Link Prediction

Supervised vs unsupervised

The supervised learner doesn’t know much

The unsupervised learner knows what it is doing

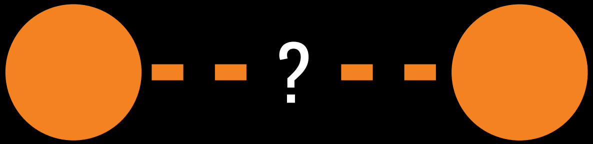
Supervised learning tasks with networks

- Node classification
  - Given a network (e.g. friendship network) and some labels (e.g. political party). Can we predict the labels of a node from the labels of their neighbours?

- Graph classification
  - Given many networks (e.g. ego-networks, brain networks) and outcomes (e.g. political party, mental disorders). Can we predict the outcomes from the topology of the network?

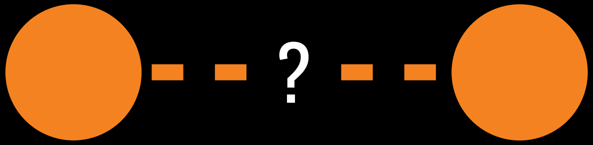
- Link prediction
  - Given a network (e.g. friendship network) and optionally some metadata (e.g. political party). Can we predict which links we are missing (or will be created)?

Link prediction



Does this link exist?

Link prediction



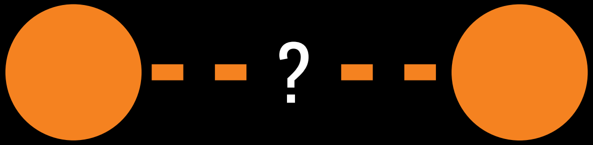
Does this link exist?

Many tasks

1. Model Validation

- Observe part of the adjacency matrix (fit model)
- Predict held out entries (cross validation)

Link prediction



Does this link exist?

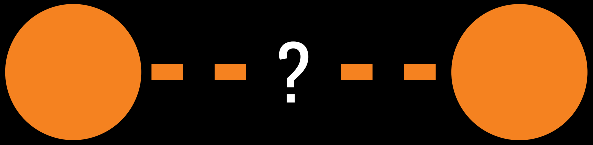
Many tasks

1. Model Validation

2. De-noising / network reconstruction

- Real-world data are noisy / contain errors

Link prediction



Does this link exist?

Many tasks

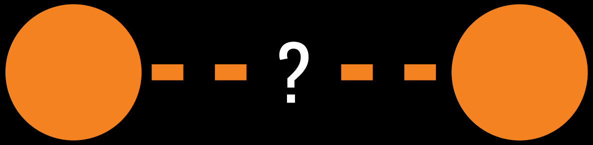
1. Model Validation

2. De-noising / network reconstruction

3. Predict missing links

- Observed edges are assumed correct
- Predict which unobserved edges exist

Link prediction



Does this link exist?

Many tasks

1. Model Validation

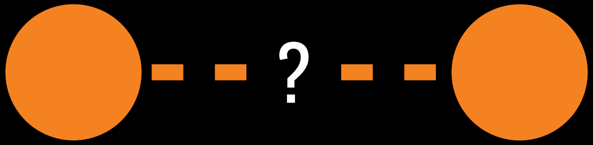
2. De-noising / network reconstruction

3. Predict missing links

4. Predict future links

- Observe the adjacency matrix at time (t)
- Predict edges in time (t+1)

Link prediction



Does this link exist?

Many tasks

1. Model Validation

2. De-noising / network reconstruction

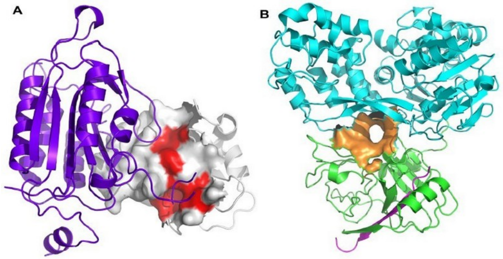
3. Predict missing links

4. Predict future links

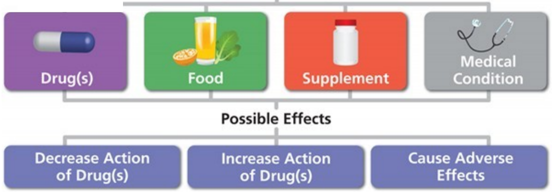
Applications

facebook LinkedIn twitter

Suggesting social and professional connections

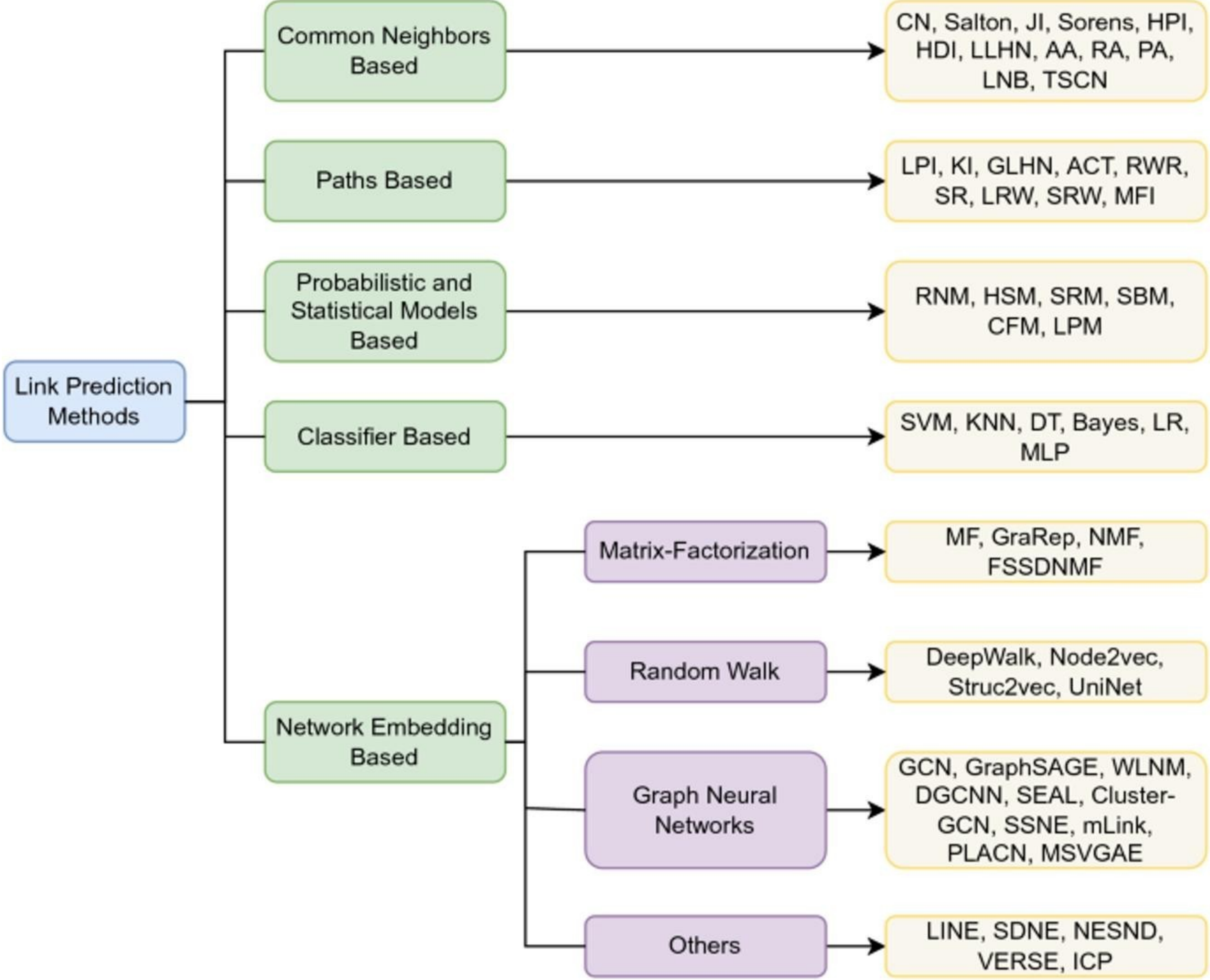


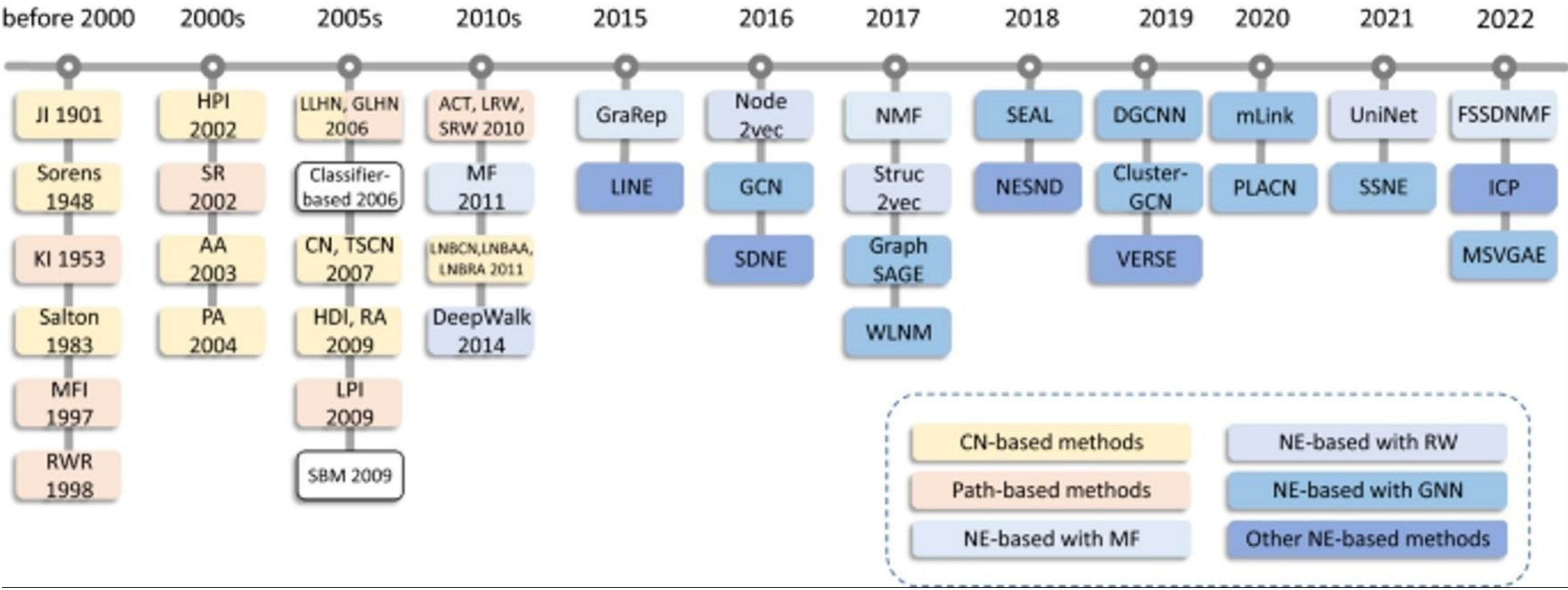
Predicting biological interactions



amazon NETFLIX

Recommending products and services


Link Prediction Methods
Common Neighbors Based
CN, Salton, JI, Sorens, HPI, HDI, LLHN, AA, RA, PA, LNB, TSCN
Paths Based
LPI, KI, GLHN, ACT, RWR, SR, LRW, SRW, MFI
Probabilistic and Statistical Models Based
RNM, HSM, SRM, SBM, CFM, LPM
Classifier Based
SVM, KNN, DT, Bayes, LR, MLP
Network Embedding Based
Matrix-Factorization
MF, GraRep, NMF, FSSDNMF
Random Walk
DeepWalk, Node2vec, Struc2vec, UniNet
Graph Neural Networks
GCN, GraphSAGE, WLNM, DGCNN, SEAL, Cluster-GCN, SSNE, mLink, PLACN, MSVGAE
Others
LINE, SDNE, NESND, VERSE, ICP



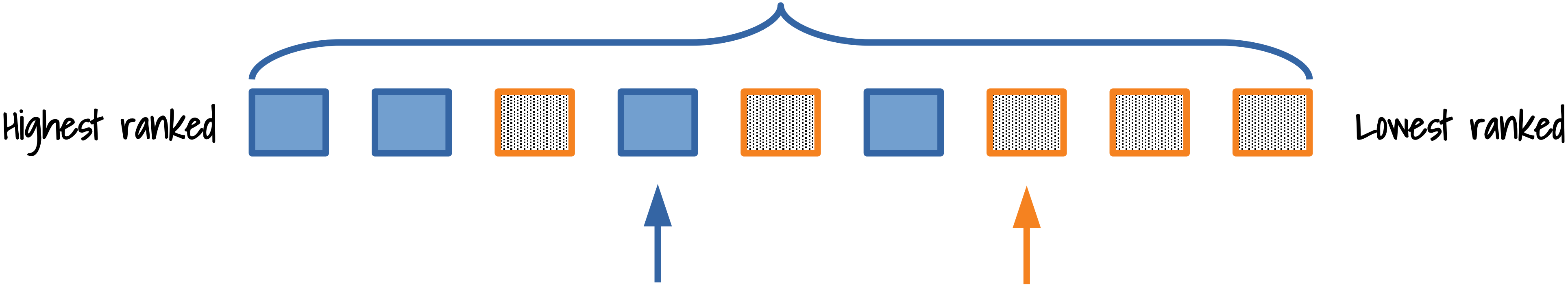
Link Prediction on Complex Networks: An Experimental Survey; Wu, Song, Ge, and Ge (2022)

Predicting missing links

Goal: Rank all non-edges according to how likely they are to exist

Assessed using measures such as AUC

AUC = Area Under the Curve



Local heuristics

Based on similarity of node connections

$\Gamma(x)$ ← neighbours of x

k_x ← degree of x

Similar neighbours

Common neighbours

$$s_{xy}^{\mathrm{CN}} = |\Gamma(x) \cap \Gamma(y)|,$$

$\Gamma(x)$ ← neighbours of x

Similar neighbours

Jaccard similarity

$$s_{xy}^{\mathrm{Jaccard}} = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|},$$

$\Gamma(x)$ ← neighbours of x

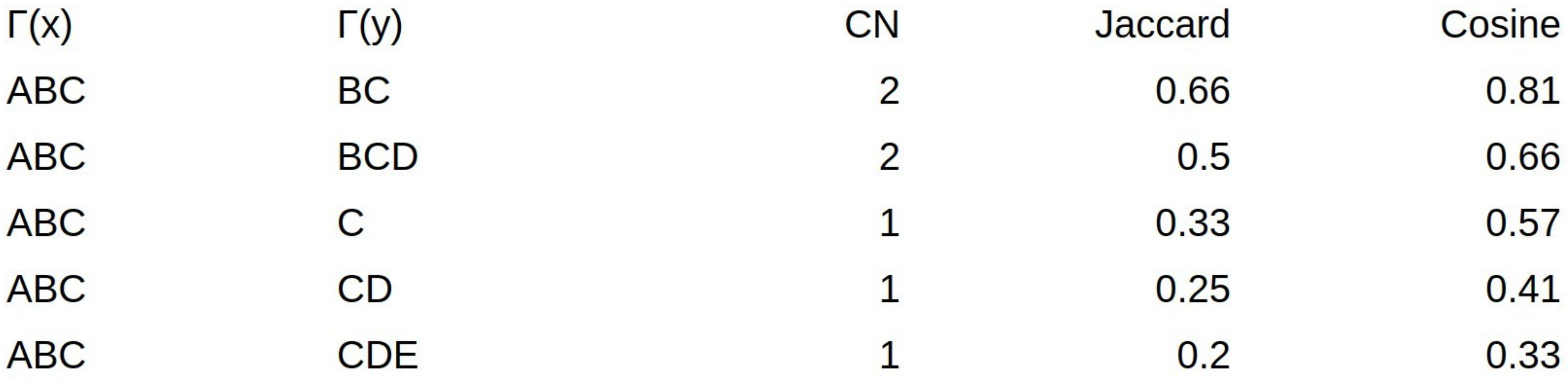
Similar neighbours

Cosine similarity

$$s_{xy}^{\text{Salton}} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k_x \times k_y}}$$

k_x ← degree of x

$\Gamma(x)$ ← neighbours of x



Γ(x)	Γ(y)	CN	Jaccard	Cosine
ABC	BC	2	0.66	0.81
ABC	BCD	2	0.5	0.66
ABC	C	1	0.33	0.57
ABC	CD	1	0.25	0.41
ABC	CDE	1	0.2	0.33

Common Neighbours ignores degrees

Jaccard and Cosine provide similar rankings

Γ(x)	Γ(y)	CN	Jaccard	Cosine
ABCDEF	DEFGH			
ABCDEF	DE			

Jaccard and Cosine do not always provide the same ranking!

Jaccard is biased towards nodes with similar degree

Γ(x)	Γ(y)	CN	Jaccard	Cosine
ABCDEF	DEFGH	3	0.38	0.55
ABCDEF	DE	2	0.33	0.58

Jaccard and Cosine do not always provide the same ranking!

Jaccard is biased towards nodes with similar degree

Other local heuristics

Adamic-Adar

$$s_{xy}^{\mathrm{AA}} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log k_z}$$

Resource Allocation

$$s_{xy}^{\mathrm{RA}} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z}$$

Other local heuristics

Adamic-Adar

$$s_{xy}^{\mathrm{AA}} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log k_z}$$

Resource Allocation

$$s_{xy}^{\mathrm{RA}} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z}$$

Preferential Attachment

$$s_{xy}^{\mathrm{PA}} = k_x \times k_y$$

Other approaches

Global heuristics : Similar to local heuristics, but considering longer path lengths

Model based : Assign probability (or “likelihood”) of edge existence

SBM