图谱操作的同时保留知识图谱的固有结构,并有助于完成诸如图谱补全、链接预测等下游任务.随着基于知识的人工智能的日益普及和应用,知识图谱的数据规模正在急剧增加.然而,大部分的知识图谱嵌入工作主要关注模型训练的结果,忽略了对于数据规模的可扩展性,在处理大规模知识图谱时表现出较差的性能.近年来的一些研究工作将数据库技术应用于机器学习算法的优化,同时提供了用于数据库内机器学习的各类工具.通过将知识图谱嵌入模型与数据库在数据管理上的优势进行有效的结合,能够在保证知识图谱嵌入模型训练的准确率和效率的同时,提供更好的可扩展性以支持大规模知识图谱数据的训练.基于此,本文提出一种数据库内置知识图谱嵌入模型训练引擎 DB4Trans.首先,设计了一种用于知识图谱嵌入模型训练的数据存储方案,对实体和关系进行编码并建立索引结构,以实现模型训练过程中对中间结果的快速访问和更新;其次,提出了一种数据库内置的模型训练优化算法,对数据库与内存间的数据批量交换方案进行设计以支持大规模数据的训练与存储;最后,在不同数据集上进行了测试,比较了模型训练与预测的时间、模型训练的准确率、存储时间和空间效率并验证了方法的可扩展性.实验结果表明,所提出的方法能够在不影响模型训练效率和准确率的同时,通过内存与数据库间的数据交换,支持在数据库内完成大规模知识图谱的训练过程.

关键词 数据库;知识图谱嵌入;模型训练引擎;DB4AI;TransE

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2022.01969

DB4Trans: In-Database Model Training Engine for Knowledge Graph Embedding

LIU Peng-Kai WANG Xin LIU Bao-Zhu CAI Shun-Ting LI Si-Zhuo

(*College of Intelligence and Computing, Tianjin University, Tianjin* 300350)

Abstract With the proliferation and application of artificial intelligence technologies, the application of knowledge graphs has a rapid growth. Knowledge graphs, consisting of entities and their relations, have been used to store structured knowledge information and have been successfully applied to many real-world applications. In recent years, the knowledge graph embedding technology embeds entities and relationships in the knowledge graph into a continuous vector space, which simplifies the graph manipulation while retaining the inherent structure of the knowledge graph and helps to accomplish downstream tasks such as graph completion and link prediction. With the increasing popularity of knowledge-based artificial intelligence, the data scale of knowledge graphs is increasing dramatically. However, most of the works on knowledge graph embedding focus on the results of model training, ignore the scalability for data size and show poor performance when processing large-scale knowledge graphs. Although these methods can achieve the desired

收稿日期:2021-07-12;在线发布日期:2022-05-18.本课题得到国家重点研发计划项目(2019YFE0198600)、国家自然科学基金面上项目(61972275)资助.**柳鹏凯**,硕士研究生,中国计算机学会(CCF)学生会员,主要研究方向为知识图谱数据管理.E-mail:liupengkai@tju.edu.cn.**王 鑫**(通信作者),博士,教授,博士生导师,中国计算机学会(CCF)杰出会员,主要研究领域为知识图谱数据管理、图数据库、大规模知识处理.E-mail:wangx@tju.edu.cn.**刘宝珠**,硕士研究生,中国计算机学会(CCF)学生会员,主要研究方向为知识图谱数据管理.**蔡顺汀**,本科生,主要研究方向为知识图谱数据管理.**李思卓**,硕士研究生,中国计算机学会(CCF)学生会员,主要研究方向为知识图谱数据管理.

results on small benchmark datasets, they ignore the scalability for data size and they cannot directly handle large-scale knowledge graph data in memory-constrained environments. Several research works have proposed frameworks for training knowledge graph embedding models in the distributed environment. Nevertheless, these works separate the data storage and the model training process from each other. Several recent types of research have applied database technology to the optimization of machine learning algorithms and provided various tools for in-database machine learning. By effectively combining the knowledge graph embedding models with the advantages of databases in data management, the training accuracy and efficiency of knowledge graph embedding can be guaranteed while providing better scalability to support the training of large-scale knowledge graph data. To this end, this paper proposes DB4Trans, an in-database model training engine for knowledge graph embedding, which focuses on the storage of training data and models in the database and the computation and iterative training in the memory to design the technical solution. DB4Trans innovatively utilizes the database to store the training data and iteratively train the model in memory, using the data exchange method between disk and memory to support the training of large-scale knowledge graph data, achieving the integration of data storage and computation. Firstly, we design a data storage scheme for training knowledge graph embedding models, encoding entities and relationships, and establishing an index structure to achieve fast access and update of intermediate results during model training; secondly, the workflow of model training is supported by the database, based on the designed storage and indexing scheme to improve the speed of reading and writing data in the process of model training to ensure the efficiency of model training, and propose the scheme of data batch exchange between database and memory to support the training and storage of large-scale data; finally, extensive experiments were conducted on various datasets to compare the time of model training and prediction, the accuracy of model training, the storage time and space efficiency, and to verify the scalability of the method. The experimental results show that the proposed method can support the training process of large-scale knowledge graphs within the database by exchanging data between the memory and the database without affecting the model training efficiency and accuracy.

Keywords database; knowledge graph embedding; model training engine; DB4AI; TransE

1 引 言

随着人工智能(Artificial Intelligence,AI)的快速发展,知识图谱已经在很多领域得到了广泛的应用.知识图谱作为人工智能的重要基石,通过图的形式表示现实世界中的实体和实体之间的关系.知识图谱由节点和边组成,节点可以表示实体或抽象的概念,边可以表示实体的属性或实体之间的关系.知识图谱通过将每条边表示为一个形如(头实体,关系,尾实体)的三元组,来对数据进行结构化的表示.然而,三元组形式的表示会使得在知识图谱上的操作及下游任务的执行变得复杂,同时也对计算性能造成一定的影响.为解决这一问题,可以将知识图谱中的实体和关系映射到连续的向量空间,并包含一些语义层面的信息,进而在问答任务[1]、关系抽取[2]等下游任务中可以更加方便地操作知识图谱.连续向量的表达可以蕴涵更多的语义,更容易被计算机理解和操作.这种将知识图谱中的实体和关系映射到连续向量空间的方法被称为知识图谱嵌入(Knowledge Graph Embedding).知识图谱嵌入通过机器学习的方法对模型进行学习,TransE[3]作为最具代表性的利用转移距离的知识图谱嵌入模型,它将实体和关系表示为同一空间内的向量.知识图谱嵌入的流程如图1所示.给定一个事实,其关系可以被表示为头实体和尾实体向量之间的转移向量.

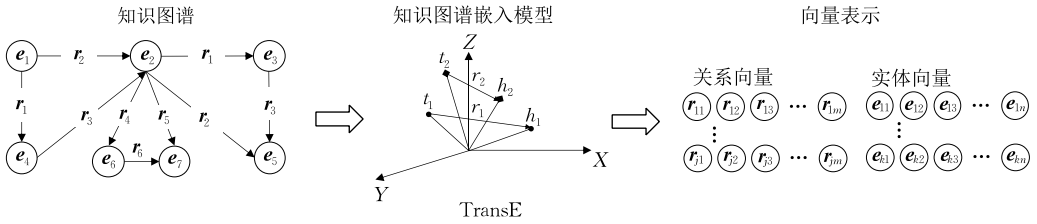


图 1　知识图谱嵌入流程

然而，现如今大多数的知识图谱嵌入研究主要关注算法的有效性，而较少关注算法对于数据规模的可扩展性. 目前，常见的知识图谱数据集已经可以达到百万顶点(10^6)，上亿条边(10^8)的规模. 知识图谱嵌入方法的训练通常需要基于监督学习，因此需要大量的数据以用于模型的训练[4]，目前大多数的知识图谱嵌入模型尽管可以在小规模的评测数据集上达到预期的效果，但是其在内存受限的环境下无法直接处理大规模的知识图谱数据. 目前的一些研究工作，基于 Spark[5] 或 PyTorch[6] 等分布式计算技术，提出了在分布式环境下，进行知识图谱嵌入模型训练的框架[7-8]. 但是，这些工作对于数据的存储和模型的训练过程是相互分离的，其通过文件 I/O 进行训练数据的访问，将数据划分到不同的工作节点后，在内存中完成模型的训练过程，无法支持“存算一体化”的模型训练，即不能同时满足数据的高效存储、查询与计算.

近年来，数据库技术被广泛应用于加速 AI 算法、优化 AI 模型的研究中. 使用数据库降低 AI 模型的应用门槛；利用视图、索引提高 AI 算法的效率；通过在数据库内设计新的算子来更好地支持机器学习相关操作等理论与技术(DB4AI)成为了研究的热点[9]. 数据库与人工智能技术的关系是双向促进的，一方面，可以应用数据库技术对大规模数据进行高效地管理与模型的训练，这是能够更加充分利用人工智能的决定因素，另一方面，人工智能技术也更多地应用于增强数据库所提供的服务质量与可靠性保障.

本文从这一点出发，基于“存算一体化”的思想，首次提出了一种数据库内置的知识图谱嵌入模型训练引擎 DB4Trans(Data Base for Translation-based knowledge graph embedding models). 主要以训练数据与模型在数据库中的存储及在内存中的计算与迭代训练两个方面为研究重点设计技术方案. DB4Trans 的总体架构如图 2 所示，首先提出训练数据及模型在数据库内的存储方案，其次，在内存中完成模型训练的计算与迭代过程，在此过程中，利用编码及索引技术降低数据的存储开销与访问开销，从而保证模型的训练效率. 最后，大规模知识图谱进行模型训练时因需要不断地进行向量计算，将产生大量的内存消耗，“存算一体化”技术能够通过不断进行数据库与内存间的数据交换来避免训练数据过大导致的内存不足问题.

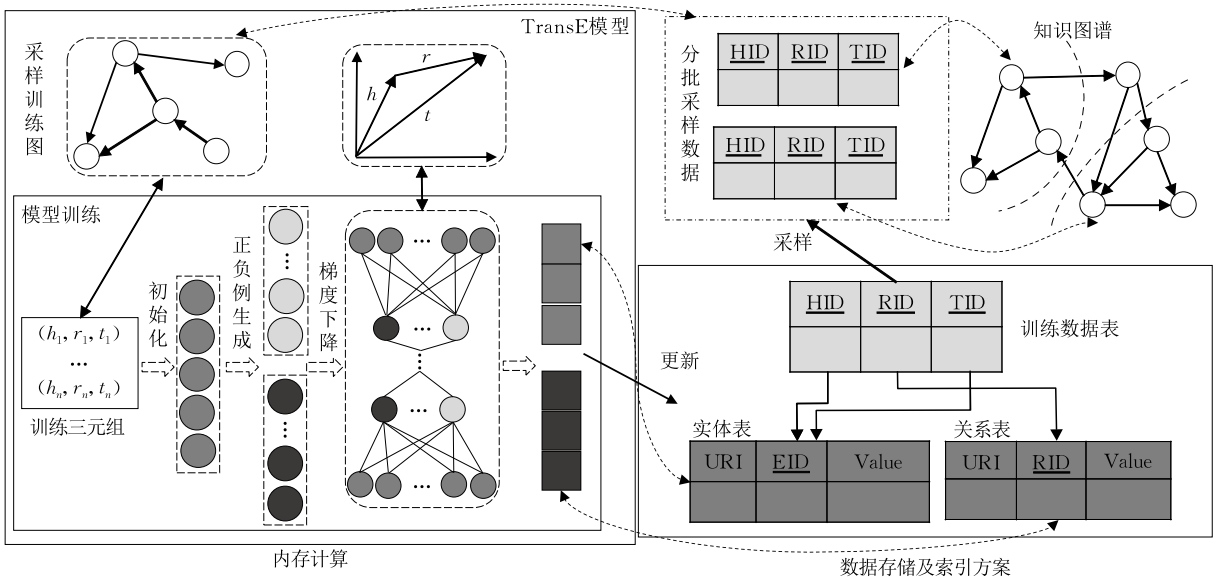


图 2　DB4Trans 总体架构图

据我们所知，DB4Trans 是首次将知识图谱嵌入模型训练与数据库技术相结合的模型训练引擎，创新性地利用数据库存储训练数据并在内存中进行模型的迭代训练，使用磁盘和内存间的数据交换方法支持大规模知识图谱数据的训练，实现数据存储和计算的融合.

本文的主要贡献点如下：

(1) 设计了一种新颖的数据库内置知识图谱嵌入模型训练引擎，基于“存算一体化”的思想，以 TransE 模型为例，设计数据存储方案和索引结构，使用关系数据库完成对知识图谱嵌入模型的训练与存储；

(2) 提出数据库内模型训练优化算法，利用数据库技术支持模型训练的工作流程，基于设计的存储及索引方案提升模型训练过程中数据的读写速度从而保证模型训练的效率，并提出数据库与内存间的数据批量交换的方案以支持大规模数据的训练与存储.

(3) 在常用于测试知识图谱嵌入的不同数据集上进行大量实验，测试所提出的方法对于模型训练的效果，与现有的知识图谱嵌入模型训练框架进行比较，验证了提出的方法能够达到与将数据全部导入内存进行训练相当的效率与准确率. 在大规模数据集上进行测试以验证所提出方法的可扩展性.

本文第 2 节介绍相关工作；第 3 节介绍预备知识；第 4 节描述数据库内置知识图谱嵌入模型训练引擎框架；第 5 节介绍在数据集上进行测试的效果并分析实验结果；第 6 节对全文进行总结.

2 相关工作

随着知识图谱的发展，各种知识图谱嵌入模型不断涌现，现有的工作基于单机或分布式环境，提出了集成各类模型的知识图谱嵌入训练框架. 同时，数据库技术作为一种非常基础而又重要的计算机技术，因其能够支持大规模数据的高效管理而广泛存在于生产环境，近年来的一些研究工作，数据库研究者对机器学习整个流程的优化做出了诸多贡献，如降低使用各类模型的复杂程度、加速机器学习算法或在数据库内提供各种机器学习功能[3]. 本节将对知识图谱模型训练的集成框架及现有的利用数据库技术来进行机器学习算法的框架与工具分别进行介绍.

2.1 知识图谱模型训练集成框架

(1) 单机模型训练框架

KB2E[10] 提供了集成各种知识图谱嵌入模型的框架，与之相似，OpenKE[11] 框架利用 GPU 加速和并行化机制来加速整个训练过程. 然而，这两种方法是在单机环境下执行的，并不适用于分布式集群，因此不能良好地支持大规模知识图谱嵌入模型的训练.

(2) 分布式模型训练框架

针对单机环境下的模型训练框架无法完成大规模知识图谱嵌入数据训练的问题，基于 Spark 和 Flink[12] 集群，SANSA[13] 提供了一种数据处理的引擎，用于在大规模知识图谱上支持多种机器学习算法，具有高容错、高可用和扩展性，能够有效地处理大规模的数据集，常用于大规模 RDF 数据集的分布式计算. 提供集成的 API 框架，将分布式内存计算技术与语义技术相结合，支持语义数据表示、查询、推理和分析的功能. 在 SANSA 中，建立的分布式机器学习层实现了最基本的基于翻译的模型，TransE. 然而，由于 SANSA 的知识图谱嵌入功能模块仍处于开发阶段，在分布式环境下训练模型的更新过程中会产生错误，进行 TransE 模型训练所获得结果的准确性较低.

DKRL[7] 提出了一种基于 Spark 的分布式表示学习框架，能够并行地训练基于翻译的知识图谱嵌入模型. DKRL 使用参数服务器(parameter server)[14] 分布式架构，一方面利用参数服务器存储模型参数，另一方面利用客户端服务器来处理训练数据，以实现并行的训练. 在 DKRL 中，多个客户端可以并行地执行训练，并与服务器进行异步通信，提高了训练效率. PDKE[8] 框架同样基于参数服务器架构，构建集成统一的训练框架，基于 PyTorch 分布式学习库实现，能够处理更大规模的知识图谱数据. 此外，国内外如阿里、腾讯等互联网企业也提出了 Plato[15]、Euler[16] 等高性能图计算框架，用于支持大规模知识图谱嵌入.

目前，所提出的集成框架的能力有限，特别是在单机环境下，现有的框架一般只能支持在小型基准数据集上进行训练，可扩展性较差. 在处理大规模知识图谱数据时，通常只能在基于分布式计算技术提出的框架下进行实现. 本文的研究重点在于单机数据库内置知识图谱嵌入模型的训练，选择单机模型训练框架作为性能评测的基准，分布式模型训练框架不在本文的考虑和比较范围之内.

2.2 数据库内置机器学习系统

为了能够降低机器学习算法的使用门槛并提高机器学习算法的性能，近年来数据库社区的一些研究工作将机器学习算法与数据库在数据管理上的优势进行有效的结合[3]，提供了用于数据库内进行机器学习的各类工具，并将数据库技术应用于机器学

习算法的优化当中.

Apache MADlib[17]是一个可扩展的开源机器学习库，为结构化和非结构化数据提供了基于 SQL 的数学、统计、图和机器学习的并行实现方法以支持数据分析与数据挖掘技术.在数据库内部抽象了很多 AI 运算符，包括数据的访问与采样，以及模型定义、训练及推理等.通过维护视图来进行模型参数的更新并支持迭代训练.

Apache SystemML[18]是一种数据库内置机器学习系统.通过用户定义的聚合函数支持矩阵运算，可以有效地在数据库内部执行矩阵运算符，并利用基于代价的编译技术，在分布式、数据并行的框架上运行机器学习算法，以生成高效的执行计划，并同时支持单节点和大规模分布式的数据处理.

BigQuery ML[19]系统将每个查询语句分解为数据库操作和 AI 操作：在数据库内执行数据库操作，在如 TensorFlow[20]、Keras[21]等的 AI 平台上执行 AI 操作，因其需要频繁地在数据库和 AI 平台之间交换数据，导致系统的性能较低.

当前，直接利用数据库技术，通过磁盘与内存间的数据交换来支持知识图谱嵌入模型训练的研究工作相对较少，本文针对特定的表示学习研究领域的知识图谱嵌入模型进行分析，不仅创新性地在关系数据库内设计并实现了支持知识图谱嵌入模型训练与预测的整体框架，同时基于“存算一体化”的思想，设计存储、索引方案及磁盘与内存间数据的批量交换算法以支持大规模知识图谱嵌入模型的训练过程.

3　预备知识

本节将详细介绍相关背景知识，包括知识图谱嵌入以及 TransE 模型的定义.表 1 给出了本文使用的主要符号及其含义.

表 1　符号列表

符号	含义	符号	含义
$G=(E,R,T)$	知识图谱G	$dbInit(T)$	初始化 T 的嵌入
$s=(h,r,t)$	三元组	$getVal(id)$	获取id对应的嵌入
$f_r(h,t)$	得分函数	$updateVal(id)$	更新id对应的嵌入
$loss$	损失函数	γ	样本间距
T	三元组集合	λ	学习率
k	向量维度	$train(\gamma,\lambda,transE)$	模型训练

定义 1.　知识图谱嵌入.给定一个知识图谱$G=(E,R,T)$，其中 E 代表实体的集合，R 代表关系的集合，$T=\{(h,r,t)\}\subset E\times R\times E$ 为三元组的集合，每一个三元组(h,r,t)是由头实体 $h\in E$，尾实体 $t\in E$ 和它们之间的关系 $r\in R$ 组成的.知识图谱嵌入模型的目标是学习向量嵌入 $\boldsymbol{h},\boldsymbol{t},\boldsymbol{r}\in\mathbb{R}^k$，$k$ 是向量的维度且 $k\ll|E|$.将实体和关系嵌入到一个低维的连续向量空间中，从而将知识图谱转化为低维向量空间的表示并与下游任务进行衔接.

一般来说，知识图谱嵌入模型根据嵌入$(\boldsymbol{h},\boldsymbol{r},\boldsymbol{t})$为每个三元组$(h,r,t)$分配一个得分.现有的大部分模型利用知识图谱中的事实来执行嵌入任务，使三元组$(h,r,t)\in G$ 的得分最大化，而使$(h,r,t)\notin G$ 的得分最小化，促使得到的向量与这些事实兼容.

TransE 模型.　知识图谱描述的事实可以视为三元组集合的形式 $D^+=\{(h,r,t)\}$.TransE 模型通过构建实体矩阵 $n\times k$ 和关系矩阵 $m\times k$（其中，n 为实体数量，m 为关系数量，k 为向量维度），将三元组实例中的关系 r 看作头实体 h 向尾实体 t 的翻译过程，即 $\boldsymbol{h}+\boldsymbol{r}\approx\boldsymbol{t}$，训练过程中不断调整嵌入向量的值，使得正例的得分函数 $f_r(h,t)=-\|h+r-t\|_{1/2}$ 最大化，负例的 $f_{r'}(h',t')$最小化，同时令损失函数 $loss=\sum_{h,r,t\in D^+}\sum_{h',r',t'\in D^-}\max(0,\gamma-f_r(h,t)+f_{r'}(h',t'))$ 最小化，其中 D^+ 和 D^- 分别表示正例三元组和负例三元组.

TransE 的工作流程如图 3 所示，为实体和关系生成初始的嵌入向量，通过对训练数据进行采样生成正例与负例，运用梯度下降来最优化目标函数，不断迭代得到最终的嵌入结果以支持下游任务.

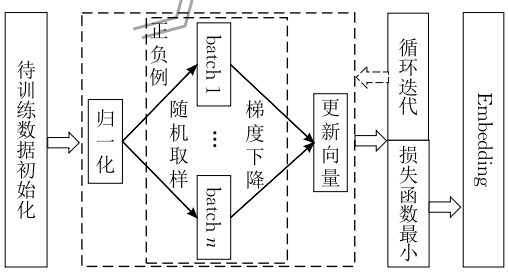


图 3　TransE 模型工作流程

DB4Trans 框架.　给定用于训练的三元组集合 T 作为输入数据.对于每一条 T 中三元组，DB4Trans 通过内存和磁盘的数据交互，从数据库获取嵌入值，在内存中更新嵌入值，并按照 TransE 的训练过程完成模型的训练，DB4Trans 框架具体训练流程如下：

(1) 通过 $dbInit(T)$ 函数进行实体和关系嵌入值的初始化，并将生成的训练数据的嵌入初始值存入数据库中，建立索引以加速数据访问；

(2) 调用 $train(\gamma,\lambda,transE)$ 函数，指定正负样

本之间的间距 γ 和学习率 λ，并根据 transE 模型定义的得分函数 $f_r(h,t)$，计算更新过程中的损失函数和梯度，分别通过 $getVal(id)$ 和 $updateVal(id)$ 获取和更新指定 id 的实体和关系的嵌入值；

(3) 根据当前运行状态，开始下一轮迭代或者结束训练过程.

4 知识图谱嵌入模型训练引擎

本节主要介绍知识图谱嵌入模型训练引擎的整体工作流程，以 TransE 模型为例，首先，给出训练数据、生成的样本数据的存储及索引方案；其次，介绍数据库内置模型训练框架的优化策略，即通过磁盘与内存的数据交换支持数据库内的大规模知识图谱的训练；最后，分析利用 DB4Trans 进行模型训练的复杂度与正确性，并讨论其对于各类基于翻译的知识图谱嵌入模型的适用性.

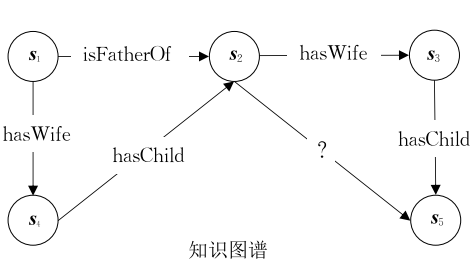
一般的知识图谱嵌入技术首先会指定实体和关系在连续向量空间中的表现形式，实体通常被表示为向量，即向量空间中确定的点，关系通常对应着向量空间中的运算，一般可以表示为向量、矩阵或张量. 然后，通过设定得分函数 $f_r(h,t)$ 来评判向量表示的合理性. 知识图谱中的事实通常具有更高的得分. 最后根据知识图谱中存在的事实，不断对表示结果进行优化，以使损失函数最小化，来学习实体和关系的嵌入.

DB4Trans 将数据库技术与知识图谱嵌入的整体流程相融合，在数据库内进行知识图谱嵌入模型的训练，(1) 将训练数据中的实体和关系进行编码后按照设计的存储方案导入并存储在数据库内；(2) 在模型训练过程中，将划分之后的训练数据分批次地导入到内存中进行计算. 在此过程中，根据实体和关系的编码值 ID，获取和更新其对应的嵌入值，并利用建立在编码之上的索引来加快数据的访问速度，加速训练的过程；(3) 在多轮迭代之后，数据库内存储的实体和关系的嵌入值，即为模型训练所得到的结果. 可以直接使用得到的训练结果进行链接预测[22]等下游任务. 下面将针对其中的技术细节进行详细的介绍.

4.1 数据存储方案

在读入知识图谱数据后，随机地将其划分为训练数据和测试数据，并将训练数据按照实体和关系分别存储到数据库的实体表(URI, EID, Value)和关系表(URI, RID, Value)中. 实体表和关系表均由 3 列组成，分别存储实体(或关系)的 URI、将实体(或关系)进行字典编码所得的 ID，以及实体(或关系)向量表示的嵌入值(Value). 其中，字典编码所得的 ID 作为表的主键，利用字典编码能够有效降低数据的存储开销并有助于索引的建立，即通过在实体和关系表中的编码 ID 列创建索引，可以满足在指定 ID 时，快速获取实体和关系的嵌入值的需求，从而提升数据访问与更新的效率. 根据训练时指定的嵌入向量的维数，在数据库中能够直接用向量类型的字段对训练结果进行存储与表示. 对于用于训练的正例数据，建立训练数据表(HID, RID, TID)来存储三元组，表中每一行对应数据中一条 (h,r,t) 三元组，存储头实体、关系与尾实体的字典编码值. 对于增加或删除的知识图谱三元组数据，训练数据表能够直接通过其编码进行表内元组的增加或删除，从而动态适应知识图谱数据的变化.

例 1. 根据上述的数据存储方案，在某个知识图谱中，包含父子(isFatherOf)、母子(hasChild)以及配偶(hasWife)三种关系，将其按照定义的数据存储方案进行相应转换，得到的存储关系如图 4 所示. 训练数据表与实体、关系表之间的箭头表示 id 字段的外键关系. 训练过程结束后，实体表和关系表的 Value 字段分别保存对应的嵌入值. 根据实体间计算得到的嵌入值，可以推断出实体之间最可能存在的关系. 如在判断实体 s_2 和 s_5 之间的关系时，他们的嵌入值之差对应关系 ifFatherOf 的嵌入值，即 s_2 和 s_5 之间可能存在父子关系.



知识图谱

HID	RID	TID
1	1	4
2	1	3
…	…	…
3	3	5

训练数据表

URI	EID	Value
s_1	1	
s_2	2	
s_3	3	
s_4	4	
s_5	5	

实体表

URI	RID	Value
hasWife	1	
isFatherOf	2	
hasChild	3	

关系表

图 4 数据存储方案示例

基于上述存储方案，算法 1 给出了 RDF 形式的知识图谱数据的导入算法，输入三元组形式的 RDF 知识图谱数据，通过哈希函数，为 RDF 图中实体和关系进行编码，实体和关系的 URI 对应唯一的 ID（第 1 行～第 7 行），将训练数据包含的三元组作为正例，对出现在三元组数据中的实体和关系进行字典编码，并将编码后的三元组存储到训练数据表中（第 8 行～第 9 行）. 在读入全部数据后，统计实体和关系的数量，与现有的基于翻译的模型的方式一致，为实体和关系的嵌入值进行初始化，即 $uniform(k)=\left(-\frac{6}{\sqrt{k}},\frac{6}{\sqrt{k}}\right)$，其中 k 为嵌入向量的维数（第 10 行～第 13 行）.

算法 1.　RDF 数据导入.

输入：RDF 图 G；嵌入维度 k

输出：实体表 E；关系表 R；训练数据表 D

1. FOR $(h,r,t)\in G$ DO
2. IF $E.get(h)=\varnothing$ THEN
3. $E.insert(h)$; //向实体表中添加实体 h
4. IF $R.get(r)=\varnothing$ THEN
5. $R.insert(r)$; //向关系表中添加关系 r
6. IF $E.get(t)=\varnothing$ THEN
7. $E.insert(t)$; //向实体表中添加实体 t
8. $hid,rid,tid\leftarrow getID(h,r,t)$;
9. $D.insert(hid,rid,tid)$;
//将对应的 id 添加到数据表 D
10. FOR $e\in E$ DO
11. $e.val\leftarrow uniform(k)$;
//对每个实体的嵌入值进行初始化
12. FOR $r\in R$ DO
13. $r.val\leftarrow uniform(k)$;
//对每个关系的嵌入值进行初始化

算法 1 的时间复杂度由两部分构成，通过哈希函数将实体和关系的 URI 映射为字典编码的 ID 值，复杂度为 $O(1)$，通过哈希的方法能够大幅度降低字典编码的时间开销. 通过遍历实体和关系数据表将嵌入值进行初始化，复杂度为 $O(|E|+|R|)$.

4.2　数据库内置模型训练优化技术

结合定义 2 介绍的 TransE 模型工作流程，基于设计的数据存储方案，给出利用数据库技术进行模型训练的具体流程，使用数据库与内存间的数据批量交换策略来优化训练的过程，以支持大规模知识图谱的训练.

现在的单机方法，如 KB2E 在进行知识图谱嵌入模型的训练时，需要将训练数据全部导入内存并进行采样生成正负例，通过不断迭代来优化目标函数以获得实体和关系的嵌入值. 然而，大规模的知识图谱数据包含的实体和关系数量较多、训练数据规模过大，这种方法因需要一次性将数据全部读入内存中，在处理大规模数据时会因受到内存大小的限制而无法直接应用.

为了在单机环境下支持大规模知识图谱的训练，通常需要对数据进行划分并存入多个文件中，进行多次文件 I/O 依次读取训练数据并将训练后的结果存入文件中. 为保证模型训练的准确率，对于大规模数据集的划分，一般采用随机采样的方式. 此外，随机采样会造成生成的文件中包含冗余的数据. 因此，通过文件 I/O 的方式会因需要额外的采样过程对文件进行划分产生额外的时间开销和存储空间开销而影响模型训练的效率.

基于现有方法存在的问题，DB4Trans 利用数据库技术优化大规模知识图谱嵌入的模型训练过程. 根据设计的数据存储方案，提出数据库和内存之间的数据批量交换算法. 通过随机采样的方式，将训练数据从数据库读取到内存中作为正例，在迭代过程中，将正例表与实体表、关系表进行连接获取嵌入值，随机替换正例的头实体或尾实体产生负例，并在内存中进行归一化、梯度下降等计算. 根据目标函数迭代计算进行模型的训练，并将实体和关系的嵌入值更新到数据库中. 从而实现对大规模知识图谱嵌入模型的训练.

算法 2 描述了从训练数据表中随机采样选取训练数据到加载到内存、获取实体和关系的嵌入值、生成负例样本进行梯度下降计算、迭代计算更新实体和关系的嵌入值的过程. 在每轮迭代的过程中，首先将当前实体和关系的嵌入值归一化（第 2 行～第 5 行），确定每次采样数据的规模（取决于内存的大小），从数据库中对训练数据表批量采样，选出用于训练的正例三元组并生成负例三元组（第 6 行～第 11 行），将采样后的数据表与实体表、关系表进行表连接操作，根据正例三元组主语、谓语、宾语的 ID 值获取其当前嵌入值（第 12 行～第 14 行）. 使用梯度下降算法进行向量的更新，计算完成后相应地修改数据表中存储的嵌入值（第 15 行～17 行）.

算法 2.　数据库内置训练优化.

输入：实体表 E；关系表 R；正例表 T；采样规模 b

输出：实体表 E；关系表 R

1. LOOP
2. FOR $e\in E$ DO
3. $e.val/\|e.val\|$;
4. FOR $r\in R$ DO
5. $r.val/\|r.val\|$;
6. $T_b\leftarrow sample(T,b)$;

//从正例表中 T 随机选择数量为 b 的三元组

7. $T_r, F_b \leftarrow \varnothing$; //$T_r$ 为训练样本对,F_b 为负例三元组
8. FOR $(sid, tid, oid) \in T_b$ DO
9. $(sid', tid, oid') \leftarrow sample(S'(sid, tid, oid))$;
//采样并生成负例三元组
10. $T_r \leftarrow T_r \bigcup \{((sid, tid, oid), (sid', tid, oid'))\}$;
11. $F_b \leftarrow F_b \bigcup \{(sid', tid, oid')\}$;
12. $T_{val} \leftarrow \pi_{E.EID, E.Value}(\rho_{\exists E.EID=(T_b.HID \text{ OR } T_b.TID)}(E))$;
//$T_b.getVal(E)$获得三元组实体当前嵌入值
13. $F_{val} \leftarrow \pi_{E.EID, E.Value}(\rho_{\exists E.EID=(F_b.HID \text{ OR } F_b.TID)}(E))$;
//$F_b.getVal(E)$获得负例三元组实体当前嵌入值
14. $R_{val} \leftarrow \pi_{R.Rid, R.Value}(\rho_{\exists R.Rid=T_b.rid}(S))$;
15. 更新 E_{val},F_{val} 与 R_{val} 中的嵌入值,使得
$$\sum_{T_r} \nabla[\gamma + \|h + r - t\|_2^2 - \|h' + r' - t'\|_2^2]_+$$
//梯度下降更新实体嵌入值
16. $E.updateVal(E_{val})$;
//将计算后的嵌入值更新到实体表中
17. $R.updateVal(R_{val})$;
//将计算后的嵌入值更新到关系表中
18. END LOOP

值得注意的是算法 2 的第 11 行~第 13 行是在 DB4Trans 中复现的 TransE 模型训练过程,TransE 算法的实现细节可以参见文献[3],在此不再赘述.另外,此过程可以通过替换目标函数来支持其他基于翻译的知识图谱嵌入模型的训练,4.3 小节将对此进行进一步的讨论.

4.3 知识图谱嵌入模型适用性

本文以 TransE 模型为例,提出数据库内的知识图谱嵌入模型训练引擎,其对于在 TransE 基础上提出的其他基于翻译的知识图谱嵌入模型同样具有适用性与通用性.现有的不同模型从不同的角度把相应的语义信息嵌入知识图谱的向量表示中,如 TransH[23]、TransR[24]等,通过设计不同的得分函数,即设计利用关系把头实体转移到尾实体的合理性的函数,融入如实体类型、关系路径等额外的信息来优化知识图谱嵌入模型,本文提出的整体框架,即数据存储、索引方案及数据库内置优化技术能够适用于该类模型.不同模型的训练结果能够以实体表和关系表的形式分别进行存储,能够直接通过实体和关系的 URI 进行查询获取训练后的嵌入值,更好地支持面向嵌入的查询.

此外,通过对知识图谱嵌入模型可解释性的研究,可以将模型的训练及预测过程进行分解并提出新的算子,如数据批量交换算子、梯度下降计算算子以及不同模型对应得分函数计算的算子.通过在数据库底层实现这些新的算子,可以获取数据库技术对不同的大规模知识图谱嵌入模型整体训练过程的支持.

定理 1. 对于一个给定的知识图谱 G,使用 DB4Trans 进行模型训练,经过 $O(|epoch|)$ 轮迭代后,将 $train(\gamma, \lambda, transE)$ 所得到的实体和关系的嵌入结果进行链接预测,所得到结果的准确率 $\tilde{u} \approx u$,其中 u 为使用原训练模型进行链接预测所得到嵌入结果的准确率.

证明. (1)在执行算法 1 进行训练数据的导入时,$dbInit(T)$ 对实体和关系进行编码并按照与原模型相同的方法进行初始化,生成嵌入的初始值;(2) DB4Trans.$sample(\cdot) \Leftrightarrow$ TransE.$sample(\cdot)$,即 DB4Trans 与 TransE 的采样过程等价.知识图谱中的三元组通过 $sample(\cdot)$ 生成正例,在训练过程中,每轮迭代对数据集中的训练三元组共进行 n 次抽样.每一次抽样中样本数的数量为 $size$,即 $size$ 为每次抽样中从整个知识图谱中产生的正例三元组的个数,并根据随机替换头实体和尾实体来相应生成 $size$ 个负例三元组;(3) DB4Trans 通过不断进行迭代并利用梯度下降法进行嵌入值的更新,令正样本的 $f_r(h,t)$ 逐渐增大,负样本的 $f_r(h,t)$ 逐渐减小,使得损失函数 $loss$ 最小化,这一过程与 TransE 相同.因此,通过 DB4Trans 进行模型训练所得结果的准确率 $\tilde{u}$ 应与使用原训练模型进行预测所得结果的准确率 u 相近. 证毕.

从模型训练过程可以看出,在 DB4Trans 中,将正例三元组提前存入数据库中,在每轮迭代的抽样中,从数据库的正例三元组表中随机读取规模为 $size$ 的正例,并通过与原模型相同的方法生成负例三元组进行模型训练.通过与原模型一致的梯度下降方法来不断根据模型定义的得分函数更新实体和关系的嵌入值.DB4Trans 没有改变原模型的训练流程.

定理 2. 通过 DB4Trans 进行模型训练的复杂度为 $O(|n| \cdot |batch| \cdot |size|)$,其中 n 是总的迭代次数,$|batch|$ 是指在每轮迭代中,对三元组进行采样的次数,$|size|$ 为每次取样的三元组的个数.

证明. DB4Trans 在 n 轮迭代后,得到训练数据中的实体和关系的嵌入值,训练过程的时间复杂度由三部分组成:(1)进行模型训练过程,共进行 n 轮迭代;(2)每轮迭代中,对训练数据进行 $batch$ 次采样;(3)每次采样所生成的正负例三元组数量为 $size$,根据这些三元组中的每一个实体和关系的 id 值,在建立的索引上进行搜索获取其嵌入值并进行梯度下降的时间复杂度为 $O(1)$.因此,DB4Trans 算法的时间复杂度为 $O(|n| \cdot |batch| \cdot |size|)$. 证毕.

根据本节介绍的 DB4Trans 中数据存储方案和数据库内置模型训练优化技术，DB4Trans 能够有效地完成数据库内 TransE 模型的训练过程并支持大规模知识图谱数据的训练. 定理 1 与定理 2 的证明也进一步说明了 DB4Trans 在模型的训练上，并没有对采样结果、得分函数以及梯度下降等过程进行计算方法的更改，所以在模型的训练效果和时间复杂度上并没有造成影响，保证了将其他基于翻译的知识图谱嵌入模型移植到 DB4Trans 时的适用性与兼容性.

5　实验与分析

本节在不同的数据集上进行实验，与 TransE 模型在单机环境下进行训练的效率和使用训练结果进行链接预测得到的准确率进行比较，验证提出的模型训练引擎的高效性及模型训练结果的有效性.

5.1　实验环境和数据集

本文使用的实验平台为单节点服务器，节点配置主频为 2.10GHz 的 Intel Xeon Silver 16 核处理器，其内存总大小为 512GB，固态硬盘大小为 7.2TB，使用的操作系统为 Linux 64-bit CentOS 7.6.

DB4Trans 以开源图数据库 AgensGraph 为后端，其在关系数据库的基础上支持了属性图的存储与查询，底层数据仍以表的形式进行存储，因此适用于本文提出的数据存储方案. KB2E 作为单机环境下的知识图谱嵌入模型集成训练框架，支持各类基于翻译的知识图谱嵌入模型的训练（如 TransE、TransR 等），我们将其作为实验部分的评测基线.

本文使用的数据集包括两个常用于进行链接预测评估任务的数据集 WordNet[25] 和 Freebase[26] 的子集. (1) WordNet 是一个描述英文词汇及它们之间关联特点的数据集，将英文名词、动词、形容词和副词与同义词联系起来，这些同义词通过语义关系相互联系，从而确定单词的定义；(2) Freebase 是一个由维基百科中提取的大量事实组成的著名知识库，包含非常多话题和类型的知识，主要包括人类、媒体、地理位置等信息，并支持信息之间的丰富关联并赋能这种关联的使用.

在本文中，我们使用 WordNet 和 Freebase 两个数据集的五个不同子数据集（D1～D5）进行实验，来测试 DB4Trans 的性能：(1) WN18 是 WordNet1995 的子集，对词汇的语义信息进行描述，该子集中关系的主要模式是对称关系、非对称关系和反转关系；(2) WN18RR 是 WN18 的子集，其中更多的保留了原数据集中的对称关系、非对称关系和组合关系，而去除了反转关系；(3) FB15k 是一个包含大规模常识性知识的知识图谱，该图谱中关系的类型主要是对称关系、非对称关系和反转关系；(4) FB15k237 是 FB15k 的子集，该数据集保留的关系主要是对称、非对称和组合关系，同样去除了反转关系；(5) 通过在 Freebase 更大规模的子集 Freebase-86m 上进行模型训练，来验证 DB4Trans 的可扩展性. 表 2 给出了实验所用数据集的具体统计信息，记录了实体、关系的数量以及训练集和测试集所包含三元组的数量.

表 2　实验数据集

数据集	实体	关系	训练集	测试集
WN18(D1)	40943	18	141142	5000
WN18RR(D2)	40943	11	86835	3134
FB15K(D3)	14951	1345	483142	59071
FB15K237(D4)	14541	237	272115	20466
FB86M(D5)	86054151	14824	304727650	16929308

5.2　实验结果

本节将从 3 个方面对本文提出的方法的性能进行分析与比较. 为了验证 DB4Trans 不会对模型训练的准确率和效率造成影响，将 DB4Trans 与 KB2E 在小规模的数据集（D1～D4）上进行模型训练，并比较 (1) 模型训练与预测的效率；(2) 模型训练结果进行链接预测的准确率. 为了验证 DB4Trans 对于处理大规模数据集的适用性，并测试其对于数据采样与训练的有效性与高效性，分析 (3) 在不同规模的数据集（D1～D5）下，DB4Trans 对数据进行存储的空间开销与数据导入、分批采样训练的时间开销.

KB2E 提供了包含 TransE 模型在内的各种基于翻译的知识图谱嵌入模型的实现，对于数据集 D1～D4，将数据全部读入内存后进行训练. 实验部分将其作为评价的基准. DB4Trans 按照提出的数据存储方案与数据库与内存间的数据批量交换算法，在数据库中进行数据的存储并在内存中进行 TransE 模型的训练. 下面将对实验方案进行详细介绍并对所得到的实验结果进行分析.

5.2.1　模型训练与预测性能

为了验证 DB4Trans 进行模型训练和预测的效率，在 D1～D4 四个常用的小规模数据集上测试，记录 KB2E 和 DB4Trans 进行 TransE 模型训练和预测的时间.

模型的训练时间主要由进行迭代的轮数和训练数据的规模所决定，随着迭代次数和数据规模的增加，训练的时间逐渐增大. 图 5(a) 为样本间距 γ 设

为 1、学习率 λ 设为 0.001、间距向量维数设为 100 的条件下，进行 1000 次迭代后两种方法所需的模型训练时间.

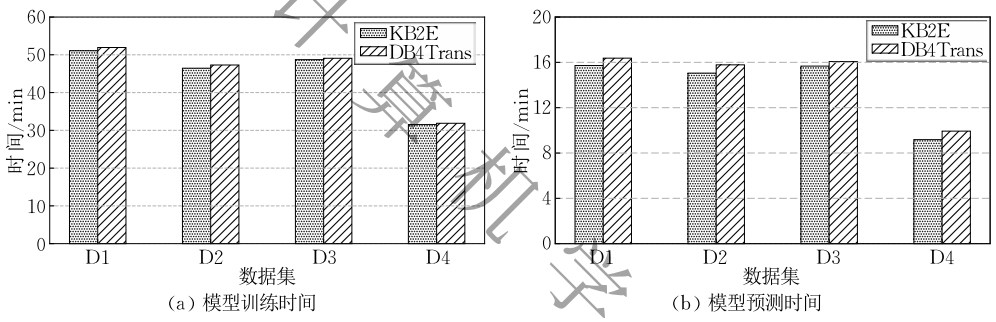
通过比较可以看出，在对数据进行训练时，两种方法所需时间比较接近，即 DB4Trans 通过数据库进行模型训练的时间可以与原 KB2E 达到相近的效率. DB4Trans 所需时间略长于 KB2E，这是因为对于小规模数据的处理时，两种方法均将数据全部导入内存后进行训练，不同之处在于 KB2E 直接将数据从文件读入，DB4Trans 则需要连接数据库并读取数据. 因此，DB4Trans 所需时间略高于 KB2E.

知识图谱嵌入模型在获得了实体和关系的向量表示之后，通常使用链接预测来评估其有效性，即通过预测知识图谱中丢失的实体来评估实体和关系嵌入的质量. 因此，需要对训练好的模型进行预测的效率进行评价，从而验证利用数据库技术来进行模型训练，从而进行预测任务的意义.

为了对模型的预测效率进行评测，使用存储在数据库中的训练得到的模型进行链接预测. 利用提出的存储方案获取实体和关系对应的嵌入向量进行计算. 如图 5(b)所示，实验结果表明模型的预测效率与模型训练效率的结果类似，DB4Trans 能够与 KB2E 进行链接预测的效率达到相同的量级，所需时间略长与 KB2E.

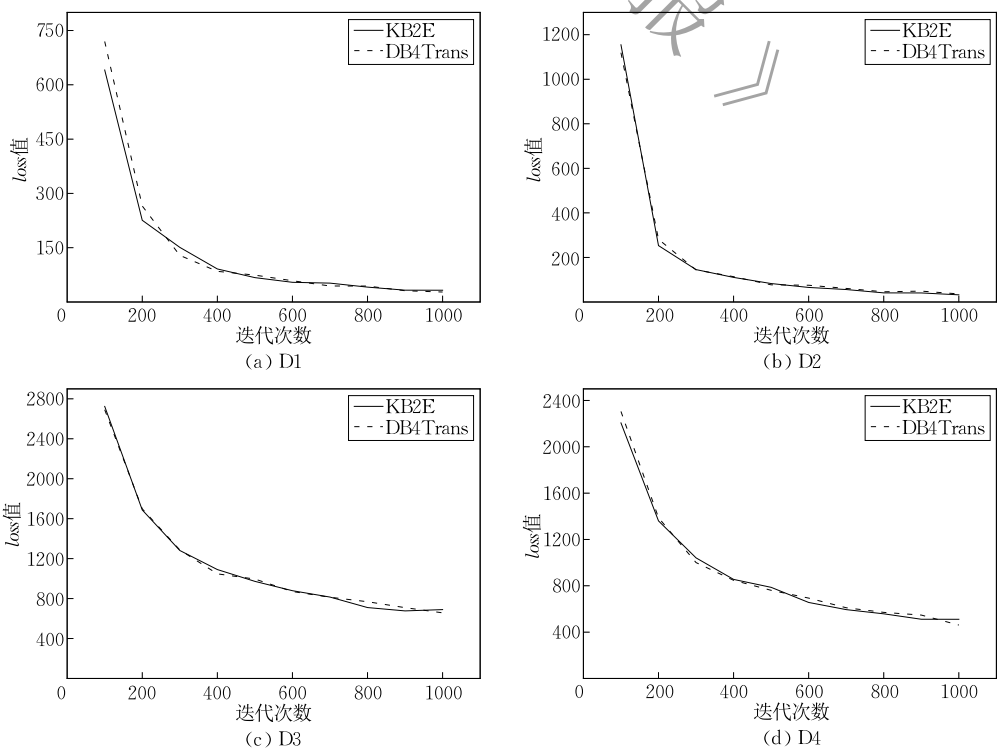
小规模数据集上模型训练与预测时间开销的测试证明了 DB4Trans 能够在不影响效率的前提下支持数据库内置的模型训练与预测. 同时，训练结束得到的模型，即实体和关系的嵌入值，存储在数据库的关系表中，可以直接通过实体和关系的 URI 对数据表进行查询获得其训练后的嵌入值，便于用户的直接访问，为更好地提高模型的复用性，降低模型的使用难度提供了支持.

在迭代的过程中，记录 *loss* 值随迭代次数变化的变化情况. 图 6(a)～图 6(d)展示了 KB2E 和



(a) 模型训练时间　　(b) 模型预测时间

图 5　不同数据集进行模型训练与预测的时间



(a) D1　(b) D2

(c) D3　(d) D4

图 6　模型训练 *loss* 值随迭代次数变化曲线

DB4Trans方法在四个小规模数据集进行训练的过程中 *loss* 值的收敛情况. 从图中可以看出，两种方法 *loss* 值的收敛曲线较为一致，能够证明 DB4Trans 与 KB2E 方法在参数相同的情况下，模型的训练程度较为接近，进一步说明了 DB4Trans 能够按照 TransE 的计算模式进行模型训练.

5.2.2　模型训练准确率

进行链接预测任务后，得到的正确三元组排名小于 N 的百分比，即 $Hits@N$，是评估实验结果有效性的常用指标.

为了验证利用DB4Trans进行 TransE模型训练所得结果的准确性，我们对上述的 D1～D4 四个数据集进行链接预测得到的结果进行分析，将 DB4Trans 和 KB2E 两种方法得到的三元组的 $Hits@10$ 的值进行比较，实验结果如图 7 所示. 可以看出，两种方法训练得到的模型在预测结果上基本相近，各数据集训练结果准确率的差异与图 6 中迭代后 *loss* 值的差异相符. 由于两种方法采样存在随机性，得到的 $Hits@10$ 的值不完全相同. 结果表明两种方法可以在模型训练的准确性上达到相当的效果，从而验证了 DB4Trans 对于模型预测的有效性，即 DB4Trans 利用数据库进行模型训练不会对预测结果的准确率造成影响.

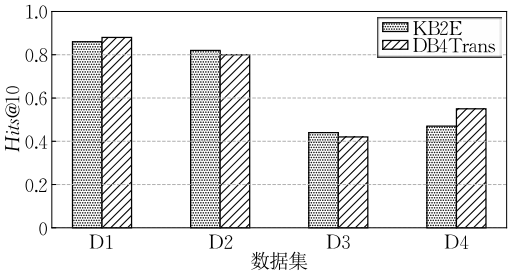


图 7　不同数据集进行模型预测结果的准确性

5.2.3　大规模数据集适用性

通过在小规模数据集下的测试，验证了 DB4Trans 不会对模型训练的效率和准确率造成影响. 在此前提下，使用更大规模的数据集来验证 DB4Trans 对于大规模数据集训练的可扩展性，以及数据库与内存间的数据批量交换算法的有效性和高效性.

现有工作提出的单机环境下的模型训练框架可扩展性较差，原因在于单机方法需要将数据全部读入内存后进行参数计算和更新在参与训练的实体和关系数量过多、训练数据规模过大时，导致占用内存过大，不能处理大规模知识图谱数据. 而 DB4Trans 能够按照本文所提出的方法，将训练数据和样本数据存储在数据库内，并通过数据库和内存的批量数据交换算法来读取数据来进行模型的训练，并最终将训练好的模型存储到数据库中，从而支持大规模知识图谱的训练与预测.

基于上述的讨论，对于传统的 KB2E 模型，无法直接对模型进行训练. 因此，需要对其原本实现的数据读取方式进行修改，将训练数据随机划分到多个文件，依次读取各文件到内存中进行分批计算得到训练结果. 而对于 DB4Trans，则不需要额外的时间开销和空间开销对文件进行划分与存储，只需要通过对存储的训练数据表进行多次采样并分批加载到内存中进行向量值的计算与更新即可完成模型的训练过程.

直观地讲，进行数据库与内存间数据批量交换的次数取决于数据规模与内存大小的比值，即每次处理数据的规模不应该超过内存的上限. 在实验中，选择 Freebase 数据集更大规模的子集，Freebase-86m(D5) 进行测试，其训练三元组数据规模为 13.6 GB. 为了模拟真实使用场景下内存大小受到限制时，本文提出的方法与传统的通过文件方法进行大规模数据训练时的时间和空间开销. 控制每次在内存中进行迭代计算的数据规模小于 2 GB.

比较随机采样划分数据读取文件与 DB4Trans 直接进行数据库与内存两种方式在时间和空间上的性能.

图 8 给出了 DB4Trans 与 KB2E 对于不同数据集进行采样的时间开销. 根据数据集的规模，时间开销由数据导入时间和数据采样时间两部分构成，对于 DB4Trans，数据导入时间指按照设计的存储方案将数据加载入数据库的时间；在处理小规模数据集(D1～D4)时，数据采样时间为数据库与内存间进行一次数据交换的时间，将训练数据全部导入内存进行训练；在处理大规模数据集(D5)时，数据采样时间为多次采样将数据集分批导入内存进行训练的时

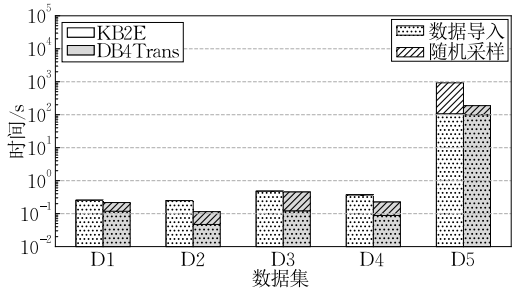


图 8　数据导入与随机采样所消耗的时间

间.对于 KB2E,数据导入时间为模型训练时进行训练数据文件读取的时间;在小规模数据集上不需要对数据进行划分,数据采样时间为 0;大规模数据集的数据采样时间为对训练数据进行划分采样的时间.

可以看出,随着训练数据规模的增加,DB4Trans 的优势表现地越来越明显.DB4Trans 通过随机采样数据与内存进行交换的效率与直接对 Freebase-86m 数据集进行随机采样划分的效率相比,能够提高一个数量级.

在存储空间方面,图 9 给出了各数据集的原始数据文件大小、按照提出的存储方案进行处理后文件存储的总大小以及导入数据库后所占用的存储空间.可以看出,通过提出的存储方案对数据进行处理后,能够显著对存储空间进行压缩.在将处理后的数据加载入数据库后,需要额外的存储空间来保存建立的索引并对数据进行组织,数据库中的存储空间开销略大于原始训练数据的规模,但仍能与原数据文件的大小保持在相同的量级(约为处理后文件大小的 3 倍).

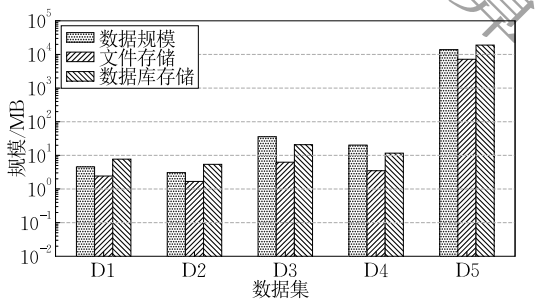


图 9 数据规模、文件与数据库存储的空间

值得一提的是,在训练大规模的数据时,文件读取方式只能够从已经划分好的文件中进行采样与训练,训练结果通常取决于划分结果的好坏,而 DB4Trans 能够直接从全部的训练数据进行采样并进行训练,从而保证了原模型的训练准确率.文件读取方式的存储开销会因采样次数过程中产生较多的冗余数据而产生额外的存储开销,所需要的存储空间则取决于划分后文件的总大小.

通过数据在内存和数据库之间的交换,数据分批次从磁盘读入内存中进行训练,经过多次迭代,将训练后的模型结果更新到磁盘中,以用于后续的预测任务.通过这样的方式,能够有效避免现有的 TransE 实现方法存在的因需要一次性将数据全部读入内存中而导致在进行训练时内存占用率会过大的问题.因此,DB4Trans 具备可扩展性并适用于大规模的数据集的训练与预测.

6 总 结

本文创新性地提出了一种数据库内置的知识图谱嵌入模型训练引擎,DB4Trans,能够在关系数据库内部进行知识图谱嵌入模型的训练,并利用基于数据库技术的优化方法,通过磁盘和内存间数据的不断交换,以支持大规模知识图谱数据的训练.以 TransE 模型为例,设计训练数据和模型的存储及编码方案,提出利用数据库内的索引和数据访问算法进行模型训练的优化方法.在不同数据集上进行实验,实验结果验证了 DB4Trans 的高效性和有效性.实验结果表明,提出的方法可以支持在数据库中进行 TransE 模型的训练.通过将知识图谱嵌入模型训练与数据库技术相结合,DB4Trans 能够在不影响原有方法训练准确率和效率的前提下,超出内存限制支持大规模知识图谱的训练,对数据规模具有良好的扩展性,同时也为今后面向知识图谱嵌入的查询工作提供了基础.

本文提出的方法适用于单机环境下利用数据库系统有效支持知识图谱嵌入模型的训练,在后续工作中,我们将进一步讨论:(1)本文所提出的数据库内置知识图谱嵌入模型训练引擎如何对基于神经网络[27]等技术的各类知识图谱嵌入模型进行融合;(2)提供数据库内进行知识图谱嵌入模型训练与预测的接口;(3)以及在分布式环境的支持下,设计分布式的数据存储方案和各类批处理算法赋能知识图谱嵌入模型训练与预测的方法并展开更深入的研究.

参 考 文 献

[1] Bordes A, Weston J, Usunier N. Open question answering with weakly supervised embedding models//Calders T, Esposito F, Hüllermeier E, et al, eds. Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Berlin, Germany: Springer, 2014: 165-180

[2] Daiber J, Jakob M, Hokamp C, et al. Improving efficiency and accuracy in multilingual entity extraction//Proceedings of the 9th International Conference on Semantic Systems. Graz, Austria, 2013: 121-124

[3] Bordes A, Usunier N, Garcia-Duran A, et al. Translating embeddings for modeling multi-relational data//Burges C J, Bottou L, Welling M, et al, eds. Neural Information Processing Systems (NIPS). Cambridge, USA: MIT Press, 2013: 1-9

[4] Niu Ze-Ping, Li Guo-Liang. In-database AI model optimization. Journal of Software, 2021, 32(3): 622-635(in Chinese)

(钮泽平，李国良. 数据库内 AI 模型优化. 软件学报，2021，32(3)：622-635)

[5] Zaharia M, Xin R S, Wendell P, et al. Apache spark: A unified engine for big data processing. Communications of the ACM, 2016, 59(11): 56-65

[6] Lerer A, Wu L, Shen J, et al. PyTorch-BigGraph: A large-scale graph embedding system. arXiv preprint arXiv: 1903.12287, 2019

[7] Chai L, Wang X, Liu B, et al. Efficient distributed knowledge representation learning for large knowledge graphs//Shao J, Yiu M, Toyoda M, et al, eds. Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data. Cham, Swiss: Springer, 2019: 398-413

[8] Dong S, Wang X, Chai L, et al. PDKE: An efficient distributed embedding framework for large knowledge graphs//Proceedings of the International Conference on Database Systems for Advanced Applications. Cham, Swiss: Springer, 2020: 588-603

[9] Zhou X H, Chai C L, Li G L, et al. Database meets AI: A survey. IEEE Transactions on Knowledge and Data Engineering, 2020. http://dbgroup.cs.tsinghua.edu.cn/ligl/papers/aidb.pdf

[10] Lin Y, Liu Z, Sun M, et al. Learning entity and relation embeddings for knowledge graph completion//Proceedings of the AAAI Conference on Artificial Intelligence. Austin, USA, 2015, 29(1): 2181-2187

[11] Han X, Cao S, Lv X, et al. OpenKE: An open toolkit for knowledge embedding//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Brussels, Belgium, 2018: 139-144

[12] Carbone P, Katsifodimos A, Ewen S, et al. Apache Flink: Stream and batch processing in a single engine. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2015, 36(4): 28-38

[13] Lehmann J, Sejdiu G, Bühmann L, et al. Distributed semantic analytics using the SANSA stack//Proceedings of the ISWC 2017 International Semantic Web Conference. Berlin, Germany: Springer, 2017: 147-155

[14] Li M, Andersen D G, Park J W, et al. Scaling distributed machine learning with the parameter server//Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). Broomfield, CO, 2014: 583-598

[15] Tencent. Plato. https://github.com/tencent/plato

[16] Alibaba. Euler-2.0. https://github.com/alibaba/euler

[17] Hellerstein J M, Ré C, Schoppmann F, et al. The MADlib analytics library or MAD skills, the SQL. Proceedings of the VLDB Endowment, 2012, 5(12): 1700-1711

[18] Boehm M, Dusenberry M W, Eriksson D, et al. SystemML: Declarative machine learning on spark. Proceedings of the VLDB Endowment, 2016, 9(13): 1425-1436

[19] Fernandes S, Bernardino J. What is BigQuery? //Proceedings of the 19th International Database Engineering & Applications Symposium. Yokohama, Japan, 2015: 202-203

[20] Abadi M, Barham P, Chen J, et al. TensorFlow: A system for large-scale machine learning//Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI'16). Savannah, USA, 2016: 265-283

[21] Chollet F. Keras: The Python deep learning library. Available online: https://keras.io/

[22] Trouillon T, Welbl J, Riedel S, et al. Complex embeddings for simple link prediction//Proceedings of the International Conference on Machine Learning (PMLR). New York, USA, 2016: 2071-2080

[23] Wang Z, Zhang J, Feng J, et al. Knowledge graph embedding by translating on hyperplanes//Proceedings of the AAAI Conference on Artificial Intelligence. Québec, Canada, 2014: 28(1): 1112-1119

[24] Wang Q, Mao Z, Wang B, et al. Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(12): 2724-2743

[25] Miller G A. WordNet: A lexical database for English. Communications of the ACM, 1995, 38(11): 39-41

[26] Bollacker K, Evans C, Paritosh P, et al. Freebase: A collaboratively created graph database for structuring human knowledge//Proceedings of the 2008 ACM SIGMOD International Conference on Management of data. New York, USA, 2008: 1247-1250

[27] Schlichtkrull M, Kipf T N, Bloem P, et al. Modeling relational data with graph convolutional networks//Gangemi A, Navigli R, Vidal M, et al, eds. European Semantic Web Conference. Cham, Swiss: Springer, 2018: 593-607

LIU Peng-Kai, M. S. candidate. His main research interest is knowledge graph data management.

WANG Xin, Ph. D., professor, Ph. D. supervisor. His main research interests include knowledge graph data management, graph database, and large-scale knowledge processing.

LIU Bao-Zhu, M. S. candidate. Her main research interest is knowledge graph data management.

CAI Shun-Ting, B. D. candidate. His main research interest is knowledge graph data management.

LI Si-Zhuo, M. S. candidate. Her main research interest is knowledge graph data management.

Background

Knowledge graphs, the important cornerstone of artificial intelligence, can represent the entities and relationships between them in the real world through graphs. Knowledge graphs provide a structural representation of data through triples. However, as the native representation form of knowledge graphs, triples can complicate the manipulation of knowledge graphs and the execution of downstream tasks, which will affect the computational performance. To tackle this issue, some knowledge graph embedding techniques are proposed to map the entities and relationships to a continuous vector space, some of which are semantic-aware, facilitating the manipulations on knowledge graphs.

Nevertheless, most of the existing researches focus on the effectiveness of the algorithms and neglect the efficiency of the algorithms. The existing methods often lack scalability and efficiency when dealing with large-scale data. Some distributed frameworks are proposed based on distributed computing technologies such as Spark or PyTorch, aiming at training knowledge graph embedding models in a distributed environment. However, these works are separated from each other for data storage and model training, and cannot simultaneously satisfy the requirement of efficient data storage, query processing, and computation.

In recent years, database technologies have been widely used in research to accelerate AI algorithms and optimize AI models. On the one hand, database technologies can be applied to efficiently manage large-scale data and train models, which is a decisive factor in wide applications of AI, and on the other hand, AI technology is also more often applied to enhance the quality of service and reliability assurance of databases.

This paper presents an in-database training engine for knowledge graph embedding models, DB4Trans (database for translation-based models). Based on the idea of "integration of storage and computation", we take a classical knowledge graph embedding model, TransE, as an example, and use the relational database to complete the training and storage of the knowledge graph embedding model. Database technologies are utilized to accelerate the workflow of model training and improve the efficiency of I/O during the model training process based on the designed storage and indexing scheme. A data batch exchange approach between database and memory is proposed to support the training and storage of large-scale data. Extensive experiments were conducted on datasets usually used for verifying the effectiveness of knowledge graph embedding models. The experimental results validate the effectiveness and efficiency of the proposed method.

This work is supported by the National Key Research and Development Program of China "Key Technologies and Systems for Distributed Knowledge Graph Data Management (2019YFE0198600)" and the National Natural Science Foundation of China "Research on Key Technologies for Distributed Storage and Querying of Large-scale Knowledge Graphs (61972275)".