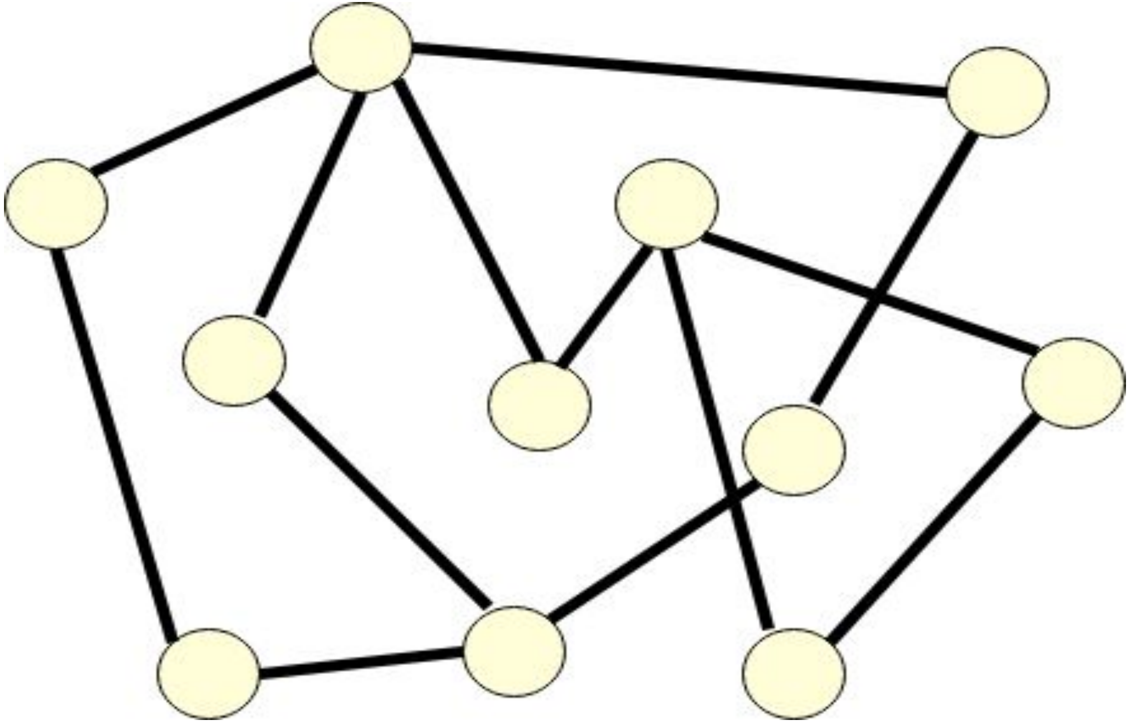
Graph Representation Learning

Kobby Panford-Quainoo
kpanford-quainoo@aimsammi.org
https://panford.github.io

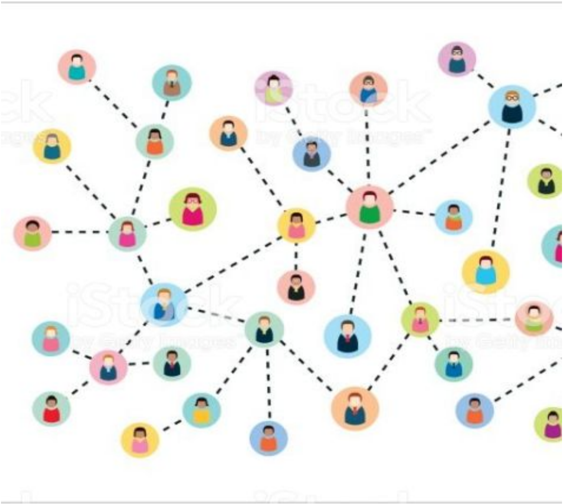
Overview

- Graph-structured data
- Graph Neural Networks
- Applications

Graphs

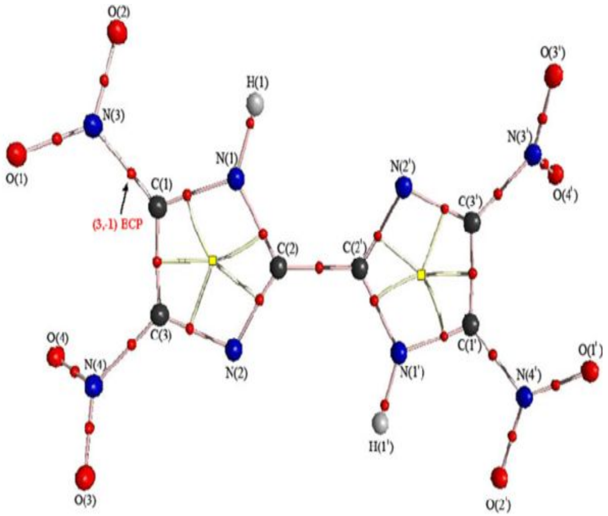


Graphs - Where we find them...



Social Media

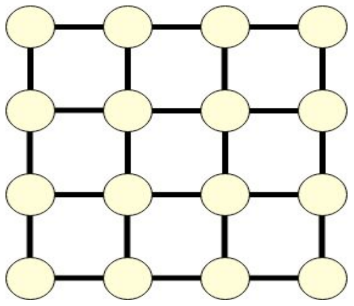
Inter-connected
devices on internet



Molecular bonds

Graphs

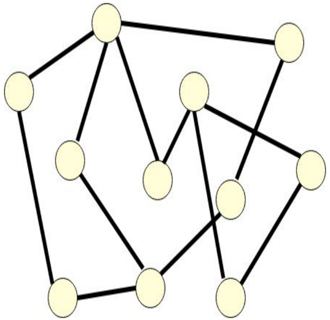
- Euclidean graph-data has regular/ periodic structure
  - Eg. Images, videos, time series
- Discrete, Non-Euclidean data structure



Euclidean/regular grid

Graphs

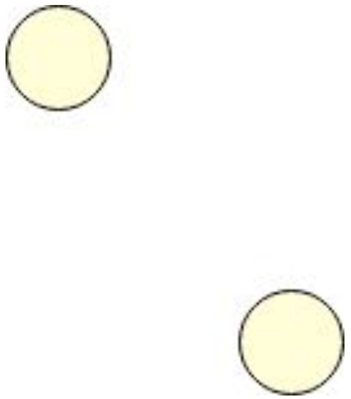
- Euclidean graph-data has regular/ periodic structure
  - Eg. Images, videos, time series
- Discrete, Non-Euclidean data structure
  - Eg. social networks, molecular structure



Non-euclidean/
non-regular grid

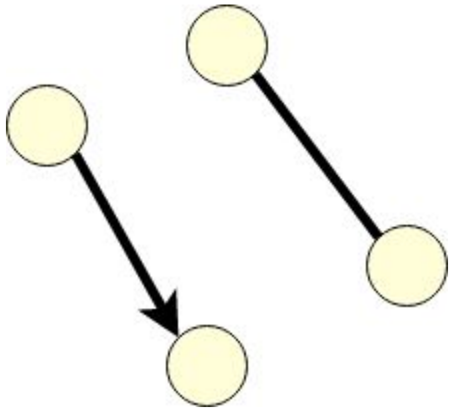
Graphs

- Discrete, Non-Euclidean data structure
- Key features are;
    - **Nodes**
    - **Node features**



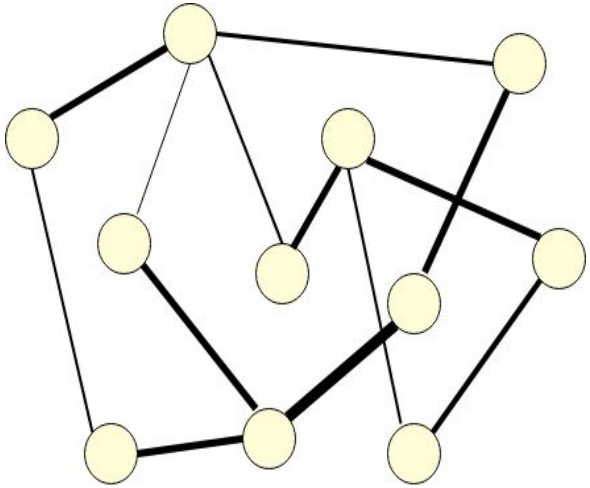
Graphs

- Discrete, Non-Euclidean data structure
- Key features are;
  - Nodes
  - Node features
  - **Edges (directed or undirected)**



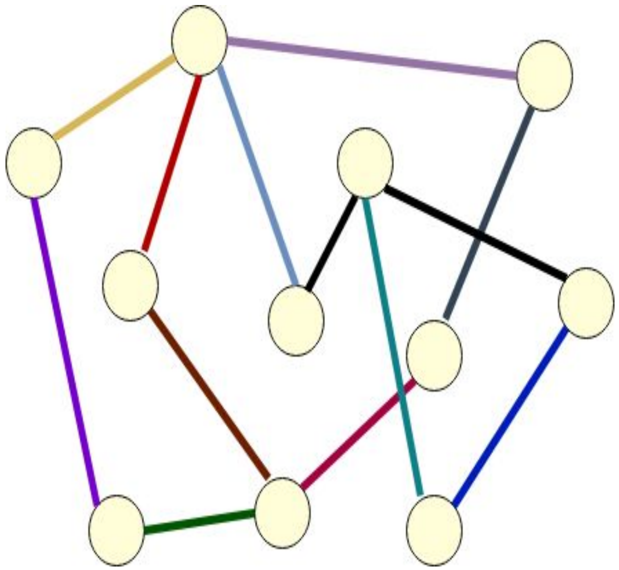
Graphs

- Discrete, Non-Euclidean data structure
- Key features are;
  - Nodes
  - Node features
  - Edges (directed or undirected)
  - **Edge weights**
  - **Edge features**



Graphs

- Discrete, Non-Euclidean data structure
- Key features are;
  - Nodes
  - Node features
  - Edges (directed or undirected)
  - Edge weights
  - Edge features
  - **Edge types** etc



Graphs - More Formally

A simple graph is given by: $\mathcal{G} = (\mathcal{V}, X, A, \mathcal{E})$

Where $\mathcal{V} \rightarrow$ set of nodes/ vertices, X $\rightarrow$ node features

$\mathcal{E} \rightarrow$ set of edges weights, A $\rightarrow$ Adjacency matrix

$$\mathrm{X} = \begin{pmatrix} x_{11} & . & . & . & x_{1d} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{n1} & . & . & . & x_{nd} \end{pmatrix}$$

Assuming a real-valued feature matrix X for each node $v \in \mathcal{V}$

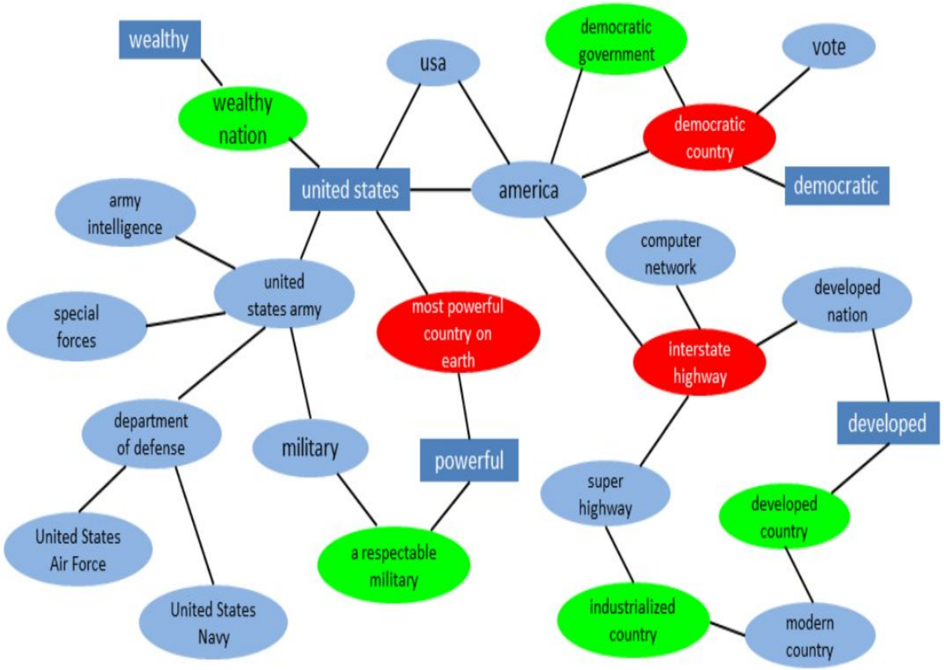
Adjacency Matrix -

A simple binary matrix -

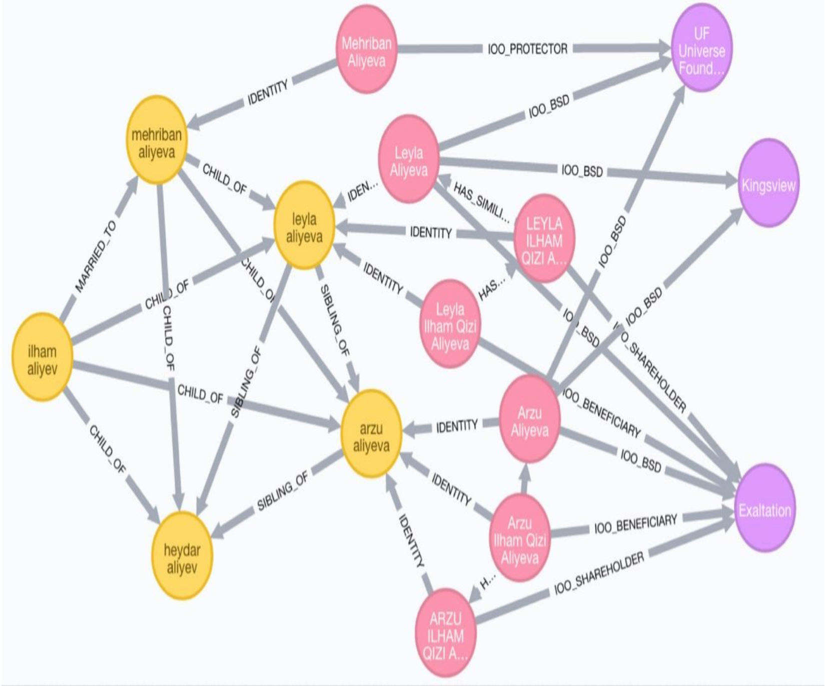
Entry e_{ij} is 1 if nodes are connected and 0 otherwise

$$\mathrm{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Knowledge Graphs

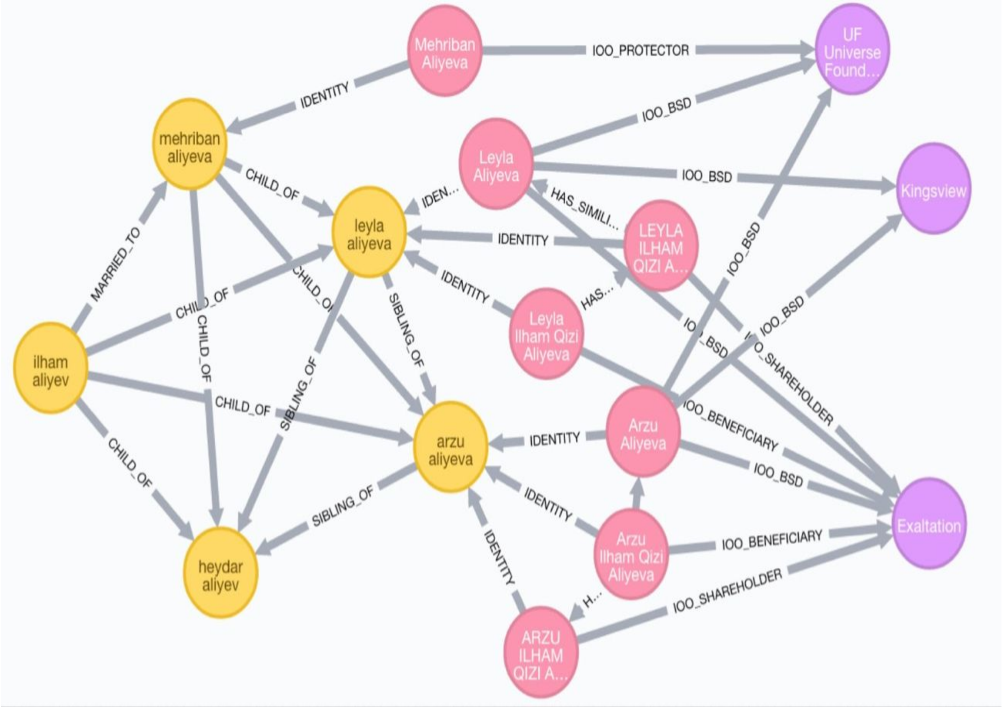


Generated by ConceptNet



Neo4j

Knowledge Graphs



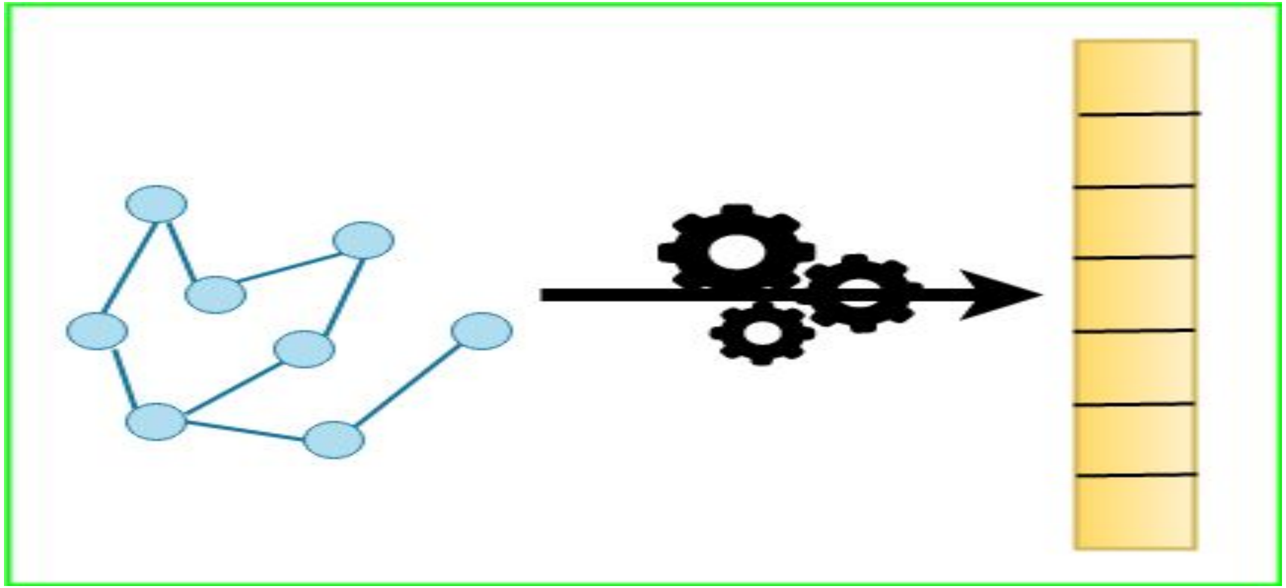
- Fact triplets $(h, r, t) \rightarrow$ head entity, relation, tail entity
- Symmetry is critical (marriage is symmetric, “is a capital of” is not)
- Usually incomplete → missing links needs to be completed
- We need embedding methods to learn the KG structure and to predict missing links

Main Downstream Tasks

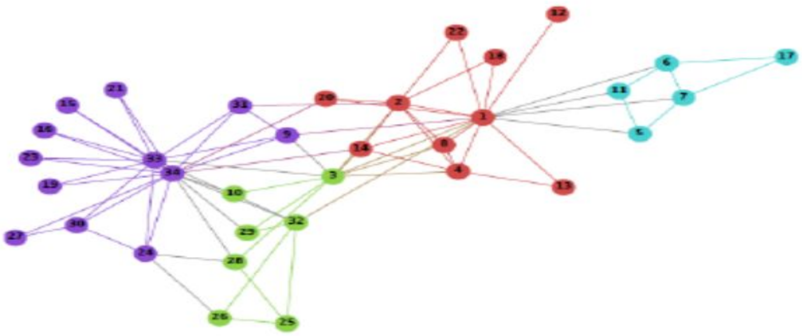
- **Node/ Graph Classification** - Label nodes/entire graph that were not labeled before in a typical semi-supervised setting
- **Link Prediction** - Predict links between nodes
- **Node Clustering/ Community Detection** - Detect clusters of nodes in graph
- **Graph Generation** - Generate a new graph

What we want

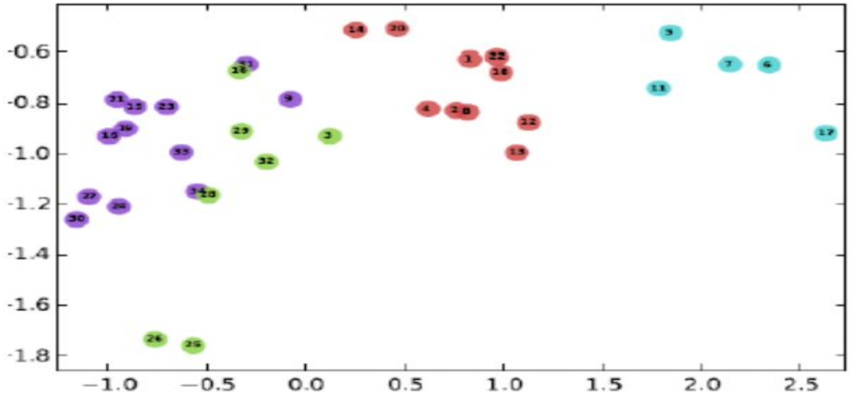
Priori: *Learn Good Embeddings for better model*



- **Zachary's Karate Club network:**



Input



Output

Fundamental Questions

Priori: *Learn Good Embeddings for better model*

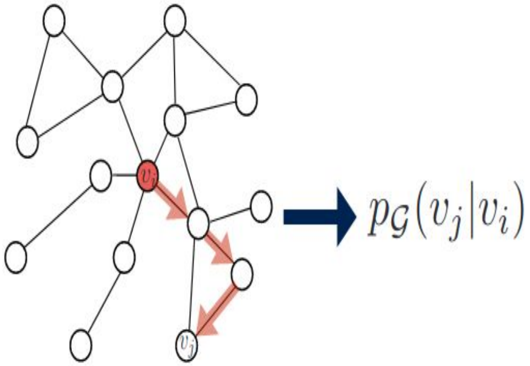
How to encode the graph structure into machine learning models;

- Leveraging a node’s *local or global neighbourhood structure* into a feature vector for **node classification**
- Encoding *pairwise information* between nodes for **link prediction**

Classical Rep. Approaches

- Transductive learning
- No parameter sharing
- No feature learned representation

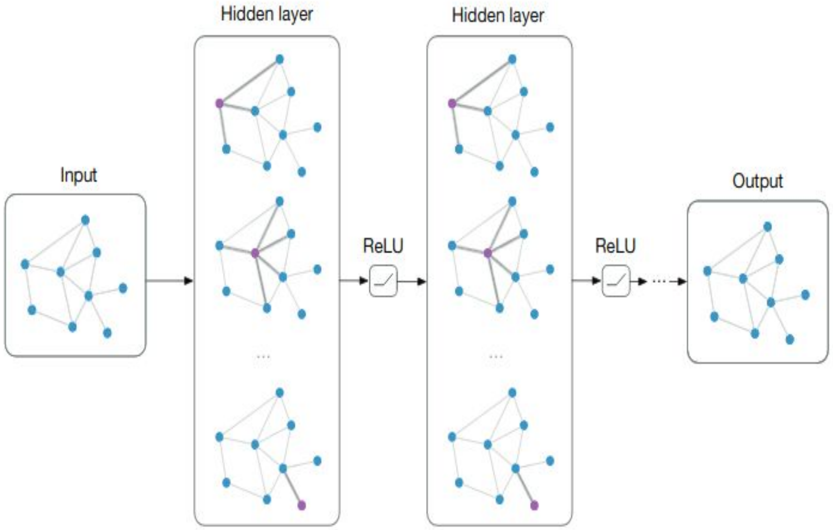
- Random walks
  - DeepWalk
  - Node2vec
  - LINE (Large-scale Information Network Embeddings)
  - HARP (Random-walk embeddings via graph pre-processing)
- Neighborhood autoencoder and aggregation
  - Deep Neural Graph Representations (DNGR)
  - Structural Deep Network Embeddings (SDNE) etc.



Hamilton, 2018

Graph Neural Networks

1. Simply a family of deep learning/ neural network methods applied on graphs
2. Many tricks and hacks in NN applies
- ReLU activation
- Graph Pooling
- Stacking layers for hierarchical feature learning
- Negative Sampling
- Subsampling
3. Many idea “Message passing”



Graph Neural Networks

- Graph Convolutional Networks
- Graph Attention Networks
- Graph Recurrent Networks
- Graph Autoencoder
- Variational Graph Autoencoder

Graph Convolutional Networks (GCN)

Kipf & Welling (ICLR 2017)

- Permutation invariance
- Weight shared between layers
- Linear complexity O(E)

Update rule: $$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

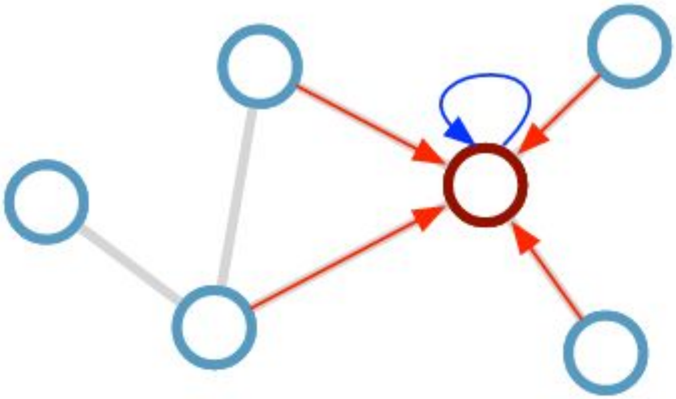


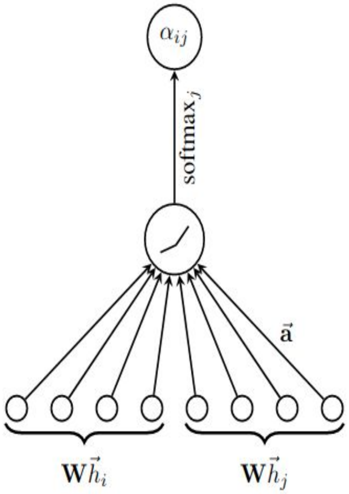
Illustration from Kipf’s slides

Graph Attention Networks

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio

Pay attention to the most influential nodes

- Computes attention vector α that weighs every neighboring node by importance

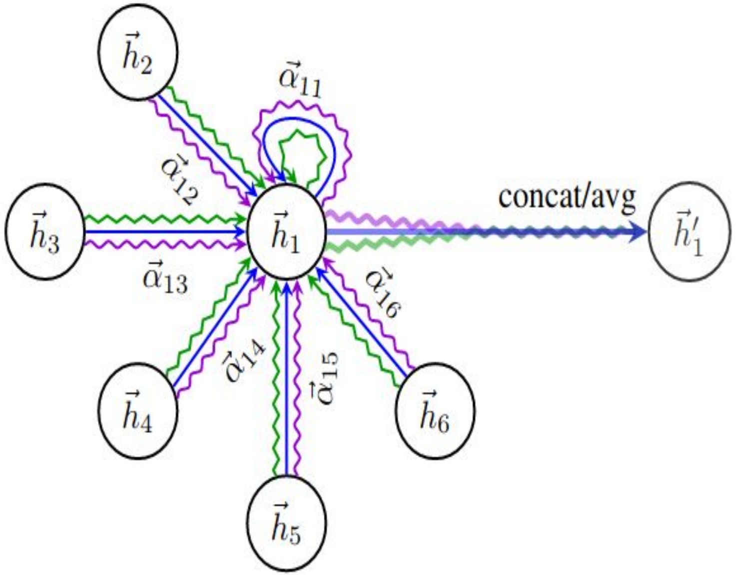


$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

Graph Attention Networks

Pay attention to the most influential nodes

- Computes attention vector \alpha that weighs every neighboring node by importance
- Multi-head attention computes and concatenates k independent attention mechanisms.
- If final layer, we average out



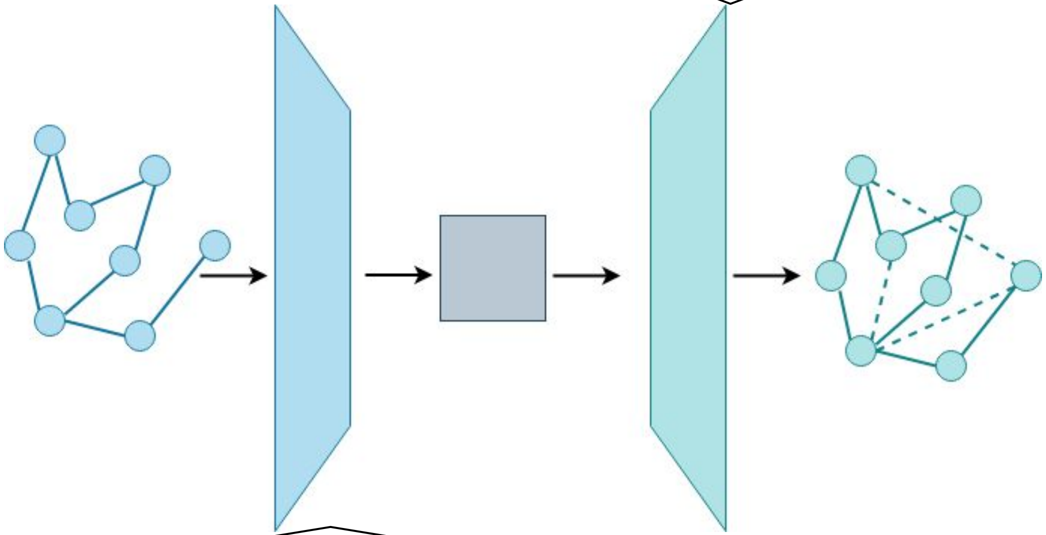
$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$

Graph Autoencoders

Encoder-Decoder model

$$p(\mathbf{A}|\mathbf{z}) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(A_{ij}|\mathbf{z}_i, \mathbf{z}_j),$$

$$p(A_{ij}|\mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^\top \mathbf{z}_j).$$

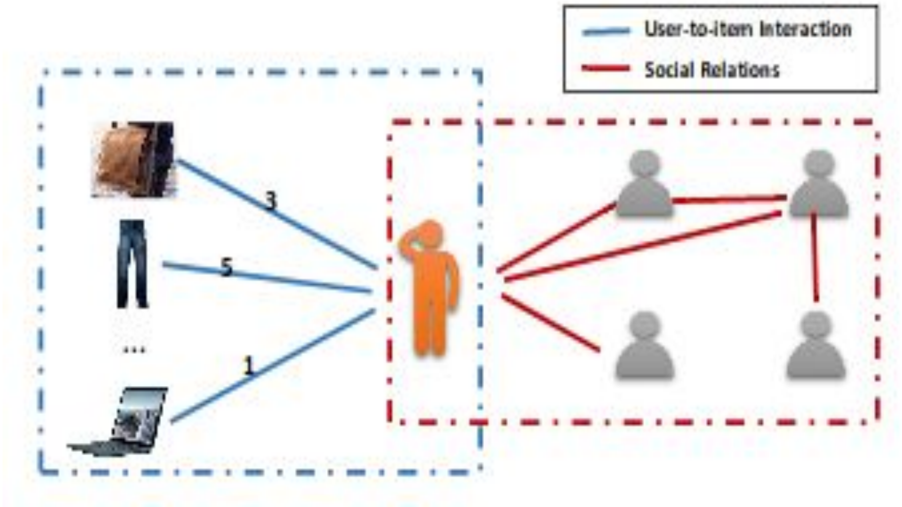


$$f(\mathbf{z}|\mathbf{X}, \mathbf{A}) = GCN(\mathbf{A})$$

Applications

Graph Neural Networks for Social Recommendation

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, Dawei Yin

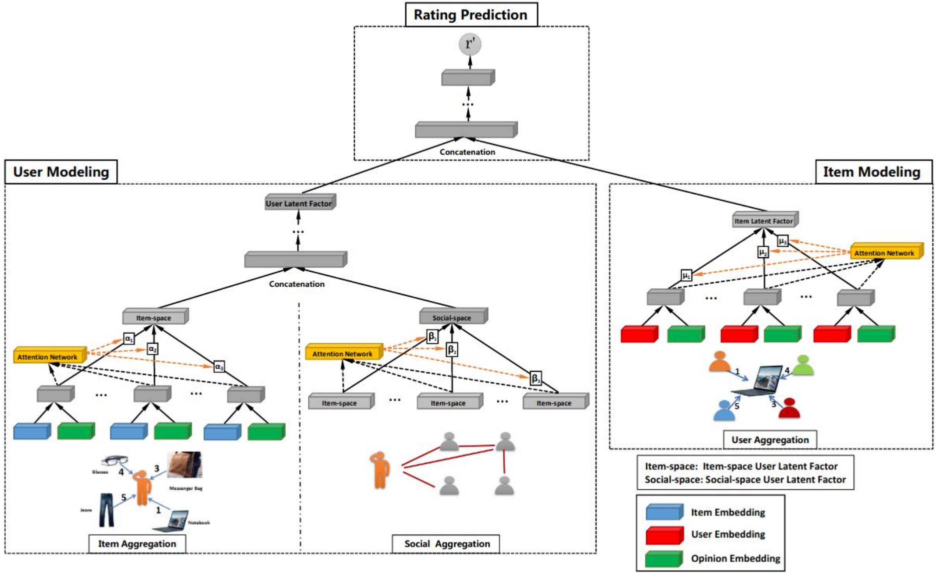


Two parts graph;

- user-item graph (Numbers denotes rating)
- User-user social graph

Graph Neural Networks for Social Recommendation

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, Dawei Yin



Separately model hidden state;

- User-item interaction - Item Aggregation (IA)
- User-user interaction - Social aggregation (SA)

Feed concatenated state representation from IA and SA into MLP and predict *rating*

Multi-Agent Game Abstraction via Graph Attention Neural Network

Yong Liu, Weixun Wang, Yujing Hu, Jianye Hao, Xingguo Chen, Yang Gao

3 Our Method

In this section, we propose a novel game abstraction approach based on two-stage attention mechanism (G2ANet). Based on the mechanism, we propose two novel MARL algorithms (GA-Comm and GA-AC).

G2ANet: Game Abstraction Based on Two-Stage Attention

We construct the relationship between agents as a graph, where each node represents a single agent, and all nodes are connected in pairs by default. We define the graph as Agent-Coordination Graph.

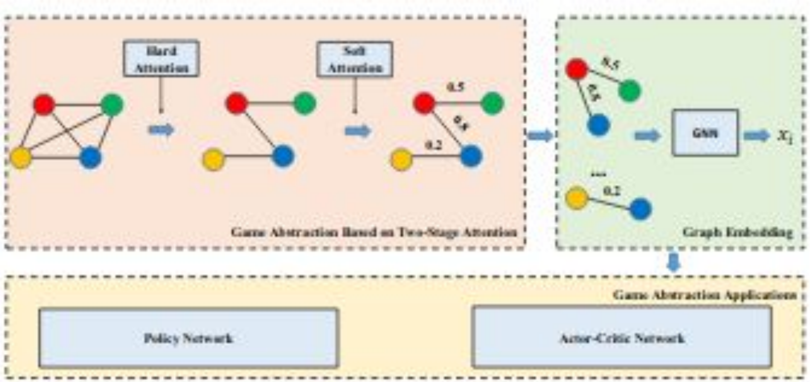


Figure 1: Game Abstraction based on two-stage attention mechanism and Graph Neural Network (GNN).

Bilateral Trade Modeling with GNN

Kobby Panford-Quainoo, Avishek Joey Bose, Michael Defferrard

- Data -- Bilateral trade data, Countries profile information
- Tasks
  - Node Classification
    - Predict income levels of countries -- High, Upper-middle, Lower-middle, Low
  - Link Prediction
    - Predict if any two countries would trade

Feature	Notation	Size	Representation
nodes	$\mathcal{V}$	111	countries
node features	X	38	population etc.[1]
edges	**A**	476	trade indicator
edge weights	$\mathcal{E}$	476	net trade val (USD)
node labels	$\mathcal{Y}$	4	income group

Others

GCap:Graph-based Automatic Image Captioning --- Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, Pinar Duygulu

MolGAN: An implicit generative model for small molecular graphs --- Nicola De Cao Thomas Kipf

Convolutional Networks on Graphs for Learning Molecular Fingerprints - David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gomez-Bombarelli, Timothy Hirzel, Alan Aspuru-Guzik, Ryan P. Adams

GNN Toolkits

Datasets:

- Cora
- Citeseer
- PubMed
- TU Dataset
- Protein interaction dataset

Python Libraries

- DeepGraph Library (DGL)
- Pytorch-Geometric
- Pytorch-BigGraph

Open Graph Benchmark:

A collection of datasets, benchmarks and evaluators for machine learning on graphs

https://ogb.stanford.edu/

Reference Materials

Ziwei Zhang, Peng Cui and Wenwu Zhu. **Deep Learning on Graphs: A survey**

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, Xavier Bresson. **Benchmarking Graph Neural Networks**

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, Maosong Sun. **Graph Neural Networks:A Review of Methods and Applications**

William L. Hamilton, Rex Ying, Jure Leskovec. **Representation Learning on Graphs: Methods and Applications**