

# About Link prediction on Knowledge graph using TuckER

by Haoyu and Joss

March 2020

# Content

Based on TuckER: Tensor Factorization for Knowledge Graph Completion (Balazević et al., 2019)

- What is a knowledge graph?
- Link prediction
- Previous Models
- TuckER
- Summary of the different models
- Advantages of TuckER
- Comparison of different algorithms
- Implementation and Experiments
- Conclusion

# What is a knowledge graph? (KG)

- Oriented labeled graph-structured database which stores relations
- Vertices: entities (objects, persons, situations, events, etc.)
- Edges: relations
- Based on real-world facts

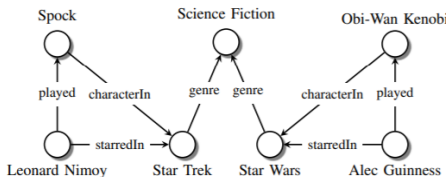


Figure: An example of knowledge graph

# What is a knowledge graph?

<i>subject</i>	<i>predicate</i>	<i>object</i>
(LeonardNimoy,	profession,	Actor)
(LeonardNimoy,	starredIn,	StarTrek)
(LeonardNimoy,	played,	Spock)
(Spock,	characterIn,	StarTrek)
(StarTrek,	genre,	ScienceFiction)

Figure: How the corresponding dataset looks like

# What is a knowledge graph?

Why do we care about link prediction?

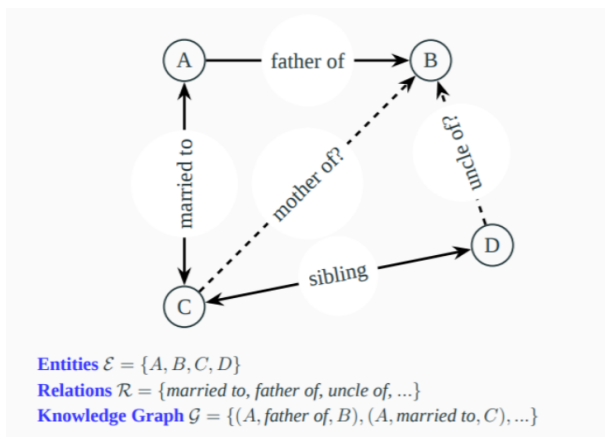


Figure: Another example of knowledge graph

# Example of known knowledge graph data sets

- FB15K (Freebase) : 362 million of facts, 4% are not symmetric but suffer from test set leakage<sup>1</sup>
- FB15k-237 : Created from FB15k by removing the inverse of many relations that are present in the training set.
- WN18 (WordNet)<sup>2</sup> : 60% are not symmetric, suffer from test set leakage
- WN18RR : Subset of WN18 created by removing the inverse of many relations that are present in the training set.

---

<sup>1</sup>inverse relations from the training set were present in the test set

<sup>2</sup>This data set is often categorized as a semantic graph

# What is a knowledge graph?

Why do we care about link prediction?

- We focus on the Open world assumption (OWA).  
The objective is to complete the graph (find out the unknown). Many problems can happen in OWA:
  - Link prediction
  - Entity resolution : which entities refer to the same entity?
  - Link-based clustering: grouping entities based on similarity of their links.

# Link prediction

Also known as knowledge graph completion

Given a knowledge graph  $G$ , we let:

- $\mathcal{E}$  denotes set of entities
- $\mathcal{R}$  denotes set of relations
- $(s, r, o) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$  is a tuple.
- $\zeta$  denotes the set of tuples that are true in a world
- $r \in \mathcal{R}$  is called symmetric if  $(a, r, b) \in \zeta \Leftrightarrow (b, r, a) \in \zeta$
- An embedding is a function  $f_s : \mathcal{E} \rightarrow V$ , or  $f_r : \mathcal{R} \rightarrow V$  or  $f_o : \mathcal{E} \rightarrow V$  where  $V$  is a vector space.
- a tensor factorization model is  $(\mathcal{E}, \mathcal{R}, f_s, f_r, f_o, \phi)$  where  $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbf{R}$  is the scoring function
- Notation:  $v[i]$  is the  $i^{\text{th}}$  entry of  $v$ .



- Linear models
  - CP
  - RESCAL
  - DistMult
  - ComplEx
  - Simple
- Non-linear models
  - ConvE
  - HypER

- $\phi(s, r, o) = e_s^\top W_r e_o$

# DistMult (Yang et al., 2015)

- Special case of RESCAL with diagonal matrix
- Scale to large knowledge graph, at cost of learning less expressive features

# Simple (Kazemi and Poole, 2018)

- $\phi((e_i, r, e_j)) = \frac{1}{2}(\langle h_{e_i}, v_r, t_{e_j} \rangle + \langle h_{e_j}, v_{r-1}, t_{e_i} \rangle)$
- Fully expressive for embedding dimension  $\min(|\mathcal{E}||\mathcal{R}|, \gamma + 1)$  :
  - for  $|\mathcal{E}||\mathcal{R}|$  bounds:  $(e_i, r_j, e_k) \in \zeta$  then set  $h_{e_i}[n] = 1 \Leftrightarrow n \bmod |\mathcal{E}| = i$  else 0 &  $v_{r_j}[n] = 1 \Leftrightarrow \lfloor \frac{n}{|\mathcal{E}|} \rfloor = j$  else 0 and  $t_{e_k}[j|\mathcal{E}| + i] = 1$
  - for  $\gamma + 1$  bounds: induction on  $\gamma$ ,  $\gamma = 0$  clear, inductive step...
- Background knowledge encode :  $(e_i, r, e_j) \in \zeta \Leftrightarrow (e_j, r, e_i) \in \zeta$  by tying  $v_{r-1}, v_r$  as  $(e_i, r, e_j) \in \zeta \Rightarrow \langle h_{e_i}, v_r, t_{e_j} \rangle > 0 \& \langle h_{e_j}, v_{r-1}, t_{e_i} \rangle > 0 \Rightarrow \langle h_{e_j}, v_r, t_{e_i} \rangle > 0 \& \langle h_{e_i}, v_{r-1}, t_{e_j} \rangle > 0$

- Linear model based on the Tucker decomposition
  - One Entity embedding matrix  $E \in \mathbb{R}^{n_e \times d_e}$
  - One Relation Embedding matrix  $R \in \mathbb{R}^{n_r \times d_r}$
  - One Core tensor  $W \in \mathbb{R}^{d_e \times d_r \times d_e}$ .
- Scoring function  $\phi(s, r, o) = W \times_1 e_s \times_2 w_r \times_3 e_o = e_s^T (W \times_2 w_r) e_o$
- 1-N scoring: Takes one  $(s, r)$  pair and scores it against all entities.
- Bernoulli negative log-likelihood loss function:

$$L = -\frac{1}{n} \sum_{i=1}^{n_e} (y^{(i)} \log(p^{(i)}) + (1 - y^{(i)}) \log(1 - p^{(i)}))$$

# Summary of the different models

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)	$f(\text{vec}(f([\underline{\mathbf{e}}_s; \underline{\mathbf{w}}_r] * w)) \mathbf{W}) \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$
Simple (Kazemi and Poole, 2018)	$\frac{1}{2}(\langle \mathbf{h}_{e_s}, \mathbf{w}_r, \mathbf{t}_{e_o} \rangle + \langle \mathbf{h}_{e_o}, \mathbf{w}_{r-1}, \mathbf{t}_{e_s} \rangle)$	$\mathbf{w}_r \in \mathbb{R}^{d_e}$	$\mathcal{O}(n_e d_e + n_r d_e)$
HypER (Balažević et al., 2019)	$f(\text{vec}(\mathbf{e}_s * \text{vec}^{-1}(\mathbf{w}_r \mathbf{H})) \mathbf{W}) \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$
TuckER (ours)	$\mathcal{W} \times_1 \mathbf{e}_s \times_2 \mathbf{w}_r \times_3 \mathbf{e}_o$	$\mathbf{w}_r \in \mathbb{R}^{d_r}$	$\mathcal{O}(n_e d_e + n_r d_r)$

Figure: Models Summary

# Summary of the different models

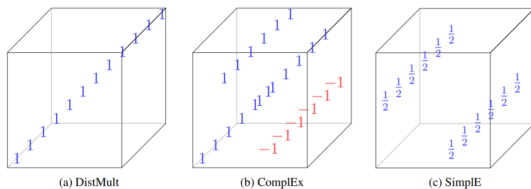


Figure: Models view as Tucker

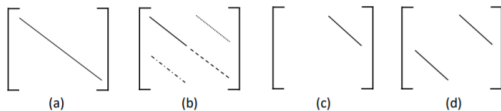


Figure: Models view as Simple

# Advantages of TuckER

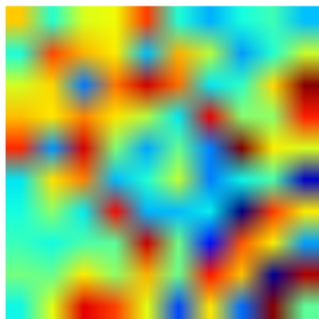
- TuckER is fully expressive
- One entity embedding matrix
- It has parameter sharing encoded in  $W$ 
  - Encode the interaction between the entities and relations
  - Induce less parameters
  - Induce asymmetry
  - Induce multi-task learning



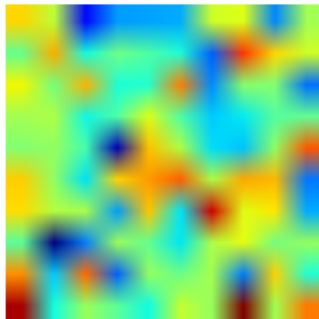
# TuckER is fully expressive

- Theorem : TuckER is fully expressive
- Proof:
  - Let  $e_s, e_o \in \mathbb{R}^{n_e}$  be the one-hot encoding of subject  $s$  and object  $o$  respectively,  $w_r \in \mathbb{R}^{n_r}$  the one-hot encoding of relation  $r$ .
  - Let the core tensor  $W \in \mathbb{R}^{n_e \times n_r \times n_e}$  be 1 at position  $(s, r, o)$  if  $(s, r, o)$  holds,  $-1$  otherwise.
  - Then the tensor product will accurately represent the ground truth from the rest:  
If  $(s, r, o)$  holds,  $\phi(s, r, o) = W \times_1 e_s \times_2 w_r \times_3 e_o = 1$   
If not,  $\phi(s, r, o) = -1$

# Influence of Parameter Sharing



(a)  $W_{\text{derivationally\_related\_form}}$



(b)  $W_{\text{hypernym}}$

Figure 3: Learned relation matrices for a symmetric (derivationally\_related\_form) and an asymmetric (hypernym) WN18RR relation.  $W_{\text{derivationally\_related\_form}}$  is approximately symmetric about the leading diagonal.

# Comparison of Different algorithms

- Comparison idea: use the rank of our ground truth

Query	Proposed Results	Correct response	Rank	Reciprocal rank
cat	catten, cati, <b>cats</b>	cats	3	1/3
tori	torii, <b>tori</b> , toruses	tori	2	1/2
virus	<b>viruses</b> , virii, viri	viruses	1	1

- Common evaluation metrics
  - Mean Reciprocal Rank (MRR)
  - Hits@ $k$ , with  $k$  a small integer (ex: Hits@1 is equivalent to the accuracy, Hits@ $n_e$  always gives 100%)

# Implementation and Experiments

Dataset	lr	dr	$d_u$	$d_r$	d#1	d#2	d#3	ls
FB15k	0.003	0.99	200	200	0.2	0.2	0.3	0.
FB15k-237	0.0005	1.0	200	200	0.3	0.4	0.5	0.1
WN18	0.005	0.995	200	30	0.2	0.1	0.2	0.1
WN18RR	0.01	1.0	200	30	0.2	0.2	0.3	0.1

Figure: Best performing hyper-parameter values for Tucker

Model	lr	dr	$d_u$	$d_r$	d#1	d#2	d#3	ls
ComplEx	0.0001	0.99	200	200	0.2	0.	0.	0.1
Simple	0.0001	0.995	200	200	0.2	0.	0.	0.1

Figure: Best performing hyper-parameter values for ComplEx and Simple on FB15k-237

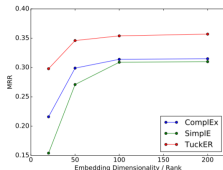


Figure: MRR vs embedding size

# Comparison of the algorithms

	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., 2019)	no	.465	.522	.477	.436	.341	.520	.376	.252
M-Walk (Shen et al., 2018)	no	.437	—	.445	.414	—	—	—	—
RotatE (Sun et al., 2019)	no	—	—	—	—	.297	.480	.328	.205
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 WN18RR and FB15k-237. The RotatE (Sun et al., 2019) results are reported without their self-adversarial negative sampling (see Appendix H in the original paper) for fair comparison.




	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 and 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., 2019)	no	.951	.958	<b>.955</b>	.947	.790	.885	.829	.734
SimplE (Kazemi and 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>

Table 4: Link prediction results on WN18 and FB15k.

Figure: Comparison of the models

# Conclusion

- TuckER outperforms state-of-the-art models
- Number of parameter grows linearly
- previous models are special case of TuckER
- How to incorporate background knowledge?
- Linear model rocks !

-  Ivan Balazevic, Carl Allen, and Timothy M. Hospedales. *TuckER: Tensor Factorization for Knowledge Graph Completion*. CoRR, abs/1901.09590, 2019. URL [hp://arxiv.org/abs/1901.09590](http://arxiv.org/abs/1901.09590)
-  Seyed Mehran Kazemi. David Poole *Simple Embedding for Link Prediction in Knowledge Graphs* In Advances in Neural Information Processing Systems.
-  Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel 2011 *A Three-Way Model for Collective Learning on Multi-Relational Data* In International Conference on Machine Learning.
-  Theo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016 *Complex Embeddings for Simple Link Prediction* In International Conference on Machine Learning.