ANNOUNCEMENTS

• Colab 2 due today
  • When you submit, you should get 0/0 on your assignment – this is because our test cases are hidden and will be graded after the assignment deadline
  • However, we have a simple autograder to make sure you are zipping files correctly: you should *not* see any errors (e.g., ModuleNotFoundError)
  • For submission details, refer Ed post ("Colab 2 released")

• Colab 3 out today
Heterogeneous graphs: a graph with multiple relation types
Recap: Relational GCN

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

![Diagram of Relational GCN](image)

**Input graph**

**Neural networks**

**Target node**
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
- **Node types**: drug, disease, adverse event, protein, pathways
- **Relation types**: has_func, causes, assoc, treats, is_a
Examples of knowledge graphs

- Google Knowledge Graph
- Amazon Product Graph
- Facebook Graph API
- IBM Watson
- Microsoft Satori
- Project Hanover/Literome
- LinkedIn Knowledge Graph
- Yandex Object Answer
Applications of Knowledge Graphs

- Serving information:

  Latest films by the director of Titanic

  **Movies featuring James Cameron**

  Image credit: Bing
Applications of Knowledge Graphs

- Question answering and conversation agents
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

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

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

---


Stanford CS224W: Knowledge Graph Completion

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu
Given an enormous KG, can we complete the KG?

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

**Example task:** predict the tail “Science Fiction” for (“J.K. Rowling”, “genre”)
Simplest encoding approach: **encoder is just an embedding-lookup**

- **Z**
  - **embedding matrix**
  - **embedding vector for a specific node**
  - **one column per node**
  - **Dimension/size of embeddings**

```plaintext
Z = 
```
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 the embedding/vector space \(\mathbb{R}^d\).
  - Associate entities and relations with shallow embeddings
  - Note we do not learn a GNN here!
- Given a true 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 closeness?
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)

<table>
<thead>
<tr>
<th>Model</th>
<th>Score</th>
<th>Embedding</th>
<th>Sym.</th>
<th>Antisym.</th>
<th>Inv.</th>
<th>Compos.</th>
<th>1-to-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>(-|h + r - t|)</td>
<td>h, t, r ∈ ℝ^k</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>TransR</td>
<td>(-|M_r h + r - M_r t|)</td>
<td>h, t, r ∈ ℝ^k, M_r ∈ ℝ^{d×k}</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DistMult</td>
<td>(&lt; h, r, t &gt;)</td>
<td>h, t, r ∈ ℝ^k</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>ComplEx</td>
<td>Re((&lt; h, r, \bar{t} &gt;))</td>
<td>h, t, r ∈ ℂ^k</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>
**Translation Intuition:**
For a triple \((h, r, t)\), \(h, r, t \in \mathbb{R}^d\),
\(h + r \approx t\) if the given fact is true
else \(h + r \neq t\)

**Scoring function:**
\[ f_r(h, t) = -||h + r - t|| \]


embedding vectors will appear in boldface
Algorithm 1 Learning TransE

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

1: \textbf{initialize} \( \ell \leftarrow \text{uniform}(\frac{-6}{\sqrt{k}}, \frac{6}{\sqrt{k}}) \) for each \( \ell \in L \)
2: \( \ell \leftarrow \ell / \| \ell \| \) for each \( \ell \in L \)
3: \( e \leftarrow \text{uniform}(\frac{-6}{\sqrt{k}}, \frac{6}{\sqrt{k}}) \) for each entity \( e \in E \)

4: \textbf{loop}
5: \( e \leftarrow e / \| e \| \) for each entity \( e \in E \)
6: \( S_{batch} \leftarrow \text{sample}(S, b) \) // sample a minibatch of size \( b \)
7: \( T_{batch} \leftarrow \emptyset \) // initialize the set of pairs of triplets
8: \textbf{for} \((h, l, t) \in S_{batch}\) \textbf{do}
9: \( (h', l, t') \leftarrow \text{sample}(S_{(h,l,t)}') \) // sample a corrupted triplet
10: \( T_{batch} \leftarrow T_{batch} \cup \{(h, l, t), (h', l, t')\} \)
11: \textbf{end for}
12: Update embeddings w.r.t. \[
\sum_{(h, l, t), (h', l, t') \in T_{batch}} \nabla [\gamma + d(h + l, t) - d(h' + l, t')]_{+}
\]
13: \textbf{end loop}

Entities and relations are initialized uniformly, and normalized.

Negative sampling with triplet 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.
Relations in a heterogeneous KG have different properties:

- **Example:**
  - **Symmetry:** If the edge \((h, "Roommate", t)\) exists in KG, then the edge \((t, "Roommate", h)\) should also exist.
  - **Inverse relation:** If the edge \((h, "Advisor", t)\) exists in KG, then the edge \((t, "Advisee", h)\) should also exist.

- **Can we categorize these relation patterns?**
- **Are KG embedding methods (e.g., TransE) expressive enough to model these patterns?**
Symmetric (Antisymmetric) Relations:

\[ r(h, t) \Rightarrow r(t, h) \ (r(h, t) \Rightarrow \neg r(t, h)) \ \forall h, t \]

**Example:**
- Symmetric: Family, Roommate
- Antisymmetric: Hyponym

**Inverse Relations:**

\[ r_2(h, t) \Rightarrow r_1(t, h) \]

**Example:** (Advisor, Advisee)

**Composition (Transitive) Relations:**

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \ \forall x, y, z \]

**Example:** My mother’s husband is my father.

**1-to-N relations:**

\[ r(h, t_1), r(h, t_2), \ldots, r(h, t_n) \] are all True.

**Example:** \( r \) is “StudentsOf”
Antisymmetric Relations:

\[ r(h, t) \implies \neg r(t, h) \quad \forall h, t \]

- **Example**: Hypernym

- **TransE** can model antisymmetric relations
  - \( h + r = t \), but \( t + r \neq h \)
Inverse Relations:

\[ r_2(h, t) \Rightarrow r_1(t, h) \]

- **Example**: (Advisor, Advisee)

**TransE** can model inverse relations

- \( h + r_2 = t \), we can set \( r_1 = -r_2 \)
Composition (Transitive) Relations:

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

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

**TransE** can model composition relations

\[ r_3 = r_1 + r_2 \]
Limitation: Symmetric Relations

- **Symmetric Relations:**
  \[ r(h, t) \Rightarrow r(t, h) \quad \forall h, t \]

  - **Example:** Family, Roommate

- **TransE cannot** model symmetric relations \( \times \)
  only if \( r = 0, \ h = t \)

For all \( h, t \) that satisfy \( r(h, t), r(t, h) \) is also true, which means \( ||h + r - t|| = 0 \) and \( ||t + r - h|| = 0 \). Then \( r = 0 \) and \( h = t \), 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 \(\times\)
  - \(t_1\) and \(t_2\) will map to the same vector, although they are different entities

\[ t_1 = h + r = t_2 \]
\[ t_1 \neq t_2 \quad \text{contradictory!} \]
Stanford CS224W: Knowledge Graph Completion: 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: model entities as vectors in the entity space $\mathbb{R}^d$ and model each relation as vector in relation space $r \in \mathbb{R}^k$ with $M_r \in \mathbb{R}^{k \times d}$ as the projection matrix.

- $h_\perp = M_r h$, $t_\perp = M_r t$
- Score function: $f_r(h, t) = -||h_\perp + r - t_\perp||$

Use $M_r$ to project from entity space $\mathbb{R}^d$ to relation space $\mathbb{R}^k$!
Symmetric Relations:
\[ r(h, t) \Rightarrow r(t, h) \quad \forall h, t \]

- **Example**: Family, Roommate

**TransR** can model symmetric relations

\[ r = 0, \quad h_\perp = M_r h = M_r t = t_\perp \checkmark \]

We can map \( h \) and \( t \) to the same location on the space of relation \( r \). \( h \) and \( t \) are still different in the entity space.
- **Antisymmetric Relations:**
  \[ r(h, t) \implies \neg r(t, h) \quad \forall h, t \]

  - **Example:** Hypernym

  - **TransR** can model antisymmetric relations
    \[ r \neq 0, M_r h + r = M_r t, \]
    Then \[ M_r t + r \neq M_r h \checkmark \]
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 \(M_r\) so that \(t_\perp = M_r t_1 = M_r t_2\)

  - Note that \(t_1\) does not need to be equal to \(t_2\)!
Inverse Relations:

\[ r_2(h, t) \Rightarrow r_1(t, h) \]

- **Example**: (Advisor, Advisee)

- **TransR** can model inverse relations

\[
\begin{align*}
  r_2 &= -r_1, M_{r_1} = M_{r_2} \\
  \text{Then } M_{r_1} t + r_1 &= M_{r_1} h \text{ and } M_{r_2} h + r_2 &= M_{r_2} t 
\end{align*}
\]

**Space of entities**: \( \mathbb{R}^d \)  

**Space of relation** \( r \): \( \mathbb{R}^k \)
Composition Relations:

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

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

High-level intuition: TransR models a triple with linear functions, they are chainable.
Composition Relations:

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

Background:
Kernel space of a matrix \( M \):

\[ h \in \text{Ker}(M), \text{ then } Mh = 0 \]
Composition Relations:

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

Assume \( M_{r_1} g_1 = r_1 \) and \( M_{r_2} g_2 = r_2 \)

- For \( r_1(x, y) \):
  \( r_1(x, y) \) exists \( \Rightarrow M_{r_1} x + r_1 = M_{r_1} y \rightarrow y - x \in g_1 + \text{Ker}(M_{r_1}) \rightarrow y \in x + g_1 + \text{Ker}(M_{r_1}) \)

- Same for \( r_2(y, z) \):
  \( r_2(y, z) \) exists \( \Rightarrow M_{r_2} y + r_2 = M_{r_2} z \rightarrow z - y \in g_2 + \text{Ker}(M_{r_2}) \rightarrow z \in y + g_2 + \text{Ker}(M_{r_2}) \)

- Then,

We have \( z \in x + g_1 + g_2 + \text{Ker}(M_{r_1}) + \text{Ker}(M_{r_2}) \)
Composition Relations:

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

We have \( z \in x + g_1 + g_2 + \text{Ker}(M_{r_1}) + \text{Ker}(M_{r_2}) \)

- Construct \( M_{r_3} \), s.t. \( \text{Ker}(M_{r_3}) = \text{Ker}(M_{r_1}) + \text{Ker}(M_{r_2}) \)
- Since
  - \( \dim \left( \text{Ker}(M_{r_3}) \right) \geq \dim \left( \text{Ker}(M_{r_1}) \right) \)
  - \( M_{r_3} \) has the same shape as \( M_{r_1} \)

We know \( M_{r_3} \) exists!

- Set \( r_3 = M_{r_3}(g_1 + g_2) \)
- We are done! We have \( M_{r_3}x + r_3 = M_{r_3}z \)
Stanford CS224W: Knowledge Graph Completion
DistMult

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

- **So far:** The scoring function $f_r(h, t)$ is negative of L1 / L2 distance in TransE and TransR
- Another line of KG embeddings adopt **bilinear** modeling
- **DistMult:** Entities and relations using vectors in $\mathbb{R}^k$
- **Score function:** $f_r(h, t) = \langle h, r, t \rangle = \sum_i h_i \cdot r_i \cdot t_i$
- $h, r, t \in \mathbb{R}^k$

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

- **DistMult**: Entities and relations using vectors in $\mathbb{R}^k$
- **Score function**: $f_r(h, t) = \langle h, r, t \rangle = \sum_i h_i \cdot r_i \cdot t_i$
  - $h, r, t \in \mathbb{R}^k$
- **Intuition of the score function**: Can be viewed as a cosine similarity between $h \cdot r$ and $t$
  - where $h \cdot r$ is defined as $\sum_i h_i \cdot r_i$
- **Example**:
  
  $$f_r(h, t_1) < 0, \quad f_r(h, t_2) > 0$$
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
    \[< h, r, t_1 > = < h, r, t_2 >\]
Symmetric Relations:

\[ r(h, t) \Rightarrow r(t, h) \quad \forall h, t \]

- **Example**: Family, Roommate

**DistMult** can naturally model symmetric relations

\[
 f_r(h, t) = <h, r, t> = \sum_i h_i \cdot r_i \cdot t_i = <t, r, h> = f_r(t, h)
\]
**Limitation: Antisymmetric Relations**

- **Antisymmetric Relations:**
  \[ r(h, t) \implies \neg r(t, h) \quad \forall h, t \]
  - **Example:** Hypernym

- **DistMult cannot** model antisymmetric relations
  \[ f_r(h, t) = \langle h, r, t \rangle = \langle t, r, h \rangle > f_r(t, h) \times \]
  - \( r(h, t) \) and \( r(t, h) \) always have same score!
Limitation: Inverse Relations

- **Inverse Relations:**
  \[ r_2(h, t) \Rightarrow r_1(t, h) \]
  - **Example:** (Advisor, Advisee)

- **DistMult cannot** model inverse relations \( \times \)
  - If it does model inverse relations:
    \[ f_{r_2}(h, t) =< h, r_2, t > =< t, r_1, h > = f_{r_1}(t, h) \]
    - This means \( r_2 = r_1 \)
    - But semantically this does not make sense: The embedding of “Advisor” should not be the same with “Advisee”.

2/9/2023
## Limitation: Composition Relations

- **Composition Relations:**

  \[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

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

- **DistMult cannot** model composition relations ×

  - **Intuition:** *DistMult* defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.
**Limitation: Composition Relations**

- **Composition Relations:**
  \[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]
  - **Example:** My mother’s husband is my father.
  - **DistMult cannot** model composition relations \( \times \)
  - **Intuition:** DistMult defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.

**Detailed derivation**

Pick one \( y \) s.t. \( f_{r_1}(x, y) > 0 \), e.g., \( y_2 \)

Then \( y_2 \cdot r_2 \) defines a new hyperplane.
Limitation: Composition Relations

- **Composition Relations:**
  \[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]
  - **Example:** My mother’s husband is my father.
  - **DistMult cannot** model composition relations ×
  - **Intuition:** DistMult defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.

**Detailed derivation**

Pick another \( y \) s.t. \( f_{r_1}(x, y) > 0 \), e.g., \( y_3 \)
Then \( y_3 \cdot r_2 \) defines another hyperplane.
**Limitation: Composition Relations**

- **Composition Relations:**
  
  $$r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \ \forall x, y, z$$

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

- **DistMult cannot** model composition relations 🗹

- **Intuition:** DistMult defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.

---

**Detailed derivation**

Combine both hyperplanes together, then for all \(z\) in the shadow area, there exists \(y \in \{y_2, y_3\}\), s.t., \(f_{r_2}(y, z) > 0\).
**Composition Relations:**

\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]

- **Example:** My mother’s husband is my father.
- **DistMult cannot** model composition relations \( \times \)
  - **Intuition:** DistMult defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.

**Detailed derivation**

Combine both hyperplanes together, then for all \( z \) in the shadow area, there exists \( y \in \{ y_2, y_3 \} \), s.t., \( f_{r_2}(y, z) > 0 \)
Limitation: Composition Relations

- **Composition Relations:**
  \[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]
  - **Example:** My mother’s husband is my father.

- **DistMult cannot** model composition relations \( \times \)
  - **Intuition:** DistMult defines a hyperplane for each (head, relation), the union of the hyperplane induced by multi-hops of relations, e.g., \((r_1, r_2)\), cannot be expressed using a single hyperplane.

**Detailed derivation**

According to the composition relations, we also want \( f_{r_3}(x, z) > 0, \forall z \in \{\text{shadow area}\} \).

However, this area inherently cannot be expressed by a single hyperplane defined by \( x \cdot r_3 \), no matter what \( r_3 \) is.
Stanford CS224W: Knowledge Graph Completion: ComplEx
Based on Distmult, **ComplEx** embeds entities and relations in **Complex vector space**

**ComplEx**: model entities and relations using vectors in $\mathbb{C}^k$

$\bar{u} = a - bi$

$u = a + bi$

$\bar{u}$ is called a conjugate
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** $f_r(h, t) = \text{Re}(\sum_i h_i \cdot r_i \cdot \bar{t}_i)$
**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** \( f_r(h, t) = \Re(\sum_i h_i \cdot r_i \cdot \bar{t}_i) \)
    - **Low** \( f_r(t, r) = \Re(\sum_i t_i \cdot r_i \cdot \bar{h}_i) \)
  
Due to the asymmetric modeling using complex conjugate.
Symmetric Relations: 

\[ r(h, t) \Rightarrow r(t, h) \quad \forall h, t \]

- **Example**: Family, Roommate

**ComplEx** can model symmetric relations

- When \( \text{Im}(r) = 0 \), we have

\[
f_r(h, t) = \text{Re}(\sum_i h_i \cdot r_i \cdot \bar{t}_i) = \sum_i \text{Re}(r_i \cdot h_i \cdot \bar{t}_i) = \sum_i r_i \cdot \text{Re}(h_i \cdot \bar{t}_i) = \sum_i r_i \cdot \text{Re}(\bar{h}_i \cdot t_i) = \sum_i \text{Re}(r_i \cdot \bar{h}_i \cdot t_i) = f_r(t, h)
\]
Inverse Relations:

\[ r_2(h, t) \Rightarrow r_1(t, h) \]

- **Example**: (Advisor, Advisee)

**ComplEx** can model inverse relations ✓

- \( r_1 = \bar{r}_2 \)
- Complex conjugate of
  
  \[ r_2 = \arg\max_r \text{Re}(\langle h, r, \bar{t} \rangle) \text{ is exactly } \]
  \[ r_1 = \arg\max_r \text{Re}(\langle t, r, \bar{h} \rangle). \]
Composition Relations:
\[ r_1(x, y) \land r_2(y, z) \Rightarrow r_3(x, z) \quad \forall x, y, z \]
- 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
### Expressiveness of All Models

- Properties and expressive power of different KG completion methods:

<table>
<thead>
<tr>
<th>Model</th>
<th>Score</th>
<th>Embedding</th>
<th>Sym.</th>
<th>Antisym.</th>
<th>Inv.</th>
<th>Compos.</th>
<th>1-to-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>$-</td>
<td></td>
<td>h + r - t</td>
<td></td>
<td>$</td>
<td>$h, t, r \in \mathbb{R}^k$</td>
<td>×</td>
</tr>
<tr>
<td>TransR</td>
<td>$-</td>
<td></td>
<td>M_r h + r - M_r t</td>
<td></td>
<td>$</td>
<td>$h, t, r \in \mathbb{R}^k$, $M_r \in \mathbb{R}^{d \times k}$</td>
<td>✓</td>
</tr>
<tr>
<td>DistMult</td>
<td>$&lt; h, r, t &gt;$</td>
<td>$h, t, r \in \mathbb{R}^k$</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>ComplEx</td>
<td>Re($&lt; h, r, \bar{t} &gt;$)</td>
<td>$h, t, r \in \mathbb{C}^k$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>
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)
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