Learning to Compose Relational Embeddings in Knowledge Graphs

Wenye Chen, Huda Hakami, Danushka Bollegala

Relation Composition

- Knowledge Graphs (KG) (e.g. Freebase) represent knowledge in the form of relations between entities
  - (Tim Cook, CEO-of, Apple)
- However, KGs are sparse, incomplete, not up to date. Many relations are missing!
- Knowledge Graph Embedding (KGE) methods (e.g. TransE, TransG, RESCAL, ComplE, RelWalk,...) can learn representations for the relations that exist in the KGE.
- We propose *Relation Composition* as a novel task, where we are given pre-trained relation embeddings for the relations that exist in the KG and must predict representations for relations by composing those.
  - **country_of_film + currency_of_country → currency_of_film_budget**
  - (The Italian Job, UK), (UK, GBP) → (The Italian Job, GBP)

Why is this useful?

- KGE methods can only learn representations for the relations that exist in the training data.

- Although they can predict links (relations) that currently do not exist between two entities in the KG, these links are limited to the relation types that exist in the training data

    - They cannot predict representations for previously unseen (not in training data) relations that are encountered during test time.

- Relation composition can be seen as an instance of zero-shot learning setting, where the representations we compute do not correspond to any of the relations we have in the training data.

- A compositional semantic approach for relation representations!

Relation Compositional Operators

- We will learn compositional operators that take pre-trained relation representations for two known relations as the input and return a representation for their composition as the output.

- We consider/propose both unsupervised and supervised relation compositional operators for this purpose.

- We do not need entity embeddings (or any information regarding the entities between which relations hold)

- We can use relation embeddings learnt using any KGE method.

    - As a running example, we use relation embeddings learnt using **RelWalk** [Bollegala+, 2019], which represents relations using matrices and report superior performance on KGE benchmarks.

- Benefits of considering relation composition for RelWalk embeddings

    - Composing matrices is more computationally complex.

    - It is more general than composing vectorial relation embeddings (diagonal matrices can be used to represent vectors)

Background — RelWalk

- relational walk (RelWalk) [Bollegala+ 2019] is a method for learning KGEs by performing a random walk over a given KG.

- The generative probabilities of head (h) and tail (t) entities for a relation R are modelled using two matrices $\mathbf{R}_1$ and $\mathbf{R}_2$.

  - $p(h \mid R, c) = \frac{1}{Z_c} \exp(\mathbf{h}^\top \mathbf{R}_1 \mathbf{c})$

  - $p(t \mid R, c') = \frac{1}{Z_c} \exp(\mathbf{t}^\top \mathbf{R}_2 \mathbf{c}')$

- We proved the following concentration lemma for such a random walk

Concentration Lemma

If the entity embedding vectors satisfy the Bayesian prior $\mathbf{v} = s\hat{\mathbf{v}}$, where $\hat{\mathbf{v}}$ is from the spherical Gaussian distribution, and s is a scalar random variable, which is always bounded by a constant κ, then the entire ensemble of entity embeddings satisfies that

$$\Pr_{c \sim C}[(1 - \epsilon_z)Z \leq Z_c \leq (1 + \epsilon_z)Z] \geq 1 - \delta$$

for $\epsilon_z = \mathcal{O}(1/\sqrt{n})$, and $\delta = \exp(-\Omega(\log^2 n))$, where $n \geq d$ is the number of words and Z_c is the partition function for c given by $\sum_{h \in \mathcal{E}} \exp(\mathbf{h}^\top \mathbf{R}_1 \mathbf{c})$.

Background — RelWalk

- Under the conditions where the concentration lemma is satisfied, we proved Theorem 1, which relates KGEs to the connections in the KG.

- We can then learn KGEs from a given KG such that the relationship given by Theorem 1 is empirically satisfied.

Theorem 1

Suppose that the entity and relation embeddings satisfy the concentration lemma. Then, we have

$$\log p(h, t \mid R) = \frac{\|\mathbf{R}_1^\top \mathbf{h} + \mathbf{R}_2^\top \mathbf{t}\|_2^2}{2d} - 2\log Z \pm \epsilon$$

for $\epsilon = \mathcal{O}(1/\sqrt{n}) + \tilde{\mathcal{O}}(1/d)$, where $Z = Z_c = Z_{c'}$.

Relation Compositional Operators

- Let us assume that two relations R_A and R_B jointly imply a third relation R_C. We denote this fact by $R_A \wedge R_B \Rightarrow R_C$

- Moreover, let relation embeddings for R_A and R_B be respectively ($\mathbf{R}_1^A$, $\mathbf{R}_2^A$) and ($\mathbf{R}_1^B$, $\mathbf{R}_2^B$). For simplicity, let us assume all relation embeddings are in $\mathbb{R}^{d \times d}$. The predicted relation embeddings $(\hat{\mathbf{R}}_1^C, \hat{\mathbf{R}}_2^C)$ for R_C are computed using two relation compositional operators (ϕ_1, ϕ_2) such that:

- $\phi_1 : \mathbf{R}_1^A, \mathbf{R}_2^A, \mathbf{R}_1^B, \mathbf{R}_2^B \rightarrow \hat{\mathbf{R}}_1^C$

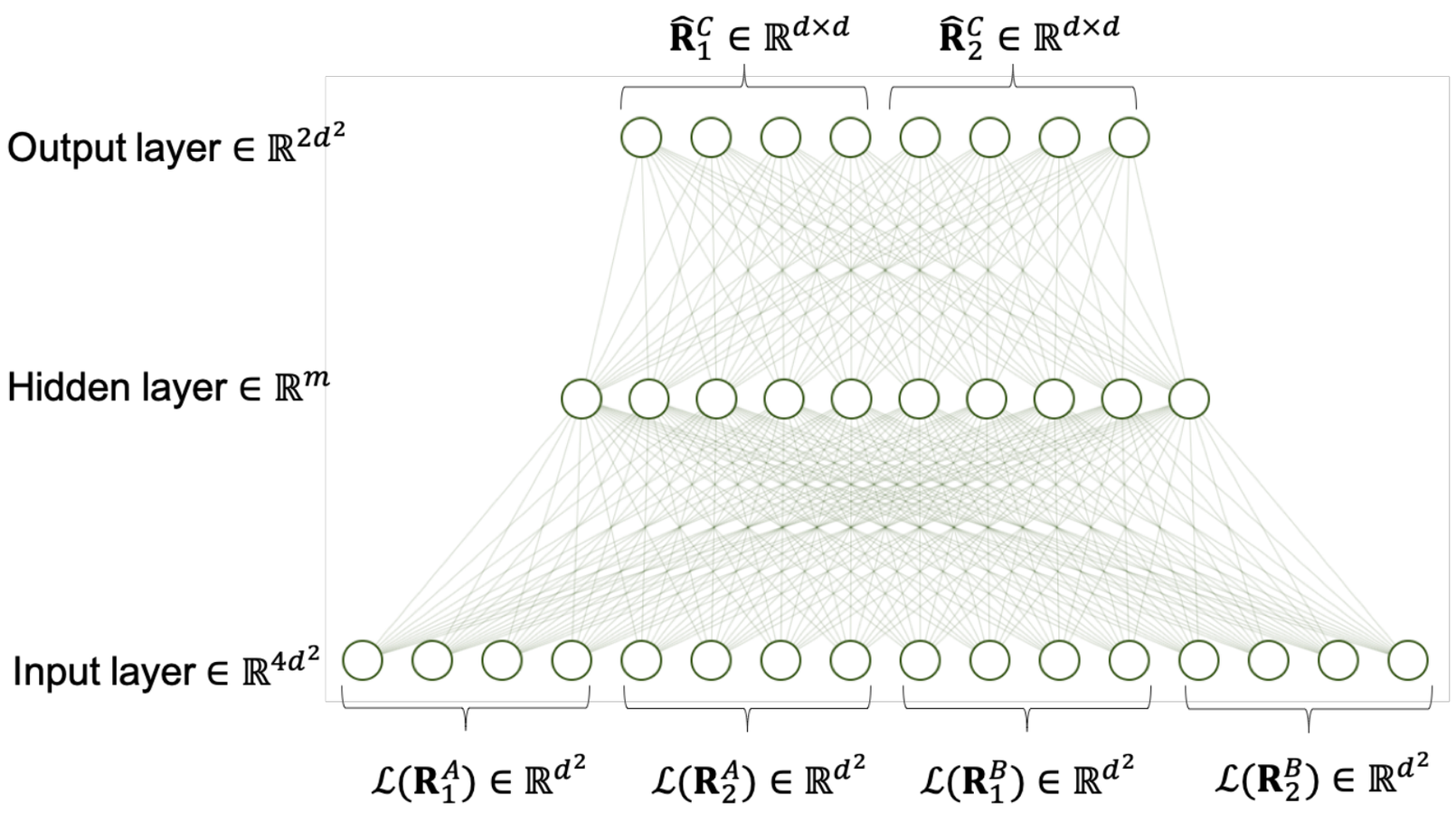
- $\phi_2 : \mathbf{R}_1^A, \mathbf{R}_2^A, \mathbf{R}_1^B, \mathbf{R}_2^B \rightarrow \hat{\mathbf{R}}_2^C$

Unsupervised Relation Composition

- Addition
  - $\mathbf{R}_1^A + \mathbf{R}_1^B = \hat{\mathbf{R}}_1^C$
  - $\mathbf{R}_2^A + \mathbf{R}_2^B = \hat{\mathbf{R}}_2^C$
- Matrix Product
  - $\mathbf{R}_1^A \mathbf{R}_1^B = \hat{\mathbf{R}}_1^C$
  - $\mathbf{R}_2^A \mathbf{R}_2^B = \hat{\mathbf{R}}_2^C$
- Hadamard Product
  - $\mathbf{R}_1^A \odot \mathbf{R}_1^B = \hat{\mathbf{R}}_1^C$
  - $\mathbf{R}_2^A \odot \mathbf{R}_2^B = \hat{\mathbf{R}}_2^C$

Supervised Relation Composition

- Limitations of the unsupervised relation compositional operators
  - Cannot be fine-tuned for the relations in a given KG.
  - Considers R_1 and R_2 independently and cannot model their interactions.
- We can use a non-linear neural network as a learnable operator!



Training settings

Forward-pass

$$\boldsymbol{x} = \mathfrak{L}(\mathbf{R}_1^A) \oplus \mathfrak{L}(\mathbf{R}_2^A) \oplus \mathfrak{L}(\mathbf{R}_1^B) \oplus \mathfrak{L}(\mathbf{R}_2^B)$$

$$\boldsymbol{h} = f(\mathbf{W}\boldsymbol{x} + \boldsymbol{b})$$

$$\boldsymbol{y} = \mathbf{U}\boldsymbol{h} + \boldsymbol{b}'$$

$$\hat{\mathbf{R}}_1^C = \mathfrak{L}^{-1}\boldsymbol{y}_{:d^2}$$

$$\hat{\mathbf{R}}_2^C = \mathfrak{L}^{-1}\boldsymbol{y}_{d^2:}$$

Loss function

$$L(\mathbf{W}, \mathbf{U}, \boldsymbol{b}, \boldsymbol{b}') = \left|\left|\mathbf{R}_1^C - \hat{\mathbf{R}}_1^C\right|\right|_2^2 + \left|\left|\mathbf{R}_2^C - \hat{\mathbf{R}}_2^C\right|\right|_2^2$$

- Learn relational embeddings for d = 20, 50, and 100 from Freebase 15k-237 dataset using RelWalk.
- This dataset contains 237 relation types for 14541 entities.
- Train, test and validation parts of this dataset contain respectively 544230, 40932 and 35070 triples.
- To preserve the asymmetry property for relations, we consider that each relation $R^<$ in the relation set has its inverse $R^>$, so that for each triple (h, $R^<$, t) in the KG we regard (t, $R^>$, h) is also in the KG.

Evaluation Dataset

- We use the relation composition (RC) dataset created by Takahashi+ [ACL 2018] from FB15-23k as follows.
  - For a relation R, the *content set* C(R) is defined as the set of (h,t) pairs such that (h, R, t) is a fact in the KG.
  - Likewise, $C(R_A \wedge R_B)$ is defined as the set of (h,t) pairs such that (h, $R_A \longrightarrow R_B$, t) is a path in the KG.
    - $R_A \wedge R_B \Rightarrow R_C$ is considered as a compositional constraint if their content sets are similar
      - i.e. $|\, C(R_A \wedge R_B) \cap C(R_C) \,| \geq 50$ and the Jaccard similarity between $C(R_A \wedge R_B)$ and $C(R_C)$ is greater than 0.4
- 154 compositional constraints are listed in this RC dataset
- We perform 5-fold cross-validation on the RC dataset

Evaluation — Relation Composition

- Relation Composition Task
  - Given two relations R_A and R_B, we predict the embedding for their composition, $\hat{R}^C$. We then find the closest test relation R^L for the predicted embedding according to
    - $d(R_L, \hat{R}_C) = \|\mathbf{R}_1^L - \hat{\mathbf{R}}_1^C\|_F + \|\mathbf{R}_2^L - \hat{\mathbf{R}}_2^C\|_F$
  - We model this as a ranking task and use Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@10 to measure the accuracy of the composition.

Results — Relation Composition

	d=20			d=50			d=100		
Method	MR	MRR	Hits@10	MR	MRR	Hits@10	MR	MRR	Hits@10
Supervised Relation Composition	**75**	**0.412**	**0.581**	**64**	**0.390**	**0.729**	**49**	**0.308**	**0.703**
Addition	238	0.010	0.012	250	0.008	0.019	247	0.007	0
Matrix Product	225	0.018	0.032	233	0.012	0.025	231	0.010	0.019
Hadamard Product	215	0.020	0.051	192	0.037	0.051	209	0.016	0.032

- Supervised relation composition achieves the best results for MR, MRR and Hits@10 with significant improvements over the unsupervised relational compositional operators.

- Hadamard product is the best among unsupervised relation compositional operators.

- However, the performance of unsupervised operators are close to the random baseline, which picks a relation type uniformly at random from the test relation types.

Evaluation — Triple Classification

- Triple Classification Task
  - Given a triple (h,R,t), predict whether it is True (a fact in the KG) or False (not).
    - A binary classification task
  - We use p(h,R,t) computed according to Theorem 1 to determine whether (h,R,t) is True or False.
- Positive triples
  - Triples that actually appear in the training dataset
- Negative triples
  - Random perturbation of positive triples to create pseudo-negative triples. For example, given (h,R,t) we replace t with t’ to create a negative triple (h,R,t’) that does not appear in the set of training triples.
- 5-fold cross-validation is performed on the RC dataset to find a threshold on the probability to predict positive/negative triples.

Results — Triple Classification

Accuracy for triple classification

Method	d=20	d=50	d=100
Supervised Relation Composition	**77.55**	**77.73**	**77.62**
Addition	68.9	70.44	69.45
Matrix Product	67.6	65.24	75.71
Hadamard Product	58.44	63.01	70.94

- Across the relational compositional operators and for different embedding dimensionalities, the proposed supervised relational composition operator achieves the best accuracy.

Conclusions

- We proposed a novel task — relation composition to predict embeddings for relations that can be composed using the pre-trained embeddings for the existing relation types in a KG.

- We compared unsupervised and supervised (modelled as a non-linear neural network) for this purpose.

- Supervised relation composition operator outperforms its unsupervised counterparts in relation composition and triple classification tasks.

  - Code: https://github.com/Bollegala/RelComp

- Future work

  - Compositions involving more than two relations!

  - Multi-hop composition!!!