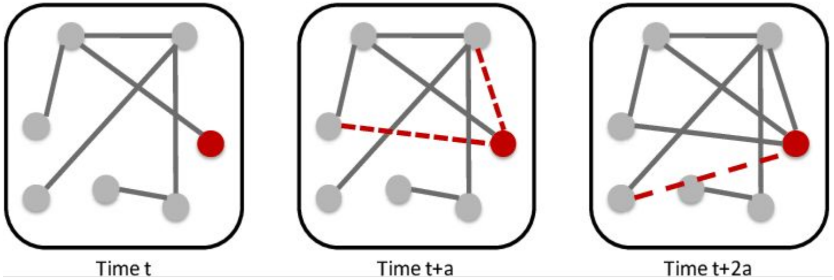
Introduction to Link Prediction

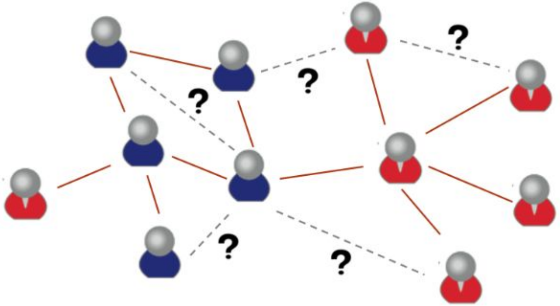
Problem definition, methods, evaluation and a case study



Link Prediction

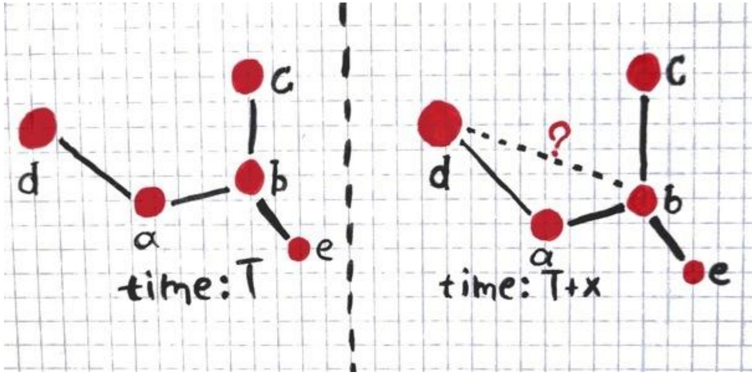
Goal

Understanding how networks evolve



Problem definition

Given a snapshot of a network at time **t**, (accurately) predict the edges that will appear in the network during the interval **(t, t+1)**



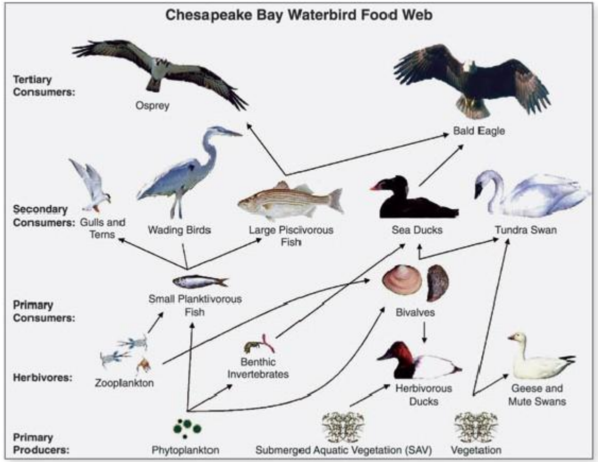
- Suggest interactions or collaborations that haven’t yet been exploited within an organization;

- Monitor terrorist networks – deducing possible interactions/missing links between terrorists (without direct evidence);

- Friendship prediction (i.e. as in Facebook)

Link prediction is used to predict future possible links in the network (e.g., Facebook).

Or, it can be used to predict missing links due to incomplete data (e.g., Food-webs)



PLOS | ONE

RESEARCH ARTICLE

Link Prediction in Criminal Networks: A Tool for Criminal Intelligence Analysis

Giulia Berlusconi[1], Francesco Calderoni[1]*, Nicola Parolini[2], Marco Verani[2], Carlo Piccardi[3]*
1 Università Cattolica del Sacro Cuore and Transcrime, Milano, Italy, 2 MOX, Department of Mathematics, Politecnico di Milano, Milano, Italy, 3 Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milano, Italy
* francesco.calderoni@unicatt.it (FC); carlo.piccardi@polimi.it (CP)

facebook

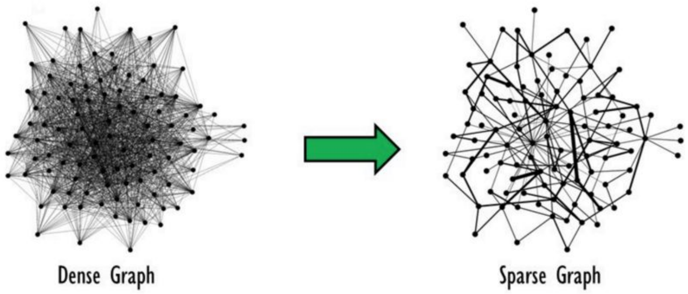
tinder

amazon

A Complex Task

Link Prediction is not an easy task because:

1. Given a graph G = (V,E) the set of possible edges to be predicted is O(|V|^2);
2. Real networks tend to be sparse



False Positive prediction issue!!

(i.e., we are likely to predict edges that will never appear)

Concretizing an Intuition…

Co-authorship network:

- Scientists who are "close" in the network (i.e. have common colleagues) $\longrightarrow$ will likely collaborate in the future

Goal:

make this intuitive notion precise and understand which measures of "proximity" leads to accurate predictions

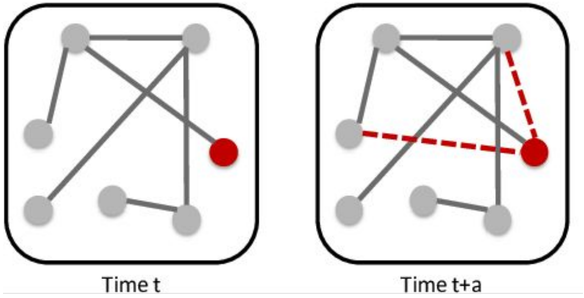
"You should spend the next week typing down names of all co-authors on your paper."

Link Prediction workflow

1. Consider as input a graph G at time t
2. Consider all the possible pairs of nodes (u,v)
3. Compute a link formation scores:

   score(u,v)

4. Build a list of all possible edges ordered by scores (from highest to lowest)
5. Verify, following that ordering, the predictions on the graph at time t+1



score is a measure of *proximity*

Link Prediction Approaches

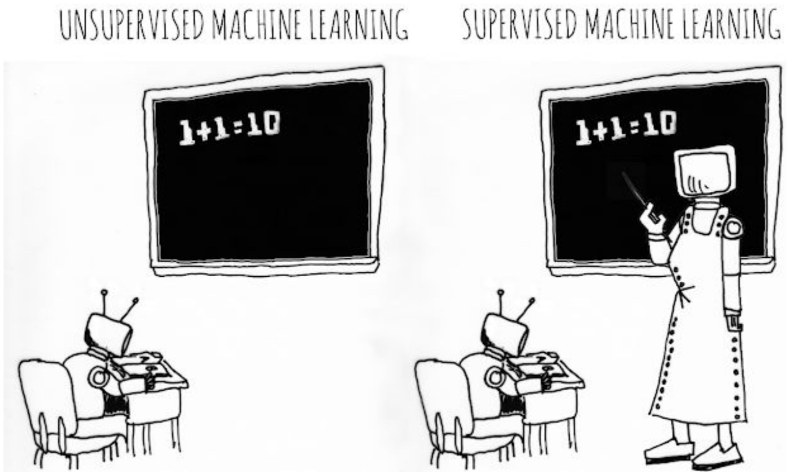
Link Prediction problem can be tackled following two different ways:

Unsupervised:

defining a set of proximity measures unrelated to the particular network

Supervised:

extracting knowledge from the network in order to improve prediction accuracy



Unsupervised Link Prediction

Unsupervised measurements rely on different structural properties of networks

- Neighborhood measures
  - Common Neighbors, Adamic Adar, Jaccard, Preferential Attachment
- Path-based measures
  - Graph distance, Katz
- Ranking
  - Sim Rank, Hitting time, Page Rank

Unsupervised Link Prediction

Neighborhood measures

How many friend we have to share in order to become friends?

Common Neighbors:

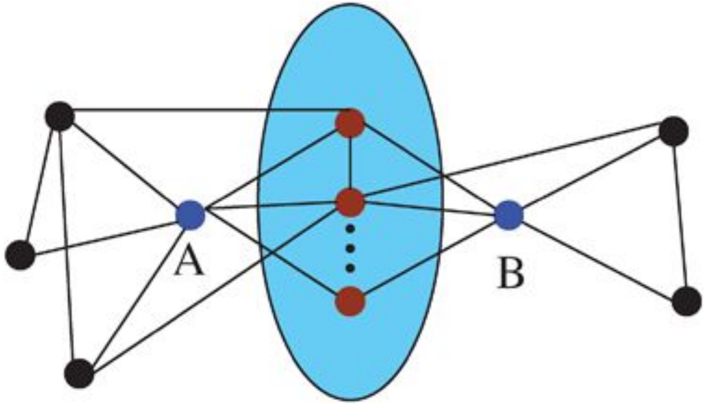
the more friends we share,
the more likely we will become friends

$$score(u, v) = |\Gamma(u) \cap \Gamma(v)|$$

Jaccard:

the more similar our friends circles are,
the more likely we will become friends

$$score(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}$$



Unsupervised Link Prediction

Neighborhood measures

How many friend we have to share in order to become friends?

Adamic Adar:

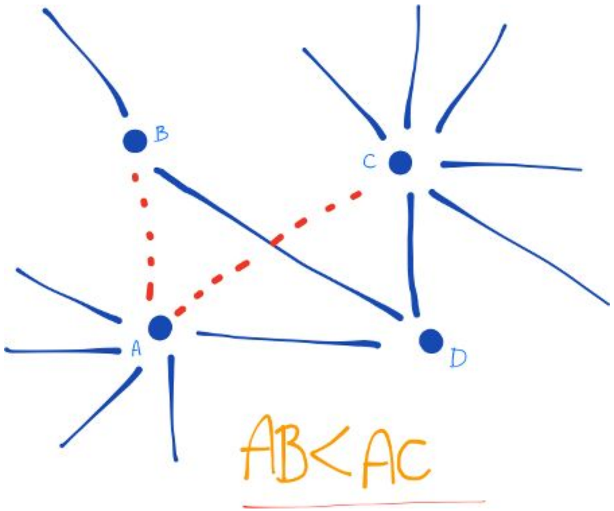
the more selective our mutual friends are,
the more likely we will become friends

$$score(u, v) = \sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log(|\Gamma(z)|)}$$

Preferential Attachment:

the more friends we have,
the more likely we will become friends

$$score(u, v) = |\Gamma(u)| * |\Gamma(v)|$$



Unsupervised Link Prediction

Path-based measures

How distant are we?

Graph Distance:

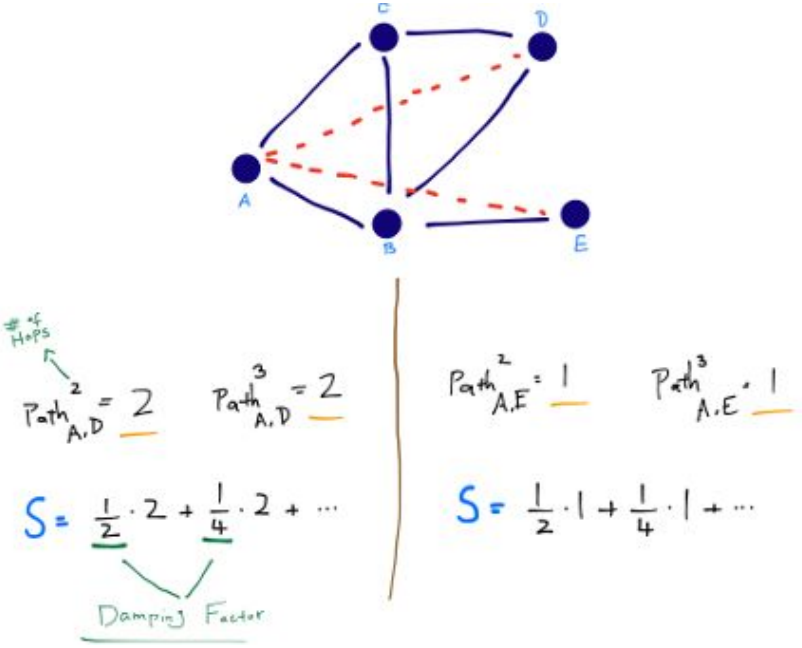
(negated) length of the shortest path between two nodes

Katz:

weighted sum over all the paths between two nodes

$$score(u, v) = \sum_{l=1}^{\infty} \beta^l \left| paths_{u,v}^{\langle l \rangle} \right|$$

where: $paths_{u,v}^{\langle l \rangle}$ ={paths of length exactly l from u to v}



Unsupervised Link Prediction

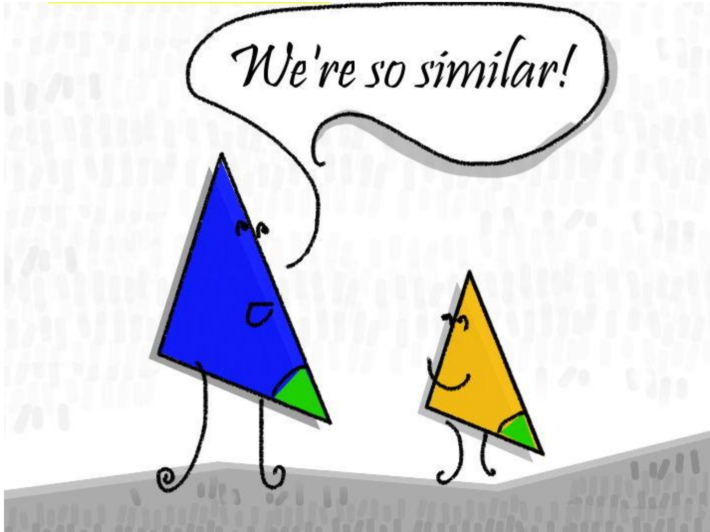
Ranking

How similar are we?

SimRank:
two nodes are *similar* to the extent that their neighborhoods are *similar*

$$similarity(u, v) = \gamma * \frac{\sum_{a \in \Gamma(u)} \sum_{n \in \Gamma(v)} similarity(a, b)}{|\Gamma(u)| * |\Gamma(v)|}$$

$$score(u, v) = similarity(u, v)$$



Unsupervised Link Prediction
Limits

Measure comparison

- No single winner
- Almost all predictors outperform the random predictor

  ⇒ there is useful information in network topology

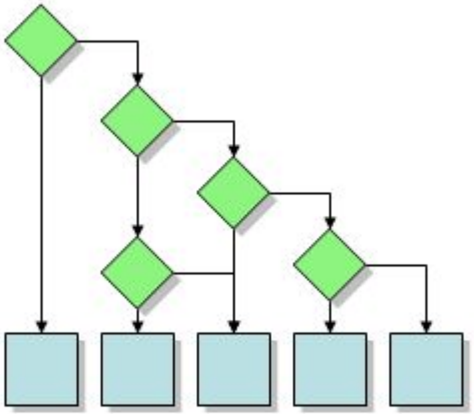
Limits

- Different kinds of networks are described by the same general closed formulae
- An average overall performance between 6% and 12%.

Supervised Link Prediction

The process is now organized in 4 steps:

1. Split the data in train/test
2. **Learning a model on the train**
3. **Use the model for prediction**
4. Compare the prediction with the test



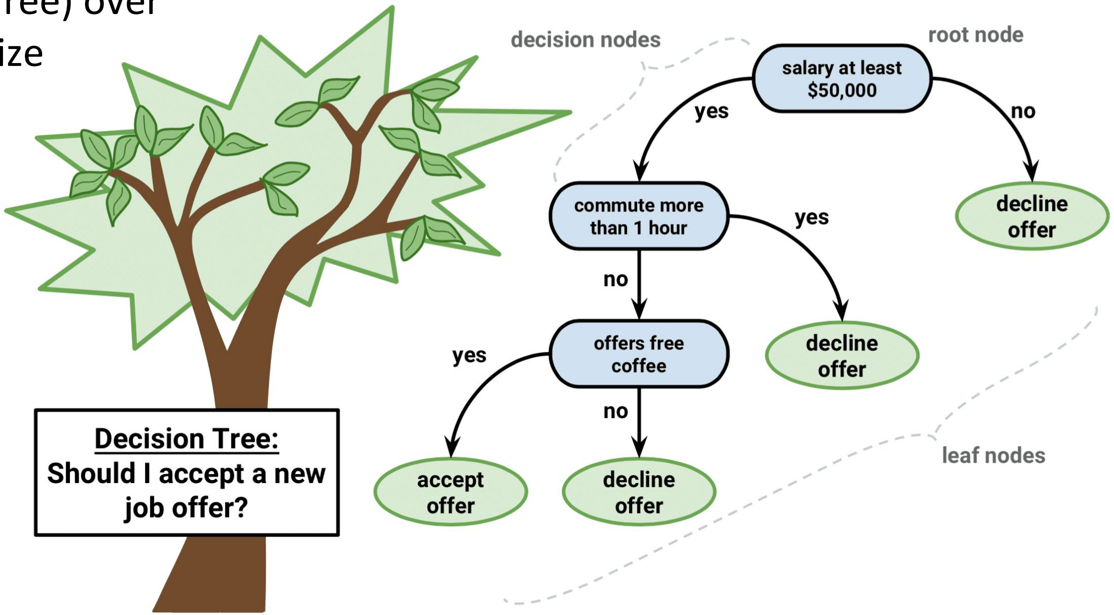
A natural way to do it:
build a "*classifier*" over a set of *network features*.

Supervised Link Prediction

Staking Unsupervised Scores

Idea

Learn a Classifier (i.e., a Decision Tree) over unsupervised LP scores to generalize the assumption they made on the network growth model

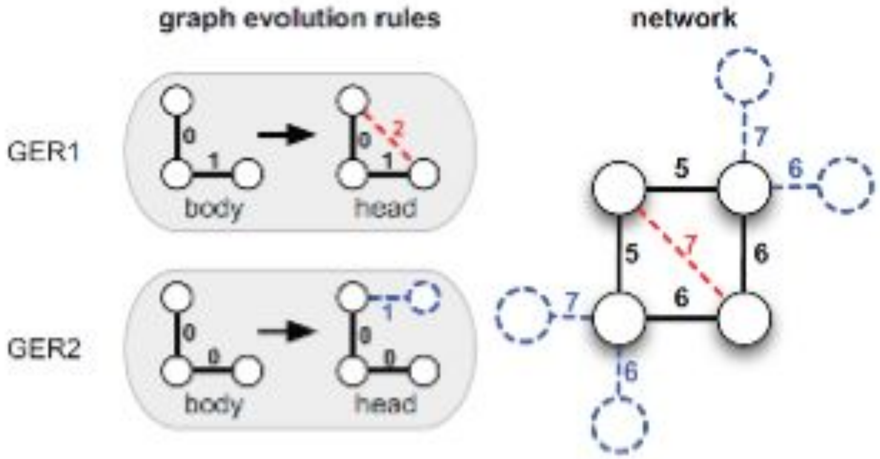


Frequent Pattern Mining

GERM

Evolution rules can be extracted from the network history and used to identify/predict recurrent patterns

- e.g., generalization of triadic closure



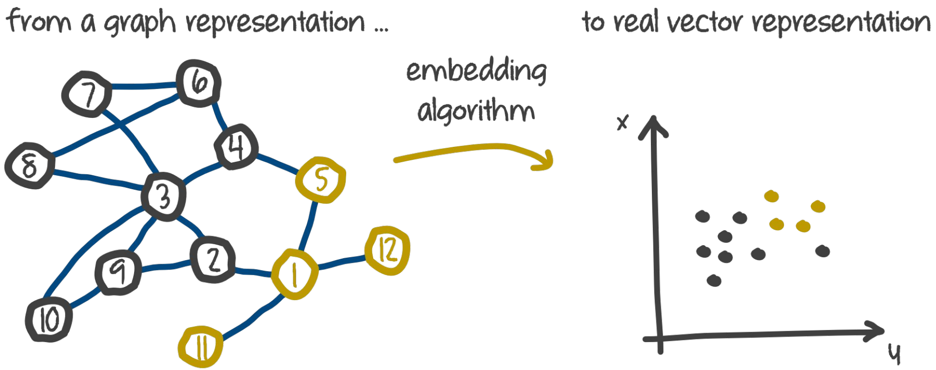
Network Embedding

Idea

Graphs can be *mapped* into vector spaces

- Node/edge similarity scores can be used to define metric spaces
- Metric spaces enable a more natural application of approaches from DM/ML

NB: Different "mappings" facilitate the solution of different classes of problems



Supervised Link Prediction

Limits

Results

- Higher performances w.r.t. unsupervised approaches

Limits

- No Free Lunch
- Model construction is often complex and, usually, more time/resource *demanding* than directly applying unsupervised scores.

Embedding is not The Answer,
only a different way to reason on graphs...

Evaluation

Given a predictor p is there a way to decide if it is a "good" one?

First Step:

verify that p outperforms the *random predictor*.

$$performance(p) = \frac{TP}{TP+FP}$$

Random Predictor

each edge has the same probability to appear in the network

$$ratio = \frac{performance(p)}{performance(p_{random})} = \frac{performance(p)}{\frac{|E_{new}|}{\frac{|V|*(|V|-1)}{2} - |E_{old}|}}$$

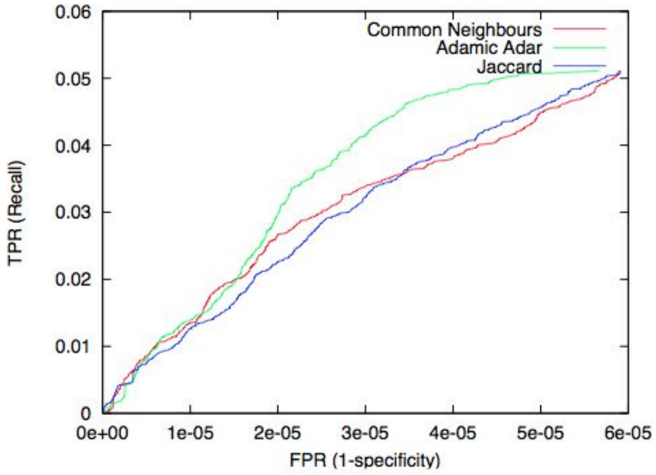
if ratio > 1 then p is meaningful

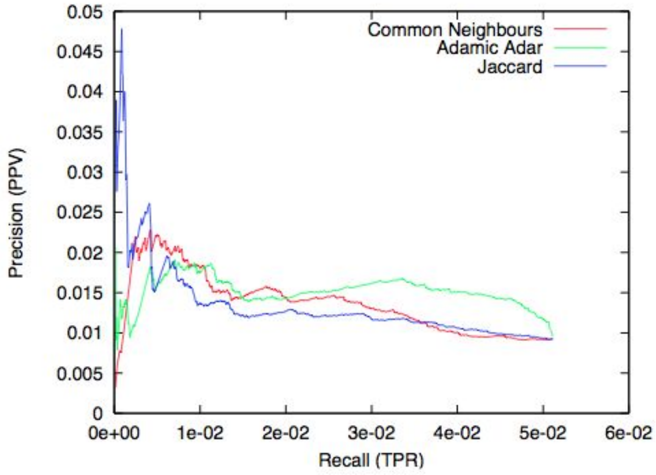
Evaluation

Comparing Predictors

Which Predictor is the best?

We need to analyze either the performances ratio, ROC and/or Precision Recall curve.





Evaluation

ROC and PR curve

Precision Vs. Recall

- Precision: PPV = TP/(TP+FP)
- Recall: TPR = TP/(TP+FN)

ROC (Receiver operating characteristic)

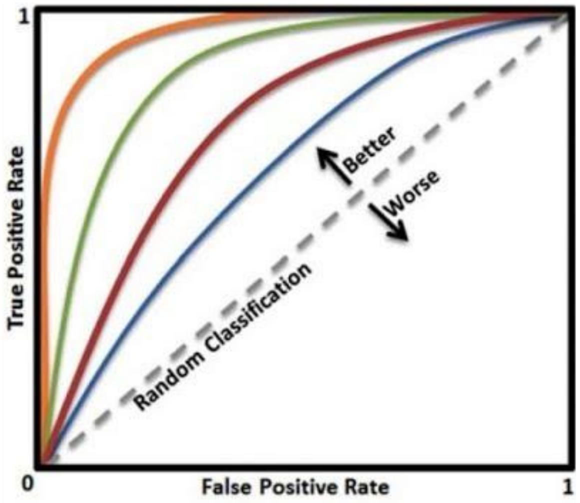
- 1-Specificity: FPR = FP/(FP+TN)
- Recall: TPR = TP/(TP+FN)

Note:

- ROC and PR spaces are isomorphic
  - the use of ROC is more widespread
- Numerical comparison can be done using the AUROC (area under the ROC curve)

	p'	n'
p	TP	FN
n	FP	TN

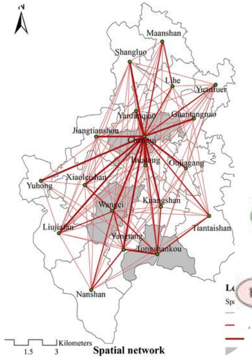
Confusion Matrix

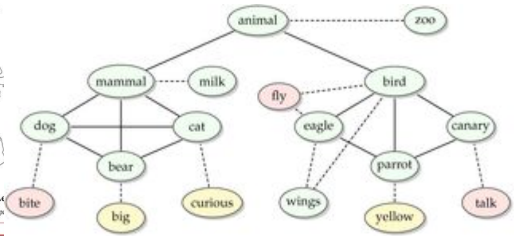


Link Prediction: something more...

Accuracy could be improved extending simple models with more complex (even semantic) informations:

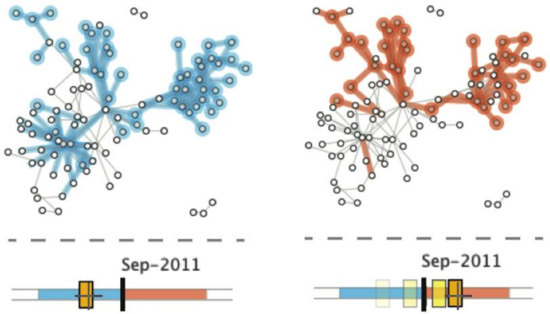
- Link strength
- Geographical information
- ...





Link Prediction needs to be revised while in some scenarios:

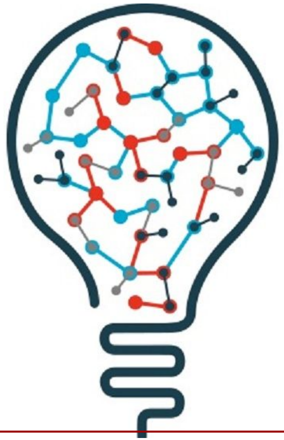
- **Dynamic Networks**
- Multiplex networks
- ...



Key Messages

Predict new link that will arise in a network is not easy:

1. Networks are, usually, sparse
2. Cold Start Problem
   - What if I don't have enough information?
     - *Can I predict bridges?*
3. Huge False Positive prediction
   - *Bridges* !?!
4. Simple approaches are "too simple"
5. Complex approaches are costly



Python Resources

- networkx
- LinkPred https://github.com/rafguns/linkpred
- LPmade https://github.com/rlichtenwalter/LPmade

Interaction Prediction in Dynamic Networks exploiting Community Discovery

Giulio Rossetti, Riccardo Guidotti, Diego Pennacchioli, Dino Pedreschi, Fosca Giannotti

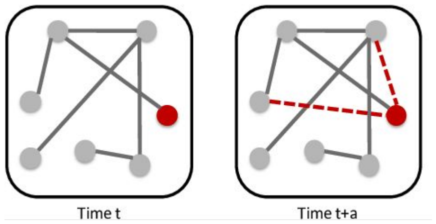
University of Pisa & ISTI-CNR

Interaction Prediction

Problem Definition

Link Prediction goal:

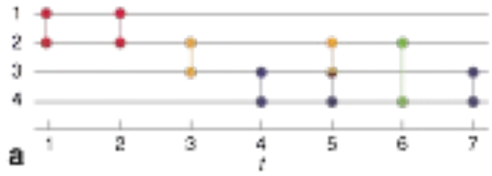
Predict ties that are not present in actual network configuration.



Ties are persistent structures that once appeared cannot disappear (i.e., friendship…)

Interaction Prediction goal:

Predict interactions that will occur (either for the first time or not) among nodes already observed in the network.

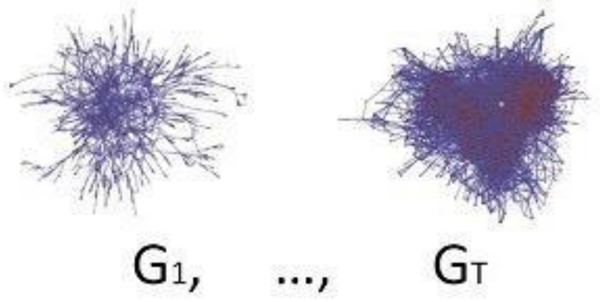


Interactions are volatile structures that can occur multiple times and whose value can vanish as time goes by
(i.e., telephone call, OSN messages…)

Interaction Prediction

A way to approach the problem

Given a set $G = \{G_0, \ldots G_t, \ldots G_T\}$ of ordered network observations, with $t \in T = \{0 \ldots T\}$, the **interaction prediction** problem aims to predict new interactions that will took place at time $T + 1$ thus composing G_{T+1}.



Idea:

- Model network evolution through temporal snapshots;
- False Positive reduction: Community Discovery as a bound for strong ties;
- Time-Aware approach: time series forecast of topological measures;
- Supervised Approach: ensemble of classifiers learnt on the topological features, tested on the forecasted ones.

Interaction Prediction

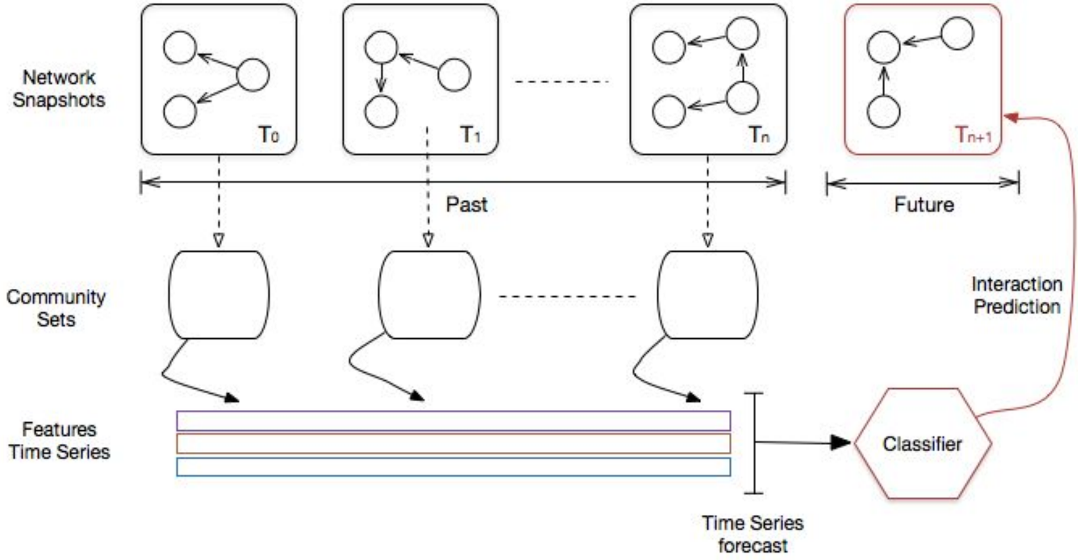
Workflow

Step 1:
For each temporal snapshot $t \in T$ compute a partition $C_t = \{C_{t,0}, \ldots, C_{t,k}\}$ of G_t using a community discovery algorithm.

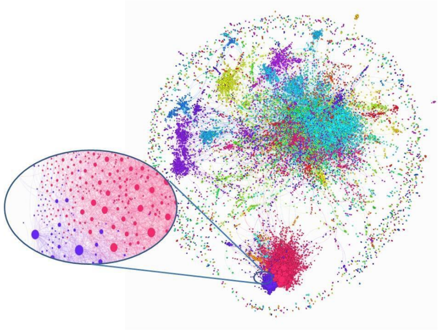
Step 2:
For each $t \in T$ compute a set of measures F for each nodes pair (u,v) belonging to at least a community in C_t

Step 3:
For each node pair (u, v) and feature $f \in F$ build a time series $S^{u,v}$ and apply a forecasting techniques in order to obtain its future expected value $f^{u,v}$

Step 4:
Use the set of expected values $f^{u,v}$ to predict future intra-community interactions.



Community Discovery



Each CD algorithm proposes its own Community Definition.

- Demon
  (ego-network based, overlap)
- Louvain
  (modularity, crisp partition)
- Infohiermap
  (conductance, crisp partition)

Features

On the identified communities we compute three set of features:

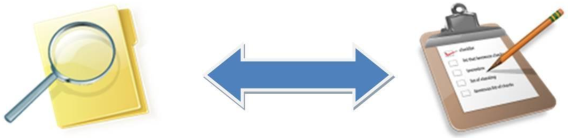
- Pairwise Structural Features:
  (i.e., Jaccard, CN, Adamic/Adar…)
- Node Topology Feature:
  (PageRank, edge betweenness…)
- Community Features:
  (i.e., density, size, shared communities, avg. clustering…)

Time series forecast

For each time series we apply several forecasting model in order to extract the expected future value

Measure	Description
Last Value (Lv)	$\Theta_t = Z_{t-1}$
Average (Av)	$\Theta_t = \frac{\sum_{i=1}^{T} z_i}{}$
Moving Average (Ma)	$\Theta_t = \frac{\sum_{i=\tau-n}^{\tau} z_i}{n}$
Linear Regression (LR)	$\Theta_{t+h} = \alpha_t + h\beta_t$

Classification

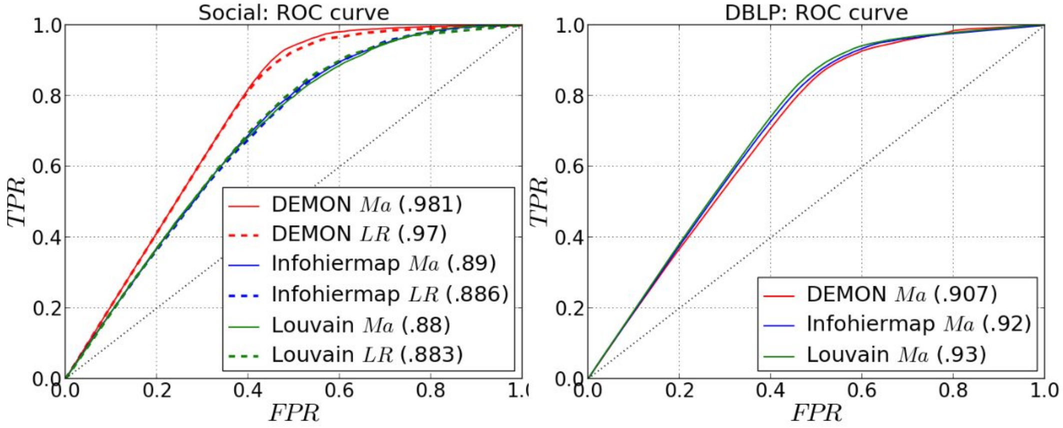


Once learned the features we design two different experiments:

- **Balanced Scenario**
  The positive and negative class are balanced through downsampling in order to design a standard baseline
- **Unbalanced Scenario**
  The data positive/negative class ratio is maintained. Due to network sparsity we observe a strong negative prevalence (~98%)

Interaction Prediction

Balanced Scenario



Very high accuracy and AUC

CD approaches contribution to IP is topology sensitive

Network	***DBLP***		***Social***	
Algorithm	**AUC**	**ACC**	**AUC**	**ACC**
DEMON Ma	0.907	85.58%	**0.981**	93.55%
DEMON LR	0.901	84.35%	0.970	91.87%
LOUVAIN Ma	**0.930**	87.72%	0.880	80.27%
LOUVAIN LR	0.926	87.48%	0.883	81.37%
INFOHIERMAP Ma	0.920	86.69%	0.890	81.34%
INFOHIERMAP LR	0.917	86.18%	0.886	80.89%

Interaction Prediction

Balanced Scenario

(cont'd - Social Case Study)

Which feature set is the most predictive?

Algorithm	*Structural*		*Topology*		*Community*	
	AUC	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**
DEMON	0.957	90.59%	0.962	91.44%	0.903	83.53%
LOUVAIN	0.850	78.63%	0.875	79.38%	0.724	66.64%
INFOHIERMAP	0.876	79.85%	0.887	80.81%	0.667	62.11%

False Positive Filtering (FSF)
vs.
No Filtering (SF)

Algorithm	Ma		LR	
	AUC	**ACC**	**AUC**	**ACC**
SF	0.901	82.88%	0.895	82.18%
FSF	**0.956**	90.10%	0.937	88.09%

All Forecast with Filtering
vs.
No Filtering

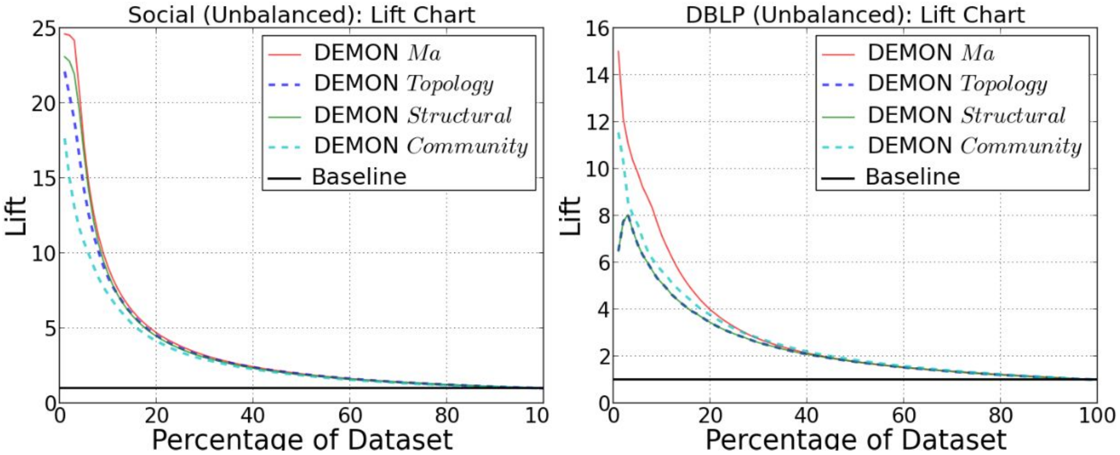
Algorithm	AUC	ACC
DEMON *All*	**0.981**	93.90%
LOUVAIN *All*	0.901	83.05%
INFOHIERMAP *All*	0.894	81.91%
FS *All*	0.959	90.44%

Interaction Prediction

Unbalanced Scenario

Negative class:

- Social 95.9%
- DBLP 98.9%



Very hard baselines

- majority classifiers scores ~.96 and ~.99 precision (always predicting “no edges”)

The proposed workflow is able to reach ~.96 and ~.45 precision w.r.t. the <u>positive class</u>

Interaction Prediction

What about weak links?

High accuracy is guaranteed by focusing the prediction on intra-community interactions.

Inter-Community Interaction Prediction

Focus on the predicting the presence/absence of *at least* a new interaction across two communities

- no identification of the “real” endpoints
- no identification of the multiplicity

Idea

1. Construct a new network where the meta-nodes are the communities
2. Apply the same workflow to such graph

Algorithm	AUC	PPV (%)
Lv	0.594	33.33
Avg	0.632	07.02
Ma	**0.647**	50.00
LR	0.596	50.00
Flat Graph	0.316	57.20
Baseline	0.504	4.01

Infohiermap performances for the inter-community prediction. Like in the balanced scenario, the Moving Average Ma fore-casted features allow for the best classification models

In bold the AUC of the best performing approach

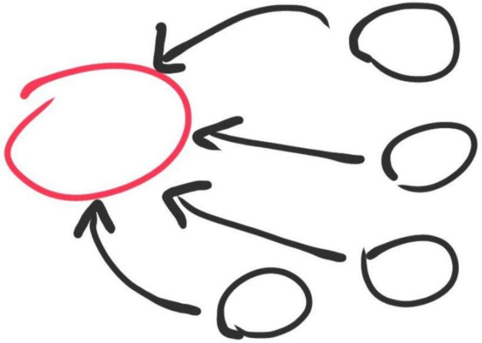
Interaction Prediction

Conclusions

Even though Interaction prediction is a complex problem it is possible to reach high accuracy through:

- Target selection:
  False Positive reduction via Community Discovery
  Weak interactions treated as “*special cases*”

- Local topology history analysis:
  Feature forecast via Time Series analysis



Moreover, each type of datasets demands a specific CD algorithm:

- One-to-one interactions (i.e. social ones)

- Many-to-many interactions (i.e. co-authorship relations)