# TransH算法及应用示例

## 一、内容简介

TransH算法是wang zhen等人在2014年提出的一种针对于TransE模型的不足而进行了改进之后的基于Translation思想的知识图谱表示学习方法，论文题目为《Knowledge Graph Embedding by Translating on Hyperplanes》。TransE算法是一个经典的知识图谱表示学习算法，通过对头实体、关系和尾实体的近似于向量加法的建模，通过负采样和随机梯度下降进行优化，最终得到实体和关系的低维向量表示。但是TransE的最大缺点在于对于复杂关系的建模缺乏能力，比如现实世界中常见的一对多、多对一、多对多的关系，因此，Trans H模型旨在解决TransE模型的这些局限性，通过将三元组中的关系抽象为向量空间中的超平面，进而在这个超平面进行头实体和尾实体之间的Translation操作。

## 二、理论基础

### 2.1 知识点一 知识图谱基本理论

2.1.1 什么是知识图谱

知识图谱是一种用图模型来描述知识和建模世界万物之间的关联关系的技术方法。知识图谱旨在利用图结构建模、识别和推断事物之间的复杂关联关系和沉淀领域知识，是实现人工智能的重要基石。

知识图谱中的基本的知识表示形式为（头实体、关系、尾实体）这样的三元组形式，一个三元组可以认为表示了一个具体的知识。

### 2.2 知识点二 表示学习与知识图谱的表示学习

2.2.1 表示学习

表示学习，英文为Representation Learning，又被称为特征学习或表征学习。在深度学习领域，表示是指通过模型的参数，采用何种形式、何种方式来表示模型的输入观测样本，其能将原始数据转换成为能够被机器学习有效利用的一种形式，最常见的就是把一个原始数据样本表示为一个低维、连续空间中的向量。

表示学习得到的低维向量表示是一种分布式表示(distributed representation)。之所以如此命名，是因为孤立地看向量中的每一维，都没有明确对应的含义；而综合各维形成一个向量，则能够表示对象的语义信息。

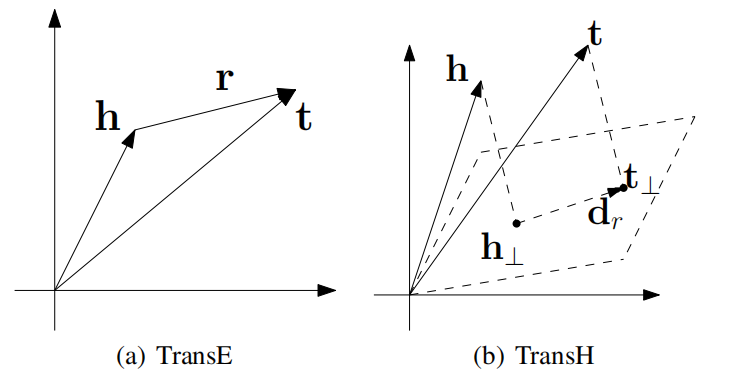
2.2.2 知识图谱的表示学习

知识图谱的表示学习，也就是对图谱中所包含的知识（实体或关系）进行表示学习。通过将实体和关系投影到低维向量空间，就能够实现对实体和关系的语义信息的表示，可以高效计算实体、关系之间的复杂的语义关联。

常见的知识图谱表示学习的应用包括：（1）相似度计算：计算实体之间的相似度；（2）知识图谱补全：知识图谱的构建是一个动态丰富的过程，利用知识表示模型，可以预测两个实体之间的关系，进而补全未包含的知识；（3）NLP任务：如知识抽取、问答系统、实体链接等；（4）推荐系统：将商品和用户等实体进行表示学习从而利用知识图谱的语义信息进行推荐；

### 2.3 知识点三 TransH核心思想

前面介绍了知识图谱表示学习的概念，也就是对图谱中的知识进行表示学习，使得计算机可以对这些知识进行计算。我们曾说到TransE模型在处理自反现象、一对多关系、多对一关系等非单一映射的复杂关系时的能力并不强大。以一对多关系为例，在这样的关系下，按照TransE模型的约束条件和优化目标，会使得head向量和relation向量的和尽可能等于尾实体向量，但是这样的话就会使得多个尾实体所得到的向量几乎是一模一样的，这是明显存在问题的。因此，在TransH模型中，通过对每一个关系向量学习一个超平面，从而使得同一个实体节点在不同关系的超平面中的向量表示不同。如下图所示，是TransH模型和TransE模型的区别：

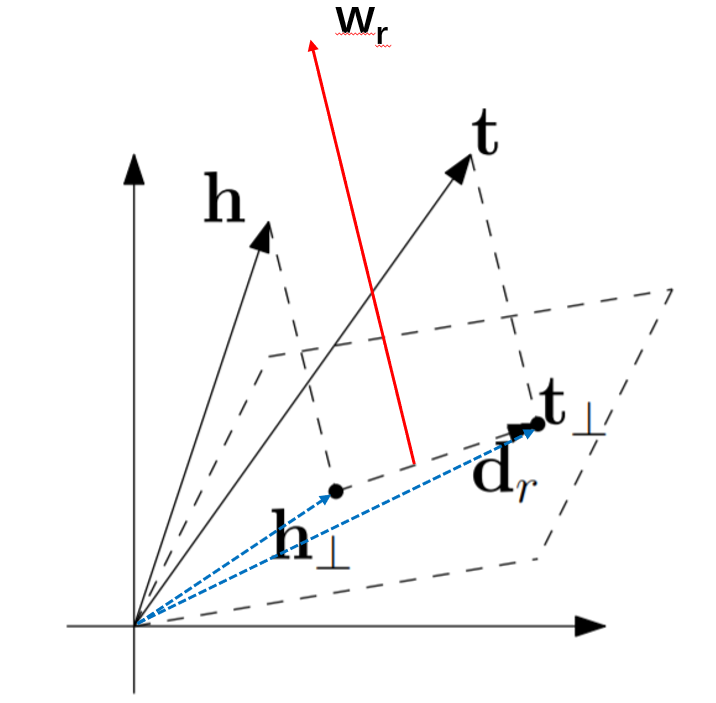


由上述示意图可知，对于一个三元组，h和t分别表示头实体向量和尾实体向量，虚线所示的平面即为关系特定（relation-specific）的超平面，该超平面的法向量定义为wr，dr表示超平面上的Translation向量，h⊥和t⊥分别表示头实体向量和尾实体向量在该超平面上的投影。因此，与TransE模型类似的，TransH模型中的Translation操作如下：

可以看到，这与TransE模型的基本思想是极其类似的，所以可以说TransH是TransE的改进版本。

因此，算法核心思想可以总结为：对实体向量进行特定关系下的映射，在映射后的平面执行TransE算法的翻译操作，进而对目标函数进行优化得到表示学习的结果。

### 2.4 知识点四 TransH算法流程



2.4.1 总体算法流程

TransH算法的总体流程如下所示：

1. 对于一个三元组，首先将头实体和尾实体映射到超平面上，得到映射后的向量h⊥和t⊥，如下所示：

其中的h、t、wr是待学习的参数；

1. 得到投影向量之后，根据下列的目标函数进行三元组的翻译计算：

对于该公式所期望的结果，若三元组为正确三元组，则其结果fr应尽可能小、若三元组为错误的三元组，则其结果应尽可能大；

1. 使用SGD算法对训练三元组数据的上述计算结果进行优化训练，在

训练的过程中需要满足三个约束条件：

简要来说，约束1保证了所有实体的embedding表示的归一化；约束2保证了wr和dr正交垂直，即保证了关系向量准确落在其对应的超平面上；约束3保证了超平面法向量的模为1；

2.4.2 算法输入输出

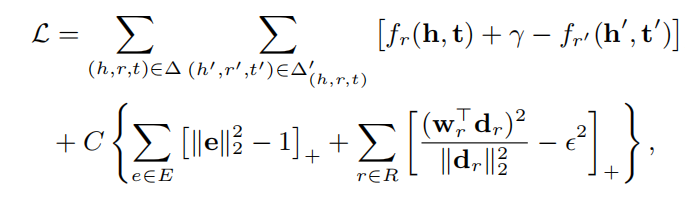
算法输入：三元组集合S；实体集合E、关系集合L；嵌入维度k；

算法输出：实体和关系的embedding向量表示；

2.4.3 负样本构造

由于负样本的构造是基于已有的三元组集合来进行头实体或尾实体的替换，从而构造出图谱中并不存在的三元组，并最大化正负三元组之间的差异；因此，实际应用中可以选择随机头实体或尾实体进行替换、也可以设置两个参数tph或hpt来选择更换头实体还是更换尾实体。

2.4.4 损失函数及优化



TransH算法要优化的损失函数如上所示， h’和t’是替换后的负样本三元组中的头实体或尾实体。损失函数的优化的理想情况为对于正确的三元组，第一项可能小，负样本三元组第一项要尽可能大，这样整体的Loss趋近于0。损失函数中的第二项为三个约束项的损失函数，C表示软约束权重，其也是训练过程中的一个超参数。

### 2.5 知识点五 TransH算法优缺点

2.5.1 TransH算法优点

（1） 解决了TransE模型不能够对实体之间的复杂关系的有效建模的局限性；

（2）提出了一种基于关系映射的知识表示学习方法，在链接预测、三元组分类任务中都取得了更好的结果；

（3）论文作者对负采样策略进行改进，提出了一种概率采样的方法，降低了错误标签三元组带来的影响；

2.5.2 TransH算法缺点

（1）TransH算法中，头实体和尾实体仍然处于同样的语义空间中。后续相关学者认为三元组的关系更多关注于实体的属性，也即不同的关系体现了不同的属性。因此，基于TransH的改进版本的TransR模型呼之欲出。

## 三、演示验证

### 3.1 运行环境搭建

无非常规环境或特殊工具包。

### 3.2 实验任务一 数据集加载

### 3.3 实验任务二 TransH核心代码

步骤2.1 初始化实体和关系的embedding

步骤2.2 对实体和关系的embedding进行规范化

步骤2.3 定义目标函数

步骤2.4 构造负样本三元组

步骤2.5 定义模型的计算过程

### 3.4 实验任务三 训练及测试

步骤3.1 载入训练集进行模型训练

Dataloader()、Model()、Model.train()

步骤3.2 使用测试集进行测试

Dataloader()、Model.test()

步骤3.3 训练过程参数的可视化

Visulization()

步骤3.4 输出实体和关系的embedding结果

Output\_Embeddings()

## 四、实验环境

以下仅供参考：

操作系统：windows10

平台软件：Anaconda、torch、Python

## 五、界面设计

