Introduction to Machine learning on graph

MANUEL DILEO

Connets LAB

Contents

- Introduction
- Classic ML lifecycle
- Shallow encoders
- Deep learning on graph
  - Data handling of graphs, preprocessing, utilities in PyG
  - Node classification with Graph Neural Networks

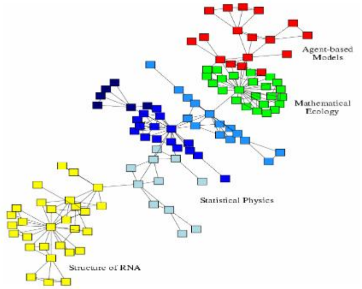
One book to rule them all

https://www.cs.mcgill.ca/~wlh/grl_book/

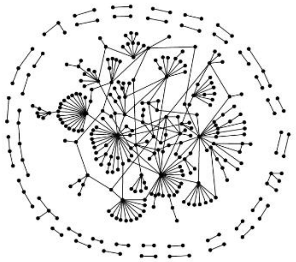
Introduction

Many Data are Networks

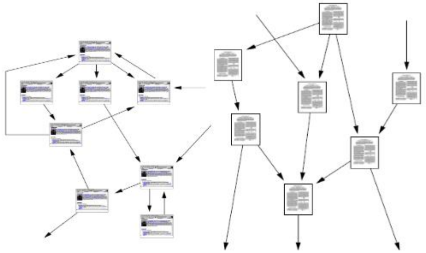
Social networks



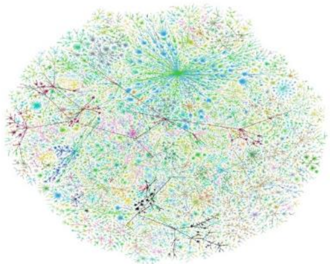
Economic networks



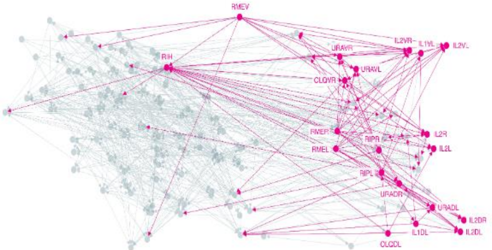
Biomedical networks



Information networks:
Web & citations



Internet



Networks of neurons

Graphs are an extremely powerful and general representation of data

Images and Text as graphs

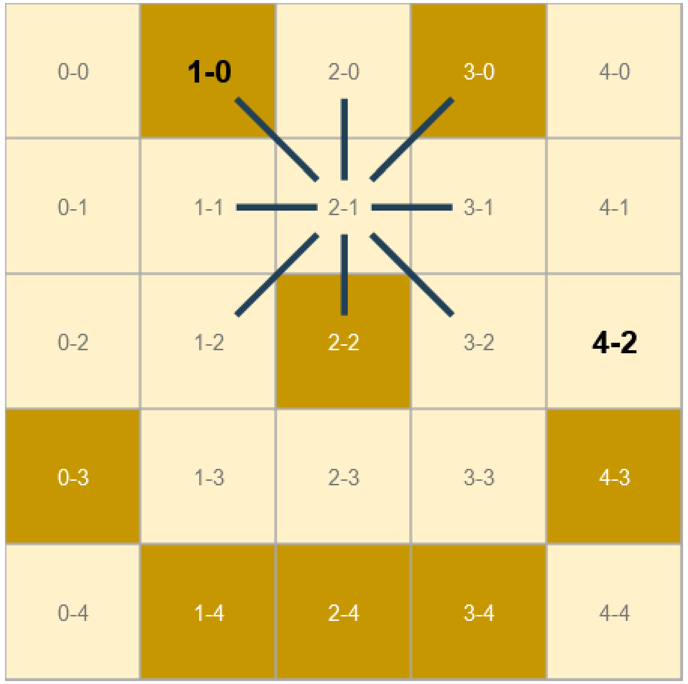
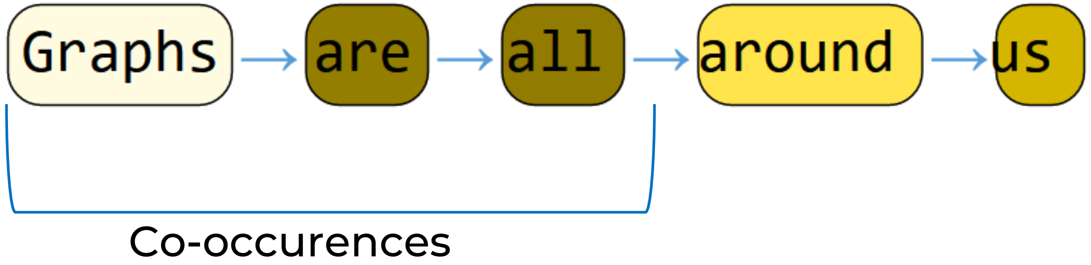


Image Pixels



Classical ML task in networks

- Node classification
- Link prediction
- Community detection
- Graph classification

How we can reason within networks?

Classical ML task in networks

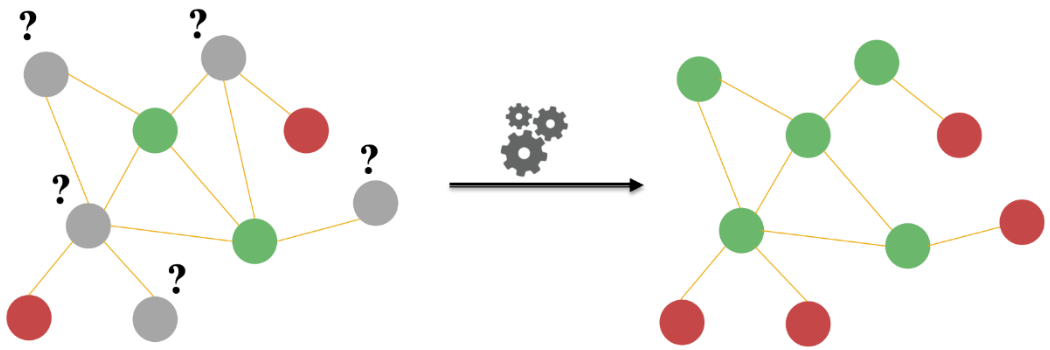
NODE CLASSIFICATION

Predict a label for a given node (e.g. the behaviour of an user in a social network)

Supervised or semi-supervised task

Network-approach as a way of improving prediction performance on classification tasks:

- Predict the topic of scientific papers
- Predict the genre of songs

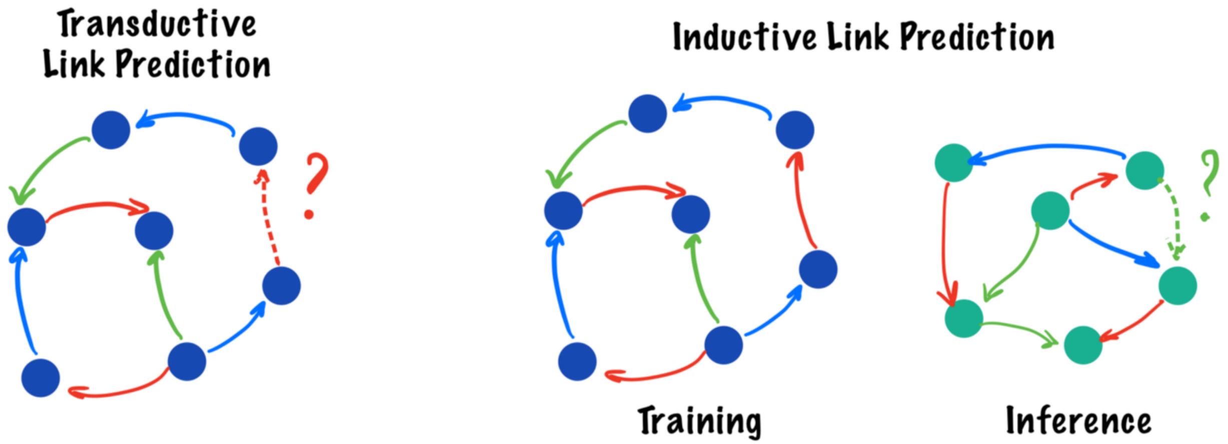


Classical ML task in networks

LINK PREDICTION

Predict wheter two nodes are linked (e.g. follow relations in a social network)

Binary classification task with unbalanced classes

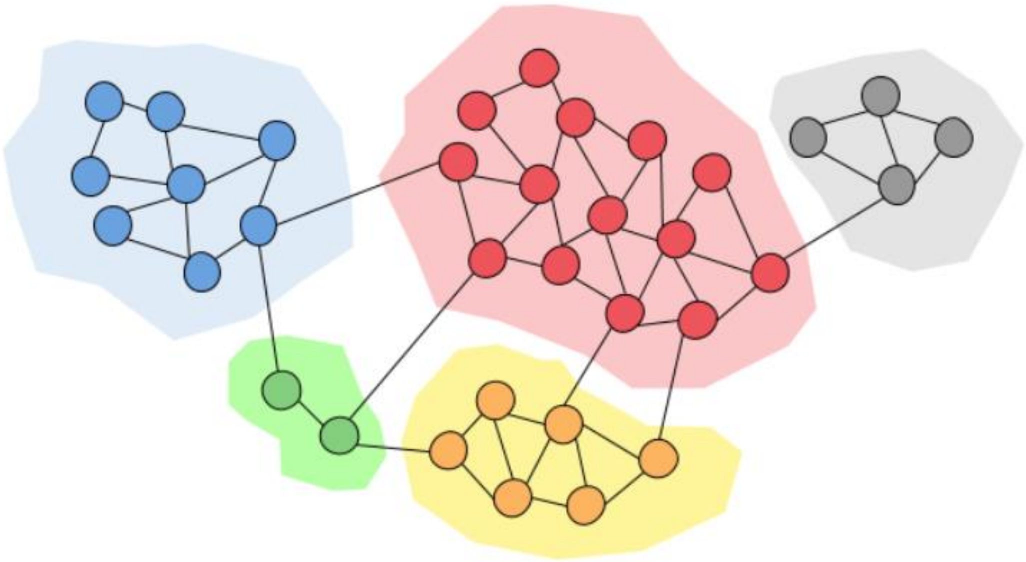


Classical ML task in networks

COMMUNITY DETECTION

Identify densely linked clusters of nodes

Unsupervised or self-supervised task

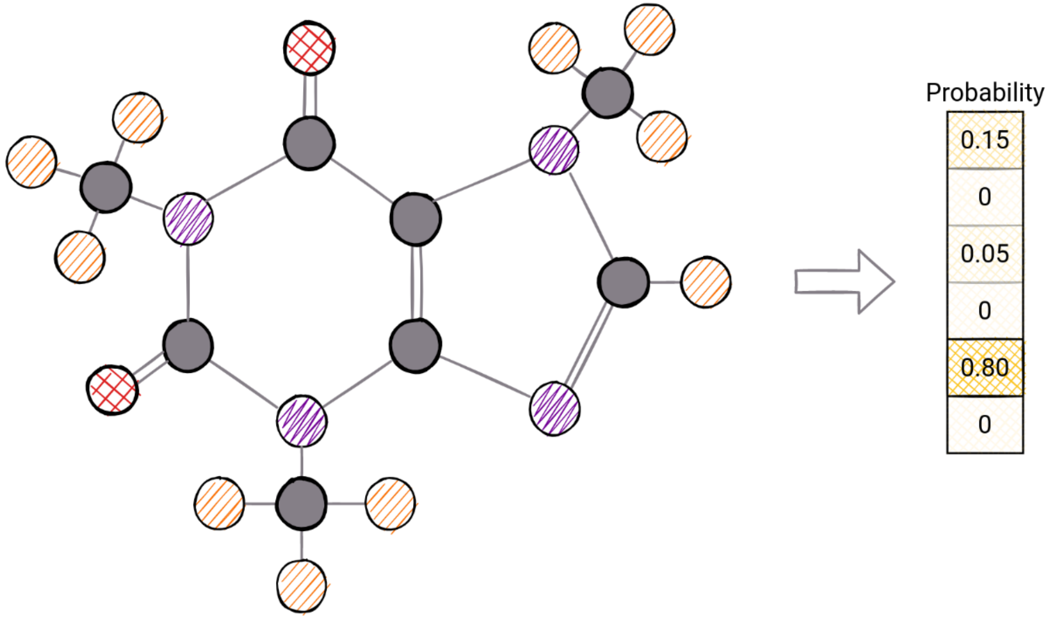


Classical ML task in networks

GRAPH CLASSIFICATION

Predict a label for a given graph in a dataset of graphs

- Disease associated to a certain brain network structure
- Role of a protein based on its molecular structure
- [...]



ML on graph methods

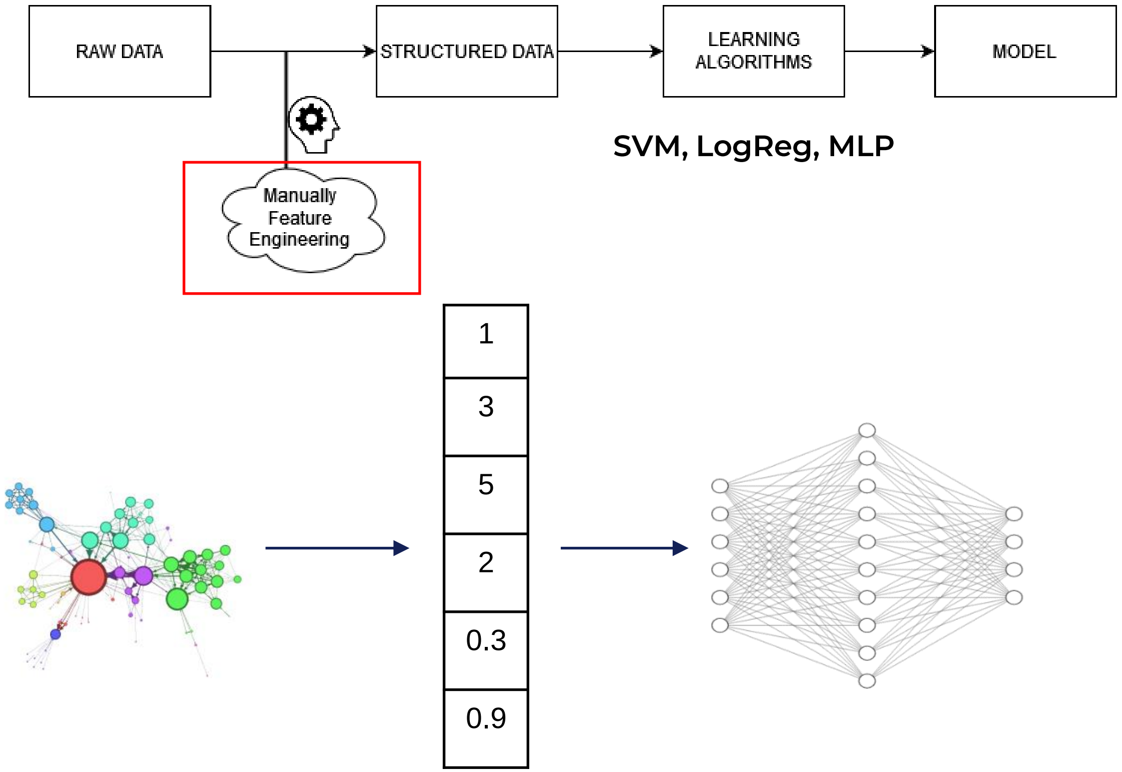
Classic ML lifecycle

FEATURE ENGINEERING

Feature engineering on the structural information of graph to obtain vectorial representation.

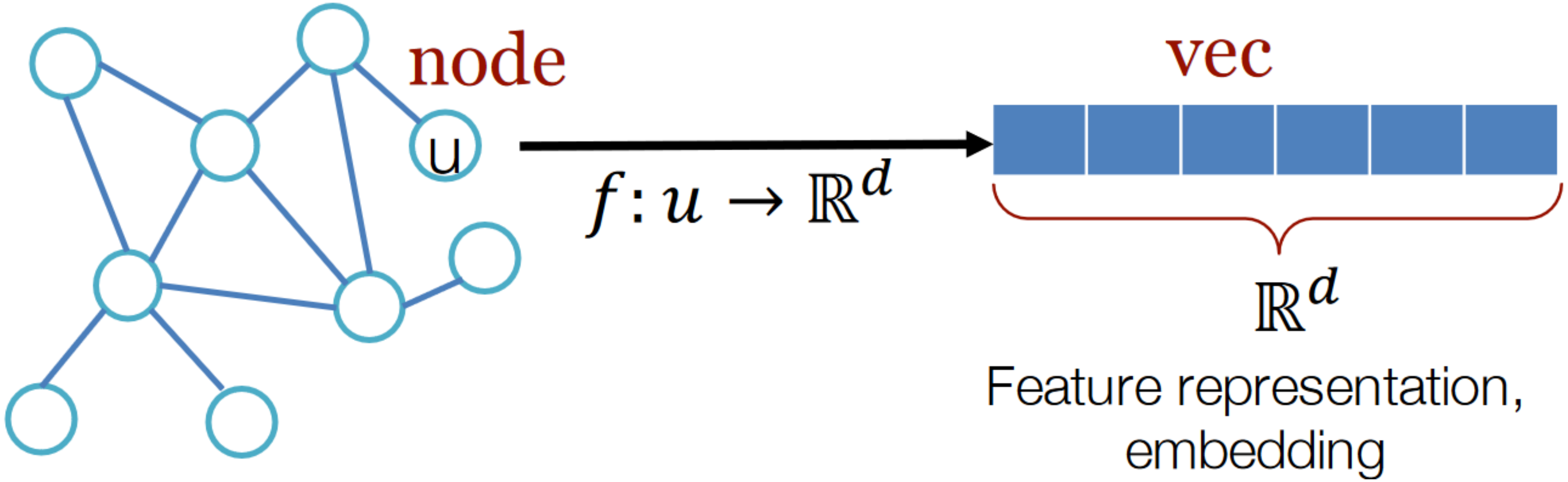
- Centrality indexes (PageRank, in/out degree, …)
- Similarity measures (neighbor jaccard's coefficient)

Long and exhausting process.



Feature learning on graphs

AUTOMATICALLY LEARN THE FEATURES

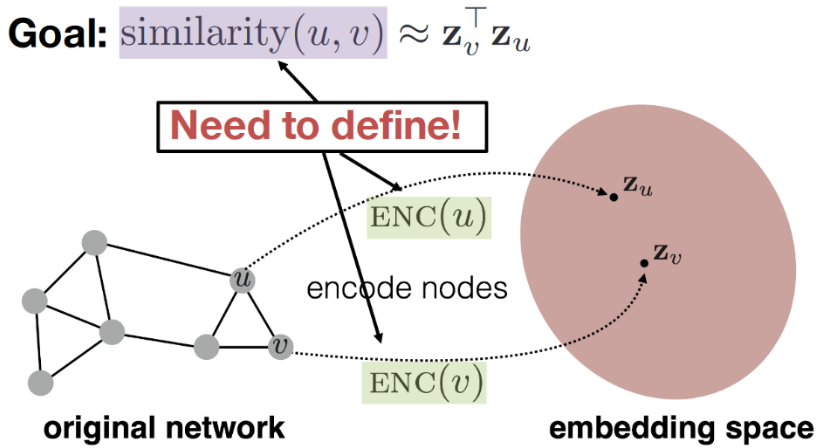


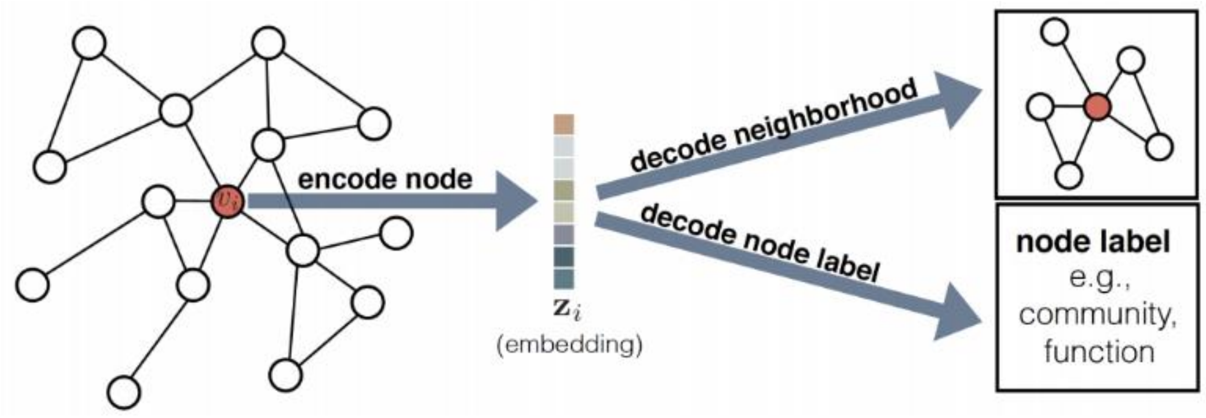
Challenges in graph computation

- Lack of consistent structure
- Node-order equivariance
  - Graphs often have no inherent ordering present amongst the nodes.
- Scalability
  - Graphs can be really large!

The Encoder-Decoder model

- Similarity function
  - measures the similarity between nodes
- Encoder function
  - generates the node embedding
- Decoder function
  - reconstructs pairwise similarity
- Loss function
  - checks the quality of the reconstruction





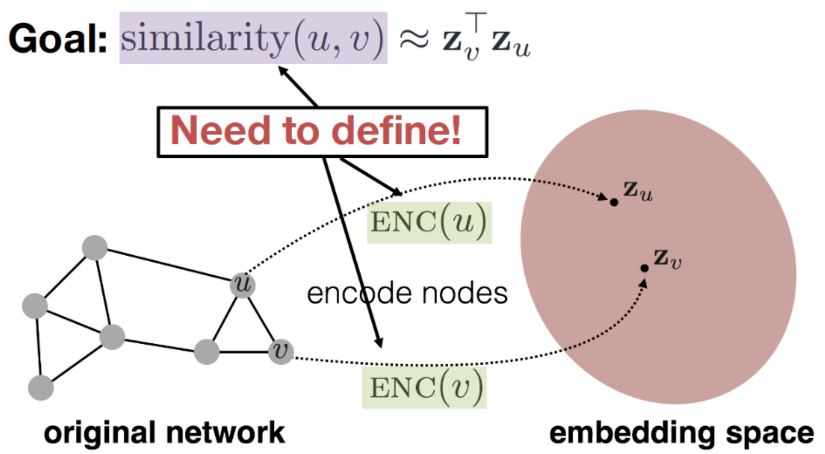
Shallow encoders

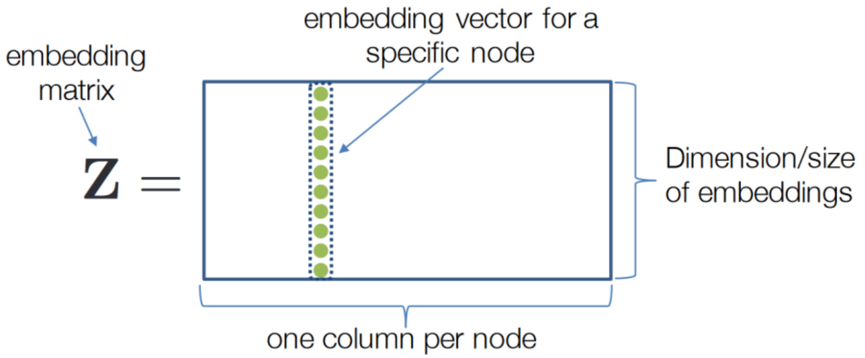
Encoder is just a lookup on an embedding matrix

Different approaches:

- Adjacency-based similarity
- Multi-hop similarity
- Random walk approacches
  - Node2Vec

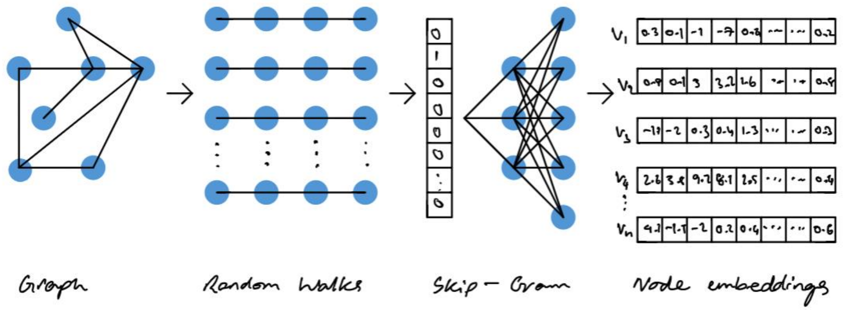
http://snap.stanford.edu/proj/embeddings-www/files/nrltutorial-part1-embeddings.pdf

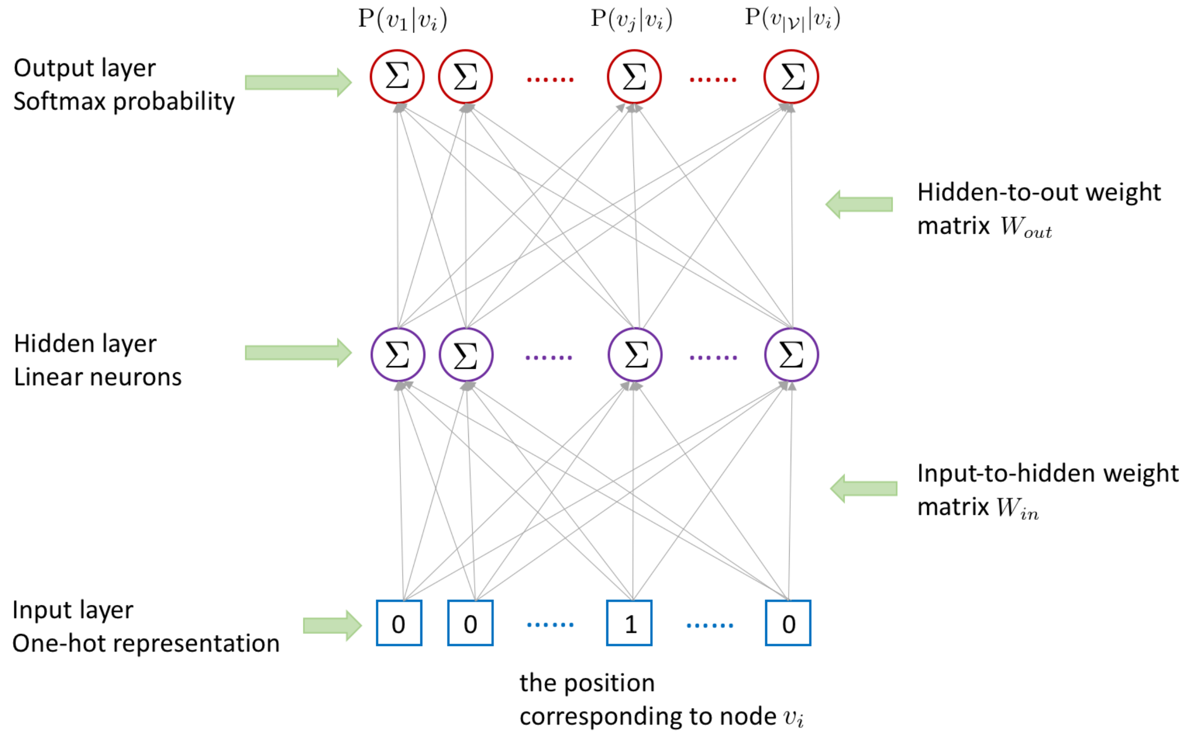




Shallow encoders

NODE2VEC





Graph Neural Networks

FROM «SHALLOW» TO «DEEP»

- Deep neural networks that can work directly on graph-structured data.
- They can incorporate node features.
  - «Shallow» encoders can work only on the structure of the graph
- The encoder is a complex function that depends on graph structure.
  - You can use encoder on unseen nodes

~~Understanding convolution on graphs~~

Message-passing layers

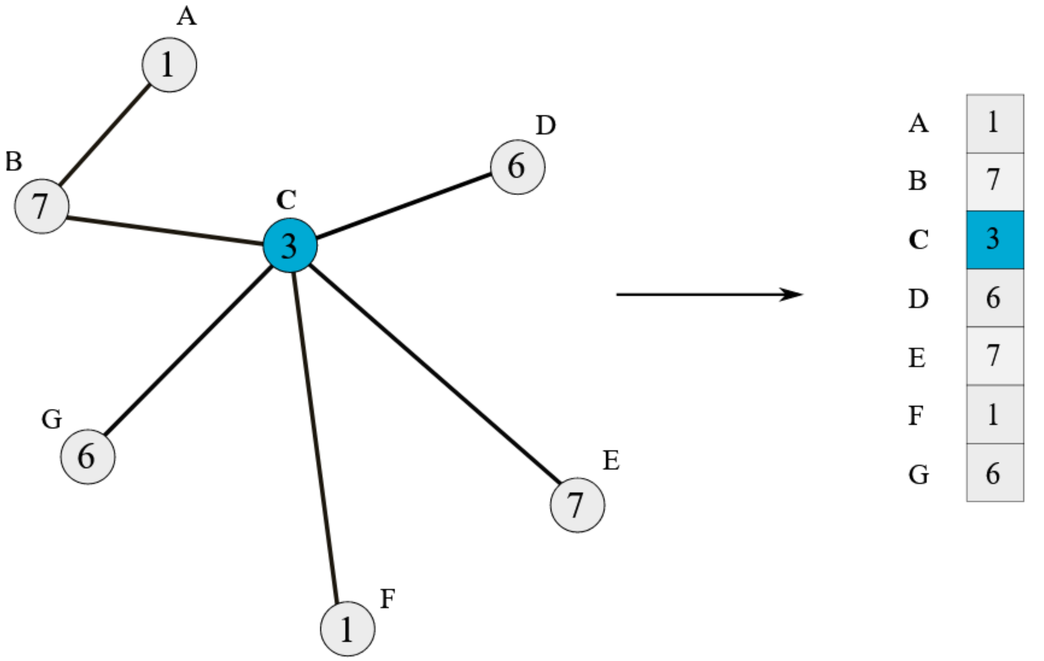
HOW CONVOLUTIONAL GNN WORKS

Consider a graph G = (V, E);

L is the Laplacian matrix L = D – A

Each node in G has a number as node feature, x is the vector of nodes feature.

What is the effect of multiplying x by the Laplacian?



Fixing a node order (indicated by the alphabets) and collecting all node features into a single vector x.

Message-passing layers

1-HOP CONVOLUTION

- Modern GNNs use **1-hop localized convolution** as hidden layers.
- These convolutions can be thought of as **'message-passing'** between adjacent nodes
- Iteratively repeat the 1-hop localized convolutions *K* times to include all nodes upto *K* hops away.

$$
\begin{aligned}
(Lx)_v &= L_v x \\
&= \sum_{u \in G} L_{vu} x_u \\
&= \sum_{u \in G} (D_{vu} - A_{vu}) x_u \\
&= D_v \, x_v - \sum_{u \in \mathcal{N}(v)} x_u
\end{aligned}
$$

Message-passing layers

AGGREGATION AND COMBINATION

- We can think of 1-hop localized convolution as arising of two steps:
  - Aggregating over immediate neighbour features x_u
  - Combining with the node's own feature x_v

What if we consider different kinds of 'aggregation' and 'combination' steps?

$$
\begin{aligned}
(Lx)_v &= L_v x \\
&= \sum_{u \in G} L_{vu} x_u \\
&= \sum_{u \in G} (D_{vu} - A_{vu}) x_u \\
&= \boxed{D_v \, x_v -} \boxed{\sum_{u \in \mathcal{N}(v)} x_u}
\end{aligned}
$$

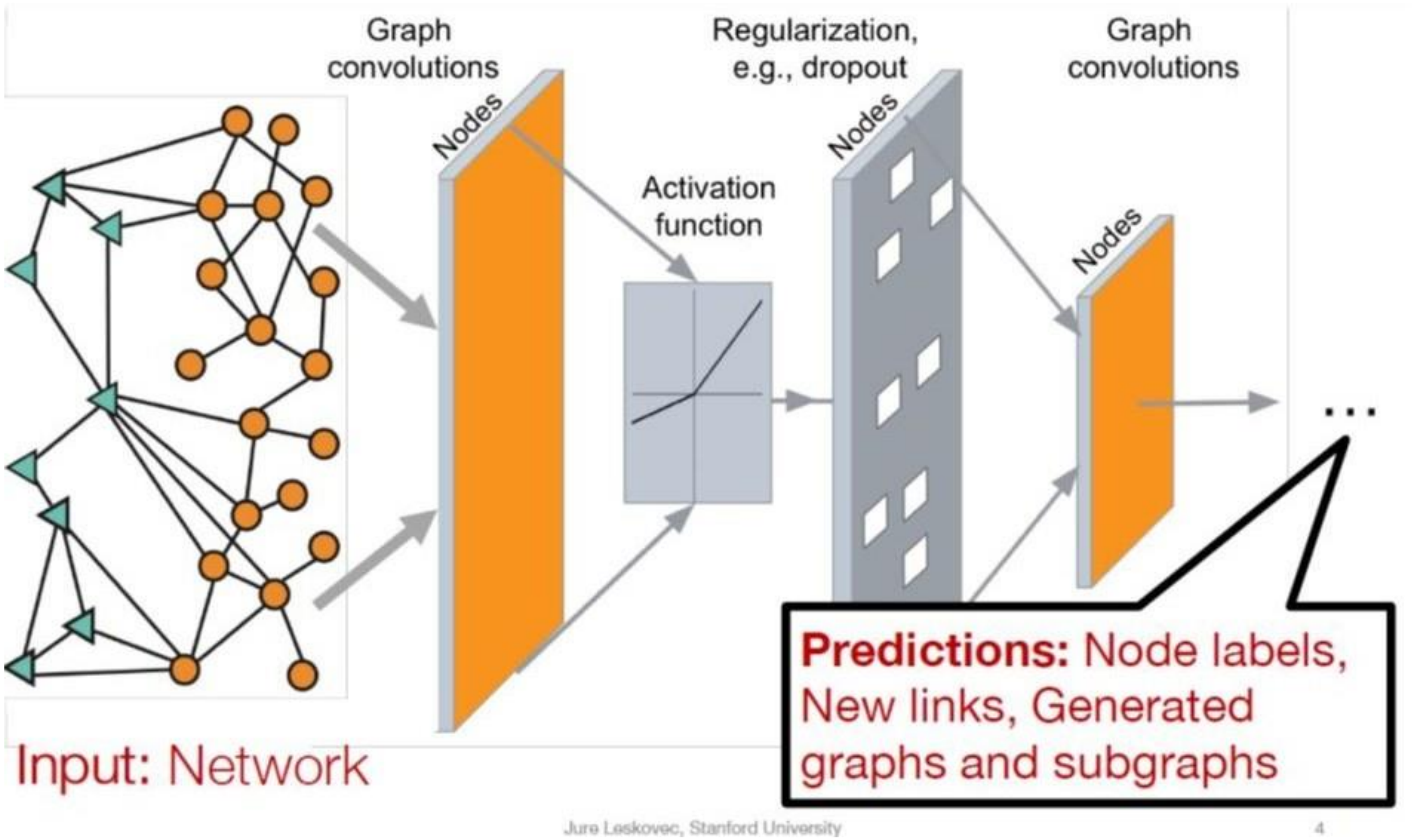
Popular message-passing layers

https://distill.pub/2021/understanding-gnns/#modern-gnns

GNN architectures

Node feature matrix X

Adjacency matrix edge_index

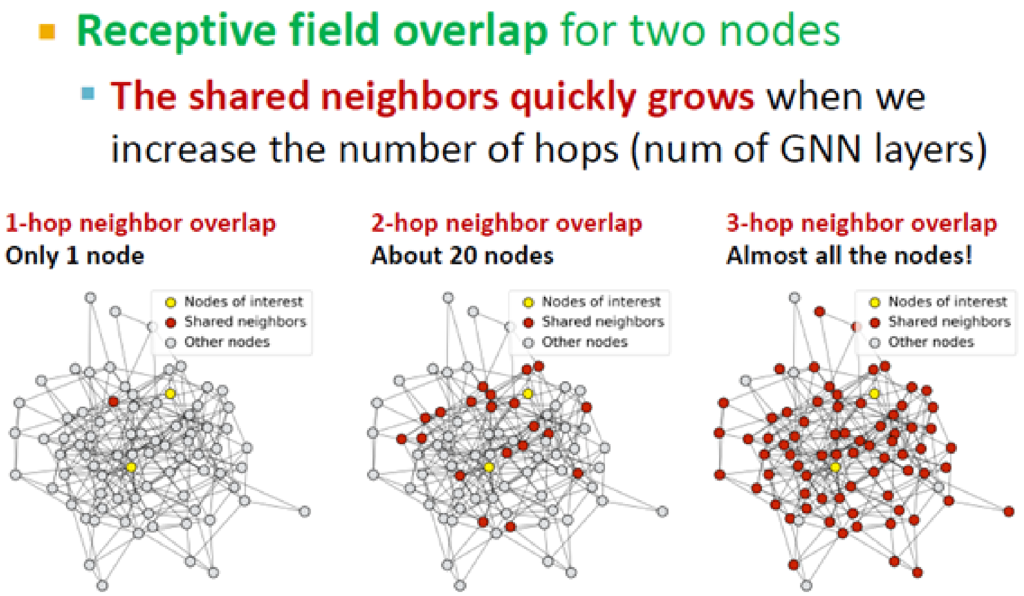


Let’s play

https://distill.pub/2021/understanding-gnns/#interactive

Stacking GNN layers

- High depth ≠ high expressiveness
- The depth influences the «receptive field»
- **Oversmoothing problem**: all the node embeddings converge to the same value



Complexity and expressiveness rely on the design of the single layers

GNN References

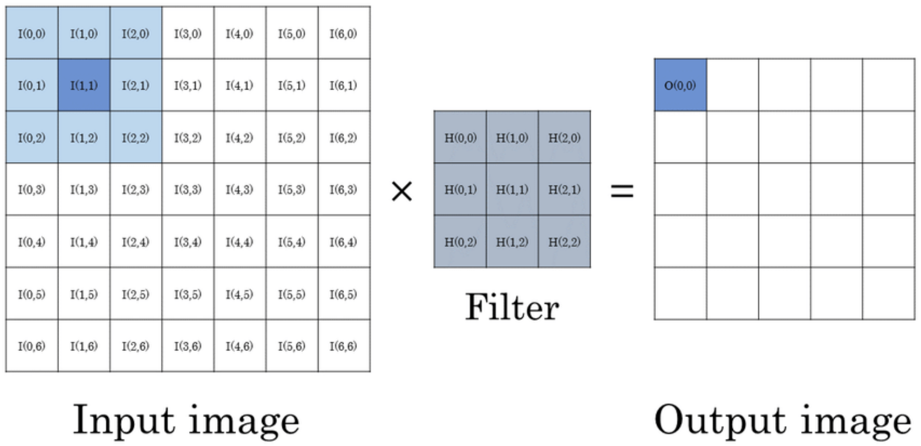
- https://distill.pub/2021/understanding-gnns/
- http://web.stanford.edu/class/cs224w/
- https://pytorch-geometric.readthedocs.io/en/latest/notes/colabs.html
- https://twitter.com/omarsar0/status/1490276912601653248?s=20&t=qtq0yg-e1leFGUwRatw5Bg
- https://arxiv.org/pdf/1901.00596.pdf
- https://towardsdatascience.com/graph-ml-in-2022-where-are-we-now-f7f8242599e0
- https://www.deeplearningbook.org/

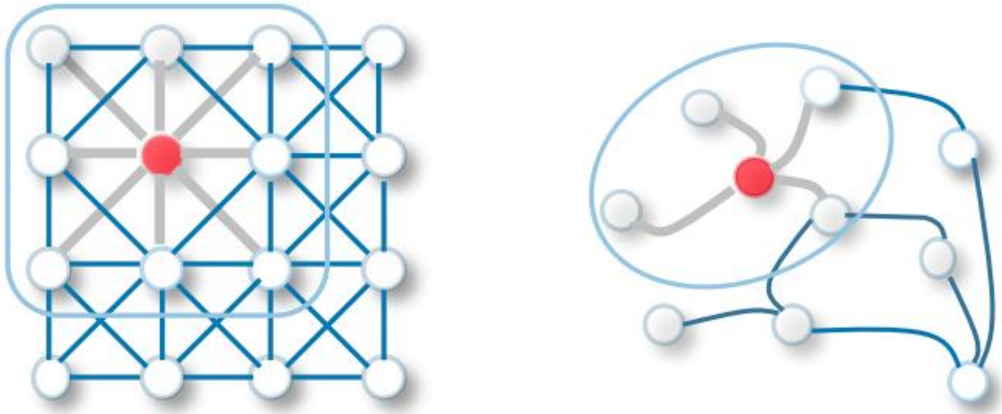
Thanks for your attention

Understanding convolution on graphs

Problem

HOW TO EXTEND CONVOLUTION ON GRAPH?





Convolutions in CNNs are inherently localized. Neighbours participating in the convolution at the center pixel are highlighted in light blue.

GNNs can perform localized convolutions mimicking CNNs. Hover over a node to see its immediate neighbourhood highlighted on the left. The structure of this neighbourhood changes from node to node.

Polynomial filters on graphs

https://distill.pub/2021/understanding-gnns/#polynomial-filters

Embedding computation

If we have *K* different polynomial filter layers, the *k-th* of which has its own learnable weights *w(k)*, we would perform the following computation:

Start with the original features.

$$h^{(0)} = x$$

Then iterate, for $k = 1, 2, \ldots$ upto K:

$$p^{(k)} = p_{w^{(k)}}(L)$$

Compute the matrix $p^{(k)}$ as the polynomial defined by the filter weights $w^{(k)}$ evaluated at L.

$$g^{(k)} = p^{(k)} \times h^{(k-1)}$$

Multiply $p^{(k)}$ with $h^{(k-1)}$: a standard matrix-vector multiply operation.

$$h^{(k)} = \sigma\left(g^{(k)}\right)$$

Apply a non-linearity σ to $g^{(k)}$ to get $h^{(k)}$.

Color Codes:

- Computed node embeddings.
- Learnable parameters.

Parameter-sharing: reuse the same filter weights across different nodes

Complex GNN architectures

IDEAS

- Add non GCN layers
  - Dense layers as pre/post processing layers
- More complex GNN layers
  - Incorporate modern DL modules (e.g. Dropout, BatchNorm)
  - 3-layer MLP as aggregation function
- Skip connection: mixture of embeddings to aggregate