Stanford CS224W: Knowledge Graph Embeddings

CS224W: Machine Learning with Graphs

Charilaos Kanatsoulis and Jure Leskovec, Stanford University

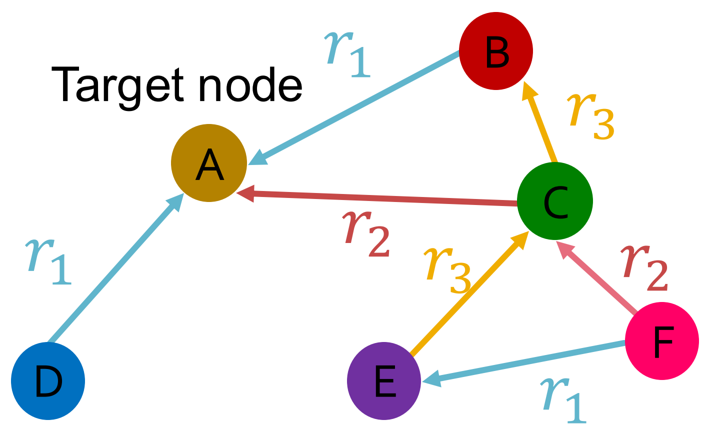
http://cs224w.stanford.edu

Announcements

- **Colab 2** due **today**
  - Gradescope submissions close at 11:59 PM
- **Colab 3** out **today**, due Thursday 11/6
- **Homework 1 grades are out now!**
  - Regrade requests are open until Thursday 10/30
- **Homework 2** due Thursday 10/30
  - Recitation session materials posted on Ed/Canvas

Recap: Heterogeneous Graphs

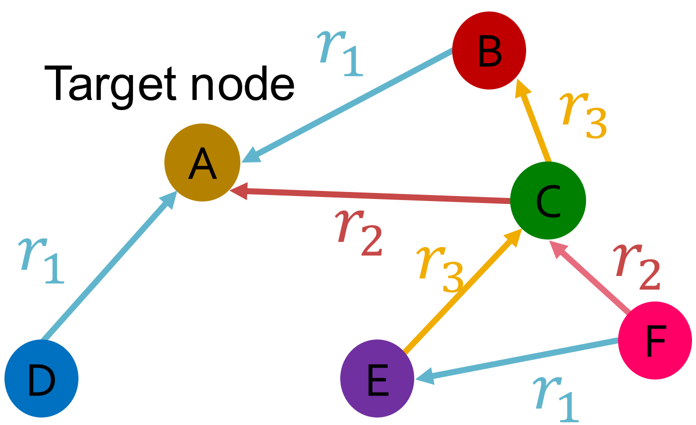
- **Heterogeneous graphs: a graph with multiple relation types**



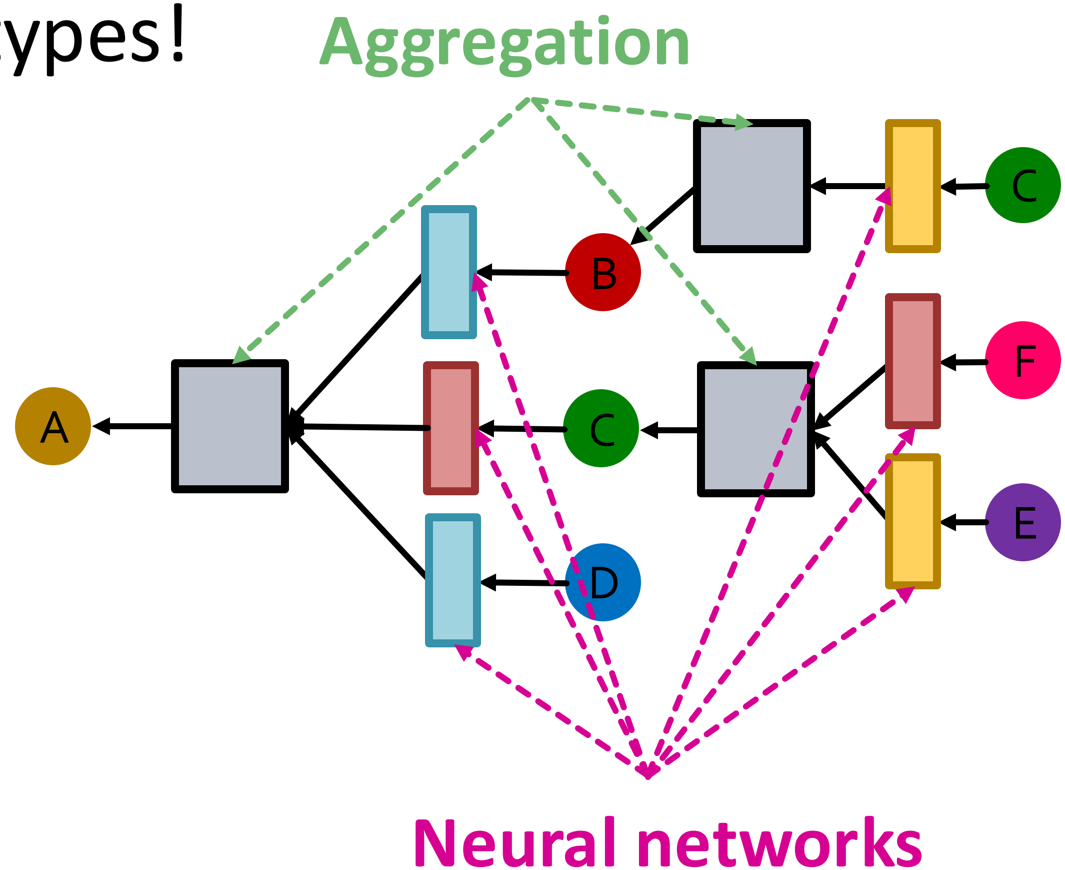
Input graph

Recap: Relational GCN

- Learn from a graph with **multiple relation types**
- Use different neural network weights for different relation types!



Input graph

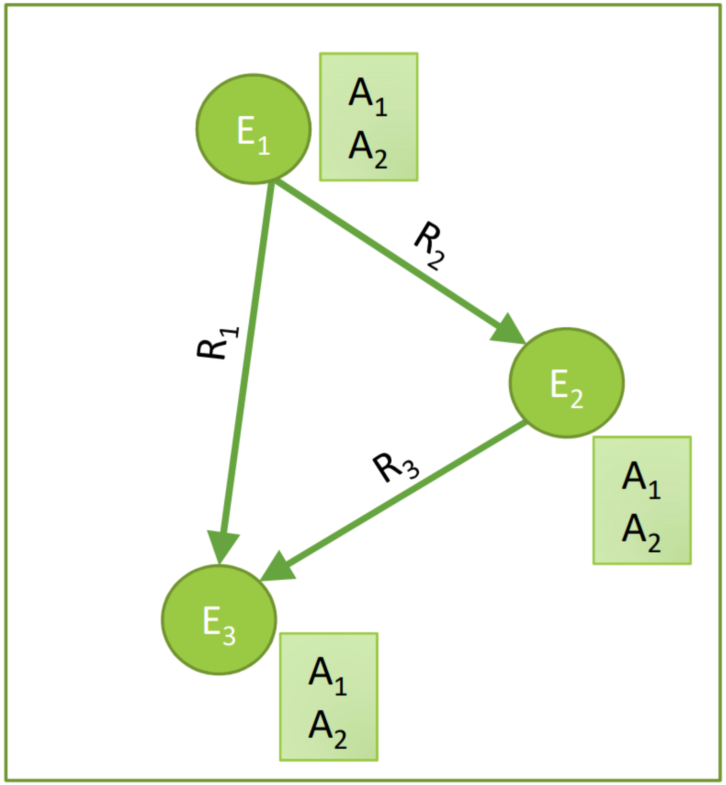


Today: Knowledge Graphs (KG)

Knowledge in graph form:

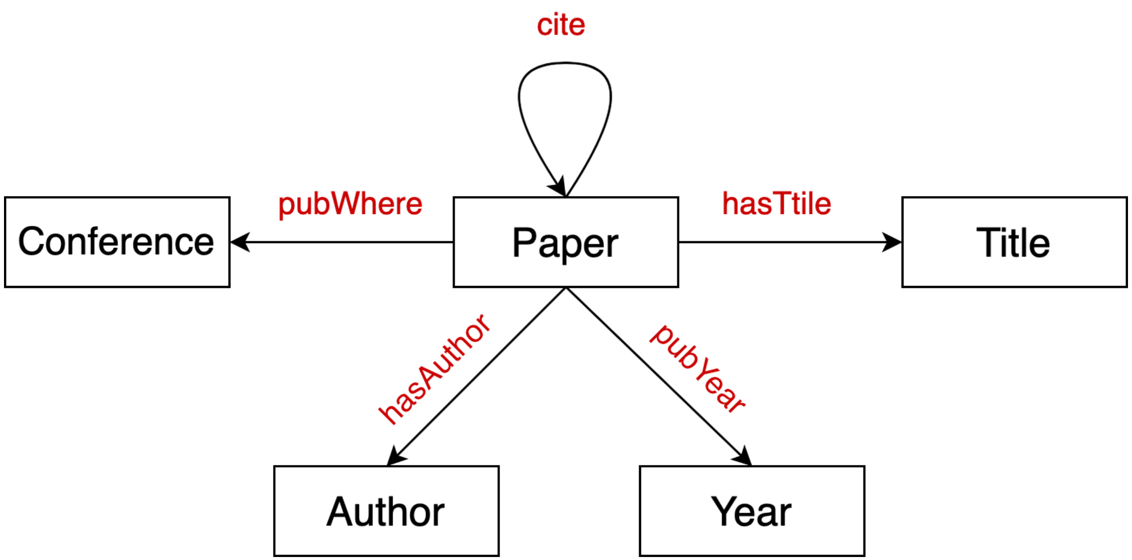
- Capture entities, types, and relationships

- Nodes are **entities**
- Nodes are labeled with their **types**
- Edges between two nodes capture **relationships** between entities
- **KG is an example of a heterogeneous graph**



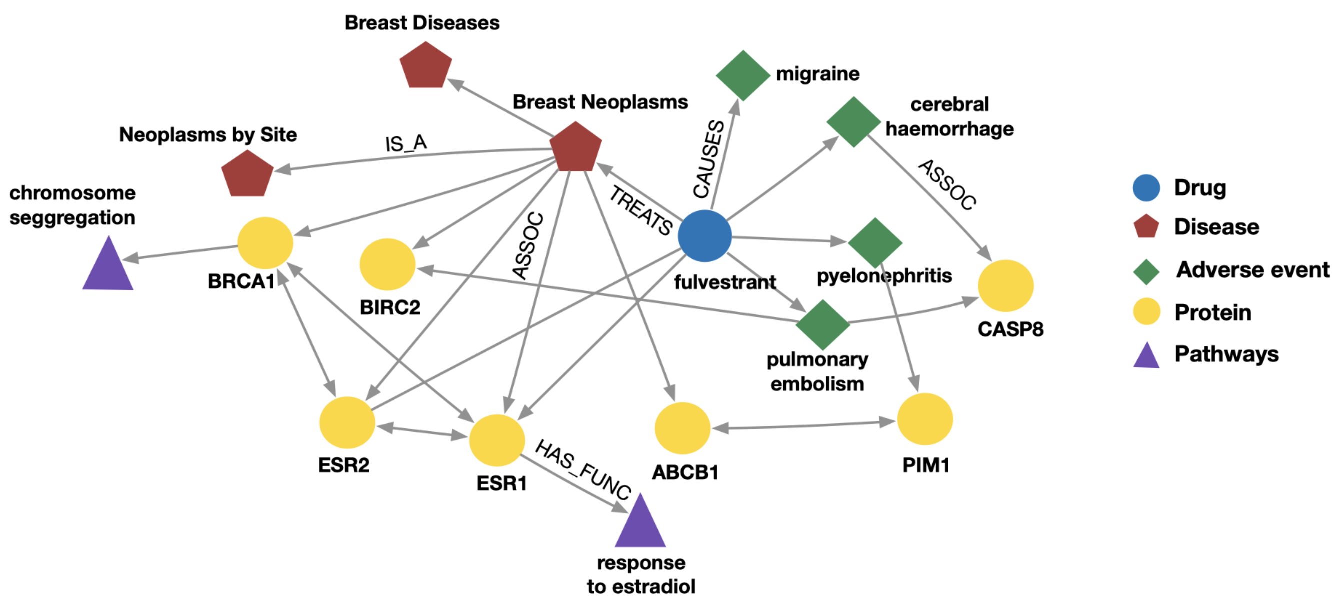
Example: Bibliographic Networks

- **Node types**: paper, title, author, conference, year
- **Relation types**: pubWhere, pubYear, hasTitle, hasAuthor, cite



Example: Bio Knowledge Graphs

- **Node types**: drug, disease, adverse event, protein, pathways
- **Relation types**: has_func, causes, assoc, treats, is_a



Knowledge Graphs in Practice

Examples of knowledge graphs

- Google Knowledge Graph
- Amazon Product Graph
- Meta Graph API
- IBM Watson
- Microsoft Satori
- Project Hanover/Literome
- LinkedIn Knowledge Graph
- Yandex Object Answer

Applications of Knowledge Graphs

- **Serving information:**

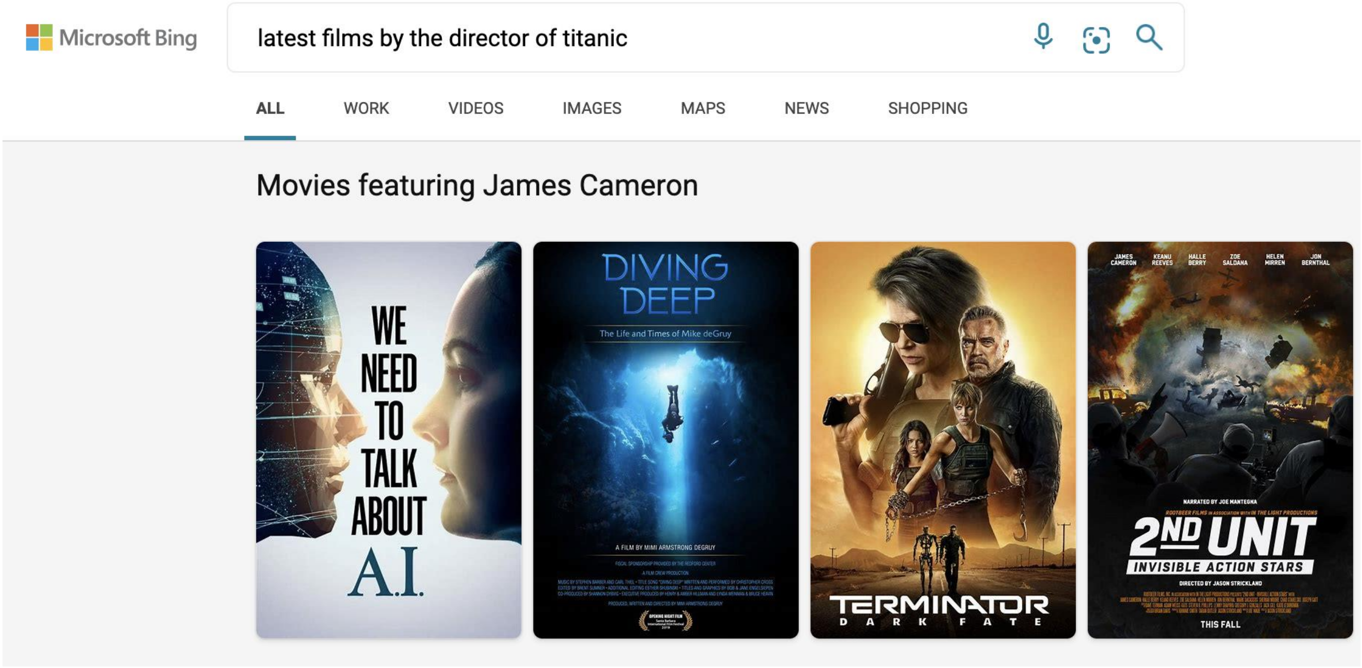


Image credit: Bing

Knowledge Graph Datasets

- **Publicly available KGs:**
  - FreeBase, Wikidata, Dbpedia, YAGO, NELL, etc.
- **Common characteristics:**
  - **Massive**: Millions of nodes and edges
  - **Incomplete**: Many true edges are missing

Given a massive KG, enumerating all the possible facts is intractable! ⇨ **Can we predict plausible BUT missing links?**

Example: Freebase

- **Freebase**
  - ~80 million **entities**
  - ~38K **relation types**
  - ~3 billion **facts/triples**

93.8% of persons from Freebase have no place of birth and 78.5% have no nationality!

- **Datasets:** FB15k/FB15k-237
  - A **complete** subset of Freebase, used by researchers to learn KG models

Dataset	Entities	Relations	Total Edges
FB15k	14,951	1,345	592,213
FB15k-237	14,505	237	310,079

[1] Paulheim, Heiko. "Knowledge graph refinement: A survey of approaches and evaluation methods." *Semantic web* 8.3 (2017): 489-508.
[2] Min, Bonan, et al. "Distant supervision for relation extraction with an incomplete knowledge base." *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2013.

Stanford CS224W: Knowledge Graph Completion

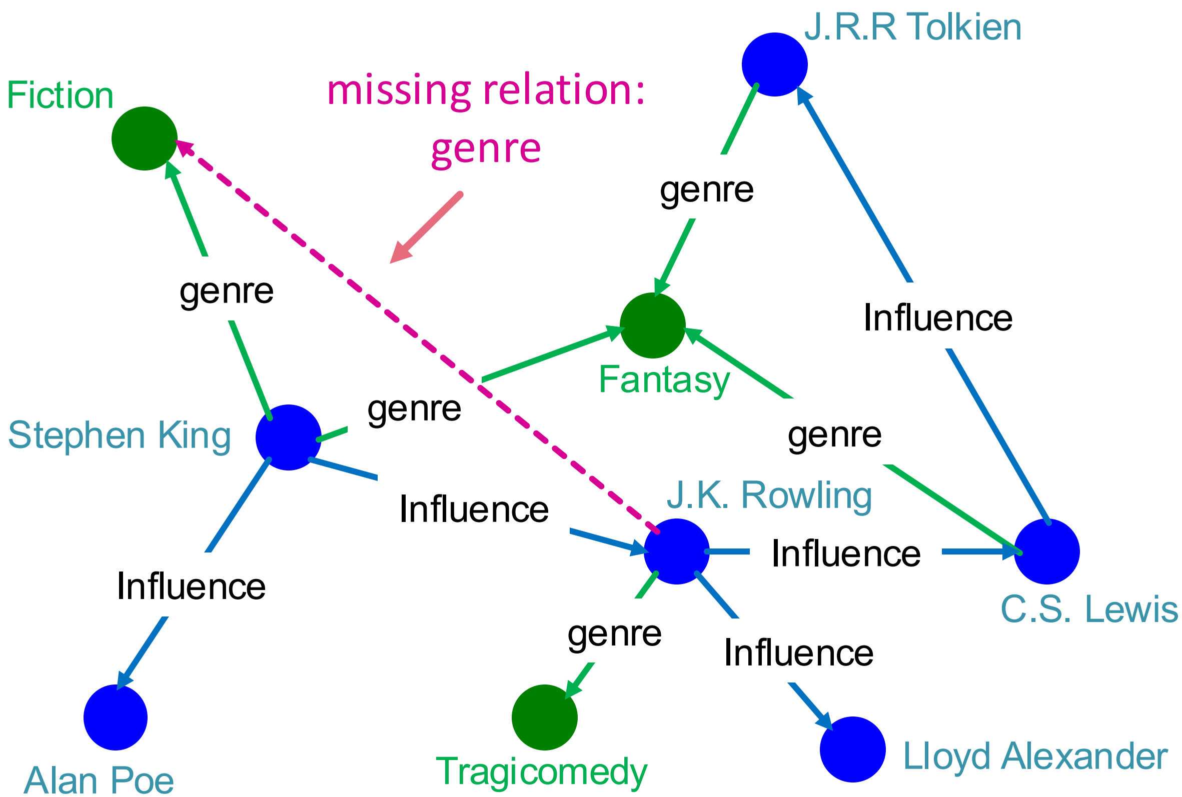
CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

KG Completion Task

Given an enormous KG, can we complete the KG?

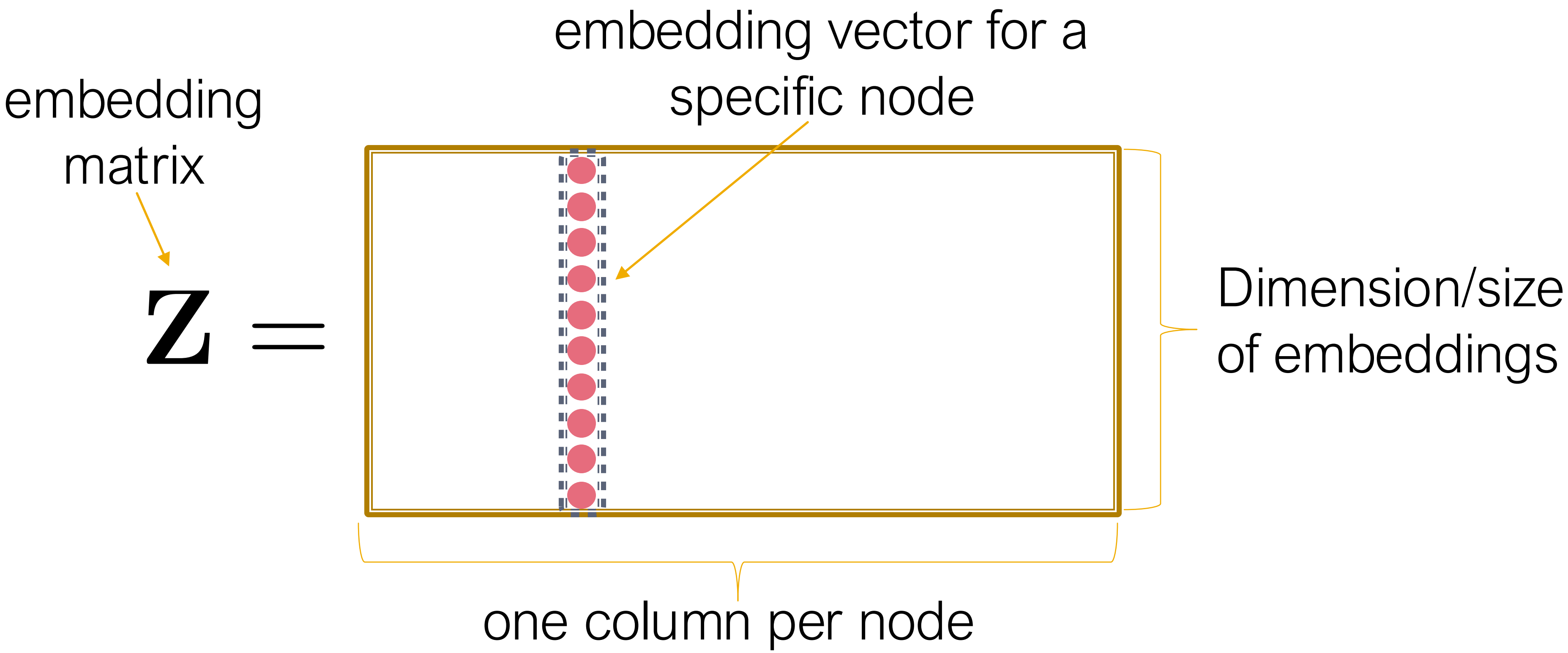
- For a given (**head**, **relation**), we predict missing **tails**.
  - (Note this is slightly different from link prediction task)

Example task: predict the tail "Science Fiction" for ("J.K. Rowling", "genre")



Recap: "Shallow" Encoding

- Simplest encoding approach: **encoder is just an embedding-lookup**

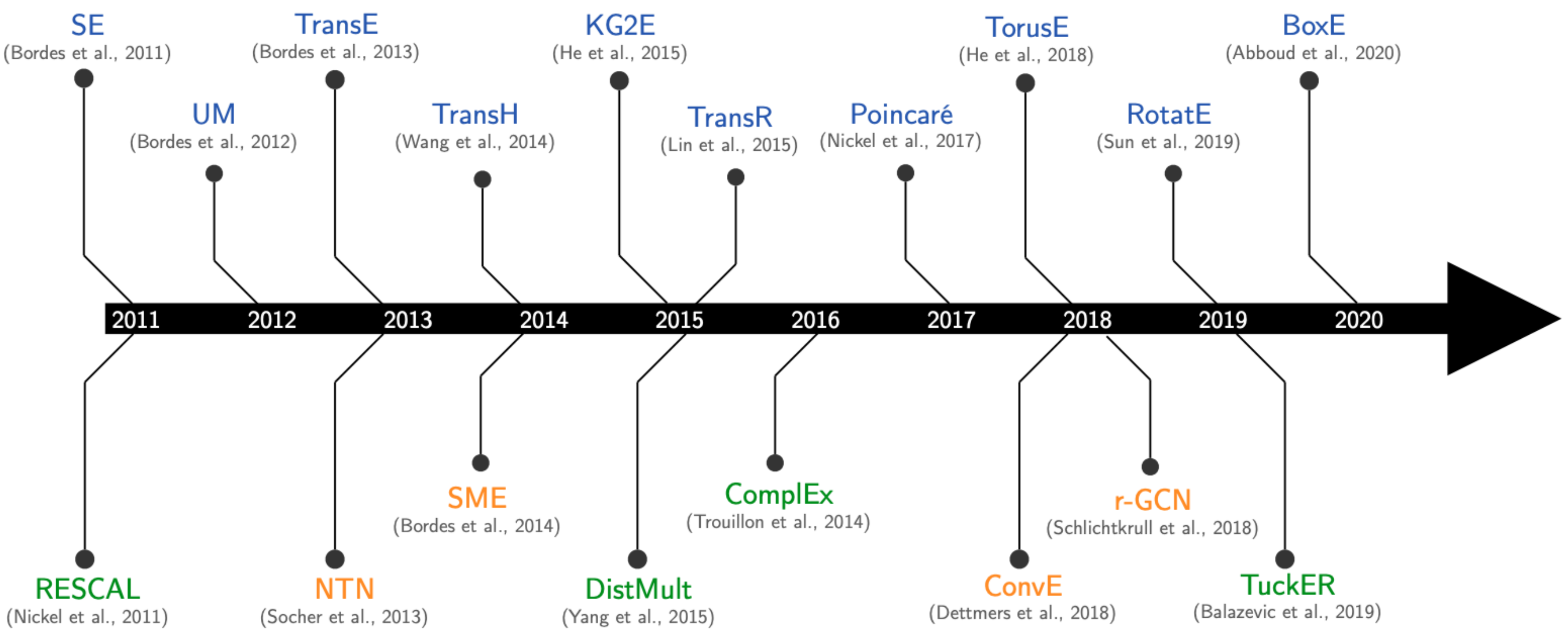


KG Representation

- Edges in KG are represented as **triples** (h, r, t)
  - head (h) has relation (r) with tail (t)
- **Key Idea:**
  - Model entities and relations in embedding space $\mathbb{R}^k$
    - Associate entities and relations with **shallow embeddings**
      - **Note we do not learn a GNN here!**
  - Given a triple (h, r, t), the goal is that the embedding of (h, r) **should be close** to the embedding of t.
    - How to embed (h, r)?
    - How to define score $f_r(h, t)$?
      - Score f_r is high if (h, r, t) exists, else f_r is low

Many KG Embedding Models

- Many KG embedding Models:



Today: Different Models

We are going to learn about different KG embedding models (shallow/transductive embs):

- Different models are…
  - …based on different geometric intuitions
  - …capture different types of relations (have different expressivity)

Model	Score	Embedding	Sym.	Antisym.	Inv.	Compos.	1-to-N
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{R}^k$	✗	✓	✓	✓	✗
TransR	$-\|\boldsymbol{M}_r\mathbf{h}+\mathbf{r}-\boldsymbol{M}_r\mathbf{t}\|$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$, $\mathbf{r}\in\mathbb{R}^k$, $\boldsymbol{M}_r\in\mathbb{R}^{k\times d}$	✓	✓	✓	✓	✓
DistMult	$<\mathbf{h},\mathbf{r},\mathbf{t}>$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{R}^k$	✓	✗	✗	✗	✓
ComplEx	$\mathrm{Re}(<\mathbf{h},\mathbf{r},\bar{\mathbf{t}}>)$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{C}^k$	✓	✓	✓	✗	✓

Stanford CS224W: Knowledge Graph Completion: TransE

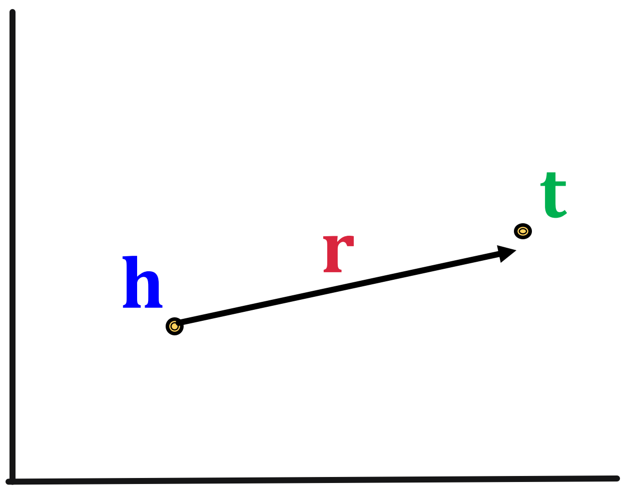
CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

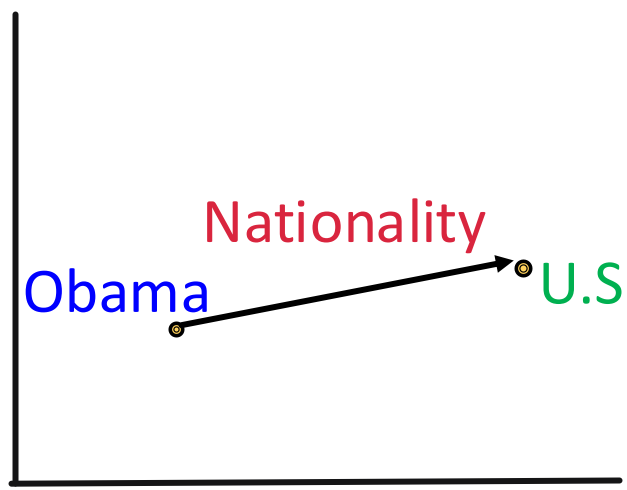
TransE

- **Intuition: Translation**
  For a triple (h, r, t), let $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$ be embedding vectors.
- **TransE:** $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ if the given link exists else $\mathbf{h} + \mathbf{r} \neq \mathbf{t}$

embedding vectors will appear in boldface

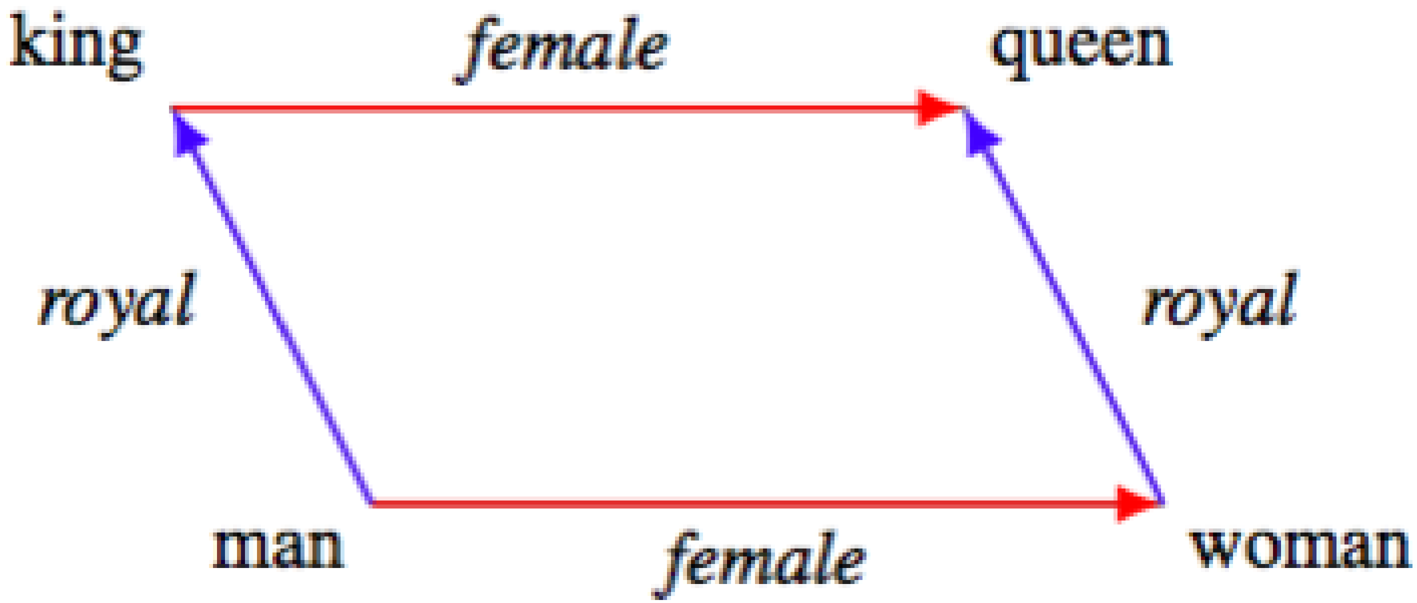
Entity scoring function: $f_r(h,t) = -||\mathbf{h} + \mathbf{r} - \mathbf{t}||$





TransE: Idea

- Entity embeddings



TransE: How to Learn

Algorithm 1 Learning TransE

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

initialize $r \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $r \in R$

$r \leftarrow r / \| r \|$ for each $r \in R$

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

loop

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

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

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

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

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

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

end for

Update embeddings w.r.t.

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

positive sample

negative sample

end loop

Initialize relations r and entities e uniformly, then normalize.
γ is the margin.

Sample triplet (h', r, t) that does not appear in the KG.

d represents distance (negative of score)

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

Connectivity Patterns in KG

- **Relations in a heterogeneous KG have different properties:**
  - Example:
    - **Symmetry:** If the edge $(h, \text{"Roommate"}, t)$ exists in KG, then the edge $(t, \text{"Roommate"}, h)$ should also exist.
    - **Inverse relation:** If the edge $(h, \text{"Advisor"}, t)$ exists in KG, then the edge $(t, \text{"Advisee"}, h)$ should also exist.
- Can we **categorize** these relation patterns?
- **Are KG embedding methods (e.g., TransE) expressive enough to model these patterns?**

Four Relation Patterns

- **Symmetric (Antisymmetric) Relations:**

$$r(h,t) \Rightarrow r(t,h) \;\; (r(h,t) \Rightarrow \neg r(t,h)) \quad \forall h,t$$

  - **Example**:
    - Symmetric: Family, Roommate
    - Antisymmetric: Hypernym (a word with a broader meaning: poodle vs. dog)
- **Inverse Relations:**

$$r_2(h,t) \Rightarrow r_1(t,h)$$

  - **Example** : (Advisor, Advisee)
- **Composition (Transitive) Relations:**

$$r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z) \quad \forall x,y,z$$

  - **Example**: My mother's husband is my father.
- **1-to-N relations:**

$$r(h,t_1), r(h,t_2), \ldots, r(h,t_n) \text{ are all True.}$$

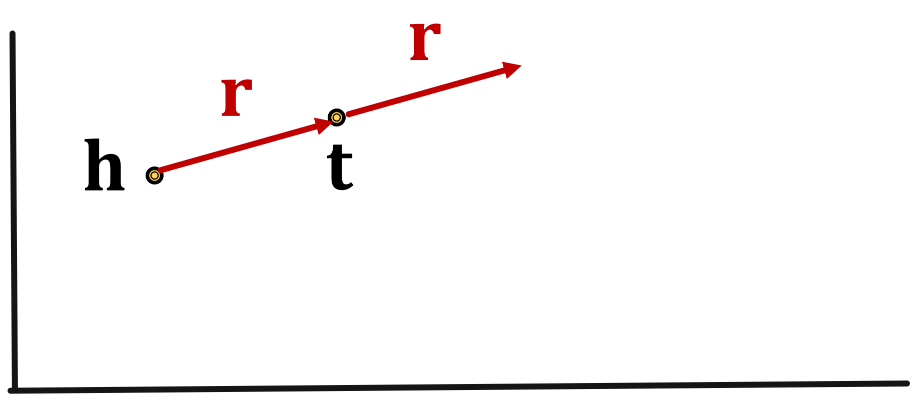
  - **Example**: r is "StudentsOf"

Antisymmetric Relations in TransE

- **Antisymmetric Relations:**

$$r(h,t) \Rightarrow \neg r(t,h) \quad \forall h,t$$

  - **Example**: Hypernym (a word with a broader meaning: poodle vs. dog)
- **TransE** can model antisymmetric relations ✓
  - $\mathbf{h} + \mathbf{r} = \mathbf{t}$, but $\mathbf{t} + \mathbf{r} \neq \mathbf{h}$

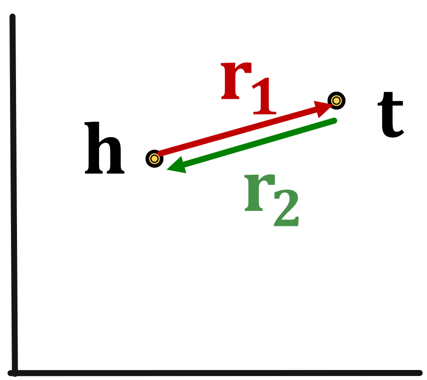


Inverse Relations in TransE

- **Inverse Relations:**

$$r_2(h,t) \Rightarrow r_1(t,h)$$

  - **Example** : (Advisor, Advisee)
- **TransE** can model inverse relations ✓
  - $\mathbf{h} + \mathbf{r}_2 = \mathbf{t}$, we can set $\mathbf{r}_1 = -\mathbf{r}_2$



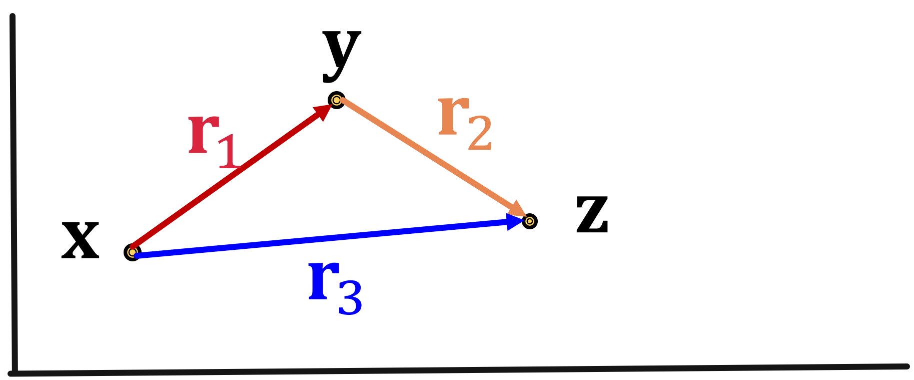
Composition in TransE

- **Composition (Transitive) Relations:**

  $$r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z$$

  - **Example**: My mother's husband is my father.
- **TransE** can model composition relations ✓

$$\mathbf{r}_3 = \mathbf{r}_1 + \mathbf{r}_2$$



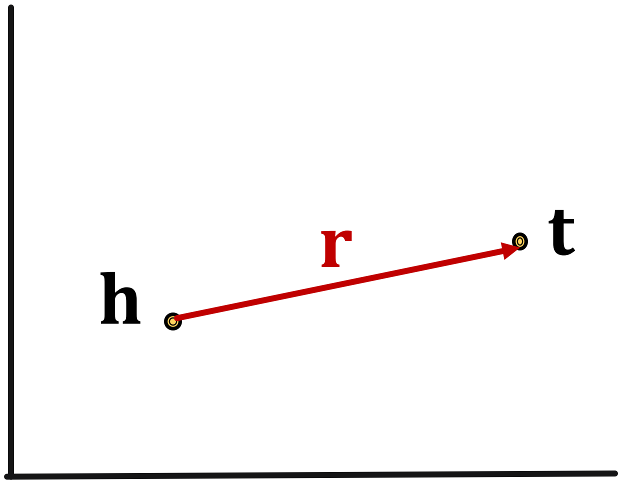
Limitation: Symmetric Relations

- **Symmetric Relations:**

$$r(h,t) \Rightarrow r(t,h) \quad \forall h,t$$

  - **Example**: Family, Roommate
- **TransE cannot** model symmetric relations ✘

  only if $\mathbf{r} = 0, \ \mathbf{h} = \mathbf{t}$



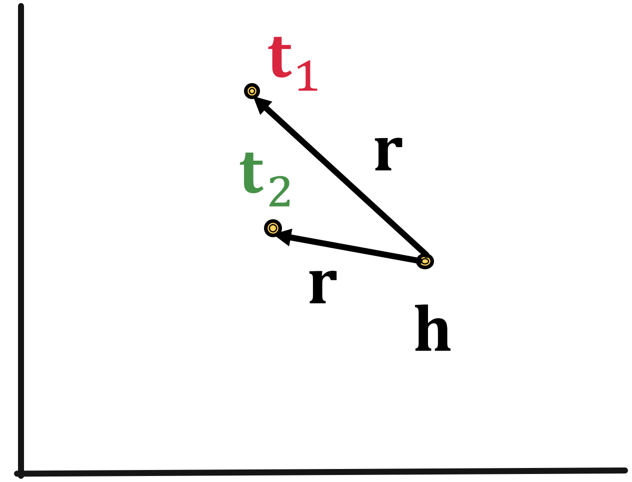
For all h, t that satisfy $r(h,t)$, $r(t,h)$ is also True, which means $||\mathbf{h} + \mathbf{r} - \mathbf{t}|| = 0$ and $||\mathbf{t} + \mathbf{r} - \mathbf{h}|| = 0$. Then $\mathbf{r} = 0$ and $\mathbf{h} = \mathbf{t}$, however h and t are two different entities and should be mapped to different locations.

Limitation: 1-to-N Relations

- **1-to-N Relations:**
  - **Example**: (h, r, t_1) and (h, r, t_2) both exist in the knowledge graph, e.g., r is "StudentsOf"
- **TransE cannot** model 1-to-N relations ✘
  - $\mathbf{t}_1$ and $\mathbf{t}_2$ will map to the same vector, although they are different entities

- $\mathbf{t}_1 = \mathbf{h} + \mathbf{r} = \mathbf{t}_2$
- $\mathbf{t}_1 \neq \mathbf{t}_2$

contradictory!



Today: KG Completion Models

What we learned so far:

Model	Score	Embedding	Sym.	Antisym.	Inv.	Compos.	1-to-N
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^k$	✗	✓	✓	✓	✗

Stanford CS224W: Knowledge Graph Completion: TransR

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

TransR

- **TransE models translation of any relation in the same embedding space.**

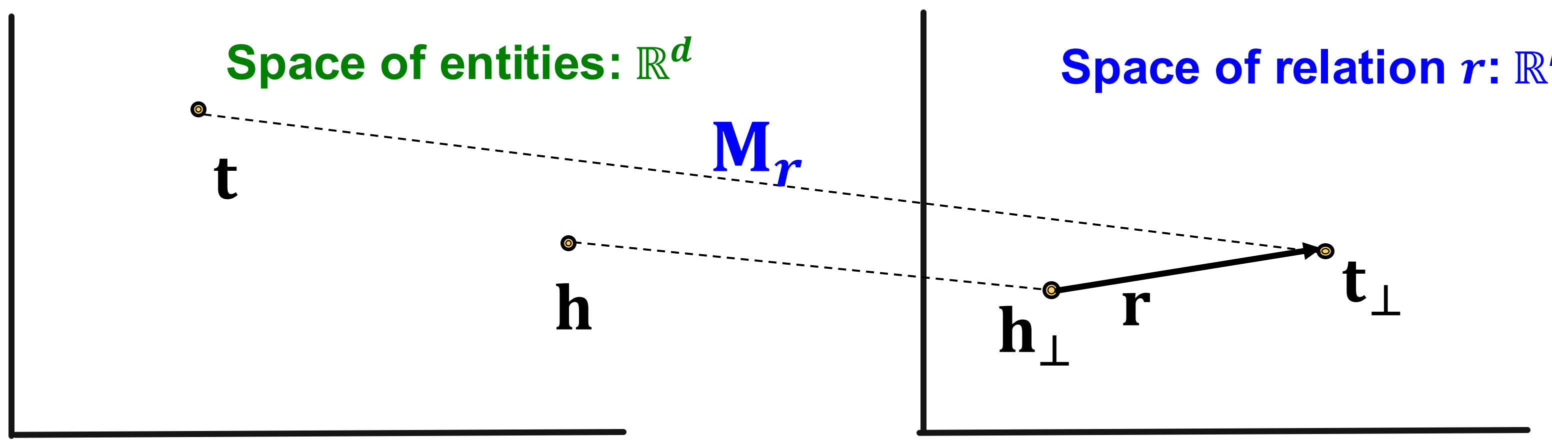
- **Can we design a new space for each relation and do translation in relation-specific space?**

- **TransR**: model **entities** as vectors in the entity space $\mathbb{R}^d$ and model each **relation** as vector in relation space $\mathbf{r} \in \mathbb{R}^k$ with $\mathbf{M}_r \in \mathbb{R}^{k \times d}$ as the projection matrix.

TransR

- **TransR**: model **entities** as vectors in the entity space $\mathbb{R}^d$ and model each **relation** as vector in relation space $\mathbf{r} \in \mathbb{R}^k$ with $\mathbf{M}_r \in \mathbb{R}^{k \times d}$ as the **projection matrix**.
- $\mathbf{h}_\perp = \mathbf{M}_r \mathbf{h},\ \mathbf{t}_\perp = \mathbf{M}_r \mathbf{t}$
- **Score function**: $f_r(h,t) = -||\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp||$

Use $\mathbf{M}_r$ to project from entity space $\mathbb{R}^d$ to relation space $\mathbb{R}^k$!



Symmetric Relations in TransR

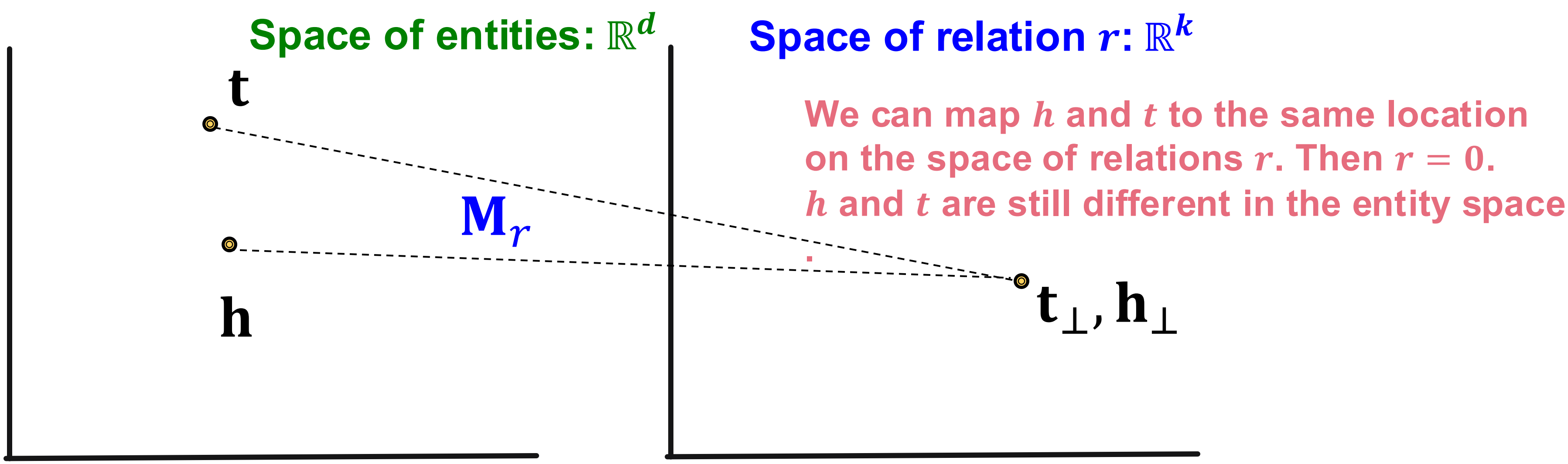
- **Symmetric Relations:**

$$r(h,t) \Rightarrow r(t,h) \quad \forall h,t$$

  - **Example**: Family, Roommate
- **TransR** can model symmetric relations

$$\mathbf{r} = 0,\ \mathbf{h}_{\perp} = \mathbf{M}_r \mathbf{h} = \mathbf{M}_r \mathbf{t} = \mathbf{t}_{\perp} ✓$$

Note different symmetric relations may have different $\mathbf{M}_r$



Antisymmetric Relations in TransR

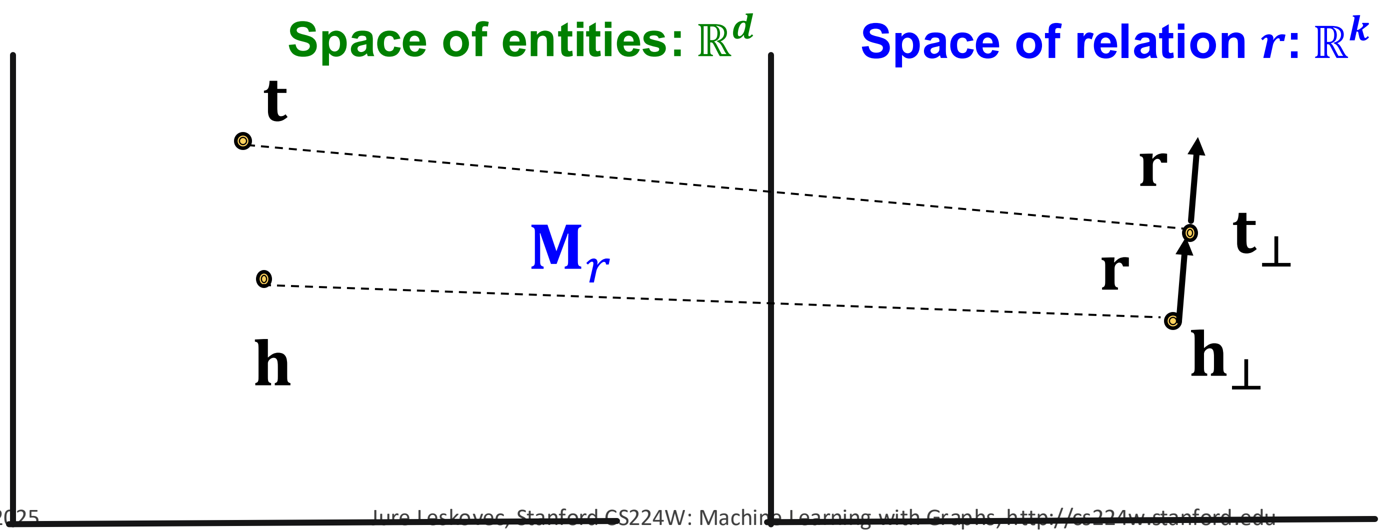
- **Antisymmetric Relations:**

$$r(h,t) \Rightarrow \neg r(t,h) \quad \forall h,t$$

  - **Example**: Hypernym
- **TransR** can model antisymmetric relations:

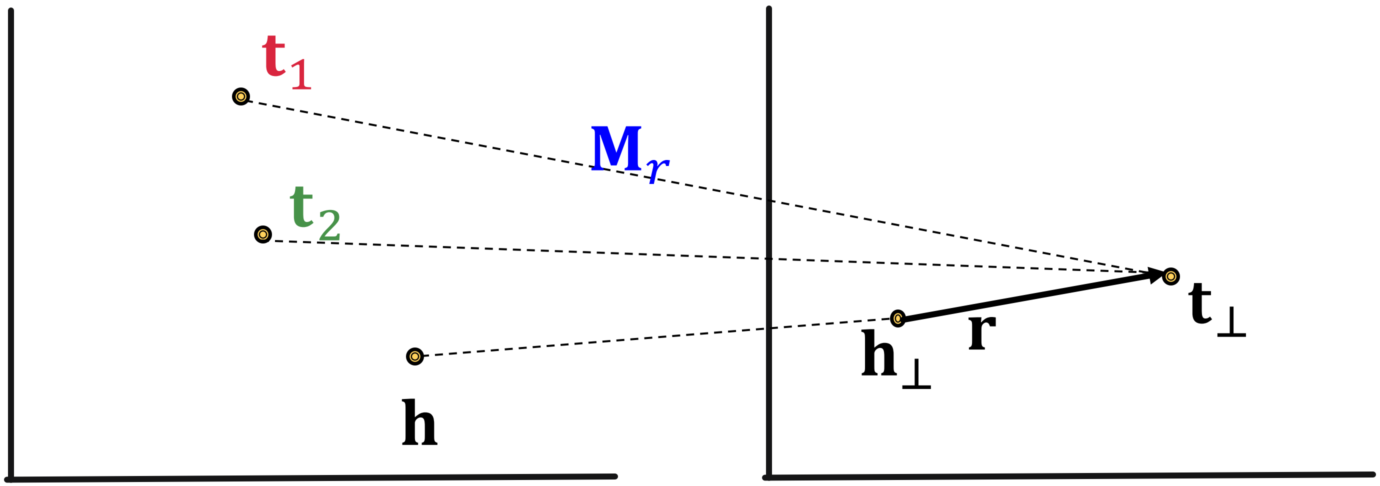
$$\mathbf{r} \neq 0, \mathbf{M}_r\mathbf{h} + \mathbf{r} = \mathbf{M}_r\mathbf{t},$$

$$\text{Then } \mathbf{M}_r\mathbf{t} + \mathbf{r} \neq \mathbf{M}_r\mathbf{h} \checkmark$$



1-to-N Relations in TransR

- **1-to-N Relations:**
  - **Example**: If (h, r, t_1) and (h, r, t_2) exist in the knowledge graph.
- TransR can model 1-to-N relations ✓
  - We can learn $\mathbf{M}_r$ so that $\mathbf{t}_\perp = \mathbf{M}_r \mathbf{t}_1 = \mathbf{M}_r \mathbf{t}_2$
  - Note that $\mathbf{t}_1$ does not need to be equal to $\mathbf{t}_2$!



Inverse Relations in TransR

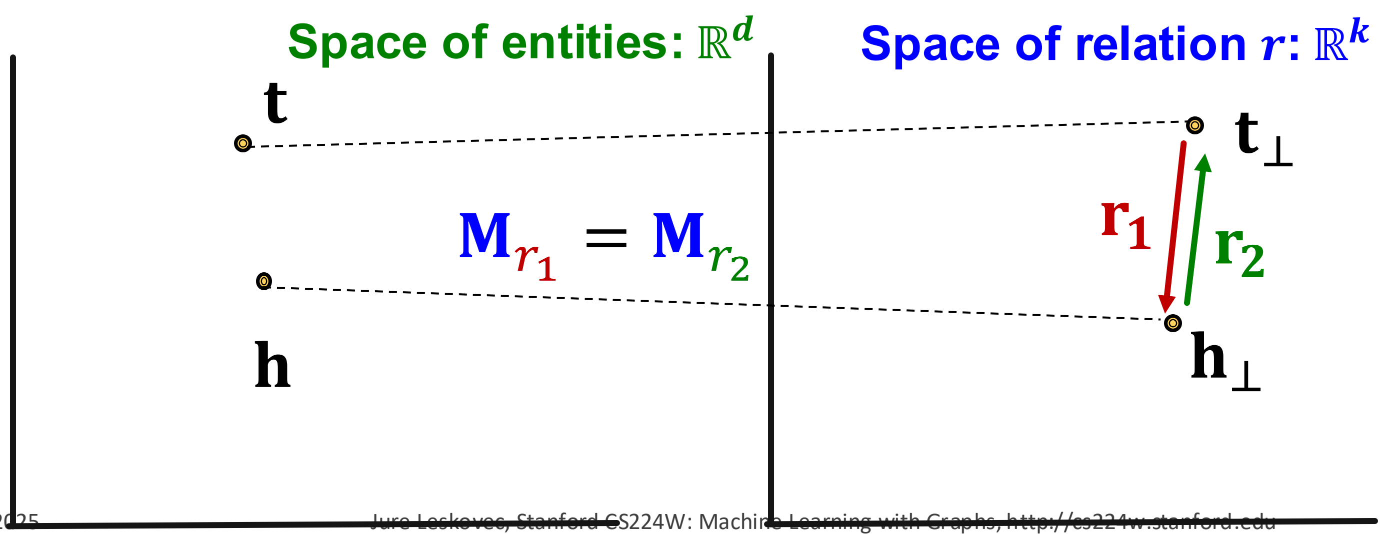
- **Inverse Relations:**

$$r_2(h,t) \Rightarrow r_1(t,h)$$

  - **Example** : (Advisor, Advisee)
- **TransR** can model inverse relations

$$\mathbf{r}_2 = -\mathbf{r}_1, \mathbf{M}_{r_1} = \mathbf{M}_{r_2}$$

Then $\mathbf{M}_{r_1}\mathbf{t} + \mathbf{r}_1 = \mathbf{M}_{r_1}\mathbf{h}$ and $\mathbf{M}_{r_2}\mathbf{h} + \mathbf{r}_2 = \mathbf{M}_{r_2}\mathbf{t}$ ✓



Composition Relations in TransR

- **Composition Relations:**

  $$r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z$$

  - **Example**: My mother's husband is my father.
- **TransR** can model composition relations

High-level intuition: TransR models a triple with linear functions. Linear functions are chainable!

- If f(x) and g(x) are linear, then f(g(x)) is also linear:
  - Let: $f(x)=a \cdot x+b$, $g(x)=c \cdot x+d$: then $f(g(x))= a(c \cdot x+d)+b$.

Composition Relations in TransR

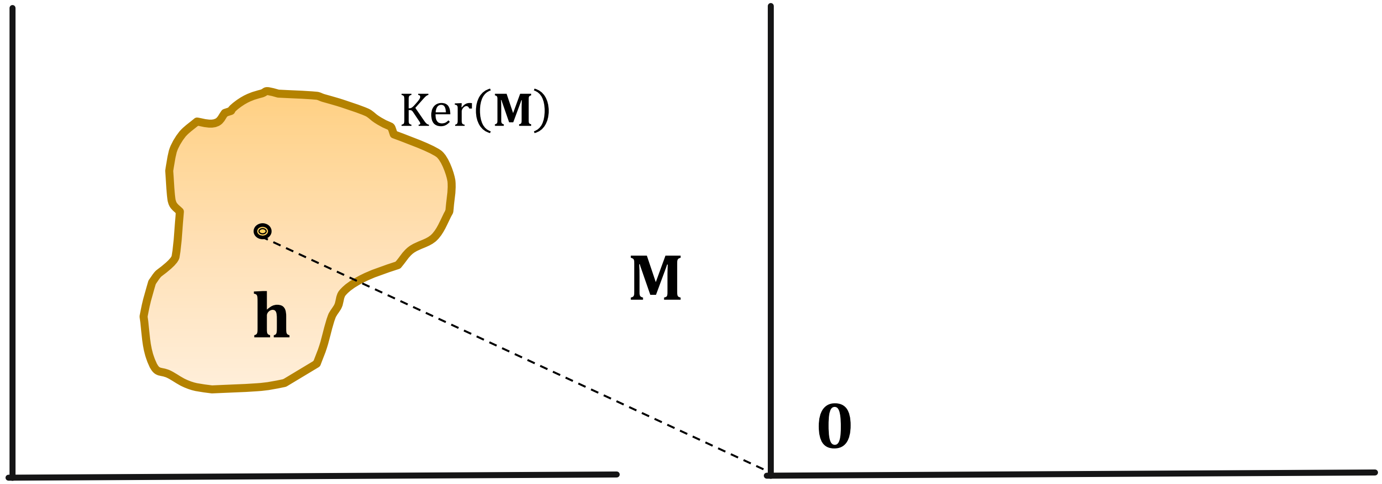
- **Composition Relations:**

$$r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z) \quad \forall x,y,z$$

Background:

Def: Kernel space of a matrix $\mathbf{M}$:

$$\mathbf{h} \in \mathrm{Ker}(\mathbf{M}), \text{ then } \mathbf{M} \cdot \mathbf{h} = \mathbf{0}$$



Composition Relations in TransR

- **Composition Relations:**

$$r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z) \quad \forall x,y,z$$

Assume $\mathbf{M}_{r_1}\mathbf{g}_1 = \mathbf{r}_1$ and $\mathbf{M}_{r_2}\mathbf{g}_2 = \mathbf{r}_2$

- For $r_1(x,y)$:

$r_1(x,y)$ exists $\Rightarrow \mathbf{M}_{r_1}\mathbf{x} + \mathbf{r_1} = \mathbf{M}_{r_1}\mathbf{y} \Rightarrow \mathbf{M}_{r_1}(\boldsymbol{y} - \boldsymbol{x}) = \boldsymbol{r}_1$

$$\mathbf{y} - \mathbf{x} \in \mathbf{g}_1 + \mathrm{Ker}(\mathbf{M}_{r_1}) \Rightarrow \mathbf{y} \in \mathbf{x} + \mathbf{g_1} + \mathrm{Ker}(\mathbf{M}_{r_1})$$

- Same for $r_2(y,z)$:

$r_2(y,z)$ exists $\Rightarrow \mathbf{M}_{r_2}\mathbf{y} + \mathbf{r_2} = \mathbf{M}_{r_2}\mathbf{z} \Rightarrow$

$$\mathbf{z} - \mathbf{y} \in \mathbf{g}_2 + \mathrm{Ker}(\mathbf{M}_{r_2}) \Rightarrow \mathbf{z} \in \mathbf{y} + \mathbf{g_2} + \mathrm{Ker}(\mathbf{M}_{r_2})$$

- Then, we have

$$\mathbf{z} \in \mathbf{x} + \mathbf{g_1} + \mathbf{g_2} + \mathrm{Ker}(\mathbf{M}_{r_1}) + \mathrm{Ker}(\mathbf{M}_{r_2})$$

Composition Relations in TransR

- **Composition Relations:**

$$r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z) \quad \forall x,y,z$$

We have $\mathbf{z} \in \mathbf{x} + \mathbf{g_1} + \mathbf{g_2} + \mathrm{Ker}(\mathbf{M}_{r_1}) + \mathrm{Ker}(\mathbf{M}_{r_2})$

- Construct $\mathbf{M}_{r_3}$, s.t.
  $\mathrm{Ker}(\mathbf{M}_{r_3}) = \mathrm{Ker}(\mathbf{M}_{r_1}) + \mathrm{Ker}(\mathbf{M}_{r_2})$

- **Since:**
  - $\dim\left(\mathrm{Ker}(\mathbf{M}_{r_3})\right) \geq \dim\left(\mathrm{Ker}(\mathbf{M}_{r_1})\right)$
  - $\mathbf{M}_{r_3}$ has the same shape as $\mathbf{M}_{r1}$

  we know $\mathbf{M}_{r_3}$ exists!

- Set $\mathbf{r}_3 = \mathbf{M}_{r_3}(\mathbf{g}_1 + \mathbf{g}_2)$

- We have $\mathbf{M}_{r_3}\mathbf{x} + \mathbf{r_3} = \mathbf{M}_{r_3}\mathbf{z}$

Today: KG Completion Models

What we learned so far:

Model	Score	Embedding	Sym.	Antisym.	Inv.	Compos.	1-to-N
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^k$	✘	✓	✓	✓	✘
TransR	$-\|\boldsymbol{M}_r\mathbf{h}+\mathbf{r} - \boldsymbol{M}_r\mathbf{t}\|$	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, $\mathbf{r} \in \mathbb{R}^k$, $\boldsymbol{M}_r \in \mathbb{R}^{k\times d}$	✓	✓	✓	✓	✓

Stanford CS224W: Knowledge Graph Completion: DistMult

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

Yang et al, Embedding Entities and Relations for Learning and Inference in Knowledge Bases, ICLR 2015

New Idea: Bilinear Modeling

- **So far:** The scoring function $f_r(h, t)$ is **negative of L1 / L2 distance** in **TransE** and **TransR**

- **Idea: Use bilinear** modeling:

  Score function: $f_r(h, t) = h \cdot A \cdot t$

  $\mathbf{h}, \mathbf{t} \in \mathbb{R}^k, \mathbf{A} \in \mathbb{R}^{k \times k}$

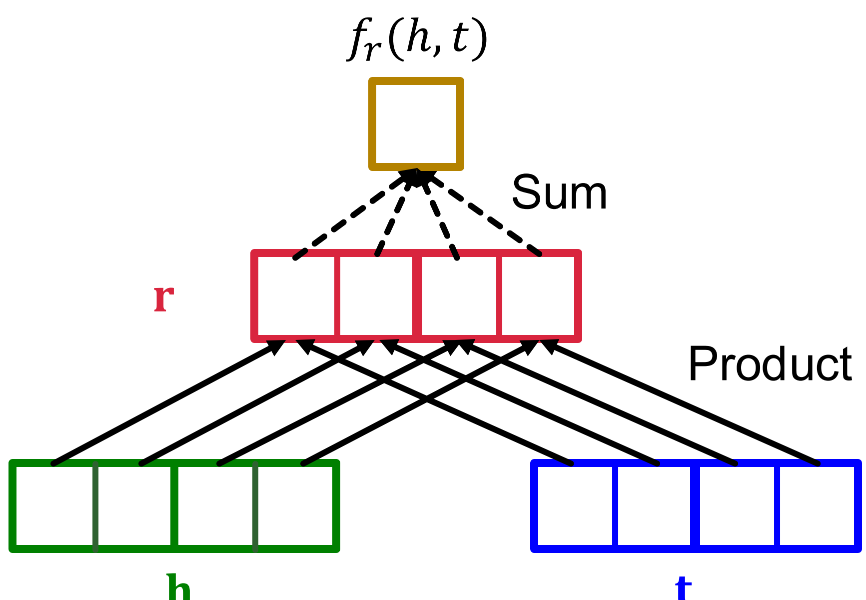
- **Problem: Too general and prone to overfitting**
  - Matrix A is too expressive
- **Fix: Limit A to be diagonal**
  - **This is called DistMult**

New Idea: Bilinear Modeling

- **DistMult**: Entities & relations are vectors in $\mathbb{R}^k$
- **Score function:**

$$f_r(h,t) = <\mathbf{h},\mathbf{r},\mathbf{t}> = \sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \mathbf{t}_i$$

- $\mathbf{h},\mathbf{r},\mathbf{t} \in \mathbb{R}^k$



DistMult

- **DistMult**: Entities and relations using vectors in $\mathbb{R}^k$
- **Score function:** $f_r(h,t) = <\mathbf{h},\mathbf{r},\mathbf{t}> = \sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \mathbf{t}_i$
  - $\mathbf{h},\mathbf{r},\mathbf{t} \in \mathbb{R}^k$
- **Intuition of the score function:** Can be viewed as a **cosine similarity** between $\mathbf{h} \cdot \mathbf{r}$ and $\mathbf{t}$

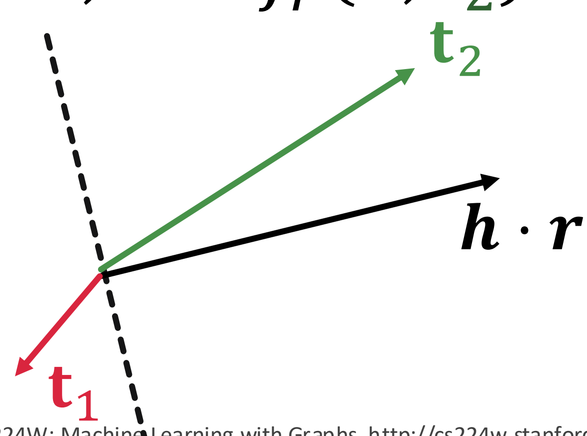
  where $\mathbf{h} \cdot \mathbf{r}$ is defined as $[\boldsymbol{h} \cdot \boldsymbol{r}]_{\boldsymbol{i}} = \boldsymbol{h}_{\boldsymbol{i}} \cdot \boldsymbol{r}_{\boldsymbol{i}}$

  Hadamard product
- **Example:**

$$f_r(h,t_1) < 0, \qquad f_r(h,t_2) > 0$$

$$\cos(\mathbf{h} \cdot \mathbf{r}, \mathbf{t}) = \frac{<\mathbf{h},\mathbf{r},\mathbf{t}>}{\|\mathbf{h} \cdot \mathbf{r}\|\|\mathbf{t}\|}$$

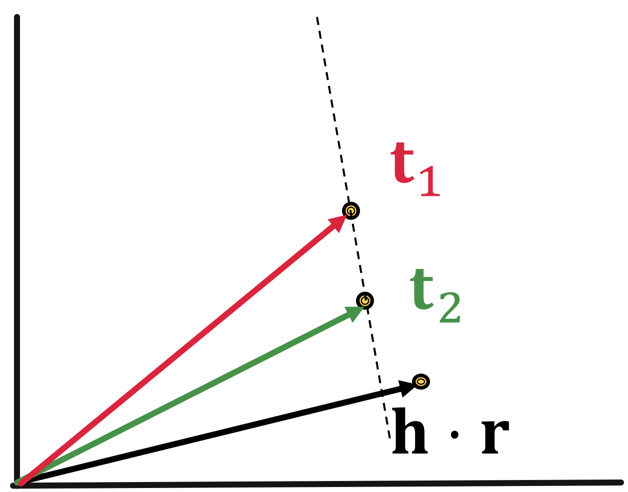
$$<\mathbf{h},\mathbf{r},\mathbf{t}> = \cos(\mathbf{h} \cdot \mathbf{r}, \mathbf{t})\|\mathbf{h} \cdot \mathbf{r}\|\|\mathbf{t}\|$$



1-to-N Relations in DistMult

- **1-to-N Relations:**
  - **Example**: If (h, r, t_1) and (h, r, t_2) exist in the knowledge graph
- **DistMult** can model 1-to-N relations ✓

$$< \mathbf{h}, \mathbf{r}, \mathbf{t}_1 > = < \mathbf{h}, \mathbf{r}, \mathbf{t}_2 >$$



Symmetric Relations in DistMult

- **Symmetric Relations:**

$$r(h,t) \Rightarrow r(t,h) \quad \forall h,t$$

  - **Example**: Family, Roommate
- **DistMult** can naturally model symmetric relations ✓

$$f_r(h,t) = <\mathbf{h},\mathbf{r},\mathbf{t}> = \sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \mathbf{t}_i =$$

$$<\mathbf{t},\mathbf{r},\mathbf{h}> = f_r(t,h)$$

Due to the commutative property of multiplication.

Limitation: Antisymmetric Relations

- **Antisymmetric Relations:**

$$r(h,t) \Rightarrow \neg r(t,h) \quad \forall h,t$$

  - **Example**: Hypernym

- **DistMult cannot** model antisymmetric relations

$$f_r(h,t) = <\mathbf{h},\mathbf{r},\mathbf{t}> = <\mathbf{t},\mathbf{r},\mathbf{h}> = f_r(t,h) \times$$

  - $r(h,t)$ and $r(t,h)$ always have same score!

DistMult cannot differentiate between head entity and tail entity! This means that all relations are modelled as symmetric regardless, i.e., even anti-symmetric relations will be represented as symmetric.

Limitation: Inverse Relations

- **Inverse Relations:**

$$r_2(h,t) \Rightarrow r_1(t,h)$$

  - **Example** : (Advisor, Advisee)
- **DistMult cannot** model inverse relations ✘
  - Assume DistMult does model inverse relations:

$$f_{r_2}(h,t) = < \mathbf{h}, \mathbf{r}_2, \mathbf{t} > = < \mathbf{t}, \mathbf{r}_1, \mathbf{h} > = f_{r_1}(t,h)$$

  - For example, $\mathbf{r}_2 = \mathbf{r}_1$ solves this (there are also exist solutions $\mathbf{r}_2 \neq \mathbf{r}_1$)
  - But semantically this does not make sense: **The embedding of "Advisor" relation should not be the same as "Advisee" relation.**

Limitation: Composition Relations

- **Composition Relations:**

  $$r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z$$

  - **Example**: My mother's husband is my father.

- **DistMult cannot** model composition of relations ✗
  - **Intuition:** Because dot product is commutative (a·b= b·a) **DistMult** does not distinguish between head and tail entities, so it cannot model composition.

Today: KG Completion Models

What we learned so far:

Model	Score	Embedding	Sym.	Antisym.	Inv.	Compos.	1-to-N
TransE	$-\|\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|\|$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{R}^k$	✗	✓	✓	✓	✗
TransR	$-\|\|\boldsymbol{M}_r\mathbf{h}+\mathbf{r}-\boldsymbol{M}_r\mathbf{t}\|\|$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$, $\mathbf{r}\in\mathbb{R}^k$, $\boldsymbol{M}_r\in\mathbb{R}^{k\times d}$	✓	✓	✓	✓	✓
DistMult	$<\mathbf{h},\mathbf{r},\mathbf{t}>$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{R}^k$	✓	✗	✗	✗	✓

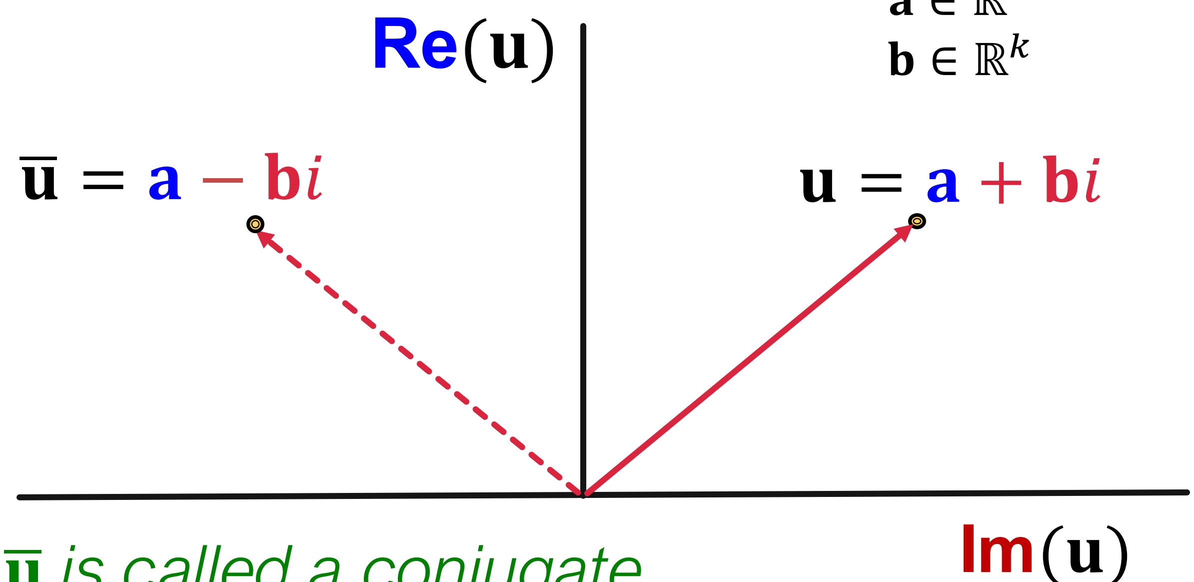
Stanford CS224W: Knowledge Graph Completion: ComplEx

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

ComplEx

- Based on Distmult, **ComplEx** embeds entities and relations in **Complex vector space**
- **ComplEx**: model entities and relations using vectors in $\mathbb{C}^k$

$\mathbf{u} \in \mathbb{C}^k$
$\mathbf{a} \in \mathbb{R}^k$
$\mathbf{b} \in \mathbb{R}^k$

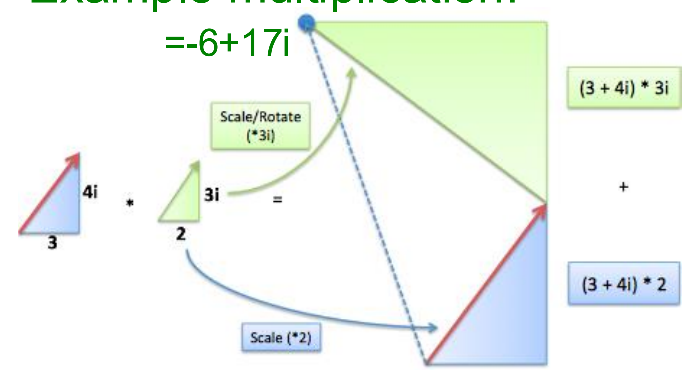


$\bar{\mathbf{u}}$ is called a conjugate

Complex multiplication:

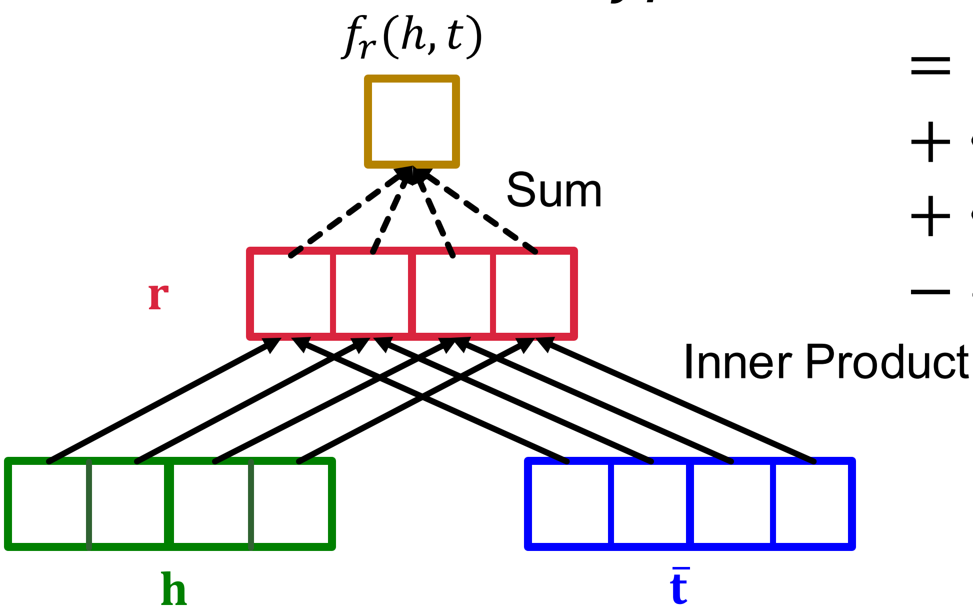
$(a + ib)(c + id) = (ac - bd) + i(ad + bc)$

Example multiplication:



ComplEx

- Based on Distmult, **ComplEx** embeds entities and relations in **Complex vector space**
- **ComplEx**: model entities and relations using vectors in $\mathbb{C}^k$
- **Score function** $f_r(h,t) = \mathrm{Re}(\sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \bar{\mathbf{t}}_i)$



$$
\begin{aligned}
&= \langle \mathrm{Re}(\mathbf{h}_i), \mathrm{Re}(\mathbf{r}_i), \mathrm{Re}(\mathbf{t}_i) \rangle \\
&+ \langle \mathrm{Re}(\mathbf{h}_i), \mathrm{Im}(\mathbf{r}_i), \mathrm{Im}(\mathbf{t}_i) \rangle \\
&+ \langle \mathrm{Im}(\mathbf{h}_i), \mathrm{Re}(\mathbf{r}_i), \mathrm{Im}(\mathbf{t}_i) \rangle \\
&- \langle \mathrm{Im}(\mathbf{h}_i), \mathrm{Im}(\mathbf{r}_i), \mathrm{Re}(\mathbf{t}_i) \rangle
\end{aligned}
$$

ComplEx Score Function

$$f_r(h,t) = \mathrm{Re}\left(\sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \bar{\mathbf{t}}_i\right)$$

$$= \sum_i \mathrm{Re}(\mathbf{h}_i \cdot \mathbf{r}_i \cdot \bar{\mathbf{t}}_i)$$

$$= \sum_i \mathrm{Re}((\mathrm{Re}(\mathbf{h}_i) + i\mathrm{Im}(\mathbf{h}_i)) \cdot (\mathrm{Re}(\mathbf{r}_i) + i\mathrm{Im}(\mathbf{r}_i)) \cdot (\mathrm{Re}(\mathbf{t}_i) - i\mathrm{Im}(\mathbf{t}_i)))$$

$$= \sum_i \begin{array}{l} \mathrm{Re}(\mathbf{h}_i)\mathrm{Re}(\mathbf{r}_i)\mathrm{Re}(\mathbf{t}_i) + \mathrm{Re}(\mathbf{h}_i)\mathrm{Im}(\mathbf{r}_i)\mathrm{Im}(\mathbf{t}_i) \\ +\mathrm{Im}(\mathbf{h}_i)\mathrm{Re}(\mathbf{r}_i)\mathrm{Im}(\mathbf{t}_i) - \mathrm{Im}(\mathbf{h}_i)\mathrm{Im}(\mathbf{r}_i)\mathrm{Re}(\mathbf{t}_i) \end{array}$$

$$= \langle \mathrm{Re}(\mathbf{h}_i), \mathrm{Re}(\mathbf{r}_i), \mathrm{Re}(\mathbf{t}_i) \rangle + \langle \mathrm{Re}(\mathbf{h}_i), \mathrm{Im}(\mathbf{r}_i), \mathrm{Im}(\mathbf{t}_i) \rangle$$
$$+ \langle \mathrm{Im}(\mathbf{h}_i), \mathrm{Re}(\mathbf{r}_i), \mathrm{Im}(\mathbf{t}_i) \rangle - \langle \mathrm{Im}(\mathbf{h}_i), \mathrm{Im}(\mathbf{r}_i), \mathrm{Re}(\mathbf{t}_i) \rangle$$

Antisymmetric Relations in ComplEx

- **Antisymmetric Relations:**

$$r(h,t) \Rightarrow \neg r(t,h) \quad \forall h,t$$

  - **Example**: Hypernym
- **ComplEx** can model antisymmetric relations ✓
  - The model is expressive enough to learn
    - **High** $f_r(h,t) = \mathrm{Re}(\sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \bar{\mathbf{t}}_i)$
    - **Low** $f_r(t,h) = \mathrm{Re}(\sum_i \boldsymbol{t}_i \cdot \mathbf{r}_i \cdot \bar{\boldsymbol{h}}_i)$

    Due to the asymmetric modeling using complex conjugate.

Symmetric Relations in ComplEx

- **Symmetric Relations:**

$$r(h,t) \Rightarrow r(t,h) \quad \forall h,t$$

  - **Example**: Family, Roommate
- **ComplEx** can model symmetric relations ✓
  - When $\mathrm{Im}(\mathbf{r}) = 0$, we have
    - $f_r(h,t) = \mathrm{Re}(\sum_i \mathbf{h}_i \cdot \mathbf{r}_i \cdot \bar{\mathbf{t}}_i) = \sum_i \mathrm{Re}(\mathbf{r}_i \cdot \mathbf{h}_i \cdot \bar{\mathbf{t}}_i) = \sum_i \mathbf{r}_i \cdot \mathrm{Re}(\mathbf{h}_i \cdot \bar{\mathbf{t}}_i) = \sum_i \mathbf{r}_i \cdot \mathrm{Re}(\bar{\mathbf{h}}_i \cdot \mathbf{t}_i) = \sum_i \mathrm{Re}(\mathbf{r}_i \cdot \bar{\mathbf{h}}_i \cdot \mathbf{t}_i) = f_r(t,h)$

Inverse Relations in ComplEx

- Inverse Relations:

$$r_2(h,t) \Rightarrow r_1(t,h)$$

  - **Example** : (Advisor, Advisee)
- **ComplEx** can model inverse relations ✓
  - $\mathbf{r}_1 = \bar{\mathbf{r}}_2$
  - Complex conjugate of $\mathbf{r}_2 = \underset{\mathbf{r}}{\operatorname{argmax}} \operatorname{Re}(<\mathbf{h}, \mathbf{r}, \bar{\mathbf{t}}>)$ is exactly $\mathbf{r}_1 = \underset{\mathbf{r}}{\operatorname{argmax}} \operatorname{Re}(<\mathbf{t}, \mathbf{r}, \bar{\mathbf{h}}>)$.

Composition and 1-to-N

- **Composition Relations:**

  $$r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z) \quad \forall x,y,z$$

  - **Example**: My mother's husband is my father.
- **1-to-N Relations:**
  - **Example**: If (h, r, t_1) and (h, r, t_2) exist in the knowledge graph
- **ComplEx** share the same property with **DistMult**
  - **Cannot model composition relations**
  - **Can model 1-to-N relations**

Today: KG Completion Models

What we learned so far:

Model	Score	Embedding	Sym.	Antisym.	Inv.	Compos.	1-to-N
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h},\mathbf{t},\mathbf{r} \in \mathbb{R}^k$	✘	✓	✓	✓	✘
TransR	$-\|\boldsymbol{M}_r\mathbf{h}+\mathbf{r} - \boldsymbol{M}_r\mathbf{t}\|$	$\mathbf{h},\mathbf{t} \in \mathbb{R}^k$, $\mathbf{r} \in \mathbb{R}^d$, $\boldsymbol{M}_r \in \mathbb{R}^{d\times k}$	✓	✓	✓	✓	✓
DistMult	$< \mathbf{h},\mathbf{r},\mathbf{t} >$	$\mathbf{h},\mathbf{t},\mathbf{r} \in \mathbb{R}^k$	✓	✘	✘	✘	✓
ComplEx	$\mathrm{Re}(< \mathbf{h},\mathbf{r},\bar{\mathbf{t}} >)$	$\mathbf{h},\mathbf{t},\mathbf{r} \in \mathbb{C}^k$	✓	✓	✓	✘	✓
RotateE	$-\|\mathbf{h}\circ\mathbf{r}-t\|$	$\mathbf{h},\mathbf{t},\mathbf{r} \in \mathbb{C}^k$	✓	✓	✓	✓	✘

- …Hadamard product: $\overset{G}{\begin{bmatrix} 3 & 5 & 7 \\ 4 & 9 & 8 \end{bmatrix}} \circ \overset{H}{\begin{bmatrix} 1 & 6 & 3 \\ 0 & 2 & 9 \end{bmatrix}} = \overset{N}{\begin{bmatrix} 3\times 1 & 5\times 6 & 7\times 3 \\ 4\times 0 & 9\times 2 & 8\times 9 \end{bmatrix}}$

TransE and RotatE: they both satisfy a weaker notion of 1-to-N - that many tails can be equidistant to r*h / r + h

Sun et al, RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space, ICLR 2019

KG Embeddings in Practice

1. Different KGs may have **drastically different relation patterns**!
2. There is not a general embedding that works for all KGs, use the **table** to select models
3. Try **TransE** for a quick run if the target KG does not have much symmetric relations
4. Then use more expressive models, e.g., **ComplEx**, **RotatE** (**TransE** in Complex space)

Empirical comparison

Model	FB15k-237			WN18RR		
	MR↓	MRR↑	H10↑	MR↓	MRR↑	H10↑
TransE	357	.294	.465	3384	.226	.501
TransR						
DisMult	254	.241	.419	5110	.43	.49
ComplEx	339	.247	.428	5261	.44	.51
RotatE	**177**	**0.338**	**0.533**	**3340**	**0.476**	**0.571**

Summary of Knowledge Graph

- Link prediction / Graph completion is one of the prominent tasks on knowledge graphs
- Introduce **TransE** / **TransR** / **DistMult** / **ComplEx** models with different embedding space and expressiveness
- **Next:** Reasoning in Knowledge Graphs