Knowledge Graph: A Survey of Approaches and Applications

Wenyu Huang, Wenzhi Zong

March 14, 2019

Knowledge Graph

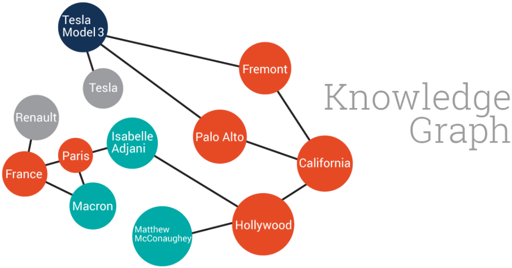


Figure: A Sample of Knowledge Graph

The Knowledge Graph represents a collection of interlinked descriptions of entities real-world objects, events, situations or abstract concepts.

Knowledge Graph

- Descriptions have a formal structure that allows both people and computers to process them in an efficient and unambiguous manner;
- Entity descriptions contribute to one another, forming a network, where each entity represents part of the description of the entities, related to it.

Knowledge Graph

Knowledge Graphs combine characteristics of several data management paradigms. The Knowledge Graph can be seen as a specific type of:

- Database, because it can be queried via structured queries;
- Graph, because it can be analyzed as any other network data structure;
- Knowledge base, because the data in it bears formal semantics, which can be used to interpret the data and infer new facts.

Knowledge Graph Embedding

The triples are effective in representing structured data, but the underlying symbolic nature of such triples usually makes KGs hard to manipulate. To tackle this issue, **Knowledge Graph Embedding**[1] has been proposed and quickly gained massive attention.

The key idea is to embed components of a KG including entities and relations into continuous vector spaces, so as to simplify the manipulation while preserving the inherent structure of the KG.

Knowledge Graph Embedding

Main methods for Knowledge graph embedding contains:

1. Tanslational distance models
2. Semantic mathcing models

Translational Distance Models

Translational distance models exploit distance-based scoring functions. They measure the plausibility of a fact as the distance between the two entities, usually after a translation carried out by the relation.

TransE[2]

Given a triple $(\mathbf{h}, \mathbf{r}, \mathbf{t})$, TransE tries to hold

$$\mathbf{h} + \mathbf{r} \approx \mathbf{t}$$

And the scoring function is then defined as:

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

TransE[2]

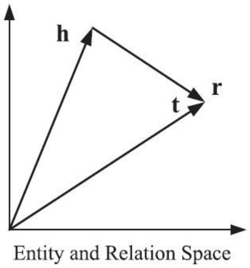


Figure: TransE

TransE[2]

Flaws:

TransE can not dealing with KGs which has 1-to-N, N-to-1, and N-to-N relations.
Take 1-to-N for example, if we have two triples $(\mathbf{h}, \mathbf{r}, \mathbf{t_1})$, $(\mathbf{h}, \mathbf{r}, \mathbf{t_2})$, and TransE tries to hold $\mathbf{h} + \mathbf{r} \approx \mathbf{t_1}$ and $\mathbf{h} + \mathbf{r} \approx \mathbf{t_2}$. That means TransE might learn very similar vector representations for t_1 and t_2 even though they are totally different entities.

TransH[3]

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

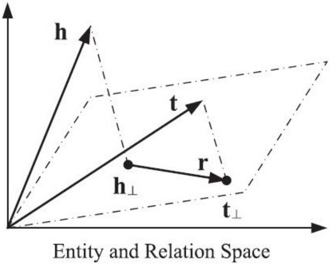


Figure: TransH

TransR[4]

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

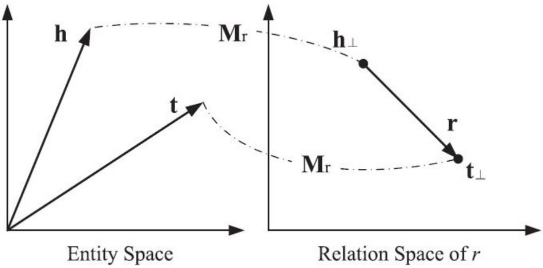


Figure: TransR

Other Extensions of TransE

- TransD
- TranSparse
- TransM
- TransF
- TransA

Other Extensions of TransE

Method	Ent. embedding	Rel. embedding	Scoring function $f_r(h,t)$
TransD [50]	$\mathbf{h}, \mathbf{w}_h \in \mathbb{R}^d$ $\mathbf{t}, \mathbf{w}_t \in \mathbb{R}^d$	$\mathbf{r}, \mathbf{w}_r \in \mathbb{R}^k$	$-\|(\mathbf{w}_r\mathbf{w}_h^\top + \mathbf{I})\mathbf{h} + \mathbf{r} - (\mathbf{w}_r\mathbf{w}_t^\top + \mathbf{I})\mathbf{t}\|_2^2$
TranSparse [51]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^k, \mathbf{M}_r(\theta_r) \in \mathbb{R}^{k\times d}$ $\mathbf{M}_r^1(\theta_r^1), \mathbf{M}_r^2(\theta_r^2) \in \mathbb{R}^{k\times d}$	$-\|\mathbf{M}_r(\theta_r)\mathbf{h} + \mathbf{r} - \mathbf{M}_r(\theta_r)\mathbf{t}\|_{1/2}^2$ $-\|\mathbf{M}_r^1(\theta_r^1)\mathbf{h} + \mathbf{r} - \mathbf{M}_r^2(\theta_r^2)\mathbf{t}\|_{1/2}^2$
TransM [52]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$-\theta_r\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{1/2}$
TransF [54]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$(\mathbf{h} + \mathbf{r})^\top\mathbf{t} + (\mathbf{t} - \mathbf{r})^\top\mathbf{h}$
TransA [55]	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d, \mathbf{M}_r \in \mathbb{R}^{d\times d}$	$-(\lvert\mathbf{h} + \mathbf{r} - \mathbf{t}\rvert)^\top\mathbf{M}_r(\lvert\mathbf{h} + \mathbf{r} - \mathbf{t}\rvert)$

Figure: Other Extensions of TransE

Gaussian Embeddings

Methods introduced so far model entities as well as relations as deterministic points in vector spaces.

KG2E[6] represents entities and relations as random vectors drawn from multivariate Gaussian distributions,i.e.,

$$\mathbf{h} \sim \mathcal{N}(\mu_h, \Sigma_h),$$
$$\mathbf{r} \sim \mathcal{N}(\mu_r, \Sigma_r),$$
$$\mathbf{t} \sim \mathcal{N}(\mu_t, \Sigma_t),$$

Gaussian Embeddings

Kullback-Leibler divergence(KL divergence):

$$
\begin{aligned}
f_r(h,t) &= -D_{KL}(\mathbf{t}-\mathbf{h}||\mathbf{r}) \\
&= -D_{KL}(\mathcal{N}(\mu_t,\Sigma_t)-\mathcal{N}(\mu_h,\Sigma_h)||\mathcal{N}(\mu_r,\Sigma_r))
\end{aligned}
$$

Probability inner product:

$$
f_r(h,t) = \int (\mathcal{N}(\mu_t,\Sigma_t)-\mathcal{N}(\mu_h,\Sigma_h))\cdot\mathcal{N}(\mu_r,\Sigma_r)dx
$$

Introduction

Semantic Matching Models measure plausibility of facts by matching of entities and relations embodied in their vector space representations.

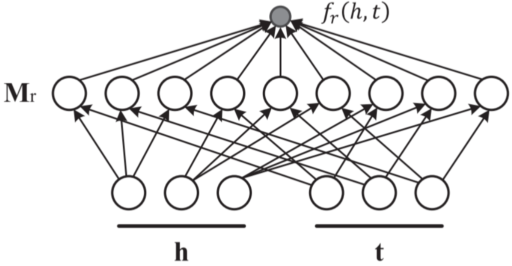
RESCAL family

RESCAL[8]:

The score of a triple (h, r, t) is defined as:

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

where $\mathbf{M}_r$ is the relation Matrix for relation r.

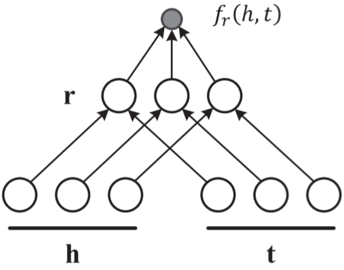


RESCAL family

DistMult[9]:

DistMult simplified RESCAL by restricting $\mathbf{M}_r$ to diagonal matrices. The score function is hence defined as:

$$f_r(h,t) = \mathbf{h}^T diag(\mathbf{r})\mathbf{t}$$



RESCAL family

HolE[10]:

HolE(Holographic Embadding) combines both the expression power of RESCAL and the efficiency of DistMult, by compose the entity representations first into $\mathbf{h} * \mathbf{t} \in \mathbb{R}^d$. The compose function is defined as:

$$[\mathbf{h} * \mathbf{t}]_i = \sum_{k=0}^{d-1} [\mathbf{h}]_k [\mathbf{t}]_{k+i \ mod \ d}$$

and the score function is hence defined as:

$$f_r(h, t) = \mathbf{r}^T \mathbf{h} * \mathbf{t} = \sum_{i=0}^{d-1} [\mathbf{r}]_i \sum_{k=0}^{d-1} [\mathbf{h}]_k [\mathbf{t}]_{k+i \ mod \ d}$$

RESCAL family

HolE[10]:

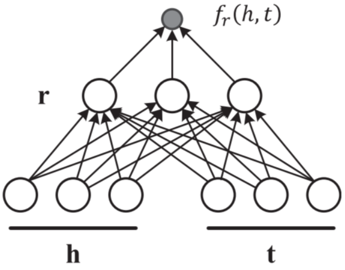


Figure: HolE

RESCAL family

Other RESCAL based methods

- TATEC
- ComplEx
- ANALOGY

Neural Networks

Semantic Matching Energy (SME)[11]

Given a fact (h, r, t), it first projects entities and relations to their vector embeddings in the input layer. The relation $\mathbf{r}$ is then combined with the head entity $\mathbf{h}$ to get $g_u(\mathbf{h}, \mathbf{r})$, and with the tail entity $\mathbf{t}$ to get $g_v(\mathbf{t}, \mathbf{r})$ in the hidden layer. And the score function is defined as

$$f_r(h, t) = g_u(\mathbf{h}, \mathbf{r})^T g_v(\mathbf{t}, \mathbf{r})$$

And SME(linear) defines

$$\begin{aligned} g_u(\mathbf{h}, \mathbf{r}) &= \mathbf{M}_u^1 \mathbf{h} + \mathbf{M}_u^2 \mathbf{r} + \mathbf{b}_u \\ g_v(\mathbf{t}, \mathbf{r}) &= \mathbf{M}_v^1 \mathbf{t} + \mathbf{M}_v^2 \mathbf{r} + \mathbf{b}_v \end{aligned}$$

Neural Networks

Semantic Matching Energy (SME)[11]

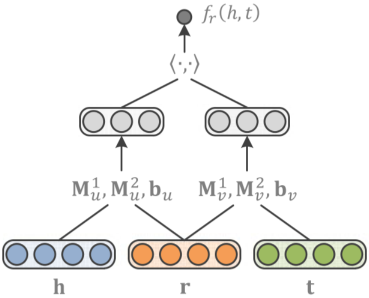


Figure: SME

Neural Networks

Neural Tensor Network (NTN)[12]

Given a fact, NTN first projects entities to their vector embeddings in the input layer. Then, the two entities $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ are combined by a relation-specific tensor $\underline{\mathbf{M}}_r \in \mathbb{R}^{d \times d \times k}$ and mapped to a non-linear hidden layer. Finally, a relation-specific linear output layer gives the score

$$f_r(h, t) = \mathbf{r}^T tanh(\mathbf{h}^T \underline{\mathbf{M}}_r \mathbf{t} + \mathbf{M}_r^1 \mathbf{h} + \mathbf{M}_r^2 \mathbf{t} + \mathbf{b}_r)$$

Neural Networks

Neural Tensor Network (NTN)[12]

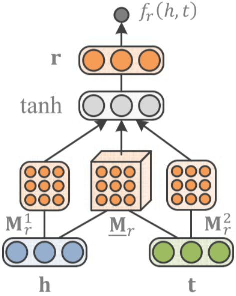


Figure: NTN

Neural Networks

Multi-Layer Perceptron (MLP)[13]

given a fact (h, r, t), the vector embeddings $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$ are concatenated in the input layer, and mapped to a non-linear hidden layer. The score is then generated by a linear output layer, i.e.,

$$f_r(h, t) = \mathbf{w}^T tanh(\mathbf{M}^1\mathbf{h} + \mathbf{M}^2\mathbf{r} + \mathbf{M}^3\mathbf{t})$$

Neural Networks

Multi-Layer Perceptron (MLP)[13]

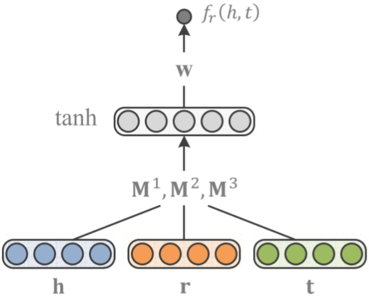


Figure: MLP

Neural Networks

Neural Association Model (NAM)[14]

NAM conducts semantic matching with a deep architecture. Given a fact (h, r, t), it first concatenates the vector embeddings of the head entity and the relation in the input layer, which gives $\mathbf{z}^{(0)} = [\mathbf{h}; \mathbf{r}] \in \mathbb{R}^{2d}$. The input $\mathbf{z}^{(0)}$ is then fed into a deep neural network consisting of L rectified linear hidden layers such that

$$\mathbf{a}^{(l)} = \mathbf{M}^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{l}$$

$$\mathbf{z}^{(l)} = ReLU(\mathbf{a}^{(l)})$$

And the score function is defined as

$$f_r(h, t) = \mathbf{t}^T\mathbf{z}^{(L)}$$

Neural Networks

Neural Association Model (NAM)[14]

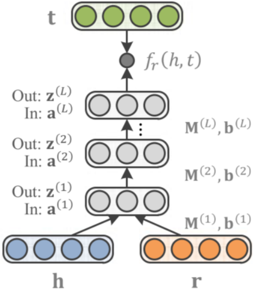


Figure: NAM

Open World Assumption

The open world assumption (OWA) states that KGs contain only true facts and non-observed facts can be either false or just missing.

So we introduce pairwise ranking loss:

$$\mathcal{L} = \min_{\Theta} \sum_{\tau^{+} \in \mathbb{D}^{+}} \sum_{\tau^{-} \in \mathbb{D}^{-}} \max(0, \gamma - f_r(h, t) + f_{r'}(h', t'))$$

Closed World Assumption

The closed world assumption (CWA) assumes that all facts that are not contained in $\mathbb{D}^+$ are false. In this case, we can use squared loss:

$$\mathcal{L} = \min_{\Theta} \sum_{h,t \in \mathbb{E}, r \in \mathbb{R}} (y_{hrt} - f_r(h,t))^2$$

to learn entity and relation representations Θ.

Training Algorithm

Algorithm 1: Training Algorithm

Input: Observed facts $\mathbb{D}^+ = \{(h, r, t)\}$

Initialize entity and relation embeddings

repeat

    $\mathbb{P} \leftarrow$ a small set of positive facts sampled from $\mathbb{D}^+$

    $\mathbb{B}^+ \leftarrow \varnothing, \mathbb{B}^- \leftarrow \varnothing$

    foreach $\tau^+ = (h, r, t) \in \mathbb{P}$ **do**

        Generate a negative fact $\tau^- = (h', r', t')$

        $\mathbb{B}^+ \leftarrow \mathbb{B}^+ \cup \{\tau^+\}, \mathbb{B}^- \leftarrow \mathbb{B}^- \cup \{\tau^-\},$

    end

    Update entity and relation embeddings w.r.t. the gradients of loss fuction

    Handle additional constraints or regularization terms

until *end*;

Output: Entity and relation embeddings

Incorporating Additional Information

The methods introduced so far conduct the embedding task using only facts observed in the KG. In fact, there is a wide variety of additional information that can be incorporated to further improve the task.

Entity Types

Semantically Smooth Embedding (SSE)[15]

$$\mathcal{R}_1 = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}||\mathbf{e}_i - \mathbf{e}_j||_2^2 w_{ij}^1$$

$$\mathcal{R}_2 = \sum_{i=1}^{n}||\mathbf{e}_i - \sum_{e_j \in \mathbb{N}_{e_i}} w_{ij}^2 \mathbf{e}_j||_2^2$$

where w_{ij}^1 refers to whether e_i and e_j belongs to the same catagory. $\mathbb{N}_{e_i}$ is the set containing nearest neighbors of entity e_i, and w_{ij}^2 refers to whether $e_j \in \mathbb{N}_{e_i}$.

Relation Paths

PTransE[16]

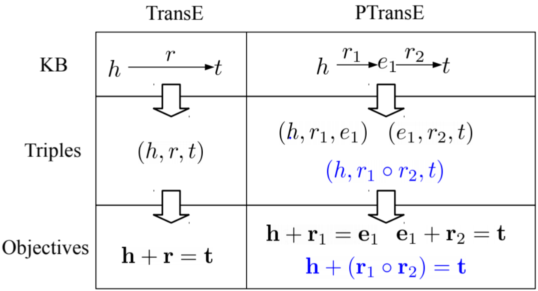


Figure: TransE and PTransE

Textual Descriptions

description-embodied knowledge representation learning (DKRL)[17]

DKRL associates each entity e with two vector representations, i.e., a structure-based $\mathbf{e}_s$ and a description-based $\mathbf{e}_d$. The former captures structural information conveyed in KG facts, while the latter captures textual information expressed in the entity description. Given a fact (h, r, t), DKRL defines the scoring function as

$$f_r(h,t) = -||\mathbf{h}_s + \mathbf{r} - \mathbf{t}_s||_1 - ||\mathbf{h}_d + \mathbf{r} - \mathbf{t}_d||_1 - ||\mathbf{h}_s + \mathbf{r} - \mathbf{t}_d||_1 - ||\mathbf{h}_d + \mathbf{r} - \mathbf{t}_s||_1$$

Logical Rules

KALE(Embeddings by jointly modeling Knowledge And Logic)[18]

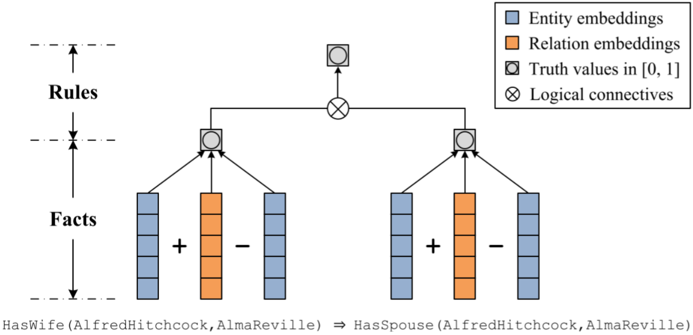
$$I(h, r, t) = 1 - \frac{1}{3\sqrt{d}}||\mathbf{h} + \mathbf{r} - \mathbf{t}||_1$$

$$I(f_1 \Rightarrow f_2) = I(f_1) \cdot I(f_2) - I(f_1) + 1$$

$$I(f_1 \wedge f_2 \Rightarrow f_3) = I(f_1) \cdot I(f_2) \cdot I(f_3) - I(f_1) \cdot I(f_2) + 1$$

Logical Rules

KALE(Embeddings by jointly modeling Knowledge And Logic)



Applications

- In-KG Applications
- Out-of-KG Applications

Link Prediction

Link prediction is typically referred to as the task of predicting an entity that has a specific relation with another given entity. e.g., predicting h given (r, t) or t given (h, r).

This can easily be achieved by using the learned embeddings and scoring function once an embedding model has been trained on the KG. e.g., $f_r(h, t) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||_{1/2}$ if TransE has been employed for KG embedding.

Hits@n is often used to measure the performance of a knowledge graph embedding method.

Triple Classification

Triple classification consists in verifying whether an unseen triple fact (h, r, t) is true or not. e.g., (AlfredHitchcock, DirectorOf, Psycho) should be classified as a true fact while (JamesCameron, DirectorOf, Psycho) a false one.

Like link prediction, we can use the learned embeddings and scoring function once an embedding model has been trained on the KG to score the triple so as to get the plausibility. Then we can give each relation r a threshold δ_r that given triple (h, r, t), it will be predict as a true fact if $f_r(h, t) > \delta_r$.

Entity Classification

Entity classification aims to categorize entities into different semantic categories. e.g., lfredHitchcock is a Person, and Psycho a CreativeWork.

We can introduce a special relation IsA to treat entity classification as a specific link prediction task, i.e., $(x, IsA, ?)$.

Entity Resolution

Entity resolution consists in verifying whether two entities refer to the same object. e.g., Trump and current POTUS refer to the same object. And entity resolution is the task that de-duplicates such nodes.

Like entity classification, We can introduce a special relation EqualTo to treat entity classification as a specific triple classification task, i.e., $(x, EqualTo, y)$.

Question Answering[19]

Learning low-dimensional vector embeddings of words and of KG constituents, so that representations of questions and of their corresponding answers are close to each other in the embedding space.
Let q donate a question and a a candidate answer. A function $S(q, a)$, based on vector embeddings, is designed to score the similarity between the question and the answer, i.e.,

$$S(q, a) = (\mathbf{W}\Phi(q))^T (\mathbf{W}\Psi(a))$$

thus the answer $\hat{a}$ is predicted as

$$\hat{a} = \arg\max_{a \in \mathbb{A}(q)} S(q, a)$$

RecommenderSystems[20]

Among different recommendation strategies, collaborative filtering techniques which model the interaction between a user and an item as a product of their latent representations, have achieved significant success. The preference of user u_i to item e_j can be represented as

$$p(\mathbf{u}_i, \mathbf{e}_j) = \mathbf{u}_i^T \mathbf{e}_j$$

So we can add KG's information to better represent the item e_j as

$$\mathbf{e}_j = \mathbf{s}_j + \mathbf{e}'_j$$

where $\mathbf{s_j}$ is the structural representation learned from KG embedding techniques and $\mathbf{e}'_j$ is the former representation of item e_j.

References

[1]Q. Wang, Z. Mao, B. Wang and L. Guo, Knowledge Graph Embedding: A Survey of Approaches and Applications

[2]A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, Translating embeddings for modeling multi-relational data

[3]Z. Wang, J. Zhang, J. Feng, and Z. Chen, Knowledge graph embedding by translating on hyperplanes

[4]Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, Learning entity and relation embeddings for knowledge graph completion

[5]G. Ji, K. Liu, S. He, and J. Zhao, Knowledge graph completion with adaptive sparse transfer matrix

[6]S. He, K. Liu, G. Ji, and J. Zhao, Learning to represent knowledge graphs with Gaussian embedding

[7]A. Bordes, X. Glorot, J. Weston, and Y. Bengio, Joint learning of words and meaning representations for open-text semantic parsing

References

[8]M. Nickel, V. Tresp, and H.-P. Kriegel, A three-way model for collective learning on multi-relational data

[9]B. Yang, W.-T. Yih, X. He, J. Gao, and L. Deng, Embedding entities and relations for learning and inference in knowledge bases

[10]M. Nickel, L. Rosasco, and T. Poggio, Holographic embeddings of knowledge graphs

[11]A. Bordes, X. Glorot, J. Weston, and Y. Bengio, A semantic matching energy function for learning with multi-relational data

[12]R. Socher, D. Chen, C. D. Manning, and A. Y. Ng, Reasoning with neural tensor networks for knowledge base completion

[13]X. Dong, et al., Knowledge vault: A web-scale approach to probabilistic knowledge fusion

[14]Q. Liu, H. Jiang, A. Evdokimov, Z.-H. Ling, X. Zhu, S. Wei, and Y. Hu, Probabilistic reasoning via deep learning: Neural associ- ation models

References

[15]S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, Semantically smooth knowledge graph embedding
[16]Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, Modeling relation paths for representation learning of knowledge bases
[17]R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, Representation learning of knowledge graphs with entity descriptions
[18]S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, Jointly embedding knowledge graphs and logical rules
[19]A. Bordes, S. Chopra, and J. Weston, Question answering with subgraph embeddings
[20]F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, Collaborative knowledge base embedding for recommender systems