Part1 论文摘要（abstract）和引言（introduction）翻译

1Abstract

我们思考一类问题：如何将多元关系数据的实体和关系嵌入到低维的向量空间。我们的目标是提出一种典型的、易于训练的模型，它不仅能减少参数的数量，而且还能放大到大规模数据集进行使用。由此，我们提出了 TransE 方法，他能够对关系进行建模——以一种将关系解释并转变为低维度向量的方式。它尽管简单，但确实有效，因为广泛的实验证明对于两个知识库的连接预测，TransE 超过了从前最好的模型。另外，它能够成功地在大规模训练集上进行训练。这类训练集普遍有千万级实体，两万五千个关系，以及超过一千七百万个训练样本。

2Introduction

多元关系数据是指有向图中包括头实体、尾实体、以及两者之间的关系类，表示为元组（h、l、t）。以这种三元组结构的任务有社交网络（成员互相连线）、推荐系统（用户和产品连线）、知识库（抽象概念、具体实体等事物相互连接）。我们的任务是对知识库的多元关系数据进行建模，同时提供一种高效的工具进行知识补全，而不需要在此基础上添加额外的知识以占用空间。

构建多元关系数据 通常，建模过程归结为提取实体之间的局部或全局连接模式，并通过使用这些模式来概括观察到的特定实体与所有其它实体之间的关系，并通过这种方式来进行预测。单一关系的局部概念是结构性的，比如：在社交网络中，我朋友的朋友还是我的朋友。除了这种方法（指依赖关系），还能够依赖实体，比如那些喜欢星战四的人基本上也喜欢星战五，但是他们或许不喜欢泰坦尼克号。与单一关系数据在对数据进行一些描述性分析之后可以做出临时但简单的建模假设相比，关系型数据的困难在于局部性的概念可能同时涉及不同类型的方法和实体。因此，对多关系数据进行建模需要更通用的方法，这些方法可以同时考虑所有异构关系来选择适当的模式。随着用户/实体聚类或矩阵分解技术在协同过滤中表示单关系数据中实体的连接模式之间的非平凡相似性，大多数现有的方法由参考[6]指出：也就是说，通过学习和操作实体和关系的潜在表征（来嵌入）。从这些方法到多关系域的自然扩展开始，例如随机块模型的非参数贝叶斯扩展 [7、10、17] 和基于张量分解[5]或集体矩阵分解的模型[13、11、12]，许多最新的方法都集中在提高模型在贝叶斯聚类框架[15]或基于能量的框架中的表达性和普遍性，用于学习低维空间中实体的嵌入[3、15、2、14]。这些模型的更大表现张力是以模型复杂性的显著增加为代价的，这会导致产生夸张的建模假设和更高的计算成本。此外，由于难以设计此类高容量模型的适当正则化，此类方法可能会过度拟合，或者由于需要解决许多局部最小值的非凸优化问题来训练它们而导致拟合不足。事实上，在[2]中表明，在具有相对大量不同关系的多个多关系数据集上，更简单的模型（线性模型而不是双线性模型）拥有和最具表现力的模型差不多好的性能。这表明即使在复杂和异构的多关系域中，简单而适当的建模假设也可以在准确性和可扩展性之间进行更好的权衡。

关系可以作为嵌入空间之间的翻译 我们引入 TransE，它是一种基于能量的模型，用于学习低位实体的插入。在 TransE 中，关系被表示为嵌入空间中的翻译，如果向量模型是成立的，那么尾部实体 t 的嵌入应该接近于头部实体 h 的嵌入加上一些依赖于关系的向量。我们的方法依赖于一组减少的参数，因为它只会为每个实体和每个关系学习一个低维向量。我们这种转译的动机在知识库中极为常见，转译便是他们的自由转换。实际上，考虑到树的自然表示，兄弟节点彼此靠近，并且给定高度的节点组织在 x 轴上，与此同时，父子类关系对应于 y 轴。由于没有平移的向量对应实体的等价关系，因此模型也可以表示兄弟关系。因此，我们使用一个低维向量来表示我们认为的知识库中的关键关系。另一个动机来自

[8]的工作，作者从自由文本中学习词嵌入，以及不同类型实体之间的一些一对一关系，例如国家和城市之间的“首都”，是“巧合而非自愿”地由模型表示为嵌入空间的翻译。这表明可能存在嵌入空间，其中不同类型实体之间的一对一关系也可以由翻译表示。我们模型的目的是强制执行嵌入空间的这种结构。我们在第四节的实验表明，尽管模型简单且初衷是为了建模层次结构，但最终在大多数类型的关系上都有着非常强大的表现。而且，对于现实世界的知识库，其链接预测方面优于最先进的模型。它的轻参数化使其能够在包含 1M 实体、25K 关系和超过 17M 训练样本的 Freebase 大规模拆分上成功进行训练。

Part2 问题描述

1 **开篇陈述**

面对大型的关系型数据，传统的构建方式会产生更大的计算成本和更复杂的模型构建。TransE 模型基于低维向量，根据能量模型，实现了更加高效和更加简洁的链路预测，不仅有助于知识库的联想与构建，同时为后面的 TransH，TransR，CTransR 和 TransD 奠定了基础。

2 **具体问题**

在低维向量空间中，将多种关系的图谱中的实体和关系在一个低维空间中进行表示，获取每个实体的表征结果。

提出一种易于训练的规范模型，该模型包含数量较少的参数，并且可以扩展到非常大的知识库。

对知识图谱中的多元关系进行建模，在不引入额外知识的情况下，高效实现知识补全，关系预测。

3 **解决方式**

文章将多元关系表示为类似于 (h、l、t) 的元组。在以此构建多元关系数据的时候，为了解决过度拟合和拟合不足的问题，基于本文的线性模型而非双线性模型能够更好、更快地解决问题。

假设向量模型是成立的，尾部实体的嵌入应该接近头部实体 h 的插入加上依赖关系的向量。使用能量模型来对抽象向量进行低维化处理使得加速链路预测的进程。并且实验也证明了该方法的可行性。

4 **期望效果**

达到同年代最高的 link prediction benchmark 水平（测试 WN18RR hits@10 benchmark）。

5 **总结问题**

通过典型模型的分析，验证了 TransE 的高表现水平（于 2014 年），为自然语言处理领域提供了一种新的解决方案。

Part3 输入输出、模型算法描述

在了解算法之前，先了解一些相关知识：

1 **距离模型的结构表示** 结构表示中，每个实体用 k 维向量表示，所有实体被投射到同一个 k 维向量空间中。对于本文，当两个实体处于一个三元组时，两个向量相近。

$$f_i(h,t)=\left| hL_1-tL_2 \right|$$

2 **神经张量网络** 神经张量网络在不同维度下将头尾实体向量连接起来。

$$s(h,l,t)=h_T L t+l_{1T} h+l_{2T} t$$

3 **算法描述** 给定一个训练集 S，三元组表示为（h，l，t）其中 h，t∈E（实体），l∈L（关系类），实体和关系的嵌入维度设为 k，我们希望 h+l 尽可能与 t 相似，因此定义能量函数：

$$d(h+l,t) = [(h+l)-t]^2 = ||h||_2^2 + ||l||_2^2 + ||t||_2^2 - 2(h^T t + l^T(t-h))$$

其相当于引入了一个高维空间下的距离，即欧氏距离。那么我们如何训练嵌入（实体嵌入和关系嵌入）呢？这里采用了正负样本方式——利用空间传递不变形原理，找到一个实体和向量空间，使得关系三元组之间的势能差值最小。

什么是正样本和负样本？ 这里的正负并不代表数值正负，而是代表关系正负。**正样本是指属于某一类别的样本，负样本是指不属于某一类别的样本，仅此而已。**

因此，根据优化原则（让有关系的样本尽可能近，没有关系的样本尽可能远（个人猜想）），采用了如下的间距排序目标优化函数：

$$L = \sum_{(h,l,t)\in S} \sum_{(h',l,t')\in S'_{(h,l,t)}} [\gamma + d(h+l,t) - d(h'+l,t')]_+$$

其中[x]+表示 x 中的正例部分，γ>0 表示距离因子。S'的替换方法是：将正确三元组的头或尾替换成其它的（每次只选择头或尾进行替换，不同时替换），得到错误的三元组。

4 **伪代码实现**

Algorithm 1 Learning TransE

input Training set $S = \{(h,\ell,t)\}$, entities and rel. sets E and L, margin γ, embeddings dim. k.

1: **initialize** $\ell \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$
2: $\ell \leftarrow \ell / \|\ell\|$ for each $\ell \in L$
3: $\mathbf{e} \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$
4: **loop**
5: $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$
6: $S_{batch} \leftarrow$sample(S, b) // sample a minibatch of size b
7: $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets
8: **for** $(h,\ell,t) \in S_{batch}$ **do**
9: $(h',\ell,t') \leftarrow$sample$(S'_{(h,\ell,t)})$ // sample a corrupted triplet
10: $T_{batch} \leftarrow T_{batch} \cup \{((h,\ell,t),(h',\ell,t'))\}$
11: **end for**
12: Update embeddings w.r.t. $\sum_{((h,\ell,t),(h',\ell,t'))\in T_{batch}} \nabla[\gamma + d(\boldsymbol{h}+\boldsymbol{\ell},\boldsymbol{t}) - d(\boldsymbol{h'}+\boldsymbol{\ell},\boldsymbol{t'})]_+$
13: **end loop**

在 1-3 步中，使用了正态分布初始化参数 entity，但是并没有对实体进行范数归一化；

在 4-13 步的循环中，第五步进行范数归一化，除以了一个参数 L2；接下来进行迭代（epoch），第六步采样一个大小为 b 的 batch，并初始化三元组（第七步）；

在 8-11 步的子循环中，对 s batch 中的每个正样本进行替换并获得一个负样本，组成 t batch，进行填充，完成样本的提取。

在 12 步，进行梯度下降训练目标函数，并调整向量，得到结果。并进行下一次迭代。不同的 batch size 会影响实验的最终学习结果。

5 **输入输出** 输入数据集以获取预测模型。

Part4 评价指标及其计算公式

1 **评价指标** 实验采用 WN18RR(WordNet 18relationships)数据集进行测试。WN18RR 共有 18 种关系、40943 个实体和 93003 个三元组，其中，有一些文本三元组是通过从训练集中反转已经存在的三元组获得的，不过这并不影响我们的实验。实验中，测试集的头实体或尾实体被移除，轮流被换为字典中的任意一个实体，因此首先计算错误三元组的得分并升序保存得分，得到 raw rank；然后保存正确的实体得分，得到 filtered rank。（某些错误的三元组会变成有效的三元组，即可能出现错误三元组比测试集三元组靠前的情况，删除错误的三元组就可以保证后续实验的正常进行）

2 **计算公式** 报告预测排名和 hits@10 的平均值，即排名前十名的正确实体的占比，随后移除所有的错误元组。

Hits@10(%)= ∑(true entity/top 10 entity)/times*100%（times 表示试验次数）

Mean rank= ∑(∑ （level of certain kind of entities）/num of entities)/times

3 **附** 其实方法不止一种。去掉中间的关系，给定已知的头实体尾实体来预测关系，同样能达到对应的效果；还可以根据样本来分析新关系，但由于样本不便大量处理不做展示。

Part5 对比方法及其引用出处

Table 3: **Link prediction results.** Test performance of the different methods.

DATASET	WN				FB15K				FB1M	
METRIC	MEAN RANK		HITS@10 (%)		MEAN RANK		HITS@10 (%)		MEAN RANK	HITS@10 (%)
Eval. setting	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Raw*
Unstructured [2]	315	304	35.3	38.2	1,074	979	4.5	6.3	15,139	2.9
RESCAL [11]	1,180	1,163	37.2	52.8	828	683	28.4	44.1	-	-
SE [3]	1,011	985	68.5	80.5	273	162	28.8	39.8	22,044	17.5
SME(LINEAR) [2]	545	533	65.1	74.1	274	154	30.7	40.8	-	-
SME(BILINEAR) [2]	526	509	54.7	61.3	284	158	31.3	41.3	-	-
LFM [6]	469	456	71.4	81.6	283	164	26.0	33.1	-	-
TransE	**263**	**251**	**75.4**	**89.2**	**243**	**125**	**34.9**	**47.1**	**14,615**	**34.0**

该图为原论文的比较方法，这里只采用 WN 的 hits@10

1 **对比方法** 我们采用其它的模型已经拥有的数据和通过实验产生的 TransE 的数据进行比较。在这种比较中，我们只比较 hits@10，因为无论在样本数量多少的情况下，hits@10 都是相对稳定的。原论文采用比较 mean rank 和 hits@10 的方法，是因为测试集的单位数量相同。

2 **引用出处**

2.1 Translating Embeddings for Modeling Multi-relational Data | Papers With Code

2.2 论文解读：（TransE）Translating Embeddings

2.3 三元组损失

2.4 一文让你轻松理解 Focal Loss、正/负样本、难/易分样本

Part6 结果

Batches count=10

Seed=0

Epochs=1200

学习率 1e-3

Margin 值 0.5

L2 正则权重 1e-5

Hits@10：0.824100

高于 Part5 中其它模型的 WN Hits@10 比率 实验有效

```python
#导包
import numpy as np
from ampligraph.datasets import load_wn18
from ampligraph.latent_features import TransE
from ampligraph.evaluation import evaluate_performance, mrr_score, hits_at_n_score

def main():

    #加载Wordnet18数据集
    X = load_wn18()

    # 用pairwise损失函数初始化TransE模型
    model = TransE(batches_count=10, seed=0, epochs=1200, k=150, eta=10,
                   # 使用adam优化器，学习率为1e-3
                   optimizer='adam', optimizer_params={'lr':1e-3},
                   #使用pairwise损失函数，margin值为0.5
                   loss='pairwise', loss_params={'margin':0.5},
                   #L2正则，正则权重为1e-5
                   regularizer='LP', regularizer_params={'p':2, 'lambda':1e-5},
                   # 显示进度条
                   verbose=True)

    #定义filter用于过滤替换头实体或尾实体生成负例时产生的正例样本
    filter = np.concatenate((X['train'], X['valid'], X['test']))

    #通过训练集、验证集训练模型
    model.fit(X['train'],
              early_stopping = False,
              )

    #在测试集上进行评估
    #可以通过filter_triples=None不进行过滤
    ranks = evaluate_performance(X['test'],
                                 model=model,
                                 filter_triples=filter,
                                 use_default_protocol=True,#分别替换头实体、尾实体
                                 verbose=True)

    # 计算并打印评价指标:
    mrr = mrr_score(ranks)
    hits_10 = hits_at_n_score(ranks, n=10)
    print("MRR: %f, Hits@10: %f" % (mrr, hits_10))

if __name__ == "__main__":
    main()
```

```
Average TransE Loss:  0.000490: 100%|██████████| 1200/1200 [04:13<00:00,  4.74epoch/s]WARNING - DeprecationWarnir

100%|██████████| 5000/5000 [00:33<00:00, 150.95it/s]
MRR: 0.402440, Hits@10: 0.824100
```

201250123

刘晓旭

2022 年 10 月 2 日