Edge Analysis

Antonio Longa[1,2]

MobS[1] Lab, Fondazione Bruno Kessler,Trento, Italy
SML[2] Lab, University of Trento, Italy

TABLE OF CONTENTS

01 Problem & Motivation

The input graph can have **missing** informations.

01 Problem & Motivation

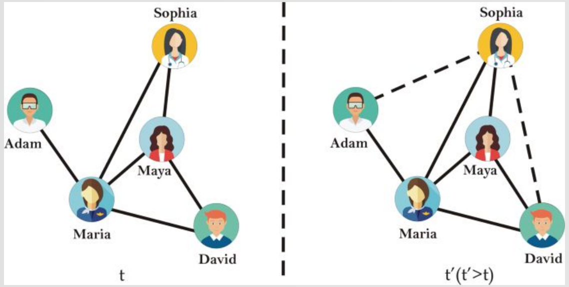
The input graph can have **missing** informations

- Edges
- Labels
- Weights
- etc...

01 Problem & Motivation

The input graph can have **missing** informations

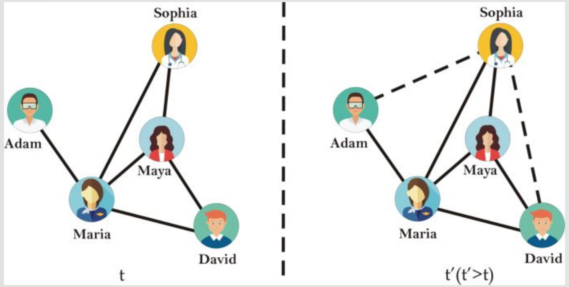
- **Edges**
- Labels
- Weights
- etc...

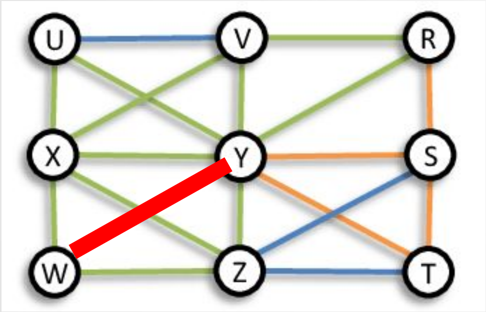


01 Problem & Motivation

The input graph can have **missing** informations

- Edges
- **Labels**
- Weights
- etc...



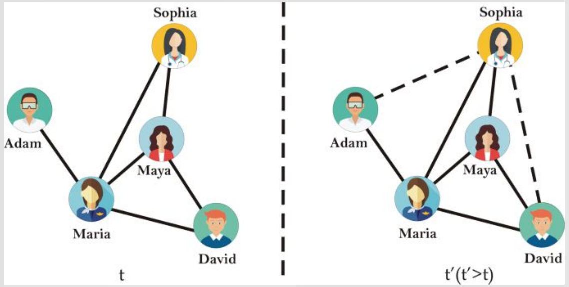


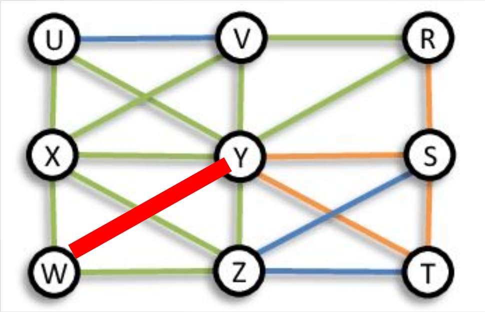
Witch color?

01 Problem & Motivation

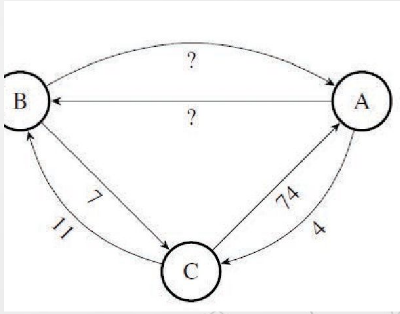
The input graph can have **missing** informations

- Edges
- Labels
- **Weights**
- etc...





Witch color?



02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

WHY IT IS IMPORTANT?

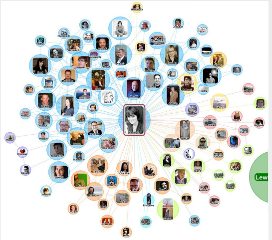
- Recommendation systems

02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

WHY IT IS IMPORTANT?

- Recommendation systems
- Privacy control on social networks

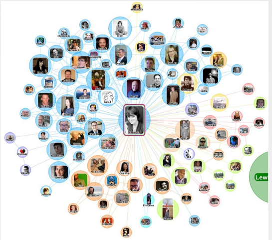


02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

WHY IT IS IMPORTANT?

- Recommendation systems
- Privacy control on social networks
- Influence detection

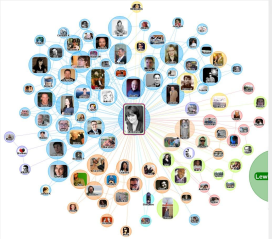


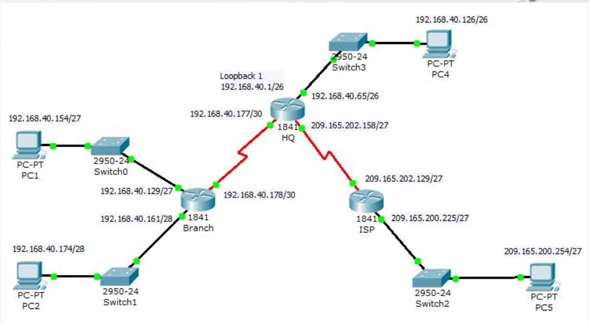
02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

WHY IT IS IMPORTANT?

- Recommendation systems
- Privacy control on social networks
- Influence detection
- Routing networks
- Etc …



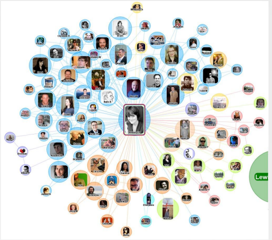


02 Link prediction

Given an input graph G, and two nodes u and v, predict if there is an edge between u and v.

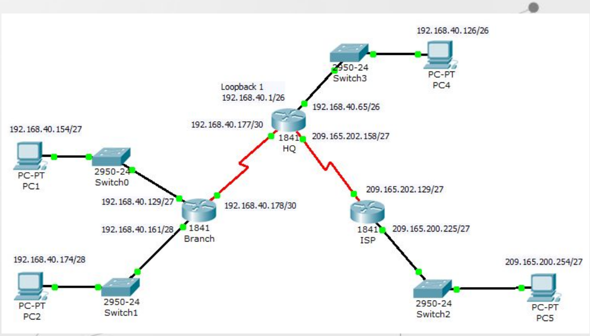
WHY IT IS IMPORTANT?

- Recommendation systems
- Privacy control on social networks
- Influence detection
- Routing networks
- Etc …



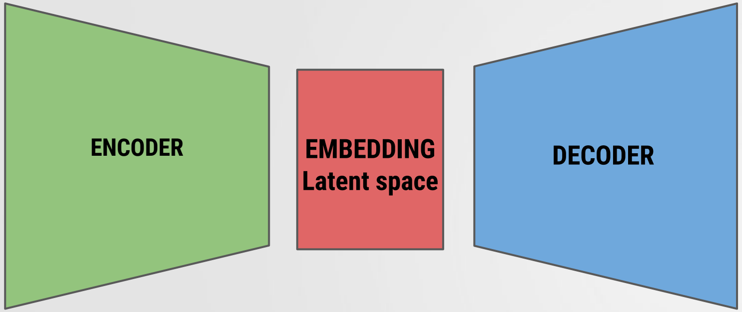
HOW TO SOLVE IT

- Topology based methods
- Node attribute based methods
- Based on embeddings
- Probabilistics relationship models
- Etc …



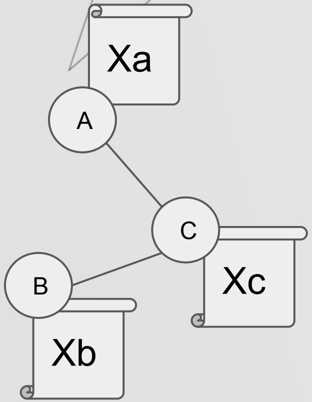
03 GAE for link prediction

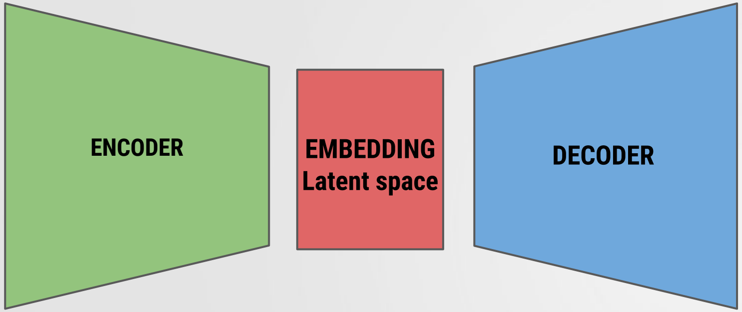
Recap GAE



03 GAE for link prediction

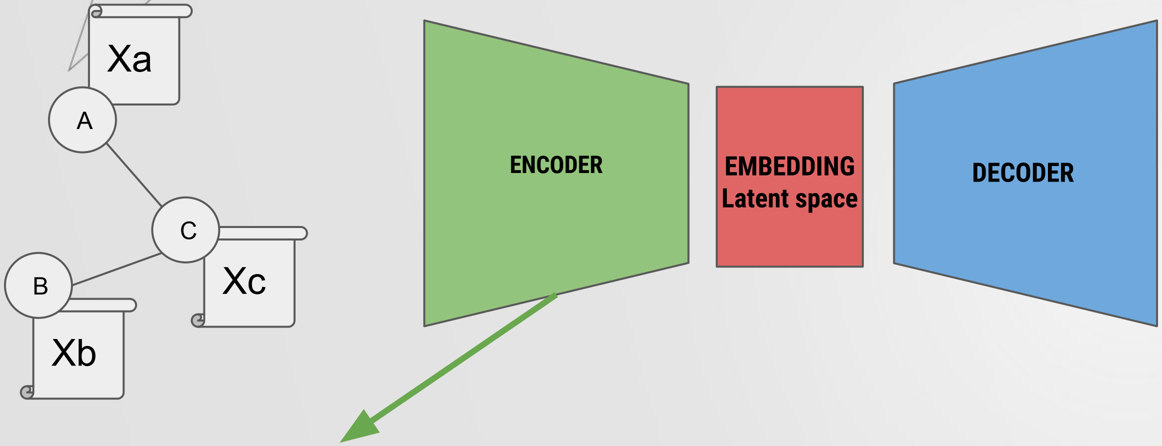
Recap GAE





03 GAE for link prediction

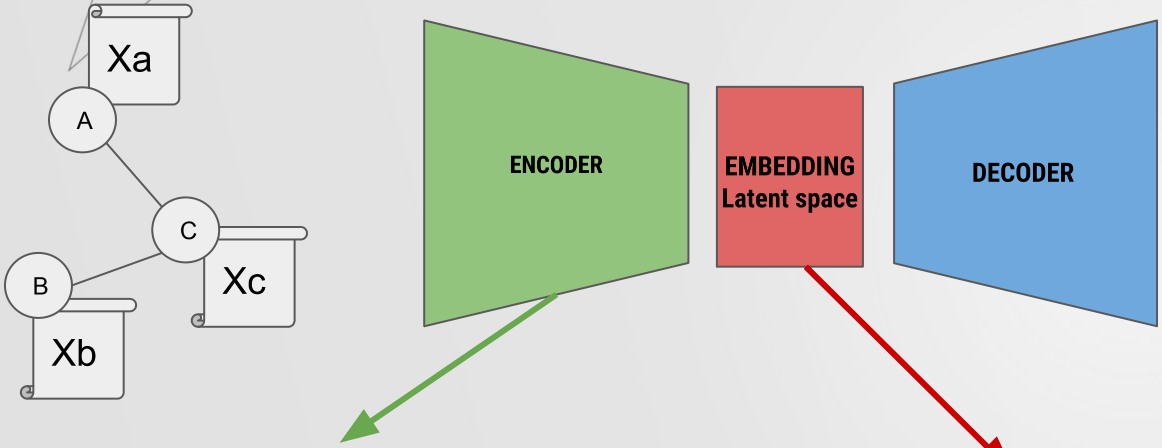
Recap GAE



ONE convolutional
Graph neural network

03 GAE for link prediction

Recap GAE

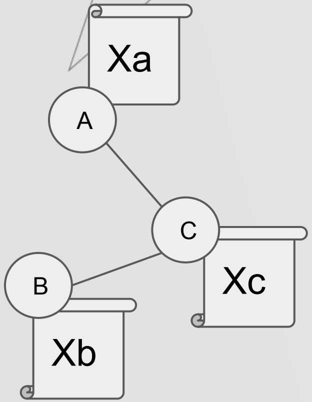


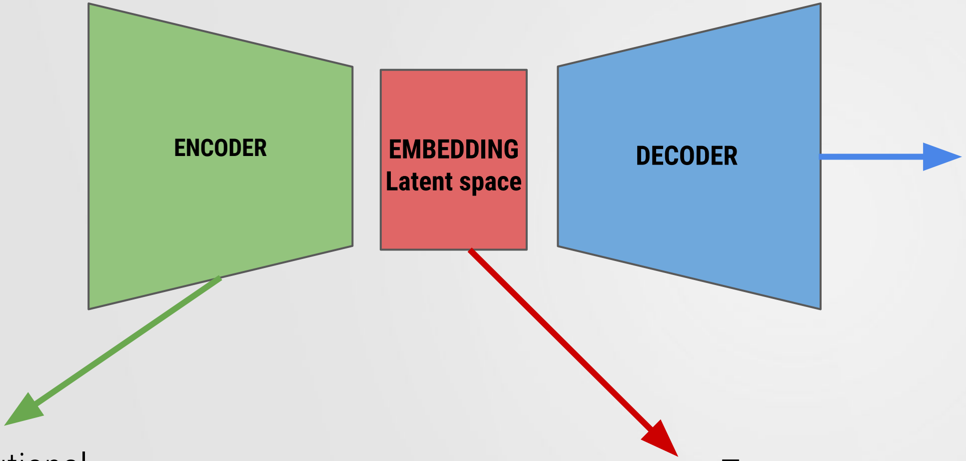
ONE convolutional
Graph neural network

$$Z = \bar{X}$$

03 GAE for link prediction

Recap GAE





Inner product
Between latent variable Z

ONE convolutional
Graph neural network

$$Z = \bar{X}$$

03 GAE for link prediction

But..

What is the difference between the GAE for node embedding?

03 GAE for link prediction

But..

What is the difference between the GAE for node embedding?

The LOSS function

03 GAE for link prediction

GAE for node embedding

recon_loss (z, pos_edge_index, neg_edge_index=None) [source]

Given latent variables `z`, computes the binary cross entropy loss for positive edges `pos_edge_index` and negative sampled edges.

PARAMETERS:

- **z** (*Tensor*) – The latent space $\mathbf{Z}$.
- **pos_edge_index** (*LongTensor*) – The positive edges to train against.
- **neg_edge_index** (*LongTensor, optional*) – The negative edges to train against. If not given, uses negative sampling to calculate negative edges. (default: `None`)

03 GAE for link prediction

GAE for node embedding

recon_loss (z, pos_edge_index, neg_edge_index=None) [source]

Given latent variables `z`, computes the binary cross entropy loss for positive edges `pos_edge_index` and negative sampled edges.

PARAMETERS:

- **z** (*Tensor*) – The latent space $\mathbf{Z}$.
- **pos_edge_index** (*LongTensor*) – The positive edges to train against.
- **neg_edge_index** (*LongTensor, optional*) – The negative edges to train against. If not given, uses negative sampling to calculate negative edges. (default: `None`)

03 GAE for link prediction

GAE for node embedding

recon_loss (z, pos_edge_index, neg_edge_index=None) [source]

Given latent variables `z`, computes the binary cross entropy loss for positive edges `pos_edge_index` and negative sampled edges.

PARAMETERS:

- **z** (*Tensor*) – The latent space $\mathbf{Z}$.
- **pos_edge_index** (*LongTensor*) – The positive edges to train against.
- **neg_edge_index** (*LongTensor, optional*) – The negative edges to train against. If not given, uses negative sampling to calculate negative edges. (default: `None`)

GAE for link prediction

```
link_labels = get_link_labels(data.train_pos_edge_index, neg_edge_index)
loss = F.binary_cross_entropy_with_logits(link_logits, link_labels)
loss.backward()
optimizer.step()
```

03 GAE for link prediction

GAE for node embedding

recon_loss (z, pos_edge_index, neg_edge_index=None) [source]

Given latent variables `z`, computes the binary cross entropy loss for positive edges `pos_edge_index` and negative sampled edges.

PARAMETERS:

- **z** (*Tensor*) – The latent space $\mathbf{Z}$.
- **pos_edge_index** (*LongTensor*) – The positive edges to train against.
- **neg_edge_index** (*LongTensor, optional*) – The negative edges to train against. If not given, uses negative sampling to calculate negative edges. (default: `None`)

GAE for link prediction

```
link_labels = get_link_labels(data.train_pos_edge_index, neg_edge_index)
loss = F.binary_cross_entropy_with_logits(link_logits, link_labels)
loss.backward()
optimizer.step()
```

03 GAE for link prediction

GAE for node embedding

Binary cross entropy loss

GAE for link prediction

Binary cross entropy with logits loss

03 GAE for link prediction

GAE for node embedding

Binary cross entropy loss

$$\ell(x, y) = L = \{l_1, \dots, l_N\}^\top$$

GAE for link prediction

Binary cross entropy with logits loss

03 GAE for link prediction

GAE for node embedding

Binary cross entropy loss

$$\ell(x, y) = L = \{l_1, \dots, l_N\}^\top$$

$$l_n = -w_n \left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n) \right]$$

GAE for link prediction

Binary cross entropy with logits loss

03 GAE for link prediction

GAE for node embedding

Binary cross entropy loss

$$\ell(x, y) = L = \{l_1, \dots, l_N\}^\top$$

$$l_n = -w_n \left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n) \right]$$

GAE for link prediction

Binary cross entropy with logits loss

$$\ell(x, y) = L = \{l_1, \dots, l_N\}^\top$$

03 GAE for link prediction

GAE for node embedding

Binary cross entropy loss

$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top$$

$$l_n = -w_n \left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n) \right]$$

GAE for link prediction

Binary cross entropy with logits loss

$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top$$

$$l_n = -w_n \left[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n)) \right]$$

03 GAE for link prediction

Jupyter - notebook

04 Label prediction

Given an input graph G, and two nodes u and v, predict the label of the edge between u and v.

04 Label prediction

Given an input graph G, and two nodes u and v, predict the label of the edge between u and v.

WHY IT IS IMPORTANT?

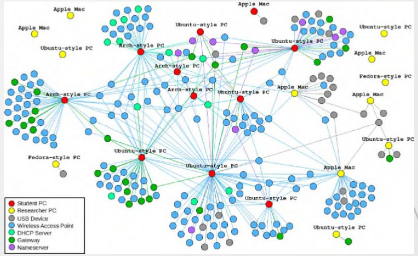
- Human mobility forecast

04 Label prediction

Given an input graph G, and two nodes u and v, predict the label of the edge between u and v.

WHY IT IS IMPORTANT?

- Human mobility forecast
- Type of relationship in social networks

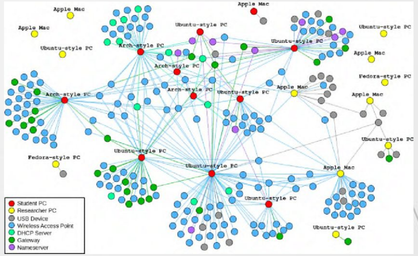


04 Label prediction

Given an input graph G, and two nodes u and v, predict the label of the edge between u and v.

WHY IT IS IMPORTANT?

- Human mobility forecast
- Type of relationship in social networks
- Traffic congestion

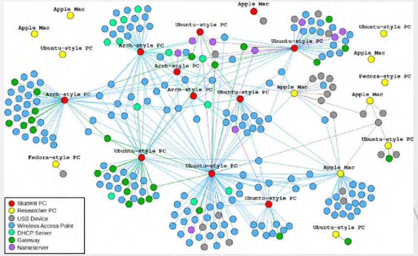


04 Label prediction

Given an input graph G, and two nodes u and v, predict the label of the edge between u and v.

WHY IT IS IMPORTANT?

- Human mobility forecast
- Type of relationship in social networks
- Traffic congestion



05 Node2Vec for Label Prediction

Jupyter - notebook