Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs

Deepak Nathani* **Jatin Chauhan*** **Charu Sharma*** **Manohar Kaul**
Department of Computer Science and Engineering, IIT Hyderbad
{deepakn1019,chauhanjatin100,charusharma1991}@gmail.com, mkaul@iith.ac.in

Manohar Kaul

I am an Assistant Professor at IIT Hyderabad and my current re
intersection of geometry, applied algebraic topology, Topologica
Machine Learning.

Contact

Assistant Professor
Dept. of Computer Science
IIT Hyderabad India
mkaul (at) iith (dot) ac (dot) in
DBLP, Google Scholar

Deepak Nathani

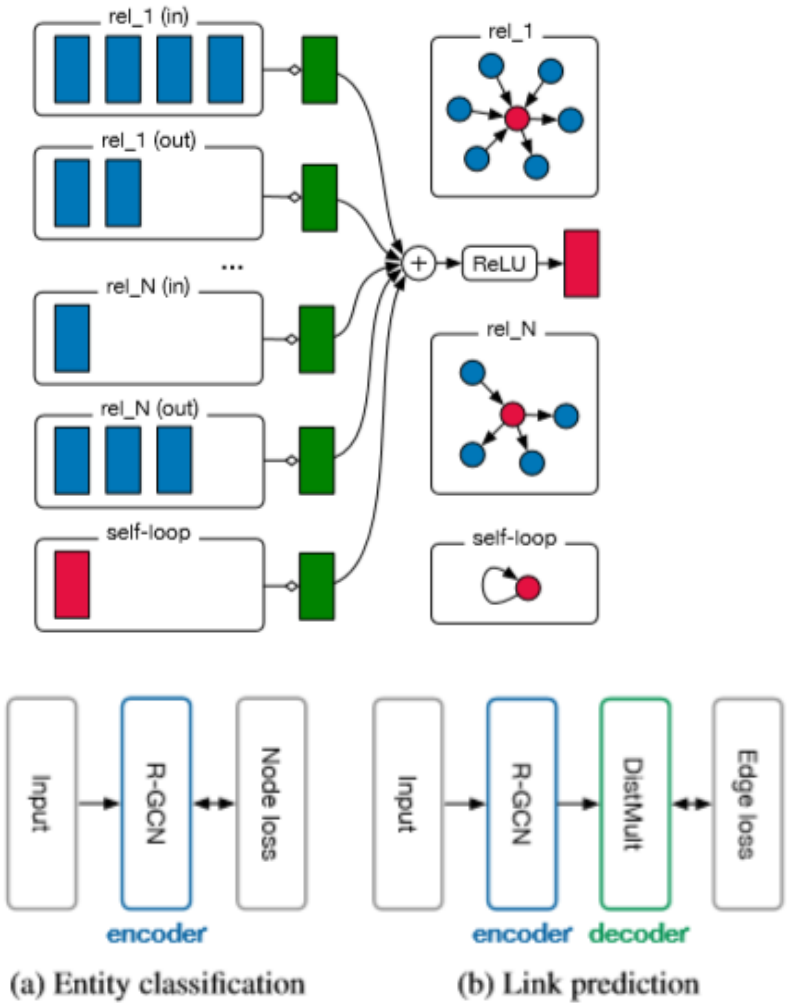
Hello! I am Deepak Nathani. I recently graduated from **Indian Institute of Technology, Hyderabad** with a **B.Tech** degree in Mechanical Engineering. Interestingly, I will also get a degree in Computer Science which was my second major. Currently I am exploring the fields of **Machine Learning, Deep Learning and Natural Language Processing**.
During my time as an undergraduate, I worked on various research problems with **Dr. Manohar Kaul**. I have also worked as a Summer Research Intern at IBM Research Labs, India where I worked under **Dr. Sumit Bhatia** and **Dr. Bapi Chatterjee**.

Among other things, I enjoy playing games, listening to music, reading books and most important of them all, I love food. 😉

Introduction

- Knowledge Graphs completion: (h , r , ?)
  - Suffer from incompleteness in form of missing entities and relations.
  - knowledge graph embedding is one of the widely used approaches. (TransE, ConvE…)
- Shortcoming:
  - treat triples independently
  - fail to cover the complex and hidden information that is inherently implicit in the local neighborhood surrounding a triple.
- Main contributions:
  - The first to learn new graph attention based embeddings that specifically target relation prediction on KGs.
  - Generalize and extend graph attention mechanisms to capture both entity and relation features in a multi-hop neighborhood of a given entity.
  - Experimental results indicate a clear and substantial improvement over state-of-the-art relation prediction methods.

Related work——RGCN



- GCN: $h_i^{(l+1)} = \sigma\left(\sum_{m \in \mathcal{M}_i} g_m(h_i^{(l)}, h_j^{(l)})\right)$
  - $h_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ is the hidden state of node v_i. $\mathcal{M}_i$ denotes the set of incoming messages for node (incoming edges). $g_m(\cdot,\cdot)$ is simply a linear transformation.
- RGCN: $h_i^{(l+1)} = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}\right)$
  - $\mathcal{N}_i^r$ denotes the set of neighbor indices of node i under relation $r \in \mathcal{R}$. $c_{i,r}$ is a problem-specific normalization constant that can either be learned or chosen in advance (such as $c_{i,r} = |\mathcal{N}_i^r|$)
  - Loss function: $\mathcal{L} = -\frac{1}{(1+\omega)|\hat{\mathcal{E}}|} \sum_{(s,r,o,y) \in \mathcal{T}} y \log l\big(f(s,r,o)\big) + (1-y) \log\big(1 - l\big(f(s,r,o)\big)\big)$,

Graph Attention Networks

- Shortcomings of GCNs:
  - gather information from the entity's neighborhood and all neighbors contribute equally in the information passing.
- GAT:
  - Attention value of the edge (ei, ej) : $e_{ij} = a(\mathbf{W}\vec{x_i}, \mathbf{W}\vec{x_j})$
  - output of a Graph Attention Layer: $\vec{x_i'} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} \vec{x_j}\right)$
  - Multi-head attention process of concatenating K attention heads: $\vec{x_i'} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{x_j}\right)$
    - α_{ij}^k are normalized attention coefficients of edge (e_i, e_j) calculated by the k-th attention mechanism

Relations are Important

- In KGs, entities play different roles depending on the relation they are associated with.
  - entity Christopher Nolan appears in two different triples assuming the roles of a brother and a director.
- GAT in KG:
  - Two embedding matrices as input: $\mathbf{H} \in \mathbb{R}^{N_e \times T} \quad \mathbf{G} \in \mathbb{R}^{N_r \times P}$
  - vector representation of a triple (e_i, r_k, e_j) : $\vec{c_{ijk}} = \mathbf{W}_1[\vec{h}_i \| \vec{h}_j \| \vec{g}_k]$
  - Absolute attention value of the triple : $b_{ijk} = \mathbf{LeakyReLU}\left(\mathbf{W}_2 c_{ijk}\right)$
  - relative attention values : $\alpha_{ijk} = \mathrm{softmax}_{jk}(b_{ijk}) = \frac{\exp(b_{ijk})}{\sum_{n \in \mathcal{N}_i} \sum_{r \in \mathcal{R}_{in}} \exp(b_{inr})}$
  - new embedding of the entity : $\vec{h_i'} = \sigma\left(\sum_{j \in \mathcal{N}_i} \sum_{k \in \mathcal{R}_{ij}} \alpha_{ijk} \vec{c_{ijk}}\right)$

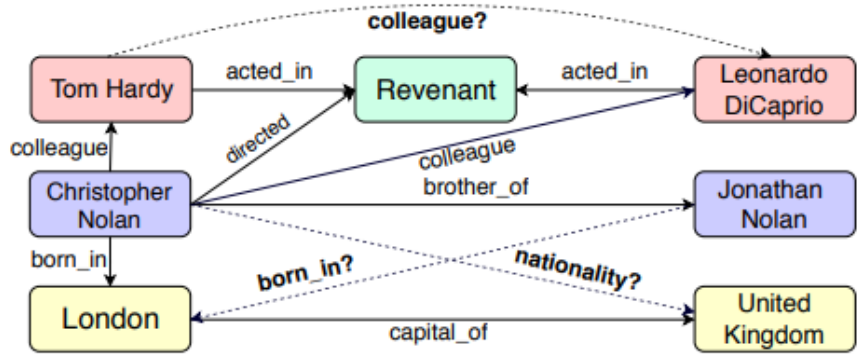
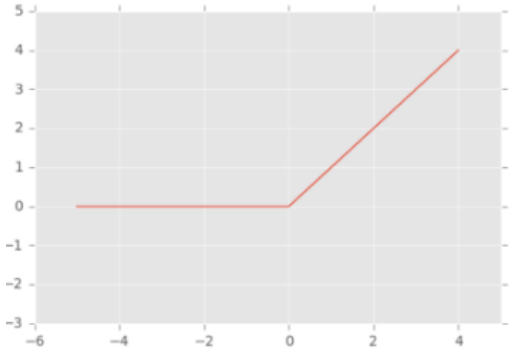


Figure 1: Subgraph of a knowledge graph contains actual relations between entities (solid lines) and inferred relations that are initially hidden (dashed lines).

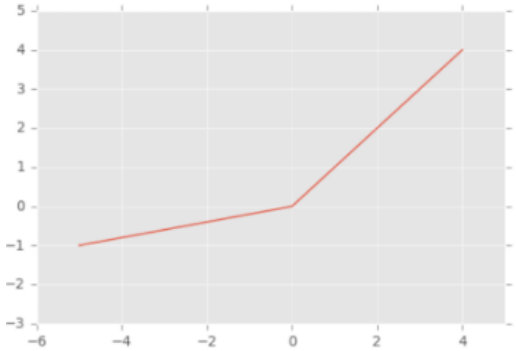
- ReLU

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases}$$



- LeakyReLU

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \lambda x, & \text{if } x \leq 0 \end{cases}$$



Relations are Important

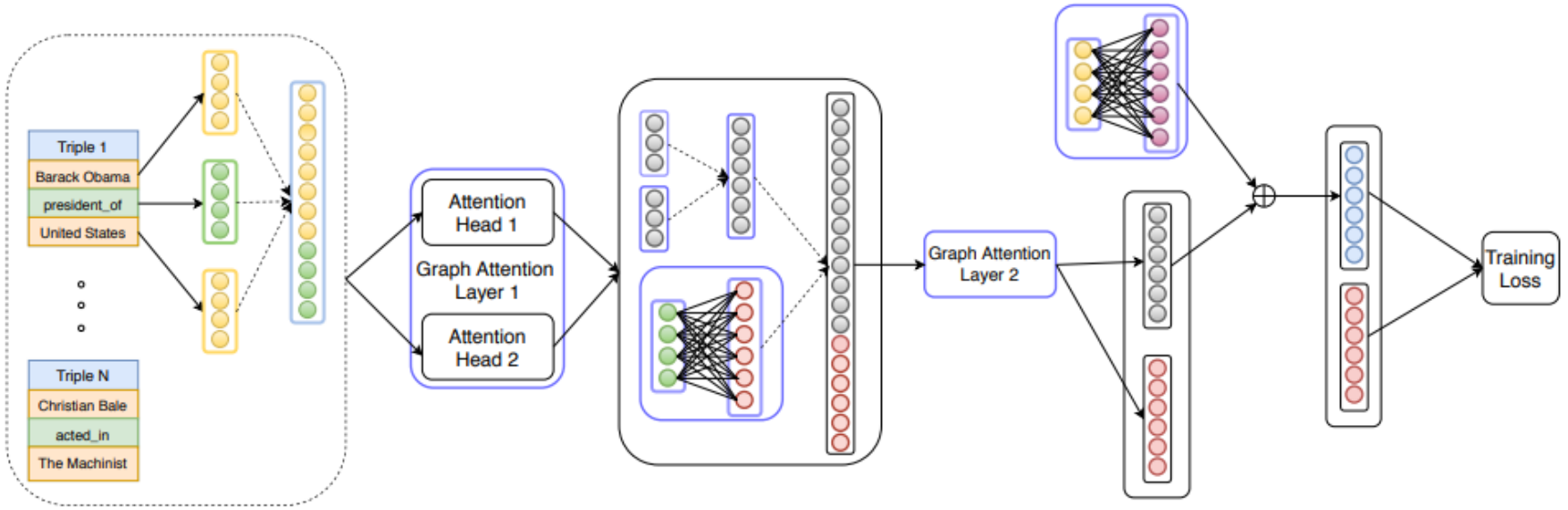


Figure 4: This figure shows end-to-end architecture of our model. Dashed arrows in the figure represent concatenation operation. Green circles represents initial entity embedding vectors and yellow circles represents initial relation embedding vectors.

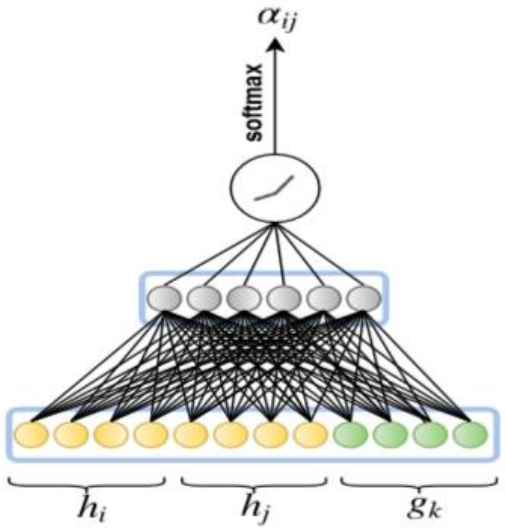


Figure 3: Attention Mechanism

- Multi-head attention: $$\vec{h}_i' = \Big\Vert_{m=1}^{M} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ijk}^m c_{ijk}^m\right)$$
- final layer employ averaging to get final embedding: $$\vec{h}_i' = \sigma\left(\frac{1}{M}\sum_{m=1}^{M}\sum_{j\in\mathcal{N}_i}\sum_{k\in\mathcal{R}_{ij}} \alpha_{ijk}^m c_{ijk}^m\right)$$
- Solve the problem that entities lose their initial embedding information $$\mathbf{H}'' = \mathbf{W}^E\mathbf{H}^t + \mathbf{H}^f$$

Auxiliary relation

- Extend the notion of an edge to a directed path by introducing an auxiliary relation for n-hop neighbors between two entities.
  - embedding auxiliary relation: the summation of embeddings of all the relations in the path.
  - first layer: all entities capture information from their direct in-flowing neighbors
  - second layer: *U.S* gathers information from entities *Barack Obama, Ethan Horvath, Chevrolet,* and *Washington D.C*, which already possess information about their neighbors *Michelle Obama* and *Samuel L. Jackson*, from a previous layer.

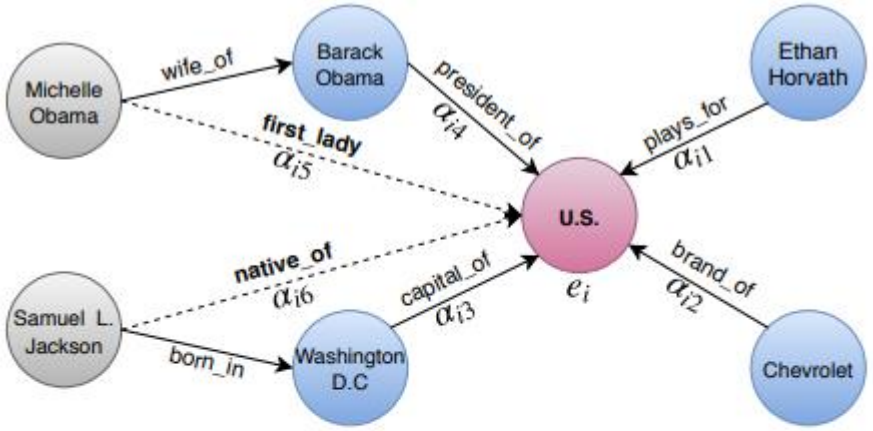


Figure 2: This figure shows the aggregation process of our graph attentional layer. α_{ij} represents relative attention values of the edge. The dashed lines represent an *auxiliary* edge from a n-hop neighbors, in this case $n = 2$.

Training Objective

- translational scoring function: $\vec{h}_i + \vec{g}_k \approx \vec{h}_j$
- Loss function:

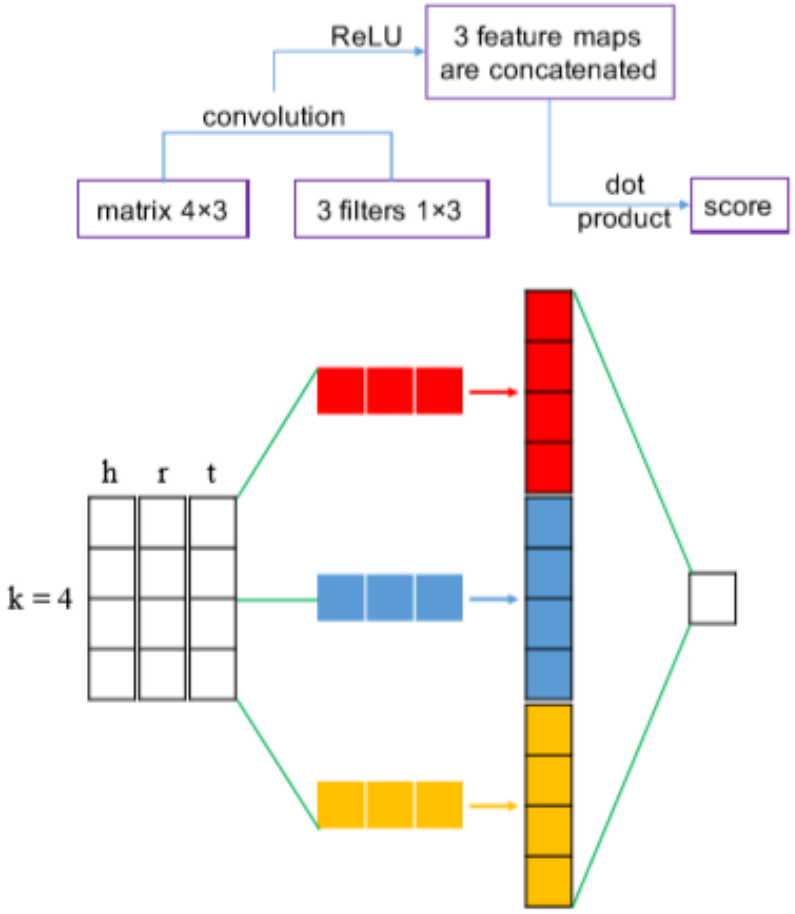
$$L(\Omega) = \sum_{t_{ij} \in S} \sum_{t'_{ij} \in S'} \max\{d_{t'_{ij}} - d_{t_{ij}} + \gamma, 0\} \qquad d_{t_{ij}} = \|\vec{h}_i + \vec{g}_k - \vec{h}_j\|_1$$

  - learn entity and relation embeddings to minimize the L1-norm dissimilarity measure.
  - Invalid triples $S' = \underbrace{\{t^k_{i'j} \mid e'_i \in \mathcal{E} \setminus e_i\}}_{\text{replace head entity}} \cup \underbrace{\{t^k_{ij'} \mid e'_j \in \mathcal{E} \setminus e_j\}}_{\text{replace tail entity}}$

Decoder——ConvKB

- ConvKB:
  - each triple (head, relation, tail) is represented as a 3-column matrix.
  - This 3-column matrix is then fed to a convolution layer to generate different feature maps.
  - These feature maps are then concatenated into a single feature vector which multiplied with a weight vector via a dot product to return a score.
- Decoder: $$f(t_{ij}^k) = \left(\Big\Vert_{m=1}^{\Omega} \mathrm{ReLU}([\vec{h}_i, \vec{g}_k, \vec{h}_j] * \omega^m) \right).\mathbf{W}$$
  - soft-margin loss : $$\mathcal{L} = \sum_{t_{ij}^k \in \{S \cup S'\}} \log(1+\exp(l_{t_{ij}^k}.f(t_{ij}^k))) + \frac{\lambda}{2}\|\mathbf{W}\|_2^2$$

$$\text{where } l_{t_{ij}^k} = \begin{cases} 1 & \text{for } t_{ij}^k \in S \\ -1 & \text{for } t_{ij}^k \in S' \end{cases}$$



Datasets & Protocol

Dataset	# Entities	# Relations	# Edges				Mean in-degree	Median in-degree
			Training	Validation	Test	Total		
WN18RR	40,943	11	86,835	3034	3134	93,003	2.12	1
FB15k-237	14,541	237	272,115	17,535	20,466	310,116	18.71	8
NELL-995	75,492	200	149,678	543	3992	154,213	1.98	0
Kinship	104	25	8544	1068	1074	10,686	82.15	82.5
UMLS	135	46	5216	652	661	6529	38.63	20

- Negative sampling:
  - Randomly sample equal number of invalid triples from both the sets to ensure robust performance on detecting both head and tail entity.
- Embedding initialization: TransE
- Two-step training procedure:
  - generalized GAT to encode information about the graph entities and relations.
  - train a decoder model like ConvKB to perform the relation prediction task.
- Evaluation Protocol: filtered setting

Results and Analysis

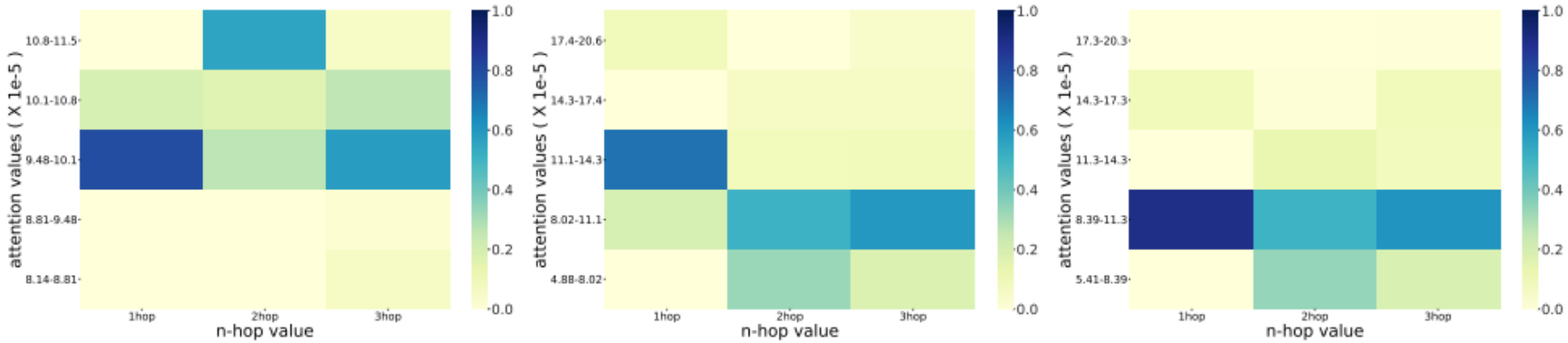
	WN18RR					FB15K-237				
			Hits@N					Hits@N		
	MR	MRR	@1	@3	@10	MR	MRR	@1	@3	@10
DistMult (Yang et al., 2015)	7000	0.444	41.2	47	50.4	512	0.281	19.9	30.1	44.6
ComplEx (Trouillon et al., 2016)	7882	0.449	40.9	46.9	53	546	0.278	19.4	29.7	45
ConvE (Dettmers et al., 2018)	4464	**0.456**	**41.9**	47	53.1	245	0.312	22.5	34.1	49.7
TransE (Bordes et al., 2013)	2300	0.243	4.27	44.1	53.2	323	0.279	19.8	37.6	44.1
ConvKB (Nguyen et al., 2018)	**1295**	0.265	5.82	44.5	55.8	216	0.289	19.8	32.4	47.1
R-GCN (Schlichtkrull et al., 2018)	6700	0.123	20.7	13.7	8	600	0.164	10	18.1	30
Our work	1940	0.440	36.1	**48.3**	**58.1**	**210**	**0.518**	**46**	**54**	**62.6**

Table 2: Experimental results on WN18RR and FB15K-237 test sets. Hits@N values are in percentage. The best score is in **bold** and second best score is underlined.

	NELL-995					Kinship				
			Hits@N					Hits@N		
	MR	MRR	@1	@3	@10	MR	MRR	@1	@3	@10
DistMult (Yang et al., 2015)	4213	0.485	40.1	52.4	61	5.26	0.516	36.7	58.1	86.7
ComplEx (Trouillon et al., 2016)	4600	0.482	39.9	52.8	60.6	2.48	0.823	73.3	89.9	97.11
ConvE (Dettmers et al., 2018)	3560	0.491	40.3	53.1	61.3	2.03	0.833	73.8	91.7	**98.14**
TransE (Bordes et al., 2013)	2100	0.401	34.4	47.2	50.1	6.8	0.309	0.9	64.3	84.1
ConvKB (Nguyen et al., 2018)	**600**	0.43	37.0	47	54.5	3.3	0.614	43.62	75.5	95.3
R-GCN (Schlichtkrull et al., 2018)	7600	0.12	8.2	12.6	18.8	25.92	0.109	3	8.8	23.9
Our work	965	**0.530**	**44.7**	**56.4**	**69.5**	**1.94**	**0.904**	**85.9**	**94.1**	98

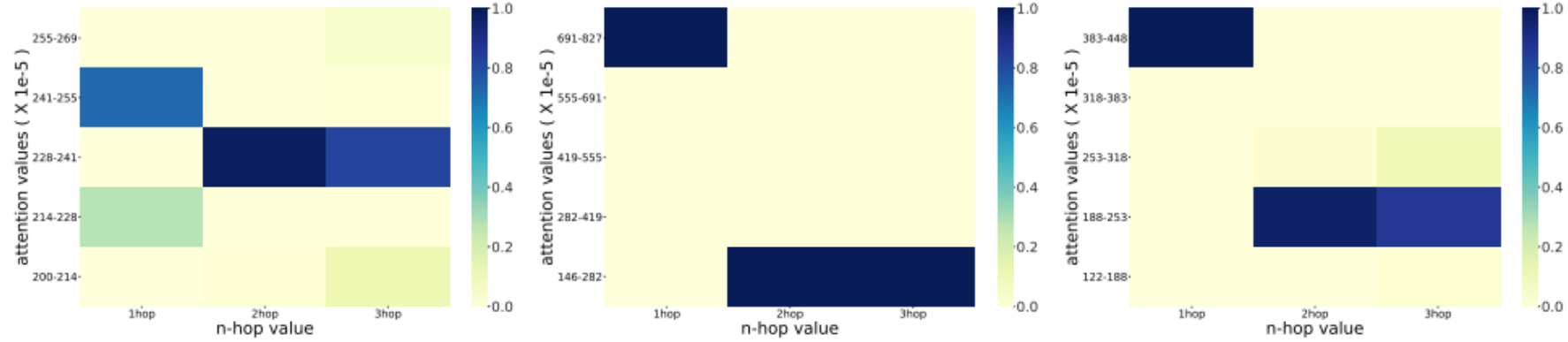
Table 3: Experimental results on NELL-995 and Kinship test sets. Hits@N values are in percentage. The best score is in **bold** and second best score is underlined.

Attention Values vs Epoch



(a) Epoch 0 (b) Epoch 1200 (c) Epoch 2400

Figure 5: Learning process of our model on FB15K-237 dataset. Y-axis represents attention values $\times 1e^{-5}$.



(a) Epoch 0 (b) Epoch 1800 (c) Epoch 3600

Figure 6: Learning process of our model on WN18RR dataset. Y-axis represents attention values $\times 1e^{-5}$

PageRank Analysis & Ablation Study

Dataset	PageRank	Relative Increase
NELL-995	1.32	0.025
WN18RR	2.44	-0.01
FB15k-237	6.87	0.237
UMLS	740	0.247
Kinship	961	0.388

Table 4: Mean PageRank $\times 10^{-5}$ vs relative increase in MRR wrt. DistMult.

- Anomaly: in case of NELL-995 versus WN18RR and attribute this to the highly sparse and hierarchical structure of WN18RR which poses as a challenge to our method that does not capture information in a top-down recursive fashion.

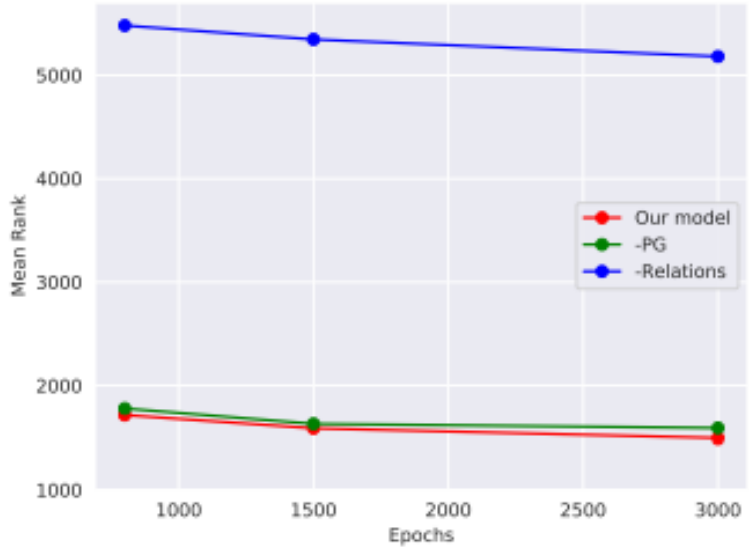


Figure 7: Epochs vs Mean Rank for our model and two ablated models on *NELL-995*. −PG (green) represents the model after removing n-hop auxiliary relations or *path generalization*, −Relations (blue) represents model without taking relations into account and Our model (red) represents the entire model.

Conclusion

- This paper proposes a novel graph attention-based approach for relation prediction.
- Improves over the state-of-the-art models by significant margins.
- Generalize and extend graph attention mechanisms to capture both entity and relation features in a multi-hop neighborhood of a given entity.

Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs

Lingbing Guo, Zequn Sun, Wei Hu*

Nanjing University, China

* Corresponding author: whu@nju.edu.cn

ICML'19, June 9–15, Long Beach, CA, USA

Wei Hu

Recent update: 2019-7-2

Wei Hu

Associate Professor, PhD Supervisor

Department of Computer Science & Technology
State Key Laboratory for Novel Software Technology
Nanjing University

email: whu at nju.edu.cn
phone: +86 (0)25 8968-1523

Lingbing Guo Research

popul

DSKG: A Deep Sequential Model for Knowledge Graph Completion

Knowledge graph (KG) completion aims to fill the missing facts in a KG,...

10/30/2018 · by Lingbing Guo, et al. · 0 · share

Recurrent Skipping Networks for Entity Alignment

We consider the problem of learning knowledge graph (KG) embeddings...

11/06/2018 · by Lingbing Guo, et al. · 0 · share

Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs

We study the problem of knowledge graph (KG) embedding. A widely-...

05/13/2019 · by Lingbing Guo, et al. · 0 · share

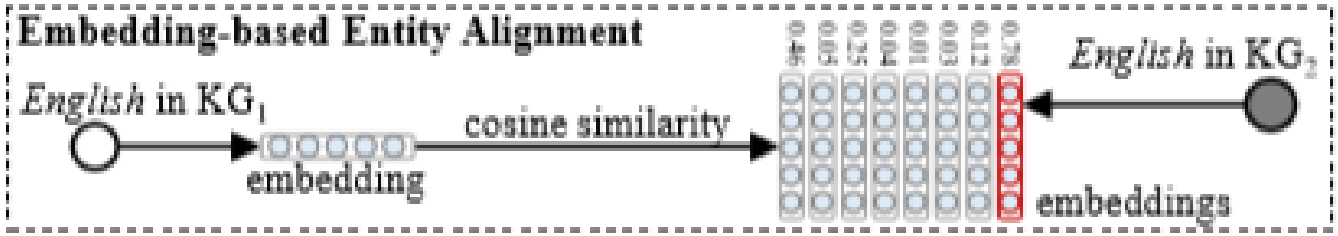
Multi-view Knowledge Graph Embedding for Entity Alignment

We study the problem of embedding-based entity alignment between...

06/06/2019 · by Qingheng Zhang, et al. · 0 · share

Knowledge graphs

- **Knowledge graphs** (KGs) store a wealth of structured facts about the real world
  - A fact (s, r, o): subject entity, relation, object entity
- KGs are far from complete and two important tasks are proposed
  1. **Entity alignment:** find entities in **different KGs** denoting the same real-world object



  2. **KG completion:** complete missing facts in a **single KG**
     - E.g., predict ? in (*Tim Berners-Lee*, *employer*, ?) or (?, *employer*, *W3C*)

Challenges

- For KG embedding, existing methods largely focus on learning from **relational triples** of entities
- Triple-level learning has two major limitations
  - **Low expressiveness**
    - Learn entity embeddings from a fairly local view (i.e., 1-hop neighbors)
    - Insufficient for multi-mapping or long-tail entities
  - **Inefficient information propagation**
    - Only use triples to deliver semantic information within/across KGs
    - Entity alignment task rely on seed alignment

Learning to exploit long-term relational dependencies

- A relational path is an **entity-relation chain**, where entities and relations appear alternately

$$United\ Kingdom \rightarrow country^{-} \rightarrow Tim\ Berners\text{-}Lee \rightarrow employer \rightarrow W3C$$

(reverse)

- RNNs perform well on sequential data $\mathbf{h}_t = \tanh(\mathbf{W}_h\mathbf{h}_{t-1} + \mathbf{W}_x\mathbf{x}_t + \mathbf{b}),$
  - **Limitations** to leverage RNNs to model relational paths
    1. A relational path have two different types: "entity" and "relation"
       - Always appear in an alternating order
    2. A relational path is constituted by triples, but these basic structure units are overlooked by RNNs

Recurrent skipping networks

- A conditional skipping mechanism allows RSNs to **shortcut** the current input entity to let it **directly** participate in predicting its object entity

$$\mathbf{h}'_t = \begin{cases} \mathbf{h}_t & x_t \in \mathcal{E} \\ \mathbf{S}_1\mathbf{h}_t + \mathbf{S}_2\mathbf{x}_{t-1} & x_t \in \mathcal{R} \end{cases}$$

- $\mathbf{h}'_t$ denotes the output hidden state of the RSN;
- $\mathbf{h}_t$ denotes the corresponding RNN output
- S1, S2 are the weight matrices, and their parameters are shared at different time steps.

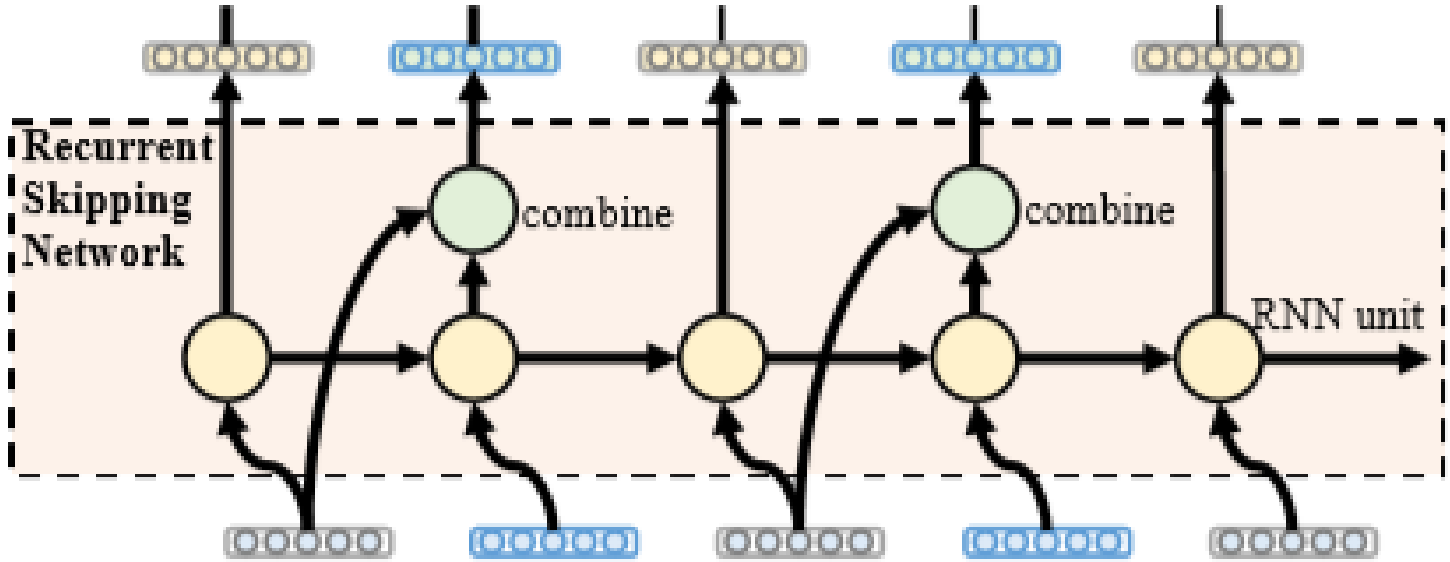


Figure 1. Example of RSNs with a 2-hop relational path

Tri-gram residual learning

- Residual learning
  - Let $F(\boldsymbol{x})$ be an original mapping, and $H(\boldsymbol{x})$ be the expected mapping
  - Compared to directly optimizing $F(\boldsymbol{x})$ to fit $H(\boldsymbol{x})$, it is easier to optimize $F(\boldsymbol{x})$ to fit residual part $H(\boldsymbol{x})$
    - An extreme case, $H(\boldsymbol{x}) = \boldsymbol{x}$

(*United Kingdom*, *country*$^-$, *Tim Berners-Lee*, *employer*, *W3C*)	
Models	Optimize $F([\cdot], employer)$ as
RNNs	$F([\cdot], employer) \coloneqq W3C$
RRNs	$F([\cdot], employer) \coloneqq W3C - [\cdot]$
RSNs	$F([\cdot], employer) \coloneqq W3C - Tim\ Berners{-}Lee$
$[\cdot]$ denotes context (*United Kingdom*, *country*$^-$, *Tim Berners-Lee*)	

- **Tri-gram** residual learning
  - $United\ Kingdom \rightarrow country^- \rightarrow Tim\ Berners\text{-}Lee \rightarrow employer \rightarrow W3C$
  - Compared to directly learning to predict *W3C* by *employer* and its mixed context, it is easier to learn the residual part between *W3C* and *Tim Berners-Lee*
    - Because they forms a triple, and we should not overlook the triple structure in the paths

Biased Random Walks

- Conventional random walks: uniform distribution
  - The unbiased random walks obtain the probability distribution of next entities.

$$\Pr(e_{i+1} \mid e_i) = \begin{cases} \frac{\pi_{e_i \to e_{i+1}}}{Z} & \exists r \in \mathcal{R}: (e_i, r, e_{i+1}) \in \mathcal{T} \\ 0 & \text{otherwise} \end{cases}$$

- Biased random walks:
  - leverage the idea of second-order random walks.
  - Depth bias in one KG:

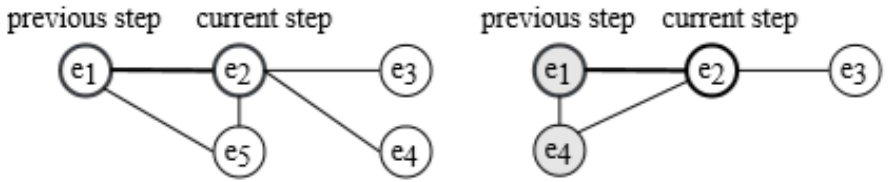
$$\mu_d(e_{i-1}, e_{i+1}) = \begin{cases} \alpha & d(e_{i-1}, e_{i+1}) = 2 \\ 1-\alpha & d(e_{i-1}, e_{i+1}) < 2 \end{cases}$$

  - Cross-KG bias:

$$\mu_c(e_{i-1}, e_{i+1}) = \begin{cases} \beta & kg(e_{i-1}) \neq kg(e_{i+1}) \\ 1-\beta & \text{otherwise} \end{cases}$$

  - Combine the depth and cross-KG biases:

$$\mu(e_{i-1}, e_{i+1}) = \mu_d(e_{i-1}, e_{i+1}) \times \mu_c(e_{i-1}, e_{i+1})$$



(a) Depth-biased random walk (b) KG-biased random walk

Figure 2. Samples of biased random walks. For simplicity, we reduce a KG as an undirected graph by merging relations and their corresponding reversed ones. e_2 is the current entity that we now stand on and e_1 is the previous one.

Algorithm 1 Biased random walk sampling

Input: Triple set $\mathcal{T}$, depth bias α, cross-KG bias β, sampling times n, max length l
Obtain biased transition probability matrices M_d, M_c;
for $i := 1$ **to** n **do**
  for each triple $(s, r, o) \in \mathcal{T}$ **do**
    $p := s \to r \to o$
    repeat
      Look up M_d, M_c and compute normalized transition probability distribution p_o of o;
      Sample next entity e from p_o;
      Sample a relation r' between o and e;
      $p := p \to r' \to e$;
    until length$(p) \geq l$;
  end for
end for

Architecture

- An **end-to-end** framework
  1. **Biased random walk sampling**
     - Deep paths carry more relational dependencies than triples
     - Cross-KG paths deliver alignment information between KGs
  2. **Recurrent skipping network**
  3. **Type-based noise contrastive estimation**
     - Evaluate loss in an optimized way

$$\mathcal{L} = -\sum_{t=1}^{T-1}\left(\log\sigma(\mathbf{h}'_t\cdot\mathbf{y}_t) + \sum_{j=1}^{k}\mathbb{E}_{\tilde{y}_j\sim Q(\tilde{y})}\left[\log\sigma(-\mathbf{h}'_t\cdot\tilde{\mathbf{y}}_j)\right]\right)$$

- If the current target is an entity, we draw negative samples from the noise probability distribution of entities.
- $Q(\tilde{y}) \propto q(\tilde{y})^{\frac{3}{4}}$, where $q(\tilde{y})$ is the frequency of $\tilde{y}$ appearing in KGs.

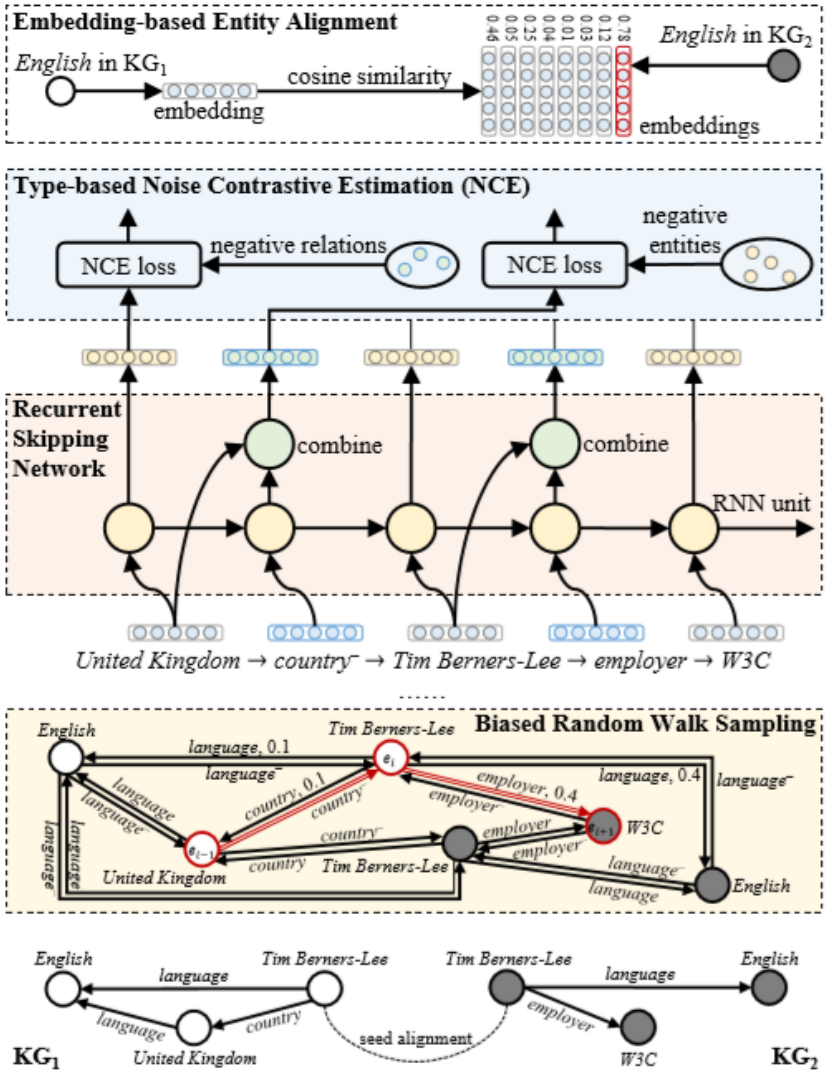


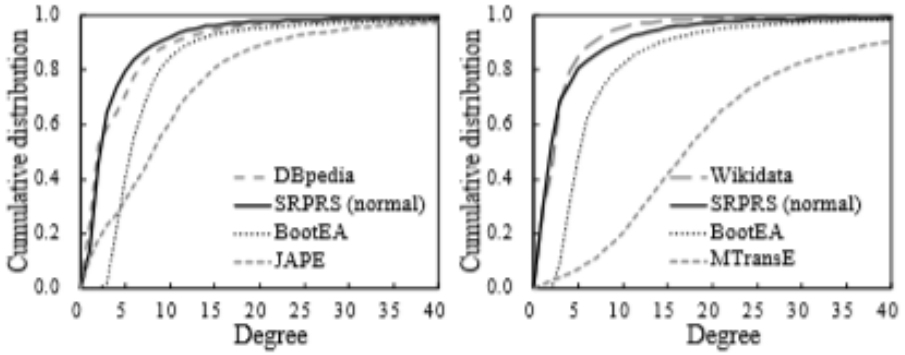
Figure 5. Architecture of the proposed method for entity alignment

Datasets & Settings

Table 7. Statistics of the entity alignment datasets

Datasets	Source KGs	Normal		Dense	
		#Rels.	#Triples	#Rels.	#Triples
DBP-WD	DBpedia (English)	253	38,421	220	68,598
	Wikidata (English)	144	40,159	135	75,465
DBP-YG	DBpedia (English)	219	33,571	206	71,257
	YAGO3 (English)	30	34,660	30	97,131
EN-FR	DBpedia (English)	221	36,508	217	71,929
	DBpedia (French)	177	33,532	174	66,760
EN-DE	DBpedia (English)	225	38,281	207	56,983
	DBpedia (German)	118	37,069	117	59,848

Each dataset contains about 15,000 entities.



(a) DBpedia (b) Wikidata

Figure 6. Comparison of degree distributions of the entity alignment datasets extracted by different methods

- Entity alignment datasets:
  - segment-based random PageRank sampling (SRPRS) method, which can fluently control the degree distributions of entities in the sampled datasets.
  - four couples: nomal & dense entity distribution
- KG completion datasets:
  - FB15K、 FB15K-237、WN18

Table 6. Experimental settings

	Entity alignment	KG completion
Embedding sizes	256	256
Batch sizes	512	2,048
Learning rates	0.003	0.0001
Bias hyper-parameters	$\alpha = 0.9, \beta = 0.9$	$\alpha = 0.7$
Path lengths	15	7

Entity Alignment Results

Table 2. Entity alignment results on the normal datasets

Methods	DBP-WD			DBP-YG			EN-FR			EN-DE		
	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	22.3	50.1	0.32	24.6	54.0	0.34	25.1	55.1	0.35	31.2	58.6	0.40
IPTransE	23.1	51.7	0.33	22.7	50.0	0.32	25.5	55.7	0.36	31.3	59.2	0.41
JAPE	21.9	50.1	0.31	23.3	52.7	0.33	25.6	56.2	0.36	32.0	59.9	0.41
BootEA	32.3	63.1	0.42	31.3	62.5	0.42	31.3	62.9	0.42	44.2	70.1	0.53
GCN-Align	17.7	37.8	0.25	19.3	41.5	0.27	15.5	34.5	0.22	25.3	46.4	0.33
TransR†	5.2	16.9	0.09	2.9	10.3	0.06	3.6	10.5	0.06	5.2	14.3	0.09
TransD†	27.7	57.2	0.37	17.3	41.6	0.26	21.1	47.9	0.30	24.4	50.0	0.33
ConvE†	5.7	16.0	0.09	11.3	29.1	0.18	9.4	24.4	0.15	0.8	9.6	0.03
RotatE†	17.2	43.2	0.26	15.9	40.1	0.24	14.5	39.1	0.23	31.9	55.0	0.40
RSNs (w/o biases)	37.2	63.5	0.46	36.5	62.8	0.45	32.4	58.6	0.42	45.7	69.2	0.54
RSNs	**38.8**	**65.7**	**0.49**	**40.0**	**67.5**	**0.50**	**34.7**	**63.1**	**0.44**	**48.7**	**72.0**	**0.57**

"†" denotes KG completion methods conducted with the source code on the joint KGs. The same to the following.

Table 3. Entity alignment results on the dense datasets

Methods	DBP-WD			DBP-YG			EN-FR			EN-DE		
	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	38.9	68.7	0.49	22.8	51.3	0.32	37.7	70.0	0.49	34.7	62.0	0.44
IPTransE	43.5	74.5	0.54	23.6	51.3	0.33	42.9	78.3	0.55	34.0	63.2	0.44
JAPE	39.3	70.5	0.50	26.8	57.3	0.37	40.7	72.7	0.52	37.5	66.1	0.47
BootEA	67.8	91.2	0.76	68.2	89.8	0.76	64.8	91.9	0.74	66.5	87.1	0.73
GCN-Align	43.1	71.3	0.53	31.3	57.5	0.40	37.3	70.9	0.49	32.1	55.2	0.40
TransR†	14.1	38.6	0.22	13.0	38.0	0.21	15.2	43.8	0.25	10.7	30.9	0.18
TransD†	60.5	86.3	0.69	62.1	85.2	0.70	54.9	86.0	0.66	57.9	81.6	0.66
ConvE†	30.8	50.5	0.38	37.2	57.0	0.44	30.0	49.7	0.37	42.3	60.3	0.49
RotatE†	62.2	86.5	0.71	65.0	87.2	0.73	48.6	80.4	0.59	63.2	83.2	0.70
RSNs (w/o biases)	74.6	90.8	0.80	80.2	95.0	0.86	73.2	90.7	0.80	71.0	87.2	0.77
RSNs	**76.3**	**92.4**	**0.83**	**82.6**	**95.8**	**0.87**	**75.6**	**92.5**	**0.82**	**73.9**	**89.0**	**0.79**

KG Completion Results

Table 4. KG completion results on FB15K

Methods	Hits@1	Hits@10	MRR
TransE‡	30.5	73.7	0.46
TransR‡	37.7	76.7	0.52
TransD‡	31.5	69.1	0.44
ComplEx	59.9	84.0	0.69
ConvE	67.0	87.3	0.75
RotatE	**74.6**	**88.4**	**0.80**
RSNs (w/o cross-KG bias)	72.2	87.3	0.78

"‡" denotes methods executed by ourselves using the source code, due to certain metrics were not evaluated.

Table 5. KG completion results on WN18

Methods	Hits@1	Hits@10	MRR
TransE‡	27.4	94.4	0.58
TransR‡	54.8	94.7	0.73
TransD‡	30.1	93.1	0.56
ComplEx	93.6	94.7	0.94
ConvE	93.5	95.5	0.94
RotatE	**94.4**	**95.9**	**0.95**
RSNs (w/o cross-KG bias)	92.2	95.3	0.94

Table 8. KG completion results on FB15K-237

Methods	Hits@1	Hits@10	MRR
TransE‡	13.3	40.9	0.22
TransR‡	10.9	38.2	0.20
TransD‡	17.8	44.7	0.27
ComplEx	15.2	41.9	0.24
ConvE	23.9	49.1	0.31
RotatE	**24.1**	**53.3**	**0.34**
RSNs (w/o cross-KG bias)	20.2	45.3	0.28

- RSNs outperformed all the translational models that also aim to learn KG embeddings rather than only complete KGs.
- Entity alignment and KG completion exist significant divergences.
- Several methods that performed pretty well on KG completion may be caused by that they were particularly designed for KG completion, and not be capable of training high-quality embeddings.
- The performance of KG completion can largely be improved with a sophisticatedly-designed structure for triples,

Further analysis

- RSNs vs. RNNs, RRNs [recurrent residual networks]
  - Achieved **better** results with only **1/30** epochs

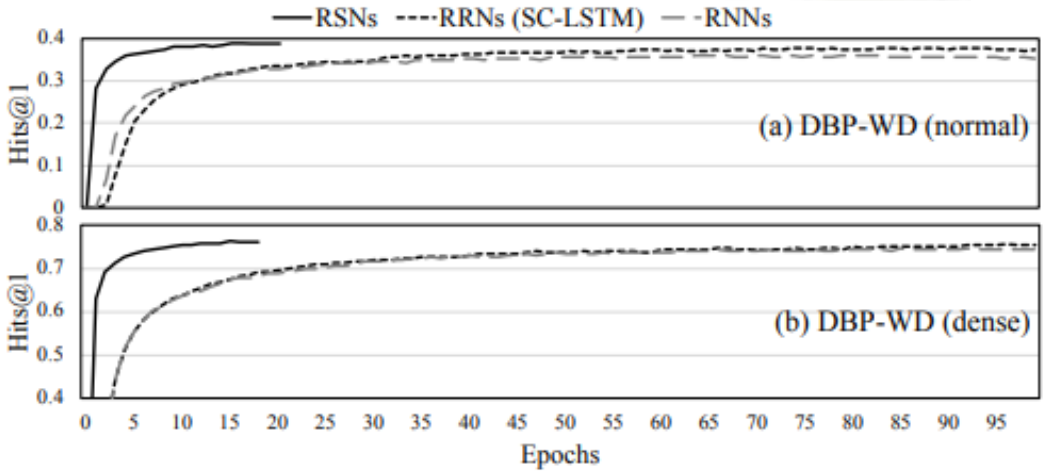
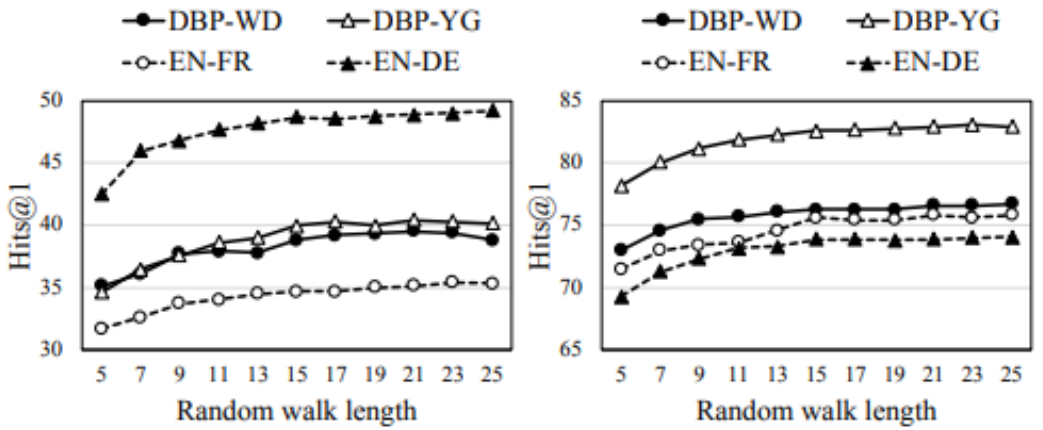


Figure 3. Hits@1 results w.r.t. epochs to converge

- Random walk length
  - On all the datasets, increased steadily from length 5 to **15**



(a) Normal datasets (b) Dense datasets

Figure 4. Hits@1 results w.r.t. random walk length

Conclusion

- We studied **path-level** KG embedding learning
  1. **RSNs:** sequence models to learn relational paths
  2. **End-to-end framework:** biased random walk sampling + RSNs
  3. **Superior** in entity alignment and **competitive** in KG completion
- Future work
  - **Unified sequence model:** relational paths & textual information