Heterogeneous Graphs

Heterogeneous Graphs: Motivation

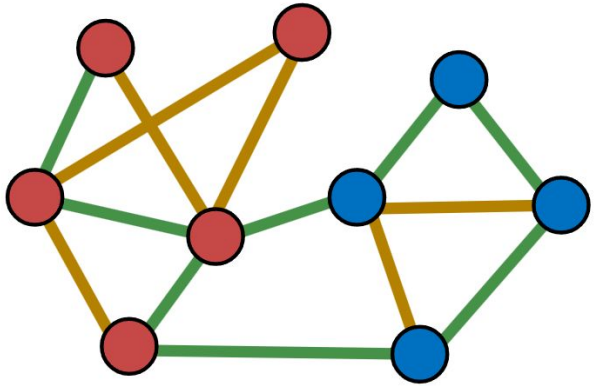
8 possible relation types!

(Paper, Cite, Paper)

(Paper, Like, Paper)

(Paper, Cite, Author)

(Paper, Like, Author)



(Author, Cite, Author)

(Author, Like, Author)

(Author, Cite, Paper)

(Author, Like, Paper)

Relation types: (node_start, edge, node_end)

- We use **relation type to describe an edge** (as opposed to edge type)
- Relation type better captures the interaction between nodes and edges

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Heterogeneous Graphs

- **A heterogeneous graph is defined as**

$$\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E}, \tau, \phi)$$

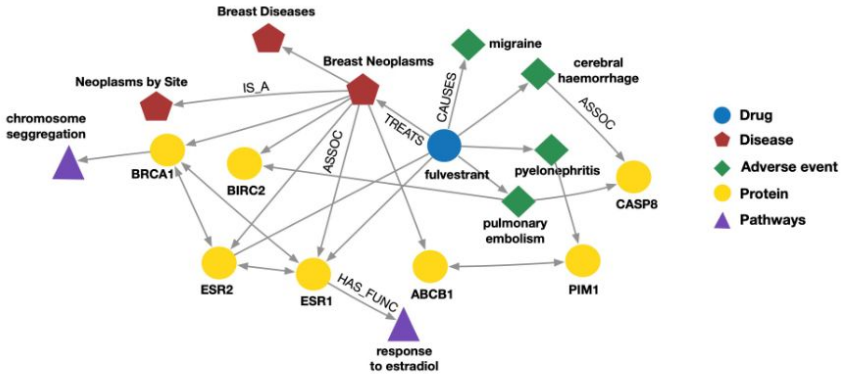
  - Nodes **with node types** $v \in V$
    - **Node type** for node v: $\tau(v)$
  - Edges **with edge types** $(u, v) \in E$
    - **Edge type** for edge (u, v): $\phi(u, v)$
    - **Relation type** for edge e is a tuple: $r(u, v) = (\tau(u), \phi(u, v), \tau(v))$

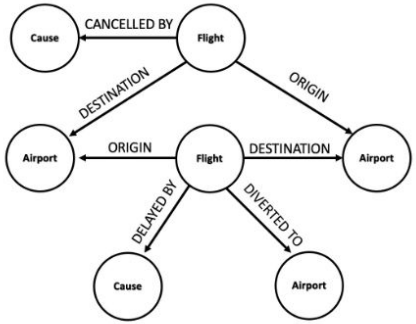
An edge can be described as a pair of nodes

- There are other definitions for heterogeneous graphs as well – describe **graphs with node & edge types**

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Many Graphs are Heterogeneous Graphs (1)





Biomedical Knowledge Graphs

Example node: Migraine
Example relation: (fulvestrant, Treats, Breast Neoplasms)
Example node type: Protein
Example edge type: Causes

Event Graphs

Example node: SFO
Example relation: (UA689, Origin, LAX)
Example node type: Flight
Example edge type: Destination

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Discussions: Type or Feature?

- **Observation:** We can also treat types of nodes and edges as features
  - **Example:** Add a one-hot indicator for nodes and edges
    - Append feature [1, 0] to each “author node”; Append feature [0, 1] to each “paper node”
    - Similarly, we can assign edge features to edges with different types
  - Then, a heterogeneous graph reduces to a standard graph
- **When do we need a heterogeneous graph?**

Taken from Stanford CS224W course: http://cs244w.stanford.edu

When do we need a heterogeneous graph

- **Case 1:** Different node/edge types **have different shapes of features**
  - An "author node" has 4-dim feature, a "paper node" has 5-dim feature
- **Case 2**: We know different relation types represent **different types of interactions**
  - (English, translate, French) and (English, translate, Chinese) require different models

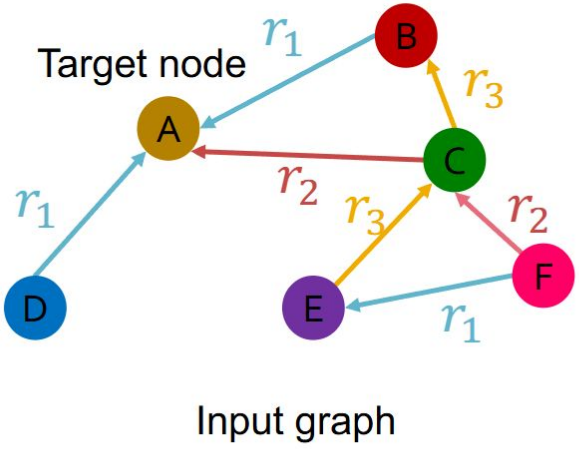
- There are many ways to **convert a heterogeneous graph to a standard graph** (that is, a homogeneous graph)
- **Example:** Add a one-hot indicator for nodes an edges
  - Append feature [1, 0] to each "author node"; Appen feature [0, 1] to each "paper node"
  - Similarly, we can assign edge features to edges with different types

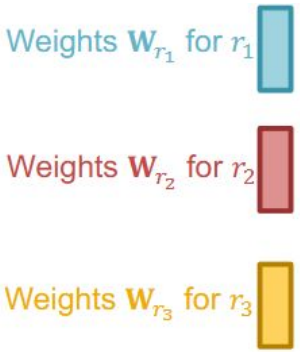
- Ultimately, **heterogeneous graph** is a **more expressive graph representation**
  - Captures **different types of interactions between entities**
- But it also **comes with costs**
  - More expensive (computation, storage)
  - More complex implementation

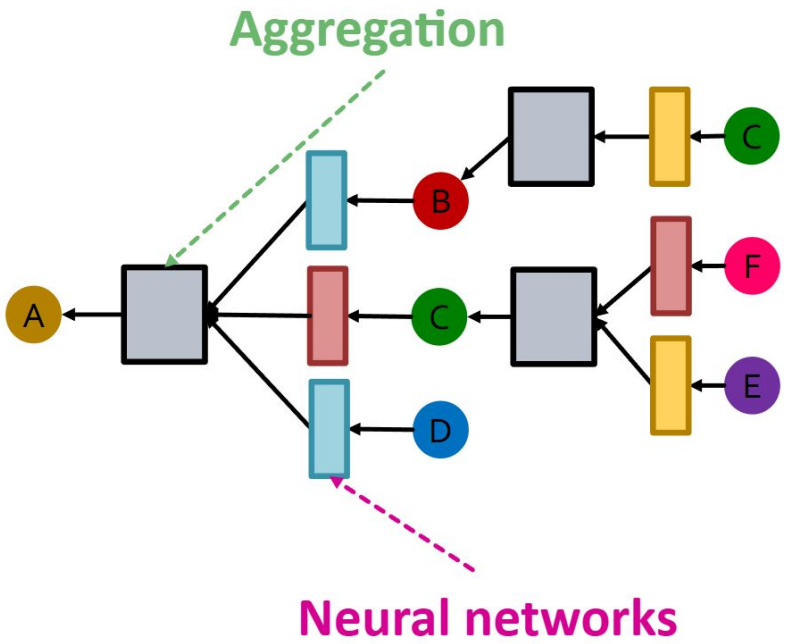
Taken from Stanford CS224W course: http://cs244w.stanford.edu

Relational GCN

- **What if the graph has multiple relation types?**
- Use different neural network weights for different relation types!







Taken from Stanford CS224W course: http://cs244w.stanford.edu

Recap: Classical GNN Layers: GCN

- **(1) Graph Convolutional Networks (GCN)**

$$\mathbf{h}_v^{(l)} = \sigma\left(\sum_{u \in N(v)} \mathbf{W}^{(l)} \frac{\mathbf{h}_u^{(l-1)}}{|N(v)|}\right)$$

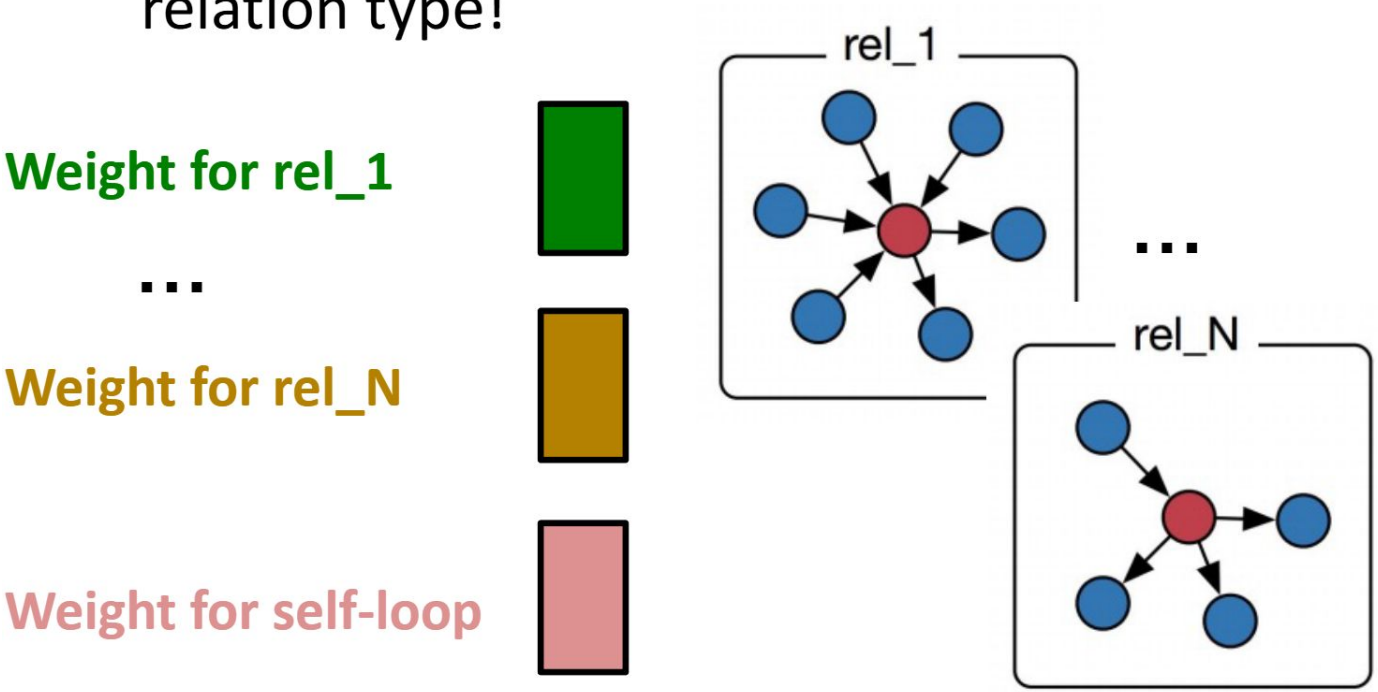
- **We add a self-loop**

$$\mathbf{h}_v^{(l)} = \sigma\left(\sum_{u \in N(v)} \mathbf{W}^{(l)} \frac{\mathbf{h}_u^{(l-1)}}{|N(v)|} + \mathbf{W}^{(l)} \mathbf{h}_v^{(l-1)}\right)$$

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Relational GCN

- Introduce a set of neural networks for each relation type!



Taken from Stanford CS224W course: http://cs244w.stanford.edu

Relational GCN: Definition

- **Relational GCN (RGCN):**

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

- **How to write this as Message + Aggregation?**
- **Message:**
  - Each neighbor of a given relation:

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

  - Self-loop:

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

- **Aggregation:**
  - Sum over messages from neighbors and self-loop, then apply activation
  - $\mathbf{h}_v^{(l+1)} = \sigma\left(\text{Sum}\left(\left\{\mathbf{m}_{u,r}^{(l)}, u \in N(v)\right\} \cup \left\{\mathbf{m}_v^{(l)}\right\}\right)\right)$

Normalized by node degree of the relation $c_{v,r} = |N_v^r|$

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Relational GCN: Scalability

- Each relation has L matrices: $\mathbf{W}_r^{(1)}, \mathbf{W}_r^{(2)} \cdots \mathbf{W}_r^{(L)}$
- The size of each $\mathbf{W}_r^{(l)}$ is $d^{(l+1)} \times d^{(l)}$

$d^{(l)}$ is the hidden dimension in layer l

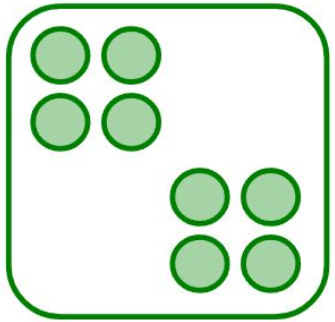
- **Rapid growth of the number of parameters w.r.t number of relations!**
  - **Overfitting becomes an issue**
- **Two methods to regularize the weights $\mathbf{W}_r^{(l)}$**
  - **(1)** Use block diagonal matrices
  - **(2)** Basis/Dictionary learning

Taken from Stanford CS224W course: http://cs244w.stanford.edu

(1) Block Diagonal matrices

- **Key insight**: make the weights **sparse**!
- Use **block diagonal matrices** for $\mathbf{W}_r$

$$\mathbf{W}_r =$$



Limitation: only nearby neurons/dimensions can interact through W

- If use B blocks, then # param reduces from $d^{(l+1)} \times d^{(l)}$ to $B \times \frac{d^{(l+1)}}{B} \times \frac{d^{(l)}}{B}$

Taken from Stanford CS224W course: http://cs244w.stanford.edu

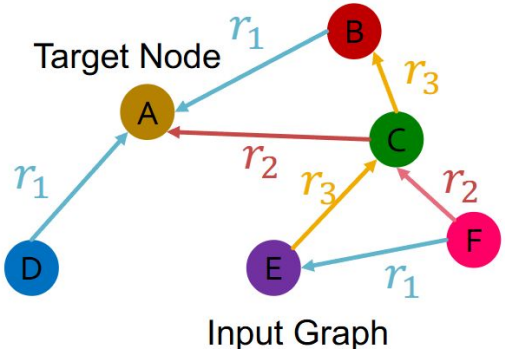
(1) Basis Learning

- **Key insight**: **Share weights** across different relations!
- Represent the matrix of each relation as a linear combination of basis transformations
  $\mathbf{W}_r = \sum_{b=1}^{B} a_{rb} \cdot \mathbf{V}_b$, where $\mathbf{V}_b$ is shared across all relations
  - $\mathbf{V}_b$ are the basis matrices
  - a_{rb} is the importance weight of matrix $\mathbf{V}_b$
- Now each relation only needs to learn $\{a_{rb}\}_{b=1}^{B}$, which is B scalars

Taken from Stanford CS224W course: http://cs244w.stanford.edu

RGCN for Entity/Node Classification

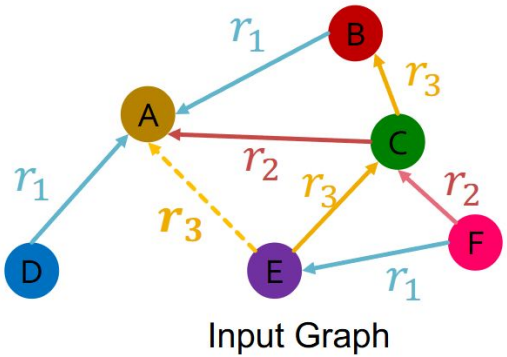
- **Goal: Predict the label of a given node**
- **RGCN** uses the representation of the final layer:
  - If we predict the class of **node A** from **k classes**
  - Take the **final layer (prediction head)**: $\mathbf{h}_A^{(L)} \in \mathbb{R}^k$, each item in $\mathbf{h}_A^{(L)}$ represents **the probability of that class**



Taken from Stanford CS224W course: http://cs244w.stanford.edu

RGCN for Link Prediction

- **Training:**



Input Graph

1. Use RGCN to score the training supervision edge (E, r_3, A)
2. Create a negative edge by perturbing the supervision edge (E, r_3, B)
3. Use GNN model to score negative edge
4. Optimize a standard cross entropy loss (as discussed in Lecture 6)
   1. Maximize the score of training supervision edge
   2. Minimize the score of negative edge

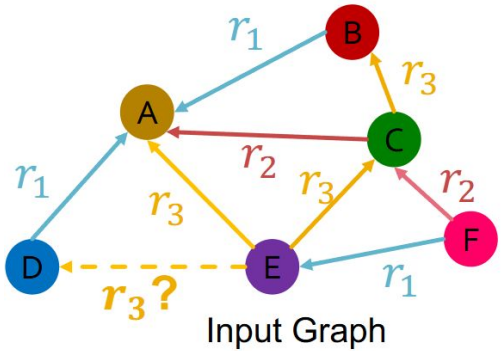
$$\ell = -\log \sigma\left(f_{r_3}(h_E, h_A)\right) - \log(1 - \sigma(f_{r_3}(h_E, h_B)))$$

σ ... Sigmoid function

Taken from Stanford CS224W course: http://cs244w.stanford.edu

RGCN for Link Prediction

- **Evaluation:**
  - **Validation time as an example, same at the test time**



Input Graph

Evaluate how the model can predict the validation edges with the relation types.
Let's predict validation edge (E, r_3, D)
Intuition: the score of (E, r_3, D) should be higher than all (E, r_3, v) where (E, r_3, v) is NOT in the training message edges and training supervision edges, e.g., (E, r_3, B)

1. Calculate the score of (E, r_3, D)
2. Calculate the score of all the negative edges: $\{(E, r_3, v) | v \in \{B, F\}\}$, since (E, r_3, A), (E, r_3, C) belong to training message edges & training supervision edges
3. Obtain the ranking RK of (E, r_3, D).
4. Calculate metrics
   1. Hits@k: $\mathbf{1}[RK \leq k]$. Higher is better
   2. Reciprocal Rank: $\frac{1}{RK}$. Higher is better

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Benchmark for Heterogeneous Graphs

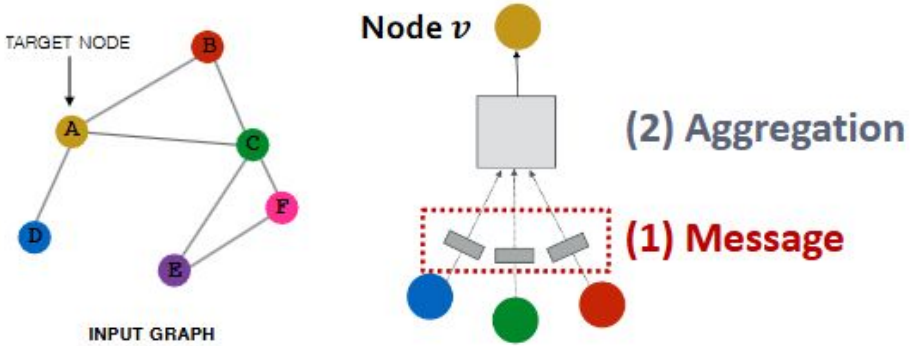
- Benchmark dataset
  - ogbn-mag from Microsoft Academic Graph (MAG)
- Four (4) **types of entities**
  - Papers: 736k nodes
  - Authors: 1.1m nodes
  - Institutions: 9k nodes
  - Fields of study: 60k nodes
- Four (4) **directed relations**
  - An author is "affiliated with" an institution
  - An author "writes" a paper
  - A paper "cites" a paper
  - A paper "has a topic of" a field of study
- **Prediction task**
  - Each paper has a 128-dimensional **word2vec** feature vector
  - Given the content, references, authors, and author affiliations from ogbn-mag, predict the **venue of each paper**
  - **349-class** classification problem due to 349 venues considered
- **Time-based dataset splitting**
  - **Training set**: papers published before 2018
  - **Test set**: papers published after 2018

Leaderboard

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Heterogeneous Message

- (1) Message computation
  - **Message function:** $\mathbf{m}_u^{(l)} = \text{MSG}^{(l)}\left(\mathbf{h}_u^{(l-1)}\right)$
    - **Intuition:** Each node will create a message, which will be sent to other nodes later
    - **Example:** A Linear layer $\mathbf{m}_u^{(l)} = \mathbf{W}^{(l)}\mathbf{h}_u^{(l-1)}$



(1) Heterogeneous message computation

- **Message function:** $\mathbf{m}_u^{(l)} = \text{MSG}_r^{(l)}\left(\mathbf{h}_u^{(l-1)}\right)$
  - **Observation:** A node could receive multiple types of messages. **Num of message type = Num of relation type**
  - **Idea:** Create a different message function for each relation type
    - $\mathbf{m}_u^{(l)} = \text{MSG}_r^{(l)}\left(\mathbf{h}_u^{(l-1)}\right)$, $r = (u, e, v)$ is the relation type between node u that sends the message, edge type e , and node v that receive the message
  - **Example:** A Linear layer $\mathbf{m}_u^{(l)} = \mathbf{W}_r^{(l)}\mathbf{h}_u^{(l-1)}$

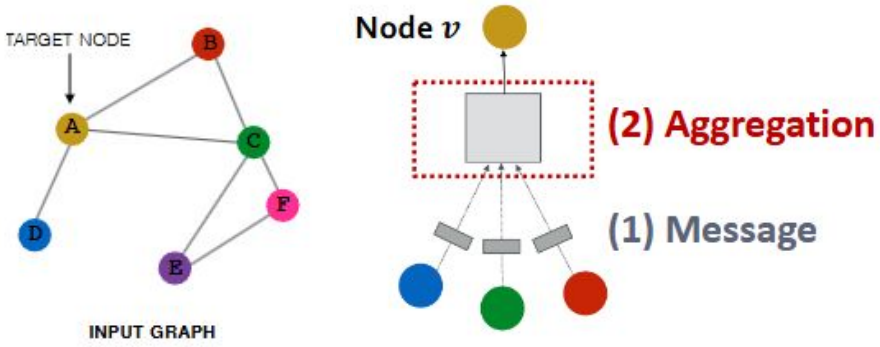
Taken from Stanford CS224W course: http://cs244w.stanford.edu

Heterogeneous Aggregation

- **(2) Aggregation**
  - **Intuition:** Each node will aggregate the messages from node v's neighbors

$$\mathbf{h}_v^{(l)} = \text{AGG}^{(l)}\left(\left\{\mathbf{m}_u^{(l)}, u \in N(v)\right\}\right)$$

  - **Example:** $\text{Sum}(\cdot)$, $\text{Mean}(\cdot)$ or $\text{Max}(\cdot)$ aggregator
    - $\mathbf{h}_v^{(l)} = \text{Sum}(\{\mathbf{m}_u^{(l)}, u \in N(v)\})$

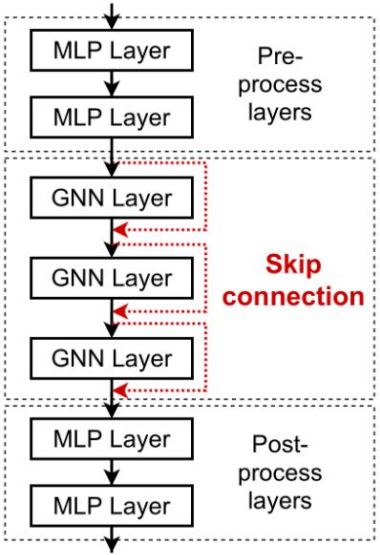


- **(2) Heterogeneous Aggregation**
  - **Observation:** Each node could receive multiple types of messages from its neighbors, and multiple neighbors may belong to each message type.
  - **Idea:** We can define a 2-stage message passing
    - $\mathbf{h}_v^{(l)} = \text{AGG}_{all}^{(l)}\left(\text{AGG}_r^{(l)}\left(\left\{\mathbf{m}_u^{(l)}, u \in N_r(v)\right\}\right)\right)$
    - Given all the messages sent to a node
    - Within each message type, aggregate the messages that belongs to the relation type with $\text{AGG}_r^{(l)}$
    - Aggregate across the edge types with $\text{AGG}_{all}^{(l)}$
  - **Example:** $\mathbf{h}_v^{(l)} = \text{Concat}\left(\text{Sum}\left(\left\{\mathbf{m}_u^{(l)}, u \in N_r(v)\right\}\right)\right)$

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Recap: Layer Connectivity

- **(3) Layer connectivity**
  - Add skip connections, pre/post-process layers



Pre-processing layers: Important when encoding node features is necessary.
E.g., when nodes represent images/text

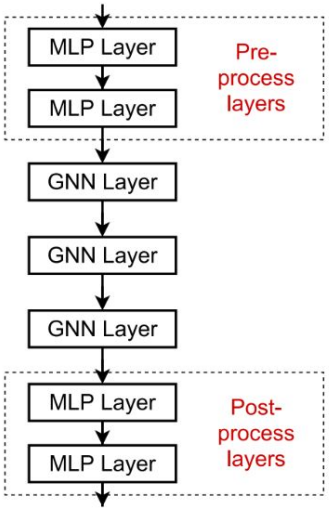
Post-processing layers: Important when reasoning / transformation over node embeddings are needed
E.g., graph classification, knowledge graphs

In practice, adding these layers works great!

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Heterogeneous GNN Layers

- **Heterogeneous** pre/post-process layers:
  - MLP layers **with respect to each node type**
    - Since the output of GNN are **node embeddings**
  - $\mathbf{h}_v^{(l)} = \mathrm{MLP}_{T(v)}(\mathbf{h}_v^{(l)})$
    - $T(v)$ is the type of node v
- Other successful GNN designs are also encouraged for heterogeneous GNNs: skip connections, batch/layer normalization, …



Taken from Stanford CS224W course: http://cs244w.stanford.edu

Heterogeneous Graph Manipulation

- **Graph Feature manipulation**
  - **2 Common options:** compute graph statistics (e.g., node degree) within each relation type, or across the full graph (ignoring the relation types)
- **Graph Structure manipulation**
  - Neighbor and subgraph sampling are also common for heterogeneous graphs.
  - **2 Common options:** sampling within each relation type (ensure neighbors from each type are covered), or sample across the full graph

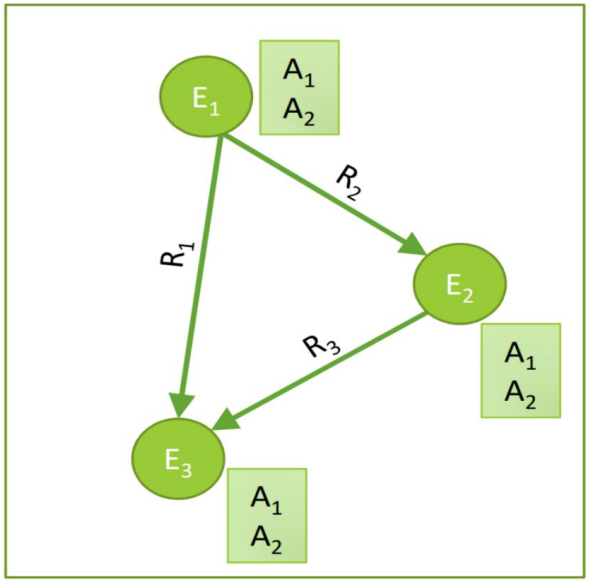
Taken from Stanford CS224W course: http://cs244w.stanford.edu

Knowledge Graphs

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**

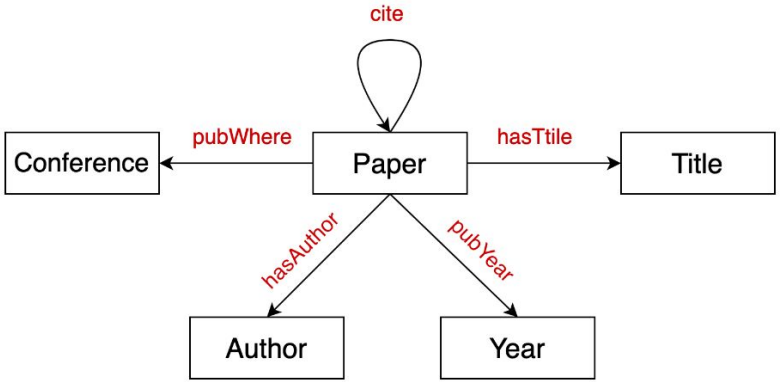


Taken from Stanford CS224W course: http://cs244w.stanford.edu

Examples

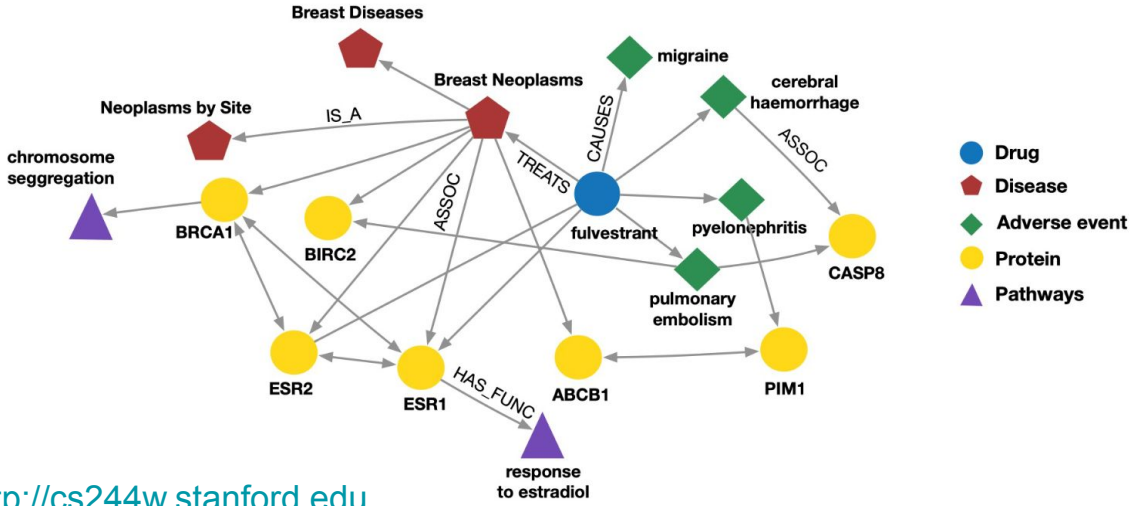
Example 1: Bibliographic Networks

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



Example 2: Bio Knowledge Graphs

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

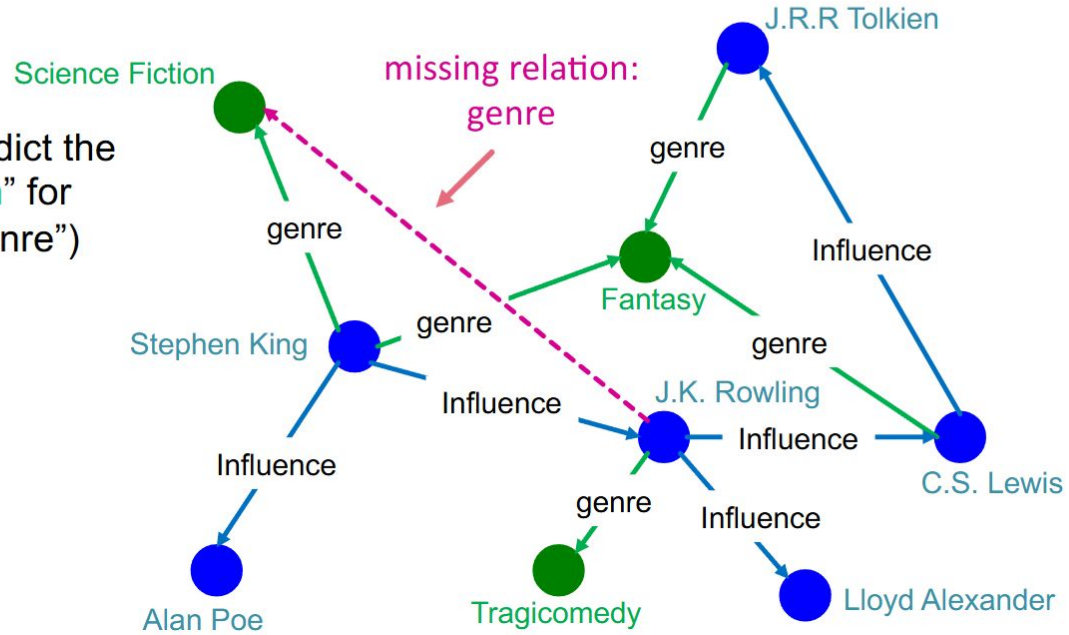


Taken from Stanford CS224W course: http://cs244w.stanford.edu

KG Completion Task

- 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")



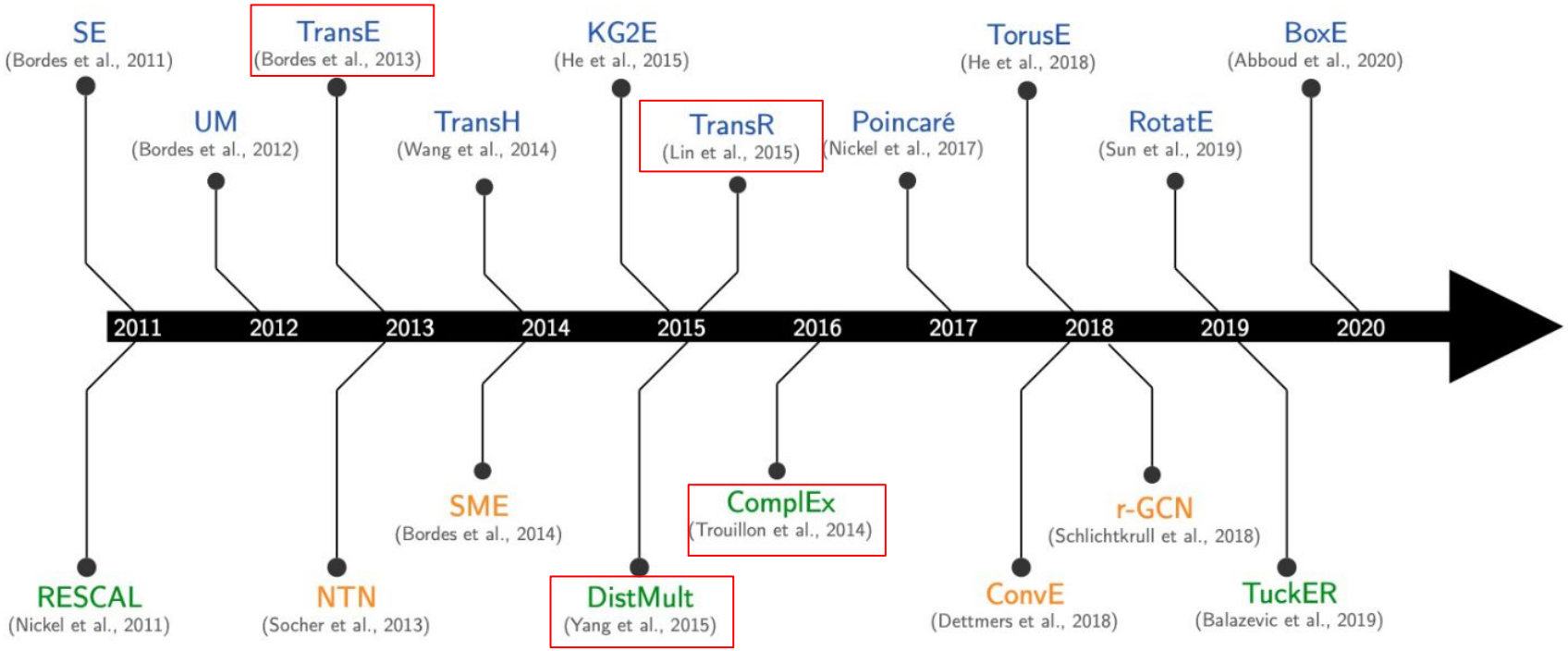
Taken from Stanford CS224W course: http://cs244w.stanford.edu

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

Taken from Stanford CS224W course: http://cs244w.stanford.edu

KG Embedding Models



Taken from Stanford CS224W course: http://cs244w.stanford.edu

TransE

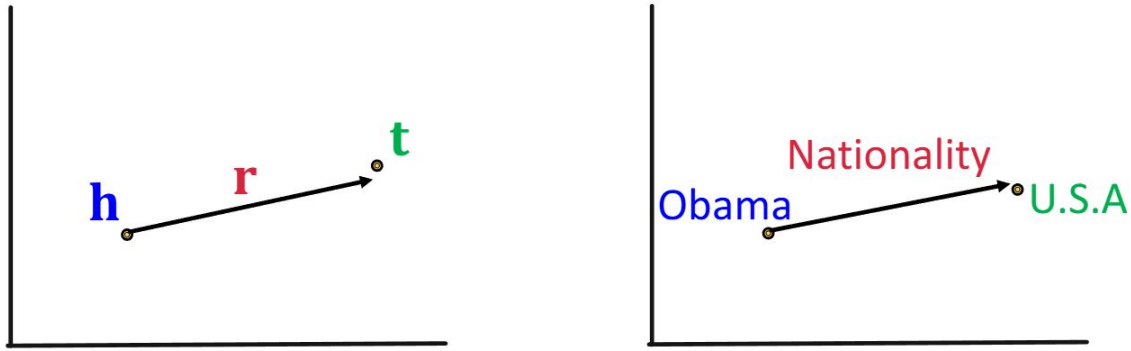
- **Intuition: Translation**

  For a triple (h, r, t), let $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$ be embedding vectors.

  embedding vectors will appear in boldface

- **TransE:** $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ if the given link exists else $\mathbf{h} + \mathbf{r} \neq \mathbf{t}$

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



Taken from Stanford CS224W course: http://cs244w.stanford.edu

TransE

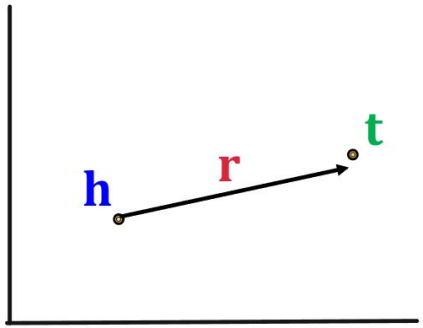
- **Intuition: Translation**

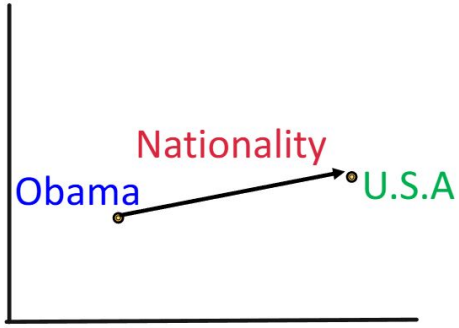
  For a triple (h, r, t), let $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$ be embedding vectors.

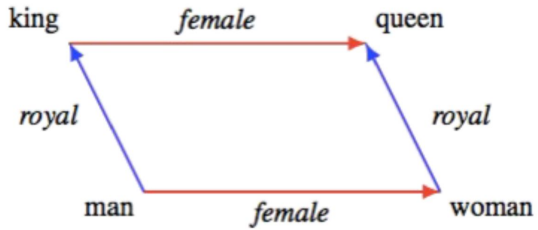
  embedding vectors will appear in boldface

- **TransE:** $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ if the given link exists else $\mathbf{h} + \mathbf{r} \neq \mathbf{t}$

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







Taken from Stanford CS224W course: http://cs244w.stanford.edu

TransE

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 $e \in E$

  $S_{batch} \leftarrow$ 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**

    $(h',r,t') \leftarrow$ 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.

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

(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

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Four Relation Patterns

- **Symmetric (Antisymmetric) Relations:**

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

  - **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:**

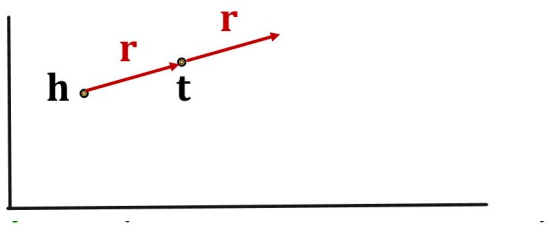
$$r(h,t_1), r(h,t_2), \ldots, r(h,t_n) \text{ are all True.}$$

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

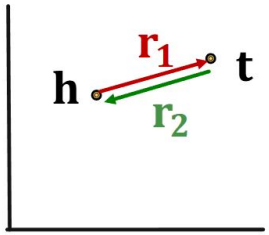
Taken from Stanford CS224W course: http://cs244w.stanford.edu

TransE

- **TransE** can model antisymmetric relations ✓
  - $\mathbf{h} + \mathbf{r} = \mathbf{t}$, but $\mathbf{t} + \mathbf{r} \neq \mathbf{h}$

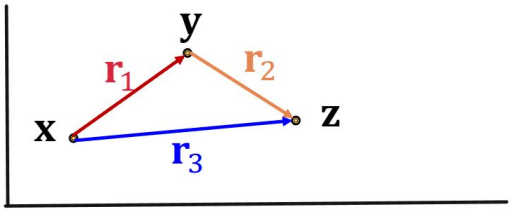


- **TransE** can model inverse relations ✓
  - $\mathbf{h} + \mathbf{r}_2 = \mathbf{t}$, we can set $\mathbf{r}_1 = -\mathbf{r}_2$



- **TransE** can model composition relations ✓

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

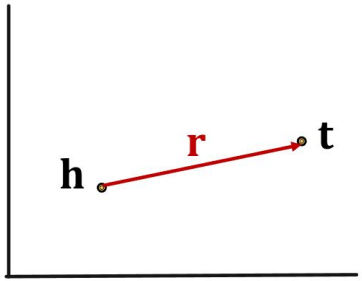


Taken from Stanford CS224W course: http://cs244w.stanford.edu

TransE

- **TransE cannot** model symmetric relations ✗

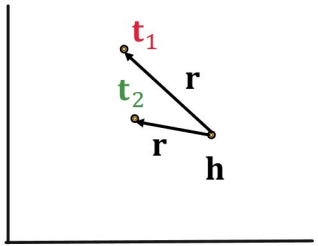
  only if $\mathbf{r} = 0,\ \mathbf{h} = \mathbf{t}$



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.

- **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!



Taken from Stanford CS224W course: http://cs244w.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.

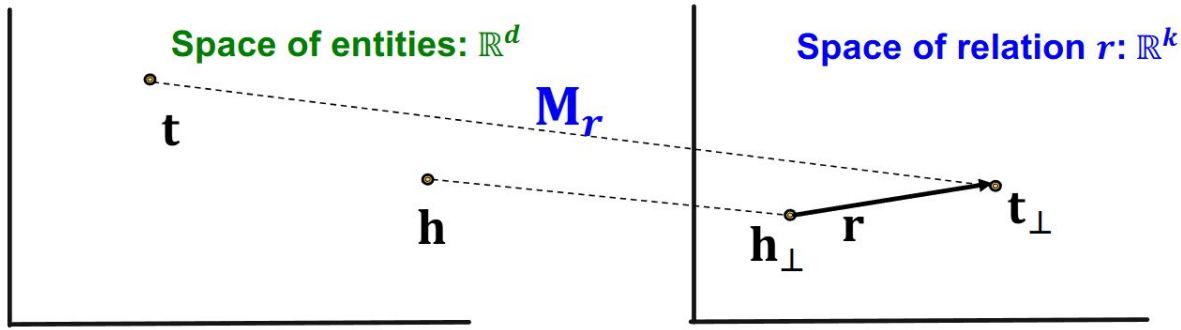
Taken from Stanford CS224W course: http://cs244w.stanford.edu

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**.

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

- $\mathbf{h}_\perp = \mathbf{M}_r\mathbf{h},\ \mathbf{t}_\perp = \mathbf{M}_r\mathbf{t}$
- **Score function**: $f_r(h,t) = -||\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp||$



Taken from Stanford CS224W course: http://cs244w.stanford.edu

DistMult

- **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**

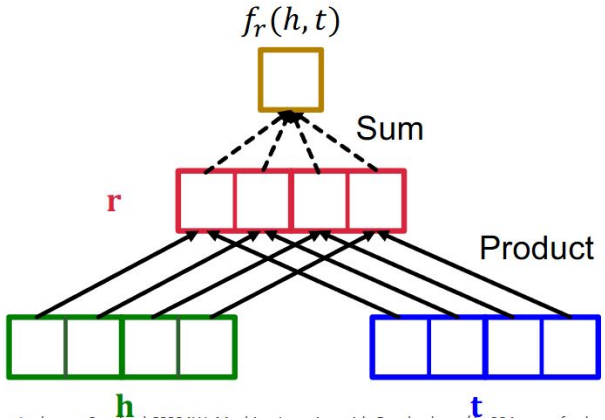
Taken from Stanford CS224W course: http://cs244w.stanford.edu

DistMult

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

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

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



Taken from Stanford CS224W course: http://cs244w.stanford.edu

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 $\mathbf{t}$

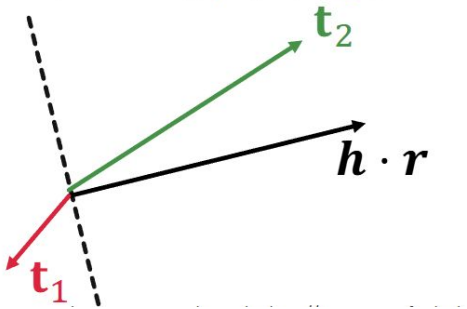
  where $\mathbf{h} \cdot \mathbf{r}$ is defined as $[\boldsymbol{h} \cdot \boldsymbol{r}]_i = \boldsymbol{h}_i \cdot \boldsymbol{r}_i$
- **Example:**

Hadamard product

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

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

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

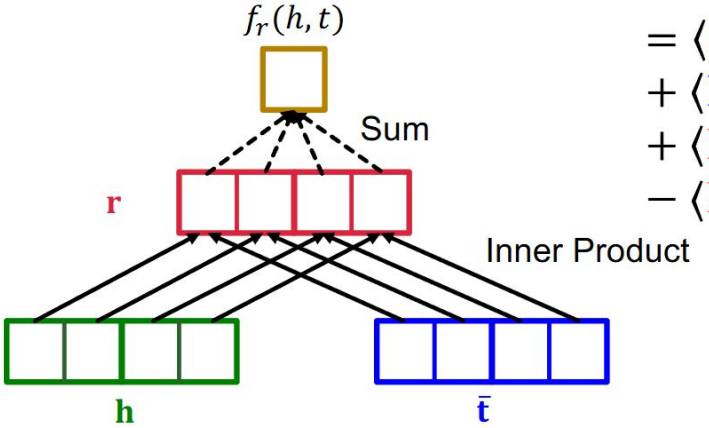


Taken from Stanford CS224W course: http://cs244w.stanford.edu

ComplEx

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

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



Taken from Stanford CS224W course: http://cs244w.stanford.edu

ComplEx Score Function

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

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

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

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

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

Taken from Stanford CS224W course: http://cs244w.stanford.edu

Relation Patterns Comparison

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	$-\|\|\mathbf{h}\circ\mathbf{r}-t\|\|$	$\mathbf{h},\mathbf{t},\mathbf{r}\in\mathbb{C}^k$	✓	✓	✓	✓	✗

Taken from Stanford CS224W course: http://cs244w.stanford.edu