CS 5683: Big Data Analytics

Machine Learning for Graphs: The Basics

Arunkumar Bagavathi

Department of Computer Science

Oklahoma State university

Topics Overview

High. Dim. Data

- Data Features
- Dimensionality Reduction
- *Application* Rec. Systems

Text Data

- Clustering
- Non-linear Dim. Reduction
- *Application* IR

Graph Data

- PageRank
- ML for Graphs
- Community Detection

Others

- Data Streams Mining
- Intro. to Apache Spark

ML for Graphs: Importance

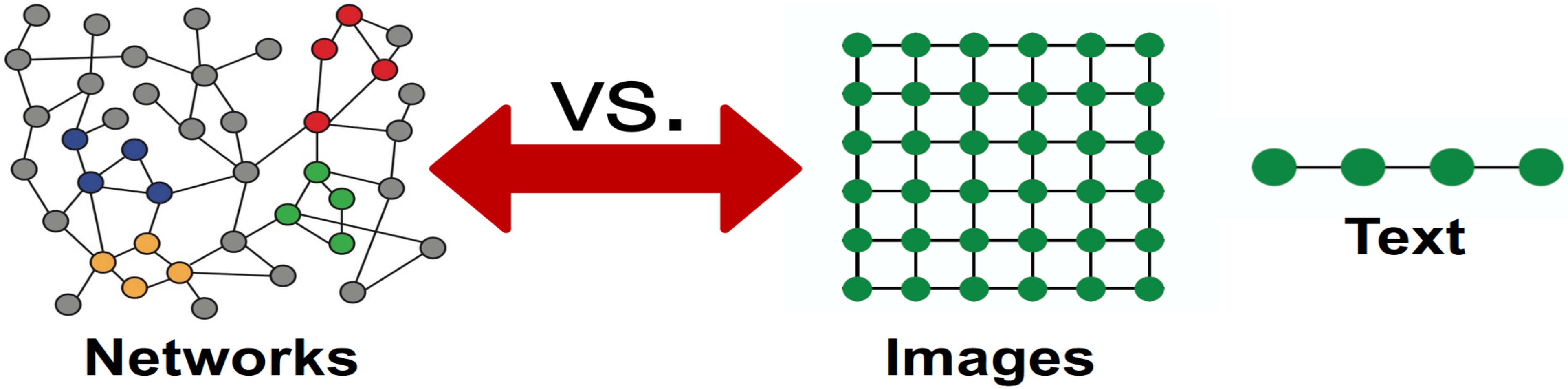
- Complex domains have a rich relational structure, which can be represented as a relational graph

- By explicitly modeling relationships we achieve better performance in downstream tasks

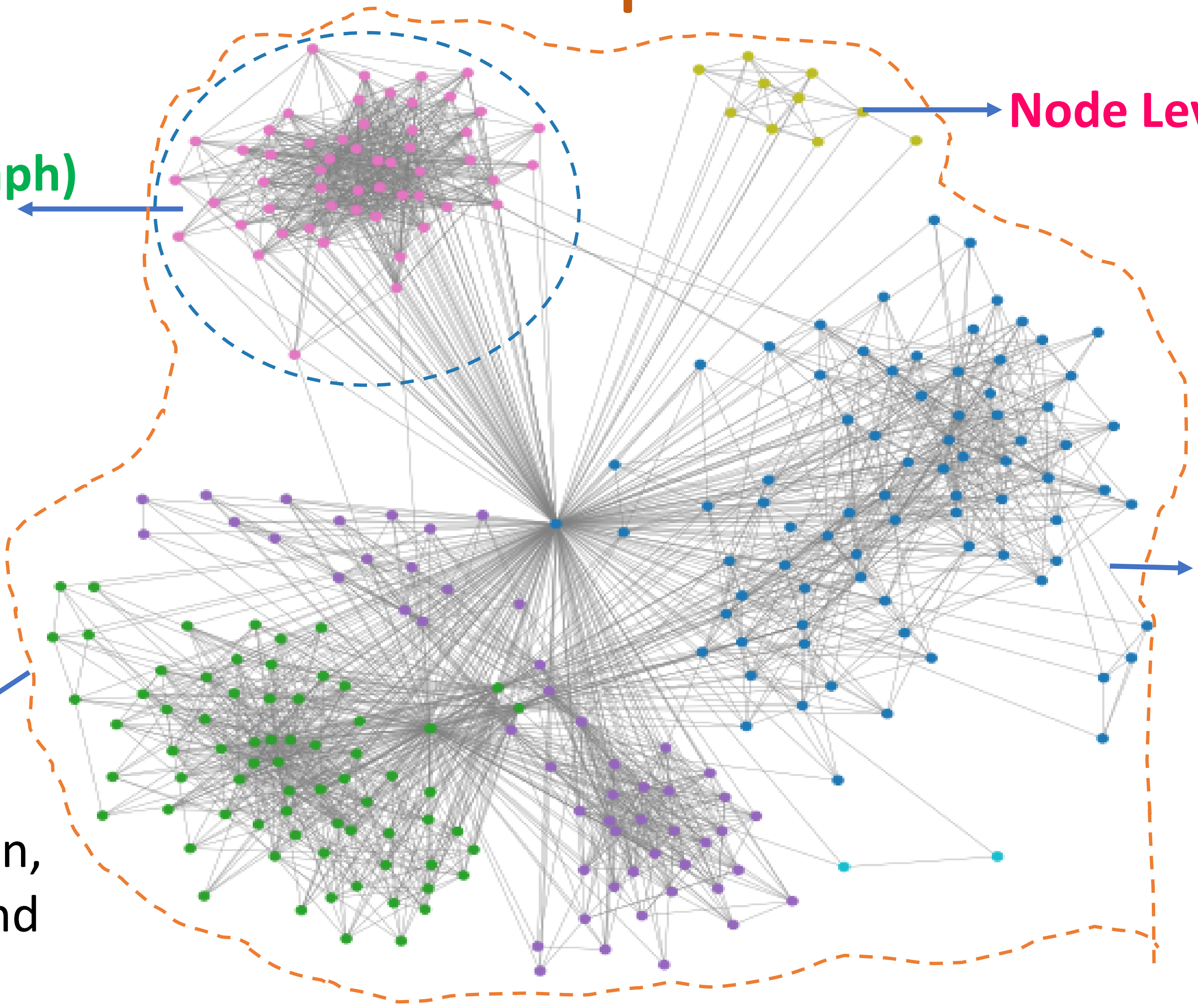
- **Question:** How can we take advantage of the relational structure and deep neural nets for better prediction?

ML for Graphs: Basic Problems

- **Networks are complex:** Arbitrary size and complex structure
- ***No fixed node ordering or reference point***
- **Often dynamic and have multimodal features**



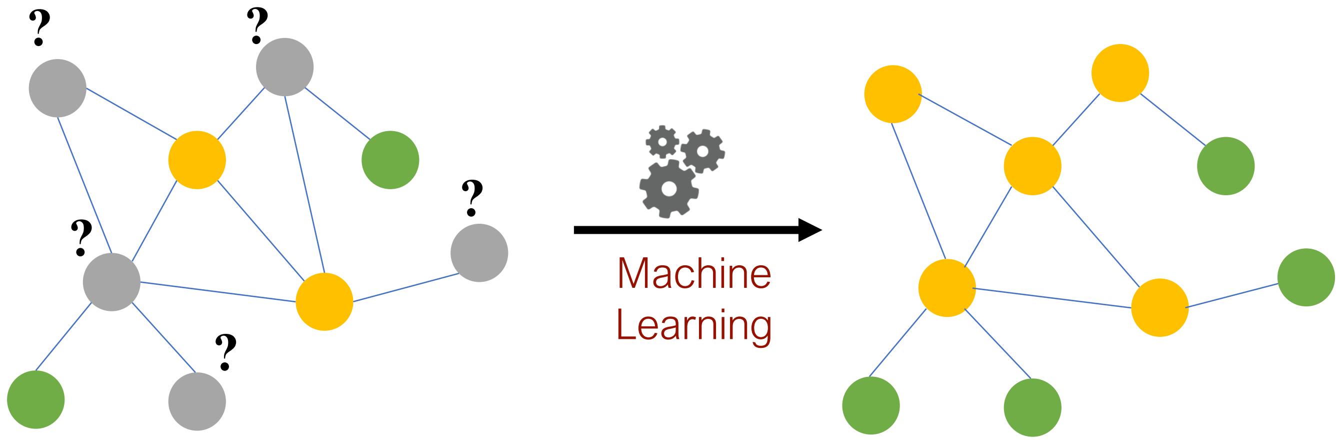
Different Graph ML Tasks



Node Level Tasks

- **Goal:** Characterize the structure and position of a node in the network
  - Node degree
  - Node importance & position
    - No. of shortest paths through nodes
    - Avg. shortest path length to other nodes
    - PageRank
  - Substructures around nodes

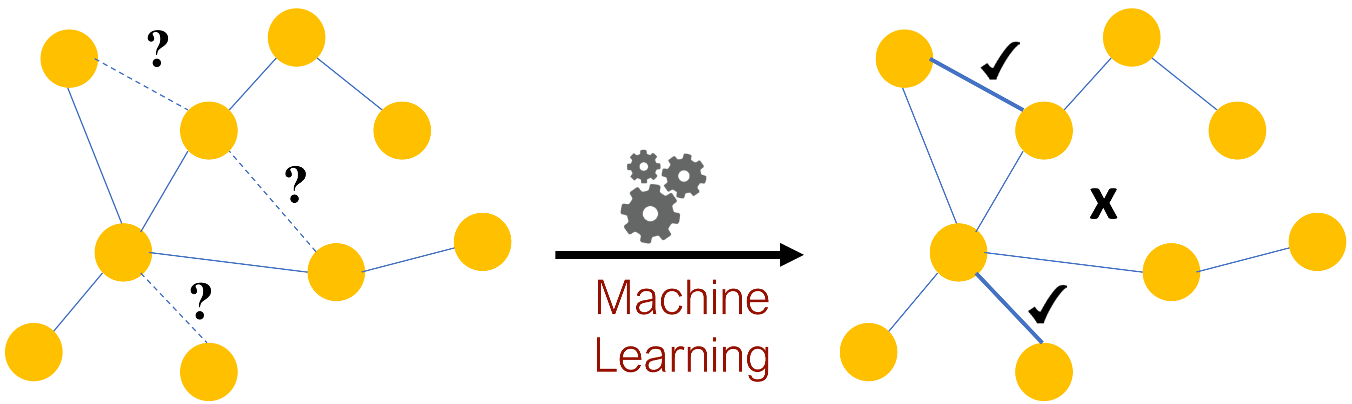
Node Classification



Edge Level Tasks

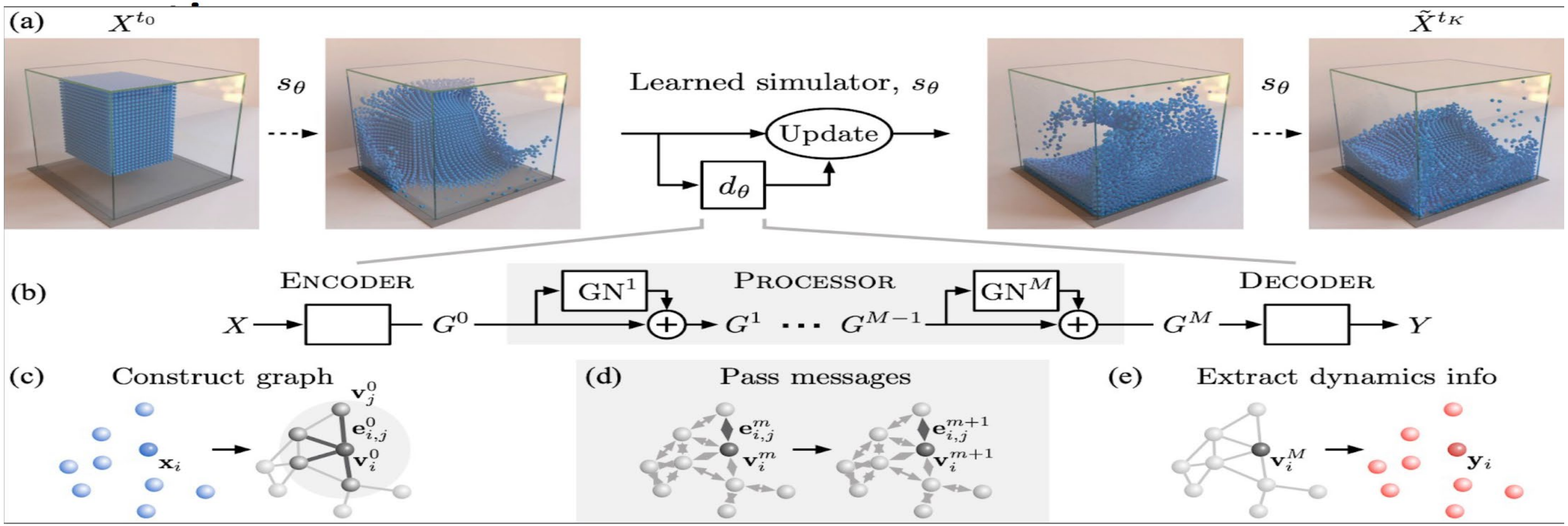
- **Goal:** Predict new/missing/unknown edges based on the existing edges and graph structure
- **Possible formulations:**
  - Edges missing at random
  - Edge prediction over time

Edge Prediction Task



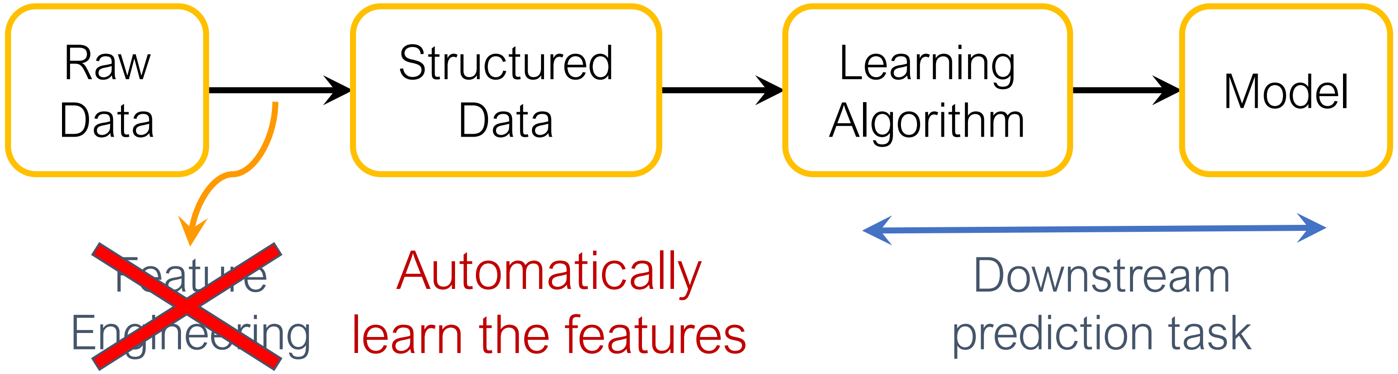
Graph Level Tasks

- **Goal:** Make predictions for an entire graph or subgraph
- **Example:**
  - Predict how the graph evolves over time
  - Find new subgraph patterns



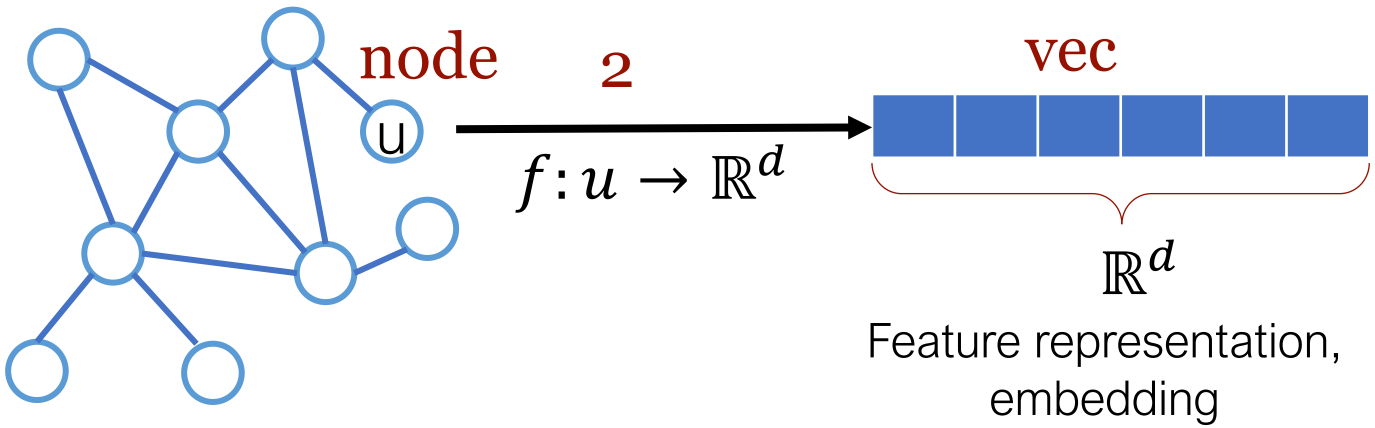
Machine Learning Lifecycle

- **(Supervised) Machine Learning Lifecycle:** More than the model to pick, the performance of the model depends on the quality of features represented in the data.
- What features to choose for prediction? Should we choose this feature or that feature every time? This is called ***feature engineering***

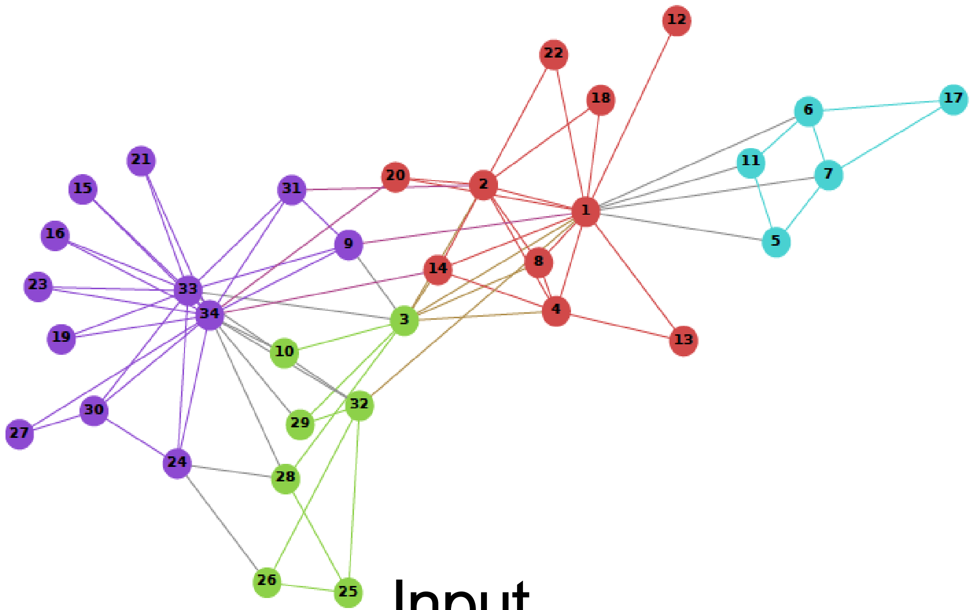


Feature Learning for Graphs

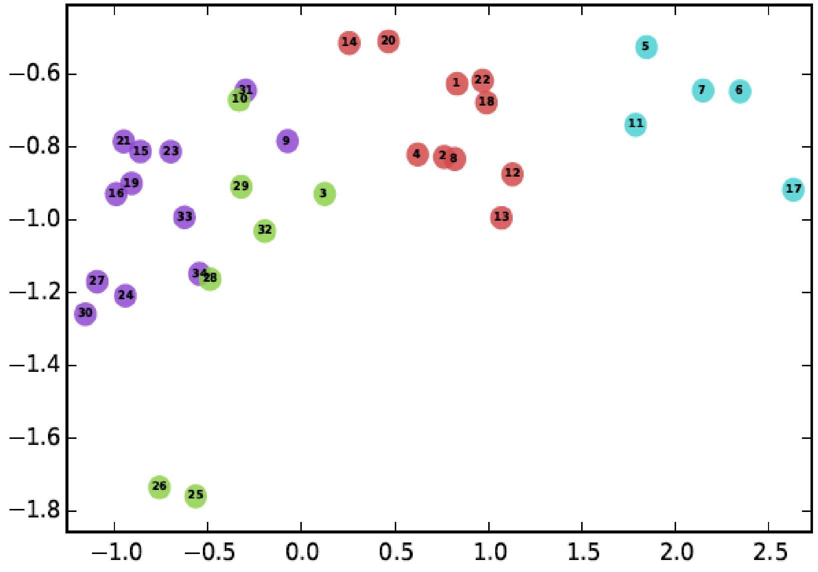
- **Goal:** Efficient task-independent feature learning for machine learning in networks/graphs.
- Design a general feature (representation) learning algorithm that produces node features based on their position in the network



Embedding Nodes



Input



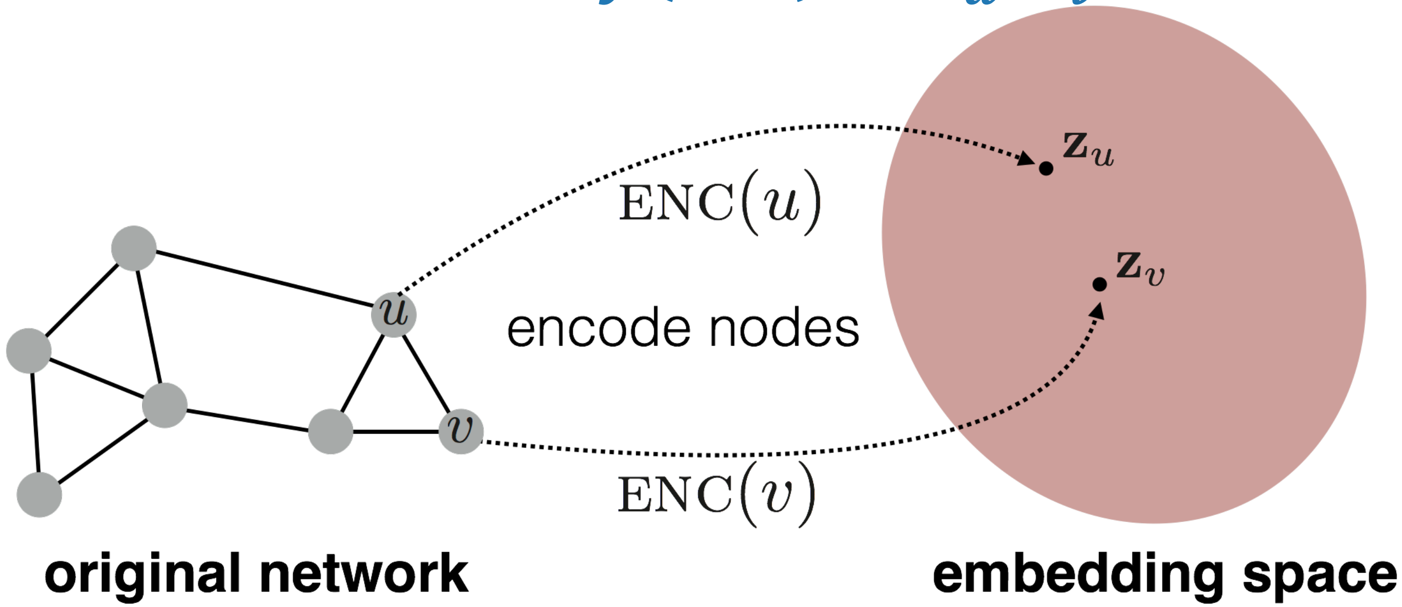
Output

- **Intuition:** Map nodes to the d-dimensional space as embeddings so that "*similar*" nodes in the graph have embeddings that are closer together and "*dissimilar*" nodes have embeddings far apart

Embedding Nodes

- **Setup:**
  - A graph ***G*** represented in an adjacency matrix ***A***
  - No extra node features or extra information is used!
- ***Goal:*** Encode nodes so that similarity in the embedding space approximates similarity of the original network

$$similarity(u, v) \approx z_u^T . z_v$$



Steps to Learn Nodes Embeddings

1. **Define an encoder function:** mapping from nodes of the network to embeddings

2. **Define a node similarity function:** a measure of similarity in the original network ($\boldsymbol{similarity(u, v)}$)

3. **Optimize encoder parameters such that:**

$$\boldsymbol{similarity(u, v) \approx z_u^T . z_v}$$

Agenda for this Class

1. **Shallow Embedding model –** *node2vec*
2. **Deep Embedding model** – GNN (GraphSage)

Questions???

Acknowledgements

Most of the slides are motivated from WWW'2018 Tutorial