# METODE INTELIGENTE DE REZOLVARE A PROBLEMELOR REALE

Laura Dioşan

Graph-based learning

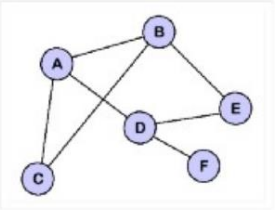
Facultatea de Matematică și Informatică
Universitatea Babeș-Bolyai

# Graph neural networks

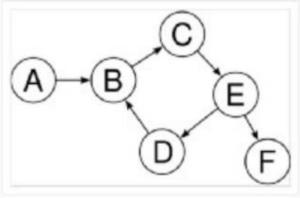
- Why?
- How?
- Applications

# Graphs

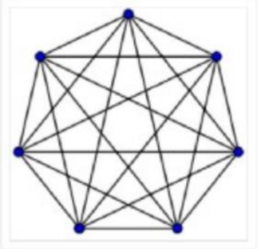
- a type of data structure having two components:
  - nodes (or vertices)
    - Homogenous
    - Heterogenous
  - edges, which connect two nodes
    - Unidirectional
    - Bidirectional
    - With or without weights
- a graph = a collection of loosely inter-connected nodes via edges




undirected graph




directed graph



complete graph

# Graph NNs

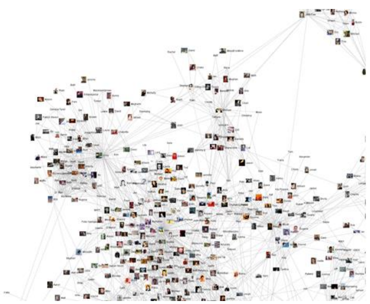
- neural network architectures that operate on a graph.

- Aim:
  - for each node in the graph to learn an embedding containing information about its neighborhood (nodes directly connected to the target node via edges).
  - This embedding can then be used for different problems like node labelling, node prediction, edge prediction, etc.

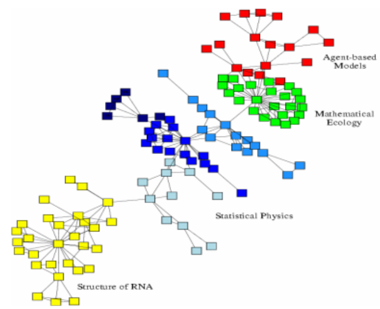
- Real-life applications
  - Social Network Analysis — Similar posts prediction, tags prediction, and recommending content to users.
  - Natural Sciences — GNNs have also gained popularity in dealing with molecular interactions like protein-protein interactions.
  - Recommender Systems — A heterogenous graph can be used to capture relationships between users and items to recommend relevant items to a buyer.

# Why graphs?

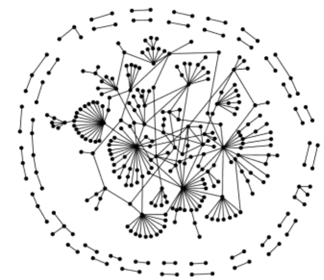
- Graphs (= networks, systems) are a general language for describing and modeling complex systems
- Many data are represented as graphs



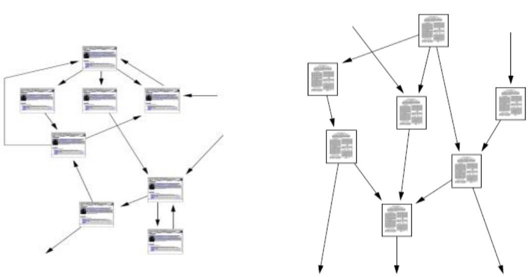
Social networks



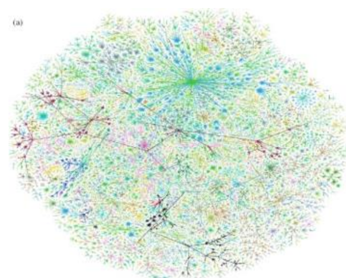

Economic networks



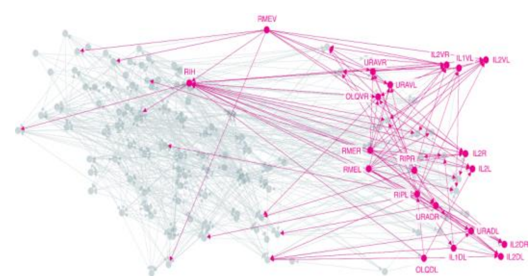
Biomedical networks



Information networks:
Web & citations



Internet

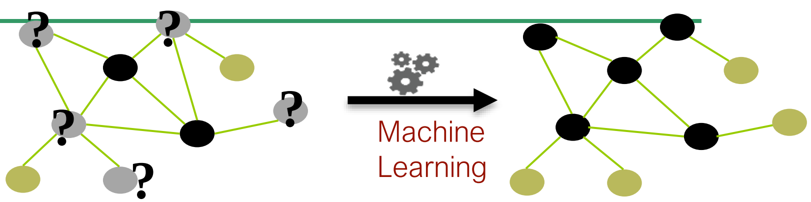


Networks of neurons

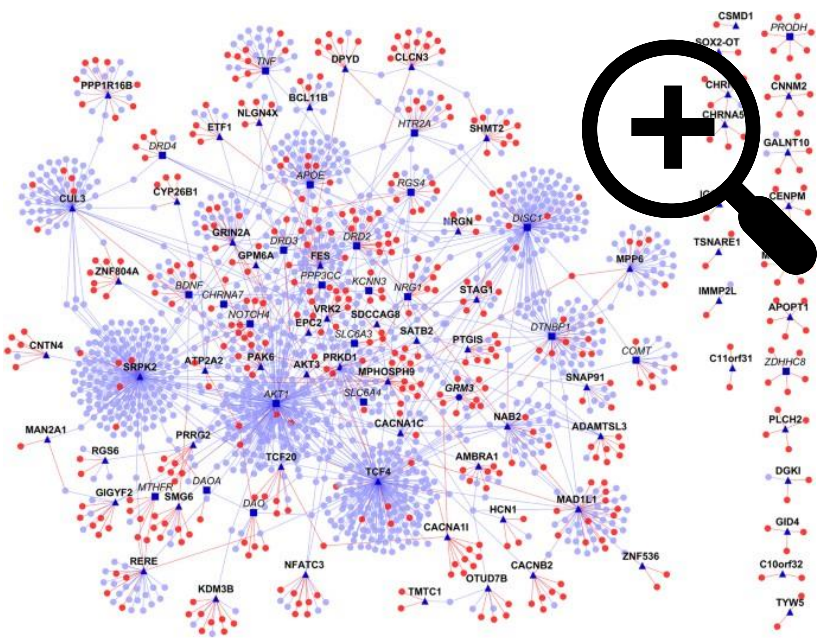
# Why graphs?

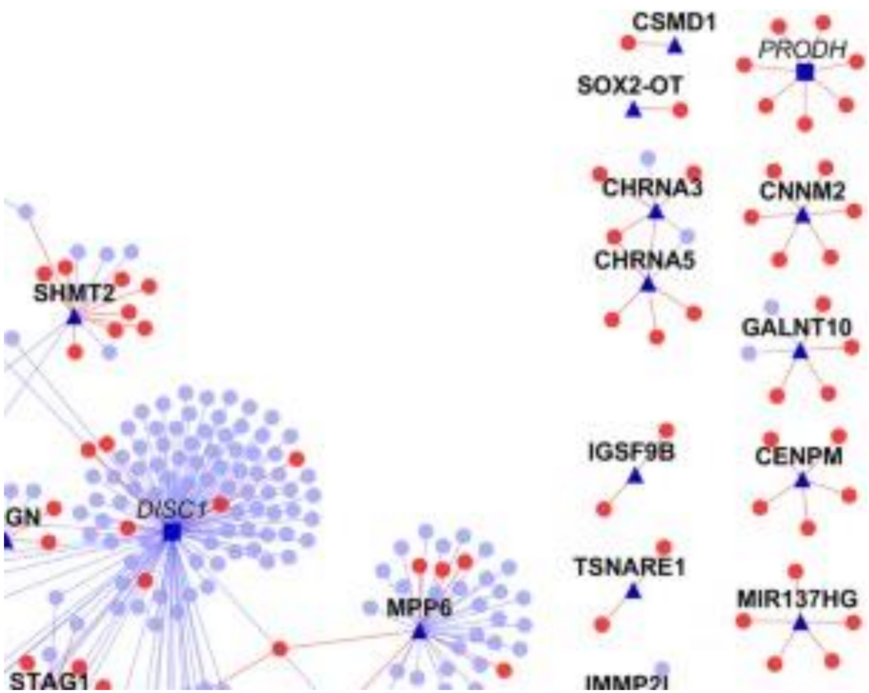
- Graphs (= networks, systems) are a general language for describing and modeling complex systems
- Many data are represented as graphs
- Universal language for describing complex data
  - Networks from science, nature, and technology are more similar than one would expect
- Shared vocabulary between fields
  - Computer Science, Social science, Physics, Economics, Statistics, Biology
- Data availability (+computational challenges)
  - Web/mobile, bio, health, and medical
- Impact!
  - Social networking, Social media, Drug design

# Machine learning with / in graphs



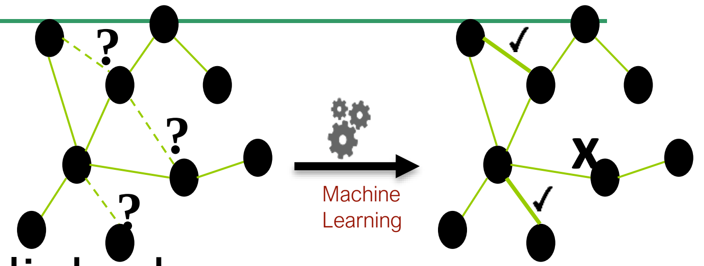

- Node classification
  - Predict the type of a given node
  - Classifying the function of proteins in the interactome!
    - See Ganapathiraju, M. K., Thahir, M., Handen, A., Sarkar, S. N., Sweet, R. A., Nimgaonkar, V. L., ... & Chaparala, S. (2016). Schizophrenia interactome with 504 novel protein–protein interactions. *NPJ schizophrenia*, *2*(1), 1-10. https://www.nature.com/articles/npjschz201612

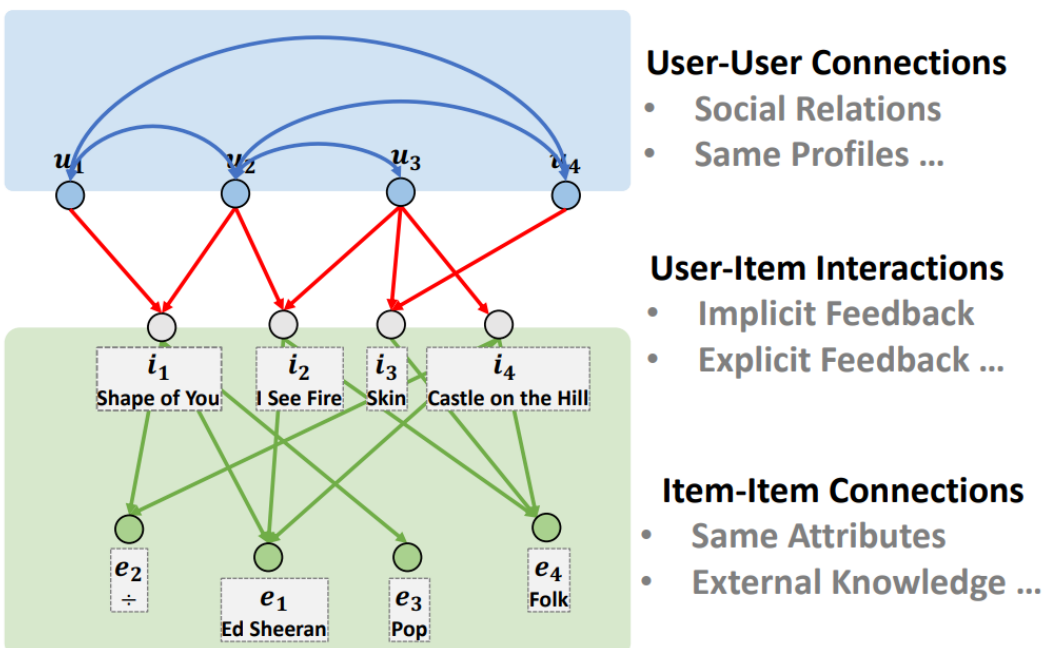




# Machine learning with / in graphs

- Node classification
- Link prediction
  - Predict whether two nodes are linked
  - Recommender systems



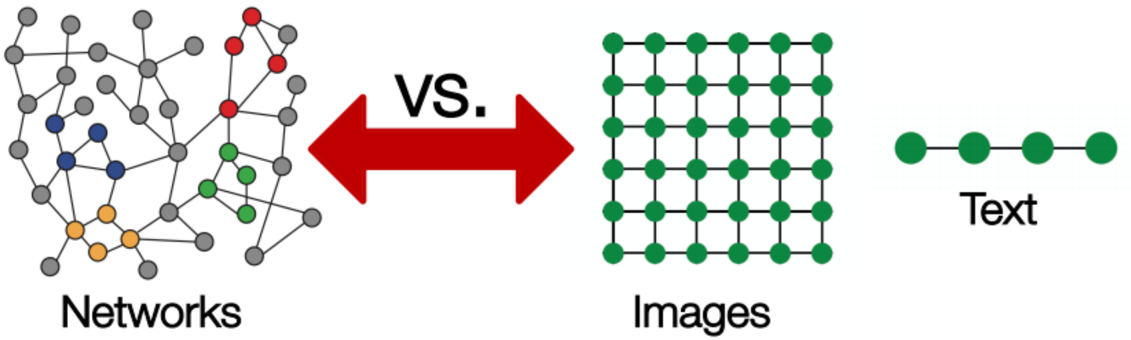




# Machine learning with / in graphs

- Node classification
  - Predict a type of a given node
- Link prediction
  - Predict whether two nodes are linked
- Community detection
  - Identify densely linked clusters of nodes
- Network similarity
  - How similar are two (sub)networks

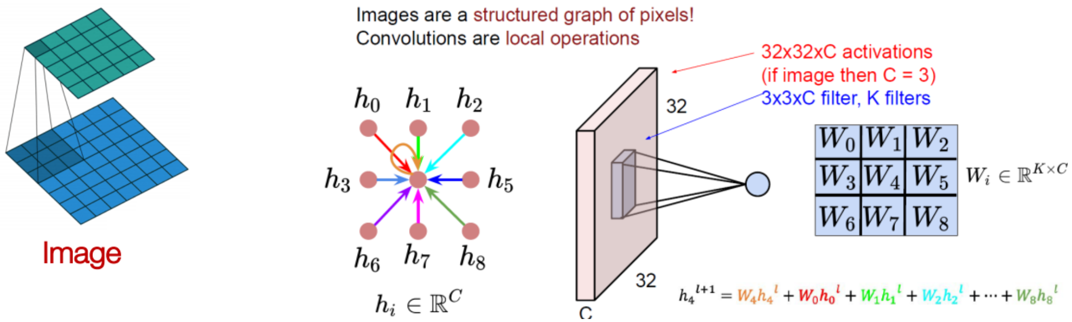
# Graph neural networks

- Data structures
  - Text/speech -> sequences -> RNN
  - Images -> regular grids (matrix) -> CNN
  - Graphs
    - Arbitrary size
    - Complex topological structure

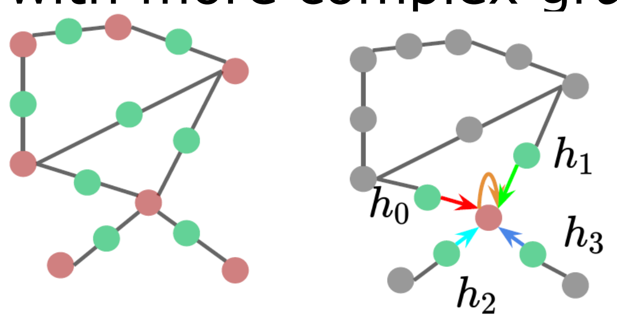


# Graph neural networks

- Convolutions over data
  - Combine the information from the current element with that from neighbours $\sum w_i h_i$




  - How to deal with more complex graphs?



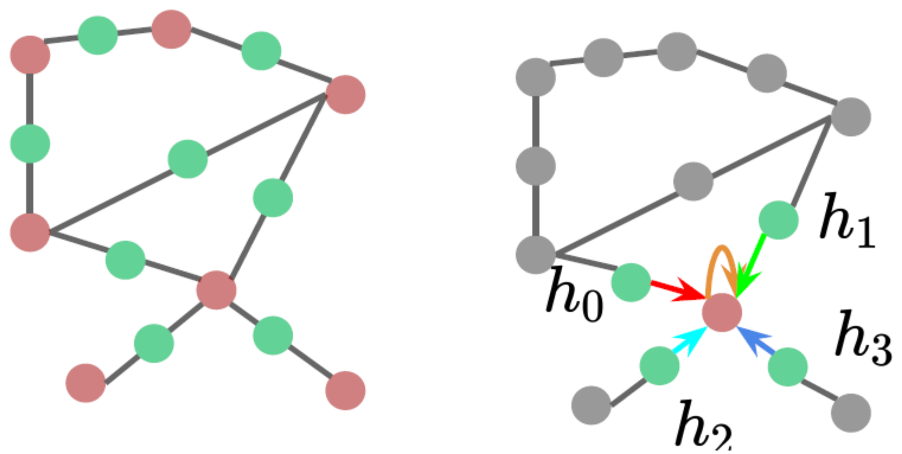

- Graph convolutions involve similar local operations on nodes.
- Nodes are now object representations and not activations
- The ordering of neighbors should not matter.
- The number of neighbors should not matter.
- N(i) are the neighbors of node i
- $c_{ij}$ is a normalization constant

$$\cancel{h_4^{\,l+1} = W_4 h_4^{\,l} + W_0 h_0^{\,l} + W_1 h_1^{\,l} + W_2 h_2^{\,l} + W_3 h_3^{\,l}}$$

$$h_i^{\,l+1} = W_i h_i^{\,l} + \sum_{j \in \mathcal{N}(i)} \frac{1}{c_{ij}} W_j h_j^{\,l}$$

# Graph neural networks

## Convolutions over graphs




- Updates from some neighbors can be more important than others.
- Attention over neighbors allows graph convolutions to focus on specific neighbors
- $\sigma$ is a non-linearity, usually ReLU or LeakyReLU.

Without attention: $$h_i^{l+1} = W_i h_i^{l} + \sum_{j \in \mathcal{N}(i)} \frac{1}{c_{ij}} W_j h_j^{l}$$

With attention: $$h_i^{l+1} = W_i h_i^{l} + \sum_{j \in \mathcal{N}(i)} \frac{1}{c_{ij}} \alpha_{ij} W_j h_j^{l}$$

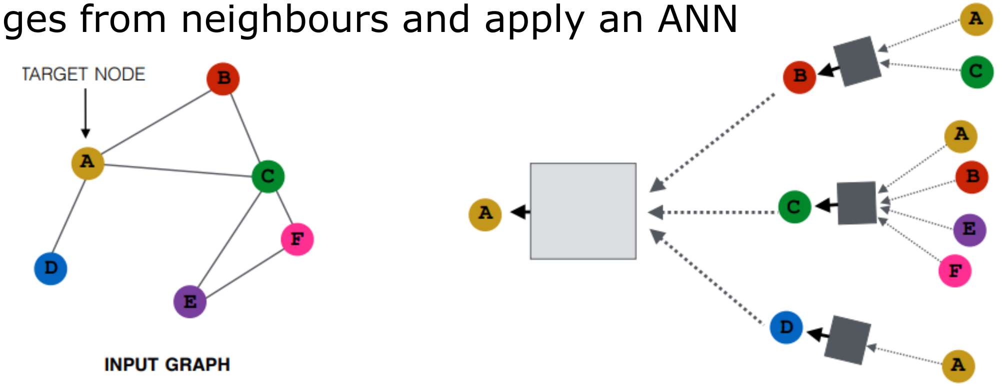
where $$\alpha_{ij} = \frac{e^{\sigma(a^T[Wh_i||Wh_j])}}{\sum_{k \in \mathcal{N}(i)} e^{\sigma(a^T[Wh_i||Wh_k])}}$$

# Graph neural networks

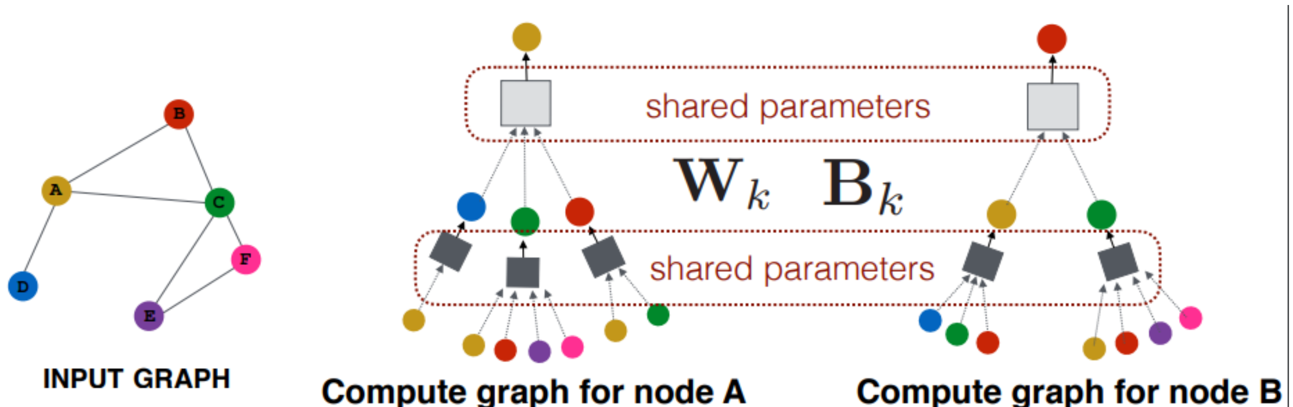
- Naïve approach
  - A fusion between the adjacency matrix and other node features -> input for an ANN
  - + easy
  - - a lot of parameters = O(no of nodes)
  - - doesn't work for graphs of different sizes
  - - not invariant to node ordering
- Graph convolutional networks
  - Node's neighbourhood defines a computational graph
  - An ML algorithm is used to learn how to transform and transmit the information across the nodes

# Graph convolutional networks

- step1: generate node embeddings based on local graph neighbourhood
  - Aggregate messages from neighbours and apply an ANN

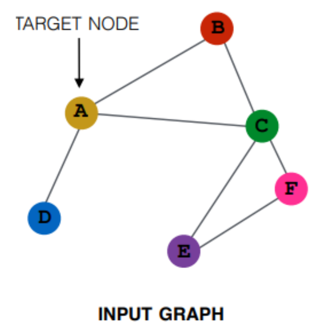


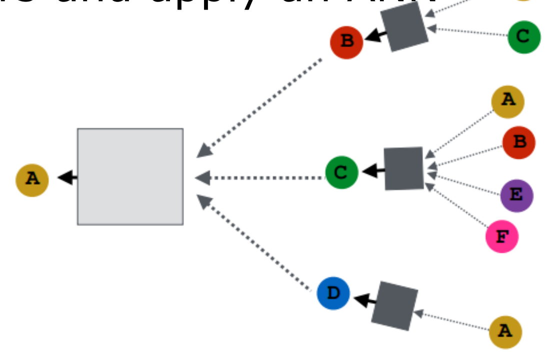

  - Every node defines a computation graph based on its neighbourhood



# Graph convolutional networks

- step1: setup for generating node embeddings based on local graph neighbourhood
  - Aggregate messages from neighbours and apply an ANN




    - Aggregation by an order invariant operator
      - Average (mean) or Max or Sum
    - ANN

Initial 0-th layer embeddings are equal to node features

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

Previous layer embedding of *v*

$$\mathbf{h}_v^k = \boxed{\sigma}\left(\mathbf{W}_k \boxed{\sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}} + \mathbf{B}_k \boxed{\mathbf{h}_v^{k-1}}\right), \quad \forall k \in \{1, \dots, K\}$$

$$\boxed{\mathbf{z}_v} = \mathbf{h}_v^K$$

Average of neighbor's previous layer embeddings

Embedding after K layers of neighborhood aggregation

Non-linearity (e.g., ReLU)

# Graph convolutional networks

- step2: prepare the model training
  - Parameters of the graph model: Wk, Bk
  - Classification weights θ
  - Loss function
    - E.g supervised binary classification task = safe or toxic drug

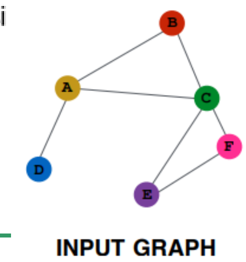
$$\mathcal{L} = \sum_{v \in V} \boxed{y_v} \log(\sigma(\boxed{\mathbf{z}_v^\top} \boxed{\boldsymbol{\theta}})) + (1 - \boxed{y_v}) \log(1 - \sigma(\boxed{\mathbf{z}_v^\top} \boxed{\boldsymbol{\theta}}))$$

**Encoder output:** node embedding

Classification weights

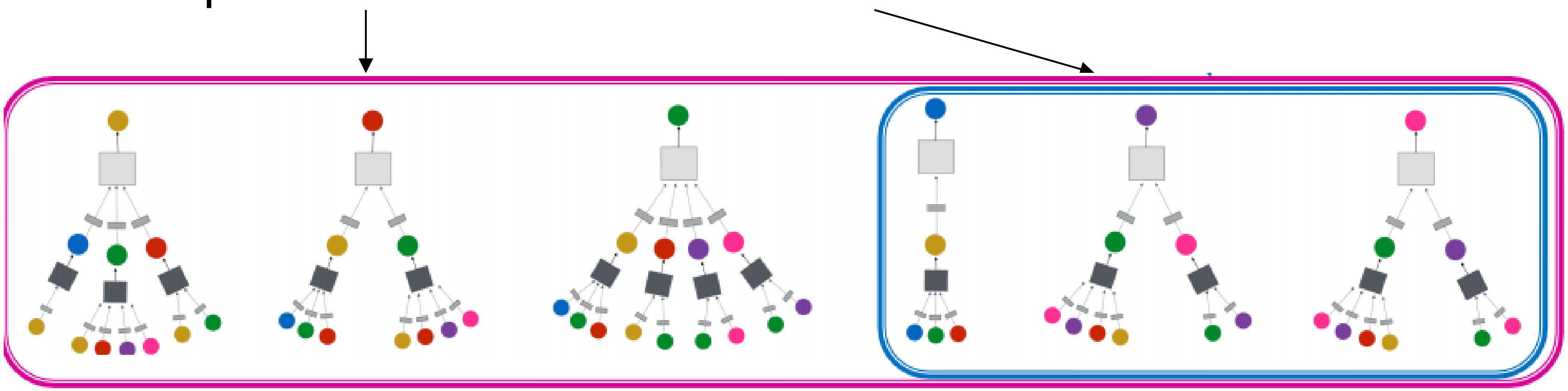
Node class label


    - E.g. unsupervised task – random walk optimization
      - See DeepWalk https://arxiv.org/pdf/1403.6652.pdf
      - See node2vec https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

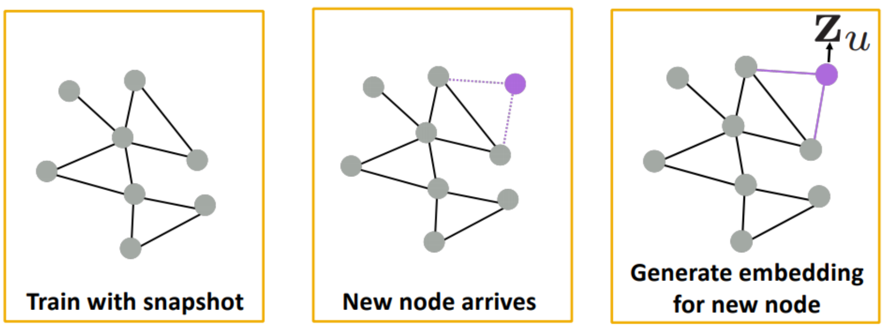



# Graph convolutional networks

- step3: train the model and test



- Step 4: generalize for new nodes / graphs



# Graph CNNs - applications

- Action Genome: Understanding Action with Spatio-Temporal Scene Graphs
  - http://actiongenome.org
  - Ji, Krishna et al. Action Genome: Actions as Compositions of Spatio-Temporal Scene Graphs, CVPR 2020
- RecSys
  - Pinterest - PinSage
    - https://medium.com/pinterest-engineering/pinsage-a-new-graph-convolutional-neural-network-for-web-scale-recommender-systems-88795a107f48
    - https://arxiv.org/pdf/1806.01973.pdf
  - Model and predict side effects of drug pairs
    - http://snap.stanford.edu/decagon/
    - https://arxiv.org/pdf/1802.00543.pdf
- Data generation
  - Drug discovery = Molecule generation (with high value of a given chemical property)
    - https://cs.stanford.edu/people/jure/pubs/gcpn-neurips18.pdf
    - https://github.com/bowenliu16/rl_graph_generation
    - https://www.cell.com/cell/pdf/S0092-8674(20)30102-1.pdf

- Additional information
  - Code example for GNN
    - https://colab.research.google.com/drive/1DIQm9rOx2mT1bZETEeVUThxcrP1RKqAn
  - Data
    - SNAP project
      - http://snap.stanford.edu/
    - Open Graph Benchmark
      - https://ogb.stanford.edu/
  - GNN and RecSys
    - https://github.com/yazdotai/graph-networks#tensorflow-implementations
    - https://next-nus.github.io/slides/tuto-cikm2019-public.pdf
  - Graph-based Deep Learning
    - https://github.com/naganandy/graph-based-deep-learning-literature
    - https://www.cs.mcgill.ca/~wlh/grl_book/
    - https://github.com/thunlp/GNNPapers

- Materials are considered from various sources like:
  - Fei-Fei Li's lecture about Graph Convolutions http://vision.stanford.edu/teaching/cs231n/slides/2020/lecture_18.pdf
  - Jure Leskovec's Lecture about Graph NNs https://web.stanford.edu/class/cs224w/
  - https://www.pyg.org/
  - Graph Representation Learning Book, by W. Hamilton https://www.cs.mcgill.ca/~wlh/grl_book/
  - Network Science, by A. Barabasi https://networksciencebook.com/
  - …