Knowledge Embedding 知识嵌入 简介

任务

模型： 将一个知识抽象为一个三元组 $(Head, Relation, Tail)$ 其中 Head 和 Tail 是实体，表示有具体所指的物体或概念，Relation 表示实体间的关系。

知识图谱使结构化的语义知识库，用于以符号形式描述物理世界中的概念和相互关系，其基本组成单位是上述三元组和实体及其相关 属性-值 对，实体间通过关系相互连接，构成网状的知识结构。如表示 北京是中国的首都 可以抽象为 （北京，首都，中国） 。可以将知识图谱视为一个图，其中的点表示实体，边表示实体间的关系。使用较多的数据集有 WordNet(Miller 1995), Freebase(Bollacker stal. 2008)等。

如果使用基本的词嵌入方法如 word2vec：相当于一个 one-hot 编码，将一个词转换成一个长度等于字典大小的向量。

```
字典：['I', 'am', 'a', 'student']
通过在字典中的位置对单词进行描述，如：
'I' ===> [1, 0, 0, 0]
'am' ===> [0, 1, 0, 0]
```

显然这样的嵌入方式只能保证在有字典的情况下将词汇抽象成一个向量，但是主要缺点在于维数过高且忽视了词汇间的逻辑联系，不能满足大规模下的应用。所以有必要对 embedding 进行研究。

现在为了提取出实体间的关系，知识嵌入主要分成了三个类别 张量分解(tensor factorization)，基于翻译模型的求解(translation-based approcah)和神经网络。本文主要的内容是有关第二类的，其中主要的方法是构建一个基于最大间隔的训练目标（参考SVM）。

在知识图谱中存在大量这样的三元组，结构也不尽相同，包括一对一，一对多，多对一和多对多。这样多样化的知识的形式为我们表示带来了很大困难。如何能描述，抽象这些实体从而快速推导出其中的关系成为了 Knowledge Embedding 的研究目标

评价指标

对一个三元组 (h, r, t) 任意用其他的实体替换头或者尾，并根据评价函数 f_r 计算每个新三元组的不相似程度，并对所有的得分进行降序排列。

1. mean rank
   对某个正确的三元组 (golden triplet) 进行上述的操作，并得到其在排序中的位置 $index_i$。对所有的正确三元组进行上述操作得到的平均数即为所求。即$MeanRank = \frac{1}{n}\sum_{i=1}^{n} index_i$
   mean rank 越小越好。

2. Hits@10
   对某个正确的三元组 (golden triplet) 进行上述的操作，并得到其在排序中的位置。Hits@10 指满足 $index_i \leq 10$ 的三元组在数据集中占的比例。
   Hits@10 越大越好。
3. 判别任务（不常用）
   给定三元组 (h, r, t) 判断其是否正确。是一个简单的二分类问题，若三元组的 loss 大于一个和关系相关的阈值 σ_r 则判为反例，否则为正例。通过分类的准确性判断模型好坏。

前置知识

在词向量中描述的固定词间关系是相同的，即加上相同的向量可以维持相同的距离，如

> vect(man) - vect(woman) ≈ vect(king) - vect(queen)

同理通过计算实体间的距离也可以对其之间的关系进行合理推测。

发展历程

较为早期的工作暂时不进行讨论，本文主要介绍 Trans 系列的部分相关工作。因为笔者的水平和时间不足，仅进行 Trans 系列前几篇的研究和探讨。

TransE

论文：Translating Embeddings for Modeling Multi-relational Data [NIPS 2013]

思想：通过 margin 的方法构建损失函数（优化目标），合理通过知识间关系将实体嵌入到一个低维空间中。

算法：

1. 输入训练集 $S = (h, l, t)$ 实体集合 E，关系集合 L 和边际距离 γ 进行嵌入的维数 k
2. 将每个实体和关系抽象为一个 k 维向量，均初始化为 $[uniform(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})] * k$ 为了保证初始时数据分布均匀和加速收敛，使用均匀分布生成初始值。
3. 将 L 中的每个关系进行单位化，在之后的迭代中关系向量不变
4. 循环
   1. 将 E 中的每个实体对应的向量进行单位化（迭代）
   2. $S_{batch} = sample(S, b)$ 在 S 中进行大小为 b 的抽样生辰一个 batch
   3. $T_{batch} = \emptyset$ 初始化三元组对（参考 SVM 为了训练成果尽量准确，需要数量相近的正反例）
   4. $for (h, l, t) \in S_{batch}$ 对 batch 中的每个三元组生成反例
      1. $(h', l, t') = Sample(S'_{h,l,t})$ 固定中间的 relation 通过替换头尾实体生成反例
      2. $T_{batch} = T_{batch} \cup (h, l, t), (h', l, t')$
   5. 通过 SGD 进行梯度下降更新参数

损失函数如下：

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

其中 $d(x,y)$ 用于衡量 x 和 y 的不相似程度，可以使用 L_1 距离或 L_2 距离。$[x]_+$ 表示 x 的整数部分，所以上式等价于

$$\Gamma = \sum_{(h,l,t)\in S} \sum_{(h',l,t')\in S'} max(0, \gamma + d(h+l,t) - (h'+l,t'))$$

算法的核心在于使 $d(h+l,t)$ 趋于 0 并使 $d(h'+l,t')$ 区域负无穷

算法缺陷：实体和关系在同一个向量空间中，对 一对一 结构的知识尚可，不能很好的嵌入其他形式的知识。不同关系中的实体向量是相同的，造成在其他形式的知识中涉及到的实体可能有相同或者相似的向量表示。

TransH

论文：Knowledge Graph Embedding by Translating on Hyperplanes [AAAI 2014]

思想：分离了实体和关系的空间，在进行距离计算之前先将实体映射到对应关系所在的超平面，改变实体在不同关系下的意义。与 TransE 的区别在于分开在超空间内的实体向量和实体空间本身的向量。实验表明对各种的样式的三元组上的预测效果相比 TransE 均有显著改进。

> 举个例子：（奥巴马，总统，美国）和（特朗普，总统，美国）
> 在 “总统” 对应的超空间内，奥巴马和特朗普的关系相近，但是在原本的实体空间内这两个实体的距离可以很远。
> 低维空间内分不开的可以在高维空间内区分。

特点：每个关系有两个向量进行描述

1. W_r：描述超平面的单位向量（法向量）
2. d_r：在超平面中描述关系的向量

算法：计算过程和损失函数和 TransE 基本相同 其中 f_r 为指定关系（即指定空间）下的 $d(h_\perp + d_r, t_\perp)$ 其中 $h_\perp$ 和 $t\perp$ 表示头尾向量在超空间上的投影。
目标函数如下：

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

限制：

1. 每个实体的向量的范数小于等于1（单位化）
2. w_r 和 d_r 正交，即$|w_r^T d_r|/||d_r||_2 \leq \epsilon$
3. 保证 w_r 为单位向量

因为加上了上述的限制，所以应对 loss 进行一定程度的改变，自然想到拉格朗日乘数法。则现在的损失函数如下：

$$\Gamma = \sum_{(h,l,t)\in S} \sum_{(h',l,t')\in S'} [\gamma + f_r(h,t) - f_r(h',t')]_+ + C\{\sum_{e\in E}[||e||_2^2 - 1] + \sum_{r\in R}[\frac{(w_r^T d_r)^2}{||d_r||_2^2} - \epsilon^2]_+\}$$

可以注意到在上式中没有涉及到第三条限制，因此在每次进行投影之前应当将 w_r 单位化。上式中的 C 是一个超参数，用于调整惩罚的力度。

特别贡献：**降低 false negative labels 的方法**

> 对头节点和尾节点赋一个采样概率
> $P_{head} = \frac{tph}{tph+hpt}$ 以此概率替换头节点
>
> $P_{tail} = \frac{hpt}{tph+hpt}$ 以此概率替换尾节点
> tph：对于每个 head 的 tail 的数量
> hpt：对于每个 tail 的 head 的数量
> ===> 用于描述知识模型 (1-to-1 / 1-to-n / n-to-1 / n-to-n)

缺陷：实际的实体和关系可能有多种属性，不能统一的将其在一个空间中进行表述。

TransR

论文：Learning Entity and Relation Embeddings for Knowledge Graph Completion [AAAI 2015]

思想：因为对同一个关系，不同的头尾实体可能侧重于关系的不同方面，即不能将关系认为是一个恒定的向量。
和 TransH 相同，TransR 也将实体投影至关系所在的空间，并在此映射关系下使有关系的实体相互靠近，无关的实体相互远离。但是这里的关系空间脱离了 TransH 中的超平面的要求，即实体空间和关系空间相互独立。

特点：假设实体 $e \in R^k$ 关系 $r \in R^d$ 每个关系都有一个投影矩阵 $M_r \in R^{k\times d}$ 用于将实体投影至关系空间。为了避免过拟合，**使用 TransE 的结果进行实体向量的初始化**并将投影的矩阵置为单位阵（在多余的部分补0）。事实上在训练时也只是在计算 loss 之前乘上投影矩阵，其他的部分和 TransE 的训练步骤相同。

聚类： Cluster-based TransR
为了区分同一关系在不同的语义下的关系，对同一关系下不同的三元组（经过 TransE 预训练）根据

$(h-t)$ 进行聚类。根据聚类的结果可以得到关系在不同的类簇中的表示 r_c。
因此其损失函数变为

$$f_r = ||h_{r,c} + r_c - t_{r,c}||_2^2 + \alpha||r_c - r||_2^2$$

其中 $h_{r,c}$ 和 $t_{r,c}$ 表示属于当前类簇的头尾实体。后面的 $||r_c - r||_2^2$ 保证了每个类簇的中心不会距原始的关系向量 r 过远，可以认为是惩罚项。

缺陷：只考虑到了不同类的关系，忽视了实体也有不同的属性。使用矩阵描述对应关系，参数过多且计算复杂度较高。

TransD

论文：Knowledge Graph Embedding via Dynamic Mapping Matrix [ACL 2015]

思想：基于 TransR 进行改进，同时考虑到了实体和关系的不同类型。对不同类型的实体应该采用不同的映射方式，例如 `location` 可以表示国家和地区的包含关系，也可以表示大陆和国家的位置关系。因此通过动态矩阵，结合实体和关系来解决上述问题。

特点：

1. 使用了动态矩阵，映射关系和头尾实体和关系均有关。对每个实体和关系均使用两个向量进行表示。一个用于表示语义，另一个用于描述映射关系（用下标 p 表示），映射关系如下：

$$M_{rh} = r_p h_p^T + I$$

$$M_{rt} = r_p t_p^T + I$$

   显然这样的投影矩阵的产生与实体和关系都有关。然后通过各自的投影矩阵将实体映射到关系空间，即：

$$h_\perp = M_{rh} h$$
$$t_\perp = M_{rt} t$$

   余下的部分和 TransR 中描述的相同。可以注意到当实体空间和关系空间的维数相同，且用于投影的向量均为0，则 TransD 退化为 TransE。
2. 使用向量运算代替了 TransR 的矩阵运算，显著降低了计算的复杂度。如下所示：

$$h_\perp = M_{rh} = h_p^T h r_p + [h^T, 0^T]^T$$

$$t_\perp = M_{rt} t = t_p^T t r_p + [t^T, 0^T]^T$$

缺陷：因为每个关系上进行学习的参数相同，简单的关系上容易过拟合，但是在复杂的关系上容易欠拟合。

简单对比

以上各个模型的参数规模和解决的问题如下表

模型	参数规模	解决问题
TransE	$O(N_e m + N_r n)$	baseline & 1-to-1
TransH	$O(N_e m + 2N_r n)$	n-to-1 & n-to-1
TransR	$O(N_e m + N_r (m+1) n)$	扩展维度
CTransR	$O(N_e m + N_r (m+c) n)$	多种问题
TransD	$O(2N_e m + 2N_r n)$	多种问题