RankMerging: a supervised learning method to predict links in social networks.

Lionel Tabourier

DyNak II

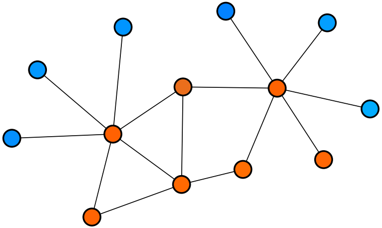
September 2014, Monday 15th

Namur Center for Complex Systems

Context and problem

The phone service provider problem

Weighted network of users : $w(i, j)$ = number of calls i–j



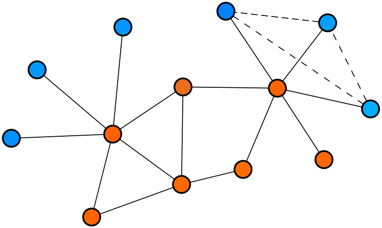
Unknown blue-blue links, yet important for commercial strategy...

How to guess these links ?

Context and problem

The phone service provider problem

Weighted network of users : $w(i, j)$ = number of calls i–j



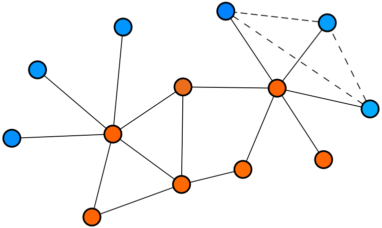
Unknown blue-blue links, yet important for commercial strategy...

How to guess these links ?

Context and problem

The phone service provider problem

Weighted network of users : $w(i,j)$ = number of calls i–j



Unknown blue-blue links, yet important for commercial strategy...

How to guess these links ?

General problem : discovering missing links

Crawled graph : (V, E) , Real graph : (V, E')

Discover links in $E' \setminus E$

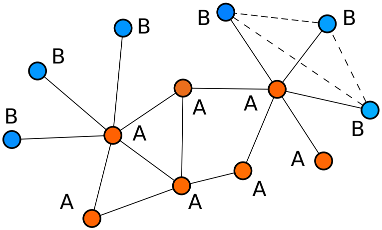
Data for supervised prediction

Dataset - Whole set (test set)

$\sim$ 1,130,000 nodes : 75% A , 25% B

$\sim$ 750,000 links

$\sim$ 50,000 B—B links to guess



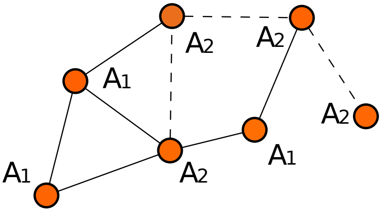
Data for supervised prediction

Dataset - Learning set

$\sim$ 850,000 nodes : $\frac{2}{3}$ A_1 , $\frac{1}{3}$ A_2

$\sim$ 600,000 links

$\sim$ 50,000 A_2—A_2 links to guess



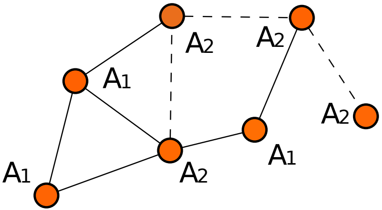
Data for supervised prediction

Dataset - Learning set

$\sim$ 850,000 nodes : $\frac{2}{3}$ A_1 , $\frac{1}{3}$ A_2

$\sim$ 600,000 links

$\sim$ 50,000 A_2—A_2 links to guess



Tunable number of predictions

depending on commercial strategy

General prediction method

An unbalanced classification problem

For any (unlinked) pair of nodes, is there a missed link or not ?

two classes : yes / no

unbalanced classes : much more pairs of nodes than missed links

General prediction method

An unbalanced classification problem

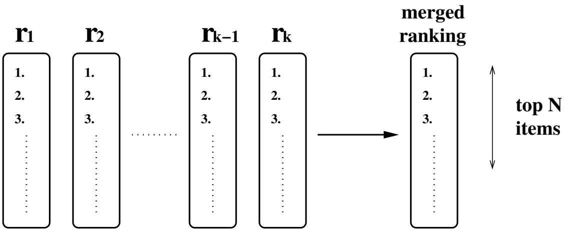
For any (unlinked) pair of nodes, is there a missed link or not ?

two classes : yes / no

unbalanced classes : much more pairs of nodes than missed links

Classification by ranking

Items (pairs) ranked according to various methods :



comes to a learning-to-rank problem

Basic unsupervised ranking methods

Structural scoring

example : common neighbors in weighted networks

$$s_{CN_w}(i,j) = \sum_{k \in \mathcal{N}(i) \cap \mathcal{N}(j)} w(i,k).w(j,k)$$

Other examples

- **local structure** : Jaccard index, Adamic-Adar index, ...
- **global structure** : Katz index, Random Walk, Hitting Time, Preferential Attachment ...

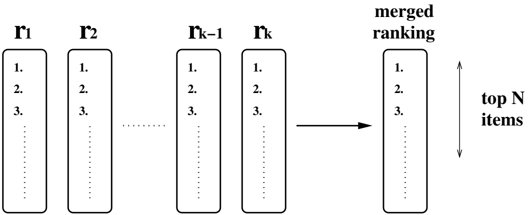
Many other : node-based, relation-based...

we only consider **structural methods**

see Al Hasan et al. (SDM06 - LACS ws)

Combining unsupervised methods

General principle



Consensus methods

see Dwork et al. (WWW'01)

- **Averaging rankings** : Borda's method
- Other methods : median (MedRank), Markov chain mixing...

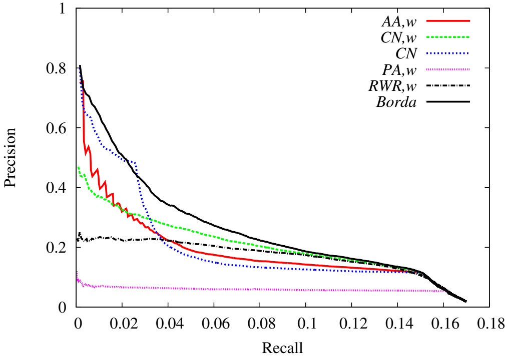
Simple to implement, (quasi-)linear complexity

Unsupervised methods : results (learning set)

Visualization metrics

Variable number of predictions, depends on strategy

Trade-off between Precision and Recall



Supervised framework

Using classic methods

see Pujari et al. (WWW'12 - MSDN ws)

Supervised methods for classification problems :
Classification Trees, Nearest Neighbor, SVM, AdaBoost...

But fixed number of predictions

Supervised framework

Using classic methods

see Pujari et al. (WWW'12 - MSDN ws)

Supervised methods for classification problems :
Classification Trees, Nearest Neighbor, SVM, AdaBoost...

But fixed number of predictions

Using pairwise transform

see Pedregosa et al. (MLMI 2012)

item X with ranks $\{x_1, x_2, ..., x_k\}$
item Y with ranks $\{y_1, y_2, ..., y_k\}$
From $\{(y_1 - x_1), (y_2 - x_2), ..., (y_k - x_k)\}$, learn if is X over Y

Back to a classification problem

But with ranking size S : $O(S^2) \Rightarrow$ too many pairs

Aggregation with *RankMerging*

Learning process

- 1. **Define window** $\mathcal{W}_i$ of size s : next top-s pairs of ranking i.
- 2. **Measure quality** : number of true links in the window.
- 3. **Select highest quality ranking** : its top pair is selected.
- 4. **Register** selected rankings : index ϕ_i.
- 5. **Update windows**.
- 6. **Iterate** from 2.

Testing process

Use learned ϕ_i during learning to **aggregate rankings** on test set.

Warning : a pair can only be predicted once.
$+$ **scaling factor** if learning set size $\neq$ testing set size

Learning example

Two rankings r_A and r_B

Grey background : **window** (size 5)
Green background : **selected**
Red background : **forbidden**

r_A	tp	r_B	tp
(1,2)		(5,18)	x
(1,4)	x	(1,2)	
(5,6)	x	(8,9)	
(6,12)	x	(5,6)	x
(5,18)	x	(7,11)	x
(3,4)		(6,9)	x
(4,9)	x	(1,14)	
(7,11)	x	(2,9)	
(2,9)		(3,7)	

$$\phi_A = 0 \ , \ \phi_B = 0$$

Learning example

Two rankings r_A and r_B

Grey background : **window** (size 5)
Green background : **selected**
Red background : **forbidden**

r_A	tp	r_B	tp
(1,2)		(5,18)	x
(1,4)	x	~~(1,2)~~	
(5,6)	x	(8,9)	
(6,12)	x	(5,6)	x
(5,18)	x	(7,11)	x
(3,4)		(6,9)	x
(4,9)	x	(1,14)	
(7,11)	x	(2,9)	
(2,9)		(3,7)	

$$\phi_A = 1 \; , \; \phi_B = 0$$

Learning example

Two rankings r_A and r_B

Grey background : **window** (size 5)
Green background : **selected**
Red background : **forbidden**

r_A	tp	r_B	tp
(1,2)		(5,18)	x
(1,4)	x	~~(1,2)~~	
(5,6)	x	(8,9)	
(6,12)	x	(5,6)	x
~~(5,18)~~	x	(7,11)	x
(3,4)		(6,9)	x
(4,9)	x	(1,14)	
(7,11)	x	(2,9)	
(2,9)		(3,7)	

$$\phi_A = 1 \ , \ \phi_B = 1$$

Learning example

Two rankings r_A and r_B

Grey background : **window** (size 5)
Green background : **selected**
Red background : **forbidden**

r_A	tp	r_B	tp
(1,2)		(5,18)	x
(1,4)	x	~~(1,2)~~	
(5,6)	x	(8,9)	
(6,12)	x	(5,6)	x
~~(5,18)~~	x	(7,11)	x
(3,4)		(6,9)	x
(4,9)	x	(1,14)	
(7,11)	x	(2,9)	
(2,9)		(3,7)	

$$\phi_A = 2 \ , \ \phi_B = 1$$

Learning example

Two rankings r_A and r_B

Grey background : **window** (size 5)
Green background : **selected**
Red background : **forbidden**

r_A	tp	r_B	tp
(1,2)		(5,18)	x
(1,4)	x	~~(1,2)~~	
(5,6)	x	(8,9)	
(6,12)	x	~~(5,6)~~	x
~~(5,18)~~	x	(7,11)	x
(3,4)		(6,9)	x
(4,9)	x	(1,14)	
(7,11)	x	(2,9)	
(2,9)		(3,7)	

$$\phi_A = 3 \ , \ \phi_B = 1$$

Test example

Link prediction on test set

learning step s	ϕ_A	ϕ_B
1	1	0
2	1	1
3	2	1
4	3	1

r_A	r_B
(2,8)	(1,8)
(3,9)	(1,7)
(1,8)	(5,11)
(5,11)	(8,10)
(4,7)	(4,5)
(3,6)	(3,9)
(2,3)	(3,6)

$\longrightarrow$

r_A	r_B
(2,8)	(1,8)
(3,9)	(1,7)
~~(1,8)~~	(5,11)
(5,11)	(8,10)
(4,7)	(4,5)
(3,6)	(3,9)
(2,3)	(3,6)

$\longrightarrow$

r_A	r_B
(2,8)	(1,8)
(3,9)	(1,7)
~~(1,8)~~	(5,11)
(5,11)	(8,10)
(4,7)	(4,5)
(3,6)	~~(3,9)~~
(2,3)	(3,6)

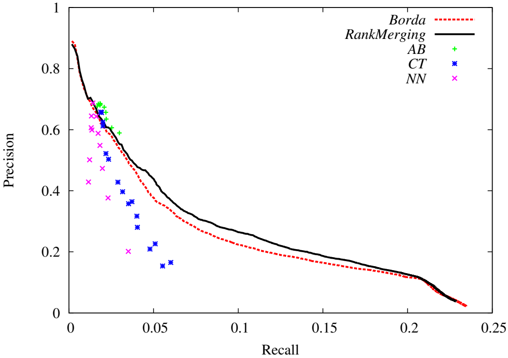
$\longrightarrow$

r_A	r_B
(2,8)	(1,8)
(3,9)	(1,7)
~~(1,8)~~	~~(5,11)~~
(5,11)	(8,10)
(4,7)	(4,5)
(3,6)	~~(3,9)~~
(2,3)	(3,6)

Results (test set)

Aggregated rankings

Adamic-Adar, Common Neighbors, Jaccard, Katz, Preferential Attachment, Random Walk with Restart and *Borda* (baseline)



+8.2% area increase to baseline

Pros and Cons

Advantages

- **Addition of any ranking increases performance**
- **Scalable :** $O(N.\alpha)$, α : number of rankings , N predictions
- **Simplicity** (see `lioneltabourier.fr/program.html`)

Add as many ranking methods as possible to improve.

Drawbacks

- Windows size imply an **averaging effect** on prediction quality

Not suited for high precision on few items.

Conclusion

PSP : Additional information sources

duration of interactions, text messages

localization ? individual attributes (age, gender) ? usages (apps) ?

General : Relevant when...

Many complementary sources of information

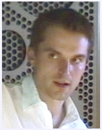
Each source yields low precision

Fields of application ?

- incomplete data sampling
- network growth microdynamic
- biomedical engineering

Other suggestions ?

Thanks for your attention !



lionel.tabourier@lip6.fr

anne-sophie.libert@unamur.be

renaud.lambiotte@unamur.be