



# TuckER

Tensor Factorization for KG Completion



# 摘要

- 知识图谱中只包含所有可能存在的正例的一小部分
- TuckER
  - 简单的线性模型
  - 基于对二元张量表示(binary tensor representation)的三元组的Tucker分解(decomposition)
  - 完全表达模型
  - 实体与关系之间的界限, 比起 ComplEx 和 SimpleE 小几个数量级。



# 非线性模型问题

1. 不透明的 (no transparent)
2. 理解差 (poorly understood)



# TuckER 简介

- 基于对三阶二元张量的 Tucker 分解：
  - Tucker分解将张量分解为**核心张量**乘以**每个模式的矩阵**。
  - 在矩阵正交且核心张量为“全正交”的特殊情况下，可以认为它是高阶奇异值分解(HOSVD)的一种形式。
  - 三个**矩阵**的行包含实体和关系嵌入向量，而**核心张量**的项决定它们之间的交互级别
- 实体不管是主体还是客体在这里是等价的。(嵌入表示与其是不是主体/客体无关)

---

# 相关论文



## 相关论文

RESCAL: 优化(包含每个主体和客体的向量嵌入之间的**双线性乘积**以及**每个关系的完整秩矩阵**的)评分函数。参数过多, 并且过拟合。

DistMult: 将RESCAL中的关系矩阵改为 **对角矩阵**。受限于拉伸变换, 只能处理对称关系。不适用非对称关系。

ComplEx: 扩展DistMult, 实体与客体嵌入不再等价, 引入**复共轭**从而将不对称关系引入张量分解, 使得模型能够处理非对称关系。



## 相关论文

Simple: 基于 Canonical Polyadic (CP) 分解的线性模型。同一实体的主体和客体嵌入是独立的 (DistMult 是 CP 的特例, 其中主体和客体的嵌入是等价的)。相当于对同一个实体的嵌入是其在主/客不同嵌入的平均。

ConvE: 非线性模型, 对主体和关系向量嵌入进行全局的二维卷积, 重塑为矩阵并且对其相连操作。获得的矩阵通过全连接层后与客体进行内积运算。缺点是不够直观。

HypER: 是一个简单的卷积模型, 使用超参数网络为每一个关系生成一个一维卷积, 从主体嵌入中获得关系特征。作者认为卷积是一种引入稀疏性和参数绑定的一种方式。可以通过张量分解理解成非线性。

缺点是将核心权重张量的大多数元素设置为0, 相当于硬正则化。

Table 1. Scoring functions of state-of-the-art link prediction models, the dimensionality of their relation parameters, and significant terms of their space complexity.  $d_e$  and  $d_r$  are the dimensionalities of entity and relation embeddings, while  $n_e$  and  $n_r$  denote the number of entities and relations respectively.  $\bar{\mathbf{e}}_o \in \mathbb{C}^{d_e}$  is the complex conjugate of  $\mathbf{e}_o$ ,  $\underline{\mathbf{e}}_s, \underline{\mathbf{w}}_r \in \mathbb{R}^{d_w \times d_h}$  denote a 2D reshaping of  $\mathbf{e}_s$  and  $\mathbf{w}_r$  respectively,  $\mathbf{h}_{e_s}, \mathbf{t}_{e_s} \in \mathbb{R}^{d_e}$  are the head and tail entity embedding of entity  $e_s$ , and  $\mathbf{w}_{r^{-1}} \in \mathbb{R}^{d_r}$  is the embedding of relation  $r^{-1}$  which is the inverse of relation  $r$ .  $*$  is the convolution operator,  $\langle \cdot \rangle$  denotes the dot product and  $\times_n$  denotes the tensor product along the  $n$ -th mode,  $f$  is a non-linear function, and  $\mathcal{W} \in \mathbb{R}^{d_e \times d_e \times d_r}$  is the core tensor of a Tucker decomposition.

Model	Scoring Function	Relation Parameters	Space Complexity
RESCAL (Nickel et al., 2011)	$\mathbf{e}_s^\top \mathbf{W}_r \mathbf{e}_o$	$\mathbf{W}_r \in \mathbb{R}^{d_e^2}$	$\mathcal{O}(n_e d_e + n_r d_r^2)$
DistMult (Yang et al., 2015)	$\langle \mathbf{e}_s, \mathbf{w}_r, \mathbf{e}_o \rangle$	$\mathbf{w}_r \in \mathbb{R}^{d_e}$	$\mathcal{O}(n_e d_e + n_r d_e)$
ComplEx (Trouillon et al., 2016)	$\text{Re}(\langle \mathbf{e}_s, \mathbf{w}_r, \bar{\mathbf{e}}_o \rangle)$	$\mathbf{w}_r \in \mathbb{C}^{d_e}$	$\mathcal{O}(n_e d_e + n_r d_e)$
ConvE (Dettmers et al., 2018)	$f(\text{vec}(f([\underline{\mathbf{e}}_s; \underline{\mathbf{w}}_r] * w)) \mathbf{W}) \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$
HypER (Balažević et al., 2018)	$f(\text{vec}(\mathbf{e}_s * \text{vec}^{-1}(\mathbf{w}_r \mathbf{H})) \mathbf{W}) \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$
Simple (Kazemi & Poole, 2018)	$\frac{1}{2}(\langle \mathbf{h}_{e_s}, \mathbf{w}_r, \mathbf{t}_{e_o} \rangle + \langle \mathbf{h}_{e_o}, \mathbf{w}_{r^{-1}}, \mathbf{t}_{e_s} \rangle)$	$\mathbf{w}_r \in \mathbb{R}^{d_e}$	$\mathcal{O}(n_e d_e + n_r d_e)$
Tucker (ours)	$\mathcal{W} \times_1 \mathbf{e}_s \times_2 \mathbf{w}_r \times_3 \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$



# Tucker Decomposition

---



# Tucker Decomposition

将一个张量分解成一系列的矩阵和一个更小的核张量。(并不唯一)

$$\mathcal{X} \approx \mathcal{Z} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{p=1}^P \sum_{q=1}^Q \sum_{r=1}^R z_{pqr} \mathbf{a}_p \otimes \mathbf{b}_q \otimes \mathbf{c}_r$$

其中 $\mathbf{X}_n$ 代表在n-th mode下的张量积。

ABC矩阵在正交情况下可以看成每种mode的主要组成成分。

核心张量 $\mathcal{Z}$ 显示了不同成分之间的交互水平。

PQR比IJK小,  $\mathcal{Z}$ 可以看成是一个压缩版本的 $\mathcal{X}$ 。

# Tucker Decomposition 用于链路预测

对KG的三阶二元张量表示使用Tucker分解, 从而进行链路预测。

实体嵌入矩阵  $E = \text{主体}A = \text{客体}C$  (等价  $n_e * d_e$  大小)

关系嵌入矩阵  $R = B$  (等价  $n_r * d_r$  大小)

$n_e$  和  $n_r$  是实体和关系分别的数量,

$d_e$  和  $d_r$  是实体和关系分别的维度。

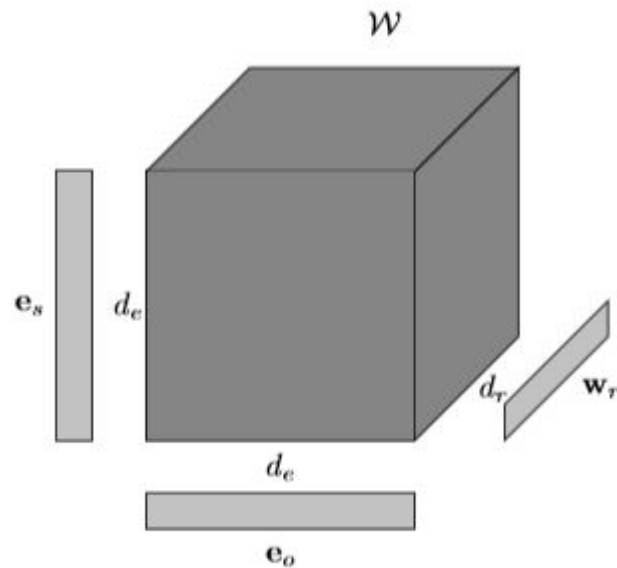


Figure 1. Visualization of the TuckerER architecture.

# Tucker Decomposition 用于链路预测

$$\phi(e_s, r, e_o) = \mathcal{W} \times_1 \mathbf{e}_s \times_2 \mathbf{w}_r \times_3 \mathbf{e}_o$$

$e_s, e_o$  是  $E$  中的行, 表示主体、客体的实体嵌入向量。  
 $w_r$  是  $R$  中的行表示关系嵌入向量。

$\mathcal{W}$  是大小为  $d_e * d_r * d_e$  的核。

$X_n$  代表在  $n$ -th mode 下的张量积。

apply logistic sigmoid to each score  $\phi(e_s, r, e_o)$  to obtain the predicted probability  $p = \sigma(\phi(e_s, r, e_o))$  of a triple being true.

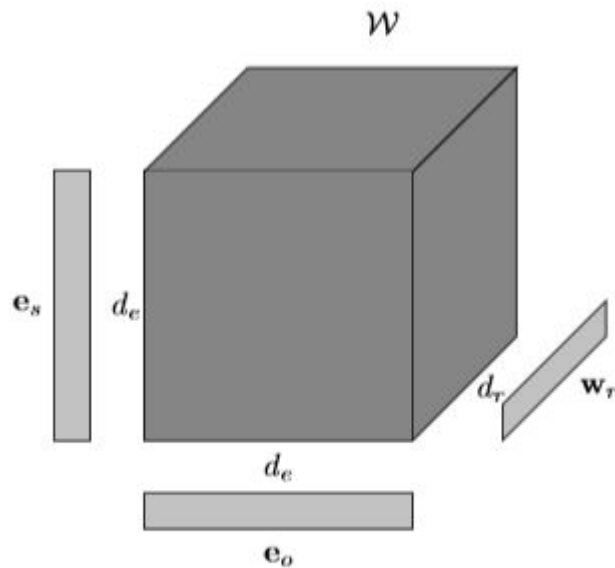


Figure 1. Visualization of the Tucker architecture.



# 模型分析

- 富有表现力
- 给定足够的实体和关系嵌入维度, 能够为嵌入分配值, 使得正确的将正例与负例分开。
- 随着实体和关系数量增加, 参数只是线性程度增加。(因为核心张量 $W$ 只依赖于实体和关系的嵌入维度, 不依赖于数量)
- 通过使用核心张量 $W$ , 与DistMult, ComplEx和SimplE等简单模型不同, TuckER不会将全部已知的知识编码到嵌入中; 一些存储在核心张量中并在所有实体和关系之间共享。
  - 思考与权重分享有什么不同?



## 如何训练

- 应用logistic sigmoid的逆后, 推导因子包含了 $\infty$  and  $-\infty$ , 所以不用分析方法计算张量因子, 而使用数值方法。
- 使用1-N得分, 即我们同时得到一对es和r与所有实体 $e_o \in E$ 。
- 所有存在的为正样本, 保持头+关系 不变换尾, 或者保持 关系+尾 不变换头 作为负样本。
- **minimize the Bernoulli negative log-likelihood loss function:**

$$L(\mathbf{p}, \mathbf{y}) = -\frac{1}{n_e} \sum_{i=1}^{n_e} \left( \mathbf{y}^{(i)} \log(\mathbf{p}^{(i)}) + (1 - \mathbf{y}^{(i)}) \log(1 - \mathbf{p}^{(i)}) \right)$$

---

# 理论分析



# 什么是fully expressive

Complex 与 Simple 是 fully expressive 的。

DistMult 不能处理非对称关系。


Trans 系列文章对于不同的关系类型加入了特定的矛盾。





## 定理 1 建立实体和关系的嵌入维度约束(分解等级)

**Theorem 1.** *Given any ground truth over a set of entities  $\mathcal{E}$  and relations  $\mathcal{R}$ , there exists a TuckER model with subject and object entity embeddings of dimensionality  $d_e = n_e$  and relation embeddings of dimensionality  $d_r = n_r$ , where  $n_e = |\mathcal{E}|$  is the number of entities and  $n_r = |\mathcal{R}|$  the number of relations, that accurately represents that ground truth.*



在实践中，我们期望完全重建基础二元张量所需的实体和关系嵌入维度远小于上述边界，因为对张量的赋值不是随机的而是遵循某种结构。

更重要的是，低分解等级实际上是一个理想的属性，迫使模型学习该结构并推广到新数据，而不是简单地记住输入。

我们期望TuckER比ComplEx和SimpleE具有更好的性能，因为核心张量中的参数共享具有较低维度的嵌入，这对于下游任务的效率可能是重要的。



## Tucker 与 之前的张量分解方法的关系

之前的模型可以看做是Tucker的特殊情况。

- RESCAL  $\mathcal{X} \approx \mathcal{Z} \times_1 \mathbf{A} \times_3 \mathbf{C}$
- Distmult
- ComplEx
- Simple



## 表现关系的非对称性

知识图中的每个关系可以通过一组特性来表征，例如对称性，反射性，传递性等。现在主要有两种方法将不对称性引入三元组的二元张量分解中：

1. 对于主体和对象实体具有不同嵌入，并且对于每个关系具有对角矩阵(或等效地为向量)，但对关系矩阵和对实体嵌入应用的转换类型施加了硬性限制
2. 另一种方式是主体和对象实体嵌入是等价的，但将关系表示为满秩矩阵，这是RESCAL的情况，缺点是参数数量与关系数量的二次增长，这常常导致过度拟合。

TuckER引入了一种处理不对称的新方法：通过将关系表示为向量 $w_r$ ，这使得参数数量与关系数 $r$ 线性增长；并且通过具有不对称的关系的核心张量 $W$ ，这使得关系之间的知识共享成为可能。

将 $W \in \mathbb{R}^{d_e \times d_r \times d_e}$ 与 $w_r \in \mathbb{R}^{d_r}$ 沿第二模式相乘，我们得到满秩关系特定矩阵 $W_r \in \mathbb{R}^{d_e \times d_e}$ ，它能够对实体嵌入执行所有可能的线性变换，即旋转，反射或拉伸，因而能够模拟不对称性。无论为特定关系建模需要什么样的转换，TuckER都能够从数据中学习它，而不是通过明确限制关系矩阵。

# 实验结果

---



# 数据集

FB15K

FB15K-237: 从FB15K中移除训练集中的一些同样在测试集和验证集中的反关系。

WN18

WN18RR: 对WN18同样处理

*Table 4. Dataset statistics.*

Dataset	# Entities ( $n_e$ )	# Relations ( $n_r$ )
FB15k	14,951	1,345
FB15k-237	14,541	237
WN18	40,943	18
WN18RR	40,943	11



## 参数设置

random search based on the validation set performance

对于FB系列:实体维度=关系维度=200

对于WN系列:实体维度200, 关系维度30

学习率从0.01, 0.005, 0.003, 0.001, 0.0005 学习率衰减:1, 0.995, 0.99

四个数据集的best result分别是:

FB15K(0.003, 0.99)	FB15K-237(0.0005,1,0)
WN18(0.005,0.995)	WN18RR(0.01,1.0)

Table 2. Link prediction results on WN18RR and FB15k-237.

	Linear	WN18RR				FB15k-237			
		MRR	Hits@10	Hits@3	Hits@1	MRR	Hits@10	Hits@3	Hits@1
DistMult (Yang et al., 2015)	yes	.430	.490	.440	.390	.241	.419	.263	.155
ComplEx (Trouillon et al., 2016)	yes	.440	.510	.460	.410	.247	.428	.275	.158
Neural LP (Yang et al., 2017)	no	—	—	—	—	.250	.408	—	—
R-GCN (Schlichtkrull et al., 2018)	no	—	—	—	—	.248	.417	.264	.151
MINERVA (Das et al., 2018)	no	—	—	—	—	—	.456	—	—
ConvE (Dettmers et al., 2018)	no	.430	.520	.440	.400	.325	.501	.356	.237
HypER (Balažević et al., 2018)	no	.465	.522	.477	.436	.341	.520	.376	.252
M-Walk (Shen et al., 2018)	no	.437	—	.445	.414	—	—	—	—
TuckER (ours)	yes	<b>.470</b>	<b>.526</b>	<b>.482</b>	<b>.443</b>	<b>.358</b>	<b>.544</b>	<b>.394</b>	<b>.266</b>

Table 3. Link prediction results on WN18 and FB15k.

	Linear	WN18				FB15k			
		MRR	Hits@10	Hits@3	Hits@1	MRR	Hits@10	Hits@3	Hits@1
TransE (Bordes et al., 2013)	no	—	.892	—	—	—	.471	—	—
DistMult (Yang et al., 2015)	yes	.822	.936	.914	.728	.654	.824	.733	.546
ComplEx (Trouillon et al., 2016)	yes	.941	.947	.936	.936	.692	.840	.759	.599
ANALOGY (Liu et al., 2017)	yes	.942	.947	.944	.939	.725	.854	.785	.646
Neural LP (Yang et al., 2017)	no	.940	.945	—	—	.760	.837	—	—
R-GCN (Schlichtkrull et al., 2018)	no	.819	<b>.964</b>	.929	.697	.696	.842	.760	.601
TorusE (Ebisu & Ichise, 2018)	no	.947	.954	.950	.943	.733	.832	.771	.674
ConvE (Dettmers et al., 2018)	no	.943	.956	.946	.935	.657	.831	.723	.558
HypER (Balažević et al., 2018)	no	.951	.958	<b>.955</b>	.947	.790	.885	.829	.734
Simple (Kazemi & Poole, 2018)	yes	.942	.947	.944	.939	.727	.838	.773	.660
TuckER (ours)	yes	<b>.953</b>	.958	<b>.955</b>	<b>.949</b>	<b>.795</b>	<b>.892</b>	<b>.833</b>	<b>.741</b>





## 原因

通过核心张量 $W$  利用关系之间的知识共享, 以及dropout的隐式正则化。

这允许模型学习忽略哪些参数而不是将它们设置为0。

dropout参数的值对结果有显著影响, 每个关系训练三元组数量较多的数据集需要较低的dropout值

## 嵌入维度

在嵌入维度20时, TuckER的性能几乎与 ComplEx和SimpleE在嵌入维度200时的性能一样好, 这支持了我们的初始假设。

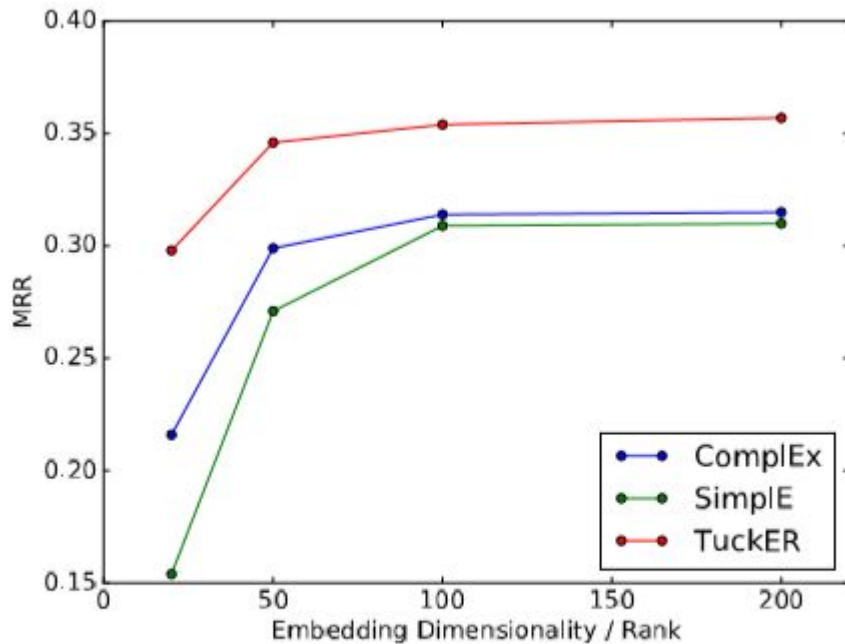


Figure 3. MRR for ComplEx, SimpleE and TuckER for different embeddings sizes  $d_e = d_r \in \{20, 50, 100, 200\}$  on FB15k-237.



# 结论

在这项工作中，我们介绍了TuckER，一种相对简单但高度灵活的线性模型，用于基于训练集三元组的三阶二元张量的Tucker分解的链路预测信息图，它在标准链路预测数据集上实现了最先进的结果。

除了充分表达外，随着知识图中实体或关系的数量增加，TuckER的参数数量相对于嵌入维度呈线性增长。我们进一步表明，先前的线性状态模型，RESCAL, DistMult, ComplEx和Simple, 都是我们模型的特例。

未来的工作可能包括探索除了dropout之外的soft正规化模型的各种方法，并找到将个体关系属性的背景知识纳入现有模型的方法。