Support and Centrality: Learning Weights for Knowledge Graph Embedding Models

EKAW 2018, Nov. 2018

Gengchen Mai Krzysztof Janowicz Bo Yan

STKO Lab, University of California, Santa Barbara

Introduction

- **Knowledge Graph** (KG): a **data repository** that describes **entities** and their **relationships** across domains according to some **schema**.
- **Examples**: Google Knowledge Graph, Microsoft's Satori, Freebase, DBpedia, YAGO, and Wikidata.

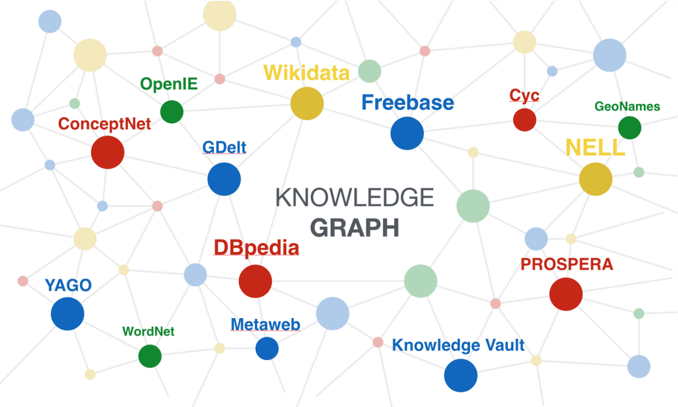
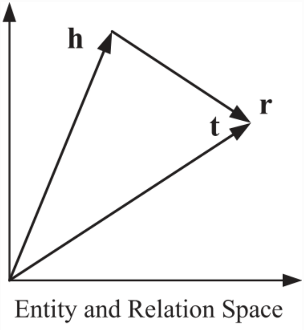


Figure From https://medium.com/@sderymail/challenges-of-knowledge-graph-part-1-d9ffe9e35214

Introduction

- **Challenge:** The symbolic representations of KGs prohibit the usage of **probabilistic models** which are widely used in many kinds of **ML applications**.
- **Knowledge Graph Embedding:** represent components of a KG including entities and relations into **continuous vectors or matrices** while preserving the structural information of the KG.



(a) TransE.

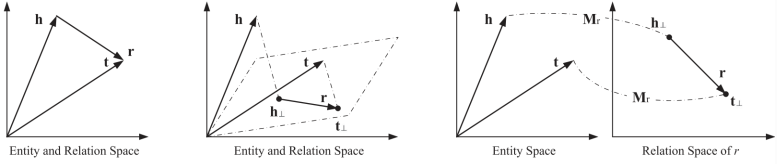
Figure from Wang et al. 2017

Introduction

- **Multiple downstream tasks**:
  - KG Completion
  - Query Expansion
  - Information Extraction
  - Information Retrieval
  - Recommender System
  - Relation Inference
  - Relation Extraction
  - Knowledge Fusion
  - Question Answering

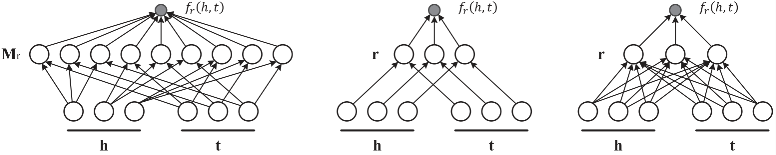
Introduction

- The major **KG Embedding** models can be classified as two categories (Wang et al. 2017):
  - **Translation-based models** (e.g. TransE, TransH, and TransR)



(a) TransE. (b) TransH. (c) TransR.

  - **Semantic matching models** (e.g. RESCAL, DisMult, and HolE).



(a) RESCAL. (b) DistMult. (c) HolE.

Introduction

- Given a knowledge graph G which contains a collection of triples/statements (h_i, r_i, t_i)
- KG embedding aims to embed entities and relations into a low-dimensional continuous vector space
- A scoring function $f_r(h, t)$ is defined on each triple (h_i, r_i, t_i) such that facts observed in the KG tend to have higher scores than those that have not been observed
  - e.g. the scoring function of TransE

$$f_r(h, t) = - \parallel \mathbf{h} + \mathbf{r} - \mathbf{t} \parallel \tag{1}$$

- The pairwise ranking loss function is usually used as the objective function to set up the learning task

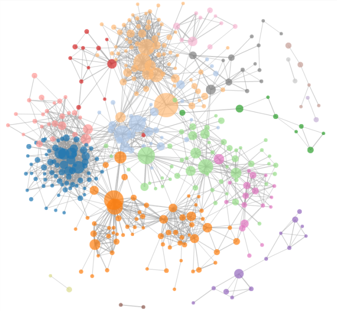
$$\mathcal{L} = \sum_{(h_i, r_i, t_i) \in S^+} \sum_{(h_i', r_i, t_i') \in S^-_{(h_i, r_i, t_i)}} [\gamma + f_r(h_i, t_i) - f_r(h_i', t_i')]_+ \tag{2}$$

Introduction

- **Problem**: Most KG embedding models treat all triple **equally**, despite the fact that their **information content**, i.e., their **contribution** to the overall graph, differers substantially.
  - Example A:
    `(:California, dbo:isPartOf, :United_States)`
  - Example B:
    `(:Gengchen_Mai, foaf:friend, :Bo_Yan)`

Introduction

- Some triples act as **foundational statements** that cannot be reconstructed from others, while most other triples can be **inferred**.
- The first kind of triples offer **support** for the second kind.
- To emphasize the **information content contribution** of each triple to the KG and to learn a suitable embedding model, each triple should be **weighted differently**. (**Core Problem**)



Information Content of Triples

How to measure IC of a triple (h_i, r_i, t_i)

- **Naive Idea**: a triple $T_i = (h_i, r_i, t_i)$ will have a higher contribution if other triples can be inferred from it.
- **IC of T_i**: If T_i is excluded from the current KG, a certain number of triples cannot be inferred from it any longer.
- **Shortcoming**:
  - **Computationally complex**: enumerating each triple and executing inferences on the entire KG
  - **Require a formal ontology**
  - **Isolated Triples (substantial)**

$$H(x) = \sum_x p(x) \log\left(\frac{1}{p(x)}\right)$$

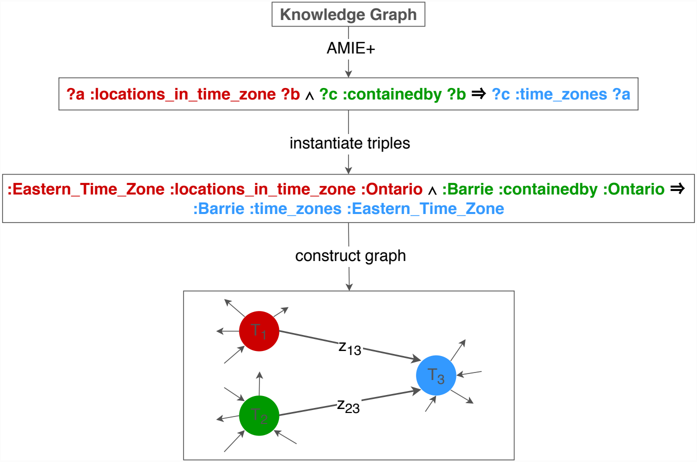
Information Content of Triples

- **Isolated Triples**: triples in a KG which can neither be used to infer any another triples nor can be inferred by any triples.
- **Naive Idea**: **Low IC**, because isolated triples cannot infer any triples and excluding them from the KG will not affect the number of inferred triples.
- **Information Theory**: **High IC**, because isolated triples cannot be compressed.
- **Alternative Method?**

Information Content of Triples

- **Rule-supported Weights Method**: measures the contribution of each triple to the global IC of the KG by investigating the **inference relationships** among these triples and use this measure to learn a **suitable KG embedding** model for the current KG
    - Rule mining
    - Rule instantiation
    - Triple inference graph construction and triple weights calculation
    - Learning a weighted KG embedding model

Information Content of Triples



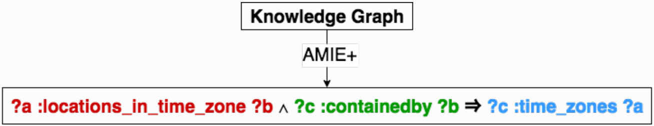
The workflow of computing the information content of each triple in a KG

Rule Mining

- Given a KG as a set of triples $S^+ = \{(h_i, r_i, t_i)\}$. For each triple (h_i, r_i, t_i), its head and tail entity are $h_i, t_i \in E$ (the set of entities) and its relation is $r_i \in L$ (the set of relations)
- **Logical rule mining**, e.g. AMIE, AMIE+ is a machine learning method to find (Horn) rules in a KG that describe the common correlations between triples.

$$R_i : B_1 \wedge B_2 \wedge \ldots \wedge B_n \Rightarrow r(x, y) \tag{3}$$

  - $B_1, B_2, \ldots, B_n, r(x, y)$: atoms in a Horn rule R_i each of which is a triple whose subject or/and object is replaced by variables.



Rule Mining

- 4 measures for mined rules quality/correctness of **AMIE+**:
  - **Frequency** f_{freq}

$$freq(R_i) = \frac{\#(instatiate(\overrightarrow{B} \Rightarrow r(x,y)))}{\#(S^+)} \tag{4}$$

  - **Head coverage** f_{hc}

$$hc(R_i) = \frac{support(\overrightarrow{B} \Rightarrow r(x,y))}{\#(r)} \tag{5}$$

    - $\#(S^+)$: the number of triples in S^+
    - $\#(r)$: the number of statements with rule head relation r
  - **Standard confidence score** (Closed-World Assumption) f_{cwa}
  - **PCA confidence score** (Partial Completeness Assumption) f_{pca}
- **3 parameters of AMIE+**:
  - **minHC**: threshold of the head coverage of the mined rules, **0.01**
  - **maxLen**: maximum rule length, **3**
  - **minConf**: threshold for the PCA confidence score, **0.1**

Rule Instantiation

- **Rule Instantiation**: variables in each atom need to be instantiated by entities in the KG such that these entities satisfy both the rule head and rule body.

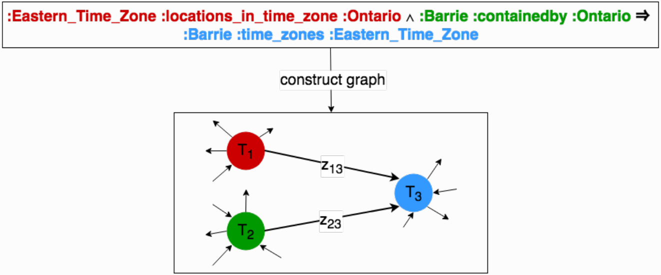
?a :locations_in_time_zone ?b ∧ ?c :containedby ?b ⇒ ?c :time_zones ?a

↓ instantiate triples

:Eastern_Time_Zone :locations_in_time_zone :Ontario ∧ :Barrie :containedby :Ontario ⇒ :Barrie :time_zones :Eastern_Time_Zone

Triple Inference Graph Construction & Weights Calculation

- Given a rule $R_h : B_1 \wedge B_2 \Rightarrow B_3$, one of its grounded rules is $GR_{hj} : T_1 \wedge T_2 \Rightarrow T_3$ with f_{freq}, f_{hc}, f_{cwa}, and f_{pca}.
- **Triple Inference Graph**: Each triple (statement) is represented as a node and each directed edge e_{ij} from node T_i to node T_j indicates that statement T_i infers statement T_j.



Triple Inference Graph Construction & Weights Calculation

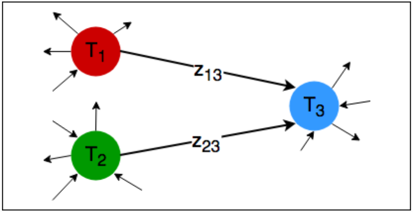
- Let GR_1, GR_2, ..., GR_k, ..., GR_r be all grounded rules which are instantiated from the mined rules from AMIE+.
- The **edge weight** z_{ij} are are derived from one of the four rule predication correctness measures f_{freq}, f_{hc}, f_{cwa}, and f_{pca}.

$$z_{ij} = \sum_{i=1}^{r} \alpha_{ik}\beta_{jk}\frac{f_k}{L_k - 1} \tag{6}$$

  - α_{ik}: an indicator function ($\alpha_{ik} = 1$ when T_i is in the rule body of GR_k; 0 otherwise)
  - β_{jk}: an indicator function ($\beta_{jk} = 1$ when T_j is the rule head of GR_k; 0 otherwise)
  - f_k: one rule predication correctness measure among f_{freq}, f_{hc}, f_{cwa}, and f_{pca}
  - L_k: the rule lengths of GR_k

Triple Inference Graph Construction & Weights Calculation

- The more incoming links T_i has, the more likely T_i is able to be inferred by other triples which implies that T_i has less information from **information theoretic compression perspective**
- **IC of T_i**: -log of the probability of inferencing a triple (statement) in the triple inference graph

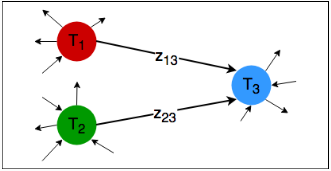


Triple Inference Graph Construction & Weights Calculation

- **Edge weighted PageRank**: providing a teleport probability which allows the random walker to jump to a random node in the graph with a certain probability at each time step
- **Isolated Triples**: have a lower inferencing probability, thus possessing richer information content

$$w_i = -log_2(PR_i) \times \frac{\#(S^+)}{\sum -log_2(PR_i)} \quad (7)$$

  - PR_i: PageRank value of each node/triple
  - $\frac{\#(S^+)}{\sum -log_2(PR_i)}$: a normalization factor to make the mean value of result triple weights to be 1.0



Learning A Weighted Knowledge Graph Embedding Model

- A weighted KG embedding model based on multiple existing models (**TransE**, **TransR**, and **HolE**)
- Given observed triples S^{+}, the scoring function $f_r(h,t)$ of $T_i = (h_i, r_i, t_i) \in S^{+}$, and triple weight w_i
- **For any translation-based models or semantic matching models** as long as they use **pairwise ranking loss functions** to set up the learning task

$$\mathcal{L} = \sum_{(h_i,r_i,t_i)\in S^{+}} \sum_{(h_i',r_i,t_i')\in S^{-}_{(h_i,r_i,t_i)}} [\gamma + w_i(f_r(h_i,t_i) - f_r(h_i',t_i'))]_{+} \quad (8)$$

  - $f_r(h_i,t_i) - f_r(h_i',t_i')$ is a measure of the distinction degree or distance for T_i and T_i'
  - Different triples have different IC, the loss function should consider T_i more if it has larger IC

Link Prediction Task

- **Dataset**:
  - **WN18**: extracted from **WordNet** in which entities are word senses and relations correspond to the lexical relationships between word senses.
  - **FB15K**: a subset extracted from **Freebase** in which entities have at least 100 mentions in Freebase and also appear in Wikilinks dataset.

Spearman's correlation coefficients between different weights on **WN18**

ρ	freq	hc	cwa	pca
freq	1	0.704	0.899	0.879
hc	-	1	0.790	0.779
cwa	-	-	1	0.889
pca	-	-	-	1

Spearman's correlation coefficients between different weights on **FB15K**

ρ	freq	hc	cwa	pca
freq	1	0.788	0.877	0.855
hc	-	1	0.805	0.848
cwa	-	-	1	0.972
pca	-	-	-	1

Link Prediction Task

- A weighted KG embedding model based on multiple existing models (**TransE**, **TransR**, and **HolE**): **TransE-RW**, **TransR-RW** and **HolE-RW**
- **Evaluation Metrics**:
  - **Mean Rank**: a **lower** Mean Rank indicates a better performance.
  - **Mean Reciprocal Rank (MRR)**: a **higher** MRR indicates a better performance.
  - **HIT@K** where K can be 1, 3, 10: a **higher** HIT@K indicates a better performance.

Link Prediction Task

- Evaluation of **TransE-RW**, **TransR-RW**

Table 3. Link Prediction Result of *TransE-RW* and *TransR-RW* (*unif* indicates using random negative sampling method; *bern* indicates using the method proposed by [12])

DataSet	WN18						FB15K					
Metric	Mean Rank		MRR		HIT@10		Mean Rank		MRR		HIT@10	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE [1]	263	251	-	-	75.4	89.2	243	125	-	-	34.9	47.1
TransM [3]	293	281	-	-	75.7	85.4	197	94	-	-	44.6	55.2
TransH (unif.) [12]	318	303	-	-	75.4	86.7	211	84	-	-	42.5	58.5
TransH (bern.) [12]	401	388	-	-	73.0	82.3	212	87	-	-	45.7	64.4
TransR (unif.) [5]	232	**219**	-	-	78.3	91.7	226	78	-	-	43.8	65.5
TransR (bern.) [5]	238	225	-	-	**79.8**	92.0	198	77	-	-	48.2	68.7
TransE-RW$_{freq}$ (unif.)	298	286	0.361	0.487	77.8	91.4	216	69	0.225	0.422	46.8	69.4
TransE-RW$_{freq}$ (bern.)	**231**	**219**	**0.391**	**0.516**	78.1	91.0	243	144	0.252	0.424	49.4	67.8
TransE-RW$_{hc}$ (unif.)	266	253	0.371	0.496	77.1	90.7	212	**67**	0.226	0.420	46.8	68.8
TransE-RW$_{hc}$ (bern.)	272	260	0.377	0.495	77.3	89.8	235	134	**0.258**	0.444	**50.2**	69.6
TransE-RW$_{cwa}$ (unif.)	281	269	0.359	0.483	77.0	90.8	213	**67**	0.225	0.418	47.0	69.0
TransE-RW$_{cwa}$ (bern.)	277	265	0.378	0.486	75.4	86.8	245	149	0.241	0.386	47.2	63.4
TransE-RW$_{pca}$ (unif.)	292	279	0.353	0.472	76.2	89.6	217	71	0.227	0.423	47.1	**69.7**
TransE-RW$_{pca}$ (bern.)	318	305	0.375	0.484	75.4	86.9	232	132	0.256	**0.445**	50.1	**69.7**
TransR-RW$_{freq}$ (unif.)	351	336	0.319	0.448	77.8	**93.4**	230	76	0.173	0.356	44.2	67.1
TransR-RW$_{freq}$ (bren.)	320	306	0.326	0.442	78.0	92.0	**196**	74	0.230	0.426	48.3	69.3

Link Prediction Task

- Evaluation of **HolE-RW**

Table 5. Link prediction results of *HolE-RW*

DataSet	WN18					FB15K				
Metric	MRR		HIT			MRR		HIT		
	Filter	Raw	1	3	10	Filter	Raw	1	3	10
HolE	0.938	0.616	93	**94.5**	94.9	0.524	0.232	40.2	61.3	73.9
ComplEx	0.941	0.587	**93.6**	**94.5**	94.7	0.692	0.242	59.9	75.9	**84**
HolE-RW$_{freq}$ (unif.)	0.91	0.624	89.5	92.1	93.4	0.702	0.699	69.0	70.0	72.1
HolE-RW$_{freq}$ (bern.)	0.913	0.645	89.5	92.7	94.0	0.675	0.671	65.8	67.5	70.6
HolE-RW$_{hc}$ (unif.)	0.932	0.688	92.3	93.6	94.5	0.646	0.64	62.5	64.4	68.2
HolE-RW$_{hc}$ (bern.)	0.922	0.686	90.8	93.2	94.1	0.705	0.699	69.2	70.4	72.6
HolE-RW$_{cwa}$ (unif.)	**0.942**	**0.693**	93.5	**94.5**	**95.5**	0.695	0.692	68.3	69.3	71.6
HolE-RW$_{cwa}$ (bern.)	0.922	0.684	91.0	93.2	93.9	**0.791**	**0.788**	**78.1**	**79.0**	81.1
HolE-RW$_{pca}$ (unif.)	0.931	0.686	92.3	93.7	94.5	0.635	0.63	61.5	63.4	67.1
HolE-RW$_{pca}$ (bern.)	0.926	0.688	91.4	93.5	94.4	0.756	0.754	74.6	75.4	77.3

Conclusion

- We propose a data-driven approach to measure the **information content of each triple** with respect to the whole knowledge graph by using **rule mining** and **PageRank**.
- We show how to compute **triple-specific weights** to improve the performance of **three KG embedding models** (TransE, TransR and HolE).
- **Link prediction tasks** on FB15K and WN18 show the effectiveness of our weighted KG embedding model over other more complex models.
  - For FB15K, TransE-RW outperforms models such as TransE, TransM, TransH, and TransR by at least **12.98%** for *Mean Rank* and at least **1.45%** for *HIT@10*.
- Our weighted KG embedding framework can be applied to **any translation-based models or semantic matching models** to improve their performance as long as they use **pairwise ranking loss functions** to set up the learning task.

Future Work

- We need to **improve the efficient of the rule mining algorithm** in order to apply our method to a larger knowledge graph.
- We will deploy our weighting method to **other KG embedding models** such as TransH.
- We will explore methods to **automatically learn the weights** during the embedding model training — similar to attention mechanisms in neural networks.
- We will explore the methods to **learn embeddings for datatype properties**.