知识图谱模型简介

林纪平

2021 年 10 月 31 日

1 知识图谱简介

1.1 知识图谱概念

知识图谱 (Knowledge Graph) 是一种使用图模型来集成数据的知识库，通常用于存储具有自由形式语义的实体和他们之间的相互关联。知识图谱通常由节点和边组成，节点代表实体或抽象概念，边可以代表实体的属性或实体之间的关系。

1.2 知识图谱的主要应用

辅助搜索：知识图谱可以用于实现直接对语义的搜索。比如 Google 知识图谱使用语义检索为 Google 搜索提供结构化以及详细主题信息，使用户可以直接通过搜索来解决他们想查询的问题 [3]。知识图谱也被应用于拓展现有的**推荐系统** (Recommender System)，可以将关于商品、用户等实体的结构化知识加入推荐模型中以改善原有模型数据稀疏的问题。知识图谱中的数据存储形式大大降低了计算实体间的关联性的复杂度并且可以有效利用实体和用户之间的关系，从而更有效地寻找偏好实体 [2]。

辅助问答：知识图谱存储的信息是人与机器进行自然语言问答的关键之一，例如 Apple 的 Siri 和 Amazon 的 Alexa 都依靠知识图谱来实现与用户之间的深度知识问答。深度知识问答在智能驾驶，智能家居，智能厨房以及智能医疗等领域的应用也是基于知识图谱的问答技术和方法 [2]。

辅助大数据分析：知识图谱也被用于辅助进行数据分析和决策。例如通过知识图谱和语义技术增强数据之间的关联，使数据更直观化，以便用户更有效的对数据进行挖掘和分析。知识图谱也可被用于从文本数据中抽取实体和关系，也可以用于辅助实体消歧，指代消解和文本理解等 [2]。

2 知识图谱表示方法

2.1 三元组

基于后续的嵌入方法，我们主要讨论**资源描述框架**（Resource Description Framework, or RDF) 的知识图谱表示框架。在 RDF 中，知识实体总是以**三元组**的形式出现，每一份实体可以被分解成如下形式：(subject,predicate,object), 即 (主语，谓语，宾语) 的形式。RDF 中的主语是一个个体，谓语通常是一个属性或关系。谓语可以连接两个个体，或者连接一个个体和一个数据类型的实例，例如 (Beijing, is capital of, China)，(Jeff Bezos, CEO of, Amazon)。在建模中，三元组也会以（head,relation,tail) 的更简洁直观的表示方法出现。

2.2 向量表示

在知识图谱嵌入的模型中，比较重要的方法是将知识图谱中的实体和关系映射到向量空间以实现更有效率的模型。对于词向量，即三元组里的三个个体，以多维向量的形式表示。常见的有**词袋模型**

(Bag-of-Words) 和**词嵌入** (Word Embedding)。后续的知识图谱嵌入也主要使用词向量来统一化复杂的知识数据。

2.3 知识图谱嵌入

在词向量的启发下，我们将主要讨论如何将知识图谱的实体和关系映射到向量空间中，并包含语义信息。不同的嵌入模型会利用不同的方式使映射后的词向量和关系向量展现语义层面的联系。这样的向量映射使得后续计算机对于模型的机器学习等操作更加简便和有效。

3 转移距离模型

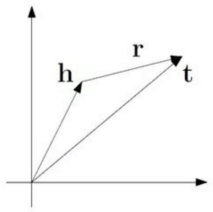
3.1 概述

转移距离模型 (Translational Distance Model) 顾名思义分为 Translate 和 Distancing 两步，即先将实体向量化 (Translate) 然后利用距离 (Distance) 来衡量实体的合理性。简而言之，转移距离模型将三元组的合理性问题转化为头尾实体的距离问题。

3.2 TransE

3.2.1 基本概念和原理

TransE(**Trans**lation **E**mbeddings for Modeling Multi-relational Data) 模型是转移距离模型中第一个研究成果，也是最经典，最简单的模型，也为其他转移距离模型打下了理论基础。其中心思想是如果一个三元组 (head,relation,tail) 成立，那么 head 和 relation 的和要尽可能与 tail 接近；相反如果三元组不成立，那么 head 和 relation 的和要尽可能与 tail 远离 [4]。TransE 模型主要对于 1 对 1(1-to-1) 的三元组关系进行建模，比如国家与首都的 1 对 1 关系 (Paris, is-capital-of, France)[1]。所以如果想要判断一个三元组的正确性，TransE 给出的得分函数 (Score Function) 是:



$$f_r(h,t) = \|\mathbf{h} - \mathbf{r} + \mathbf{t}\|,\ \|\cdot\| = \|\cdot\|_1 \, or \, \|\cdot\|_2^2$$

3.2.2 模型训练 [1]

TransE 模型通过机器学习进行训练，定义实体集 (Entity Set)，E，关系集 (Relation Set)，L，和一个训练组 (Training Set)，$S = \{(h,l,t) \,|\, h,t \in E, l \in L\}$。对于每个训练集里的正确三元组 $(h,l,t) \in S$，我们通过替换三元组中的头或尾来创建一组对应的“错误”三元组，即 $(h',l,t) \notin S$，或 $(h,l,t') \notin S$，$h',t' \in E$。对于每一对这样的三元组，我们定义评估距离为 $d(\mathbf{h}+\mathbf{l},\mathbf{t})$, 和 $d(\mathbf{h}'+\mathbf{l},\mathbf{t}')$，并且希望前者尽量小，后者尽量大。距离函数 d 既可以选择 L_1-norm 也可以选择 L_2-norm。这样对于整个训练集，我们可以得到损失函数（Loss Function):

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

Algorithm 1 Learning TransE

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

initialize $\boldsymbol{\ell} \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$

  $\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} / \|\boldsymbol{\ell}\|$ for each $\ell \in L$

  $\mathbf{e} \leftarrow$ uniform$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$

loop

  $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$

  $S_{batch} \leftarrow$sample(S, b) // sample a minibatch of size b

  $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets

  for $(h, \ell, t) \in S_{batch}$ **do**

    $(h', \ell, t') \leftarrow$sample$(S'_{(h,\ell,t)})$ // sample a corrupted triplet

    $T_{batch} \leftarrow T_{batch} \cup \left\{\left((h, \ell, t), (h', \ell, t')\right)\right\}$

  end for

  Update embeddings w.r.t. $\sum_{\left((h,\ell,t),(h',\ell,t')\right) \in T_{batch}} \nabla\left[\gamma + d(\boldsymbol{h} + \boldsymbol{\ell}, \boldsymbol{t}) - d(\boldsymbol{h'} + \boldsymbol{\ell}, \boldsymbol{t'})\right]_+$

end loop

图 1: TransE 训练算法

其中 $S' = \{(h', l, t) \,|\, h' \in E\} \cup \{h, l, t' \,|\, t' \in E\}$ 为每个三元组对应的“错误”三元组，γ 为误差参数。为了训练整个模型，我们采用**随机梯度下降法** (Stochastic Gradient Descent), 对于

$$\sum_{((h,l,t),(h',l,t')\in T} \nabla[\gamma + d(\mathbf{h} + \mathbf{l}, \mathbf{t}) - d(\mathbf{h}' + \mathbf{l}, \mathbf{t}')]_+$$

进行模型训练。其中集合 T 包含每一对正确和“错误”的三元组。图 1 中包含了 [1] 中给出的具体训练算法。

3.2.3 评估

对于 TransE 模型的评估，我们主要运用两个评价指标：**平均排位** (Meanrank)(越小越好) 以及 **hits@10**(越大越好)。对于训练好的 embedding 和一个测试集，对每个测试集中的三元组 (h, l, t)，计算 $\mathbf{h} + \mathbf{l}$
(或 $\mathbf{t} - \mathbf{l}$)，并将结果与 $\mathbf{t}$ ($\mathbf{h}$) 进行比较，计算出 $\mathbf{h} + \mathbf{l}$ ($\mathbf{t} - \mathbf{l}$) 在所有测试集中的尾实体 (头实体) 由小到大排序后的排位 (rank)，最后算出平均排位 (meanrank) 以及排位在前 10 所占的百分比 (hits@10)。这两个评估方法也将被用于后续模型的评估以方便进行模型之间的比较。在 [1] 中，对于 WordNet 数据集 (包含多于 40k 实体)，TransE 得到了 251 的平均排位和 89.2% 的 hits@10。图 2 中包含了 [1] 中对多种知识数据库中 TransE 对比其他模型的表现。

3.2.4 优势与劣势

TransE 作为转移距离模型的始作俑者，其拥有最简便的算法复杂度以及更少的变量，以至于使得建模更加效率，尤其对于 1 对 1 的三元组知识数据库。并且其模型简单的特性也使其容易拓展，演变出针对某些情况的特殊转移距离模型 (比如后续的 TransH 和 TransR)。

由于 TransE 评估距离的定义，TransE 并不在处理 1 对多，多对 1，和多对多的知识数据库中占有优势。例如在一对多，预测尾实体的例子中 (Matt Damon, is-acter-of, Good Will Hunting) 和 (Matt Damon, is-actor-of, Downsizing)，TransE 的距离评估指标无法区分 Good Will Hunting 和 Downsizing

DATASET	WN				FB15K				FBIM	
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**

图 2: TransE 评估结果

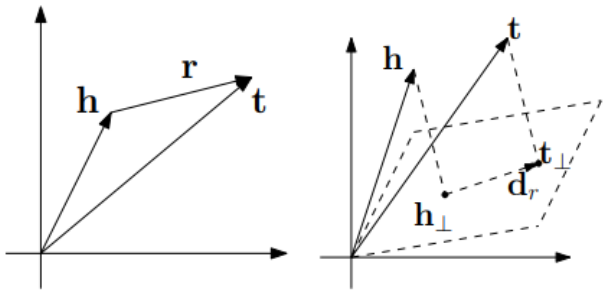
这两个个体，使得这两个个体的映射向量非常接近，并没有考虑他们的相似性和语义信息。

3.3 TransH

3.3.1 基本概念

为了解决 TransE 在一对多，多对一，多对多等情况中无法区分实体语义和准确预测多实体一侧的问题，TransH(**Trans**lating on **H**yperplanes) 模型提出了利用将实体向量投影到**超平面** (Hyperplane) 来解决这个问题并且尽量维持 TransE 的复杂度 [5]。在 TransH 中，对于一个三元组 (h, r, t) 中的关系部分 r，TransH 应用了一个向量 $\mathbf{d}_r$ 和一个超平面向量 $\mathbf{w}_r$, $\|\mathbf{w}_r\| = 1$ 来表示。这样对于一个正确的三元组 (h, r, t)，我们希望 $\mathbf{h}$ 在 $\mathbf{w}_r$ 上的投影，$\mathbf{h}_\perp$，加上 $\mathbf{d}_r$ 尽可能接近 $\mathbf{t}$ 在 $\mathbf{w}_r$ 上的投影，$\mathbf{t}_\perp$。这就可以解决在 3.2.4 中 TransE 遇到的问题，因为多个不同的尾实体 $\mathbf{t}$ 可以在 $\mathbf{w}_r$ 上得到相同的投影 $\mathbf{t}_\perp$。这样我们可以得到 TransH 的得分函数 [5] 为：

$$f_r(h,t) = \|\mathbf{h}_\perp + \mathbf{d}_r - \mathbf{t}_\perp\|_2^2,\ \mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r,\ \mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r$$



(a) TransE (b) TransH

图 3: TransE 和 TransH 的对比图示

3.3.2 替换正确三元组

在训练 TransH 模型的过程中，也需要对于每一个正确三元组来匹配一个“错误”三元组。[5] 中定义 Δ 为正确三元组的集合，Δ' 为“错误”三元组的集合。TransH 在定义“错误”三元组时和 TransE 有一些区别：在 TransE 中，对于每一个正确三元组，我们有相同的概率去替换头或尾实体；然而在 TransH 中，选择头或尾去替换的概率会根据知识数据库的类型被优化。例如在一对多的情况下，头实体有更大的概率被替换；在多对一的情况下，尾实体有更大的概率被替换。具体来说，在整个数据集中，定义平均尾实体数/头实体数为 tph，平均头实体数/尾实体数为 hpt。那么对于每一个正确三元组 (h, r, t)，我们有 $\frac{tph}{tph+hpt}$ 的概率替换头实体，$\frac{hpt}{tph+hpt}$ 的概率替换尾实体。

3.3.3 模型训练 [5]

TransH 的训练方法与 TransE 相似，主要的区别在于得分函数和损失函数的不同以及更多变量产生的更多限制 (constraints)。对于一个训练集 Δ, 损失函数为：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r',t')\in\Delta'_{(h,r,t)}} [f_r(h,t) + \gamma - f_{r'}(h',t')]_+$$

当最小化损失函数时，我们需要考虑以下限制：

$$\begin{aligned} &\forall\, \mathbf{e} \in E, \|\mathbf{e}\|_2 \leqslant 1 \\ &\forall\, r \in R, |\mathbf{w}_r^\top \mathbf{d}_r| / \|\mathbf{d}_r\|_2 \leqslant \varepsilon \\ &\forall\, r \in R, \|\mathbf{w}_r\|_2 = 1 \end{aligned}$$

那么结合这些限制，我们最终需要最小化的损失函数为：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r',t')\in\Delta'_{(h,r,t)}} [f_r(h,t) + \gamma - f_{r'}(h',t')]_+ + C\left\{\sum_{\mathbf{e}\in E}[\|\mathbf{e}\|_2^2 - 1]_+ + \sum_{r\in R}\left[\frac{(\mathbf{w}_r^\top \mathbf{d}_r)^2}{\|\mathbf{d}_r\|_2^2} - \varepsilon^2\right]_+\right\}$$

同样我们采取随机梯度下降法计算损失函数的梯度并训练模型。

3.3.4 评估与结论

[5] 中同样采用平均排位和 hits@10 两种指标来评估模型，在图 4 中，TransH 在 FB15k 中的表现要优于 TransE, 然而 TransE 在 WN11 中的表现要优于 TransH。对于数据关系相对复杂的 FB15k，TransH 对于处理一对多, 多对一的优势显现了出来，然而在结构较为单一的数据集中，TransE 的表现依旧较好。

Dataset	WN18				FB15k			
Metric	MEAN		HITS@10		MEAN		HITS@10	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
Unstructured (Bordes et al. 2012)	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL (Nickel, Tresp, and Kriegel 2011)	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE (Bordes et al. 2011)	1,011	985	68.5	80.5	273	162	28.8	39.8
SME (Linear) (Bordes et al. 2012)	545	533	65.1	74.1	274	154	30.7	40.8
SME (Bilinear) (Bordes et al. 2012)	526	509	54.7	61.3	284	158	31.3	41.3
LFM (Jenatton et al. 2012)	469	456	71.4	81.6	283	164	26.0	33.1
TransE (Bordes et al. 2013b)	263	**251**	75.4	**89.2**	243	125	34.9	47.1
TransH (unif.)	318	**303**	75.4	**86.7**	211	**84**	42.5	**58.5**
TransH (bern.)	400.8	388	73.0	82.3	212	**87**	45.7	**64.4**

图 4: TransH 在链接预测中的表现

在图 5 中，我们可以看到在链接预测中，TransH 的表现要好于 TransE，并且在一对多，多对一的数据中有明显提高。在比较等概率替换头尾 (unif.) 和优化过的替换法 (bern.) 时，优化替换头尾的概率在大部分情况下会提高模型预测的准确性。

综上，即便 TransH 在较为简单的数据集中的表现稍逊于算法更简单的 TransE，但在整体准确性上，TransH 相较于 TransE 有明显的提升，且算法的复杂度并没有很大的提高。

Task	Predicting left (HITS@10)				Predicting right (HITS@10)			
Relation Category	1-to-1	1-to-n	n-to-1	n-to-n	1-to-1	1-to-n	n-to-1	n-to-n
Unstructured (Bordes et al. 2012)	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE (Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (Linear) (Bordes et al. 2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (Bilinear) (Bordes et al. 2012)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE (Bordes et al. 2013b)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif.)	**66.7**	**81.7**	**30.2**	**57.4**	**63.7**	**30.1**	**83.2**	**60.8**
TransH (bern.)	**66.8**	**87.6**	**28.7**	**64.5**	**65.5**	**39.8**	**83.3**	**67.2**

图 5: TransE 和 TransH 在 FB15k 中不同关系形式中的对比

3.4 TransR

3.4.1 基本原理

在 [6] 中，作者首先提到了 TransE 和 TransH 模型都将实体和关系映射到同一维度的向量空间中，即 $\mathbb{R}^k$ 中。然而统一的映射方法并没有考虑到实体在不同关系下展现的不同属性。例如对于同一组“电影”实体，在“拍摄国家”和“上映时间”的两种不同关系下会展现出不同的关系。比如对于“肖申克的救赎”和“绿皮书”这两个实体，他们的拍摄国家均为美国，然而前者的上映时间是 1994 年，后者的上映时间是 2019 年。但是 TransE 和 TransH 并不能区分他们在不同关系下展现的不同属性，因为这两个实体在“拍摄国家”这个关系中应该更加靠近并在“上映时间”这个关系中更加远离，TransE 和 TransH 并没有额外的空间去满足这一需求。TransR(**Trans**lation in Corresponding **R**elation Space) 为了解决这一问题，提出对于每一个关系 r 定义一个**线性变换矩阵** M_r 然后将头尾实体变换到对应的新向量空间进行比较。这样的线性变换既能处理 TransE 中一对多，多对一的问题，又可以解决上述实体具有基于关系而变换的复杂属性问题。

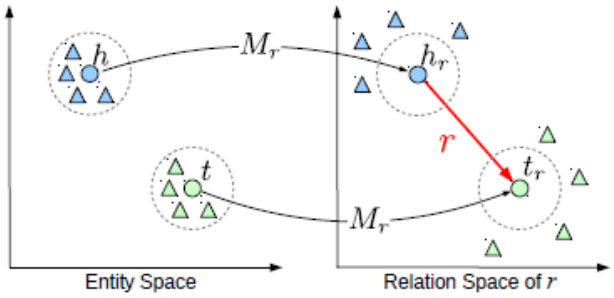


图 6: 在 TransR 中拆分的实体表示空间和关系表示空间

3.4.2 CTransR

基于 TransR[6]，作者又提到了关系本身的复杂性。例如在 (location, contains, location) 的三元组中，“contains”可以有不同的细化差别，并且在这些细化差别的关系中，实体也可能产生不同的语义或属性。例如国家-城市，国家-大学，大洲-国家等 [4]。为了处理这种情况，作者提出了 TransR 的衍生模型 CTransR(**C**luster Based **TransR**)。CTransR 先利用 TransE 获得所有实体和关系的映射向量，然后都每一个关系下的实体进行分组 (clusters)，根据 $\mathbf{h}-\mathbf{t}$ (其中 $\mathbf{h}$ 和 $\mathbf{t}$ 是通过 TransE 获得的向量) 进行对于特定关系的学习, 最后在每一个分组中得到一个关系向量 $\mathbf{r}_c$ 和对于整个关系的矩阵 $\mathbf{M}_r$。简而言之，在 TransR 中，经 $\mathbf{M}_r$ 变换后的空间中只有一个关系向量 $\mathbf{r}$，而在 CTransR 中，经 $\mathbf{M}_r$ 变换后

的空间有多个关系向量 $\mathbf{r}_c$，数量等同于当时的分组数量。

3.4.3 得分函数

在 TransR 中，对于每个三元组 (h, r, t), 首先定义他们的映射空间为 $\mathbf{h}, \mathbf{t} \in \mathbb{R}^k$, $\mathbf{r} \in \mathbb{R}^d$, 其中 $r \neq d$。我们需要额外训练 $\mathbf{M}_r \in \mathbb{R}^{r \times d}$，并且将 $\mathbf{h}$ 和 $\mathbf{r}$ 变换到新的向量空间。所以我们有 $\mathbf{h}_r = \mathbf{h}\mathbf{M}_r \in \mathbb{R}^d$ 和 $\mathbf{t}_r = \mathbf{t}\mathbf{M}_r \in \mathbb{R}^d$。那么我们得到关于 TransR 的得分函数为：

$$f_r(h, t) = \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_2^2$$

其中我们有以下限制：

$$\|\mathbf{h}\|_2 \leqslant 1, \|\mathbf{t}\|_2 \leqslant 1, \|\mathbf{r}\|_2 \leqslant 1, \|\mathbf{h}\mathbf{M}_r\|_2 \leqslant 1, \|\mathbf{t}\mathbf{M}_r\|_2 \leqslant 1$$

在 CTransR 中，对于一个关系 r, 在对应的 $\mathbb{R}^d$ 空间中可能有多个关系向量 $\mathbf{r}_c$, 所以我们定义 $\mathbf{h}_{r,c} = \mathbf{h}\mathbf{M}_r$ 和 $\mathbf{t}_{r,c} = \mathbf{t}\mathbf{M}_r$。与此同时，我们还需要控制 $\mathbf{r}_c$ 与 $\mathbf{r}$ 之间的距离，以保证由 $\mathbf{r}$ 而衍生出的 $\mathbf{r}_c$ 不会距离原先的 $\mathbf{r}$ 太远。所以 CTransR 的得分函数被定义为：

$$f_r(h, t) = \|\mathbf{h}_{r,c} + \mathbf{r}_c - \mathbf{t}_{r,c}\|_2^2 + \alpha\|\mathbf{r}_c - \mathbf{r}\|_2^2$$

其中新增的 $\alpha > 0$ 即为用于控制距离的参数，在 [6] 后续的训练中，α 通常在 $\{0.1, 0.01, 0.001\}$ 之中。

3.4.4 模型训练

TransR 也采用了匹配正确和错误三元组的方法来计算整个训练集的损失函数，唯一的不同点在于更多的参数和限制以及不同的得分函数。在替换头尾实体时，TransR 采用了 3.3.2 中提到的利用伯努利分布法根据关系形式的发布情况来决定替换头或尾的概率。TransR 的训练过程依然采用随机梯度下降法。下面是具体的训练集损失函数：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r,t')\in\Delta'} [f_r(h, t) + \gamma - f_{r'}(h', t')]_+$$

3.4.5 评估与结论

在图 7 中，我们可以看到 TransR 和 CTransR 在链接预测中的效果要好于 TransE 和 TransH。并且 CTransR 相比于 TransR 也有少幅度的优化。在图 8 中，我们也可以观察到 TransR 和 CTransR 在各种数据形式中的链接预测效果要好于其他两种转移距离模型。由于 TransR 应用了矩阵作为参数，训练的复杂程度较 TransE 和 TransH 相比有很大提升，并且训练过程中还涉及了矩阵之间的运算。[6] 中提到了训练 TransE 和 TransH 分别需要 5 和 30 分钟，然而 TransR 的训练需要大概 3 小时。

总结来说，TransR 在预测效果上在各方面都要好于比较基础的 TransE 和 TransR，然而更加准确的结果带来的是更多更复杂的参数以及更缓慢的训练速度。CTransR 也是第一次在转移距离模型中尝试的探索单一关系中可能产生的内部对于实体的影响，其对于模型准确度的提升也证明准确处理复杂关系对知识图谱的嵌入和建模是不可或缺的一部分。

Data Sets	WN18				FB15K			
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
Unstructured (Bordes et al. 2012)	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL (Nickel, Tresp, and Kriegel 2011)	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE (Bordes et al. 2011)	1,011	985	68.5	80.5	273	162	28.8	39.8
SME (linear) (Bordes et al. 2012)	545	533	65.1	74.1	274	154	30.7	40.8
SME (bilinear) (Bordes et al. 2012)	526	509	54.7	61.3	284	158	31.3	41.3
LFM (Jenatton et al. 2012)	469	456	71.4	81.6	283	164	26.0	33.1
TransE (Bordes et al. 2013)	263	251	75.4	89.2	243	125	34.9	47.1
TransH (unif) (Wang et al. 2014)	318	303	75.4	86.7	211	84	42.5	58.5
TransH (bern) (Wang et al. 2014)	401	388	73.0	82.3	212	87	45.7	64.4
TransR (unif)	232	219	78.3	91.7	226	78	43.8	65.5
TransR (bern)	238	225	**79.8**	92.0	**198**	77	48.2	68.7
CTransR (unif)	243	230	78.9	**92.3**	233	82	44	66.3
CTransR (bern)	**231**	**218**	79.4	**92.3**	199	**75**	**48.4**	**70.2**

图 7: TransR 在链接预测中与其他模型的对比

Tasks	Predicting Head(Hits@10)				Predicting Tail(Hits@10)			
Relation Category	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
Unstructured (Bordes et al. 2012)	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE (Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear) (Bordes et al. 2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (bilinear) (Bordes et al. 2012)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE (Bordes et al. 2013)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif) (Wang et al. 2014)	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransH (bern) (Wang et al. 2014)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (unif)	76.9	77.9	**38.1**	66.9	76.2	38.4	76.2	69.1
TransR (bern)	78.8	**89.2**	34.1	69.2	79.2	37.4	**90.4**	72.1
CTransR (unif)	78.6	77.8	36.4	68.0	77.4	37.8	78.0	70.3
CTransR (bern)	**81.5**	89.0	34.7	**71.2**	**80.8**	38.6	90.1	**73.8**

图 8: TransR 在 FB15k 中对于不同关系的链接预测表现

3.5 TransD

3.5.1 基本概念

根据 TransR 中一个比较明显的缺点，矩阵运算带来的缓慢训练速度以及大量额外的参数问题，TransD(**Trans**lation via **D**ynamic Mapping Matrix) 基于 TransR 提出了**双嵌入向量**的概念，并且提出了用一个与实体相关的向量以及一个与关系相关的向量通过外积计算，动态地得到关系投影矩阵的方法 [2]。在一个实体的两个嵌入向量中，其中一个用来表达实体 (或关系) 本身的语义，另一个用来获得关系投影矩阵。例如对于三元组 (h, r, t)，TransD 中需要通过训练得到的嵌入向量有

$$\mathbf{h}, \mathbf{h}_p, \mathbf{t}, \mathbf{t}_p \in \mathbb{R}^n, \ \mathbf{r}, \mathbf{r}_p \in \mathbb{R}^m$$

其中关系投影矩阵不需要通过额外的训练得到，而是被定义为：

$$\mathbf{M}_{rh} = \mathbf{r}_p \mathbf{h}_p^\top + I^{m \times n}$$
$$\mathbf{M}_{rt} = \mathbf{r}_p \mathbf{t}_p^\top + I^{m \times n}$$

这样通过向量外积来定义关系投影矩阵的方法避免了像在 TransR 中的矩阵与向量的乘积运算，大幅提高了模型的训练效率。

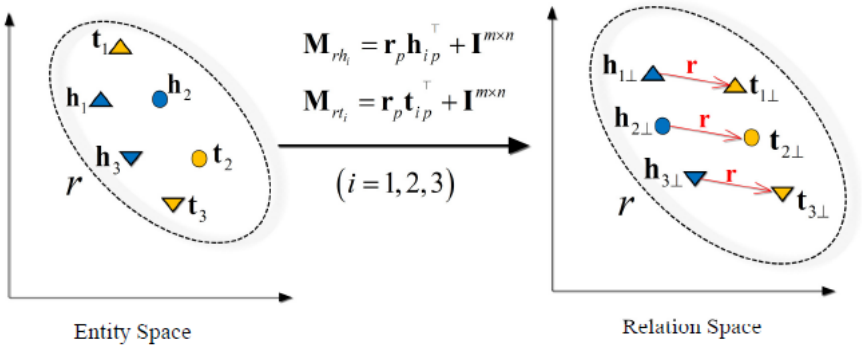


图 9: TransD 的实体和关系空间 [10]

3.5.2 得分函数与模型训练

根据关系投影矩阵，TransD 用与 TransR 相似的方法将实体空间中的向量投影到相应的关系空间，所以我们定义投影后的向量为：

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h},\ \mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t}$$

以及相应的得分函数为：

$$f_r(\mathbf{h},\mathbf{t}) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2$$

并在训练中的限制有：

$$\|\mathbf{h}\|_2 \leqslant 1, \|\mathbf{t}\|_2 \leqslant 1, \|\mathbf{r}\|_2 \leqslant 1, \|\mathbf{h}_\perp\|_2 \leqslant 1, \|\mathbf{t}_\perp\|_2 \leqslant 1$$

TransD 同样采用匹配正确和错误三元组的方法来计算最后的损失函数。TransD 运用了相同概率替换头尾实体 (uniform distribution) 和不同概率替换头尾实体 (bernoulli distribution)。最后对于训练集 Δ, 我们有：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r,t')\in\Delta'} [\gamma - f_r(\mathbf{h},\mathbf{t}) + f_{r'}(\mathbf{h}',\mathbf{t}')]_+$$

3.5.3 与其他转移距离模型的比较

TransE 是 TransD 的一种特殊情况，当 $m = n$ 时，并当所有投影向量为 $\mathbf{0}$ 时，TransD 等同于 TransE，因为 TransE 并没有考虑在不同关系下实体产生的多样性和实体本身可能的多样性。
TransH 也与 TransD 相似，因为在 $m = n$ 时，TransD 中投影后的向量为：

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h} = \mathbf{h} + \mathbf{h}_p^\top \mathbf{h}\mathbf{r}_p$$
$$\mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t} = \mathbf{t} + \mathbf{t}_p^\top \mathbf{t}\mathbf{r}_p$$

这时与 TransH 的不同是 TransH 的投影后的向量只与关系向量有关，而 TransD 的投影后的向量与关系和实体向量都有关。
关于 TransR/CTransR，TransD 是基于他们的优化模型。TransD 通过对于每个实体和关系定义两个嵌入向量并通过他们定义了两个关系投影矩阵来避免训练过程中的矩阵和向量的乘法运算，从而提高

了运算效率。如果将 TransD 写为 TrnasR 中的形式，并假设 $m \geqslant n$ (WLOG)，我们会得到：

$$\mathbf{h}_{\perp} = \mathbf{M}_{rh}\mathbf{h} = \mathbf{h} + \mathbf{h}_p^{\top}\mathbf{h}\mathbf{r}_p + \left[\mathbf{h}^{\top}, \mathbf{0}^{\top}\right]^{\top}$$
$$\mathbf{t}_{\perp} = \mathbf{M}_{rt}\mathbf{t} = \mathbf{t} + \mathbf{t}_p^{\top}\mathbf{t}\mathbf{r}_p + \left[\mathbf{t}^{\top}, \mathbf{0}^{\top}\right]^{\top}$$

这样的运算方法使 TransD 训练速度更快，并可以被应用到更大的知识图谱中 [10]。

3.5.4 链接预测结果

尽管 TransD 通过动态获取的关系投影矩阵减少了运算复杂度以及参数的数量，TransD 在链接预测的任务上的表现并没有比 TransR 更差。反而通过对全局特征 (实体和关系) 的捕捉使得最后的结果普遍好于 TransR[2]。由图 10 和图 11 中可见，在实体和关系都更为复杂的 FB15k 数据集中，TransD 对于链接的预测较 TransR 有明显的提升，尤其在一对多预测尾实体和多对一预测头实体中，由于都考虑到了实体和关系的多样性，以至于有了更好的准确度。

Data sets	WN18				FB15K			
Metric	Mean Rank		Hits@10		Mean Rank		Hits@10	
	Raw	Filt	Raw	Filt	Raw	Filt	Raw	Filt
Unstructured (Bordes et al. 2012)	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL (Nickle, Tresp, and Kriegel 2011)	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE (Bordes et al. 2011)	1,011	985	68.5	80.5	273	162	28.8	39.8
SME (linear) (Bordes et al.2012)	545	533	65.1	74.1	274	154	30.7	40.8
SME (Bilinear) (Bordes et al. 2012)	526	509	54.7	61.3	284	158	31.3	41.3
LFM (Jenatton et al. 2012)	469	456	71.4	81.6	283	164	26.0	33.1
TransE (Bordes et al. 2013)	263	251	75.4	89.2	243	125	34.9	47.1
TransH (unif) (Wang et al. 2014)	318	303	75.4	86.7	211	84	42.5	58.5
TransH (bern) (Wang et al. 2014)	401	388	73.0	82.3	212	87	45.7	64.4
TransR (unif) (Lin et al. 2015)	232	219	78.3	91.7	226	78	43.8	65.5
TransR (bern) (Lin et al. 2015)	238	225	**79.8**	92.0	198	77	48.2	68.7
CTransR (unif) (Lin et al. 2015)	243	230	78.9	92.3	233	82	44.0	66.3
CTransR (bern) (Lin et al. 2015)	231	218	79.4	92.3	199	75	48.4	70.2
TransD (unif)	242	229	79.2	**92.5**	211	**67**	**49.4**	**74.2**
TransD (bern)	**224**	**212**	79.6	92.2	**194**	91	**53.4**	**77.3**

图 10: TransD 的链接预测结果

Tasks	Prediction Head (Hits@10)				Prediction Tail (Hits@10)			
Relation Category	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
Unstructured (Bordes et al. 2012)	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE (Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear) (Bordes et al.2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (Bilinear) (Bordes et al. 2012)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE (Bordes et al. 2013)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif) (Wang et al. 2014)	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransH (bern) (Wang et al. 2014)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (unif) (Lin et al. 2015)	76.9	77.9	38.1	66.9	76.2	38.4	76.2	69.1
TransR (bern) (Lin et al. 2015)	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
CTransR (unif) (Lin et al. 2015)	78.6	77.8	36.4	68.0	77.4	37.8	78.0	70.3
CTransR (bern) (Lin et al. 2015)	81.5	89.0	34.7	71.2	80.8	38.6	90.1	73.8
TransD (unif)	80.7	85.8	**47.1**	**75.6**	80.0	**54.5**	80.7	**77.9**
TransD (bern)	**86.1**	**95.5**	**39.8**	**78.5**	**85.4**	**50.6**	**94.4**	**81.2**

图 11: TransD 在 FB15k 中的链接预测

4 其他距离模型

4.1 概述

在了解了基本的转移距离模型后，在知识图谱嵌入领域还有一些其他基于定义实体之间距离的模型。并且转移距离模型的理念是基于下列部分模型的理念而产生的，也包含了即 TransD 之后更近期的研究成果，如 TransMS。这一章将简要介绍一下这些距离模型，包括基本理念和训练中的得分函数和结论，其中一部分模型的结论可以在第 3 章节的链接预测结果图中和本章节末的图中找到。

4.2 非结构化模型 (Unstructured Model)

非结构化模型忽略了关系的向量嵌入，只对实体进行嵌入。非结构化模型属于简化版的 TransE，即令所有关系向量 $\mathbf{r} = \mathbf{0}$，并没有考虑不同关系之间的区别 [10]，其得分函数为：

$$f_r(h,t) = \|\mathbf{h} - \mathbf{t}\|_2^2$$

4.3 结构化模型 (SE)

结构化模型 (**S**tructured **E**mbedding) 对于每一对头尾实体运用两个与关系有关的矩阵，$\mathbf{M}_{r,1}, \mathbf{M}_{r,2}$，并通过这两个矩阵分别对首尾向量进行投影。由于该模型需要对两个矩阵分别进行训练，所以不能精确捕捉头尾实体之间的关系 [6]。其得分函数为两个投影后向量的 L_1-norm：

$$f_r(h,t) = \|\mathbf{M}_{r,1}\mathbf{h} - \mathbf{M}_{r,2}\mathbf{t}\|_1$$

4.4 TransMS

在 [11] 中，作者提到在复杂的知识数据中，三元组不仅仅代表了主谓宾的关系，而有时首尾之间也会有语义关系。TransMS (**Trans**lates and Transmits **M**ultidirectional **S**emantics) 不仅从关系中传递语义信息，也尝试了在首尾实体之间传递语义信息，并使用了非线性函数来增加模型的准确性。

TransMS 也应用了向量嵌入对三元组的实体进行嵌入，并在定义得分函数时考虑到了三者之间的语义联系。即包含语义信息的实体向量被定义为：

$$\begin{aligned}\mathbf{h}_\perp &= \tanh(-\mathbf{t} \otimes \mathbf{r}) \otimes \mathbf{h} \\ &= -\tanh(\mathbf{t} \otimes \mathbf{r}) \otimes \mathbf{h} \\ \mathbf{t}_\perp &= \tanh(\mathbf{h} \otimes \mathbf{r}) \otimes \mathbf{t}\end{aligned}$$

在嵌入关系向量中，TransMS 增加了一个误差参数 $\alpha \in \mathbb{R}$，即关系向量为：

$$\mathbf{r}_\perp = \mathbf{r} + \alpha(\mathbf{h} \otimes \mathbf{t})$$

上述定义中 $\otimes$ 为 Hadamard Product，即矩阵对应元素之间的乘积 (entry-wise product)，或表达为 $c_{ij} = a_{ij}b_{ij}$。最后的得分函数为：

$$\begin{aligned}f_r(\mathbf{h},\mathbf{t}) &= \|\mathbf{h}_\perp + \mathbf{r}_\perp - \mathbf{t}_\perp\|_{l_{1/2}} \\ &= \|-\tanh(\mathbf{t} \otimes \mathbf{r}) \otimes \mathbf{h} + \mathbf{r} + \alpha(\mathbf{h} \otimes \mathbf{t}) - \tanh(\mathbf{h} \otimes \mathbf{r}) \otimes \mathbf{t}\|_{l_{1/2}}\end{aligned}$$

在训练中，TransMS 使用了在 TransE 中提出的损失函数：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r,t')\in\Delta'} [f_r(h,t) + \gamma - f_{r'}(h',t')]_+$$

训练的结果在 FB15k 中的结果如下图，可见 TransMS 在复杂关系中对于首尾实体的预测效果相较于其他模型有明显的提升。

Tasks	Head Prediction (Hit@10)				Tail Prediction (Hit@10)			
Relation Type	**1-to-1**	**1-to-N**	**N-to-1**	**N-to-N**	**1-to-1**	**1-to-N**	**N-to-1**	**N-to-N**
SE [Bordes *et al.*, 2011]	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear/bilinear) [Bordes *et al.*, 2012]	35.1/30.9	53.7/69.6	19.0/19.9	40.3/38.6	32.7/28.2	14.9/13.1	61.6/76.0	43.3/41.8
TransE [Bordes *et al.*, 2013]	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif/bern) [Wang *et al.*, 2014]	66.7/66.8	81.7/87.6	30.2/28.7	57.4/64.5	63.7/65.5	30.1/39.8	83.2/83.3	60.8/67.2
TransR (unif/bern) [Lin *et al.*, 2015]	76.9/78.8	77.9/89.2	38.1/34.1	66.9/69.2	76.2/79.2	38.4/37.4	76.2/90.4	69.1/72.1
CTransR (unif/bern) [Lin *et al.*, 2015]	78.6/81.5	77.8/89.0	36.4/34.7	68/71.2	77.4/80.8	37.8/38.6	78.0/90.1	70.3/73.8
TransD (unif/bern) [Ji *et al.*, 2015]	80.7/86.1	85.8/95.5	47.1/39.9	75.6/78.5	80.0/85.4	54.5/50.4	80.7/94.4	77.9/81.2
TranSparse (separate,US,unif/bern) [Ji *et al.*, 2016]	83.2/87.1	85.2/ 95.8	51.4/44.4	80.3/81.2	82.6/87.5	60.0/57.0	85.5/**94.5**	82.5/83.7
GTrans_DW (unif/bern) [Tan *et al.*, 2018]	77.4/76.3	87.1/86.0	23.4/20.3	65.6/61.2	74.3/75.6	32.3/31.7	84.4/83.5	67.2/63.9
GTrans_SW (unif/bern) [Tan *et al.*, 2018]	80.1/84.9	93.0/95.0	48.4/39.9	75.4/75.9	79.4/84.4	51.8/47.7	91.2/**94.5**	77.8/78.8
TransMS (unif/bern)	**89.5/91.4**	94.4/**95.9**	**78.5**/44.9	**85.6**/78.5	**90.0/91.6**	**84.8**/54.1	91.7/93.6	**87.7**/82.0

图 12: TransMS 的链接预测结果 [11]

4.5 ManifoldE

[12] 介绍了 ManifoldE(A **Manifold**-Based **E**mbedding Model)，利用几何学和流形来定义三元组的准确度。而根据流形的不同，展现出不同的模型形式。这种模型并没有像之前的模型类似于 $\mathbf{h}+\mathbf{r}=\mathbf{t}$ 的准确要求，而是要求正确的三元组只需要位于一个流形中，而不是一个准确的点。这种改变解决了原有准确链接预测的一些问题，例如 overfitting，并提升了准确度和运算效率。

在 ManifoldE 中，如果给定一个头实体向量和关系向量，那么尾实体向量应该位于一个流形之中，所以得分函数被定义为三元组与流形之间的距离：

$$f_r(h,t) = \|\mathcal{M}(h,r,t) - D_r^2\|_2^2$$

其中 D_r 是流形参数，$\mathcal{M}:\mathbb{E}\times\mathbb{L}\times\mathbb{E}\to\mathbb{R}$ 是流形函数，$\mathbb{E},\mathbb{L}$ 分别为实体和关系集。

当流形为球体时，我们有:

$$\mathcal{M}(h,r,t) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_2^2$$

可以看到如果 $D_r=0$ 时，该模型同等于 TransE 模型，即流形为一个点。如果利用再生核希尔伯特空间 (RKHS, Reproducing Kernel Hilbert Space) 中的球体来表示流形的话，我们有：

$$\begin{aligned}\mathcal{M}(h,r,t) &= \|\phi(h)+\phi(r)-\phi(t)\|^2 \\ &= \mathbf{K}(h,h)+\mathbf{K}(t,t)+\mathbf{K}(r,r)-2\mathbf{K}(h,t)-2\mathbf{K}(r,t)+2\mathbf{K}(r,h)\end{aligned}$$

其中 ϕ 为从原空间到希尔伯特空间的函数，通常 $\mathbf{K}$ 会有如下形式：

$$\begin{aligned}\mathbf{K}(a,b) &= \mathbf{a}^\top\mathbf{b} \quad \text{(Linear Kernel)} \\ \mathbf{K}(a,b) &= e^{-\frac{\|\mathbf{a}-\mathbf{b}\|_2^2}{\sigma^2}} \quad \text{(Gaussian Kernel)} \\ \mathbf{K}(a,b) &= (\mathbf{a}^\top\mathbf{b}+d)^p \quad \text{(Polynomial Kernel)}\end{aligned}$$

除了球体，流形也可以是一个超平面。[12] 中提到，当两个流形没有交集时，嵌入向量会损失一部

分语义信息，然而超平面只有在完全平行时才没有交集。所以在超平面中，我们对关系进行对头尾实体分别嵌入向量 $\mathbf{r}_h, \mathbf{r}_t$，并有：

$$\mathcal{M}(h,r,t) = (\mathbf{h}+\mathbf{r}_h)^\top(\mathbf{t}+\mathbf{r}_t)$$

例如给定头实体向量和关系向量，正确三元组的尾实体向量应落在以 $\mathbf{h}+\mathbf{r}_h$ 方向的超平面中。在超平面流形中，文章中也提到了使用绝对值进行训练，即：

$$\mathcal{M}(h,r,t) = |\mathbf{h}+\mathbf{r}_h|^\top|\mathbf{t}+\mathbf{r}_t|$$

使用绝对值会使得到的尾实体预测数量翻倍，也就是对每次预测会匹配两个相应的尾实体，而不是原先的一个，这样做会增加模型嵌入的灵活性。此外，在超平面设定中也可以将原空间投射到希尔伯特空间中，即：

$$\mathcal{M}(h,r,t) = \mathbf{K}(\mathbf{h}+\mathbf{r}_h, \mathbf{t}+\mathbf{r}_t)$$

在模型训练中，ManifoldE 仍然采用了匹配正负三元组的方法：

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta}\sum_{(h',r,t')\in\Delta'}[f_r(h,t)+\gamma-f_{r'}(h',t')]_+$$

并且文中提到了 ManifoldE 的计算复杂度与 TransE 相比仅相差一个一次的常数积，表示为:

$$O(\text{ManifoldE}) = O(\lambda \times O(\text{TransE}))$$

下图为 ManifoldE 在 FB15k 中链接预测的结果，分为使用球体和超平面作为流形的结果：

Tasks	Predicting Head(HITS@10)				Predicting Tail(HITS@10)			
Relation Category	1-1	1-N	N-1	N-N	1-1	1-N	N-1	N-N
TransE [Bordes *et al.*, 2013]	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH [Wang *et al.*, 2014]	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR [Lin *et al.*, 2015b]	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
PTransE [Lin *et al.*, 2015a]	90.1	92.0	58.7	86.1	90.1	70.7	87.5	88.7
KG2E [He *et al.*, 2015]	92.3	93.7	**66.0**	69.6	92.6	67.9	94.4	73.4
ManifoldE Sphere	92.4	95.5	53.2	84.4	92.8	68.7	94.8	86.8
ManifoldE Hyperplane	**92.5**	**97.0**	59.9	**88.2**	**93.4**	**70.8**	**94.9**	**90.0**

图 13: ManifoldE 的链接预测结果 [12]

4.6 TransF

在 [13] 中，作者从 TransR 的不足之处展开讨论，提到了 TransR 对于每一个关系分别进行映射的方式忽略了关系和关系之间的潜在结构关系，例如在“国籍是”和“出生地是”这两个关系中，前者的信息通常会从后者的信息中推导出来，然而 TransR 直接将两种关系完全分开处理，忽略了关系与关系之间的潜在关系。TransF(**Trans**lation-Based Embedding Method with **F**actorized Projection Matrices) 模型基于 TransR 的理念，使用固定数量的矩阵基底 (matrix bases) 来定义关系映射矩阵。其中每一个关系矩阵都是其中一个基地来生成 (span) 的。在 TransF 中, 需要训练固定数量的矩阵 $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{d_e \times d_r}$，

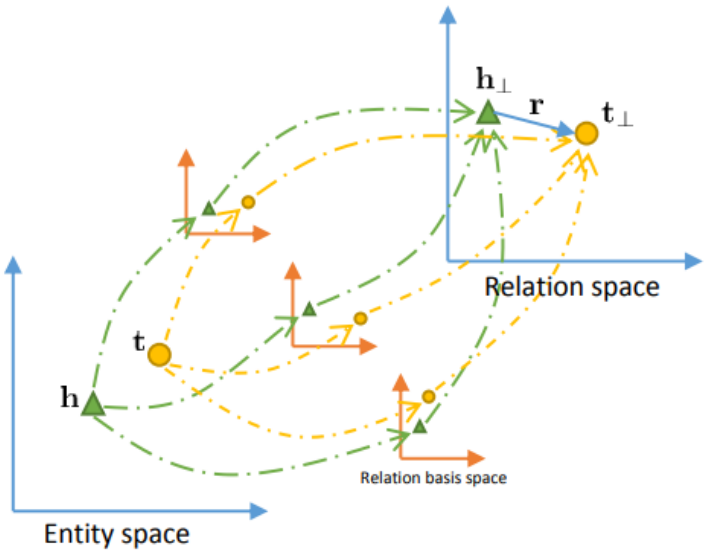


图 14: TransF 的实体与关系空间 [13]

然后组成相应的映射矩阵 $\mathbf{M}_{r,h}, \mathbf{M}_{r,t}$，具体为：

$$\mathbf{M}_{r,h} = \sum_{i=1}^{s} \alpha_r^{(i)} \mathbf{U}^{(i)} + \mathbf{I}$$

$$\mathbf{M}_{r,t} = \sum_{i=1}^{s} \beta_r^{(i)} \mathbf{V}^{(i)} + \mathbf{I}$$

其中 $s \in \mathbb{R}$ 为参数数量，$\mathbf{I}$ 为单位矩阵。通过这两个映射矩阵，我们得到：

$$\mathbf{h}_\perp = \mathbf{M}_{r,h}\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_{r,t}\mathbf{t}$$

从而得到的得分函数为：

$$f_r(h,t) = \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_{l_{1/2}}$$

下图为 TrnasF 在 FB15k 中的链接预测结果：

Hits@10 RESULTS ON FB15K ARRANGED BY DIFFERENT RELATION TYPES.

Method	Head Entity Prediction (HEP)				Tail Entity Prediction (TEP)			
	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
TransE	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR	78.8	89.2	24.1	69.2	79.2	37.4	90.4	72.1
CTransR	81.5	89.0	34.7	71.2	80.8	38.6	90.1	73.8
TransD	86.1	95.5	39.8	78.5	85.4	50.6	94.4	81.2
TransSparse (s)	86.8	95.5	44.3	80.9	86.6	56.6	94.4	83.3
TransSparse (us)	87.1	**95.8**	44.4	81.2	87.5	57.0	**94.5**	83.7
TransF	**88.1**	94.9	**53.2**	**82.8**	**88.8**	**62.1**	93.4	**85.8**

图 15: TransF 的链接预测结果 [13]

4.7 TransA[14]

TransA (**Trans**lating Using **A**daptive Approach for Knowledge Graph Embedding) 提出了对表示向量的自适应度量方法，并提出了一个更加灵活的嵌入方法 [15]。由于传统的得分函数只是考虑了向量之间的距离，如 $\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_2^2$，[14] 中提到这种过于简单的判断指标无法准确地对复杂的知识库建模：首先由于传统的方法会取距离最近的向量，这种方法缺乏灵活性，以至于无法准确处理复杂的知识数据库，例如一对多，多对一，和多对多的情况；其次，这种距离判断方法对每一个维度采用相同的指标，这样就更容易导致产生错误的实体与关系的匹配。基于这两个结论，TransA 摒弃了传统的欧几里得距离 (Euclidean Distance)，采用了马氏距离 (Mahalanobis Distance)。因此，在 TransA 中的得分函数为：

$$f_r(h,t)=(|\mathbf{h}+\mathbf{r}-\mathbf{t}|^{\top})\mathbf{W_r}(|\mathbf{h}+\mathbf{r}-\mathbf{t}|)$$

其中 $|\mathbf{h}+\mathbf{r}-\mathbf{t}|=(|h_1+r_1-t_1|,|h_2+r_2-t_2|,\cdots,|h_n+r_n-t_n|)$，$\mathbf{W_r}$ 是对于适应性指标的加权矩阵 (非负且对称矩阵)。其中一个与传统得分函数不同的是 TransA 中使用绝对值来衡量 $\mathbf{h}+\mathbf{r}$ 与 $\mathbf{t}$ 之间的损失。并且我们可以用科列斯基分解 (LDL Decomposition) 得到：

$$\mathbf{W_r}=\mathbf{L_r}^{\top}\mathbf{D_r}\mathbf{L_r}$$
$$f_r=(\mathbf{L_r}|\mathbf{h}+\mathbf{r}-\mathbf{t}|)^{\top}\mathbf{D_r}(\mathbf{L_r}|\mathbf{h}+\mathbf{r}-\mathbf{t}|)$$

其中 $\mathbf{L_r}$ 可以看作是变换矩阵 (下三角矩阵)，且 $\mathbf{D_r}=diag(\omega_1,\omega_2,\omega_3,\cdots)$ 是对角矩阵且每个映射维度会有 ω_i 的加权。由于在大部分情况下，关系向量只会收到部分数据的影响，也就是受部分维度的影响，而其他无关的维度在传统模型中会导致误差。并且利用马氏距离利用在椭球上的投影减少了错误的预测，在下图中可以看到在欧几里得距离投影的球形空间中有更多的错误预测 (红色点)

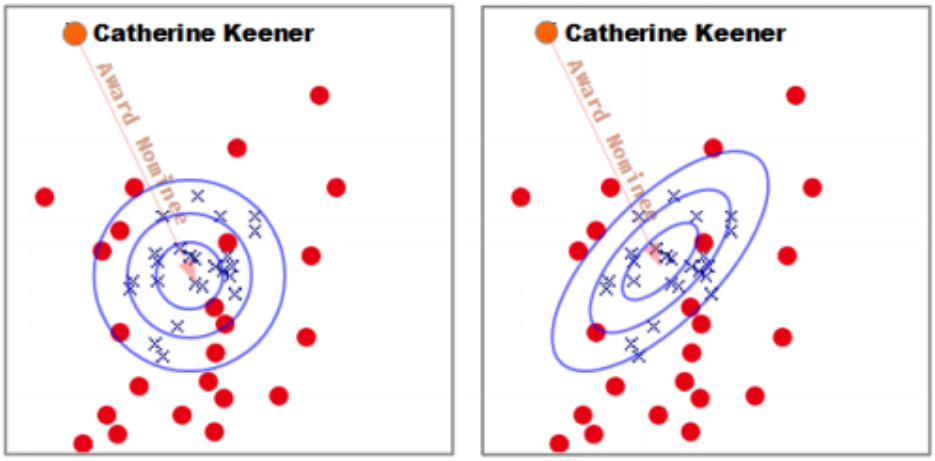


图 16: TransA 马氏距离模型与传统距离模型的投影区别 [14]

在模型训练中，目标函数被定义为

$$\min \sum_{(h,r,t)\in\Delta}\sum_{(h',r',t')\in\Delta'}[f_r(h,t)+\gamma-f_{r'}(h',t')]_+ +$$
$$\lambda\left(\sum_{r\in R}\|\mathbf{W_r}\|_F^2\right)+C\left(\sum_{e\in E}\|\mathbf{e}\|_2^2+\sum_{r\in R}\|\mathbf{r}\|_2^2\right)$$
$$s.t.\quad [\mathbf{W_r}]_{ij}\geqslant 0$$

其中 C 控制实体参数，λ 控制适应性加权矩阵，并且 $\mathbf{W_r}$ 可以通过直接求导数等于 0 得出，然后将负数项变为 0，即：

$$\begin{aligned}\mathbf{W_r} = & -\sum_{(h,r,t)\in\Delta}\left(|\mathbf{h}+\mathbf{r}-\mathbf{t}||\mathbf{h}+\mathbf{r}-\mathbf{t}|^\top\right)\\ & +\sum_{(h',r',t')\in\Delta'}\left(|\mathbf{h}'+\mathbf{r}'-\mathbf{t}'||\mathbf{h}'+\mathbf{r}'-\mathbf{t}'|^\top\right)\end{aligned}$$

由于加权矩阵可以通过直接计算得到，所以 TransA 的参数数量几乎等同于 TransE, 很大程度上提高了运算效率。下图是 TransA 在 FB15k 中进行链接预测的结果，可见其在一对多/多对一的预测准确度上有较明显的提升。

Tasks	**Predicting Head(HITS@10)**				**Predicting Tail(HITS@10)**			
Relation Category	1-1	1-N	N-1	N-N	1-1	1-N	N-1	N-N
SE(Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (Bordes et al. 2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
TransE (Bordes et al. 2013)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (Wang et al. 2014)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (Lin et al. 2015)	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
TransA	**86.8**	**95.4**	**42.7**	**77.8**	**86.7**	**54.3**	**94.4**	**80.6**

图 17: TransA 的链接预测结果 [14]

4.8 KG2E

4.8.1 概述

[17] 中指出传统的知识图谱表示方法都是运用一个指定的空间来嵌入信息，例如在 TransE 和 TransR 中利用向量空间中的一个点来表示实体和关系。[17] 指出这种方式总是对每个实体或关系进行统一的处理，并没有考虑到其中的不确定性。所以，KG2E 模型提出利用概率模型来表示知识图谱，即将每个实体和信息都用一个随机变量来表示和表达信息，在这里使用的是高阶的正态分布 (Multi-Dimensional Gaussian Distribution)，其中平均数 (Mean) 代表其位置，协方差矩阵 (Covariance Matrix) 代表可能存在的不确定性。

4.8.2 信息的不确定性

[17] 指出知识图谱中的不确定性可能存在于头尾实体信息的不平衡，模糊的关系，不平衡的多对多关系等。文中假设如果一个实体存在于更少的三元组中或一个关系连接了更多的三元组 (尤其在复杂语境中时)，他们会有更多的不确定性。

4.8.3 模型基本原理

在模型中，对于每个头实体，$\mathcal{H}$，尾实体，$\mathcal{T}$，和关系，$\mathcal{R}$，分别匹配一个高阶的正态分布 $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$。在 KG2E 中，对于一个三元组 (h,r,t)，和他们匹配的分布 $(\mathcal{H},\mathcal{R},\mathcal{T})$，如果 $\mathcal{H}-\mathcal{T}$ 的分布与 $\mathcal{R}$ 相似，那么相应三元组的得分会高。具体来说，KG2E 运用 KL 散度 (Kullback-Leibler Divergence) 和期望可能性 (Expected Likelihood) 两种方法来判断两个概率分布的差距。在图 18 的例子中，每个单词代表

实体或关系的位置，其中有下划线的是关系，周围的圈代表随机性。图中可以推断出希拉里·克林顿 (Hillary Clinton) 的出生地 (place of birth) 是芝加哥 (Chicago)，国籍 (nationality) 是美国 (USA)。

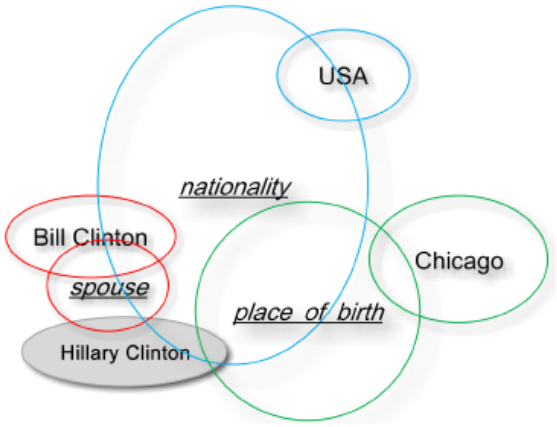


图 18: KG2E 中使用随机变量代表实体的实例 [17]

4.8.4 KL-Divergence

在利用 KL-Divergence 来优化得分函数时，通常假设三元组不具有对称性。对于三元组 (h, r, t)，定义下列概率分布：

$$\mathcal{H} \sim \mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h), \quad \mathcal{R} \sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r), \quad \mathcal{T} \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

在假设头实体和尾实体的概率在特定关系下处于独立关系时，我们有：

$$\mathcal{P}_e = \mathcal{H} - \mathcal{T} \sim \mathcal{N}(\boldsymbol{\mu}_h - \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_h + \boldsymbol{\Sigma}_t), \quad \mathcal{P}_r = \mathcal{R} \sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$$

根据 KL-Divergence 的定义，我们有：

$$\begin{aligned} \mathcal{E}(h, r, t) &= \mathcal{E}(\mathcal{P}_e, \mathcal{P}_r) = \mathcal{D}_{\mathcal{KL}}(\mathcal{P}_e, \mathcal{P}_r) \\ &= \int_{x \in \mathcal{R}^{k_e}} \mathcal{N}(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r) \log \frac{\mathcal{N}(x; \boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e)}{\mathcal{N}(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)} dx \\ &= \frac{1}{2} \left\{ tr(\boldsymbol{\Sigma}_r^{-1} \boldsymbol{\Sigma}_e) + (\boldsymbol{\mu}_r - \boldsymbol{\mu}_e)^\top \boldsymbol{\Sigma}_r^{-1} (\boldsymbol{\mu}_r - \boldsymbol{\mu}_e) - \log \frac{det(\boldsymbol{\Sigma}_e)}{det(\boldsymbol{\Sigma}_r)} - k_e \right\} \end{aligned}$$

我们还可以考虑三元组的对称性，这样的得分函数为：

$$\mathcal{E}(h, r, t) = \frac{1}{2} (\mathcal{D}_{\mathcal{KL}}(\mathcal{P}_e, \mathcal{P}_r) + \mathcal{D}_{\mathcal{KL}}(\mathcal{P}_r, \mathcal{P}_e))$$

4.8.5 Expected Likelihood

在 Expected Likelihood 的方法中，[17] 用两个随机变量的内积来决定 $\mathcal{P}_e$ 和 $\mathcal{P}_r$ 的相似性：

$$\begin{aligned} \mathcal{E}(\mathcal{P}_e, \mathcal{P}_r) &= \int_{x \in \mathcal{R}^{k_e}} \mathcal{N}(x; \boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e) \mathcal{N}(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r) dx \\ &= \mathcal{N}(0; \boldsymbol{\mu}_e - \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_e + \boldsymbol{\Sigma}_r) \end{aligned}$$

[17] 中额外使用 log 使得计算和对比更加方便，并将结果作为最终的得分函数：

$$\mathcal{E}(h,r,t) = \log \mathcal{E}(\mathcal{P}_e, \mathcal{P}_r) = \log \mathcal{N}(0; \boldsymbol{\mu}_e - \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_e + \boldsymbol{\Sigma}_r)$$
$$= \frac{1}{2}\left\{(\boldsymbol{\mu}_e - \boldsymbol{\mu}_r)^\top (\boldsymbol{\Sigma}_e + \boldsymbol{\Sigma}_r)^{-1}(\boldsymbol{\mu}_e - \boldsymbol{\mu}_r) + \log\det(\boldsymbol{\Sigma}_e + \boldsymbol{\Sigma}_r) + k_e \log(2\pi)\right\}$$

4.8.6 模型训练

对于 KG2E 的训练，[17] 依然采用了随机梯度下降法来实现，并使用 unif 和 bern 两种方法来替换正确三元组，得到的损失函数为:

$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r',t')\in\Delta'_{(h,r,t)}} [\mathcal{E}(h,r,t) + \gamma - \mathcal{E}(h',r',t')]_+$$

4.8.7 评估和结论

下图中包含了 KG2E 在 FB15k 中链接预测的结果，其中 KL 代表使用通过 KL-Divergence 得到的模型，EL 代表使用通过 Expected Likelihood 得到的模型。

Tasks	Prediction Head (Hits@10)				Prediction Tail (Hits@10)			
Relation Category	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
Unstructured (Bordes et al. 2012)	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE (Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear) (Bordes et al.2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (bilinear) (Bordes et al. 2012)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE (Bordes et al. 2013)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif) (Wang et al. 2014)	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransH (bern) (Wang et al. 2014)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (unif) (Lin et al. 2015)	76.9	77.9	38.1	66.9	76.2	38.4	76.2	69.1
TransR (bern) (Lin et al. 2015)	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
CTransR (unif) (Lin et al. 2015)	78.6	77.8	36.4	68.0	77.4	37.8	78.0	70.3
CTransR (bern) (Lin et al. 2015)	81.5	89.0	34.7	**71.2**	80.8	38.6	90.1	**73.8**
KG2E_EL (unif)	49.2	87.9	55.2	53.1	47.8	54.9	88.4	56.2
KG2E_EL (bern)	51.8	89.4	47.9	51.6	52.0	43.6	89.3	55.0
KG2E_KL (unif)	**92.2**	**93.7**	**67.4**	69.1	**91.2**	**69.7**	**93.6**	71.2
KG2E_KL (bern)	**92.3**	**94.6**	**66.0**	**69.6**	**92.6**	**67.9**	**94.4**	**73.4**

图 19: KG2E 在 FB15k 中的链接预测结果 [17]

5 语义模型

5.1 概述

语义匹配模型的得分函数基于相似性而构造，并通过匹配嵌入实体和关系之间潜在的语义来衡量事件的合理性 [9]。本章节将会简要地介绍几种常见的语义匹配模型。

5.2 语义匹配模型 (SME)

语义匹配模型 (**S**emantic **M**atching **E**nergy Model) 意在通过矩阵捕捉实体和关系之间的联系。SME 分为两种语义匹配模型：线性与双线性模型，他们的得分函数分别为 [6]：

$$f_r(h,t) = (\mathbf{M}_1\mathbf{h} + \mathbf{M}_2\mathbf{r} + \mathbf{b}_1)^\top(\mathbf{M}_3\mathbf{t} + \mathbf{M}_4\mathbf{r} + \mathbf{b}_2), \quad \text{(linear)}$$
$$f_r(h,t) = ((\mathbf{M}_1) \otimes (\mathbf{M}_2\mathbf{r}) + \mathbf{b}_1)^\top((\mathbf{M}_3\mathbf{t}) \otimes (\mathbf{M}_4\mathbf{r}) + \mathbf{b}_2), \quad \text{(bilinear)}$$

其中 $\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3, \mathbf{M}_4$ 为参数矩阵，b_1, b_2 为误差向量，$\otimes$ 为 Hadamard Product。

5.3 隐语义模型 (LFM)

隐语义模型 (**L**atent **F**actor **M**odel) 将每个实体映射到一个向量并对每个关系定义一个矩阵，并定义得分函数为：

$$f_r(h,t) = \mathbf{h}^\top\mathbf{M}_r\mathbf{t}$$

通过训练关系矩阵，LFM 可以通过以上较为简便的算法来获取头尾实体之间的语义关系。

5.4 单层神经网络模型 (SLM)

单层神经网络模型 (**S**ingle **L**ayer **M**odel) 是神经张量模型 (NTN) 的基础模型。该模型通过建造一个非线性神经网络去表示得分函数，其中 $\mathbf{M}_{r1}, \mathbf{M}_{r2}$ 和 $\mathbf{b}_r$ 是根据关系 r 定义的 [10]：

$$f_r(\mathbf{h},\mathbf{t}) = \mathbf{u}_r^\top \tanh(\mathbf{M}_{r1}\mathbf{h} + \mathbf{M}_{r2}\mathbf{t} + \mathbf{b}_r)$$

5.5 神经张量模型 (NTN)

神经张量模型 (**N**eural **T**ensor **N**etwork) 拓展了 SLM 模型，通过计算非线性神经网络的二次相关性，获得的得分函数为：

$$f_r(\mathbf{h},\mathbf{t}) = \mathbf{u}_r^\top \tanh(\mathbf{h}^\top\hat{\mathbf{W}}_r\mathbf{t} + \mathbf{M}_r\begin{bmatrix}\mathbf{h}\\ \mathbf{t}\end{bmatrix} + \mathbf{b}_r)$$

其中 $\hat{\mathbf{W}}_r$ 为 3 阶张量，$\mathbf{M}_r$ 为参数矩阵，b_r 为误差向量。由于涉及 3 阶张量的计算，NTN 的参数数量很多且计算复杂度非常高，但是 NTN 也是对实体关系考虑最全面的模型，只是由于他的高复杂度使其很难运用于较大的知识图谱 [10]。

5.6 RESCAL

RESCAL 也叫做双线性模型，运用张量分解模型来分析关系数据的内部结构 [8]。它将每个实体嵌入到向量中捕获语义，并将关系映射到矩阵中表示。其得分函数可以用双线性函数表示 [9]，该函数描述了头尾实体之间所有的成对相互作用:

$$f_r(h,t) = \mathbf{h}^\top\mathbf{M}_r\mathbf{t} = \sum_{i=0}^{d-1}\sum_{j=0}^{d-1}[\mathbf{M}_r]_{ij} \cdot [\mathbf{h}]_i \cdot [\mathbf{t}]_j$$

5.7 DistMult

DistMult 是 RESCAL 的简化版, 即限制 $\mathbf{M}_r$ 为对角矩阵。DistMult 将每个关系嵌入 $\mathbf{r} \in \mathbb{R}^d$ 后, 定义 $\mathbf{M}_r = diag(\mathbf{r})$, 所以得分函数为 [9]：

$$f_r(h,t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t} = \sum_{i=0}^{d-1} [\mathbf{r}]_i \cdot [\mathbf{h}]_i \cdot [\mathbf{t}]_i$$

由于该得分函数具有 $f_r(h,t) = f_r(t,h)$ 的特点，所以只能描述对称关系，无法描述更复杂的知识数据。

5.8 ComplEx[9]

ComplEx(Complex Embedding) 在复数领域拓展了 DistMult，即嵌入的向量被定义在 $\mathbb{C}^d$，而不是在 $\mathbb{R}^d$。相似于 DistMult, Complex 先将关系嵌入 $\mathbf{r} \in \mathbb{C}^d$, 再定义 $\mathbf{M}_r = diag(\mathbf{r})$，其得分函数会取结果的实数部分 [9]，即：

$$f_r(h,t) = Re(\mathbf{h}^\top \mathbf{M}_r \bar{\mathbf{t}}) = Re\left(\sum_{i=0}^{d-1} [\mathbf{r}]_i \cdot [\mathbf{h}]_i \cdot [\bar{\mathbf{t}}]_i\right)$$

在 ComplEx 中，$f_r(h,t) \neq f_r(t,h)$，所以 ComplEx 可以更灵活地表达对称与非对称的关系 [2]。

5.9 ANALOGY

ANALOGY 对知识图谱中的类比推理的基本结构进行分析，并对 DistMult 中增加了两个对关系矩阵的约束 [2][9]：

$$\mathbf{M}_r \mathbf{M}_r^\top = \mathbf{M}_r^\top \mathbf{M}_r, \;\; \mathbf{M}_r \mathbf{M}_r' = \mathbf{M}_r' \mathbf{M}_r$$

其中后者代表了在复合关系中关系矩阵的可交换性 [16]。ANALOGY 的得分函数与 DistMult 相同，为：

$$f_r(h,t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t}$$

6 时序知识图谱

6.1 时序知识图谱概述

目前大多数对于知识图谱的研究着重于静态的知识图谱，即事实不随时间改变而改变 [18]。然而对于时序知识图谱的研究同样十分重要，因为大部分事实信息是具有时效性的，例如 (Donald Trump, is-President-of, USA) 这条信息只有在 2016 到 2020 年之间是正确的。所以说如果想要知识图谱包含时间的信息，即实现时序知识图谱的完善，那么在嵌入过程中就需要增加一个时间变量，t，即时序图谱中每条事实都是一个**四元组**，(e_i, r, e_j, t)，其中 t 包含了事实的时间信息，例如 (Donald Trump, is-President-of, USA，[2016,2020])。

在 [19] 中提到，t 可以写成 $[t_b, t_e]$ 来代表事件的有效时间区间。对于特定时间发生的非持续事件，有 $t_b = t_e$，对于正在发生的事件，有 $t = [t_b, +\infty]$。对于时序知识图谱，我们可以预测四元组的任意实体，例如“在 2017 年谁是美国的总统”等与时间相关的问题。

6.2 TAE/ILP

6.2.1 TAE 基本概念和定义

TAE(**T**ime-**A**ware Knowledge Graph **E**mbedding Model) 模型是第一个尝试将时间信息整合到知识图谱中的模型，即第一个时序知识图谱模型 [19]。在 TAE 中，[19] 假设关系具有时间连续性，例如对一个人来说，存在一个关系的**时间线**：出生地是 (wasBornIn) → 毕业地是 (graduatedFrom) → 死亡于 (diedIn)。根据这一点，TAE 定义**时序关系组** (Temporal Ordering Relation Pair) 为两个具有时间顺序，且共享同一头实体的关系，例如 ⟨wasBornIn, diedIn⟩。其中前者叫做**前序关系** (Prior Relation)，后者叫做**后序关系** (Subsequent Relation)。其中符合时间顺序的关系组称为**正关系组**，例如 ⟨wasBornIn, diedIn⟩；不符合时间顺序的关系组称为**负关系组**，例如 ⟨diedIn, wasBornIn⟩。

对于关系在时间中的变化，定义**时间进化矩阵** $\mathbf{T} \in \mathbb{R}^{n\times n}$ 来对时间的变化进行建模，其中 n 为关系向量的维度。我们假设在时序关系组中，前序关系可以通过 $\mathbf{T}$ 得到后序关系，即

$$\mathbf{r}_1\mathbf{T} \approx \mathbf{r}_2, \quad \mathbf{r}_2\mathbf{T} \neq \mathbf{r}_1$$

这种方式可以在训练中自动识别前序和后序的关系。

下图是简单阐述了时间进化矩阵如何在 TAE 中对实体向量进行投影。$\mathbf{T}$ 在 TAE 中的作用等同于向量 r 在 TransE 中的作用。不同点在于 TAE 需要同时符合上述的两个条件，即 $\mathbf{r}_1\mathbf{T}$ 需要靠近 $\mathbf{r}_2$，并且 $\mathbf{r}_2\mathbf{T}$ 需要尽可能远离 $\mathbf{r}_1$。

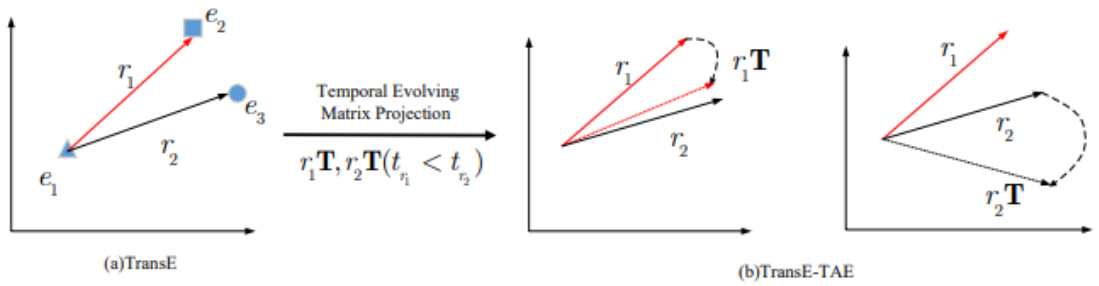


图 20: Time-Aware 中时间进化矩阵的作用图示 [19]

6.2.2 TAE 模型训练

TAE 的模型训练方法与传统的距离模型相似，但由于 TAE 中有额外的时间矩阵，需要额外分开进行训练。对于一个训练集中的正确四元组 $(e_i, r_k, e_j, t_{r_k}) \in \Delta_t$，我们可以找到一个对应的四元组 $(e_i, r_l, e_m, t_{r_l}) \in \Delta_t$，其中他们共享头实体 e_i，且 $\langle r_k, r_l\rangle$ 是一个时序关系组。如果 $t_{r_k} < t_{r_L}$，则该时序关系组为正关系组，写作 $y^+ = \langle r_k, r_l\rangle$，反之为负关系组，写作 $y^- = \langle r_k, r_l\rangle^{-1} = \langle r_l, r_k\rangle$。对于时间进化矩阵的训练，基本的训练原则是正关系组的得分需要小于对应负关系组的得分 (这样总得分函数会尽可能小)，所以我们定义**时间得分函数** (Temporal Score Function) 为：

$$g(\langle r_k, r_l\rangle) = \|\mathbf{r}_k\mathbf{T} - \mathbf{r}_l\|_{l_1/l_2} \quad (L_1 \text{ or } L_2 \text{ Distance})$$

对于**总得分函数** (Joint Score Function)，在事实部分，我们依然采用传统的距离模型的得分和损失函数，这里称为**事实得分函数**，即：

$$f(e_i, r, e_j) = \|\mathbf{e}_i + \mathbf{r} - \mathbf{e}_j\|_{l_1/l_2}$$
$$\mathcal{L} = \sum_{x^+ \in \Delta} \sum_{x^- \in \Delta'} [\gamma + f(x^+) - f(x^-)]_+$$

最后，我们结合事实和时间的得分函数，得到最终的损失函数为：

$$\mathcal{L} = \sum_{x^+ \in \Delta} \left[\sum_{x^- \in \Delta'} [\gamma_1 + f(x^+) - f(x^-)]_+ + \lambda \sum_{y^+ \in \Omega_{e_i, t_{r_k}}, y^- \in \Omega'_{e_i, t_{r_k}}} [\gamma_2 + g(y^+) - g(y^-)]_+ \right]$$

其中第一项控制了事实的准确性，第二项控制了时间的准确性，λ 起到了两者之间的平衡作用；$x^+ = (e_i, r_k, e_j) \in \Delta$ 是正确三元组，$x^- = (e'_i, r_k, e'_j)$ 是通过替换头或尾实体得到的“错误”三元组。其中关于四元组 (e_i, r_k, e_j, t_{r_k}) 的正关系组的集合为：

$$\begin{aligned}\Omega_{e_i, t_{r_k}} = &\{\langle r_k, r_l \rangle \,|(e_i, r_k, e_j, t_{r_k}) \in \Delta_t, (e_i, r_l, e_m, t_{r_l}) \in \Delta_t, t_{r_k} < t_{r_l}\} \\ &\cup \{\langle r_l, r_k \rangle \,|(e_i, r_k, e_j, t_{r_k}) \in \Delta_t, (e_i, r_l, e_m, t_{r_l}) \in \Delta_t, t_{r_k} > t_{r_l}\}\end{aligned}$$

其中 r_k 和 r_l 共享同一头实体，且 $\langle r_k, r_l \rangle$ 是正关系组。反之，$\Omega'_{e_i, t_{r_k}}$ 是相应的负关系组集合，即 $\langle r_k, r_l \rangle$ 是负关系组。此外，损失函数还需要满足一下要求：

$$\|\mathbf{e}_i\|_2, \|\mathbf{r}_k\|_2, \|\mathbf{r}_l\|_2, \|\mathbf{e}_j\|_2, \|\mathbf{r}_k\mathbf{T}\|_2, \|\mathbf{r}_l\mathbf{T}\|_2 \leqslant 1$$

6.2.3 ILP 基本概念

在传统的链接预测中，经常会出现错误的三元组，在时序知识图谱中，当这些三元组变为带有时间变量的四元组时，我们可以通过时间变量在特定情况下排除一些错误的四元组。[19] 中主要提到了以下三种情况：

不共时现象 (Temporal Disjointness)：对于某些事件，发生的事件不能有重合。例如一个人在一个给定时间点只能是一个人的夫妻，即 $(e_1, \text{wasSpouseOf}, e_2, [1990, 2010]) \wedge (e_1, \text{wasSpouseOf}, e_3, [2005, 2013]) \wedge e_2 \neq e_3 \rightarrow \text{false}$。

时序现象 (Temporal Ordering)：对于某些事件，有的事件永远在另一事件之后发生，例如一个人毕业的事件必须在其出生之后才能发生，即 $(e_1, \text{wasBornIn}, e_2, t_1) \wedge (e_1, \text{graduatedFrom}, e_3, t_2) \wedge t_1 > t_2 \rightarrow \text{false}$。

时间独立现象 (Temporal Span)：对于某些事件，其发生时间只能在一个时间段，意味着在其他时间段中该事件都不成立。例如一个人只能在一个时间段中成为总统，即如果 $(e_1 \text{ presidentOf}, e_2, t) \in \Delta$，那么 $(e_1, \text{presidentOf}, e_2, t') \in \Delta', \ \forall t' \neq t$。

对于这三种情况，[19] 中提出将这些问题当作**线性规划** (**I**nteger **L**inear **P**rogram Formulation) 问题来处理。

6.2.4 模型目标函数

在 ILP 中，对于每个三元组 (e_i, r_k, e_j)，用 $w_{ij}^{(k)} = f(e_i, r_k, e_j)$ 来代表在嵌入模型中这个三元组的可行性，并使用一个二元参数 $x_{ij}^{(k)}$ 来表示是否四元组 (e_i, r_k, e_j, t) 在时间 t 时成立。该模型的目标是找

到最佳的分配方式以最大化总体的可行性，同时满足时间上的条件 (即 6.2.3 中的三条)。总体的目标函数可写为：

$$\max \sum_{x_{ij}^{(k)}} w_{ij}^{(k)} x_{ij}^{(k)}$$

对于这个目标函数，关于 6.2.3 中的条件有以下三个限制：

$$x_{ij}^{(k)} + x_{il}^{(k)} \leqslant 1, \forall k \in \mathcal{C}^d, t_{x_{ij}^{(k)}} \cap t_{x_{il}^{(k)}} \neq \emptyset$$
$$x_{ij}^{(k)} + x_{il}^{(k)} \leqslant 1, \forall (k,k') \in \mathcal{C}^o, t_{x_{ij}^{(k)}} \geqslant t_{x_{il}^{(k)}}$$
$$x_{ij}^{(k)} = 0, \forall k \in \mathcal{C}^s, t_{x_{ij}^{(k)}} \cap t_{\Delta} = \emptyset$$

其中 $\mathcal{C}^d, \mathcal{C}^o, \mathcal{C}^s$ 分别是有不共时现象，时序现象和时间独立现象的关系集合。

6.2.5 合并两种模型

由于两种模型互补的特质，[19] 中将两种模型合并，其中 ILP 会考虑到更多和更详细的时间限制，而 TAE 会为 ILP 提供更准确的嵌入和目标函数。
具体来说，对于一个没有见过的四元组 (e_i, r_k, e_j, t)，我们用 $x_{ij}^{(k,t)}$ 来表示其是否是正确四元组。然后利用 TAE 的嵌入模型来计算 ILP 目标函数中的可行性变量 $v_{ij}^{(k,t)}$，最后得到的目标函数为：

$$\max \sum_{x_{ij}^{(k,t)}} v_{ij}^{(k,t)} x_{ij}^{(k,t)}$$

其中目标函数中的限制与 6.2.4 中的相同。

6.2.6 评估和结论

为了得到时序知识图谱的数据，[19] 从知识数据库例如 YG15K 和 FB87 中分类出具有时间信息的知识数据，例如“死亡于”(diedIn) 的关系具有时间效应，而“邻国有”(hasNeighbor) 的关系没有时间效应。下图为 TAE/ILP 和合并模型在链接预测中的表现，可以看到与非时序知识图谱模型相比，对于时序知识图谱的链接预测，三种模型的表现均有比较明显的提升，并且合并模型的准确度要好于单独的两个模型。此外，由于 TAE/ILP 独立训练时间的特点，它可以依附于传统的距离模型例如 TransE, TransH, 和 TransR。

DataSets	YG15k				YG36k				FB42				FB87			
Metric	MeanRank		Hits@10(%)		MeanRank		Hits@10(%)		MeanRank		Hits@10(%)		MeanRank		Hits@10(%)	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE	990	971	26.6	29.5	179	163	65.7	75.6	383	341	39.5	46.6	105	54	47.7	70.2
TransE-TAE	**245**	**244**	**34.4**	**35.3**	**62**	**58**	**75.4**	**81.9**	**328**	**300**	**45.0**	**51.3**	**92**	**50**	**53.8**	**77.5**
TransE-ILP	-	-	**40.5**	**41.9**	-	-	**80.1**	**85.4**	-	-	**53.5**	**55.2**	-	-	**62.1**	**82.7**
TransE-TAE-ILP	-	-	**44.8**	**46.1**	-	-	**84.7**	**89.4**	-	-	**65.1**	**70.2**	-	-	**65.7**	**85.3**
TransH	986	966	25.7	28.0	174	158	65.3	77.8	378	333	40.3	48.1	102	58	45.3	71.8
TransH-TAE	**243**	**241**	**33.4**	**34.7**	**63**	**58**	**75.3**	**81.6**	**320**	**291**	**46.4**	**52.7**	**93**	**52**	**55.3**	**78.5**
TransH-ILP	-	-	**41.7**	**42.6**	-	-	**81.5**	**85.6**	-	-	**53.7**	**56.4**	-	-	**63.5**	**81.1**
TransH-TAE-ILP	-	-	**43.3**	**46.6**	-	-	**85.4**	**88.7**	-	-	**65.3**	**71.4**	-	-	**67.2**	**86.0**
TransR	976	955	29.5	30.2	175	153	68.3	80.1	371	325	42.5	49.2	96	52	49.3	72.1
TransR-TAE	**253**	**251**	**33.5**	**33.9**	**56**	**45**	**79.5**	**86.9**	**318**	**282**	**47.2**	**54.9**	**88**	**47**	**56.5**	**79.9**
TransR-ILP	-	-	**41.9**	**44.3**	-	-	**82.6**	**82.5**	-	-	**57.8**	**57.1**	-	-	**63.4**	**87.9**
TransR-TAE-ILP	-	-	**45.4**	**47.7**	-	-	**85.8**	**89.5**	-	-	**66.5**	**72.3**	-	-	**68.2**	**88.2**

图 21: TAE/ILP 在链接预测中的表现 [19]

6.3 HyTE

6.3.1 基本概念

另一种常见的时序知识图谱模型是基于将时间项看做超平面进行嵌入的 HyTE(**Hy**perplane-based **T**emporally aware Knowledge Graph **E**mbedding)。HyTE 直接将时间信息嵌入到知识信息中，然后将整个知识图谱按照时间点分为多个子图谱，使得每个子图是一个静态的知识图谱，再将每个子图中的实体和关系投影到相应时间所代表的超平面上 [20]。

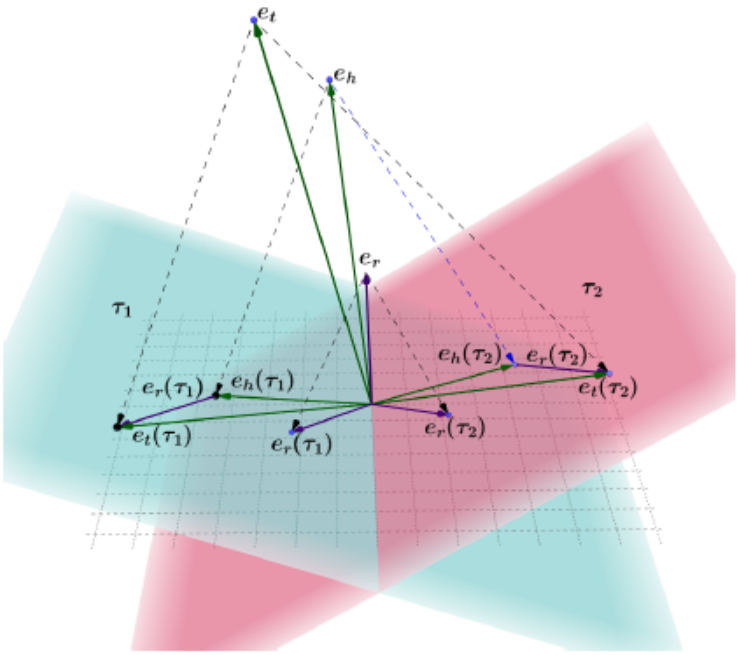


图 22: HyTE 的模型原理 [20]

6.3.2 模型原理

[20] 依然在传统模型中的三元组上额外添加了时间的维度，形成了四元组 $(h, r, t, [\tau_s, \tau_e])$，其中 τ_s 和 τ_e 分别代表了该事件 $((h, r, t))$ 成立的开始和结束时间。并且 HyTE 将知识图谱分成多个子图谱的并集，即：

$$\mathbb{G} = \mathbb{G}_{\tau_1} \cup \mathbb{G}_{\tau_2} \cup \cdots \cup \mathbb{G}_{\tau_T}$$

其中 τ_i 为具体的时间点。对于一个四元组 $(h, r, t, [\tau_s, \tau_e])$，我们将 (h, r, t) 加入到所有符合这个时间区间的子图谱中，即加入到 $\mathbb{G}_\tau$，$\tau \in [\tau_s, \tau_e]$，并将在时间 τ 中的正确三元组的集合被定义为 $\mathcal{D}_\tau^+$。由于传统模型中针对一对多的方法无法考虑到时间因素对尾实体的影响，所以 HyTE 用超平面来表示时间，在模型中用单位向量 $\mathbf{w}_{t_1}, \mathbf{w}_{t_2}, \cdots, \mathbf{w}_{t_T}$ 来表示。因此我们可以将空间按照时间分类，例如在时间 τ 成立的三元组 (在子图谱 $\mathbb{G}_\tau$ 中) 将被投影到超平面 $\mathbf{w}_\tau$ 中。具体来说，对于超平面 $\mathbf{w}_\tau$，我们有：

$$\begin{aligned} P_\tau(\mathbf{e}_h) &= \mathbf{e}_h - \mathbf{w}_\tau^\top \mathbf{e}_h \mathbf{w}_\tau \\ P_\tau(\mathbf{e}_t) &= \mathbf{e}_t - \mathbf{w}_\tau^\top \mathbf{e}_t \mathbf{w}_\tau \\ P_\tau(\mathbf{e}_r) &= \mathbf{e}_r - \mathbf{w}_\tau^\top \mathbf{e}_r \mathbf{w}_\tau \end{aligned}$$

其中 $\|\mathbf{w}_\tau\|_2 = 1$，并且我们希望对于一个在时间 τ 有效的三元组满足:

$$P_\tau(\mathbf{e}_h) + P_\tau(\mathbf{e}_r) \approx P_\tau(\mathbf{e}_t)$$

最后我们将每个时间综合起来，得到最后的损失函数为：

$$\mathcal{L} = \sum_{\tau\in[T]} \sum_{x\in\mathcal{D}_\tau^+} \sum_{y\in\mathcal{D}_\tau^-} [f_\tau(x) - f_\tau(y) + \gamma]_+$$

其中 $\|\mathbf{e}_p\|_2 \leqslant 1\ \forall p \in \mathcal{E},\ \ \|\mathbf{w}_\tau\|_2 = 1,\ \forall \tau \in [T]$。对于替换正确的四元组，[20] 采用了两种不同的方法：

第一种方法是**实体替换法**，Time Agnostic Negative Sampling (TANS)，即考虑所有不属于在知识图谱中的四元组：

$$\begin{aligned}\mathcal{D}_\tau^- =&\{(h', r, t, \tau) \,|\, h' \in \mathcal{E}, (h', r, t) \notin \mathcal{D}^+\} \cup \\ &\{(h, r, t', \tau) \,|\, t' \in \mathcal{E}, (h, r, t') \notin \mathcal{D}^+\}\end{aligned}$$

第二种方法是实体时间替换法，Time Dependent Negative Sampling (TDNS)，即考虑属于知识图谱但不存在于时间为 τ 的子图谱中的四元组：

$$\begin{aligned}\mathcal{D}_\tau^- =&\{(h', r, t, \tau) \,|\, h' \in \mathcal{E}, (h', r, t) \in \mathcal{D}^+, (h', r, t, \tau) \notin \mathcal{D}_\tau^+\} \cup \\ &\{(h, r, t', \tau) \,|\, t' \in \mathcal{E}, (h, r, t') \in \mathcal{D}^+, (h, r, t', \tau) \notin \mathcal{D}_\tau^+\}\end{aligned}$$

6.3.3 评估和结论

[20] 中选用了 YAGO11k 和 Wikidata12K 这两个时序知识图谱进行测试，其中替换四元组的方法采用了 TDNS 方法，下图中可以看到对时序知识图谱的链接预测中 HyTE 的准确率较传统模型有很大的改善。

Dataset	YAGO11K				Wikidata12K			
Metric	Mean Rank		Hits@10(%)		Mean Rank		Hits@10(%)	
	tail	head	tail	head	tail	head	tail	head
Trans-E (Bordes et al., 2013)	504	2020	4.4	1.2	520	740	11.0	6.0
TransH (Wang et al., 2014)	354	1808	5.8	1.5	423	648	23.7	11.8
HolE (Nickel et al., 2016b)	1828	1953	29.4	13.7	734	808	25.0	12.3
t-TransE (Jiang et al., 2016)	292	1692	6.2	1.3	283	413	24.5	14.5
HyTE	**107**	**1069**	**38.4**	**16.0**	**179**	**237**	**41.6**	**25.0**

图 23: HyTE 在链接预测中的表现 [20]

7 总结

作为一个新兴的研究领域，知识图谱的应用前景十分广泛。目前已经有很多关于知识图谱嵌入的表示和学习模型。这里只是给出一个较为简单的总结，下图是在 [18] 中总结的较为全面的模型简介。

Category	Model	Ent. embed.	Rel. embed.	Scoring Function $f_r(h,t)$
Polar coordinate	HAKE [19]	$\mathbf{h}_m, \mathbf{t}_m \in \mathbb{R}^k$ $\mathbf{h}_p, \mathbf{t}_p \in [0, 2\pi)^k$	$\mathbf{r}_m \in \mathbb{R}_+^k$ $\mathbf{r}_p \in [0, 2\pi)^k$	$-\left\|\mathbf{h}_m \circ \mathbf{r}_m - \mathbf{t}_m\right\|_2 -$ $\lambda \left\|\sin\left(\left(\mathbf{h}_p + \mathbf{r}_p - \mathbf{t}_p\right)/2\right)\right\|_1$
Complex vector	ComplEx [23]	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{r} \in \mathbb{C}^d$	$\mathrm{Re}\left(< \mathbf{r}, \mathbf{h}, \bar{\mathbf{t}} >\right) = \mathrm{Re}\left(\sum_{k=1}^K \mathbf{r}_k \mathbf{h}_k \bar{\mathbf{t}}_k\right)$
	RotatE [24]	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{r} \in \mathbb{C}^d$	$\left\|\mathbf{h} \circ \mathbf{r} - \mathbf{t}\right\|$
	QuatE [25]	$\mathbf{h}, \mathbf{t} \in \mathbb{H}^d$	$\mathbf{r} \in \mathbb{H}^d$	$\mathbf{h} \otimes \frac{\mathbf{r}}{\left\|\mathbf{r}\right\|} \cdot \mathbf{t}$
Manifold & Group	ManifoldE [28]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\left\|\mathcal{M}(h, r, t) - D_r^2\right\|^2$
	TorusE [15]	$[\mathbf{h}], [\mathbf{t}] \in \mathbb{T}^n$	$[\mathbf{r}] \in \mathbb{T}^n$	$\min_{(x,y) \in ([h]+[r]) \times [t]} \left\|x - y\right\|_i$
	DihEdral [31]	$\mathbf{h}^{(l)}, \mathbf{t}^{(l)} \in \mathbb{R}^2$	$\mathbf{R}^{(l)} \in \mathbb{D}_K$	$\sum_{l=1}^L \mathbf{h}^{(l)\top} \mathbf{R}^{(l)} \mathbf{t}^{(l)}$
	MuRP [29]	$\mathbf{h}, \mathbf{t} \in \mathbb{B}_c^d, b_h, b_t \in \mathbb{R}$	$\mathbf{r} \in \mathbb{B}_c^d$	$-d_{\mathbb{B}}\left(\exp_{\mathbf{0}}^c\left(\mathbf{R}\log_{\mathbf{0}}^c\left(\mathbf{h}\right)\right), \mathbf{t} \oplus_c \mathbf{r}\right)^2 + b_h + b_t$
	AttH [30]	$\mathbf{h}, \mathbf{t} \in \mathbb{B}_c^d, b_h, b_t \in \mathbb{R}$	$\mathbf{r} \in \mathbb{B}_c^d$	$-d_{\mathbb{B}}^{c_r}\left(\mathrm{Att}\left(\mathbf{q}_{\mathrm{Rot}}^H, \mathbf{q}_{\mathrm{Ref}}^H; \mathbf{a}_r\right) \oplus^{c_r} \mathbf{r}_r^H, \mathbf{e}_t^H\right)^2 + b_h + b_t$
Gaussian	KG2E [26]	$\mathbf{h} \sim \mathcal{N}\left(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h\right)$ $\mathbf{t} \sim \mathcal{N}\left(\boldsymbol{\mu}_t, \Sigma_t\right)$ $\boldsymbol{\mu}_h, \boldsymbol{\mu}_t \in \mathbb{R}^d$ $\Sigma_h, \Sigma_t \in \mathbb{R}^{d \times d}$	$\mathbf{r} \sim \mathcal{N}\left(\boldsymbol{\mu}_r, \Sigma_r\right)$ $\boldsymbol{\mu}_r \in \mathbb{R}^d, \Sigma_r \in \mathbb{R}^{d \times d}$	$\int_{x \in \mathcal{R}^{k_e}} \mathcal{N}\left(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r\right) \log \frac{\mathcal{N}(x; \boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e)}{\mathcal{N}(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)} dx$ $\log \int_{x \in \mathcal{R}^{k_e}} \mathcal{N}\left(x; \boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e\right) \mathcal{N}\left(x; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r\right) dx$
	TransG [27]	$\mathbf{h} \sim \mathcal{N}\left(\boldsymbol{\mu}_h, \boldsymbol{\sigma}_h^2 \mathbf{I}\right)$ $\mathbf{t} \sim \mathcal{N}\left(\boldsymbol{\mu}_t, \Sigma_t\right)$ $\boldsymbol{\mu}_h, \boldsymbol{\mu}_t \in \mathbb{R}^d$	$\boldsymbol{\mu}_r^i \sim \mathcal{N}\left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_h, \left(\sigma_h^2 + \sigma_t^2\right) \mathbf{I}\right)$ $\mathbf{r} = \sum_i \pi_r^i \boldsymbol{\mu}_r^i \in \mathbb{R}^d$	$\sum_i \pi_r^i \exp\left(-\frac{\left\|\boldsymbol{\mu}_h + \boldsymbol{\mu}_r^i - \boldsymbol{\mu}_t\right\|_2^2}{\sigma_h^2 + \sigma_t^2}\right)$
Translational Distance	TransE [16]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$-\left\|\mathbf{h} + \mathbf{r} - \mathbf{t}\right\|_{1/2}$
	TransR [17]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^k, \mathbf{M}_r \in \mathbb{R}^{k \times d}$	$-\left\|\mathbf{M}_r \mathbf{h} + \mathbf{r} - \mathbf{M}_r \mathbf{t}\right\|_2^2$
	TransH [20]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r}, \mathbf{w}_r \in \mathbb{R}^d$	$-\left\|\left(\mathbf{h} - \mathbf{w}_r^\top \mathbf{h} \mathbf{w}_r\right) + \mathbf{r} - \left(\mathbf{t} - \mathbf{w}_r^\top \mathbf{t} \mathbf{w}_r\right)\right\|_2^2$
	TransA [34]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d, \mathbf{M}_r \in \mathbb{R}^{d \times d}$	$\left(\left|\mathbf{h} + \mathbf{r} - \mathbf{t}\right|\right)^\top \mathbf{W}_\mathbf{r} \left(\left|\mathbf{h} + \mathbf{r} - \mathbf{t}\right|\right)$
	TransF [35]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\left(\mathbf{h} + \mathbf{r}\right)^\top \mathbf{t} + \left(\mathbf{t} - \mathbf{r}\right)^\top \mathbf{h}$
	ITransF [36]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\left\|\boldsymbol{\alpha}_r^H \cdot \mathbf{D} \cdot \mathbf{h} + \mathbf{r} - \boldsymbol{\alpha}_r^T \cdot \mathbf{D} \cdot \mathbf{t}\right\|_\ell$
	TransAt [37]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$P_r\left(\sigma\left(\mathbf{r}_h\right) \mathbf{h}\right) + \mathbf{r} - P_r\left(\sigma\left(\mathbf{r}_t\right) \mathbf{t}\right)$
	TransD [33]	$\mathbf{h}, \mathbf{t}, \mathbf{w}_h \mathbf{w}_t \in \mathbb{R}^d$	$\mathbf{r}, \mathbf{w}_r \in \mathbb{R}^k$	$-\left\|\left(\mathbf{w}_r \mathbf{w}_h^\top + \mathbf{I}\right) \mathbf{h} + \mathbf{r} - \left(\mathbf{w}_r \mathbf{w}_t^\top + \mathbf{I}\right) \mathbf{t}\right\|_2^2$
	TransM [211]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$-\theta_r \left\|\mathbf{h} + \mathbf{r} - \mathbf{t}\right\|_{1/2}$
	TranSparse [212]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^k, \mathbf{M}_r\left(\theta_r\right) \in \mathbb{R}^{k \times d}$ $\mathbf{M}_r^1\left(\theta_r^1\right), \mathbf{M}_r^2\left(\theta_r^2\right) \in \mathbb{R}^{k \times d}$	$-\left\|\mathbf{M}_r\left(\theta_r\right) \mathbf{h} + \mathbf{r} - \mathbf{M}_r\left(\theta_r\right) \mathbf{t}\right\|_{1/2}^2$ $-\left\|\mathbf{M}_r^1\left(\theta_r^1\right) \mathbf{h} + \mathbf{r} - \mathbf{M}_r^2\left(\theta_r^2\right) \mathbf{t}\right\|_{1/2}^2$
Semantic Matching	TATEC [213]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d, \mathbf{M}_r \in \mathbb{R}^{d \times d}$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t} + \mathbf{h}^\top \mathbf{r} + \mathbf{t}^\top \mathbf{r} + \mathbf{h}^\top \mathbf{D} \mathbf{t}$
	ANALOGY [22]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^{d \times d}$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$
	CrossE [42]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\sigma\left(\tanh\left(\mathbf{c}_r \circ \mathbf{h} + \mathbf{c}_r \circ \mathbf{h} \circ \mathbf{r} + \mathbf{b}\right) \mathbf{t}^\top\right)$
	SME [39]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$g_{\text{left}}(\mathbf{h}, \mathbf{r})^\top g_{\text{right}}(\mathbf{r}, \mathbf{t})$
	DistMult [32]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\mathbf{h}^\top \mathrm{diag}(\mathbf{M}_r) \mathbf{t}$
	HolE [21]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\mathbf{r}^\top (h \star t)$
	HolEx [40]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\sum_{j=0}^l p\left(\mathbf{h}, \boldsymbol{r}; \boldsymbol{c}_j\right) \cdot \boldsymbol{t}$
	SE [8]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r^1, \mathbf{M}_r^2 \in \mathbb{R}^{d \times d}$	$-\left\|\mathbf{M}_r^1 \mathbf{h} - \mathbf{M}_r^2 \mathbf{t}\right\|_1$
	SimplE [48]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r}, \mathbf{r}' \in \mathbb{R}^d$	$\frac{1}{2}\left(\mathbf{h} \circ \mathbf{r}\mathbf{t} + \mathbf{t} \circ \mathbf{r}'\mathbf{t}\right)$
	RESCAL [49]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^{d \times d}$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$
	LFM [51]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{u}_r, \mathbf{v}_r \in \mathbb{R}^p$	$\mathbf{h}^\top \sum_{i=1}^d \boldsymbol{\alpha}_i^r \mathbf{u}_i \mathbf{v}_i^\top \mathbf{t}$
	TuckER [52]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}_e^d$	$\mathbf{r} \in \mathbb{R}_r^d$	$\mathcal{W} \times_1 \mathbf{h} \times_2 \mathbf{r} \times_3 \mathbf{t}$
	LowFER [53]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\left(\mathbf{S}^k \mathrm{diag}\left(\mathbf{U}^T \mathbf{h}\right) \mathbf{V}^T \mathbf{r}\right)^T \mathbf{t}$
Neural Networks	MLP [3]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\sigma(\mathbf{w}^\top \sigma(\mathbf{W}[\mathbf{h}, \mathbf{r}, \mathbf{t}]))$
	NAM [54]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\sigma\left(\mathbf{z}^{(L)} \cdot \mathbf{t} + \mathbf{B}^{(L+1)} \mathbf{r}\right)$
	ConvE [55]	$\mathbf{M}_h \in \mathbb{R}^{d_w \times d_h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^{d_w \times d_h}$	$\sigma\left(\mathrm{vec}\left(\sigma\left(\left[\mathbf{M}_h; \mathbf{M}_r\right] * \boldsymbol{\omega}\right)\right) \mathbf{W}\right) \mathbf{t}$
	ConvKB [43]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$\mathrm{concat}\left(\sigma\left(\left[\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}\right] * \boldsymbol{\omega}\right)\right) \cdot \mathbf{w}$
	HypER [56]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\sigma\left(\mathrm{vec}\left(\mathbf{h} * \mathrm{vec}^{-1}\left(\mathbf{w}_r \mathbf{H}\right)\right) \mathbf{W}\right) \mathbf{t}$
	SACN [44]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$g\left(\mathrm{vec}\left(\mathbf{M}\left(\mathbf{h}, \mathbf{r}\right)\right) W\right) \mathbf{t}$
	NTN [18]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r}, \mathbf{b}_r \in \mathbb{R}^k, \widehat{\mathbf{M}} \in \mathbb{R}^{d \times d \times k}$ $\mathbf{M}_{r,1}, \mathbf{M}_{r,2} \in \mathbb{R}^{k \times d}$	$\mathbf{r}^\top \sigma\left(\mathbf{h}^T \widehat{\mathbf{M}} \mathbf{t} + \mathbf{M}_{r,1} \mathbf{h} + \mathbf{M}_{r,2} \mathbf{t} + \mathbf{b}_r\right)$

图 24: 当前知识图谱模型总结 [18]

参考文献

[1] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in NIPS, 2013, pp. 2787–2795. https://papers.nips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

[2] 王昊奋，漆桂林，陈华钧. 知识图谱方法、实践与应用北京：电子工业出版社，2019.8

[3] Knowledge Graph-Wikipedia, https://en.wikipedia.org/wiki/Knowledge_graph

[4] L. Xu, ”Summary of Translate Model for Knowledge Graph Embedding,” 2018.12, https://towardsdatascience.com/summary-of-translate-model-for-knowledge-graph-embedding-29042be64273

[5] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes,” in AAAI, 2014, pp. 1112–1119. https://persagen.com/files/misc/wang2014knowledge.pdf

[6] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion,” in AAAI, 2015, pp. 2181–2187. https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/viewFile/9571/9523/

[7] 知识图谱：知识表示之 TransR 模型, https://jjzhou012.github.io/blog/2019/01/20/knowledge-representation-TransR.html

[8] M. Nickel, V. Tresp, and H.-P. Kriegel, “A three-way model for collective learning on multi-relational data,” in ICML, vol. 11, 2011, pp. 809–816. https://icml.cc/2011/papers/438_icmlpaper.pdf

[9] Knowledge Graph Embedding: A Survey, https://zhuanlan.zhihu.com/p/106024679

[10] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, “Knowledge graph embedding via dynamic mapping matrix,” in ACL-IJCNLP, vol. 1, 2015, pp. 687–696. https://aclanthology.org/P15-1067.pdf

[11] S. Yang, J. Tian, H. Zhang, J. Yan, H. He, and Y. Jin, “TransMS: knowledge graph embedding for complex relations by multidirectional semantics,” in IJCAI, 2019, pp. 1935–1942. https://www.ijcai.org/proceedings/2019/0268.pdf

[12] H. Xiao, M. Huang, X. Zhu, “From one point to a manifold: Orbit models for knowledge graph embedding,” in IJCAI, 2016, pp. 1315–1321. https://www.ijcai.org/Proceedings/16/Papers/190.pdf

[13] K. Do, T. Tran, S. Venkatesh, “Knowledge Graph Embedding with Multiple Relation Projections,” 2018, https://arxiv.org/pdf/1801.08641.pdf

[14] TransA H. Xiao, M. Huang, Y. Hao, and X. Zhu, “TransA: An adaptive approach for knowledge graph embedding,” in AAAI, 2015, pp. 1–7. https://arxiv.org/pdf/1509.05490.pdf

[15] J. Wang 论文解读：(TransA) TransA: An Adaptive Approach for Knowledge Graph Embedding, https://blog.csdn.net/qq_36426650/article/details/103507252

[16] H. Liu, Y. Wu, and Y. Yang, “Analogical inference for multi-relational embeddings,” in ICML, 2017, pp. 2168–2178. https://arxiv.org/pdf/1705.02426.pdf

[17] S. He, K. Liu, G. Ji, and J. Zhao, “Learning to represent knowledge graphs with gaussian embedding,” in CIKM, 2015, pp. 623–632.
http://www.nlpr.ia.ac.cn/cip/∼liukang/liukangPageFile/Learning%20to%20Represent%20Knowledge%20Graphs%20with%20Gaussian%20Embedding.pdf

[18] S. Ji, S. Pan, E. Cambria, P. Marttinen, P. Yu, “A Survey on Knowledge Graphs: Representation, Acquistion, and Applications,” in IEEE, 2021, pp.1-21.
https://arxiv.org/pdf/2002.00388.pdf#page=24&zoom=100,0,84

[19] T. Liu, T. Ge, L. Sha, B. Chang, S. Li, Z. Sui, “Towards Time-Aware Knowledge Graph Completion,” in Proceedings of COLING 2016, 2016, pp.1715-1724.
https://aclanthology.org/C16-1161.pdf

[20] S. S. Dasgupta, S. N. Ray, and P. Talukdar, “Hyte: Hyperplane-based temporally aware knowledge graph embedding,” in EMNLP, 2018, pp. 2001–2011.
https://aclanthology.org/D18-1225.pdf