Edge Sign Prediction in Social Networks

Géraud Le Falher — Inria Lille Nord Europe, Magnet Team
Joint work with Nicolò Cesa-Bianchi, Claudio Gentile and Fabio Vitale

July 17, 2017

Directed signed social networks

- Directed signed social networks have the usual positive relations, driven by the *homophily* assumption,
- but also negative relations, e.g. distrust, enemyship.
- This gives rise to **new problems**.
- For instance, observing some signs, can we predict the remaining ones?

Motivations

Being able to predict edge signs let us solve **practical, real world problems**:

- "Frenemy" detection [1];
- Automatic moderation of large scale online interactions;
- Cyber bullying prevention, at school or in online games [2].

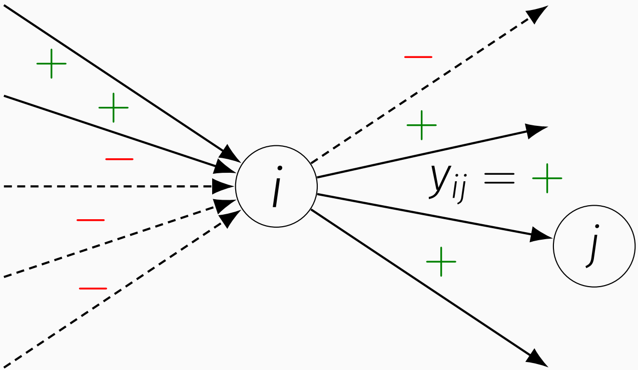
Contributions

1. A generative model based on trollness and trustworthiness, justifying existing heuristics and providing a new principled predictor
2. A maximum likelihood approximation by a label propagation algorithm, leveraging a reduction from edge to node classification
3. A natural complexity measure leading to an efficient online algorithm

Outline

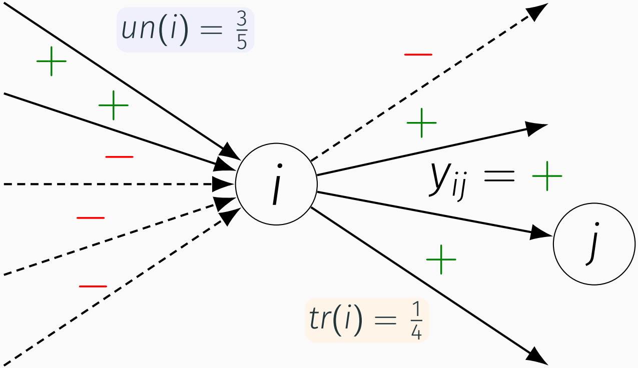
Notations

$G = (V, E)$ is a directed graph with **no side information** but full topology.



Notations

$G = (V, E)$ is a directed graph with **no side information** but full topology.



The trollness of i $tr(i)$ is its fraction of negative outgoing links, its untrustworthiness $un(i)$ is its fraction of negative incoming links.

Generative model

$\mu(p, q)$ is an arbitrary prior distribution over $[0, 1] \times [0, 1]$

$(p_i, q_i) \sim \mu(p, q)$ $(p_j, q_j) \sim \mu(p, q)$

i j

Generative model

$\mu(p, q)$ is an arbitrary prior distribution over $[0, 1] \times [0, 1]$

$(p_i, q_i) \sim \mu(p, q)$ $\qquad\qquad$ $(p_j, q_j) \sim \mu(p, q)$

$$i \xrightarrow[\mathbb{P}(y_{ij} = +1) = \frac{1}{2}\left(p_i + q_j\right)]{} j$$

Generative model

$\mu(p, q)$ is an arbitrary prior distribution over $[0, 1] \times [0, 1]$

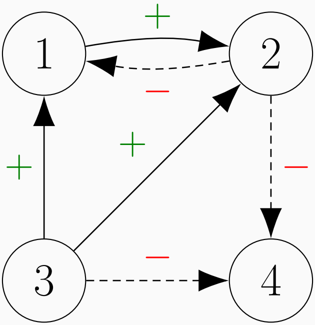
$(p_i, q_i) \sim \mu(p, q)$ $\qquad\qquad$ $(p_j, q_j) \sim \mu(p, q)$

$$i \xrightarrow[\mathbb{P}(y_{ij} = +1) = \frac{1}{2}\left(p_i + q_j\right)]{} j$$

The Bayes optimal prediction for $y_{i,j}$ is thus

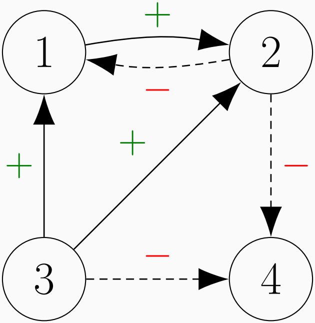
$$y^*(i, j) = \textsc{sgn}\left(\mathbb{P}\left(y_{i,j} = +1\right) - \frac{1}{2}\right)$$

Edge-to-node reduction: construction

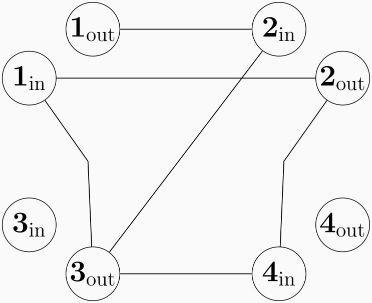


(a) G

Edge-to-node reduction: construction

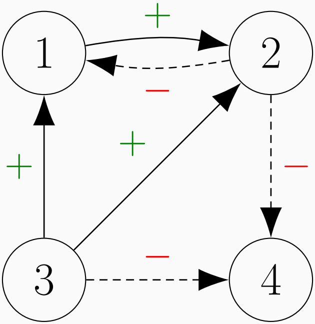


(a) G

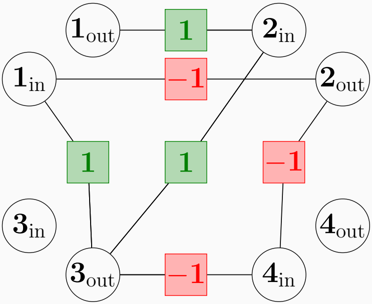


(b) G'

Edge-to-node reduction: construction

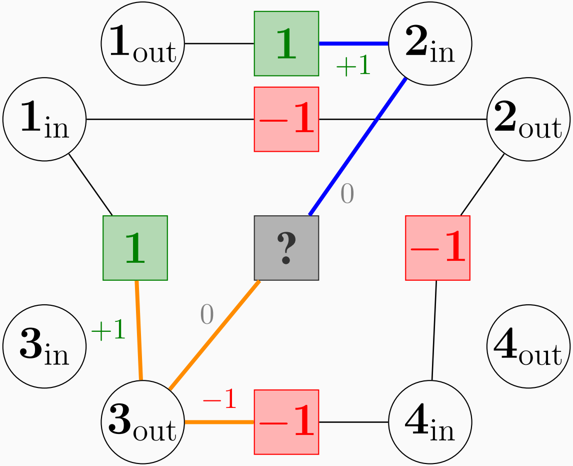


(a) G

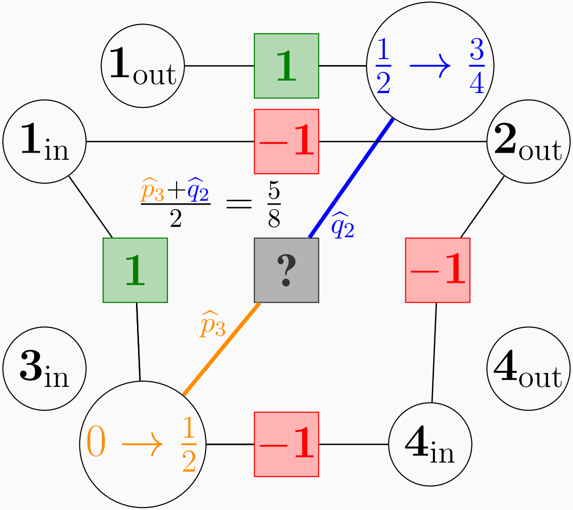


(b) G'

Edge-to-node reduction: propagation



Edge-to-node reduction: propagation



Outline

Batch settings

Given a graph G labeled by our generative model, we observe a training set E_0.

We present two methods to predict the labels of $E \setminus E_0$:

- An approximation of the Bayes optimal predictor in an active setting
- An approximation of Maximal Likelihood parameters in a passive setting

Approximation to Bayes via Active Learning

The complementary to 1 of trollness and untrustworthiness (estimated on E_0) are used as proxy for p_i and q_j so that

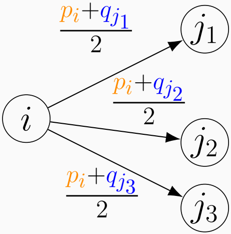
$$\widehat{y}(i,j) = \mathrm{SGN}\Big(\underbrace{\big(1-\widehat{tr}(i)\big) + \big(1-\widehat{un}(j)\big) - \tau}_{\approx \frac{1}{2}\left(p_i + q_j\right) = \mathbb{P}(y_{ij}=+1)} - \frac{1}{2}\Big)$$

Approximation to Bayes via Active Learning

The complementary to 1 of trollness and untrustworthiness (estimated on E_0) are used as proxy for p_i and q_j so that

$$\widehat{y}(i,j) = \text{SGN}\Big(\underbrace{\big(1-\widehat{tr}(i)\big) + \big(1-\widehat{un}(j)\big) - \tau}_{\approx\frac{1}{2}(p_i+q_j)=\mathbb{P}(y_{ij}=+1)} - \tfrac{1}{2}\Big)$$

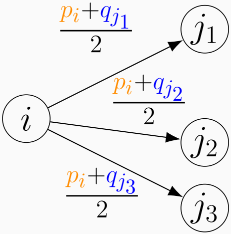
$1-\widehat{tr}(i)$ is an indirect observation of p_i



Approximation to Bayes via Active Learning

The complementary to 1 of trollness and untrustworthiness (estimated on E_0) are used as proxy for p_i and q_j so that

$$\widehat{y}(i,j) = \mathrm{SGN}\Big(\underbrace{\big(1-\widehat{tr}(i)\big) + \big(1-\widehat{un}(j)\big) - \tau}_{\approx\frac{1}{2}(p_i+q_j)=\mathbb{P}(y_{ij}=+1)} - \tfrac{1}{2}\Big)$$



$1-\widehat{tr}(i)$ is an indirect observation of p_i

Letting

$$\overline{q}_i = \frac{1}{d_{\mathrm{out}}(i)} \sum_{j \text{ s.t.} (i,j)\in E} q_j$$

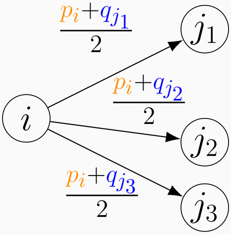
we have

$$1-\widehat{tr}(i) \approx \frac{1}{2}\left(p_i + \overline{q}_i\right) \quad \text{and} \quad 1-\widehat{un}(j) \approx \frac{1}{2}\left(q_j + \overline{p}_j\right)$$

Approximation to Bayes via Active Learning

The complementary to 1 of trollness and untrustworthiness (estimated on E_0) are used as proxy for p_i and q_j so that

$$\widehat{y}(i,j) = \text{SGN}\Big(\underbrace{\big(1-\widehat{tr}(i)\big) + \big(1-\widehat{un}(j)\big) - \tau}_{\approx\frac{1}{2}(p_i+q_j)=\mathbb{P}(y_{ij}=+1)} - \tfrac{1}{2}\Big)$$



$1-\widehat{tr}(i)$ is an indirect observation of p_i

Letting

$$\overline{q}_i = \frac{1}{d_{\text{out}}(i)} \sum_{j \text{ s.t.} (i,j)\in E} q_j$$

we have

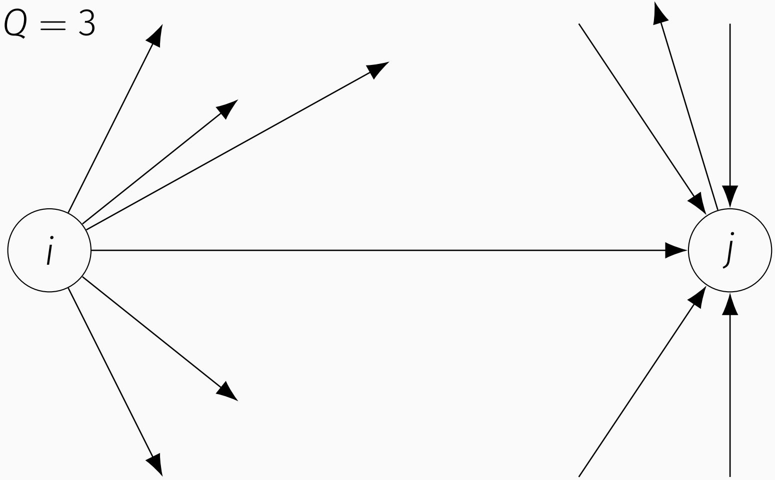
$$1-\widehat{tr}(i) \approx \frac{1}{2}\left(p_i + \overline{q}_i\right) \quad \text{and} \quad 1-\widehat{un}(j) \approx \frac{1}{2}\left(q_j + \overline{p}_j\right)$$

Thus we need to subtract

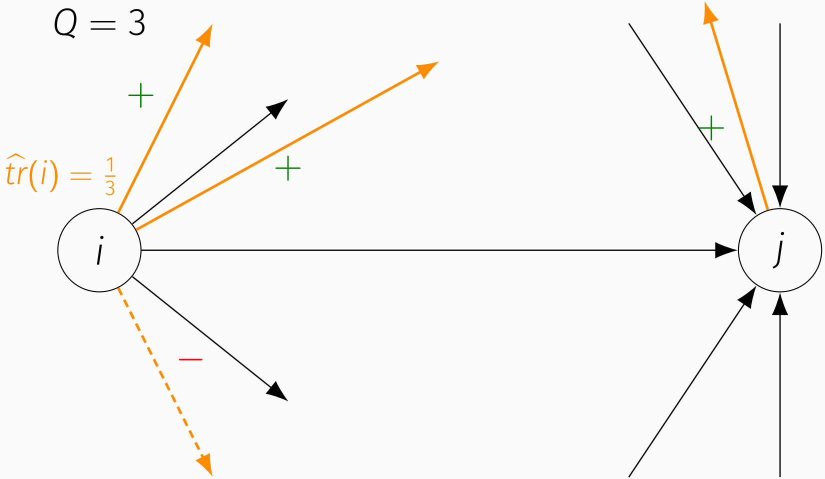
$$\tau = \frac{1}{2}\left(\mu_p + \mu_q\right)$$

as $\overline{p}_j$ and $\overline{q}_i$ concentrates around their mean μ_p and μ_q.

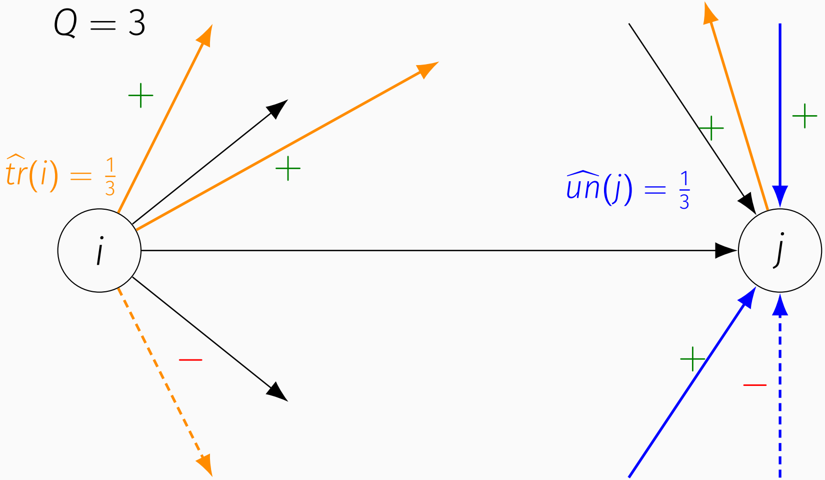
Active algorithm



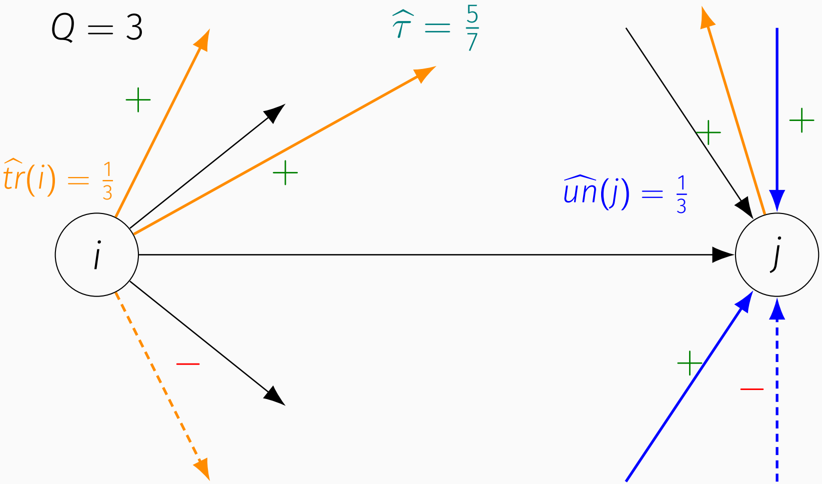
Active algorithm



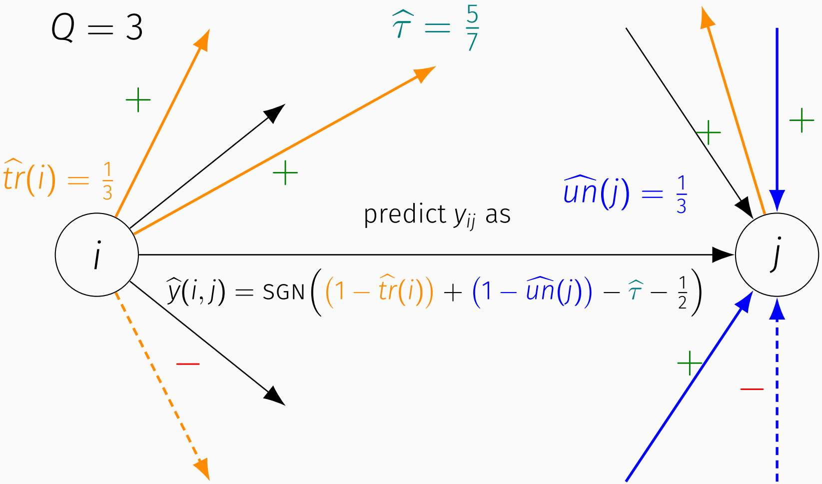
Active algorithm



Active algorithm



ACTIVE ALGORITHM



How much sampling is needed?

Setting

$$Q = \frac{1}{2\varepsilon^2} \ln \frac{4|V|}{\delta}$$

we query $\Theta\big(|V| \ln |V|\big)$ edges.

How much sampling is needed?

Setting

$$Q = \frac{1}{2\varepsilon^2} \ln \frac{4|V|}{\delta}$$

we query $\Theta\big(|V| \ln |V|\big)$ edges.

This is enough to guarantee that

$$\left| \left[\big(1 - \widehat{tr}(i)\big) + \big(1 - \widehat{un}(j)\big) - \widehat{\tau} \right] - \left[\frac{p_i + q_j}{2} \right] \right| \leq 8\epsilon$$

holds with probability at least $1 - 10\delta$ simultaneously for all non-queried edges $(i, j) \in E$ such that $d_{\mathrm{out}}(i), d_{\mathrm{in}}(j) \geq Q$.

Correct prediction as long as $\mathbb{P}(y_{i,j} = +1)$ is bounded away from $\frac{1}{2}$.

Maximum Likelihood in Passive Setting

- The training set E_0 has a likelihood of $\mathbb{P}\left(E_0 \mid \{p_i, q_i\}_{i=1}^{|V|}\right)$.

Maximum Likelihood in Passive Setting

- The training set E_0 has a likelihood of $\mathbb{P}\left(E_0 \mid \{p_i, q_i\}_{i=1}^{|V|}\right)$.
- We would like to approximate $y^*(i,j)$ by resorting to a maximum likelihood estimator of the parameters $\{p_i, q_i\}_{i=1}^{|V|}$ based on E_0.

Maximum Likelihood in Passive Setting

- The training set E_0 has a likelihood of $\mathbb{P}\left(E_0 \mid \{p_i, q_i\}_{i=1}^{|V|}\right)$.
- We would like to approximate $y^*(i, j)$ by resorting to a maximum likelihood estimator of the parameters $\{p_i, q_i\}_{i=1}^{|V|}$ based on E_0.
- The gradient of the log-likelihood function w.r.t. p_ℓ is

$$\sum_{\ell,j \in E_0; y_{\ell j}=+1} \frac{1}{p_\ell + q_j} - \sum_{\ell,j \in E_0; y_{\ell j}=-1} \frac{1}{2 - p_\ell - q_j}$$

Maximum Likelihood Approximation

- Those equations are non linear, thus we approximate them further.

Maximum Likelihood Approximation

- Those equations are **non linear**, thus we approximate them further.
- This approximation is equivalent to setting to zero the gradient w.r.t. $(\boldsymbol{p}, \boldsymbol{q}) = \{p_i, q_i\}_{i=1}^{|V|}$ of the quadratic function

$$f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{(i,j)\in E_0} \Bigg(\underbrace{\frac{p_i + q_j}{2}}_{\in[0,1]} - \underbrace{\frac{1 + y_{i,j}}{2}}_{\in[0,1]} \Bigg)^2$$

Maximum Likelihood Approximation

- Those equations are **non linear**, thus we approximate them further.
- This approximation is equivalent to setting to zero the gradient w.r.t. $(\boldsymbol{p}, \boldsymbol{q}) = \{p_i, q_i\}_{i=1}^{|V|}$ of the quadratic function

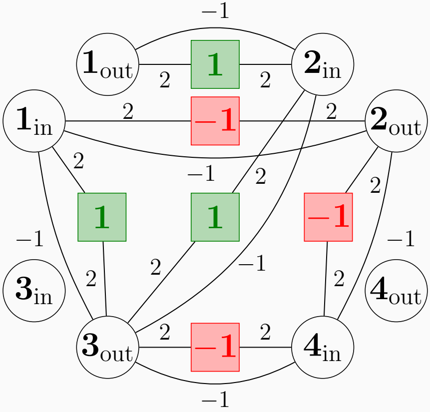
$$f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{(i,j)\in E_0} \Big(\underbrace{\frac{p_i + q_j}{2}}_{\in[0,1]} - \underbrace{\frac{1 + y_{i,j}}{2}}_{\in[0,1]} \Big)^2$$

- We follow a label propagation approach by **making the test labels appear** and minimizing $f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) + f_{E\setminus E_0}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j})$ w.r.t. both $(\boldsymbol{p}, \boldsymbol{q})$ and all $y_{i,j} \in [-1, +1]$, for $(i,j) \in E \setminus E_0$.

Label propagation: regularized objective

We use a weighted version of G' with negative edges, which introduces an extra regularization term.

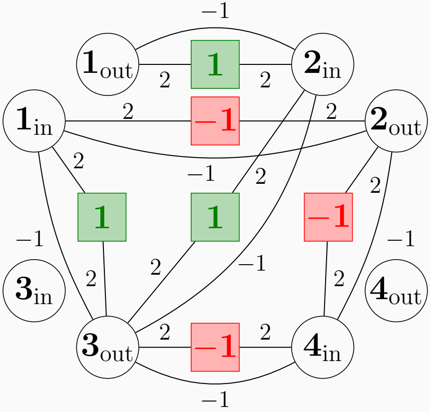
$$\underbrace{\widehat{f}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j\,(i,j)\in E\setminus E_0})}_{\text{energy function on } G''} = f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) + f_{E\setminus E_0}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j}) + \widetilde{\text{regul}}$$



Label propagation: regularized objective

We use a weighted version of G' with negative edges, which introduces an extra regularization term.

$$\underbrace{\widehat{f}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j\,(i,j)\in E\setminus E_0})}_{\text{energy function on } G''} = f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) + f_{E\setminus E_0}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j}) + \widetilde{\text{regul}}$$



We run diameter(G) iterations of label propagation and use a binary threshold over the estimated $y_{i,j}$ to predict signs.

Outline

Setting

The signs are **adversarial** rather than generated by our model.

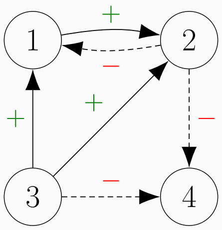
At each round, the learner is asked to predict one label, which is then revealed to him and the procedure repeats.

Labeling regularity

Letting Y be the vector of all labels, $\Psi_{\mathrm{out}}(i, Y)$ is the number of least used label outgoing from i, and $\Psi_{\mathrm{out}}(Y) = \sum_{i \in V} \Psi_{\mathrm{out}}(i, Y)$.

Likewise for incoming edges, $\Psi_{\mathrm{in}}(Y) = \sum_{j \in V} \Psi_{\mathrm{in}}(j, Y)$ and finally $\Psi_G(Y) = \min \left\{ \Psi_{\mathrm{in}}(Y), \Psi_{\mathrm{out}}(Y) \right\}$.

node i	1	2	3	4	
$\Psi_{\mathrm{out}}(i, Y)$	0	0	1	0	$\Psi_{\mathrm{out}}(Y) = 1$
$\Psi_{\mathrm{in}}(i, Y)$	1	0	0	1	$\Psi_{\mathrm{in}}(Y) = 2$



Online algorithm and bounds

- Our algorithm is a combination of Randomized Weighted Majority instances built on top of each other.

Online algorithm and bounds

- Our algorithm is a combination of Randomized Weighted Majority instances built on top of each other.
- On average, it makes $\Psi_G(Y) + O\left(\sqrt{|V|\Psi_G(Y)} + |V|\right)$ mistakes.

Online algorithm and bounds

- Our algorithm is a combination of Randomized Weighted Majority instances built on top of each other.
- On average, it makes $\Psi_G(Y) + O\left(\sqrt{|V|\Psi_G(Y)} + |V|\right)$ mistakes.
- On the lower side, for any directed graph G and any integer K, there exists a labeling Y forcing at least $\frac{K}{2}$ mistakes to any online algorithms, while $\Psi_G(Y) \leq K$.

Outline

5 Real world datasets

Citations i the work of j to praise it or criticise it.

Wikipedia i vote for or against j promotion to adminship.

Slashdot i consider j as a friend or foe.

Epinion i trust or not the reviews made by j.

Wik. Edits i reacted to a Wikipedia edit made by j, to enhance it or revert it.

These datasets are severely unbalanced toward the positive class. Hence, we evaluate using the Matthews Correlation Coefficient (MCC):

$$\mathrm{MCC} = \frac{tp \times tn - fp \times fn}{\sqrt{(tp+fp)(tp+fn)(tn+fp)(tn+fn)}} \begin{cases} 1 & \text{all predictions correct} \\ 0 & \text{random predictions} \\ -1 & \text{all predictions incorrect} \end{cases}$$

Our methods

- Our global label propagation algorithm (called L. Prop. here), with a threshold set by cross-validation on E_0.

- We also exploit

$$\widehat{y}(i,j) = \mathrm{SGN}\left((1-\widehat{tr}(i)) + (1-\widehat{un}(j)) - \tau - \frac{1}{2} \right)$$

  in a passive context by computing $\widehat{tr}(i)$ and $\widehat{un}(i)$, and estimating τ, on the training set E_0. We call this method $\mathrm{BLC}(tr, un)$ (Bayes Learning Classifier based on *tr*ollness and *un*trustworthiness).

- For reference, a logistic regression model where each edge (i,j) is associated with the features $[1-\widehat{tr}(i), 1-\widehat{un}(j)]$ computed on E_0 (LogReg).

Competitors

- Global
  - A logistic regression model built on RankNodes scores computed with a PageRank-inspired algorithm tailored to directed graphs with negative edges [3].
  - A global LowRank matrix completion method, assuming that the adjacency matrix is a partial observation of an underlying complete graph with k clusters [4].
- Local
  - A logistic regression model built on a high number of so-called "Bayesian" features defined by [5].
  - A logistic regression model built on 16 Triads features, as signed graphs exhibit specific triangle patterns according to the status theory [6].

Results I

Table 1: 100× MCC results on Epinion as $|E_0|$ grows

	Global	3%	9%	15%	20%	25%	time (ms)
LogReg		43.51	54.85	59.29	61.45	62.95	32
blc(tr, un)		41.39	53.23	57.76	60.06	61.93	**7**
L. Prop.	✓	**51.47**	**58.43**	**61.41**	**63.14**	**64.47**	**1226**
RankNodes	✓	52.04	60.21	62.69	64.13	65.22	**2341**
LowRank	✓	36.84	43.95	48.61	51.43	54.51	121530
Bayesian		31.00	48.24	56.88	61.49	64.45	116838
16 Triads		34.42	49.94	54.56	56.96	58.73	129

Results II

Table 2: 100× MCC results on Citations as $|E_0|$ grows

	Global	3%	9%	15%	20%	25%	time (ms)
LogReg		15.19	26.46	32.98	36.57	39.90	2
blc(tr, un)		15.09	26.40	32.98	36.72	40.16	**<1**
L. Prop.	✓	19.00	30.25	35.73	38.53	41.32	**16**
RankNodes	✓	12.28	24.44	31.03	34.57	38.26	**128**
LowRank	✓	8.85	17.08	22.57	25.57	29.24	1894
Bayesian		10.91	23.75	32.25	36.52	40.32	5398
16 Triads		8.62	16.42	22.01	24.77	27.13	5

Results comments

1. Global methods outperform our local one, however they are much slower, preventing them to scale to larger graphs.

2. Our global method L. Prop. is very competitive in terms of MCC performance in the **small training set** regime while being faster.

3. Our Bayes approximator BLC(tr, un) closely mirrors a more involved LogReg model, making its training useless. Moreover, the learned weights of trollness and trustworthiness are almost equal across all datasets.

Outline

Discussion

We presented two methods to perform edge sign prediction in Directed Signed Social Networks. Both are derived from a simple generative model of edge sign.

BLC(tr, un) is local, thus scalable, and although it requires a large training set to meet its theoretical guarantees[1], it works well in practice.

L. Prop. is global yet faster than state of the art methods while enjoying competitive performance and relying on the same theoretical foundations.

[1] i.e. being Bayes optimal w.h.p. for all edges simultaneously.

Future work

Further directions include:

- Maximizing the utility of a limited query budget in active setting.
- Extending the generative model to weighted graph.
- Designing an adaptive query strategy.
- Exploiting side information.

References I

S.-H. Yang *et al.*, "Friend or frenemy?: Predicting signed ties in social networks," in *Proc. of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2012, pp. 555–564.

H. Kwak *et al.*, "Exploring cyberbullying and other toxic behavior in team competition online games," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 3739–3748.

Z. Wu *et al.*, "The troll-trust model for ranking in signed networks," in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, ACM, 2016, pp. 447–456.

References II

K.-y. Chiang *et al.*, "Prediction and Clustering in Signed Networks: A Local to Global Perspective," *Journal of Machine Learning Research*, vol. 15, pp. 1177–1213, 2014.

D. Song *et al.*, "Link sign prediction and ranking in signed directed social networks," *Social Network Analysis and Mining*, vol. 5, no. 1, 2015.

J. Leskovec *et al.*, "Predicting positive and negative links in online social networks," in *Proc. of the 19th international conference on World wide web*, 2010, p. 641.

P. Auer *et al.*, "Adaptive and self-confident on-line learning algorithms," *J. Comput. Syst. Sci.*, vol. 64, no. 1, pp. 48–75, 2002.

Thank you!
Questions?

Bias term derivation

The in- and out-neighborhood of i is denoted by $\mathcal{E}_{\mathrm{out}}(i)$ and $\mathcal{E}_{\mathrm{in}}(i)$, along with degree quantities: $d_{\mathrm{out}}(i) = |\mathcal{E}_{\mathrm{out}}(i)| = d^-_{\mathrm{out}}(i) + d^+_{\mathrm{out}}(i)$, $d_{\mathrm{in}}(i) = |\mathcal{E}_{\mathrm{in}}(i)| = d^-_{\mathrm{in}}(i) + d^+_{\mathrm{in}}(i)$.

$1 - \widehat{tr}(i) = \frac{\widehat{d}^-_{\mathrm{out}}(i)}{\widehat{d}_{\mathrm{out}}(i)}$ is the empirical probability of drawing a $+1$-labeled edge from $\mathcal{E}_{\mathrm{out}}(i)$, which according to our model is

$$\frac{1}{d_{\mathrm{out}}(i)} \sum_{j \in \mathcal{E}_{\mathrm{out}}(i)} \mathbb{P}(y_{i,j} = 1) = \frac{1}{d_{\mathrm{out}}(i)} \sum_{j \in \mathcal{E}_{\mathrm{out}}(i)} \frac{p_i + q_j}{2}$$

$$= \frac{1}{2}\left(p_i + \frac{1}{d_{\mathrm{out}}(i)} \sum_{j \in \mathcal{E}_{\mathrm{out}}(i)} q_j\right) = \frac{1}{2}\left(p_i + \overline{q}_i\right)$$

where $\overline{q}_i$, being a sample mean of i.i.d. $[0,1]$-valued random variables independently drawn from the prior marginal $\int_0^1 \mu(p,\cdot)\,dp$, concentrates around its expectation μ_q.

The same argument for $(1 - \widehat{un}(j))$ proves that the bias term τ is the same for all edges.

Label propagation full objective

$$
\begin{aligned}
\hat{f}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j}{}_{(i,j)\in E\setminus E_0}) = \sum_{(i,j)\in E} \Bigg(& \frac{1}{2}\left(\frac{1+y_{i,j}}{2} - p_i\right)^2 + \frac{1}{2}\left(\frac{1+y_{i,j}}{2} - q_j\right)^2 \\
& + \left(\frac{p_i + q_j - 1}{2}\right)^2 \Bigg) \\
= & f_{E_0}(\boldsymbol{p}, \boldsymbol{q}) + f_{E\setminus E_0}(\boldsymbol{p}, \boldsymbol{q}, y_{i,j}) \\
& + \frac{1}{2}\sum_{i\in V}\left(d_{\mathrm{out}}(i)\Big(p_i - \frac{1}{2}\Big)^2 + d_{\mathrm{in}}(i)\Big(q_i - \frac{1}{2}\Big)^2 \right)
\end{aligned}
$$

Online algorithm, 1. RWM node instances

For each node i, we predict the sign of edge outgoing from i by relying on two constant experts, always predicting -1 or always predicting $+1$. The best one will make $\Psi_{\text{out}}(i, Y)$ mistakes. We combine them in a Randomized Weighted Majority algorithm (RWM) instance associated with i, call it $RWM_{out}(i)$. The instance expected number of mistakes is therefore [7], denoting by $M(i, j)$ the indicator function of a mistake on edge (i, j)

$$\sum_{j \in \mathcal{E}_{\text{out}}(i)} \mathbb{E}\, M(i, j) = \Psi_{\text{out}}(i, Y) + O\left(\sqrt{\Psi_{\text{out}}(i, Y)} + 1\right)$$

We use the same technique to predict incoming edges of each node j, the instance $RWM_{in}(j)$ having the following average number of mistakes

$$\sum_{i \in \mathcal{E}_{\text{in}}(j)} \mathbb{E}\, M(i, j) = \Psi_{\text{in}}(j, Y) + O\left(\sqrt{\Psi_{\text{in}}(j, Y)} + 1\right)$$

Online algorithm, 2. combining instances

We then define two meta experts: RWM_{out}, which predicts $y_{i,j}$ as $RWM_{out}(i)$, and RWM_{in}, which predicts $y_{i,j}$ as $RWM_{in}(j)$. Summing over all nodes, the number of mistakes of these two experts satisfy

$$\sum_{i \in V} \sum_{j \in \mathcal{E}_{\mathrm{out}}(i)} \mathbb{E}\, M(i,j) = \Psi_{\mathrm{out}}(Y) + O\left(\sqrt{|V|\Psi_{\mathrm{out}}(Y)} + |V|\right)$$

$$\sum_{j \in V} \sum_{i \in \mathcal{E}_{\mathrm{in}}(j)} \mathbb{E}\, M(i,j) = \Psi_{\mathrm{in}}(Y) + O\left(\sqrt{|V|\Psi_{\mathrm{in}}(Y)} + |V|\right)$$

Online algorithm, 3. final prediction

Our final predictor is a RWM combination of RWM_{out} and RWM_{out}, whose expected number of mistakes is

$$\sum_{(i,j)\in E} \mathbb{E}\, M(i,j) = \Psi_G(Y) + O\Bigg(\sqrt{|V|\Psi_G(Y)} + |V| + \sqrt{\Big(\Psi_G(Y) + |V| + \sqrt{|V|\Psi_G(Y)}\Big)}\Bigg)$$

$$= \Psi_G(Y) + O\Big(\sqrt{|V|\Psi_G(Y)} + |V|\Big)$$

Datasets properties

Table 3: Dataset properties.

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	$\frac{\lvert E\rvert}{\lvert V\rvert}$	$\frac{\lvert E^+\rvert}{\lvert E\rvert}$	$\frac{\Psi_{G''}(Y)}{\lvert E\rvert}$	$\frac{\Psi_G(Y)}{\lvert E\rvert}$
Citations	4,831	39,452	8.1	72.33%	.076	.191
Wikipedia	7,114	103,108	14.5	78.79%	.063	.142
Slashdot	82,140	549,202	6.7	77.40%	.059	.143
Epinion	131,580	840,799	6.4	85.29%	.031	.074
Wik. Edits	138,587	740,106	5.3	87.89%	.034	.086

$$\Psi_{G''}(Y) = \min_{\boldsymbol{p},\boldsymbol{q}\in[0,1]^{|V|}} \sum_{(i,j)\in E} \left(\frac{1+y_{i,j}}{2} - \frac{p_i+q_j}{2} \right)^2$$

Additional Results I

Table 4: MCC results on Wikipedia as $|E_0|$ grows

	3%	9%	15%	20%	25%	time
LogReg	32.32	45.57	50.70	<u>52.98</u>	54.49	4
blc(tr, un)	31.83	44.74	49.64	52.00	53.52	1
L. Prop.	<u>33.92</u>	45.75	50.44	52.58	54.22	35
RankNodes	26.90	41.60	48.02	51.42	53.42	210
Bayesian	19.94	38.25	46.82	50.45	52.78	14090
LowRank	19.45	30.75	35.31	38.16	39.94	4859
16 Triads	4.29	24.04	34.42	38.55	41.51	11

Additional Results II

Table 5: MCC results on Slashdot as $|E_0|$ grows

	3%	9%	15%	20%	25%	time
LogReg	32.34	42.16	46.44	48.71	50.23	21
blc(tr, un)	31.78	41.19	45.23	47.79	49.43	6
L. Prop.	**36.62**	**45.70**	49.65	**51.88**	**53.30**	655
RankNodes	42.90	47.46	**48.59**	52.09	53.46	1919
Bayesian	25.11	37.00	43.28	47.03	49.46	77042
LowRank	34.32	39.42	41.09	43.10	44.37	56252
16 Triads	20.95	39.14	46.27	49.44	51.51	78

Additional Results III

Table 6: MCC results on Wik. Edits as $|E_0|$ grows

	3%	9%	15%	20%	25%	time
LogReg	26.02	35.27	38.21	39.58	40.48	28
blc(tr, un)	26.23	35.13	37.72	38.74	39.48	7
L. Prop.	33.92	38.33	38.63	39.16	39.14	824
RankNodes	23.59	33.38	36.81	38.56	39.80	2939
Bayesian	20.02	33.87	40.14	43.37	45.76	103264
LowRank	20.13	26.68	29.97	31.89	34.23	130037
16 Triads	1.11	11.07	18.12	21.53	23.89	104