# Learning from Networks

## Computer Engineering

Fabio Vandin

October 10th, 2024

# Project

There is a **non-compulsory** project (= you can still get the grade of *30 e lode* without doing the project)

**Rules:**

- groups of 3(?) students
- you choose the topic/problem/data
- deadline to complete the project: January 16$^{th}$, 2025 (1 week before the first exam)
- max 8 points for the final grade

**Important: if you have a *great project* you want to work on *by yourself*, you need my approval before Tuesday, Oct. 22nd!**

# Project (2)

Three typical types of projects:

- **application project**: pick an application/dataset that interests you, and explore how best to apply what we learn to analyze the dataset
- **algorithmic project**: pick a problem or family of problems, and develop a new algorithm, or a novel variant of an existing algorithm, to solve it.
- **theoretical project**: prove some interesting/non-trivial properties of a new or an existing algorithm

Some projects will combine elements of applications, algorithms and theory.

**IMPORTANT**: you can also pick a topic related to the analysis of networks not covered by the course!

# Project (3)

Useful resources on the course website:

- **page "Datasets and repositories"**: links to datasets or collections of datasets, can be used for the project
- **page "Some interesting papers"**: can provide ideas on motivation, experimental analyses, algorithms, theoretical results, etc.

**Feel free to explore other resources!**

# Project (4)

**Deadlines (tentative):**

- **project proposal: deadline November 14th, 2024 (7 weeks)**
  - max 2 pages + references with: title, motivation (question, data), method (computational problem, algorithms,…), intended experiments (implementation, machine for experiments,...)
- **midterm report: deadline December 12th, 2024 (11 weeks)**
  - max 2 pages+references: what has changed from the project proposal?
- **final report: deadline January 16th, 2025**
  - max 6 pages + references with detailed description of what has been done + references + links to code (e.g., bitbucket/github repository)
  - **Required** : additional 1 page with detailed contribution of each member, including fraction of the work done by each member **[NOT INCLUDED? 0 POINTS]**

# Course Topics

Introduction and basic notions/algorithms (refresher)

Graph analytics and network features (e.g., centralities, clustering coefficient)

Network patterns and motifs

Network embeddings

Graph neural networks

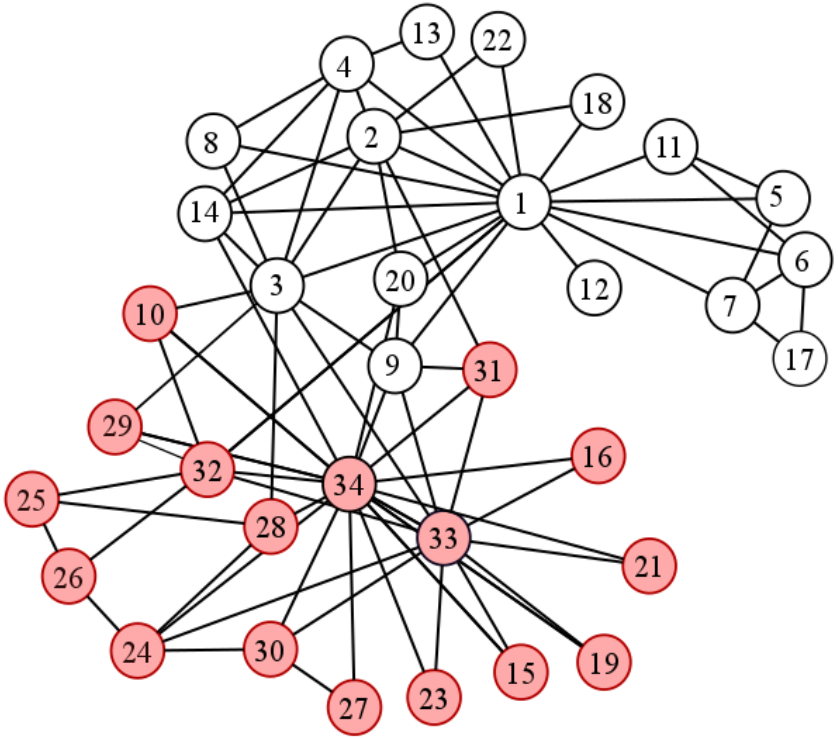
Network clustering

Advanced topics (e.g., temporal networks, uncertain graphs, polarization bubbles…)

# Graph analytics and network features

Some of the features that can be computed for:

- Nodes
- Graphs

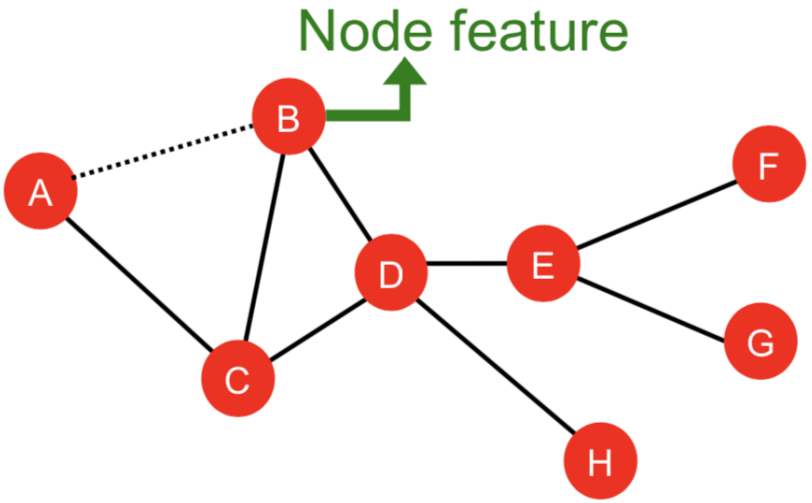


# Graph analytics and network features: node-level

Some of the most commonly used features for *nodes in a graph*

We will cover *centralities*:

- Closeness
- Betweenness
- PageRank (maybe)




We will see:

- definitions
- algorithms (and their analysis) for exact solution
- algorithms for approximate solutions: scale to large graphs

# Graph analytics and network features: node-level

Other node-level features

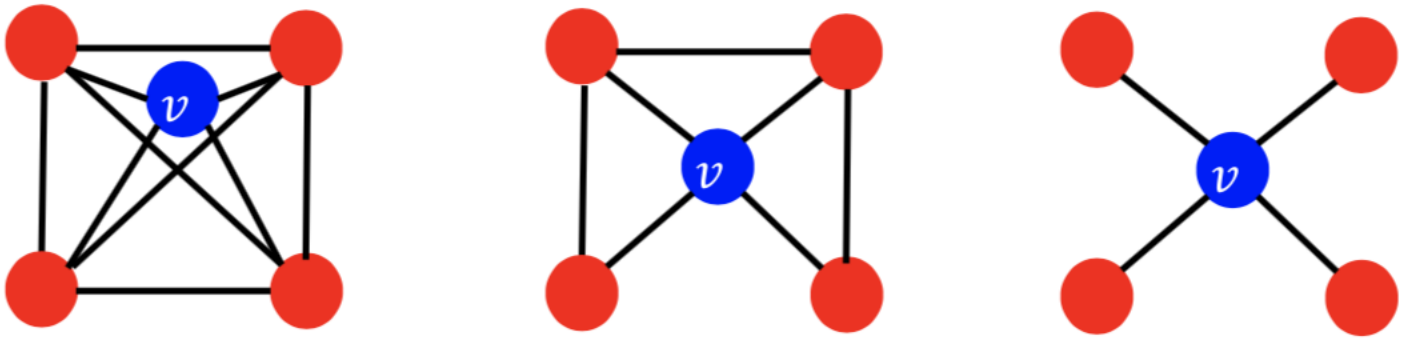
- clustering coefficient
- graphlets

# Clustering Coefficient

Measures how connected the neighbours of a node *v* are

$$e_v = \frac{\#(\text{edges among neighboring nodes})}{\binom{k_v}{2}} \in [0,1]$$

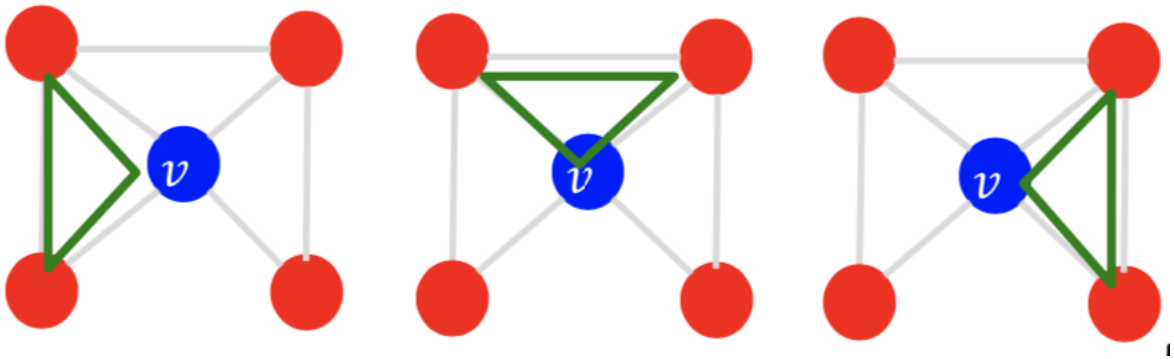
$\#(\text{node pairs among } k_v \text{ neighboring nodes})$



# Graphlets

Given node *v*, its **ego-network** is the network containing *v* and its neighbours

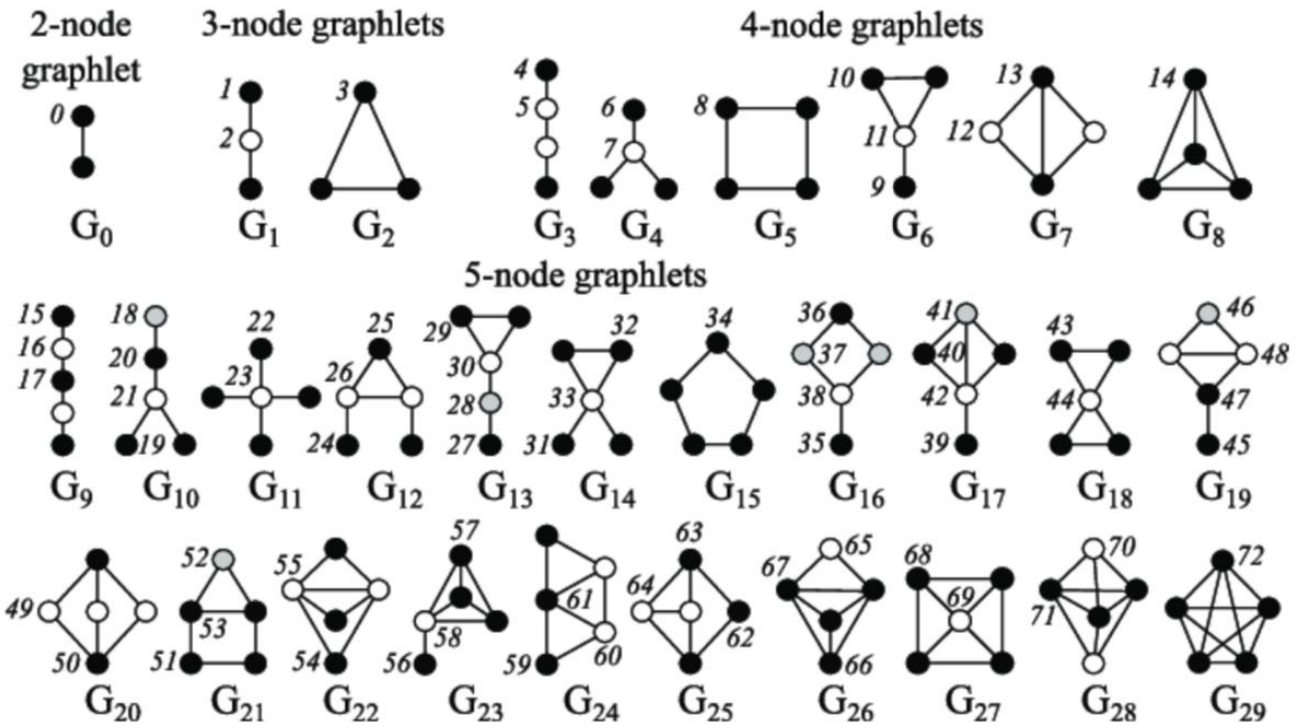
Clustering coefficient of *v* ≈ number of triangles in the **ego-network** of v



# Graphlets (continue)

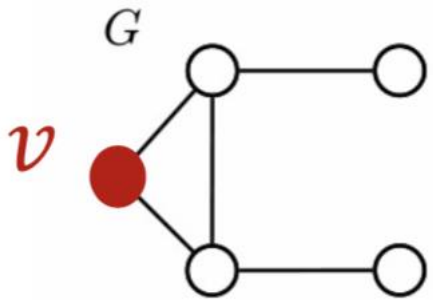
What about generalizing to connected non-isomorphic subgraphs?

connected non-isomorphic subgraph = **graphlet**

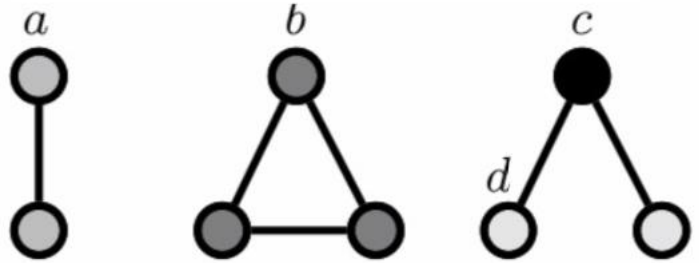


# Graphlets (continue)

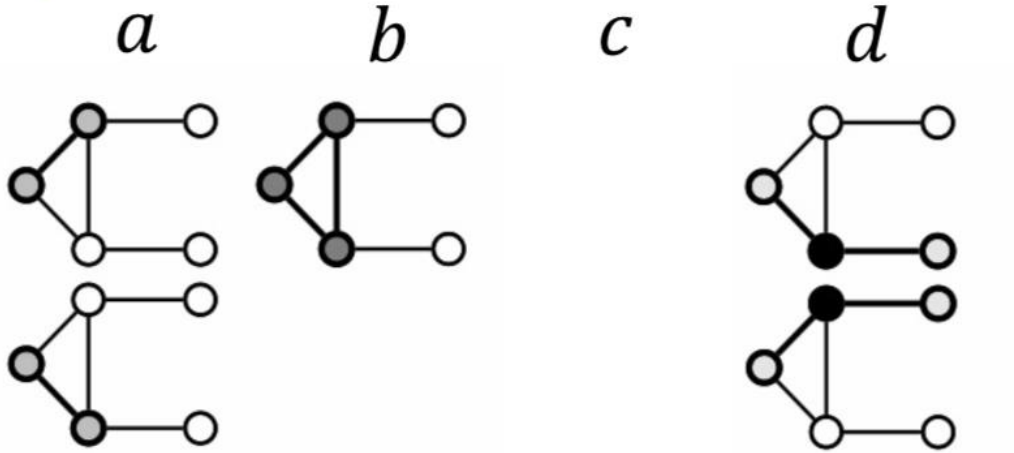
Graphlet Degree Vector: count vector of graphlets **rooted** at a given node




List of graphlets




Graphlet instances:




GDV of node $v$:

$a, b, c, d$

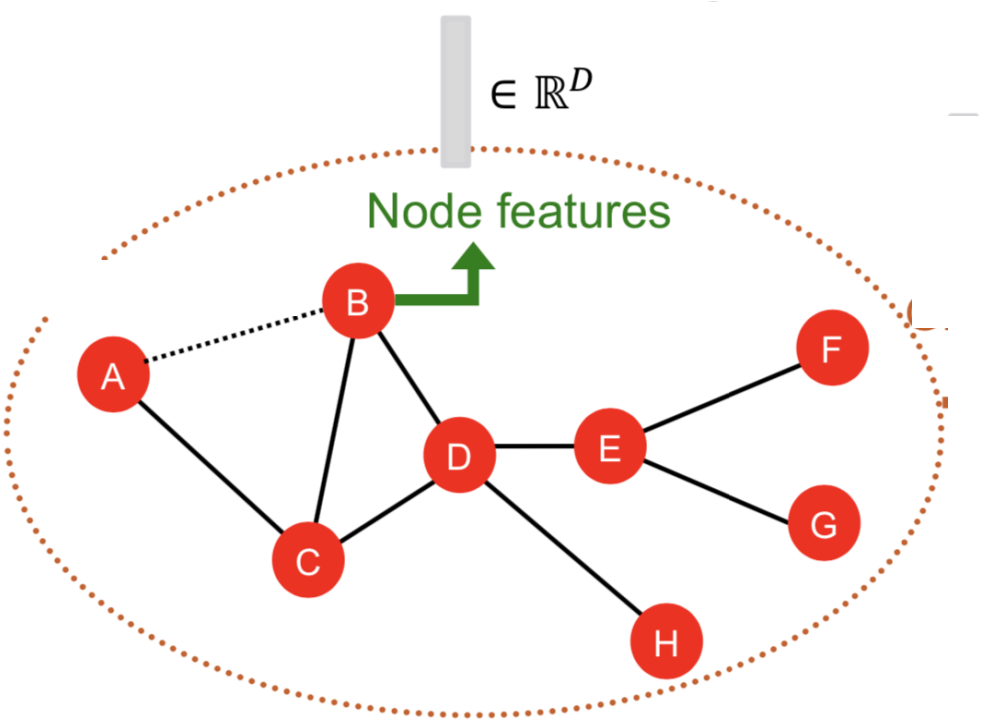
[2,1,0,2]

# Graph analytics and network features

Summary:

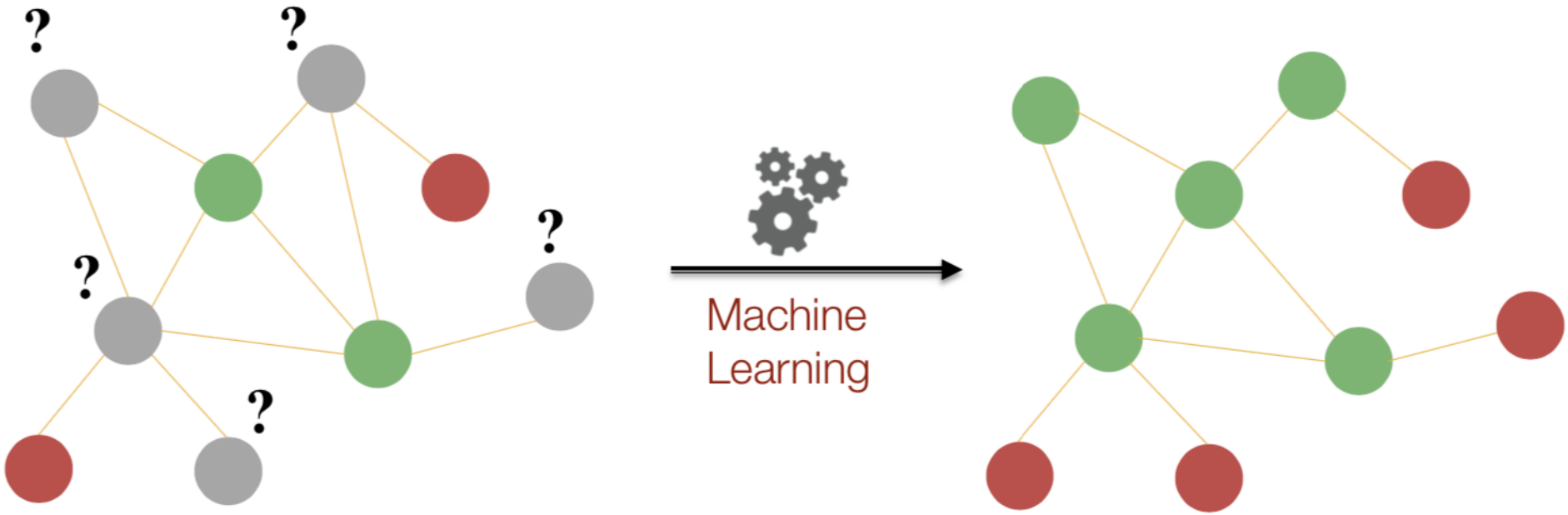
- Centralities
- Clustering Coefficient
- Graphlets

What could these be used for?



# Application of nodel-level features

Use them in a “traditional” ML pipeline:



# Project Proposal Example [ONLY MAIN POINTS!]

**Title**: Finding Significant Genes in Protein-Protein Interaction Networks with Node Features

**Motivation:**

- What are the most important genes in PPI networks according to different centrality measures? Do different measures agree?
- Data: PPI from public repository [DETAILS: number of nodes, edges, etc.]

**Method:**

- Problem: compute the centrality scores for all nodes in a PPI
- Algorithms: we are going to try to use an exact algorithm. If that does not work, we will use approximations…

**Intended experiments:**

- we will use the implementations available at [URL]. It provides exact methods and approximate methods
- machine for experiments: [DETAILS]
- experiments: compute centralities on the data; compare the rankings [DETAILS]

# Project Proposal Example

**Title**: Do Genes Scores in PPI and Disease Importance Agree?

**Motivation:**

- Do gene scores reflect the importance of genes in diseases?
- Data: human PPI from public repository; list of important genes for a disease from public repository DETAILS: number of nodes, edges, etc.]

**Method:**

- Problem: compute some of the node features for all nodes in a PPI
- Algorithms: we are going to try to use an exact algorithm. If that does not work, we will use approximations…

**Intended experiments:**

- we will use the implementations available at [URL]. It provides exact methods and approximate methods
- machine for experiments: [DETAILS]
- experiments: compute gene scores on the data; compare the ranking with the list of important genes for a disease [DETAILS: how to perform the comparison?]

# Project Proposal Example

**Title**: Comparison of Node Scores in Different Social Networks

**Motivation:**

- Do different social networks have similar distributions of node scores?
- Data: the 3 social networks from public repositories [DETAILS: number of nodes, edges, etc.]
- **Method:**
- Problem: compute the following node features for all nodes in all social networks [DETAILS]. Since graphs are large, we may restrict to simpler features [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute features; compare the distributions in the different networks [DETAILS: how to perform the comparison?]

# Project Proposal Example

**Title**: Are Network Features Useful for Nodes Classification in Social Networks?

**Motivation:**

- We found this paper that uses external nodes features to predict the role of a node in a network.
- Does the inclusion of network features (node level) improve the classification performance?
- Data: this social networks and external node features publicly available from the paper [DETAILS: number of nodes, edges, external features, etc.]
- **Method:**
- Problem: compute the following node-level network features for all nodes in the social networks [DETAILS]. Run ML method used in the paper with and without such features
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute features; compare the accuracy of the models with and without node-level network features [DETAILS]

# Graph analytics and network features: graph-level

Want features that characterize the structure of an entire graph

Graph-level feature:

- Diameter
- Clustering coefficient of the whole graph
- Graphlet/ counts over the all graph

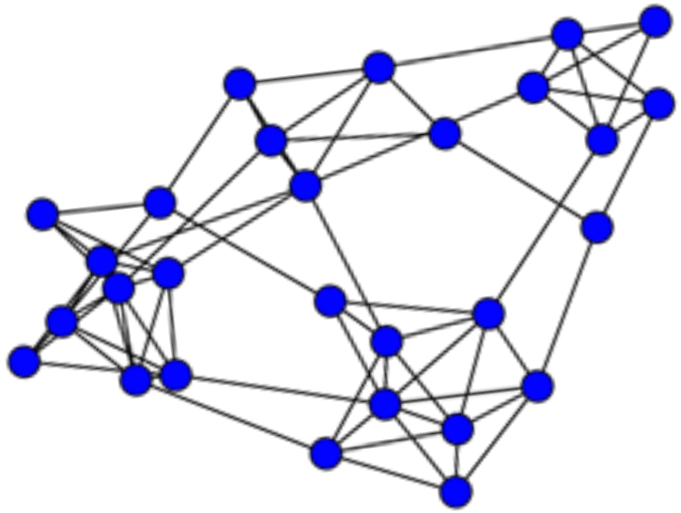
# Clustering Coefficient

Graph $G=(V,E)$

$t(G)$ = number of triangles in $G$

*bin(n,3) = binomial coefficient*

Clustering coefficient of $G$ = $t(G)$ / *bin(n,3)*

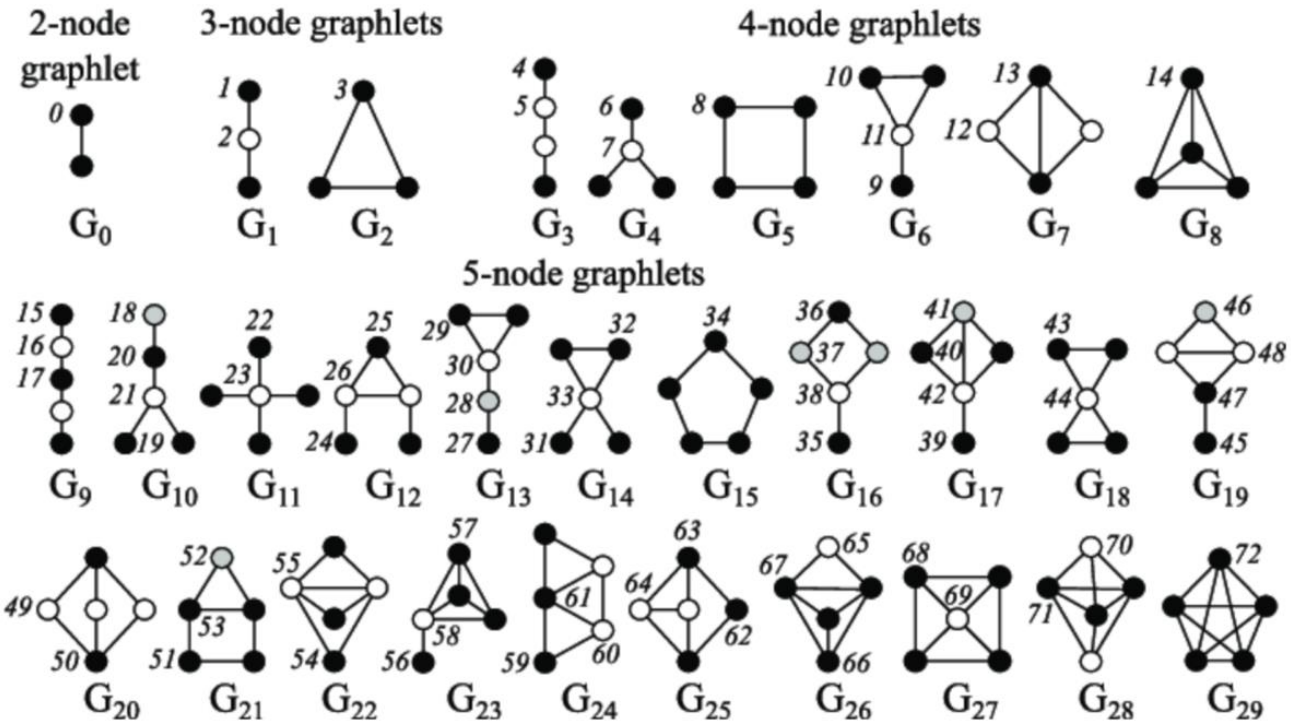


Exact computation?

Approximate computation?

# Graphlet Counts

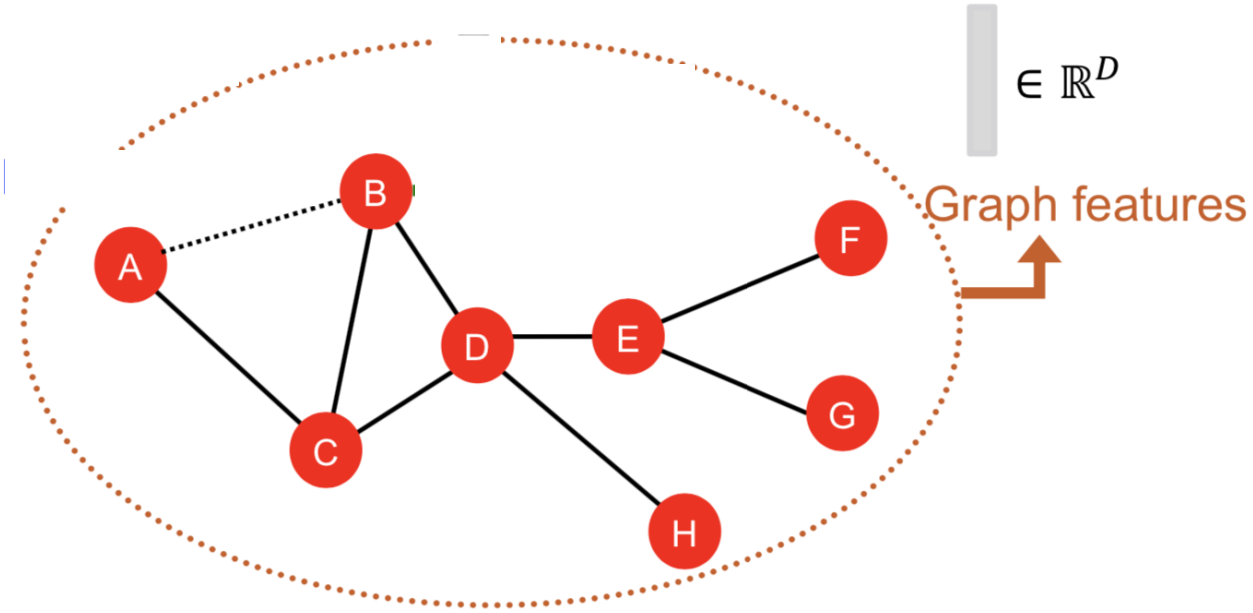
Count the number of graphlets in the whole graph




Exact computation?

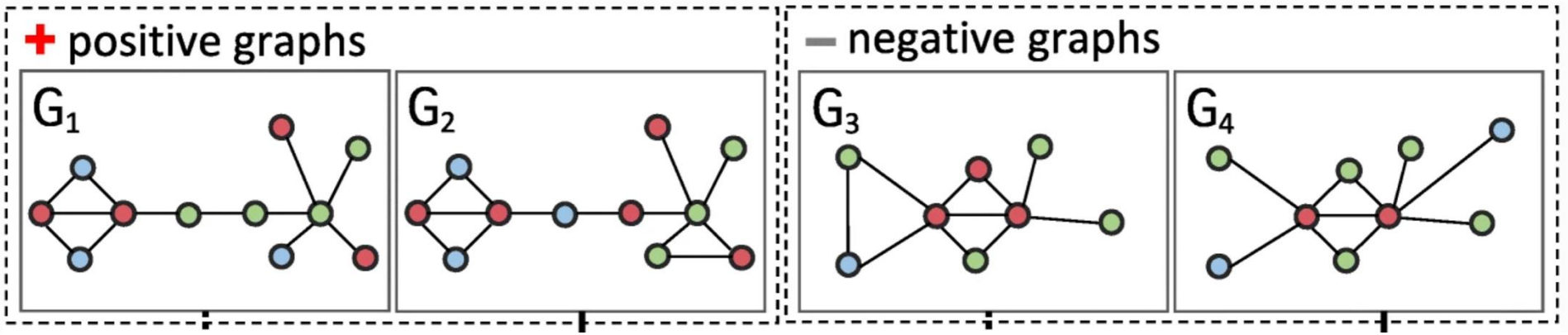
Approximate computation?

# Graph analytics and network features: graph level




What are they useful for?

# Graph analytics and network features: graph level




- Compute vector of features for each graph
- Use vectors as input to ML method to learn a prediction model
- Example: prediction of molecular activity

# Project Proposal Example

**Title**: Are Graph Features Useful for Molecular Graph Classification?

**Motivation:**

- We found this paper that uses external graph features to predict the activity of a molecule represented as a graph
- Does the inclusion of network features (graph level) improve the classification performance?
- Data: these graphs and external graph features publicly available from the paper [DETAILS: number of graphs, sizes, external features, etc.]

**Method:**

- Problem: compute the following graph-level features for all graphs in input [DETAILS]. Run ML method used in the paper with and without such features
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute features; compare the accuracy of the models with and without graph-level features [DETAILS]

# Project Proposal Example

**Title**: Do Graph-Level Features Capture the Domain of a Network?

**Motivation:**

- We want to understand if networks from different domains (biology, social networks, knowledge networks) have similar graph-level features
- Data: these graphs from publicly available datasets [DETAILS]

**Method:**

- Problem: compute the following graph-level features for all graphs in input [DETAILS]. Since graphs are large we may restrict to following features: [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute features; compare the features for networks in the different domains [DETAILS]

# Project Proposal Example

**Title**: Evaluation of Sampling Algorithms to Compute Graphlet Counts

**Motivation:**

- The following 2 recent papers proposes new algorithms to approximate graphlet counts in a network
- Data: these graphs from publicly available datasets [DETAILS]

**Method:**

- Problem: compute the approximations using the 2 algorithms on the data and compare performance
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: run both algorithms; compare run-time and quality of approximation [DETAILS]

# Network Patterns and Motifs

Graphlets/small patterns may provide *interesting* information about a network

**Common analysis**: find patterns that are not explained by simple properties of the network, e.g. the nodes degree

**motifs**

Corresponds to find *statistically significant subgraph patterns: e.g.,* subgraphs that appear in a graph *more than expected by chance*

Significance is reflected by a score (e.g., Z-score) or a probability (*p*-value)

# Network Patterns and Motifs (continue)

Basic definitions

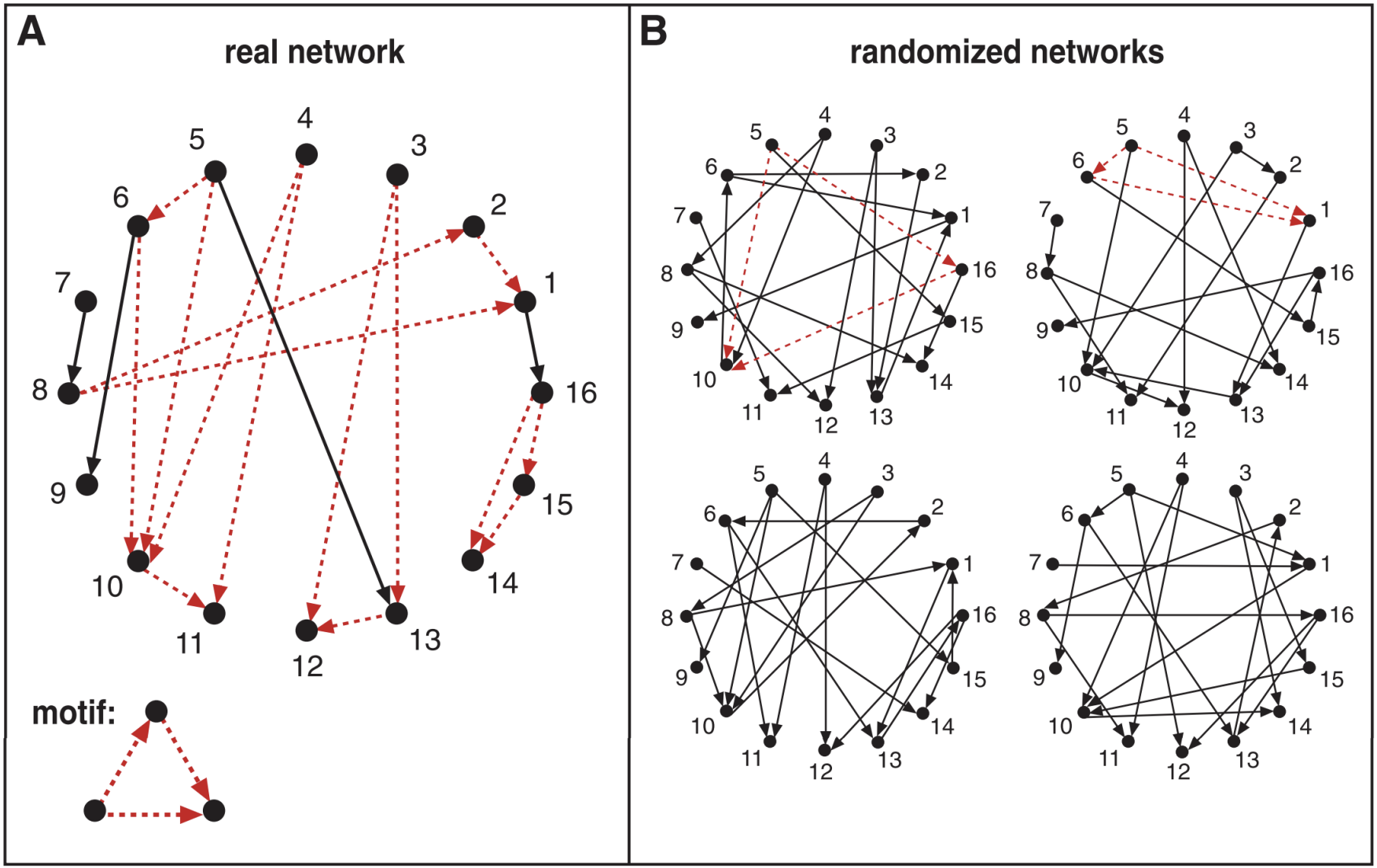
Statistical significance: basics

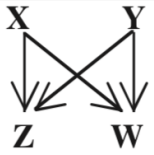
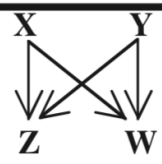
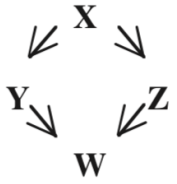
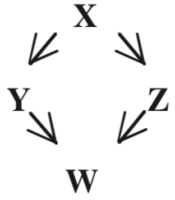
Permutation approaches (*how to generate a random network?*)

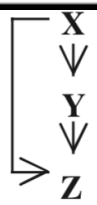
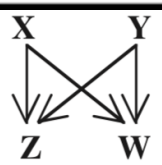
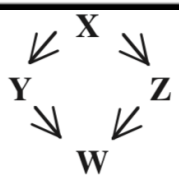
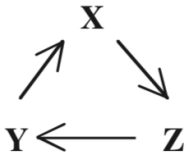
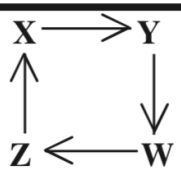
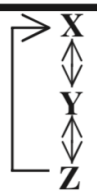
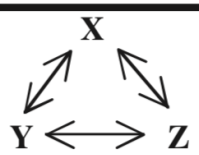
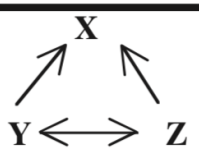
sembra interessante

# Statistically Significant Motifs: Example

[Milo et al. *Network Motifs: Simple Building Blocks of Complex Networks*, Science, 2002]



| Network | Nodes | Edges | $N_{real}$ | $N_{rand} \pm SD$ | $Z$ score | $N_{real}$ | $N_{rand} \pm SD$ | Z score | $N_{real}$ | $N_{rand} \pm SD$ | $Z$ score |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Gene regulation (transcription)** | | |   | | **Feed-forward loop** |   | | **Bi-fan** | | | |
| *E. coli* | 424 | 519 | 40 | 7 ± 3 | 10 | 203 | 47 ± 12 | 13 | | | |
| *S. cerevisiae** | 685 | 1,052 | 70 | 11 ± 4 | 14 | 1812 | 300 ± 40 | 41 | | | |
| **Neurons** | | |   | | **Feed-forward loop** |   | | **Bi-fan** |   | | **Bi-parallel** |
| *C. elegans*† | 252 | 509 | 125 | 90 ± 10 | 3.7 | 127 | 55 ± 13 | 5.3 | 227 | 35 ± 10 | 20 |
| **Food webs** | | |   | | **Three chain** |   | | **Bi-parallel** | | | |
| Little Rock | 92 | 984 | 3219 | 3120 ± 50 | 2.1 | 7295 | 2220 ± 210 | 25 | | | |
| Ythan | 83 | 391 | 1182 | 1020 ± 20 | 7.2 | 1357 | 230 ± 50 | 23 | | | |
| St. Martin | 42 | 205 | 469 | 450 ± 10 | NS | 382 | 130 ± 20 | 12 | | | |
| Chesapeake | 31 | 67 | 80 | 82 ± 4 | NS | 26 | 5 ± 2 | 8 | | | |
| Coachella | 29 | 243 | 279 | 235 ± 12 | 3.6 | 181 | 80 ± 20 | 5 | | | |
| Skipwith | 25 | 189 | 184 | 150 ± 7 | 5.5 | 397 | 80 ± 25 | 13 | | | |
| B. Brook | 25 | 104 | 181 | 130 ± 7 | 7.4 | 267 | 30 ± 7 | 32 | | | |

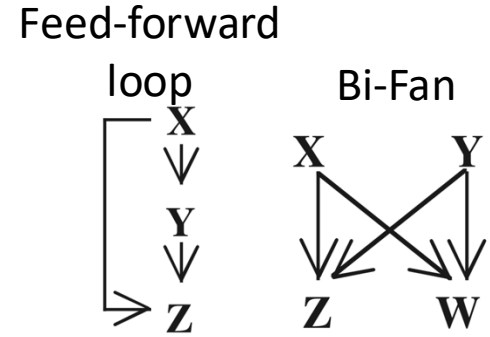
| Network | Nodes | Edges | $N_{real}$ | $N_{rand} \pm SD$ | $Z$ score | $N_{real}$ | $N_{rand} \pm SD$ | Z score | $N_{real}$ | $N_{rand} \pm SD$ | $Z$ score |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Electronic circuits (forward logic chips)** | | |   | | **Feed-forward loop** |   | | **Bi-fan** |   | | **Bi-parallel** |
| s15850 | 10,383 | 14,240 | 424 | 2 ± 2 | 285 | 1040 | 1 ± 1 | 1200 | 480 | 2 ± 1 | 335 |
| s38584 | 20,717 | 34,204 | 413 | 10 ± 3 | 120 | 1739 | 6 ± 2 | 800 | 711 | 9 ± 2 | 320 |
| s38417 | 23,843 | 33,661 | 612 | 3 ± 2 | 400 | 2404 | 1 ± 1 | 2550 | 531 | 2 ± 2 | 340 |
| s9234 | 5,844 | 8,197 | 211 | 2 ± 1 | 140 | 754 | 1 ± 1 | 1050 | 209 | 1 ± 1 | 200 |
| s13207 | 8,651 | 11,831 | 403 | 2 ± 1 | 225 | 4445 | 1 ± 1 | 4950 | 264 | 2 ± 1 | 200 |
| **Electronic circuits (digital fractional multipliers)** | | |   | | **Three-node feedback loop** |   | | **Bi-fan** |   | | **Four-node feedback loop** |
| s208 | 122 | 189 | 10 | 1 ± 1 | 9 | 4 | 1 ± 1 | 3.8 | 5 | 1 ± 1 | 5 |
| s420 | 252 | 399 | 20 | 1 ± 1 | 18 | 10 | 1 ± 1 | 10 | 11 | 1 ± 1 | 11 |
| s838‡ | 512 | 819 | 40 | 1 ± 1 | 38 | 22 | 1 ± 1 | 20 | 23 | 1 ± 1 | 25 |
| **World Wide Web** | | |   | | **Feedback with two mutual dyads** |   | | **Fully connected triad** |   | | **Uplinked mutual dyad** |
| nd.edu§ | 325,729 | 1.46e6 | 1.1e5 | 2e3 ± 1e2 | 800 | 6.8e6 | 5e4±4e2 | 15,000 | 1.2e6 | 1e4 ± 2e2 | 5000 |

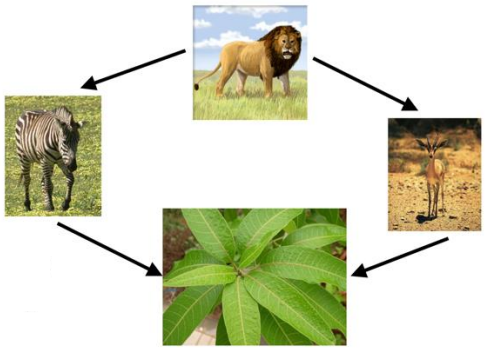
# Some Observations

Networks of neurons and gene networks contain similar motifs:

Feed-forward loops and bi-fan structures

Both are information processing networks with sensory and acting components



Food webs have parallel loops:

Prey of a particular predator share prey

WWW network has bidirectional links

Design that allows the shortest path between sets of related pages

# Project Proposal Example

**Title**: Do Networks from Different Domains have Different Motifs?

**Motivation:**

- We want to understand if networks from different domains (biology, social networks, knowledge networks) have similar motifs
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: compute the motifs for all networks in input [DETAILS]. Since networks are large we may restrict to motifs up to size [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute motifs; compare the motifs for networks in the different domains [DETAILS]

# Project Proposal Example

**Title**: What are the Motifs in Our Favorite Social Network?

**Motivation:**

- we want to understand what motifs are more surprising in our favorite social network
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: compute the motifs for the network [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: compute motifs; try to understand what these motifs represent [DETAILS]

# Network Patterns and Motifs (continue)

Related problem:

- given a *collection of networks* find *frequent* patterns/graphlets, i.e., that appear in a large fraction of the networks

Exact approaches?

Approximate approaches?

Statistically significant patterns?

# Project Proposal Example

**Title**: Evaluation of Algorithms to Find Significant Subgraphs in a Collection of Networks

**Motivation:**

- We want to compare these 2 methods from recent papers that find significant subgraphs
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

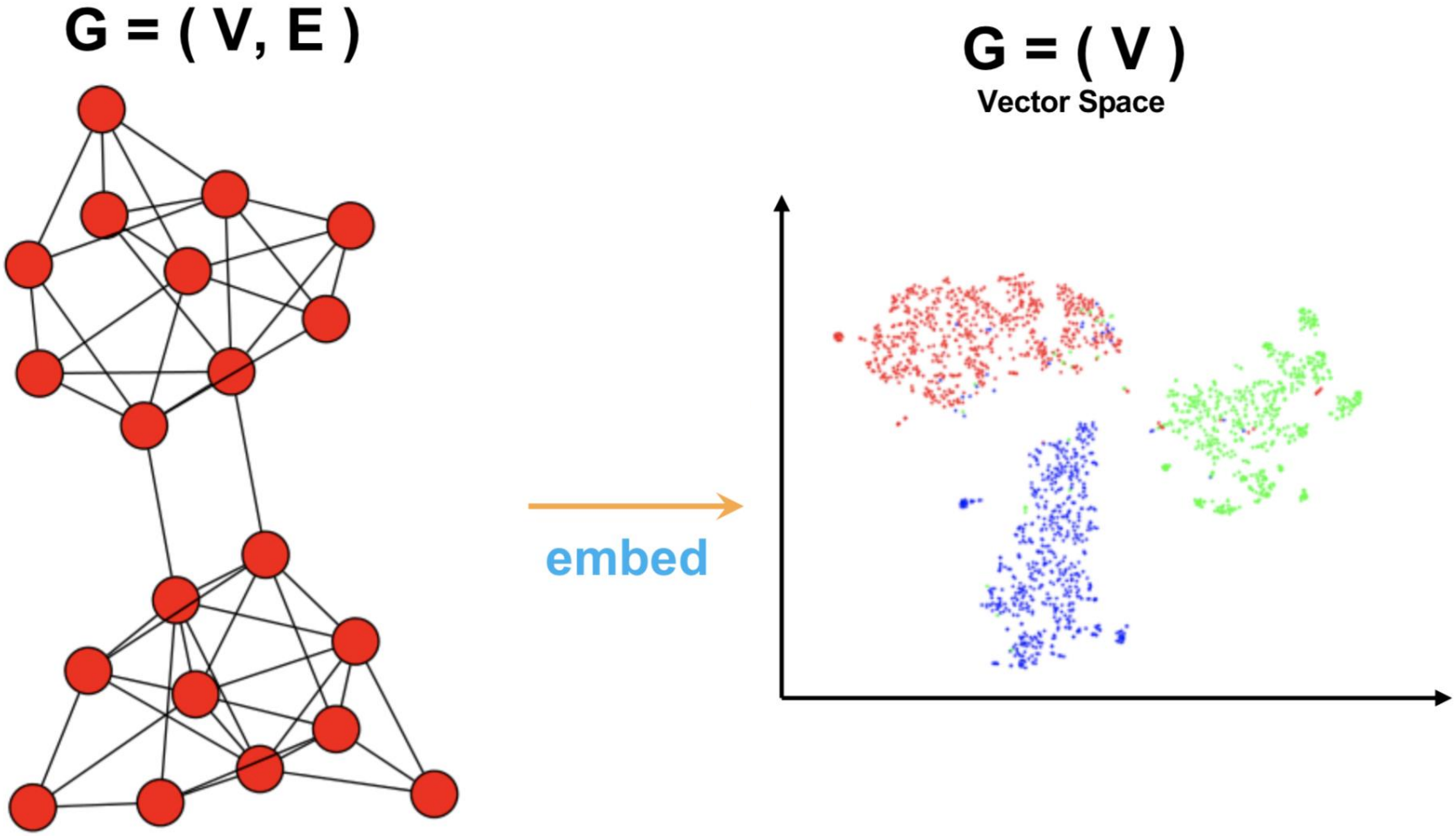
- Problem: [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

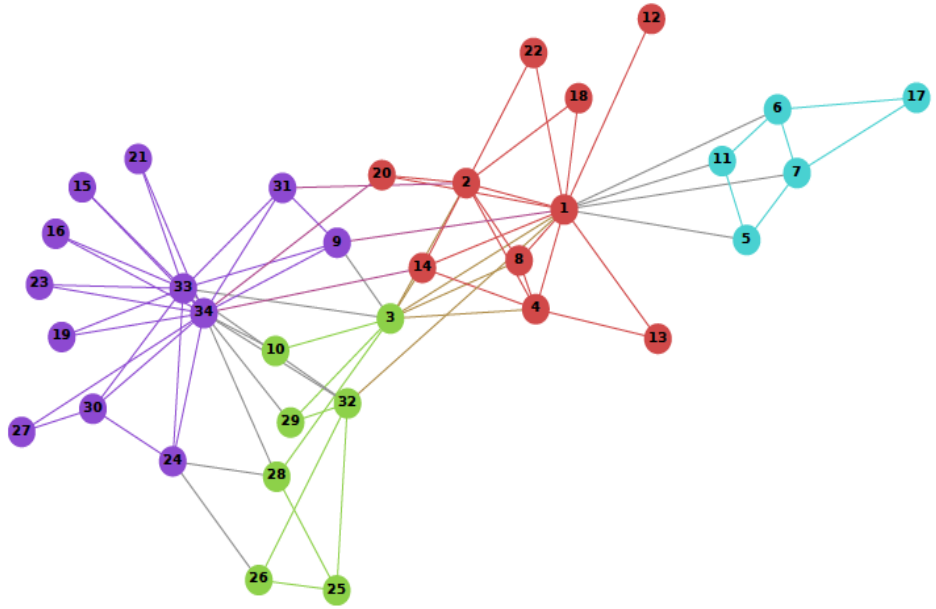
# Graph Embeddings

How can we meaningfully represent nodes as vectors?

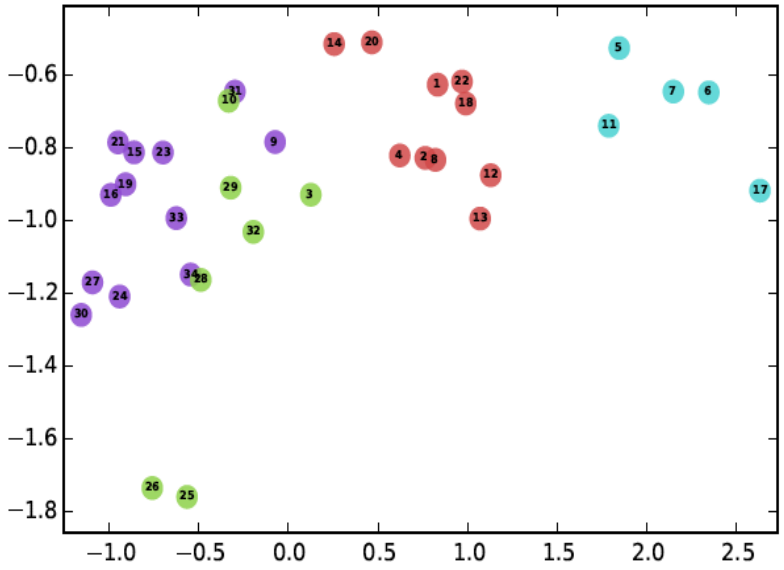


# Embedding Nodes

Input



Output

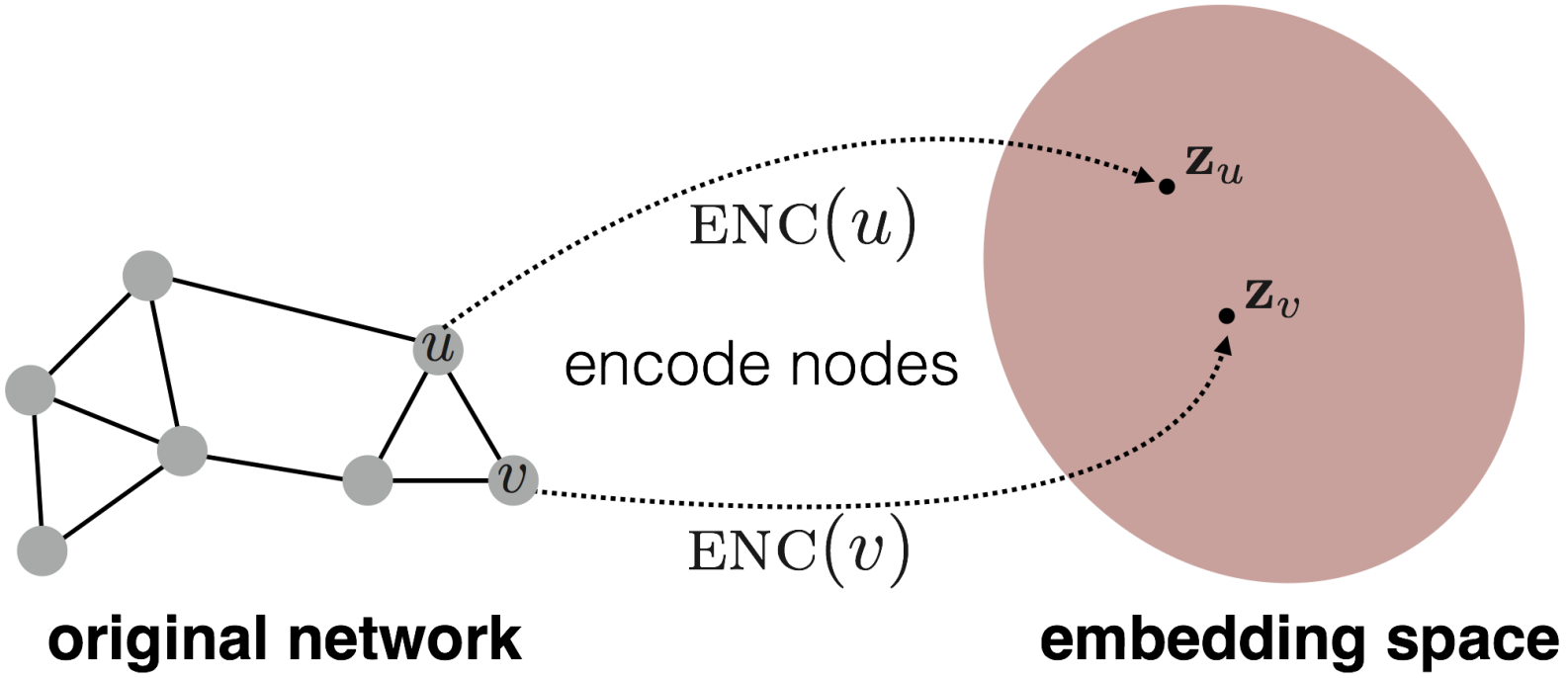



Colors: results of clustering (not available in input)

**Intuition**: find embedding of nodes to d-dimensions so that “similar” nodes in the graph have embeddings that are close together.

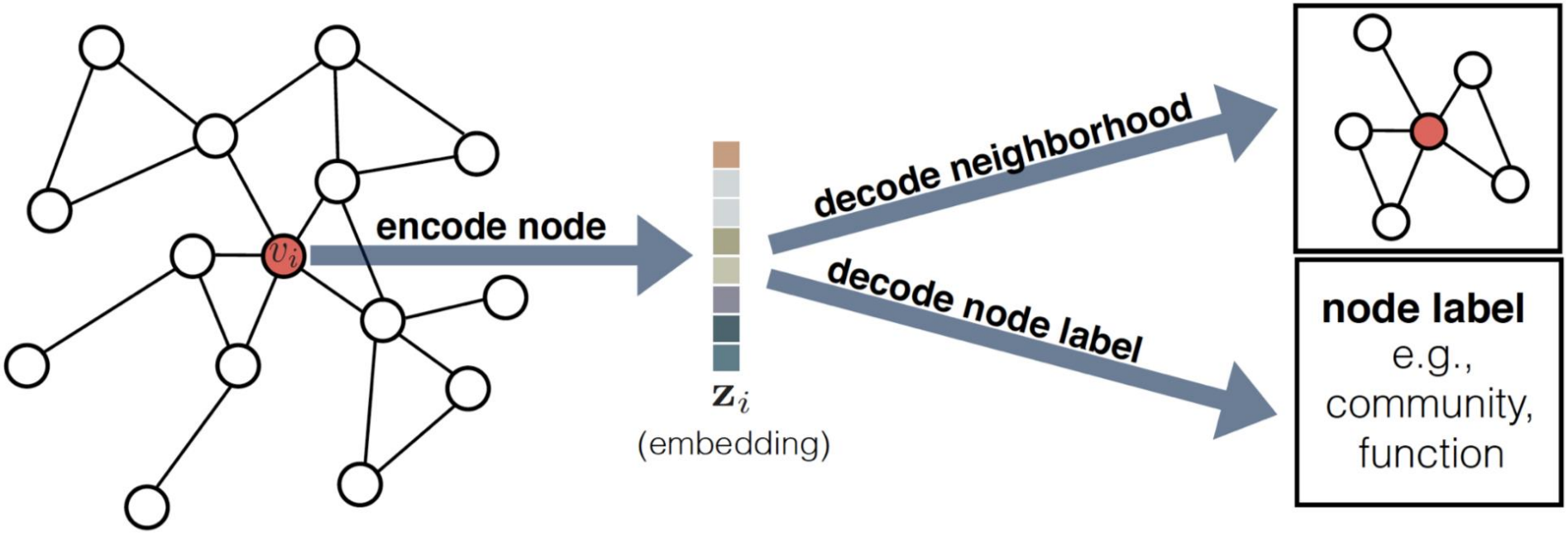
# Embedding Nodes

**Goal**: *encode* nodes so that similarity in the embedding space (e.g., dot product) approximates similarity in the original network.



# A Useful Framework

Encoder-Decoder Approach:




**Intuition**: joint optimization of encoder and decoder

system learns to compress information about graph structure into (low-dimensional) embedding space

# Encoder-Decoder Approach

Formalization

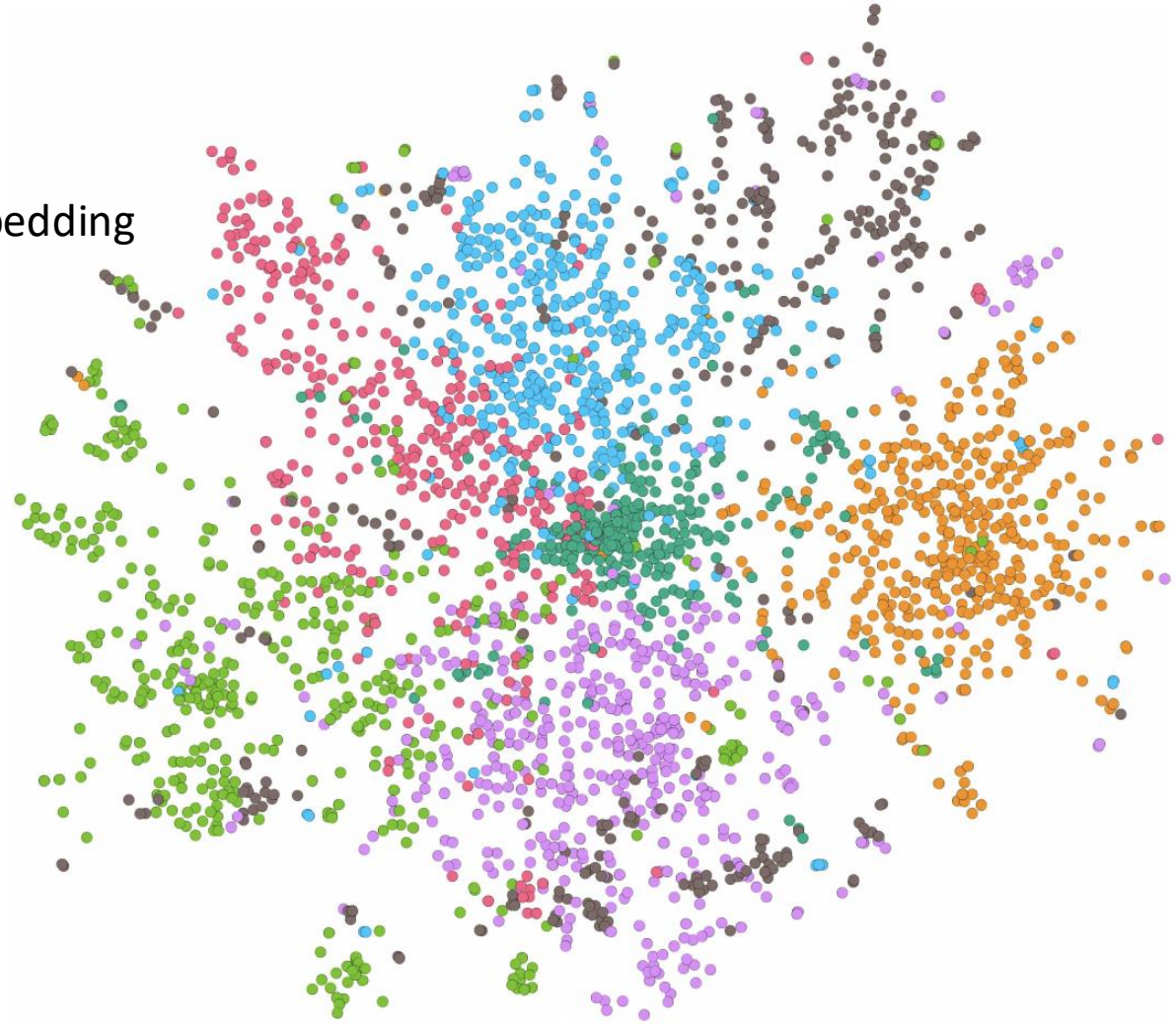
Methods:

- adiacency-based similarity approaches
- random walks-based approaches (node2vec)

# Example: random walk-based approach

CORA dataset
dataset of academic papers of seven different classes
graph = citation relations between the papers (2708 nodes, 5429 edges)

2D visualization of embedding
(t-SNE algorithm)
color = paper class



# Project Proposal Example

**Title**: Evaluation of Node Embedding Methods

**Motivation:**

- We want to compare these 2 methods for node embeddings
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

# Graph Neural Networks: from *Shallow* to *Deep Embeddings*

We have seen *shallow embeddings:*

the embedding of a node is the column of a matrix **Z**

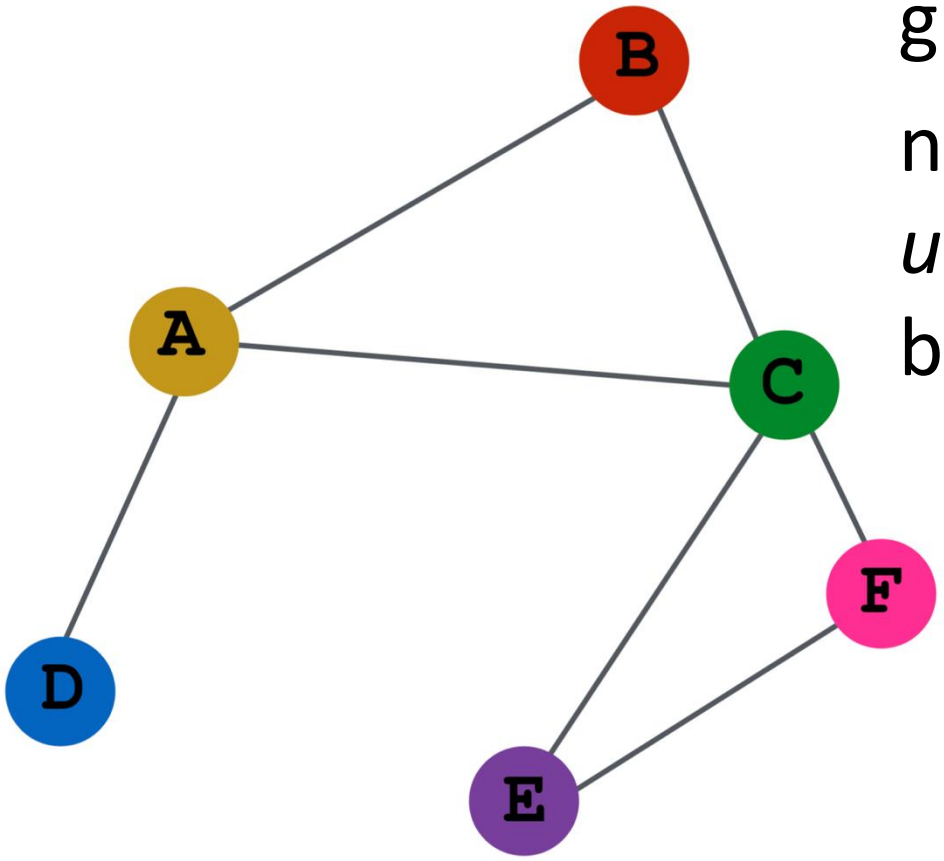
**Limitations:**

- what is the embedding of a new node?
- how can we incorporate *nodes features*?

***Deep embeddings*:**

the encoder function is a complex function that depends on the graph  graph neural networks

# Topic: *Graph Neural Networks*



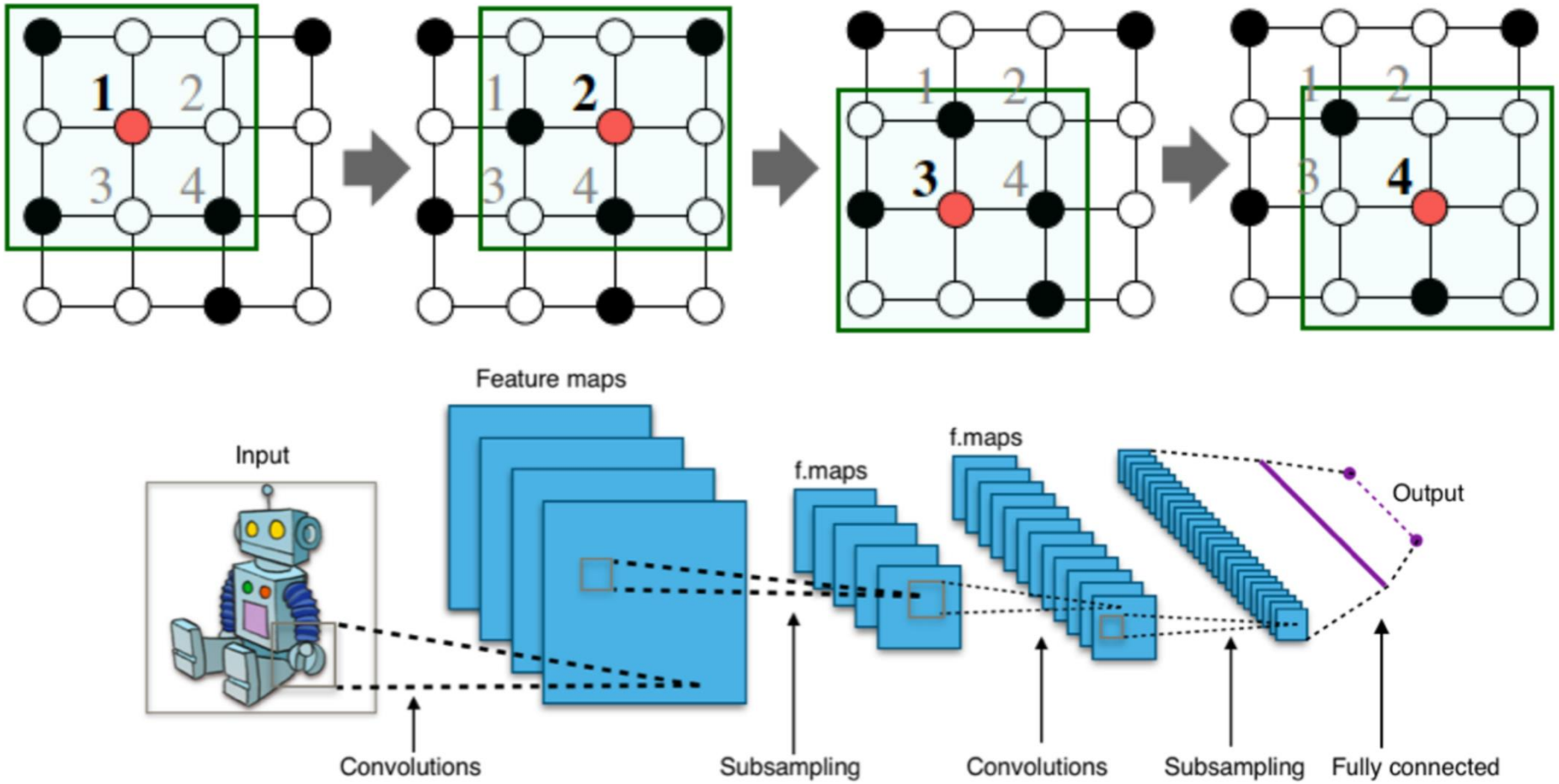

**Input**:

graph G=(V,E)

nodes features: for node *u*, its features are given by vector $\mathbf{x}_u$

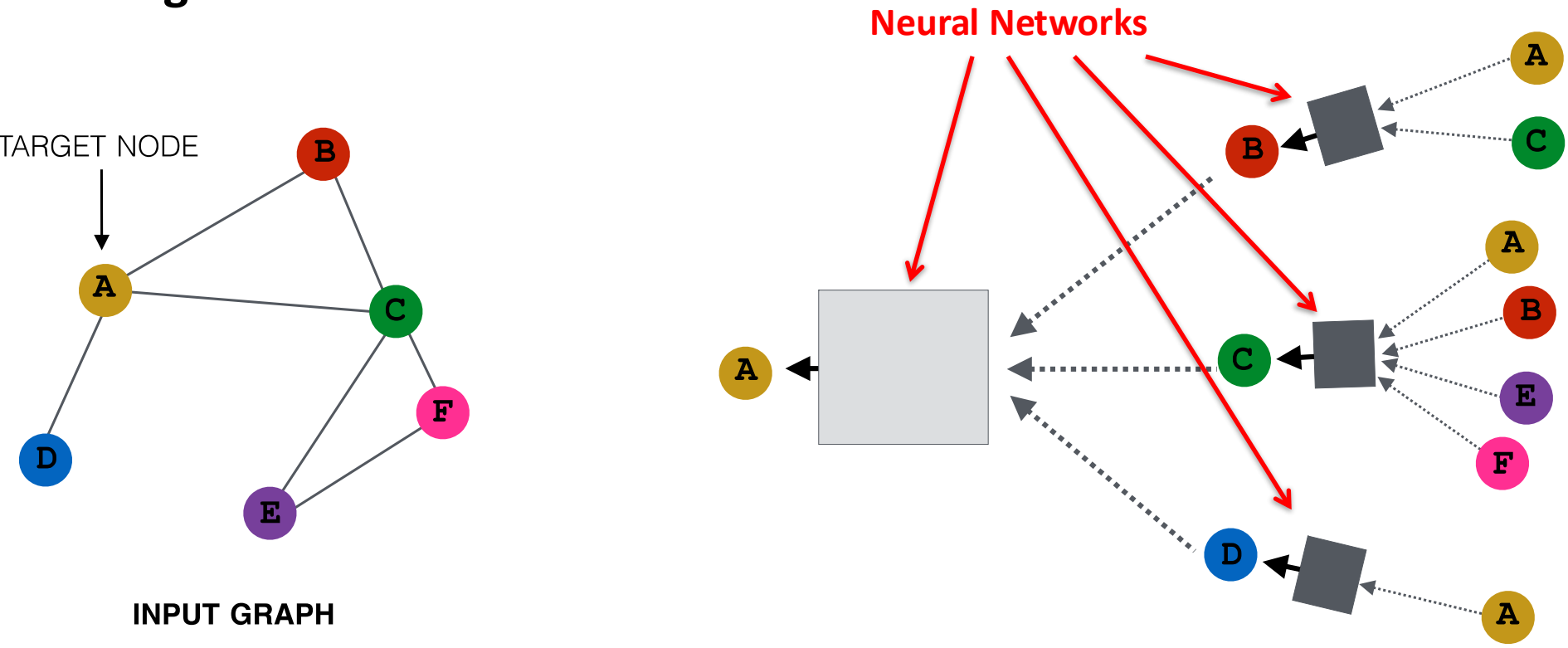
How do we build a neural network to obtain (node) embeddings considering the network structure?

# ML for Images: Convolutional Neural Networks (CNNs)



# Idea: Local Neighborhood Aggregation

**Key idea:** Generate node embeddings based on **local network neighborhoods**




**Intuition:** Nodes aggregate information from their neighbours (and themselves) using neural networks

# Graph Neural Networks (GNNs)

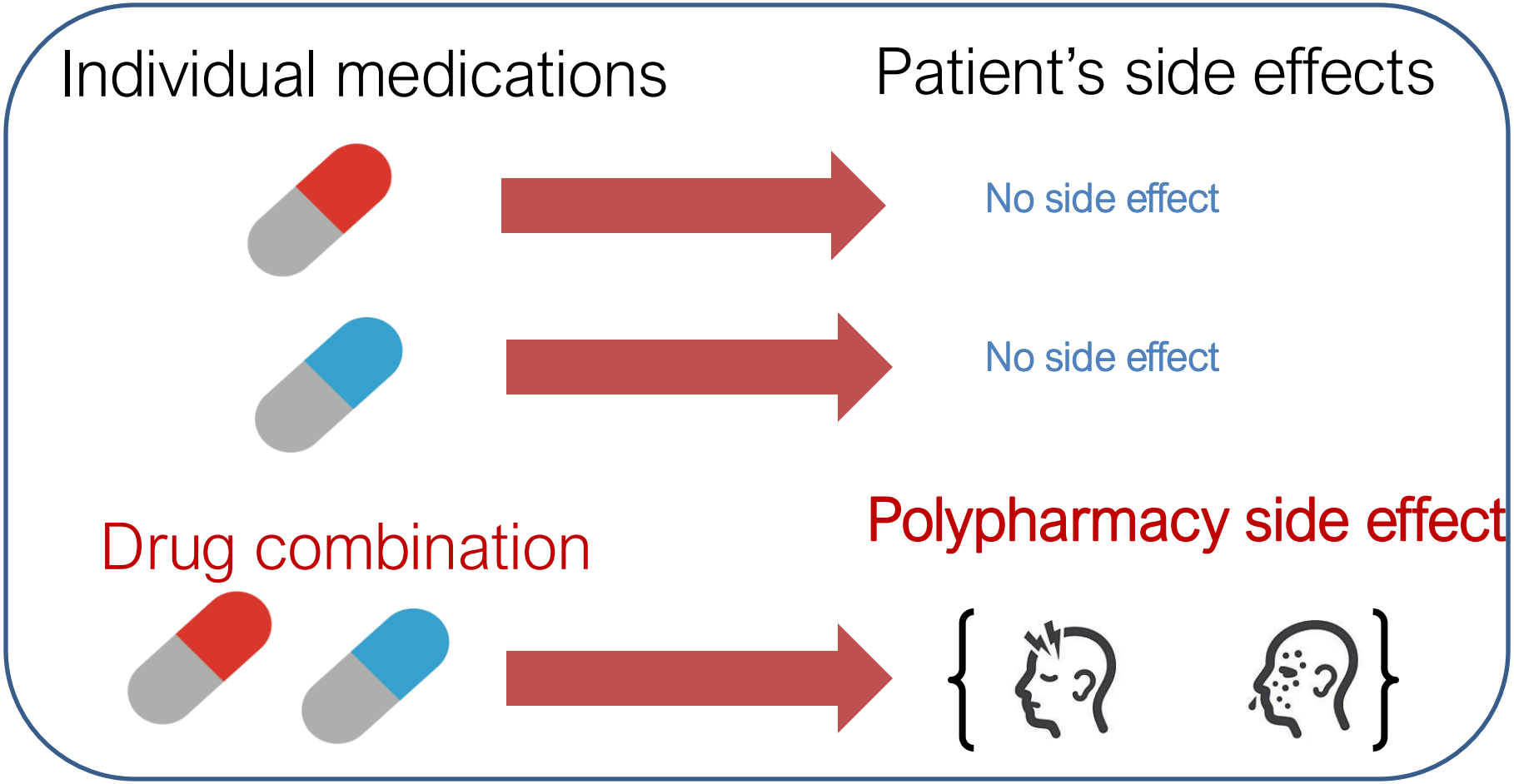
Neural Message Passing framework

Basic GNN

Generalized neighborhood aggregation

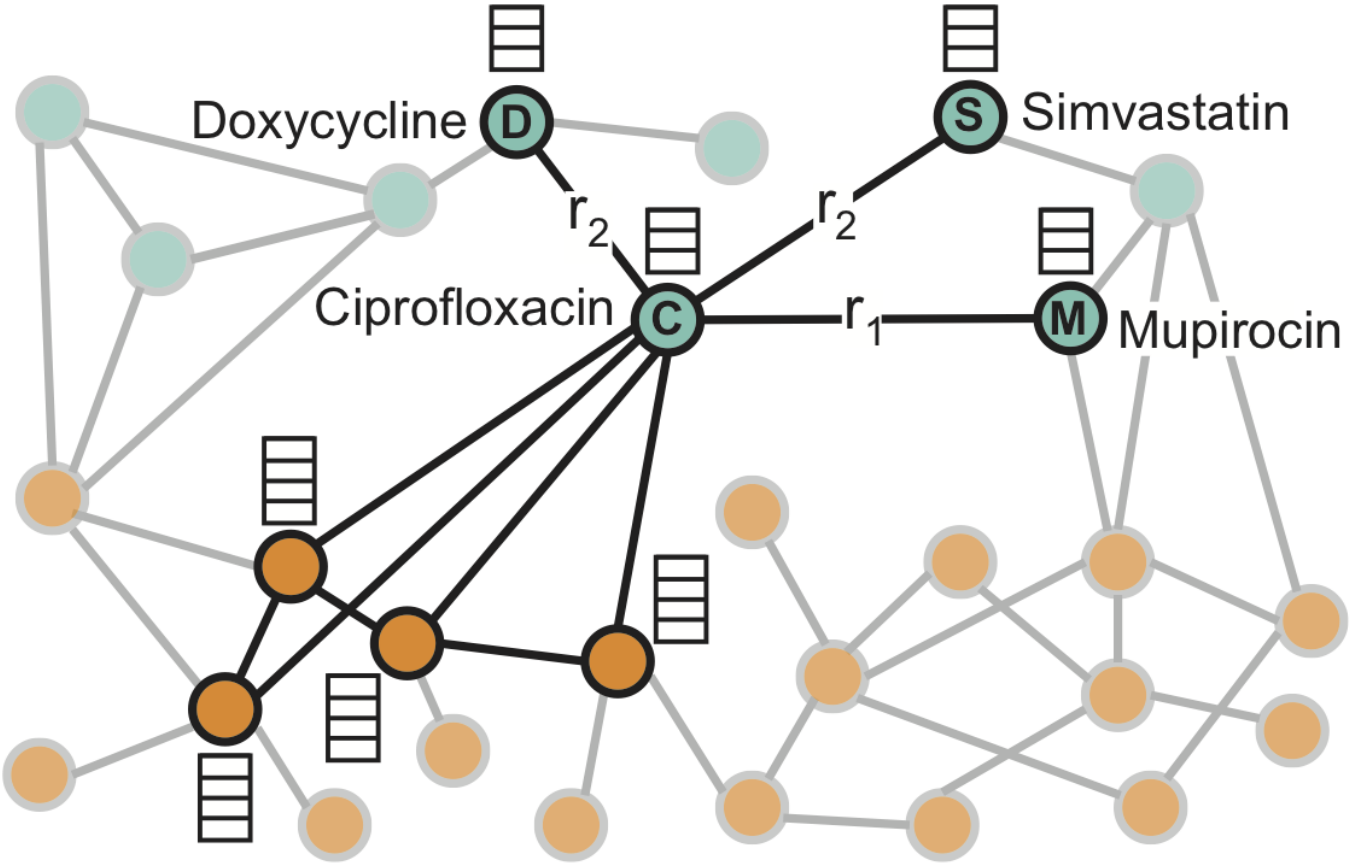
# Application: Polypharmacy Side Effects

**Goal**: Predict side effects of taking multiple drugs.




[Zitnik et al. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 2018.]

# Data: Heterogeneous Graphs



# Prediction Performance

| Approach | AUROC | AUPRC | AP@50 |
|---|---|---|---|
| *Decagon* | 0.872 | 0.832 | 0.803 |
| RESCAL tensor factorization | 0.693 | 0.613 | 0.476 |
| DEDICOM tensor factorization | 0.705 | 0.637 | 0.567 |
| DeepWalk neural embeddings | 0.761 | 0.737 | 0.658 |
| Concatenated drug features | 0.793 | 0.764 | 0.712 |

# Project Proposal Example

**Title**: Evaluation of GNN for Predicting Disease Genes

**Motivation:**

- We will implement a simple GNN to predict whether genes in a PPI are involved in the development of disease
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

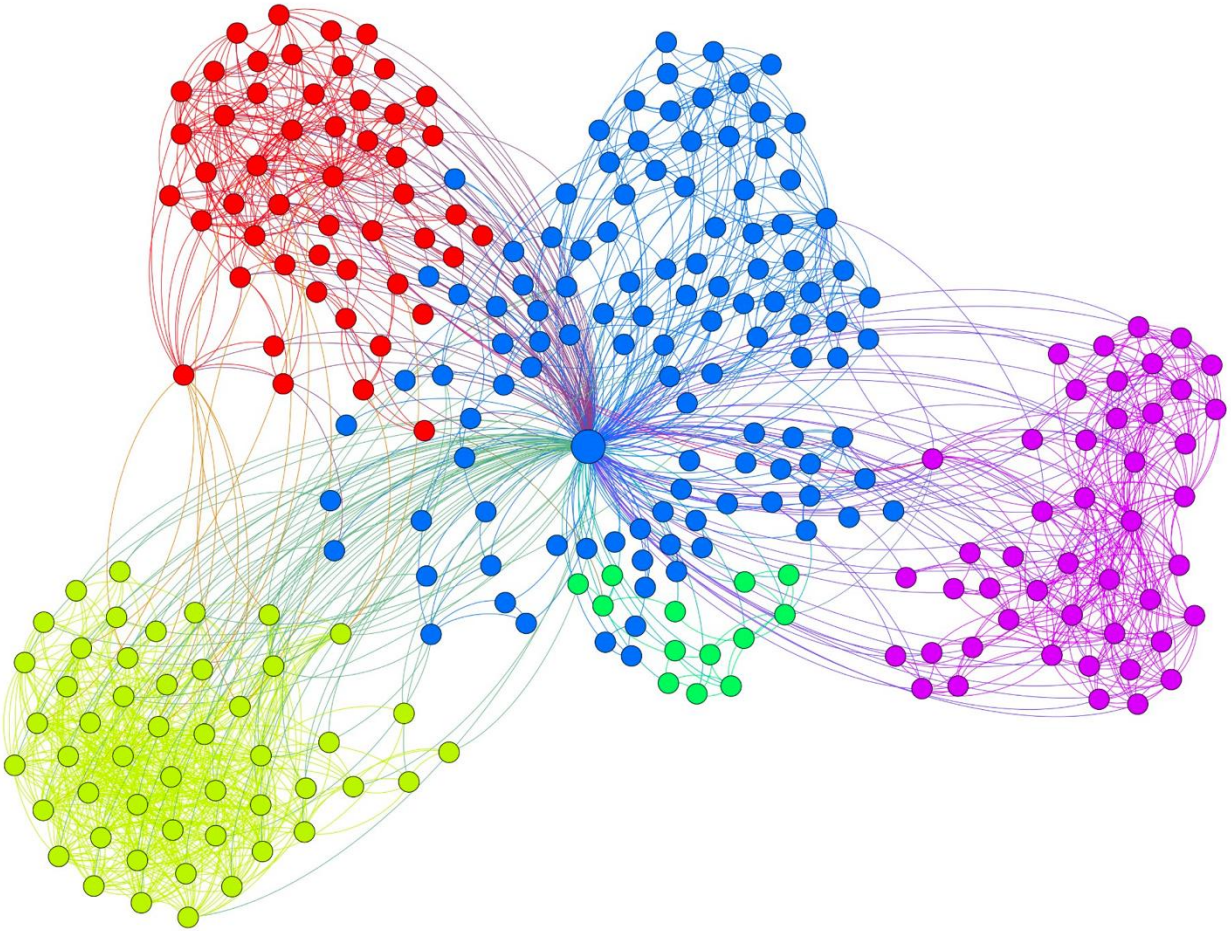
- Problem: [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

# Graph Clustering

Meaningful clustering of the nodes of a network?



# Graph Clustering

Several approaches:

apply embedding technique, then run "standard" clustering (e.g., k-means) on resulting vectors

Adapt some technique from "standard" clustering:

Techniques that are specialized for graphs

# Graph Clustering

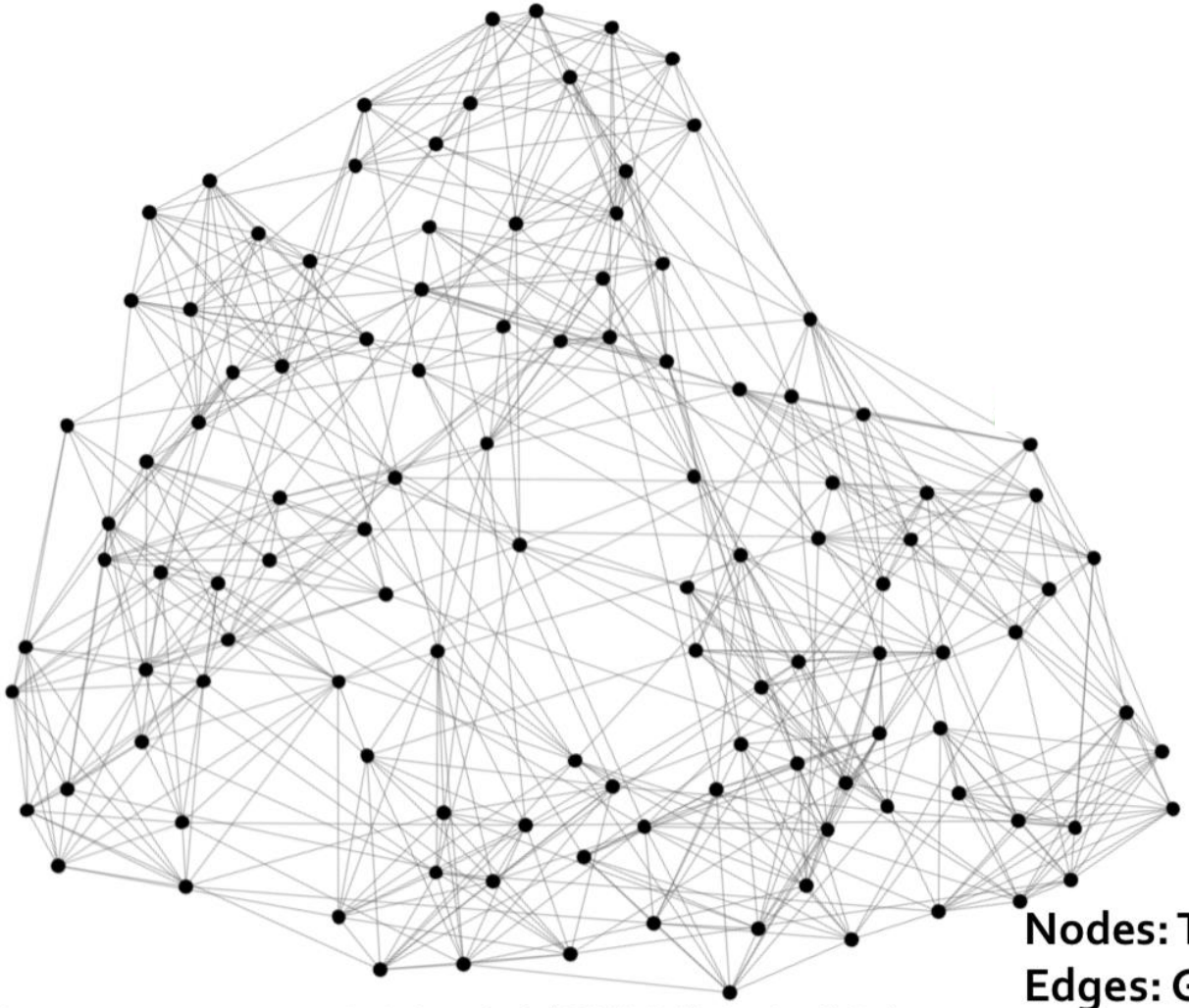
Definitions

Some methods (e.g., modularity-based clustering…)

# What is it Useful For?

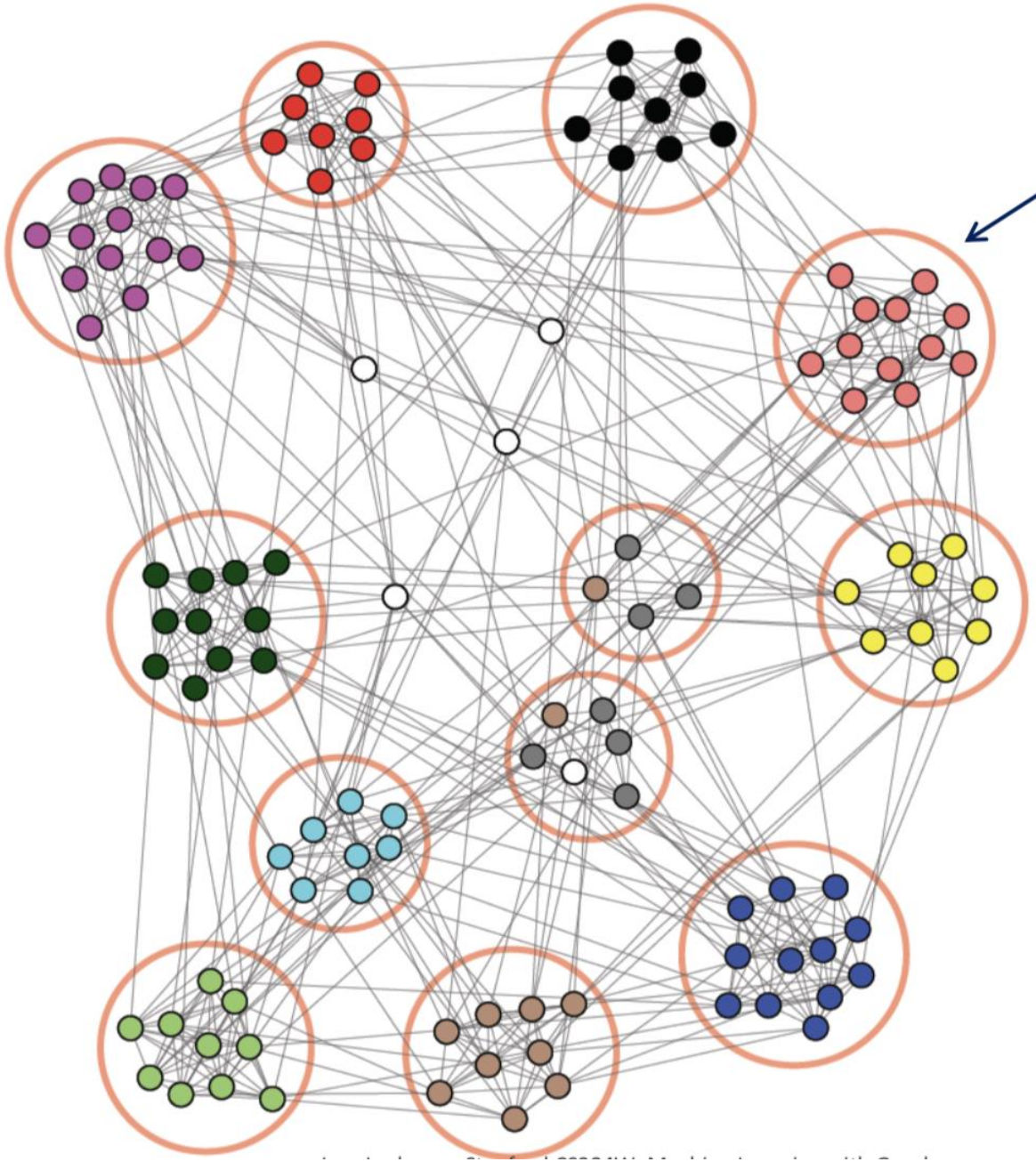
Finding networks *communities*

*Example: NCAA football network*



Nodes: Teams

Edges: Games played



## NCAA conferences

- Mid American
- Big East
- Atlantic Coast
- SEC
- Conference USA
- Big 12
- Western Athletic
- Pacific 10
- Mountain West
- Big 10
- Sun Belt
- Independents

**Nodes: Teams**
**Edges: Games played**

# Project Proposal Example

**Title**: Comparison of Graph Clustering Algorithms
**Motivation:**

- We want to compare these 3 methods for graph clustering
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: [DETAILS]
- Algorithms: [DETAILS]

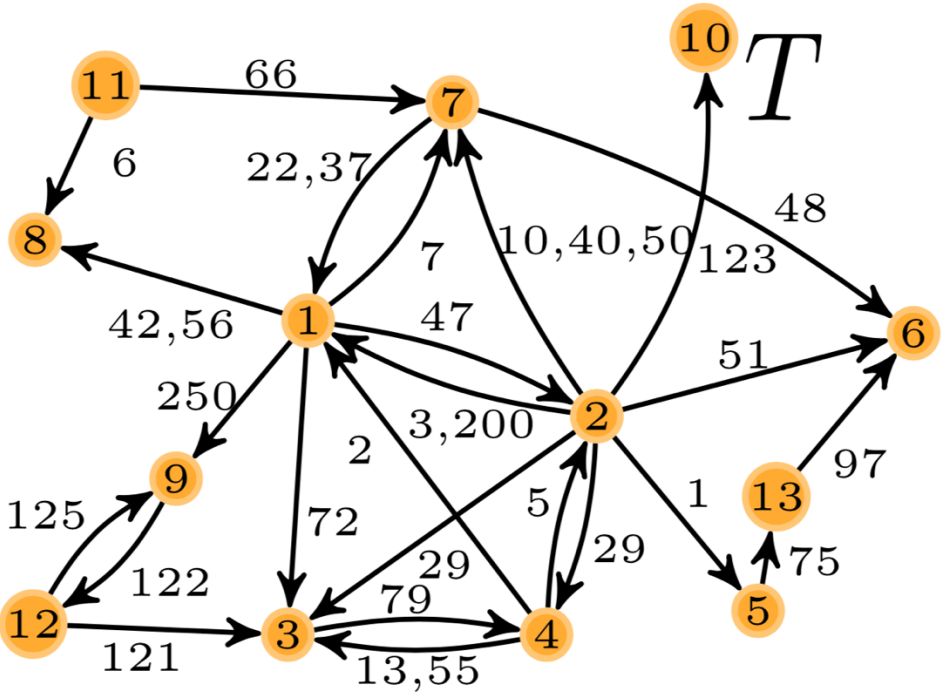
**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

# Advanced Topics

Analysis of Uncertain Graphs

Temporal Network: edges are labeled with time of events




Several interesting problems (e.g., finding frequent patterns)

# Project Proposal Example

**Title**: Improved Sampling Algorithm to Approximate the Count of Temporal Patterns

**Motivation:**

- The following paper proposes a sampling method to approximate the counts of temporal motifs. We think we can do better.
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: [DETAILS]
- Algorithms: [DETAILS]

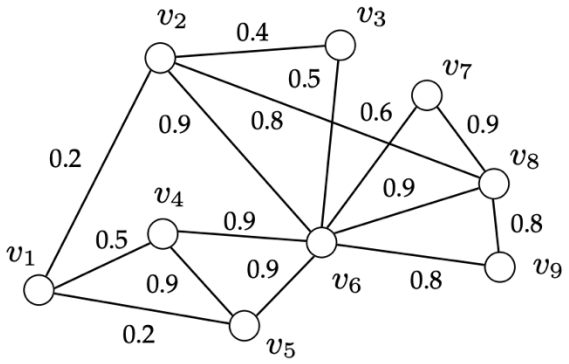
**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

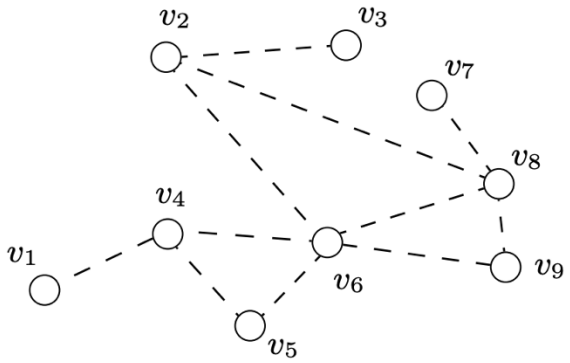
# Advanced Topics

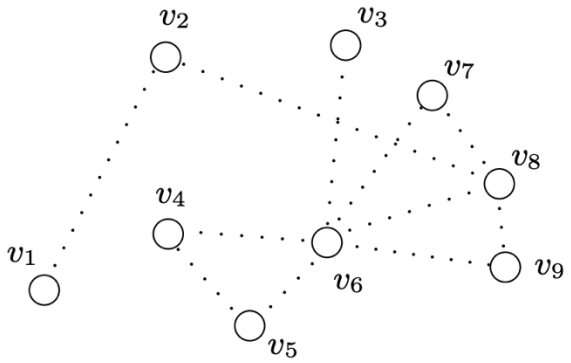
Analysis of Uncertain Graphs

Uncertain graph: edges have probabilities (of existing)




(a) $\mathcal{G}$




Several interesting problems (e.g., clustering)

# Project Proposal Example

**Title**: Evaluation of Scalability of Algorithms for Clustering Uncertain Graphs

**Motivation:**

- The following 2 papers describe methods for clustering uncertain graphs. They perform experiments on small graphs. We want to understand if the work on large graphs.
- Data: these networks from publicly available datasets [DETAILS]

**Method:**

- Problem: [DETAILS]
- Algorithms: [DETAILS]

**Intended experiments:**

- we will use the implementations available at [URL] [DETAILS]
- machine for experiments: [DETAILS]
- experiments: [DETAILS]

**Feel free to explore other topics of your interest!**