Graph Neural Networks

Lecture 7

Danila Biktimirov
Applied Computer Science
Neapolis University Pafos

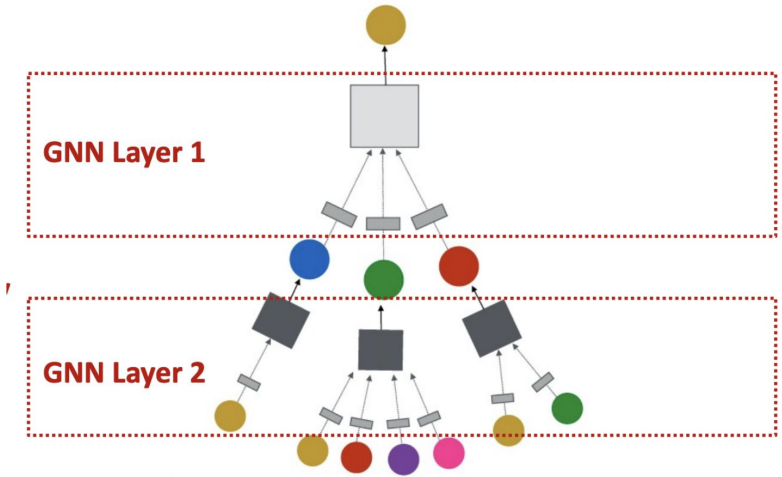
21 March 2025

Overview

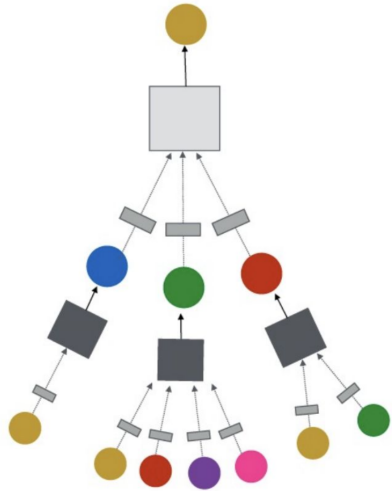
1. **Previously on...**

Previously on...

Recap: Layer Connection

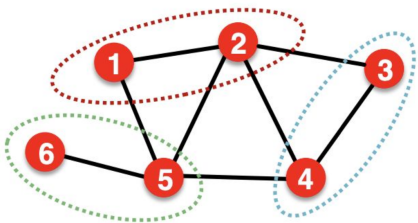


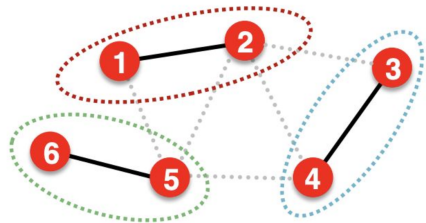
Recap: Augmentations



(4) Graph augmentation

Recap: Splitting



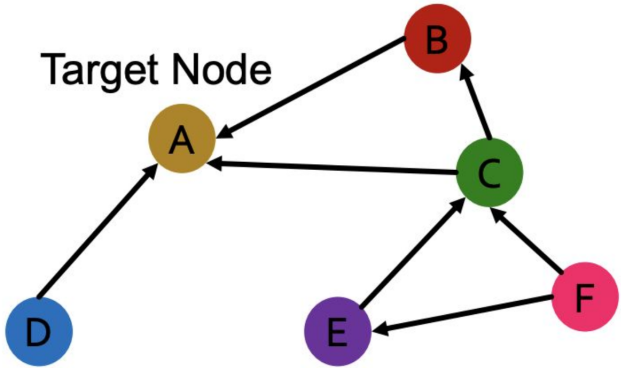


Knowledge Graphs

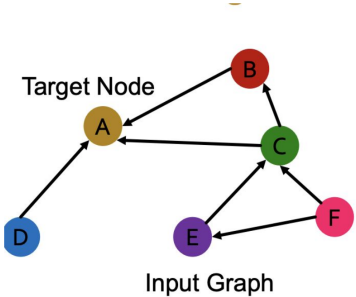
Heterogeneous Graph

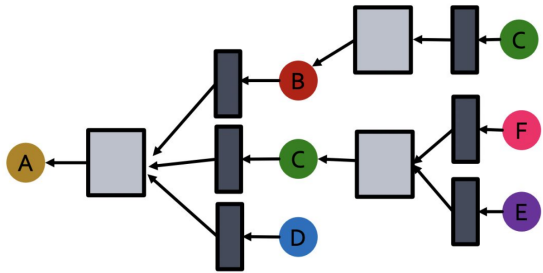
$$G = (V, E, R, T)$$

Directed Graph

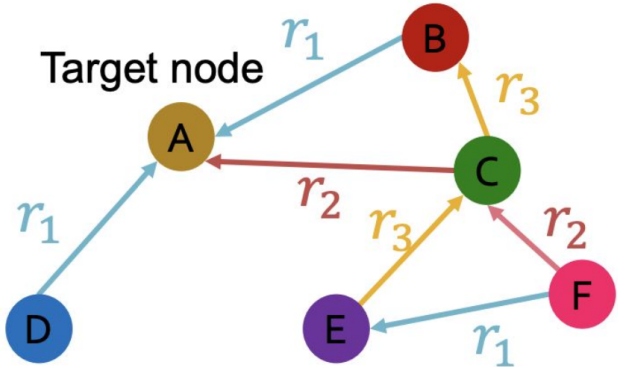


Computational Graph



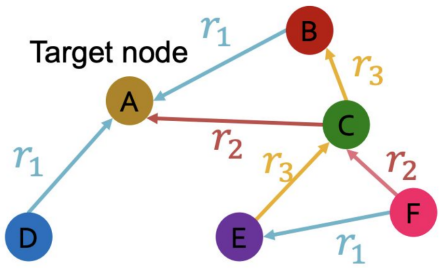


Knowledge Graph



Input graph

Message Layers



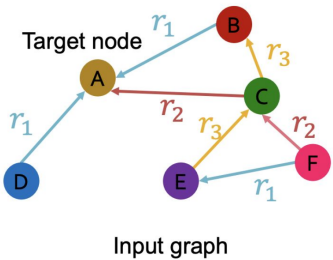
Input graph

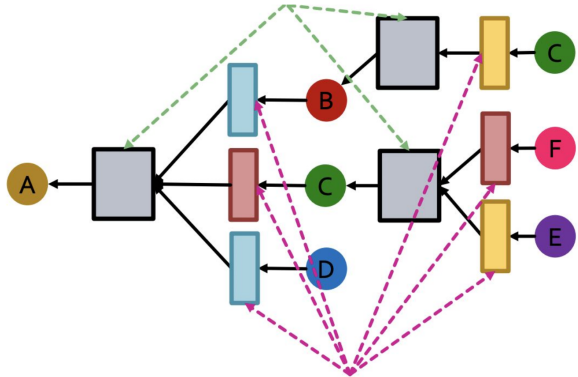
Weights $\mathbf{W}_{r_1}$ for r_1

Weights $\mathbf{W}_{r_2}$ for r_2

Weights $\mathbf{W}_{r_3}$ for r_3

Computational Graph in a Knowledge Graph





RGCN

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\sum_{r\in R}\sum_{u\in N_v^r}\frac{1}{c_{v,r}}\mathbf{W}_r^{(l)}\mathbf{h}_u^{(l)} + \mathbf{W}_0^{(l)}\mathbf{h}_v^{(l)}\right)$$

Message

$$\mathbf{m}_{u,r}^{(l)} = \frac{1}{c_{v,r}}\mathbf{W}_r^{(l)}\mathbf{h}_u^{(l)}$$

$$\mathbf{m}_v^{(l)} = \mathbf{W}_0^{(l)}\mathbf{h}_v^{(l)}$$

Aggregation

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\mathrm{Sum}\left(\left\{\mathbf{m}_{u,r}^{(l)} \mid u \in N(v)\right\} \cup \left\{\mathbf{m}_v^{(l)}\right\}\right)\right)$$

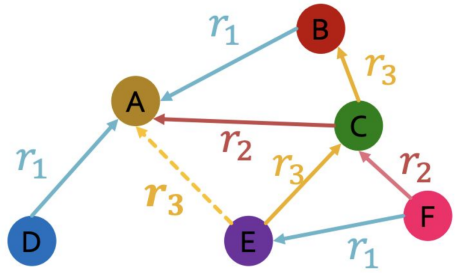
Problems with RGCN

Due to the large number of parameters, RGCN overfits quickly

Possible solutions:

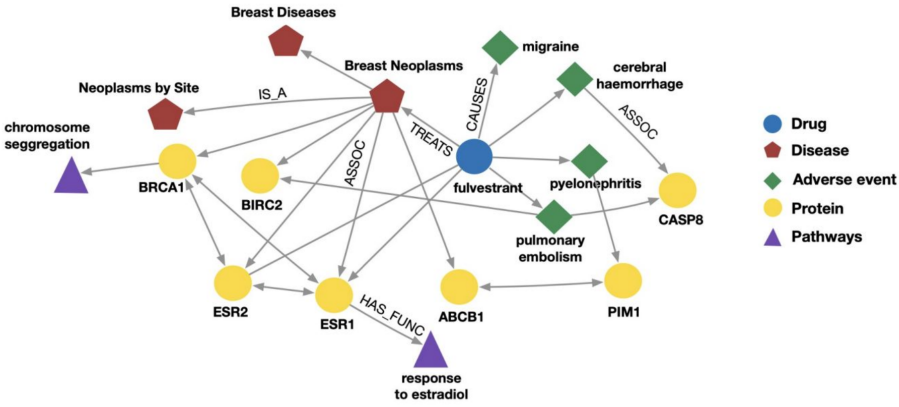
1. Diagonal matrix
2. Basis decomposition

Example



$$\ell = -\log \sigma\left(f_{r_3}(h_E, h_A)\right) - \log(1 - \sigma(f_{r_3}(h_E, h_B)))$$

Knowledge Graphs



TransE

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

TransE Learning Algorithm

Algorithm 1 Learning TransE

input Training set $S = \{(h, \ell, t)\}$, entities and rel. sets E and L, margin γ, embeddings dim. k.

initialize $\boldsymbol{\ell} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$

$\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} / \|\boldsymbol{\ell}\|$ for each $\ell \in L$

$\mathbf{e} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$

Entities and relations are initialized uniformly, and normalized

loop

  $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$

  $S_{batch} \leftarrow$ sample(S, b) // sample a minibatch of size b

  $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets

  for $(h, \ell, t) \in S_{batch}$ **do**

    $(h', \ell, t') \leftarrow$ sample$(S'_{(h,\ell,t)})$ // sample a corrupted triplet

Negative sampling with triplet that does not appear in the KG

    $T_{batch} \leftarrow T_{batch} \cup \{((h, \ell, t), (h', \ell, t'))\}$

  end for

  Update embeddings w.r.t.

$$\sum_{\left((h,\ell,t),(h',\ell,t')\right) \in T_{batch}} \nabla \left[\gamma + d(\boldsymbol{h} + \boldsymbol{\ell}, \boldsymbol{t}) - d(\boldsymbol{h'} + \boldsymbol{\ell}, \boldsymbol{t'})\right]_+$$

positive sample

negative sample

d represents distance (negative of score)

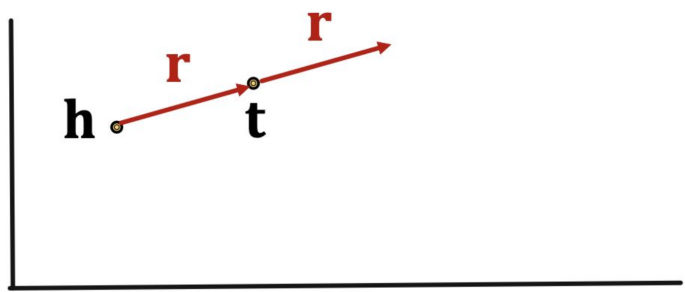
loop

Contrastive loss: favors lower distance (or higher score) for valid triplets, high distance (or lower score) for corrupted ones

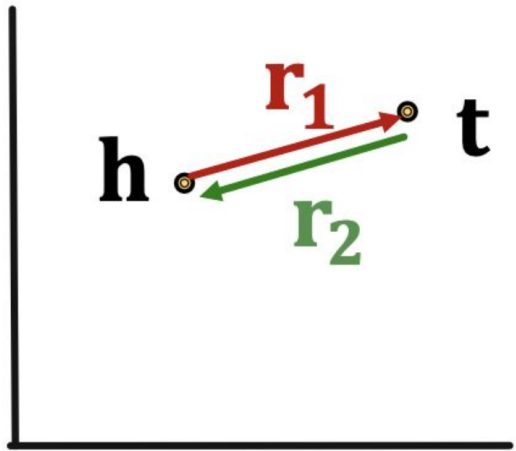
Relation Patterns

- (Anti)Symmetry: $r(h,t) \Rightarrow r(t,h)$ ($r(h,t) \Rightarrow \neg r(t,h)$) $\forall h, t$
- Inversion: $r_2(h,t) \Rightarrow r_1(t,h)$
- Transitivity: $r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z)$ $\forall x, y, z$
- 1-to-N: $r(h,t_1), r(h,t_2), \ldots, r(h,t_n)$ are all True

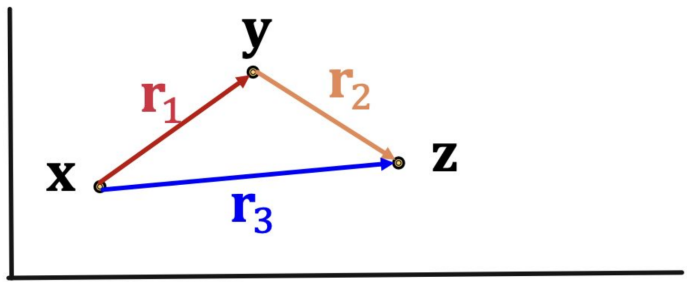
TransE: Antisymmetry



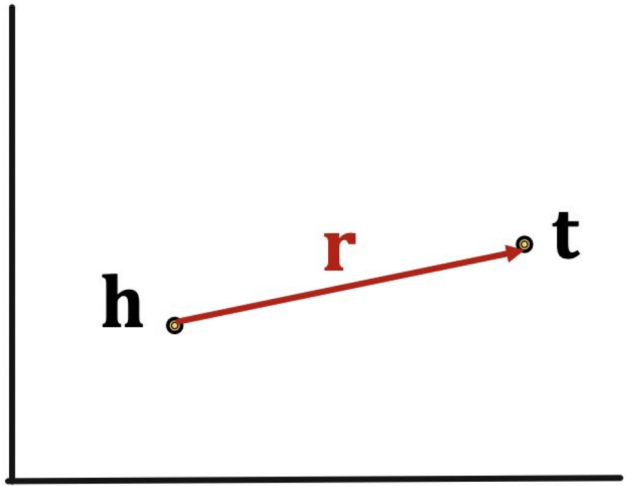
TransE: Inversion



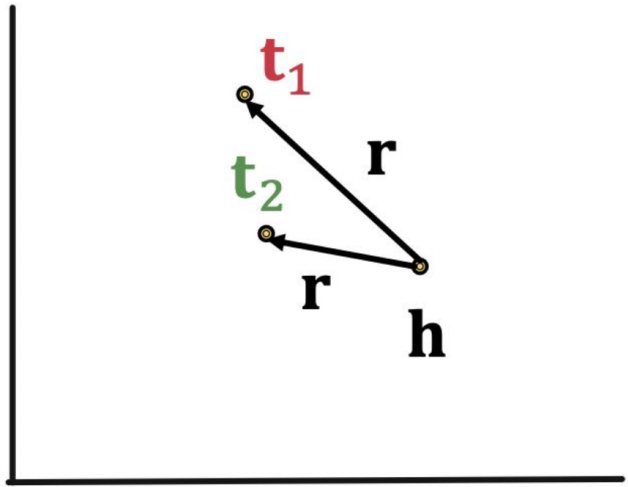
TransE: Composition



TransE: Symmetry



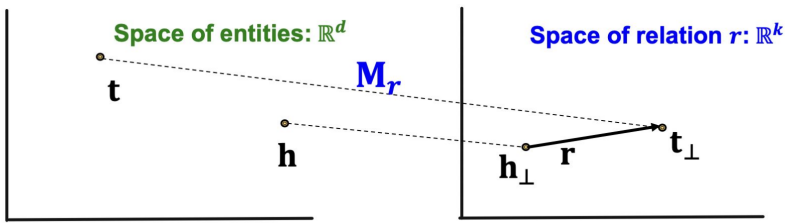
TransE: 1-to-N



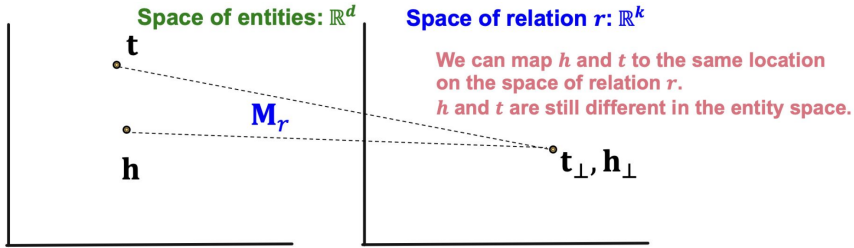
TransR

$$\mathbf{h}_\perp = M_r\mathbf{h}, \quad \mathbf{t}_\perp = M_r\mathbf{t}$$

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

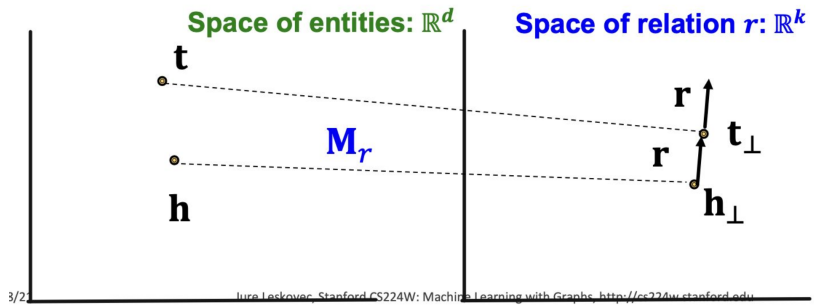


TransR: Symmetry

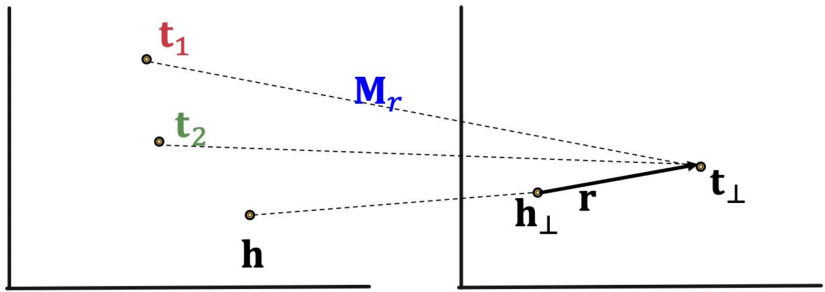


TransR: Antisymmetry

$$r \neq 0, \quad M_r\mathbf{h} + r = M_r\mathbf{t} \Rightarrow M_r\mathbf{t} + r \neq M_r\mathbf{h}$$

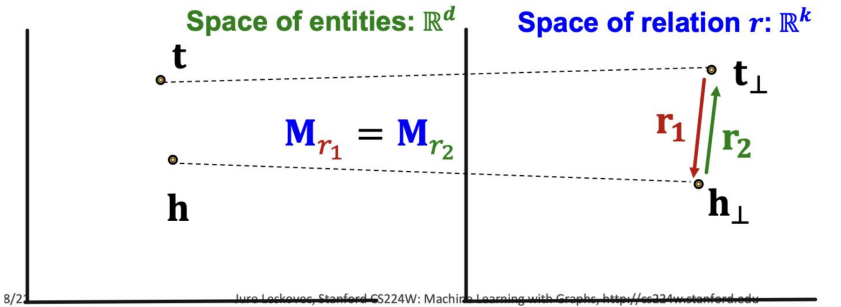


TransR: 1-to-N


$\mathbf{t}_1$
$\mathbf{M}_r$
$\mathbf{t}_2$
$\mathbf{t}_\perp$
$\mathbf{h}_\perp$
$\mathbf{r}$
$\mathbf{h}$

TransR: Inversion

$$M_{r_1} = M_{r_2}$$



The End?