Introduction to

Graph Machine Learning

Rizal Fathony
rizal@fathony.com

*Some of the materials are modified from Juree Leskovec's graph presentations

Rizal Fathony

Lead Data Scientist
Grab | Trust, Identity, and Safety
2021-Present | Indonesia

Visiting Lecturer
Universitas Gadjah Mada | DIKE
2022-Present | Yogyakarta, Indonesia

Post-Doctoral Researcher
Bosch | Bosch Center of Artificial Intelligence
2020-2021 | Pittsburgh, Pennsylvania, USA

Post-Doctoral Research Fellow
Carnegie Mellon University |Computer Science Department
2019-2021 | Pittsburgh, Pennsylvania, USA

Education

Ph.D. in Computer Science
University of Illinois at Chicago
2014-2019 | Chicago, Illinois, USA

Master's in Computer Science
University of Illinois at Chicago
2012-2014 | Chicago, Illinois, USA

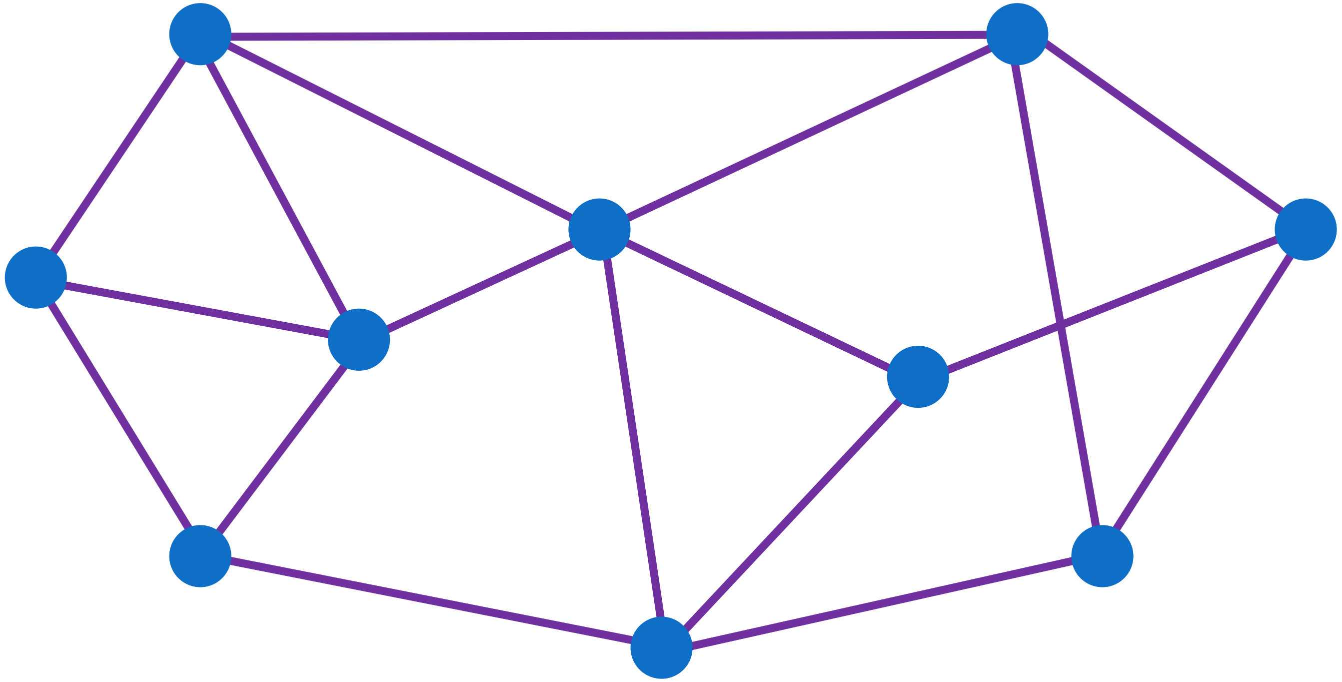
Bachelor's in Statistical Computing
Sekolah Tinggi Ilmu Statistik
2003-2007 | Jakarta, Indonesia

What is a Graph?

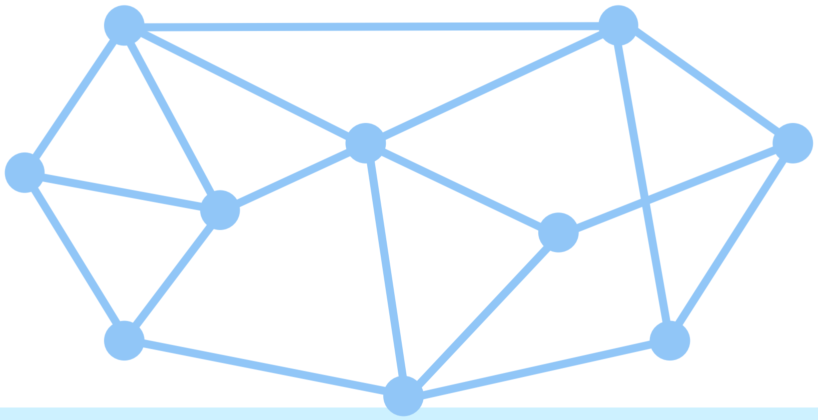
Mathematical structure for pairwise relations

Nodes

Edges

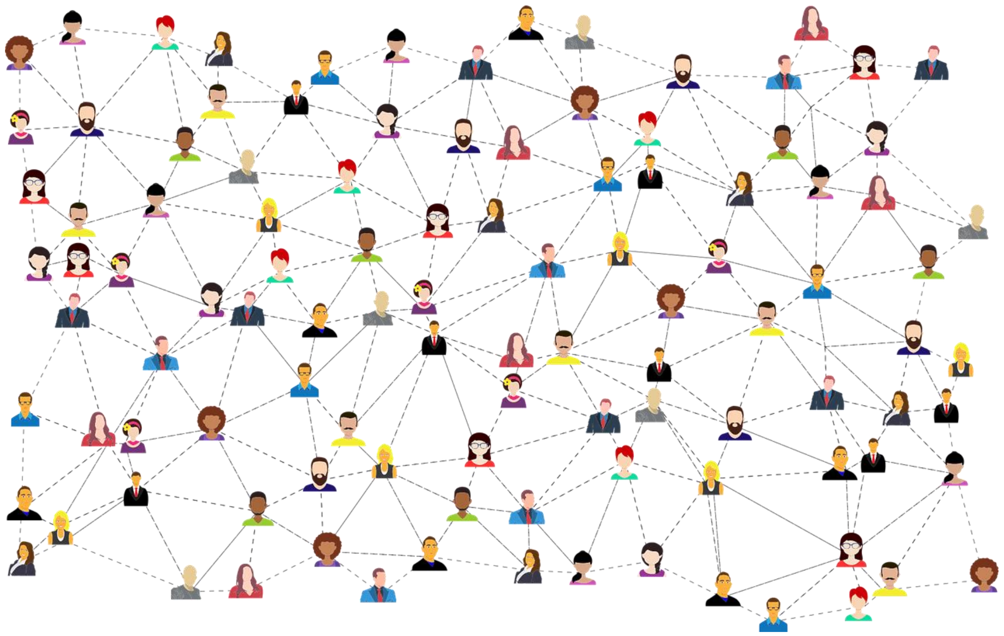


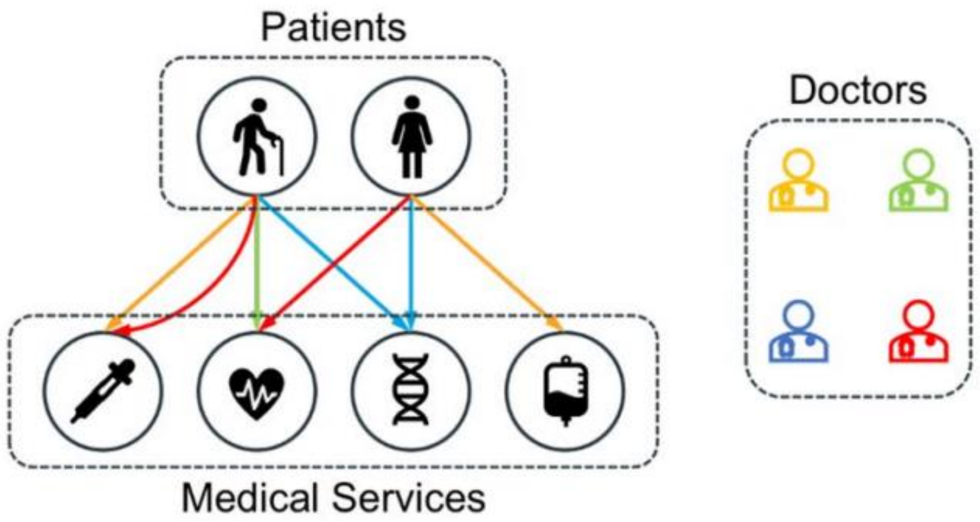
Why Graphs?



Graph is a general method for describing and modeling complex systems

Many Data are Graphs





Bipartite Graph

Social Networks

Nodes: Person/Account

Edges: Friendship/Follows

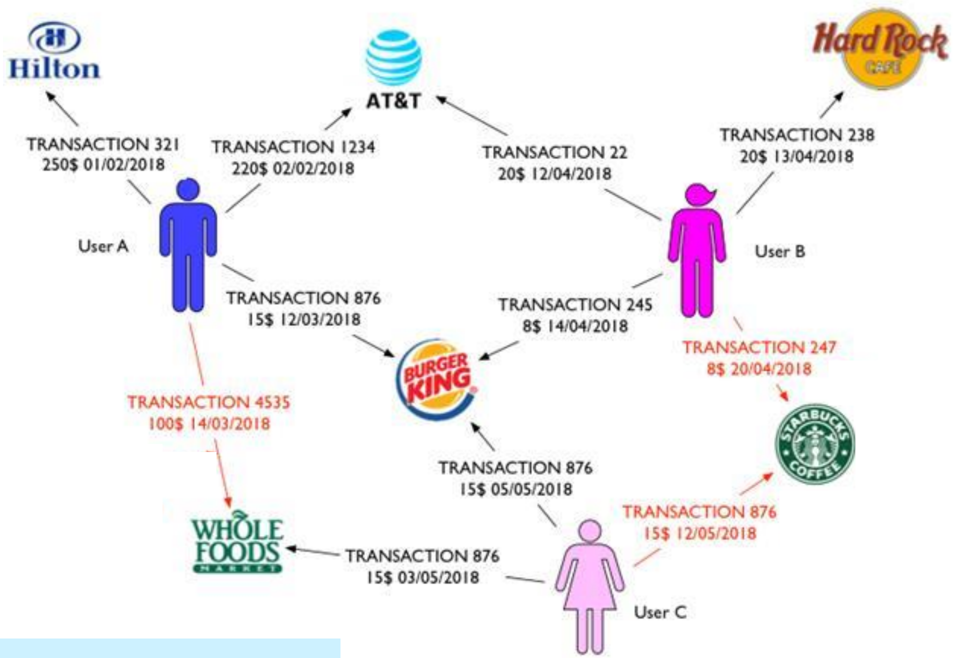
Health Records

Nodes: Patient, Medical Service

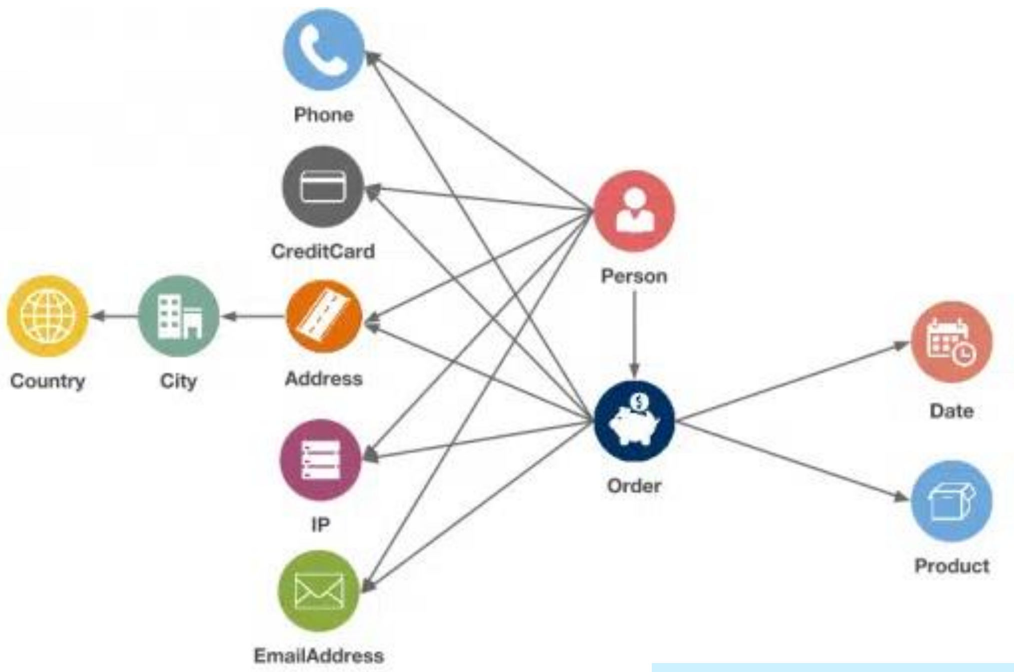
Edges: Treatment by Doctors

https://brandblast.com/strategy/avatars-and-favicons/
https://www.nature.com/articles/s41598-021-85255-w.pdf

Many Data are Graphs



Directed Graph



Heterogenous Graph

Financial Transactions

Nodes: Customer, Merchant

Edges: Transaction/Payment

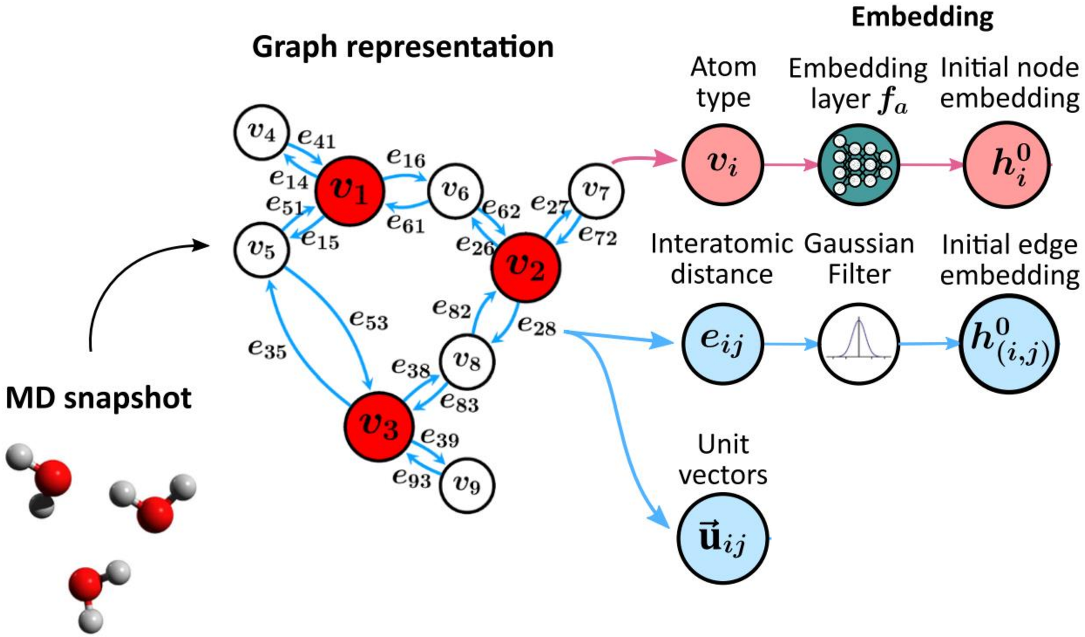
E-Commerce Data

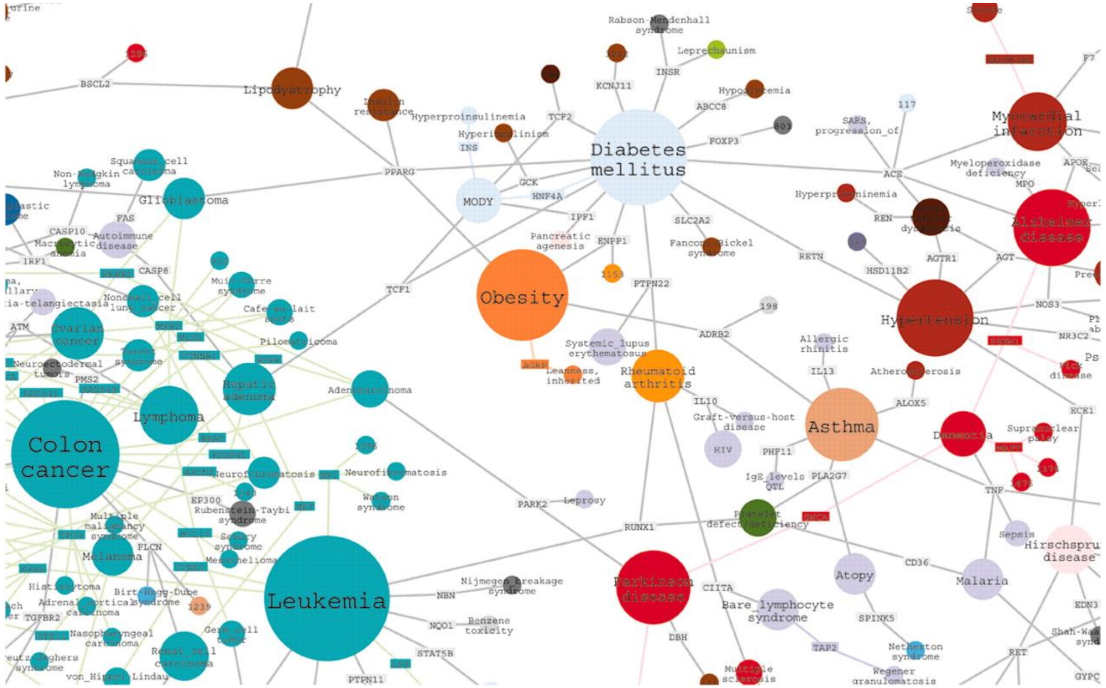
Nodes: Person, Product, Credit Cards, …

Edges: Has Phone, Has Address, Orders, …

https://livebook.manning.com/book/graph-powered-machine-learning/chapter-3/v-8/
https://7wdata.be/data-analysis/fraud-detection-in-retail-with-graph-analysis/

Many Data are Graphs





Molecular Modeling

Nodes: Atom

Edges: Chemical Bond

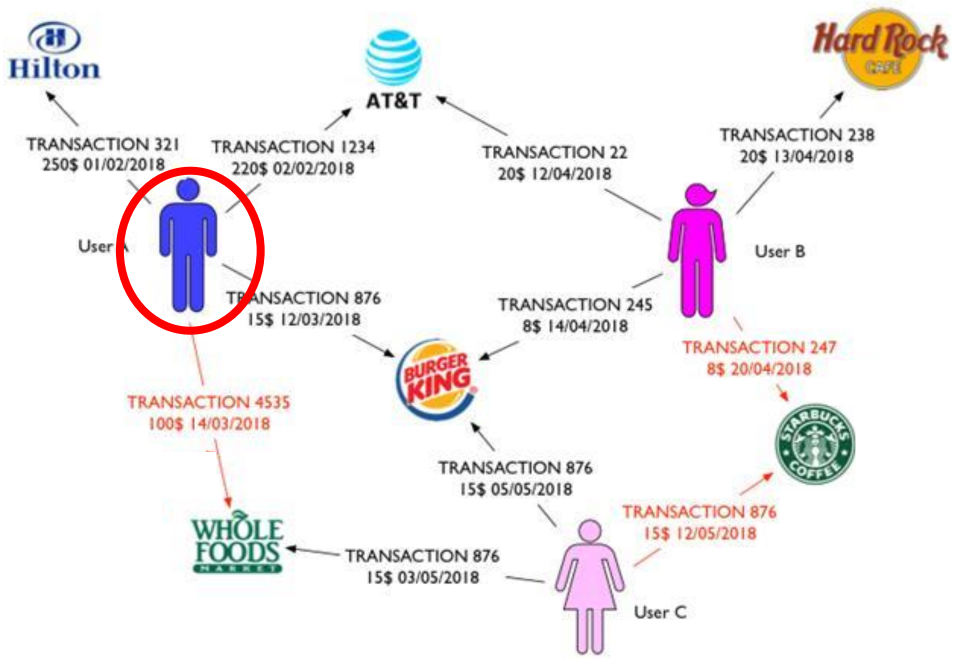
Human Disease Network

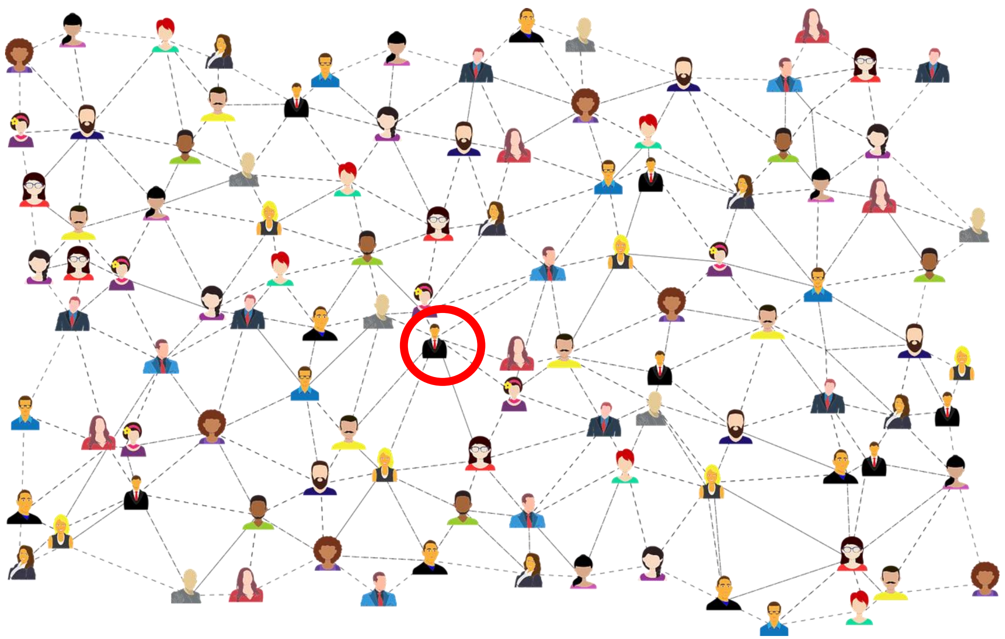
Nodes: Disease

Edges: Genetic Link

https://www.nature.com/articles/s41524-021-00543-3.pdf
https://studentwork.prattsi.org/infovis/labs/the-human-disease-network/

Task Example: Node Prediction





Input: Transactional Graph

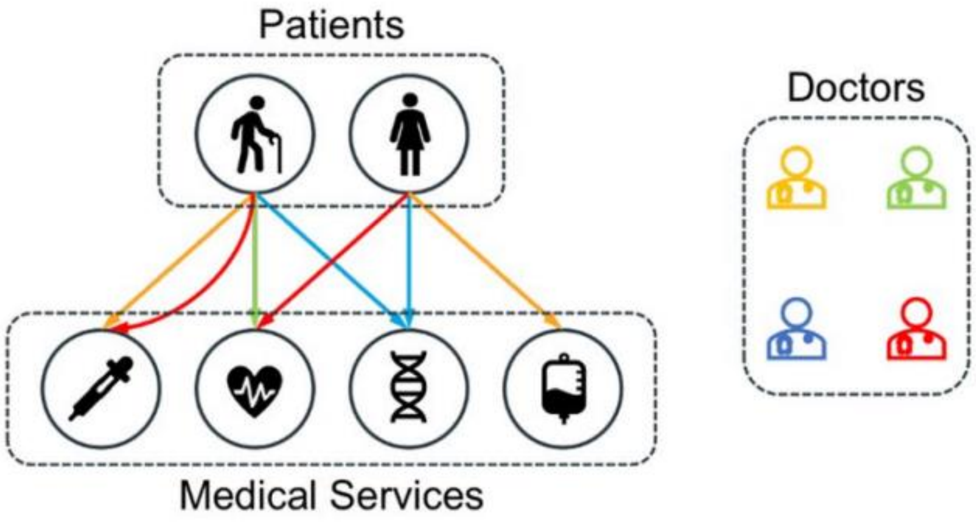
Task: Find user that use stolen credit card in the transactions

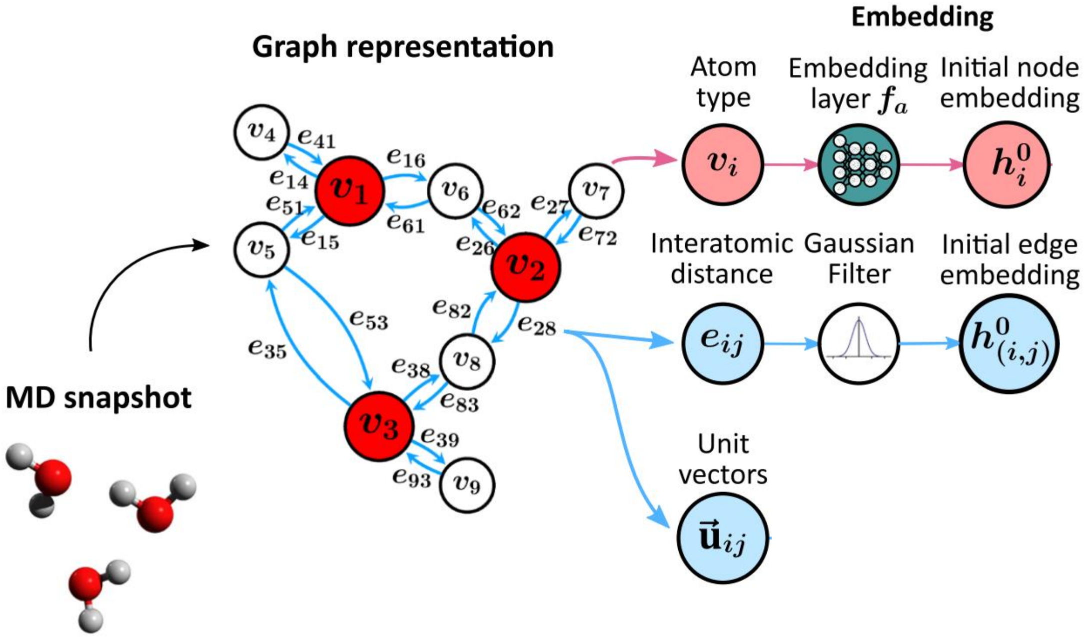
Input: Social Network

Task: Identify fake user with influence power

https://livebook.manning.com/book/graph-powered-machine-learning/chapter-3/v-8/
https://brandblast.com/strategy/avatars-and-favicons/

Task Example: Edge Prediction





Input: Health Records Graph

Task: Predict if a patient need to see a doctor for medical treatment

Input: Molecular Graph

Task: Predict how strong the chemical bonds' force for a given molecule

https://eng.uber.com/uber-eats-graph-learning/
https://www.nature.com/articles/s41524-021-00543-3.pdf

What's Next?

1 Introduction

2 ML Algorithms

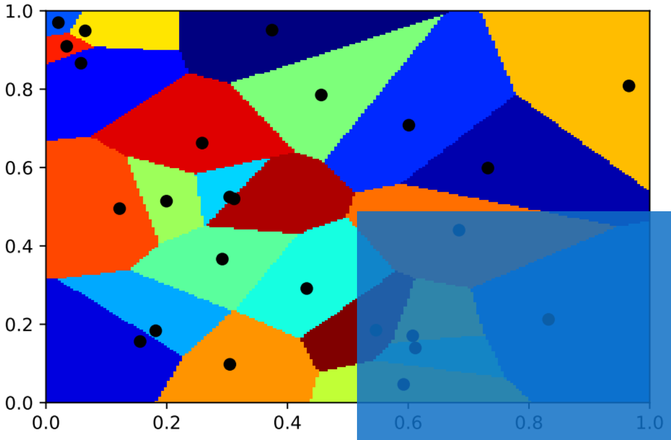
3 Node Embedding

4 Graph Neural Networks

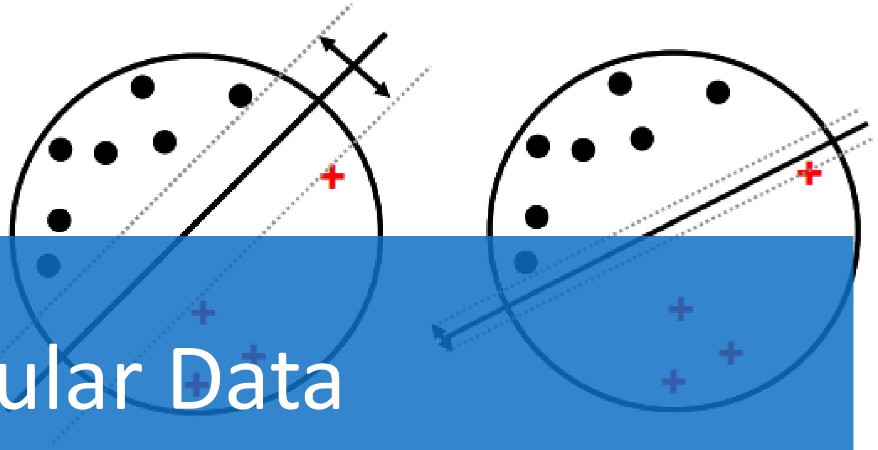
Machine Learning Algorithms

From Classical to Graph ML

Classical Machine Learning Algorithms

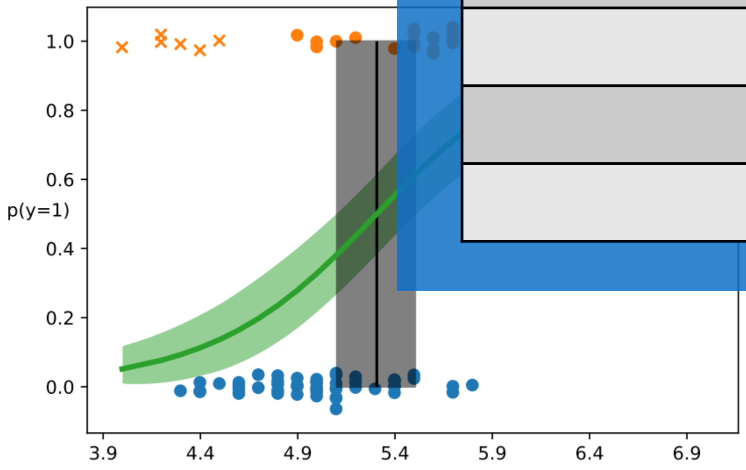


K-Nearest Nei

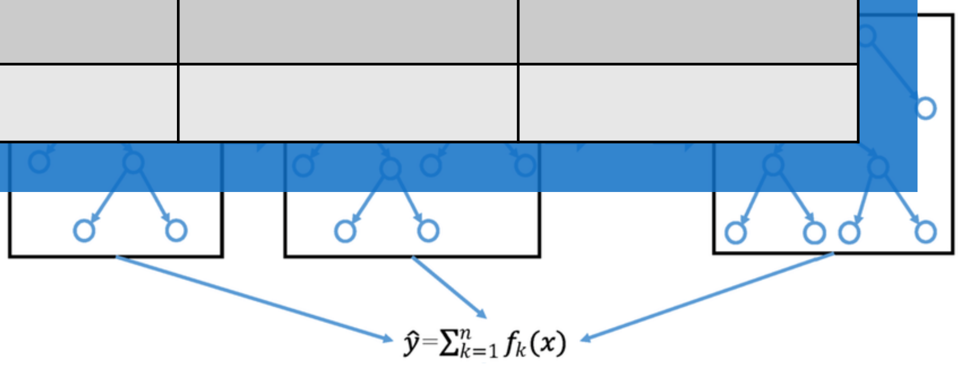


Tabular Data

Feature 1	Feature 2	Feature 3	Feature 4



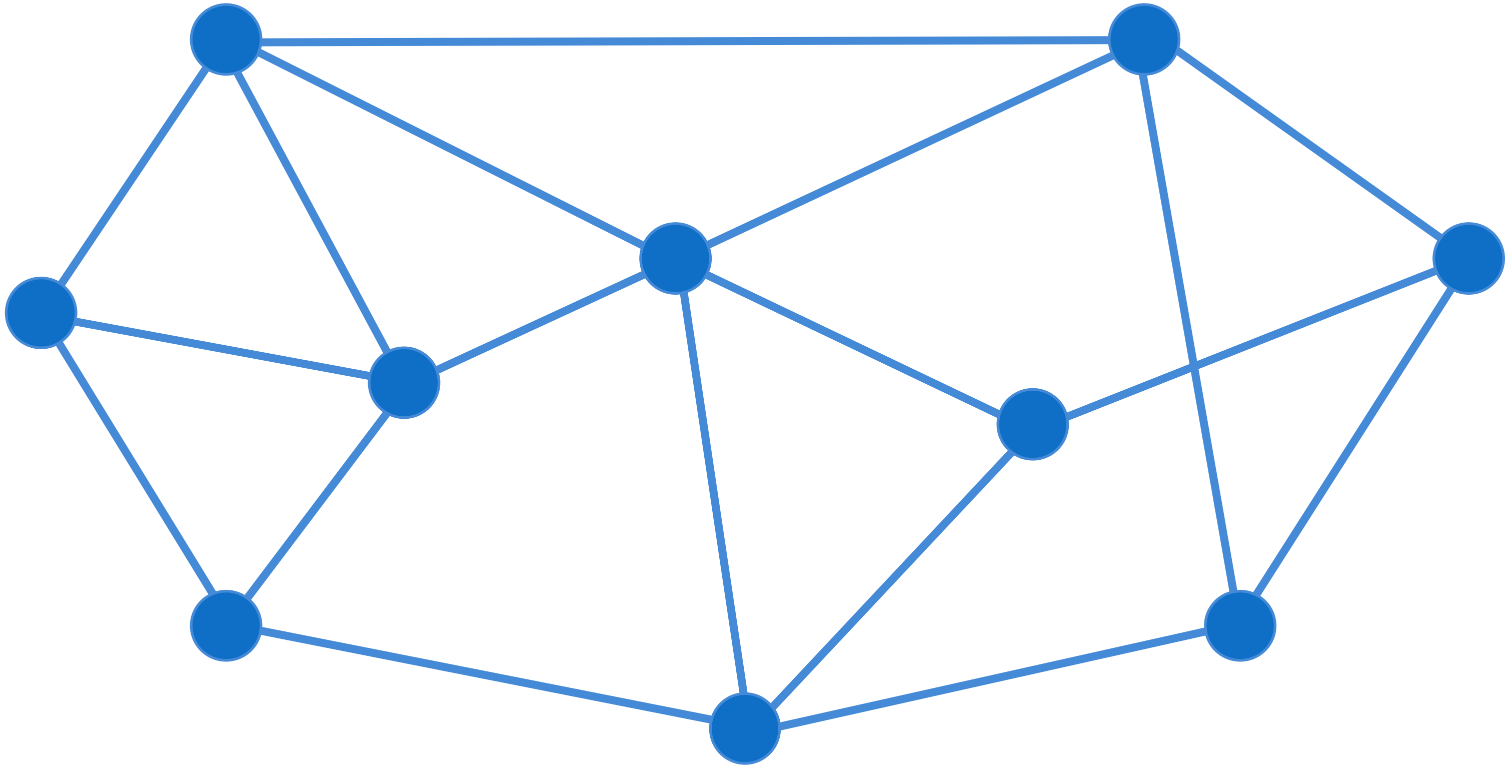
Logistic Regression



(Boosted) Decision Tree

https://probml.github.io/pml-book/book1.html

Classical ML for Graph Data?



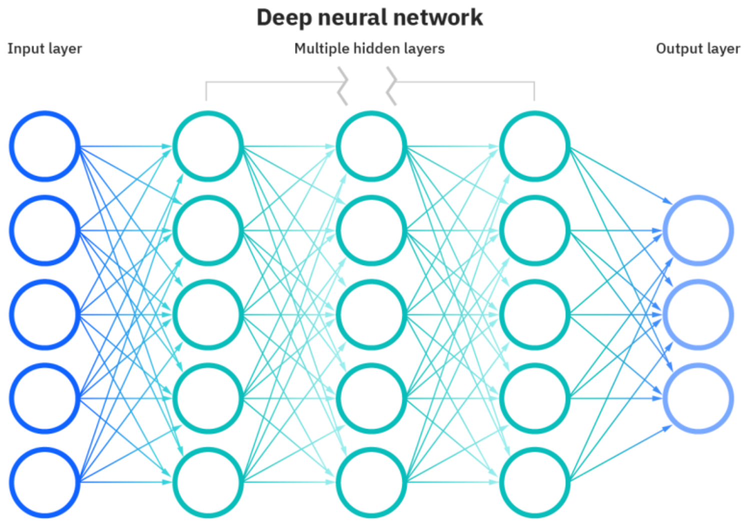
Graph

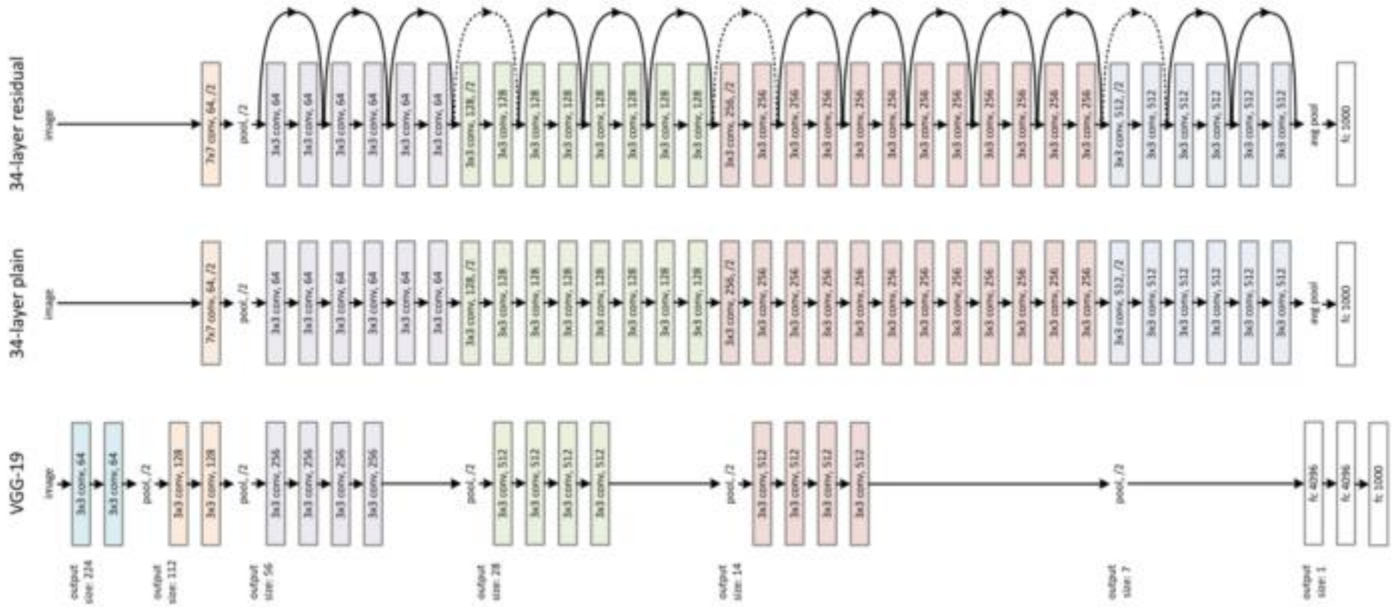
Classical ML for Graph Data?

Tabular

Neural Networks and Deep Learning

Multiple layers of learning





Multi Layer Perceptron (MLP)

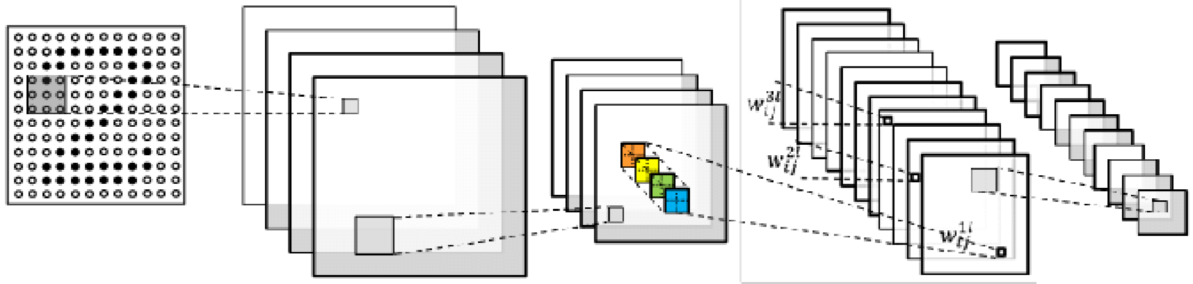
Residual Networks (ResNet)

Capable to learn from *"raw"* data

https://www.ibm.com/cloud/learn/neural-networks
https://medium.com/analytics-vidhya/deep-residual-learning-for-image-recognition-resnet-94a9c71334c9

Grids and Sequences

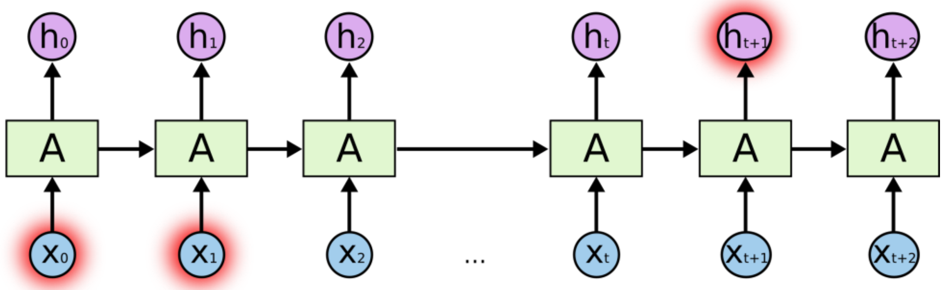
Grids



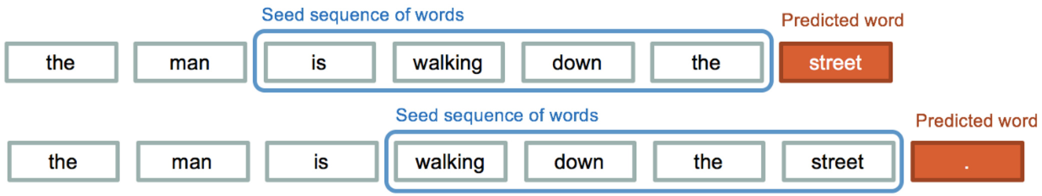
Convolutional Neural Networks (CNN)

Images

Sequences



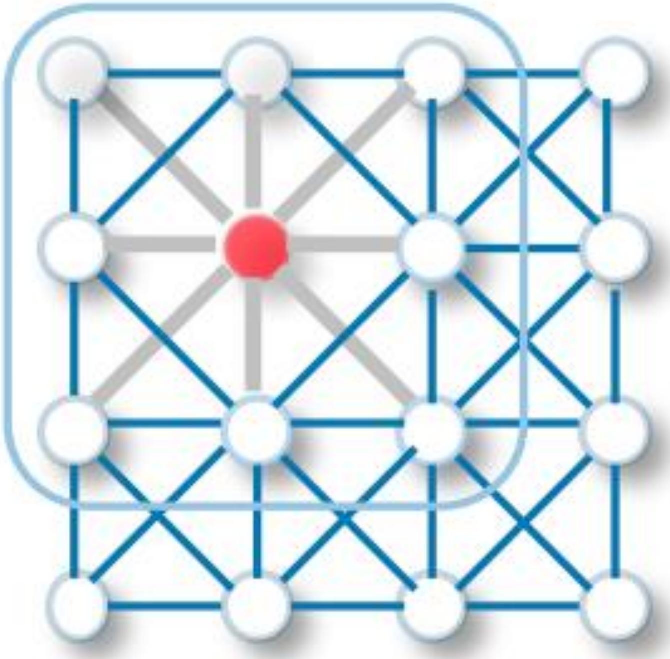
Recurrent Networks (RNN)



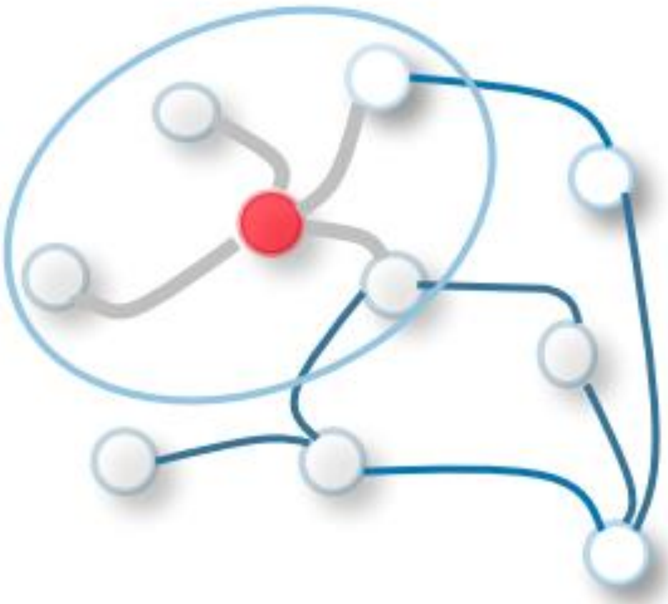
Text

https://www.researchgate.net/figure/Feed-forward-of-a-CNN-model-CONV-CONV-layers-are-the-main-components-of-a-CNN-model_fig1_305727136
https://kotakode.com/blogs/4437/Mengenal-LSTM-Networks---Part-3

Grid and Sequence as Graph



Grid Computation Flow



Graph Computation Flow

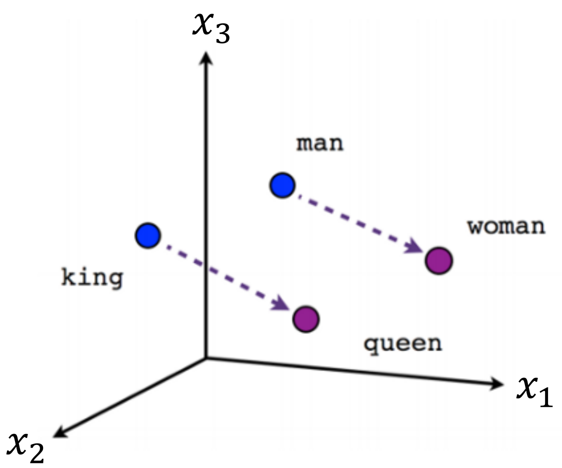
Node Embedding

Nodes + neighbors → numbers

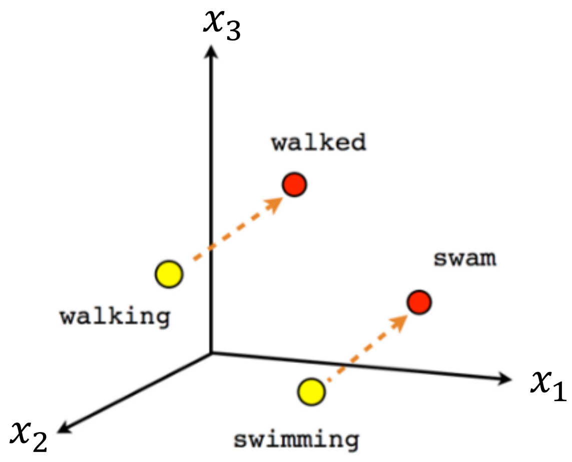
Inspiration from word embedding: **word2vec**

Map **words** to **numerical** features

similar word → similar values
preserve word associations



Male-Female



Verb tense

king – man + woman ≈ queen

walked – walking ≈ swam - swimming

word2vec training process:

predict the neighboring words

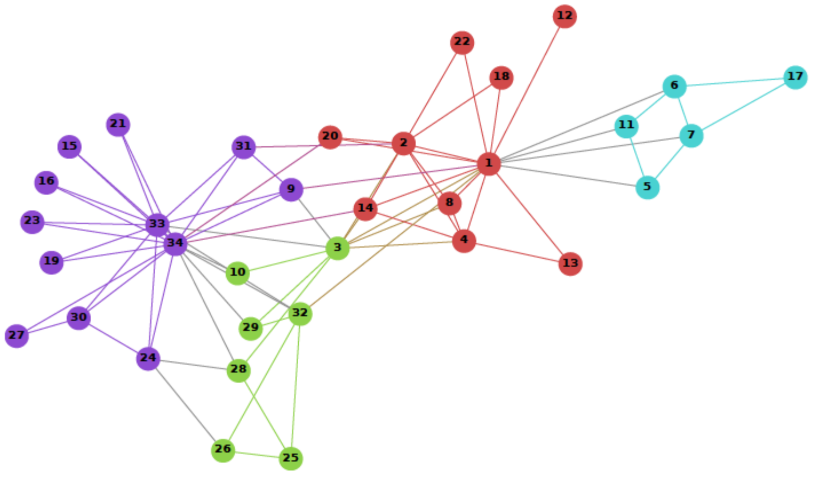
Source Text	Training Samples
The quick brown fox jumps over the lazy dog. ⟹	(the, quick) (the, brown)
The **quick** brown fox jumps over the lazy dog. ⟹	(quick, the) (quick, brown) (quick, fox)
The quick **brown** fox jumps over the lazy dog. ⟹	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown **fox** jumps over the lazy dog. ⟹	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8
http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

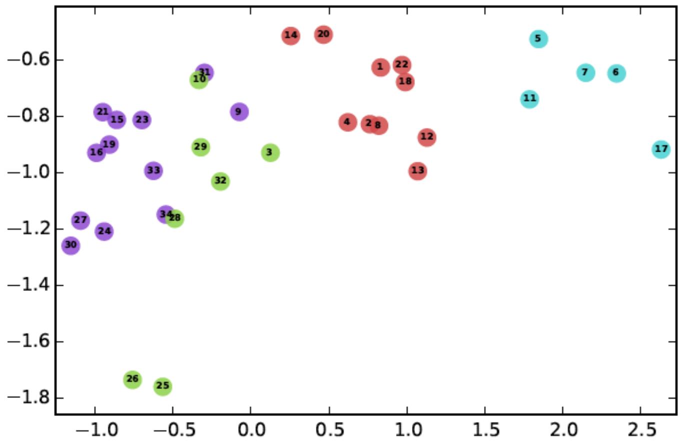
Node embedding algorithm

Map nodes in a graph to
numerical features (embedding)

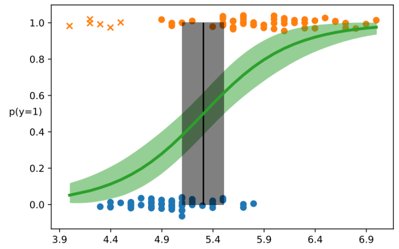
similar nodes → **similar** embeddings

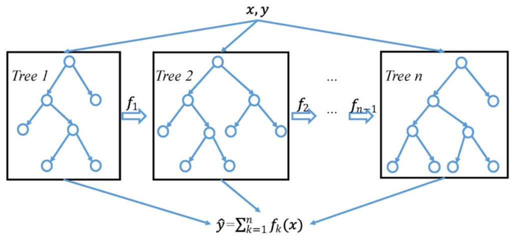


Graph



Embedding





Classifier
(ML models)

http://snap.stanford.edu/proj/embeddings-www/

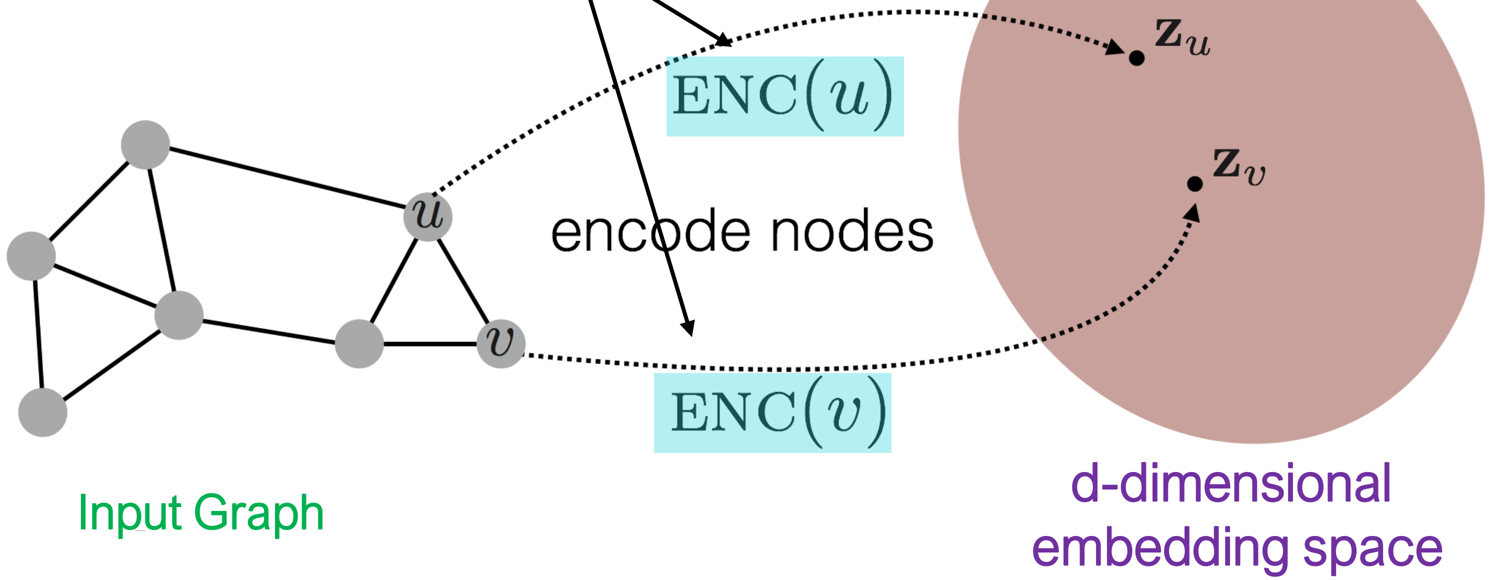
Node embedding components

1. Encoder
2. Similarity metric
3. Optimization Algorithm

$$\text{similarity}(u, v) \approx \mathbf{z}_v^\top \mathbf{z}_u$$

dot product between node embeddings

Need to define!



http://snap.stanford.edu/proj/embeddings-www/

Simple node embedding (example)

1. Encoder

"Shallow encoder"

Each node is assigned a unique embedding vector
(i.e., we directly optimize the embedding of each node)

used by most node embedding models

$$\mathrm{ENC}(v) = \mathbf{z}_v$$

d-dimensional embedding

node in the input graph

2. Similarity metric

Key ingredient that differentiate node embedding methods

- Adjacency based similarity

if two nodes are connected via an edge = similar

Adjacency matrix

$\mathbf{A}$

$A_{u,v} = 1$, if u and v are connected via an edge
$A_{u,v} = 0$, otherwise

3. Optimization Algorithm

Loss function:

$$\boxed{\mathcal{L}} = \boxed{\sum_{(u,v)\in V\times V}} \|\boxed{\mathbf{z}_u^\top \mathbf{z}_v} - \boxed{\mathbf{A}}_{u,v}\|^2$$

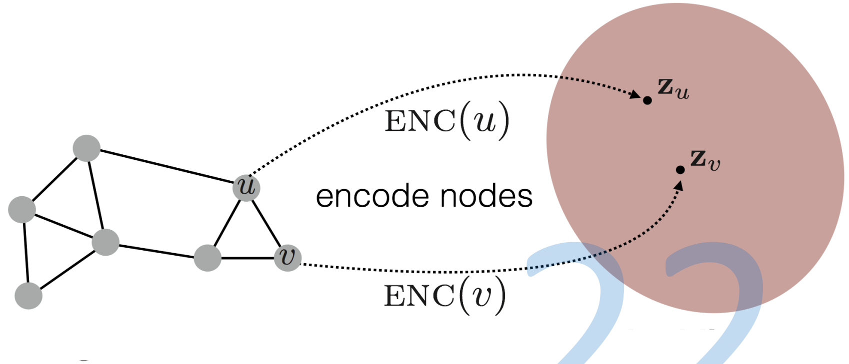
loss (what we want to minimize)

sum over all node pairs

embedding similarity

(weighted) adjacency matrix for the graph

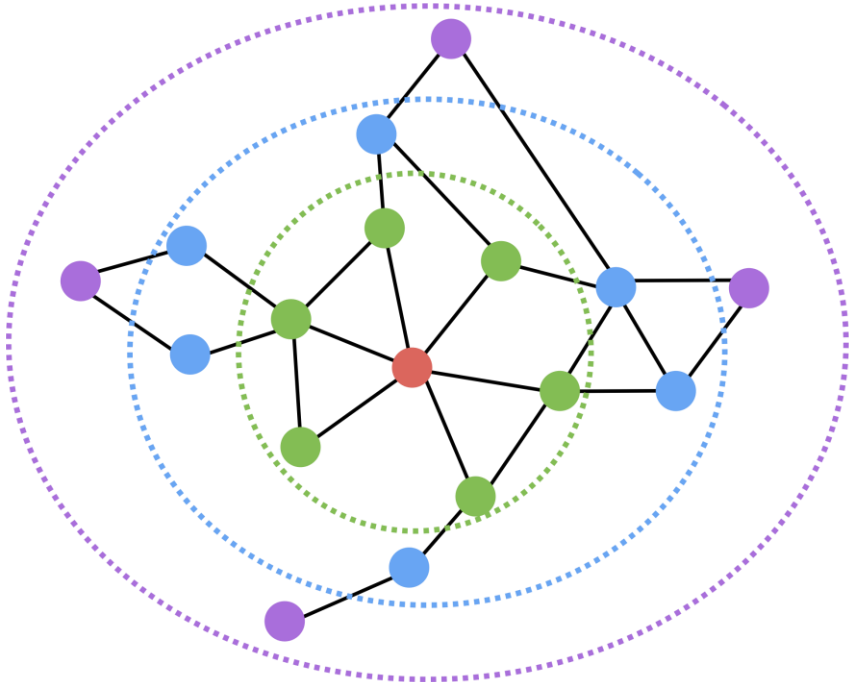
Optimize with SGD!



http://snap.stanford.edu/proj/embeddings-www/

K-Hop Similarity

Neighboring nodes achievable in k-hop should have similar embedding



- **Red:** Target node
- **Green**: 1-hop neighbors
  - **A** (i.e., adjacency matrix)
- **Blue**: 2-hop neighbors
  - $\mathbf{A}^2$
- **Purple:** 3-hop neighbors
  - $\mathbf{A}^3$

Objective:

$$\mathcal{L} = \sum_{(u,v) \in V \times V} \| \mathbf{z}_u^\top \mathbf{z}_v - \boxed{\mathbf{A}_{u,v}^k} \|^2$$

adjacency matrix's k-th power = k-hop

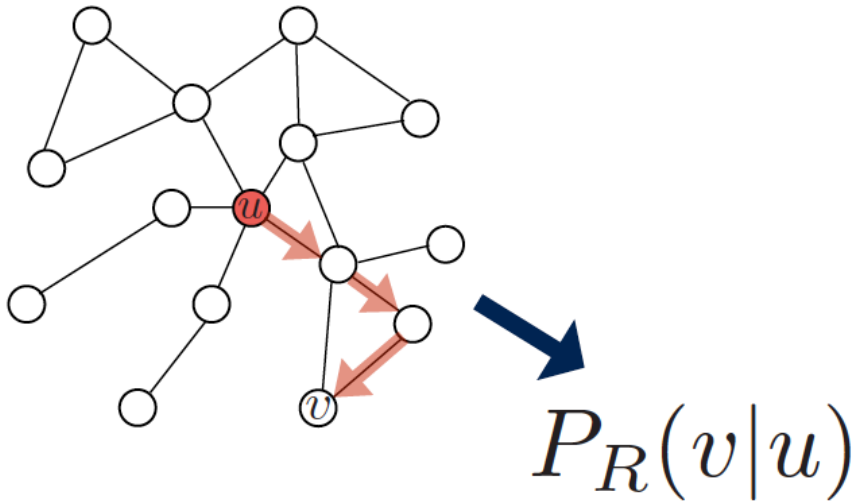
Theorem: Raising an adjacency matrix A of simple graph G to the n-th power gives the number of n-length walks between two vertices v_i, v_j of G in the resulting matrix.

http://snap.stanford.edu/proj/embeddings-www/

Random Walk Similarity

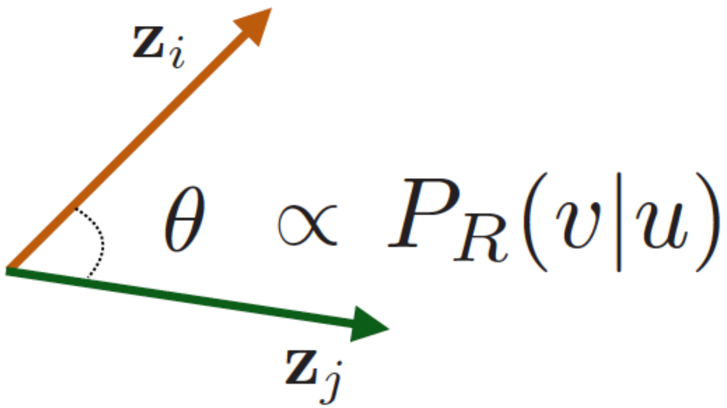
Random walk: start from node u, repeatedly jump (walk) to a neighboring node

$P_R(v|u)$: probability of visiting node v from random walks starting from node u



A random walk from u to v

Embedding similarity should approximate $P_R(v|u)$



Cosine similarity

Random Walk Optimization

Objective

maximize likelihood of
random walk co-occurrences

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

For each node u collect $N_R(u)$, the set of nodes visited on random walks starting from u.

Parameterize

use softmax

$$P(v|\mathbf{z}_u) = \frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}$$

Optimization: minimize $\mathcal{L}$

$$\mathcal{L} = \boxed{\sum_{u \in V}} \boxed{\sum_{v \in N_R(u)}} - \log \boxed{\left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)} \right)}$$

sum over all nodes u

sum over nodes v seen on random walks starting from u

predicted probability of u and v co-occuring on random walk

Nested sum over nodes gives O(|V|^2) complexity!!

Approximate the normalization constant

Negative sampling

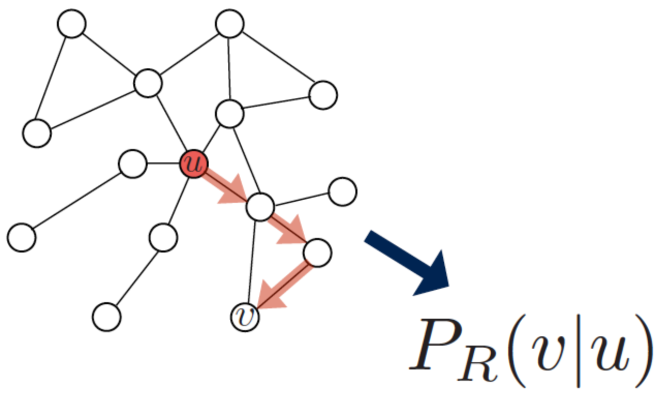
- Most pairs of nodes are not connected (negative sample)

- Instead of normalizing w.r.t. all nodes, just normalize against k random negative samples.

Why random walk?

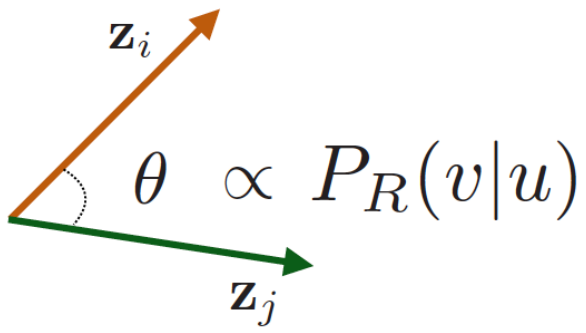
Efficiency

Do not need to consider all node pairs when training; only need to consider pairs that co-occur on random walks.



Expressiveness

Flexible stochastic definition of node similarity that incorporates both local and higher-order neighborhood information.

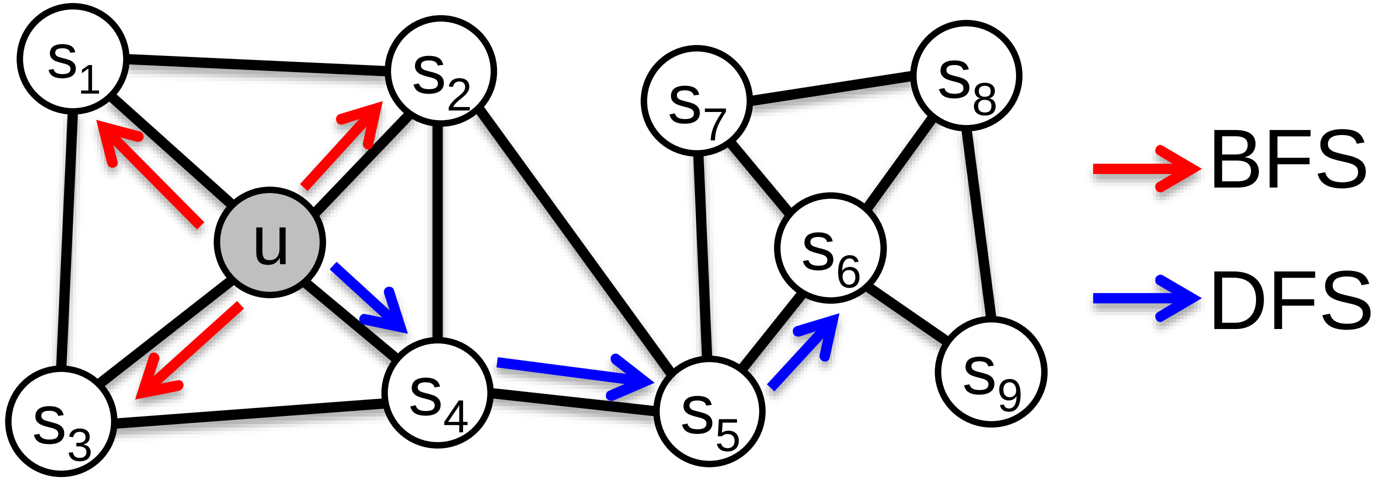


http://snap.stanford.edu/proj/embeddings-www/

node2vec: biased random walk similarity

Biased random walk to encourage:
local and **global** views

local microscopic view → breadth first search (BFS) walk
global macroscopic view → depth first search (DFS) walk



$N_{BFS}(u) = \{ s_1, s_2, s_3 \}$ Local microscopic view

$N_{DFS}(u) = \{ s_4, s_5, s_6 \}$ Global macroscopic view

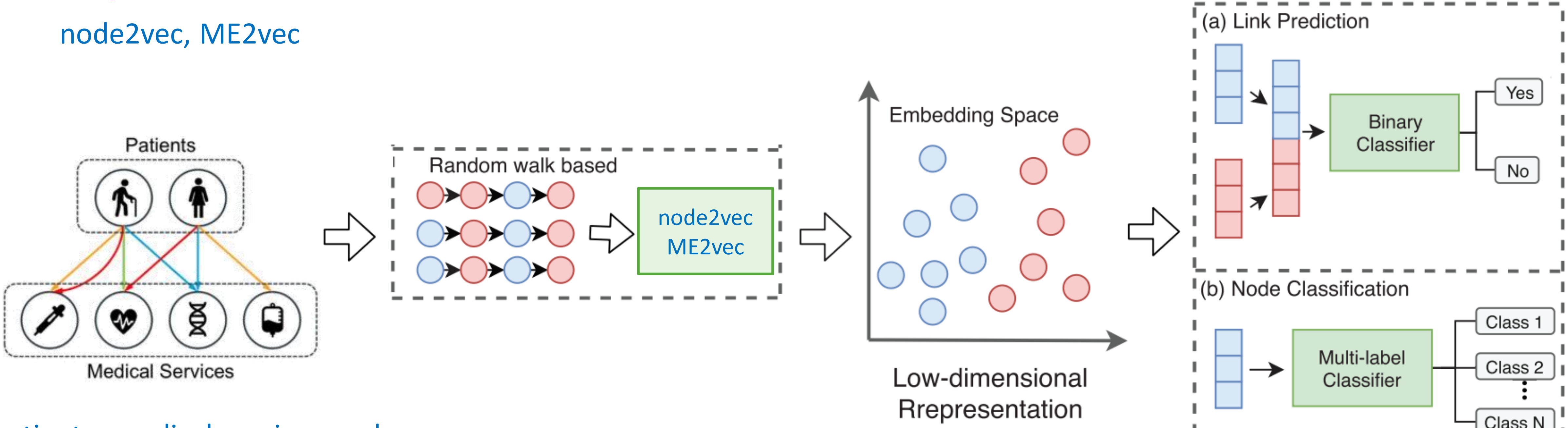
Application: Health Records

Leveraging graph-based hierarchical medical entity embedding for healthcare applications

Tong Wu et.al (2021) [Nature Scientific Reports | Advanced Analytics, IQVIA Inc]

node2vec, ME2vec

Logistic Regression



Downstream Prediction Tasks

patient – medical service graph
patient – doctor graph

patient, medical service,
doctor embedding

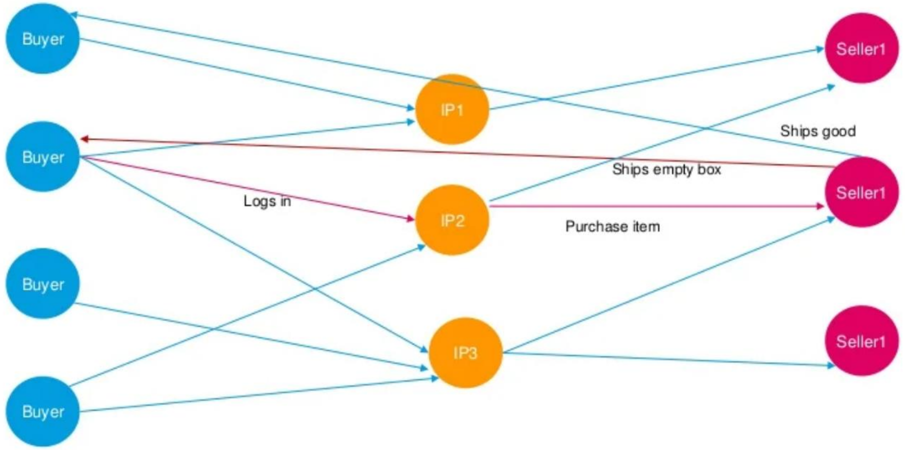
Prediction Tasks:

1. Predict patient diagnostic [node classification]
2. Predict if patient need to see a doctor [link prediction]
3. Readmission prediction [node classification]

https://www.nature.com/articles/s41598-021-85255-w.pdf
https://arxiv.org/abs/1906.05017

Application: PayPal's Collusion Fraud Prevention

PayPal



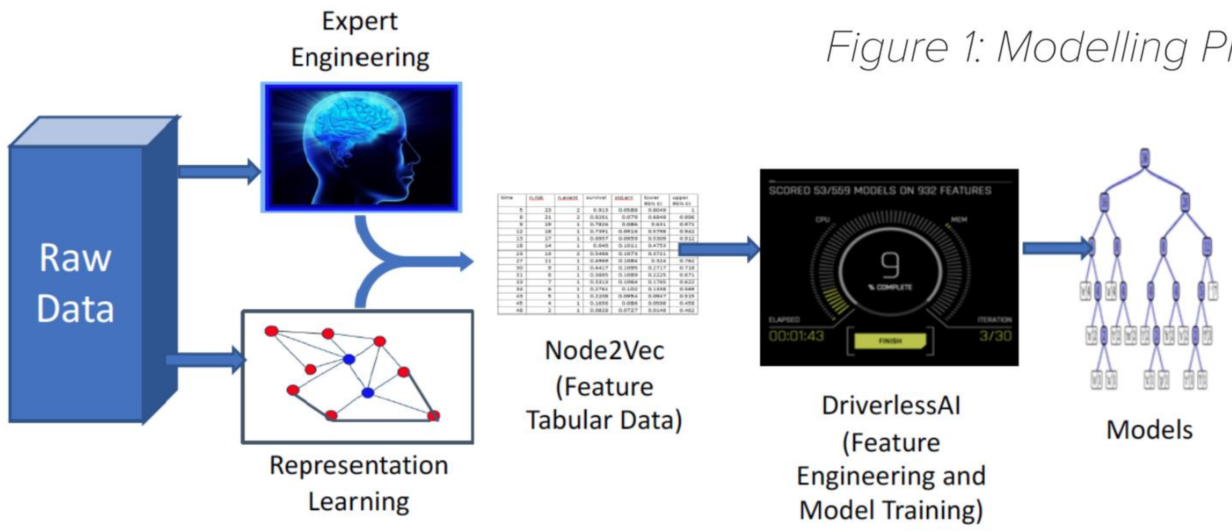


Figure 1: Modelling Pipeline

DATA

Training Data:

- Subset of one year's transactions.
- 1.5 billion edges, .5 million nodes.

Test Data:

- 3 months

Number of Features:

- 400-600

https://www.slideshare.net/Hadoop_Summit/graph-representation-learning-to-prevent-payment-collusion-fraud
https://www.h2o.ai/content/dam/h2o/en/marketing/documents/2020/01/PayPal-Customer-Case-Study-rnd2-1.pdf

Thank You