

CS247: ADVANCED DATA MINING


09: Graph and Network: Graph Embedding

Instructor: Yizhou Sun

yzsun@ccs.neu.edu

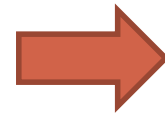
February 25, 2024

Graph Embedding

- What is Graph Embedding 
- Shallow Network Embedding
- Knowledge Graph Embedding
- Graph Neural Network
- Summary

How to represent nodes?

- A naïve solution

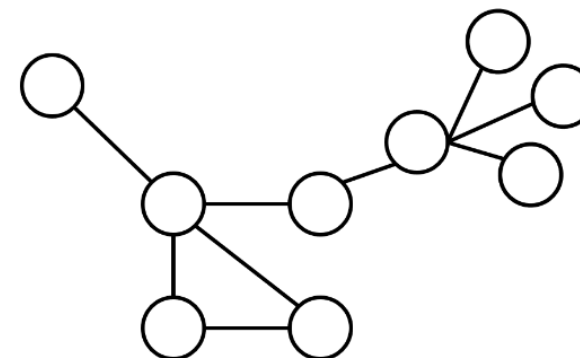
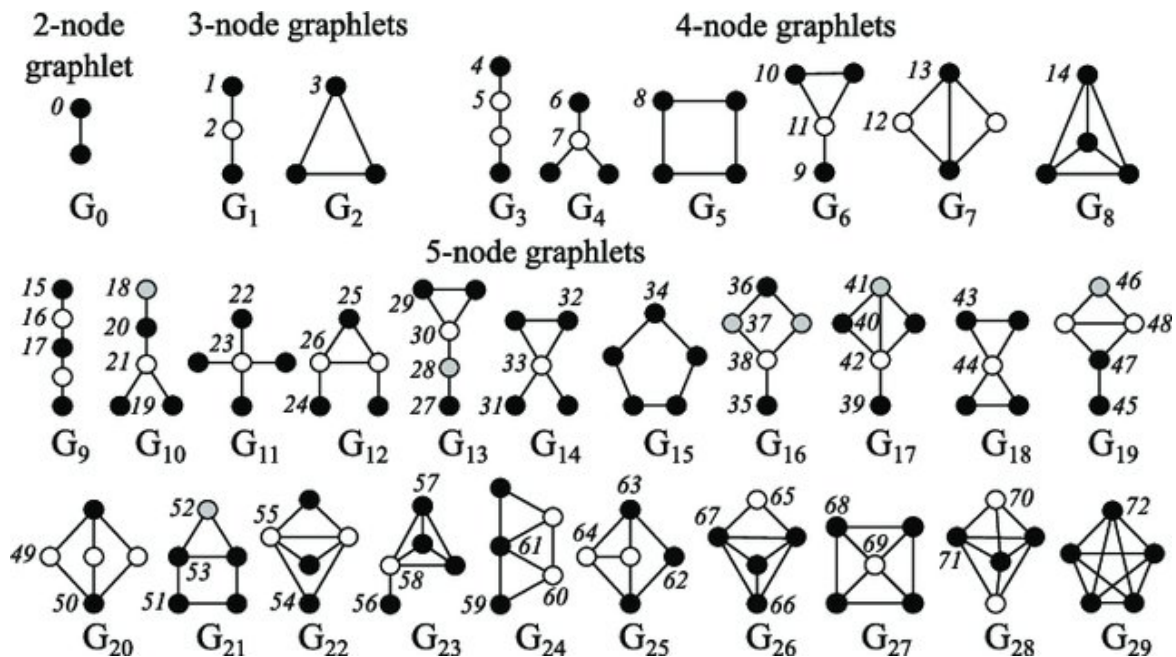


	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	0	1
D	1	0	0	0	0	0
E	0	1	0	0	0	0
F	0	1	1	0	0	0

- **Limitations:**
 - Extremely High-dimensional
 - No global structure information integrated
 - Permutation-variant

Even more challenging for graph representation

- Ex. Graphlet-based feature vector



Source: DOI: [10.1093/bioinformatics/btv130](https://doi.org/10.1093/bioinformatics/btv130)

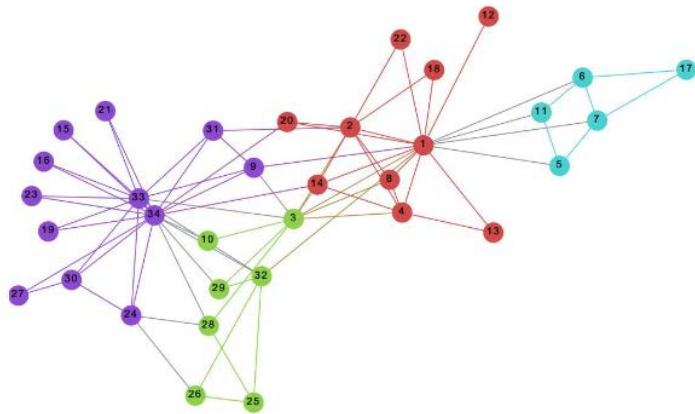
Requires subgraph isomorphism test: NP-hard

Source: https://haotang1995.github.io/projects/robust_graph_level_representation_learning_using_graph_based_structural_attentional_learning

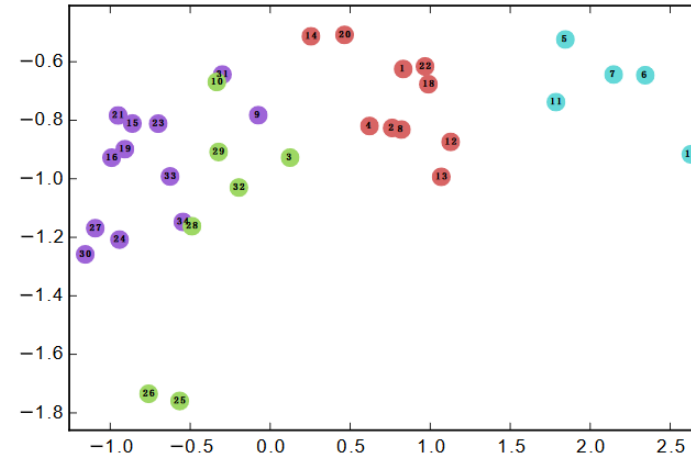
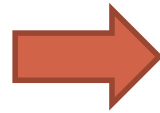
						...
12	1	4	1	6	0	...

A Better Solution

- Map each node into a low dimensional vector
 - $\phi: V \rightarrow R^d$




(a) Input: karate network



(b) Output: representations

Source: DeepWalk

Graph Embedding

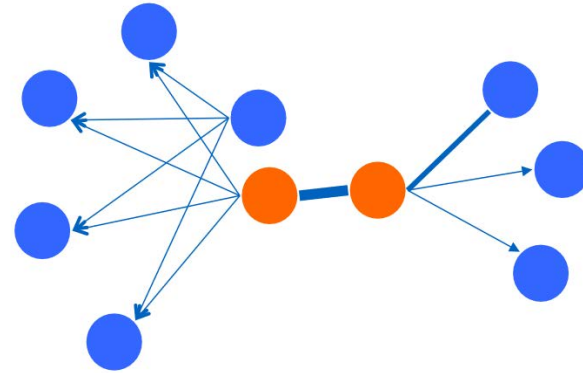
- What is Graph Embedding
- Shallow Network Embedding 
- Knowledge Graph Embedding
- Graph Neural Network
- Summary

Shallow Network Embedding Approaches

- Inspired by word embedding
 - A node's embedding is determined by its context
- How to define the local context of a node?
 - DeepWalk [Perozzi, KDD'14]
 - LINE [Tang, WWW'15]
 - Node2Vec [Grover, KDD'16]

LINE: Large-scale Information Network Embedding

- First-order proximity



- Assumption: Two nodes are similar if they are connected

$$p_1(v_i, v_j) = \frac{\exp(\vec{u}_i^T \vec{u}_j)}{\sum_{(m,n) \in E \times V} \exp(\vec{u}_m^T \vec{u}_n)} \quad u_i: \text{embedding vector for node } i$$

- Limitation: links are sparse, not sufficient

Objective function for first-order proximity

- Minimize the KL divergence between empirical link distribution and modeled link distribution

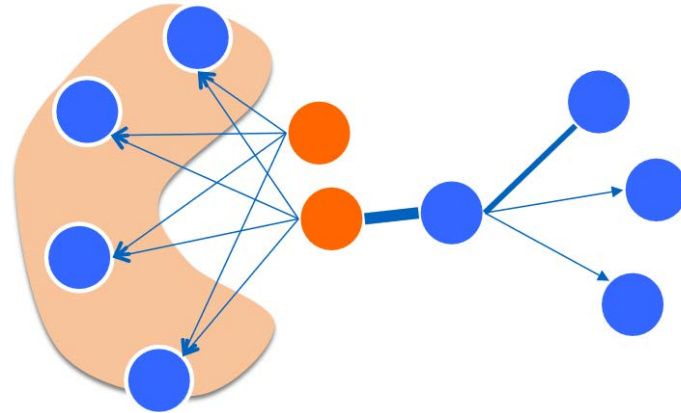
$$\hat{p}_1(v_i, v_j) = \frac{w_{ij}}{\sum_{(m,n) \in E} w_{mn}}$$

$$O_1 = KL(\hat{p}_1, p_1) = - \sum_{(i,j) \in E} w_{ij} \log p_1(v_i, v_j)$$

w_{ij}: weight over edge(i,j)

Second-Order Proximity

- Assumption:
 - Two nodes are similar if their neighbors are similar



$$p_2(v_j | v_i) = \frac{\exp(\vec{u}'_j{}^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}'_k{}^T \cdot \vec{u}_i)}$$

u_i : target embedding vector for node i
 u'_j : context embedding vector for node j

Objective function for second-order proximity

- Minimize the KL divergence between empirical link distribution and modeled link distribution

- Empirical distribution

$$\hat{p}_2(v_j | v_i) = \frac{w_{ij}}{\sum_{k \in V} w_{ik}}$$

- Objective function

$$O_2 = \sum_i d_i KL(\hat{p}_2(\cdot | v_i), p_2(\cdot | v_i)) = - \sum_{(i,j) \in E} w_{ij} \log p_2(v_j | v_i)$$
$$d_i = \sum_k w_{ik}$$

Negative Sampling for Optimization

- For second-order proximity derived objective function
 - For each positive link (i, j), sample K negative links (i, n)
 - An edge with weight w can be considered as w binary edges
 - New objective function

$$\log \sigma(\vec{u}'_j \cdot \vec{u}_i) + \sum_{i=1}^K E_{v_n \sim P_n(v)} [\log \sigma(-\vec{u}'_n \cdot \vec{u}_i)]$$

negative distribution: $P_n(v) \propto d_v^{3/4}$

Announcement

- Midterm grading is out
- Homework grading is out
- Course project sign-up is out
 - Presentation score will be given by audiences
 - Participation score will be given to audiences who rate the presenting groups

Midterm Statistics


- Mean: 78.8
Median: 85
Max: 98
- 75.6% students receive scores higher than 75

slido

Midterm Feedback

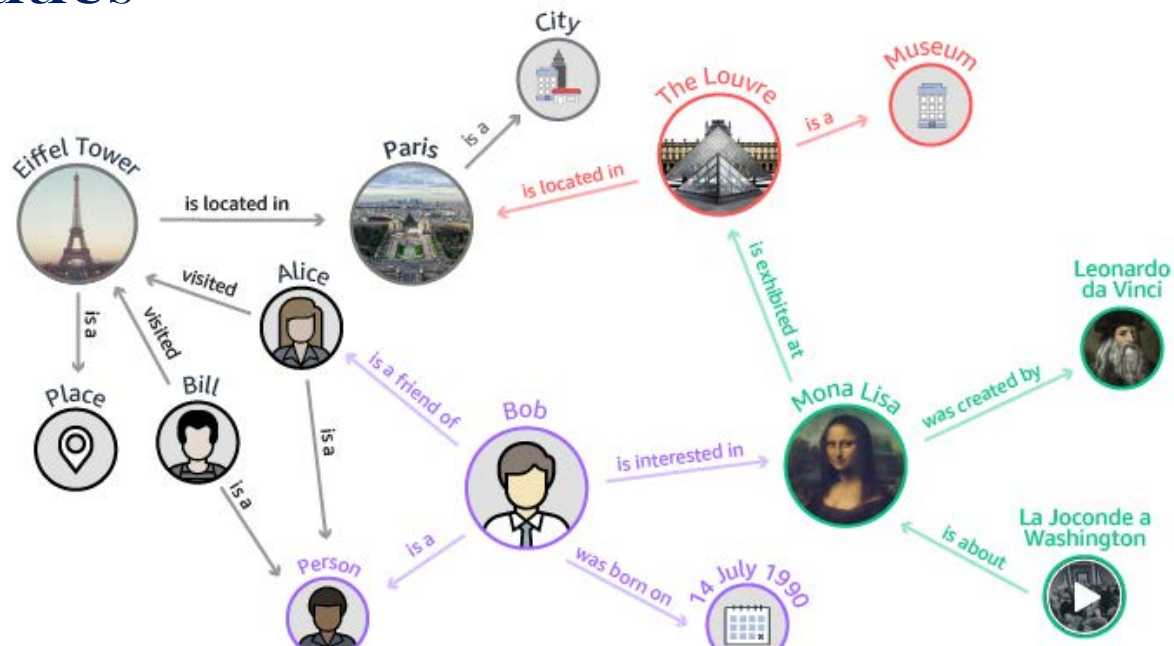
ⓘ Start presenting to display the poll results on this slide.

Graph Embedding

- What is Graph Embedding
- Shallow Network Embedding
- Knowledge Graph Embedding 
- Graph Neural Networks
- Summary

Knowledge Graph

- What are knowledge graphs?
 - Multi-relational graph data
 - (heterogeneous information network)
 - Provide structured representation for semantic relationships between real-world entities



A triple (h, r, t) represents a fact, ex:
(Eiffel Tower, is located in, Paris)

KGs are everywhere

General-purpose KGs



Bio & Medical KGs



Product Graphs & E-commerce



Common-sense KGs & NLP



Application in Search Engine

- When you search in Google

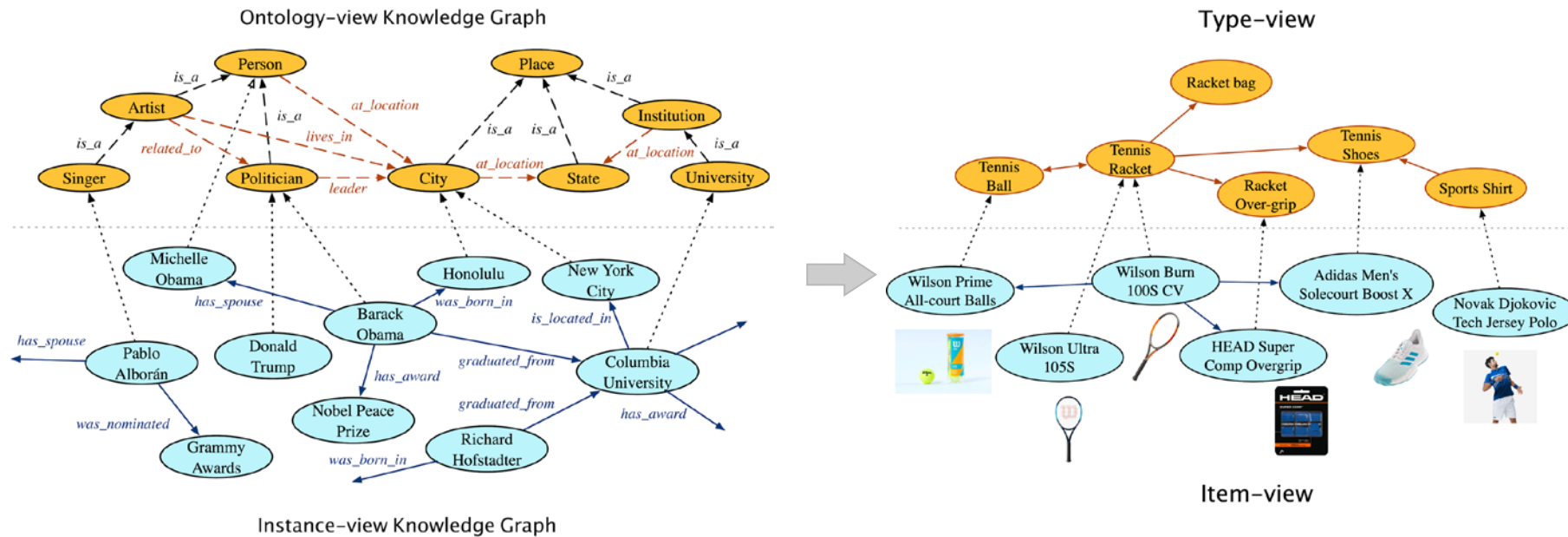
The image shows a Google search for "mike bloomberg". The search bar is highlighted with a red box. Below the search bar, there are navigation tabs for All, News, Images, Videos, Books, and More. The search results show "About 73,600,000 results (0.89 seconds)". The top result is "Mike Bloomberg 2020 | Fighting for our future" with a link to www.mikebloomberg.com/. Below this, there are sections for "About Mike" and "Get Involved". The "Top stories" section features three articles: "Trump Bars Bloomberg News Journalists From Campaign Events" (The New York Times, 6 hours ago), "Trump attacks 'Mini Mike Bloomberg' after campaign bars news outlet | TheHill" (TheHill, 1 hour ago), and "Trump attacks Bloomberg News after his campaign says it will deny press..." (Washington Post, 51 mins ago). Below the top stories is a link to "More for mike bloomberg".

On the right side, there is a knowledge panel for "Michael Bloomberg", CEO of Bloomberg L.P. The panel includes a profile picture, a description: "Michael Rubens Bloomberg is an American politician, businessman, and author. He is the co-founder, CEO, and majority owner of Bloomberg L.P. He was mayor of New York City from 2002 to 2013. On November 24, 2019 he announced his candidacy for the 2020 United States presidential election. Wikipedia", and various facts: "Party: Democratic Party Trending", "Born: February 14, 1942 (age 77 years), Brighton, MA", "Height: 5' 8\"", "Net worth: 54.6 billion USD (2019)", "Partner: Diana Taylor (2000-)", and "Children: Georgina Bloomberg, Emma Bloomberg". Below the facts are social media profiles for Twitter, Facebook, Instagram, and YouTube. At the bottom of the panel is a "People also search for" section with images and names of related people: Diana Taylor (Partner), Andrew Cuomo (Trending), Georgina Bloomberg (Daughter), Larry Ellison, and Larry Page.

A blue arrow points from the text "Facts from KG" to the knowledge panel, indicating that the information is derived from a Knowledge Graph.

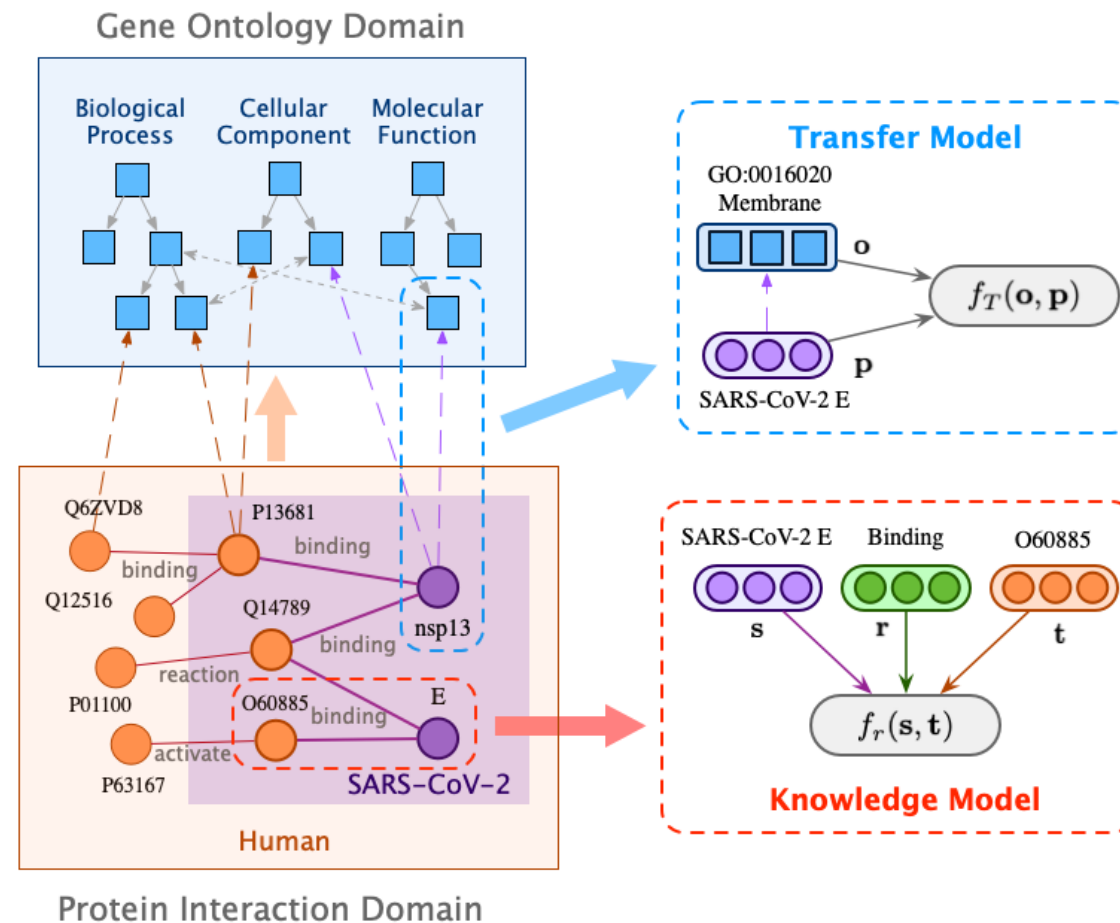
Applications to Product KG

- Hao et al., “P-Companion: A Principled Framework for Diversified Complementary Product Recommendation”, CIKM’20



Application to Biological KG

- Hao et al., “[Bio-JOIE: Joint Representation Learning of Biological Knowledge Bases](#)”, ACM BCB’20 (**best student paper award**)

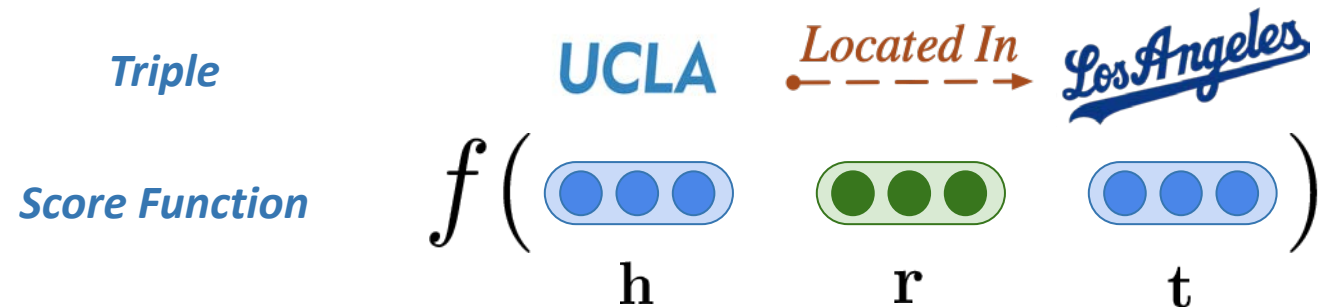


Knowledge Graph Embedding

- Goal: Encode entities as low-dimensional vectors and relations as parametric algebraic operations
 - Input: Relation facts (triples)
 - Output: representations of objects and relations

Key Idea of KG embedding algorithms

- Define a score function for a triple: $f_r(\mathbf{h}, \mathbf{t})$
 - According to entity and relation representation



- Define a loss function to guide the training
 - E.g., an observed triple scores higher than a negative one

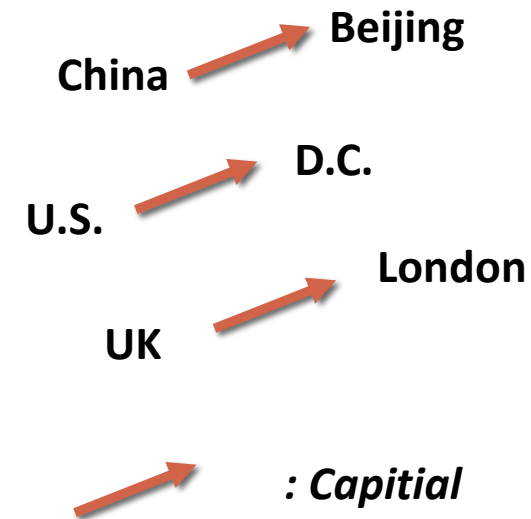
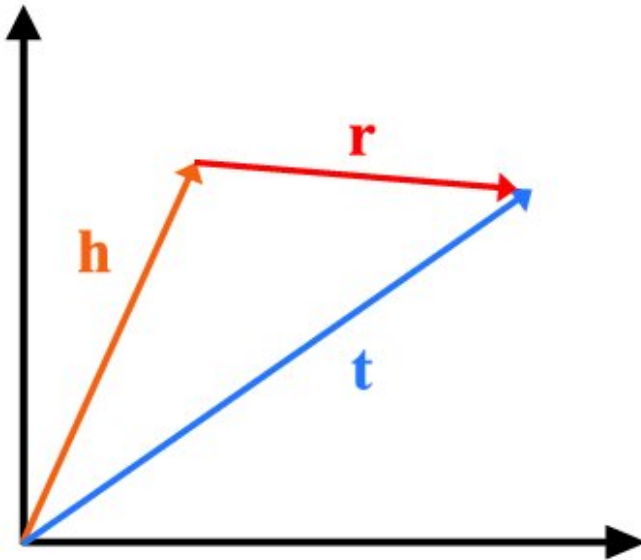
Summary of Existing Approaches

Model	Score Function	
SE (Bordes et al., 2011)	$-\ W_{r,1}\mathbf{h} - W_{r,2}\mathbf{t}\ $	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^k, W_{r,\cdot} \in \mathbb{R}^{k \times k}$
TransE (Bordes et al., 2013)	$-\ \mathbf{h} + \mathbf{r} - \mathbf{t}\ $	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$
TransX	$-\ g_{r,1}(\mathbf{h}) + \mathbf{r} - g_{r,2}(\mathbf{t})\ $	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$
DistMult (Yang et al., 2014)	$\langle \mathbf{r}, \mathbf{h}, \mathbf{t} \rangle$	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$
Complex (Trouillon et al., 2016)	$\text{Re}(\langle \mathbf{r}, \mathbf{h}, \bar{\mathbf{t}} \rangle)$	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{C}^k$
HolE (Nickel et al., 2016)	$\langle \mathbf{r}, \mathbf{h} \otimes \mathbf{t} \rangle$	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$
ConvE (Dettmers et al., 2017)	$\langle \sigma(\text{vec}(\sigma([\bar{\mathbf{r}}, \bar{\mathbf{h}}] * \Omega))\mathbf{W}), \mathbf{t} \rangle$	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$
RotatE	$-\ \mathbf{h} \circ \mathbf{r} - \mathbf{t}\ ^2$	$\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{C}^k, r_i = 1$

Source: Sun et al., RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space (ICLR'19)

TransE: Score Function

- Relation: translating embedding



- Score function

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

Bordes et al., Translating embeddings for modeling multi-relational data, NeurIPS 2013

TransE: Objective Function

- Objective Function
 - Margin-based ranking loss
 - $L = \sum_{(h,r,t) \in S} \sum_{(h',r,t') \in S'_{(h,r,t)}} [\gamma + d(\mathbf{h} + \mathbf{r}, \mathbf{t}) - d(\mathbf{h}' + \mathbf{r}, \mathbf{t}')]_+$
 - $[x]_+$ denotes the positive part of x , i.e., $\max(0, x)$
 - $\gamma > 0$ denotes the margin hyperparameter
 - The higher the bigger difference between positive triple and negative one
 - S : positive triple set; S' : corrupted triple set (negative triples)
- Optimization: stochastic gradient descent

TransE: Limitations

- One-one mapping: $t = \phi_r(h)$
 - Given (h,r), t is unique
 - Given (r,t), h is unique
- Anti-symmetric
 - If $r(h,t)$ then $r(t,h)$ is not true
 - Cannot model symmetric relation, e.g., friendship
- Anti-reflexive
 - $r(h,h)$ is not true
 - Cannot model reflexive relations, e.g., synonym

DistMult

- Bilinear score function

- $f_r(\mathbf{h}, \mathbf{t}) = \mathbf{h}^T \mathbf{M}_r \mathbf{t}$

- Where \mathbf{M}_r is a diagonal matrix with diagonal vector \mathbf{r}

- A simplification to neural tensor network (NTN)

- Objective function

- $L = \sum_{(h,r,t) \in \mathcal{S}} \sum_{(h',r,t') \in \mathcal{S}'_{(h,r,t)}} [\gamma - f_r(\mathbf{h}, \mathbf{t}) + f_r(\mathbf{h}', \mathbf{t}')]_+$

- Limitation

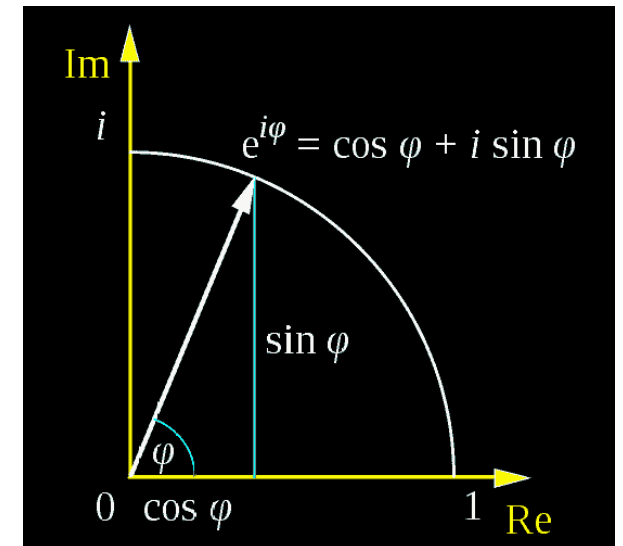
- Can only model symmetric relation

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

Yang et al., Embedding entities and relations for learning and inference in knowledge bases, ICLR 2015

RotatE: Score Function

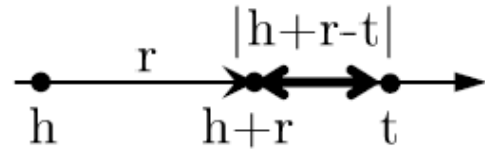
- Relation: rotation operation in complex space
 - head and tail entities in complex vector space, i.e., $\mathbf{h}, \mathbf{t} \in \mathbb{C}^k$
 - each relation r as an element-wise rotation from the head entity \mathbf{h} to the tail entity \mathbf{t} , i.e.,
 - $\mathbf{t} = \mathbf{h} \circ \mathbf{r}$, i.e., $t_i = h_i r_i$, where $|r_i| = 1$, for $i = 1, \dots, k$
 - Equivalently, $r_i = e^{i\theta_{r,i}}$, i.e., rotate h_i with $\theta_{r,i}$
- Score function:
 - $f_r(h, t) = -\|\mathbf{h} \circ \mathbf{r} - \mathbf{t}\|$



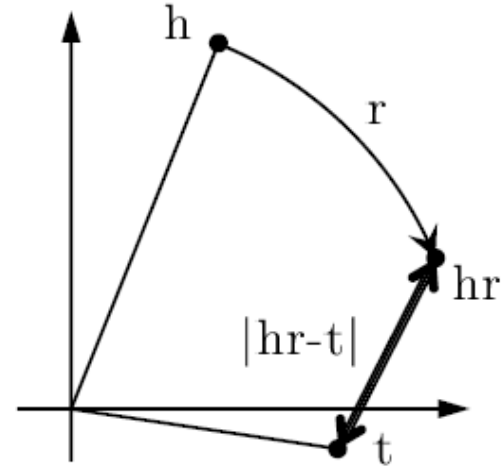
Zhiqing Sun, Zhihong Deng, Jian-Yun Nie, and Jian Tang. “RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space.” ICLR’19.

RotatE: Geometric Interpretation

- Consider 1-d case



(a) TransE models r as translation in real line.



(b) RotatE models r as rotation in complex plane.

RotatE: Objective function

- Smarter negative sampling

- The negative triple with higher score is more likely to be sampled

$$p(h'_j, r, t'_j | \{(h_i, r_i, t_i)\}) = \frac{\exp \alpha f_r(\mathbf{h}'_j, \mathbf{t}'_j)}{\sum_i \exp \alpha f_r(\mathbf{h}'_i, \mathbf{t}'_i)}$$

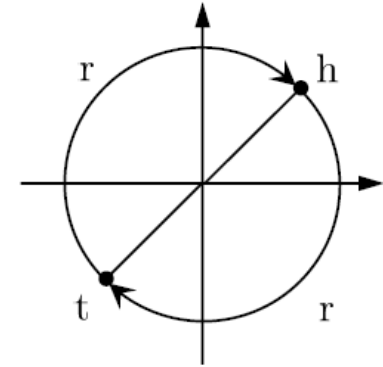
- Cross-entropy loss

$$L = -\log \sigma(\gamma - d_r(\mathbf{h}, \mathbf{t})) - \sum_{i=1}^n p(h'_i, r, t'_i) \log \sigma(d_r(\mathbf{h}'_i, \mathbf{t}'_i) - \gamma)$$

RotatE: Pros and Cons

- Pros:


- Can model relations with different properties
- Symmetric: $r_i = +1$ or -1
- Anti-symmetric: $r \circ r \neq \mathbf{1}$
- Inverse relations: $r_2 = \bar{r}_1$
 - E.g., hypernym is the **inverse** relation of hyponym
- Composition relations: $r_3 = r_1 \circ r_2$, i. e., $\theta_3 = \theta_1 + \theta_2$, if $r_j = e^{i\theta_j}$ for $j = 1, 2, 3$



- Cons:

- One-one mapping
- Relations are commutative: i.e., $r_1 \circ r_2 = r_2 \circ r_1$
 - Which is not always true, e.g., father's wife \neq wife's father

Graph Embedding

- What is Graph Embedding
- Shallow Network Embedding
- Knowledge Graph Embedding
- Graph Neural Networks 
- Summary

Limitation of Shallow Network Embedding

- Too many parameters
 - Each node is associated with an embedding vector, which are parameters
- Not inductive
 - Cannot handle new nodes
- Cannot handle node attributes

From shallow embedding to Graph Neural Networks

- The embedding function (encoder) is more complicated
 - Shallow embedding
 - $\phi(v) = U^T x_v$, where U is the embedding matrix and x_v is the one-hot encoding vector
 - Graph neural networks
 - $\phi(v)$ is a neural network depending on the graph structure

Notations

- An attributed graph $G = (V, E)$
 - V : vertex set
 - E : edge set
 - A : adjacency matrix
 - $X \in R^{d_0 \times |V|}$: feature matrix for all the nodes
 - $N(v)$: neighbors of node v
 - h_v^l : Representation vector of node v at Layer l
 - Note $h_v^0 = x_v$
 - $H^l \in R^{d_l \times |V|}$: representation matrix

The General Architecture of GNNs

- For a node v at layer t

$$h_v^{(t)} = f \left(\underbrace{h_v^{(t-1)}}_{\text{representation vector from previous layer for node } v}, \underbrace{\left\{ h_u^{(t-1)} \mid u \in \mathcal{N}(v) \right\}}_{\text{representation vectors from previous layer for node } v\text{'s neighbors}} \right)$$

representation vector
from previous layer for
node v

representation vectors
from previous layer for
node v 's neighbors

- A function of representations of neighbors and itself from previous layers
 - **Aggregation** of neighbors
 - **Transformation** to a different space
 - **Combination** of neighbors and the node itself

Compare with CNN

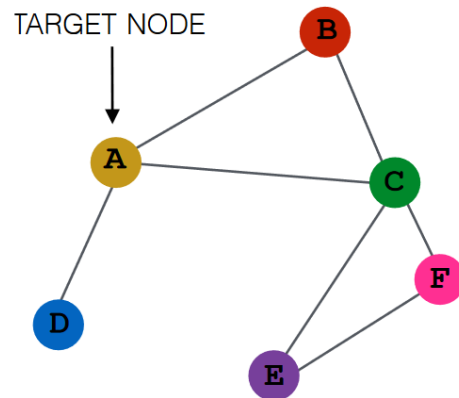
- Recall CNN
 - Regular graph
- GNN
 - Extend to irregular graph structure

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0

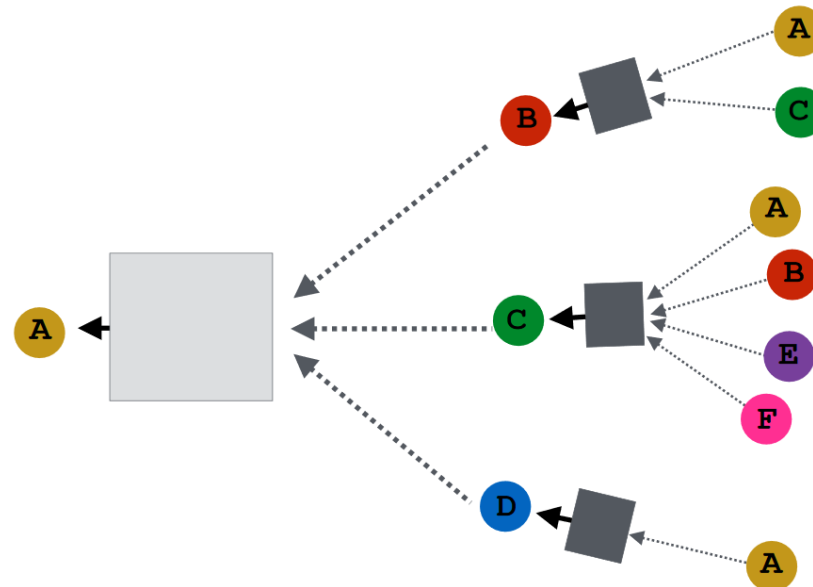
Image

4		

Convolved Feature



INPUT GRAPH



Graph Convolutional Network (GCN)

- Kipf and Welling, ICLR'17

- $f(H^{(l)}, A) = \sigma \left(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right), \hat{A} = A + I$

- f : graph filter

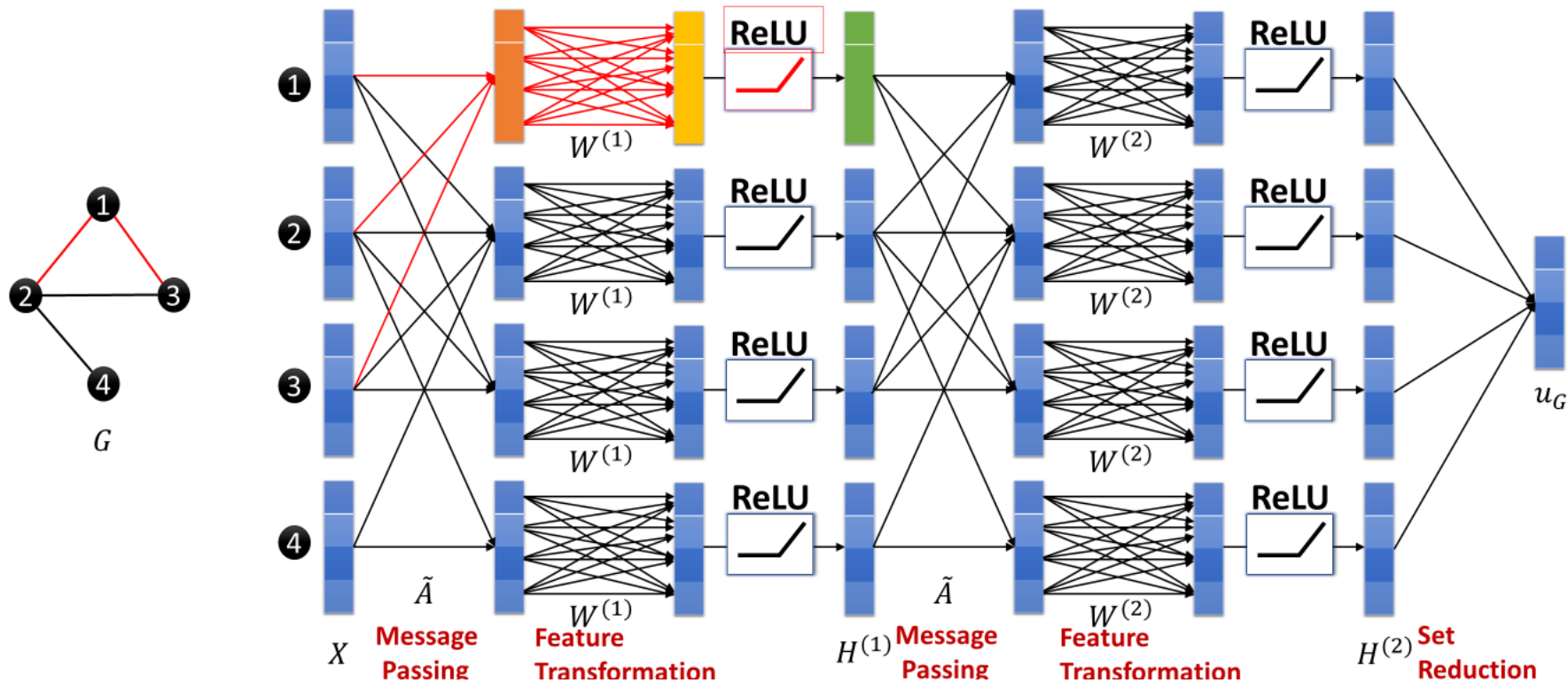
- From a node v 's perspective

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

W_k : weight matrix at Layer k , shared across different nodes

A toy example of 2-layer GCN on a 4-node graph

- Computation graph



Question

- How many parameters are there in a GCN?
 - Assuming initial features are with d_0 dimensions
 - Representation in later layers are with d dimensions

GraphSAGE

- Inductive Representation Learning on Large Graphs

William L. Hamilton*, Rex Ying*, Jure Leskovec, NeurIPS'17

$$\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$$
$$\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$$

A more general form

$$\mathbf{h}_v^k = \sigma\left([\mathbf{W}_k \cdot \overleftarrow{\text{AGG}}(\{\mathbf{h}_u^{k-1}, \forall u \in N(v)\}), \overrightarrow{\mathbf{B}}_k \mathbf{h}_v^{k-1}]\right)$$

More about AGG

- Mean

$$\text{AGG} = \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}$$

- LSTM

$$\text{AGG} = \text{LSTM}([\mathbf{h}_u^{k-1}, \forall u \in \pi(N(v))])$$

- $\pi(\cdot)$: a random permutation

- Pool

$$\text{AGG} = \gamma \{\mathbf{Q}\mathbf{h}_u^{k-1}, \forall u \in N(v)\}$$

- $\gamma(\cdot)$: Element-wise mean/max pooling of neighbor set

Message-Passing Neural Network

- Gilmer et al., 2017. Neural Message Passing for Quantum Chemistry. *ICML*.
- *A general framework that subsumes most GNNs*
 - Can also include **edge** information
- Two steps
 - Get messages from neighbors at step k

$$\mathbf{m}_v^k = \sum_{u \in N(v)} M(\mathbf{h}_u^{k-1}, \mathbf{h}_v^{k-1}, \mathbf{e}_{u,v}) \quad \text{e.g., Sum or MLP}$$

- Update the node latent represent based on the msg

$$\mathbf{h}_v^k = U(\mathbf{h}_v^{k-1}, \mathbf{m}_v^k) \quad \text{e.g., LSTM, GRU}$$

A special case: GGNN, Li et al., Gated graph sequence neural networks, ICLR 2015

Graph Attention Network (GAN)

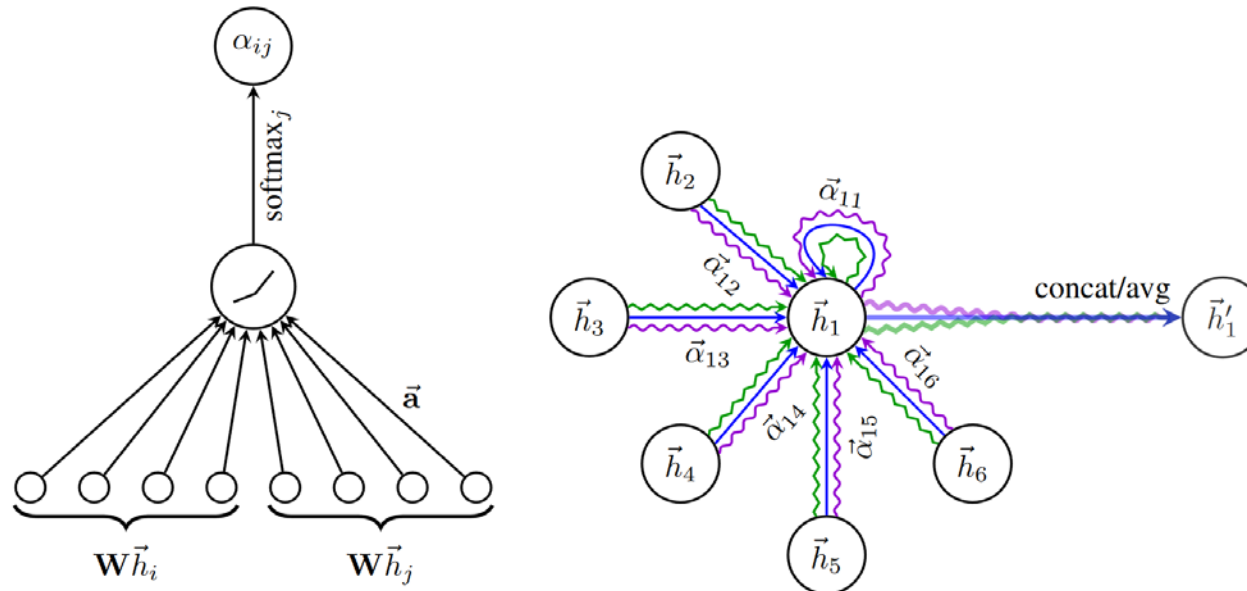
- How to decide the importance of neighbors?
 - GCN: a predefined weight
 - Others: no differentiation
- GAN: decide the weights using learnable attention
 - Velickovic et al., 2018. Graph Attention Networks. *ICLR*.

$$\vec{h}'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} \vec{h}_j \right)$$

The attention mechanism

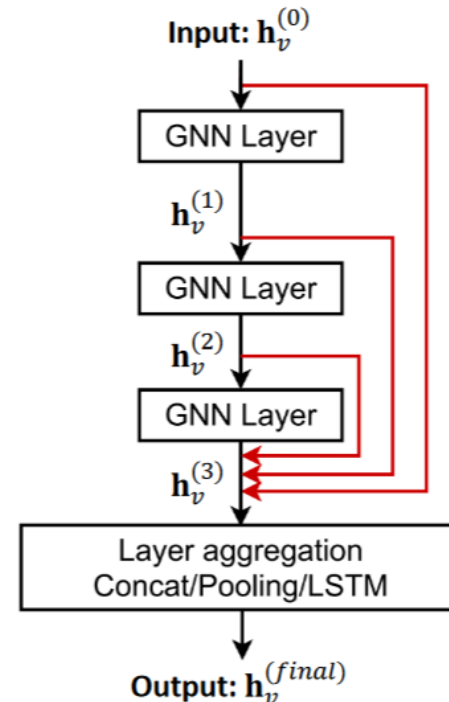
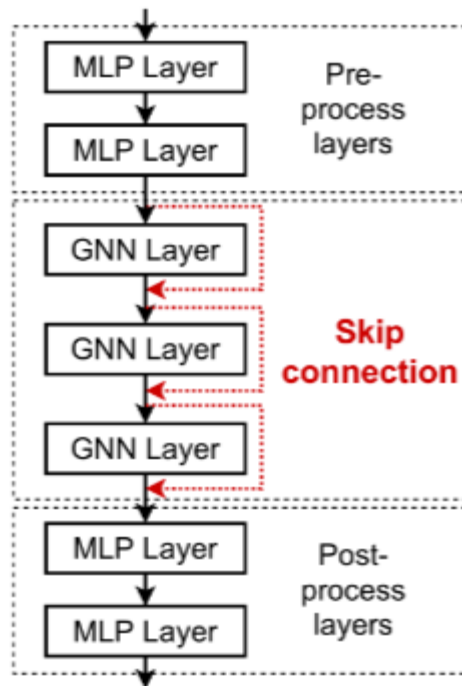
- Potentially many possible designs

$$\alpha_{ij} = \frac{\exp \left(\text{LeakyReLU} \left(\vec{\mathbf{a}}^T [\mathbf{W} \vec{h}_i \parallel \mathbf{W} \vec{h}_j] \right) \right)}{\sum_{k \in \mathcal{N}_i} \exp \left(\text{LeakyReLU} \left(\vec{\mathbf{a}}^T [\mathbf{W} \vec{h}_i \parallel \mathbf{W} \vec{h}_k] \right) \right)}$$



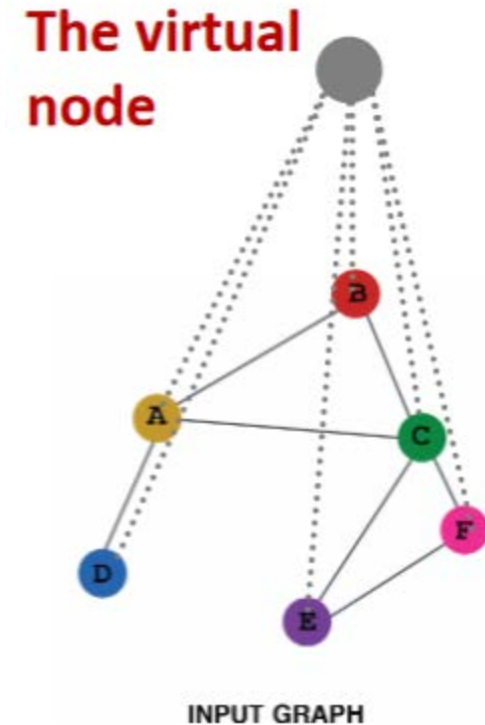
Some practical issues (1): over smoothing

- Adding more GNN layers is not always helpful
 - Over smoothing issue!
 - Solution: (1) skip connection; (2) add MLP layers or use deep transformation within each GNN layer



Some practical issues (2): Sparse Graph

- Augment sparse graphs
 - Add virtual edges, e.g., length-2 path
 - Add virtual nodes, e.g., a cluster node



Downstream Tasks for Graphs

Typical Graph Functions

- Node level

- Similarity search
- **Link prediction**
- **Classification**
- Community detection
- Ranking

- Graph level

- **Similarity search**
- Frequent pattern mining
- Graph isomorphism test
- Graph matching
- **Classification**
- Clustering
- Graph generation

1. Semi-supervised Node Classification

- Decoder using $z_v = h_v^L$
 - Feed into another fully connected layer
 - $\hat{y}_v = \sigma(\theta^T z_v)$
- Loss function
 - Cross entropy loss
 - In a binary classification case
 - $l_v = -y_v \log \hat{y}_v - (1 - y_v) \log(1 - \hat{y}_v)$

Applications of Node Classification

- Social network
 - An account is bot or not
- Citation network
 - A paper's research field
- A program-derived graph
 - The type of a variable

2. Link Prediction

- Decoder using $z_v = h_v^L$
 - Given a node pair (u, v)
 - Determine its probability $p_{uv} = \sigma(z_u^T R z_v)$
 - R could be different for different relation type
- Loss function
 - Cross entropy loss
 - $l_{uv} = -y_{uv} \log p_{uv} - (1 - y_{uv}) \log(1 - p_{uv})$

Link Prediction Applications

- Social network
 - Friend recommendation
- Citation network
 - Citation recommendation
- Medical network
 - Drug and target binding or not
- A program-derived graph
 - Code autocomplete

3. Graph Classification

- Decoder using $h_G = g(\{z_v\}_{v \in V})$
 - $g(\cdot)$: a read out function, e.g., sum
 - Feed h_G into another fully connected layer
 - $\hat{y}_G = \sigma(\theta^T h_G)$
- Loss function
 - Cross entropy loss
 - In a binary classification case
 - $l_G = -y_G \log \hat{y}_G - (1 - y_G) \log(1 - \hat{y}_G)$

Graph Classification Applications

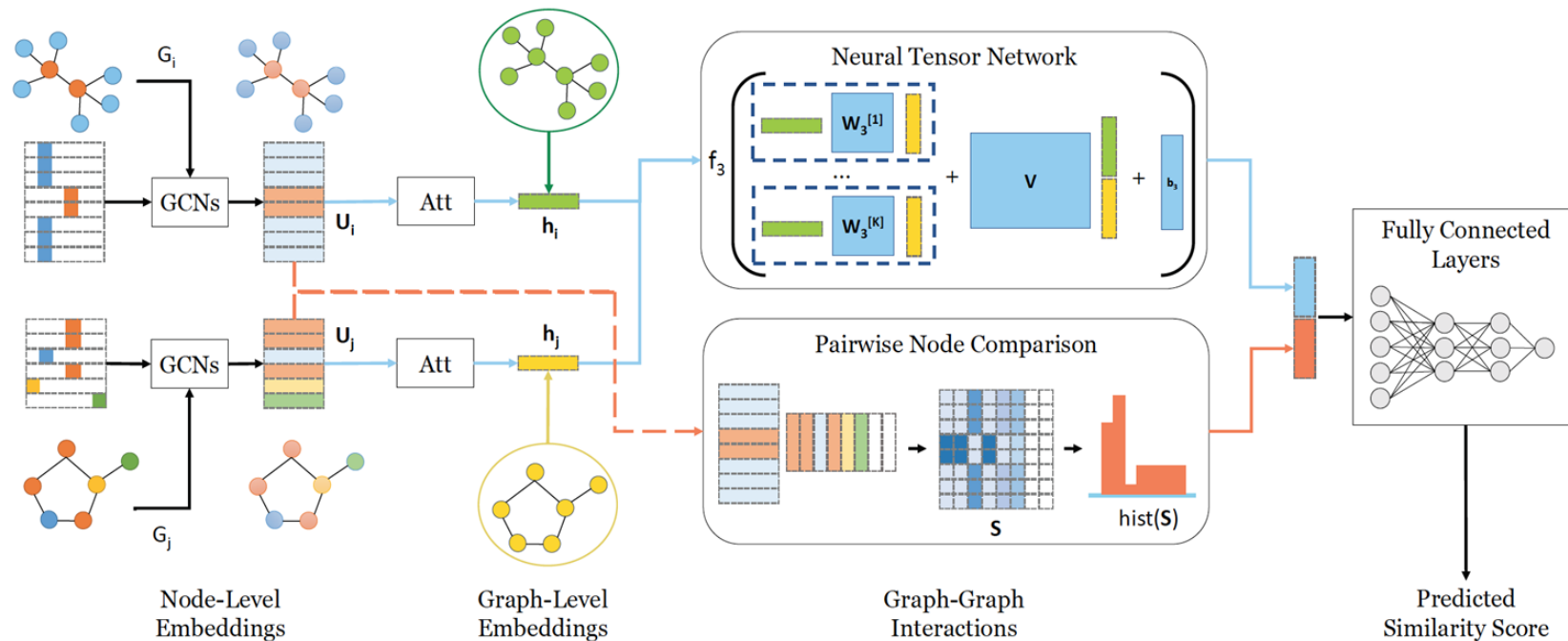
- Chemical compounds
 - Toxic or not
- Proteins
 - Has certain function or not
- Program-derived graphs
 - Contains bugs or not

4. Graph Similarity Computation

- Decoder using $h_G = g(\{z_v\}_{v \in V})$
 - Given a graph pair (G_1, G_2)
 - Determine its score $s_{G_1 G_2} = h_{G_1}^T R h_{G_2}$
- Loss function
 - E.g., Square loss
 - $l_{G_1 G_2} = (y_{G_1 G_2} - s_{G_1 G_2})^2$

A Concrete solution by SimGNN [Bai et al., AAAI 2019]

- Goal: learn a GNN $\phi: \mathcal{G} \times \mathcal{G} \rightarrow R^+$ to approximate Graph Edit Distance between two graphs



1. Attention-based graph-level embedding
2. Histogram features from pairwise node similarities

Graph Similarity Computation Applications

- Drug database
 - Drug similarity search
- Program database
 - Code recommendation
 - Search ninja code for novice code
 - Search java code for COBOL code



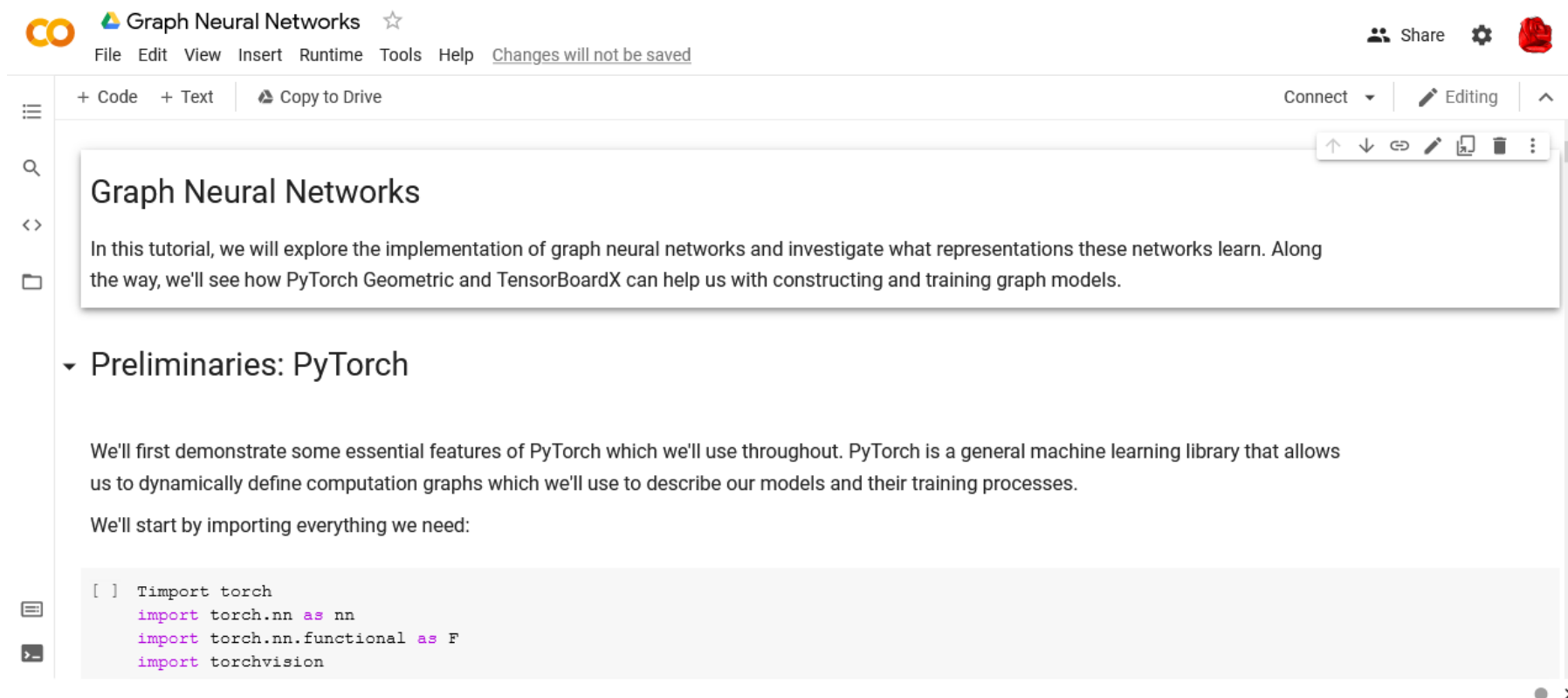
Wanted urgently: People who know a half century-old computer language so states can process unemployment claims

By Alicia Lee, CNN

Updated 4:00 PM ET, Wed April 8, 2020

Tutorial on GNN coding


- <https://colab.research.google.com/drive/1DIQm9rOx2mT1bZETEEVUThxcrP1RKqAn>



The screenshot shows a Google Colab notebook interface. At the top, the title is "Graph Neural Networks" with a star icon. Below the title, there is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A status bar indicates "Changes will not be saved". On the right side, there are icons for "Share", "Settings", and a red error icon. The notebook content is displayed in a text area with a search icon on the left. The text reads: "Graph Neural Networks" followed by a paragraph: "In this tutorial, we will explore the implementation of graph neural networks and investigate what representations these networks learn. Along the way, we'll see how PyTorch Geometric and TensorBoardX can help us with constructing and training graph models." Below this is a section header "Preliminaries: PyTorch" with a dropdown arrow. The text continues: "We'll first demonstrate some essential features of PyTorch which we'll use throughout. PyTorch is a general machine learning library that allows us to dynamically define computation graphs which we'll use to describe our models and their training processes." and "We'll start by importing everything we need:". At the bottom, there is a code block with the following Python code:

```
[ ] import torch
import torch.nn as nn
import torch.nn.functional as F
import torchvision
```

Graph Embedding

- What is Graph Embedding
- Shallow Network Embedding
- Knowledge Graph Embedding
- Graph Neural Networks
- Summary 

Summary

- Graph embedding
- Shallow embedding
 - E.g., LINE
- Knowledge graph embedding
 - E.g., TransE
- Graph neural networks
 - E.g., GCN

References

- Bryan Perozzi, Rami Al-Rfou, Steven Skiena, DeepWalk: Online Learning of Social Representations, KDD'14
- Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, Qiaozhu Mei, LINE: Large-scale Information Network Embedding, WWW'15
- Aditya Grover, Jure Leskovec, node2vec: Scalable Feature Learning for Networks, KDD'16
- Kipf & Welling, [Semi-Supervised Classification with Graph Convolutional Networks](#), ICLR 2017
- Tutorial: <http://snap.stanford.edu/proj/embeddings-www/files/nrltutorial-part2-gnns.pdf>
- Tutorial: <http://tkipf.github.io/misc/SlidesCambridge.pdf>
- Bordes et al., Translating Embeddings for Modeling Multi-relational Data, NIPS 2013
- Yang et al., Embedding entities and relations for learning and inference in knowledge bases, ICLR 2015
- Zhiqing Sun, Zhihong Deng, Jian-Yun Nie, and Jian Tang. "RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space." ICLR'19.

slido



Audience Q&A Session

ⓘ Start presenting to display the audience questions on this slide.