Aalto University
School of Science
and Technology

Structured Prediction of Network Response

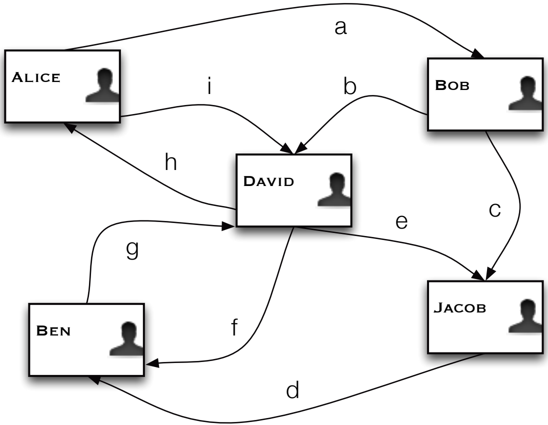
Hongyu Su, Aristides Gionis, Juho Rousu

Helsinki Institute for Information Technology
Department of Information and Computer Science
Aalto University, Finland
hongyu.su@aalto.fi

June 22, 2014

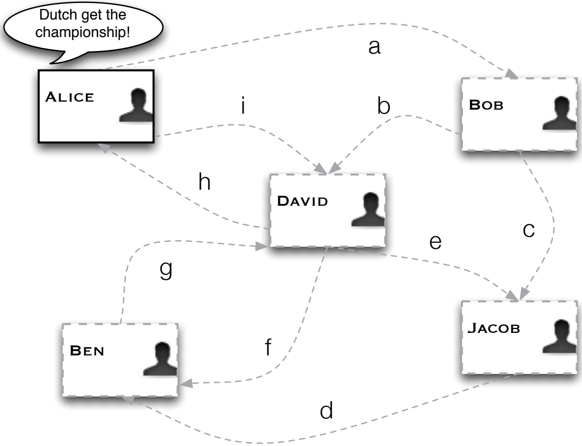
Example: Predicting Network Response

A twitter (follower-ship) network consists of five users.



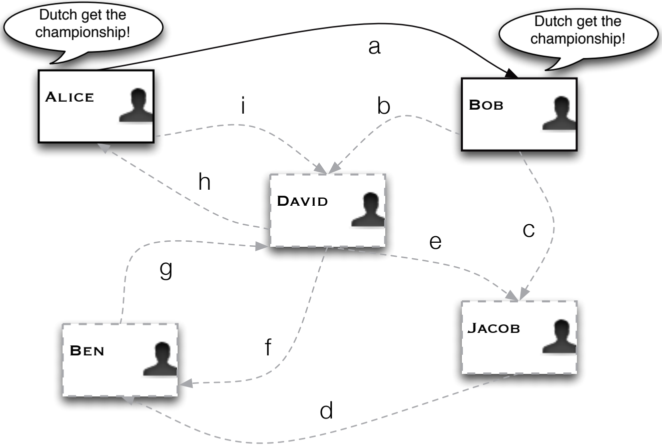
Example: Predicting Network Response

Alice tweets a message after World Cup final.



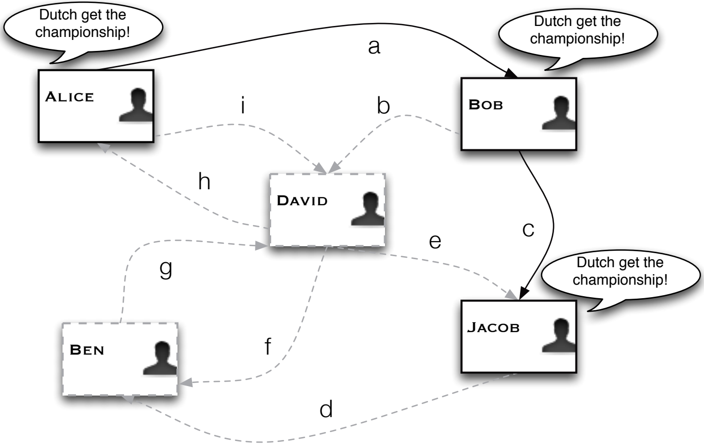
Example: Predicting Network Response

Bob sees the message and retweets the message from Alice.



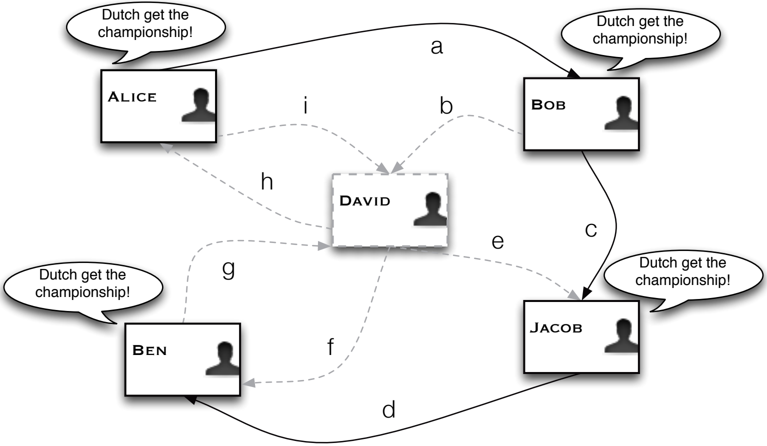
Example: Predicting Network Response

Jacob retweets the message from Bob.



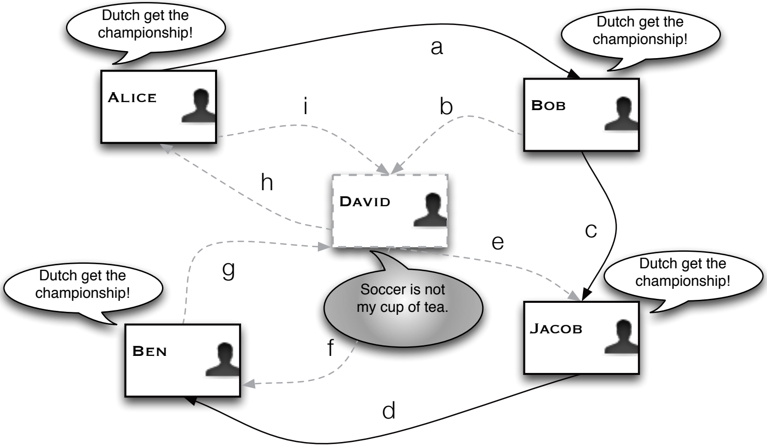
Example: Predicting Network Response

Ben retweets the message from Jacob.



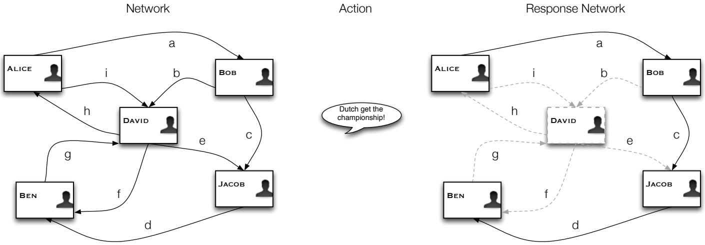
Example: Predicting Network Response

David does not like soccer.



Network Response Problem

- Definition:
  - Given a complex network G, and an action **a** performed on the network.
  - Task: predict the subnetwork that responses to the action.
    - Which nodes $v \in V_{\mathbf{a}}$ perform the action? $V_{\mathbf{a}} = \{Alice, Bob, Jacob, Ben\}$
    - Which directed edges $e \in E_{\mathbf{a}}$ relay the action from one node to its neighbors? $E_{\mathbf{a}} = \{a, c, d\}$

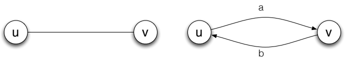


Contributions

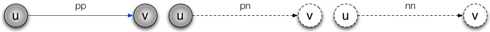
- Context-sensitive prediction. The influence from node u to node v not only depends on their connections but also depends on the action $\mathbf{a}$ under consideration.
- Structure output learning. We model the problem as predicting for each action (e.g. a tweet) a response network (*directed acyclic graph*).
- Efficient inference algorithms to discover the response network:
  - Semidefinite programming relaxation of integer quadratic programming with approximation guarantee.
  - Greedy algorithm as a scalable approach.
- Empirically performance is shown to be better than the state-of-the-art.

Model

- Model is defined on directed network.
  - Any undirected network can be seen as special case by replacing undirected edges with two directed ones.

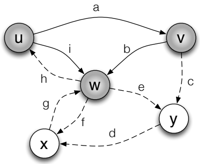


- Notation of edge labels:



- Feature maps:
  - *Input feature*: Encode $\mathbf{a}$ as $\phi(\mathbf{a})$ (e.g. bag-of-word of a tweet).
  - *Output feature*: Encode $G_{\mathbf{a}}$ as $\psi(G_{\mathbf{a}})$ (e.g. a set of edges and their labels)

$$\begin{aligned}\psi(G_{\mathbf{a}}) &= \{a_{pp}, a_{pn}, a_{nn}, b_{pp}, b_{pn}, b_{nn}, c_{pp}, c_{pn}, c_{nn}, \cdots\} \\ &= \{1, \quad 0, \quad 0, \quad 1, \quad 0, \quad 0, \quad 0, \quad 1, \quad 0, \cdots\}\end{aligned}$$



Structure Output Prediction Model

- Compatibility score for $(\mathbf{a}, G_{\mathbf{a}})$: $F(\mathbf{a}, G_{\mathbf{a}}, \mathbf{w}) = \langle \mathbf{w}, \varphi(\mathbf{a}, G_{\mathbf{a}}) \rangle$
  - $\mathbf{w}$ is the feature weight to be learned.
  - $\varphi(\mathbf{a}, G_{\mathbf{a}}) = \phi(\mathbf{a}) \otimes \psi(G_{\mathbf{a}})$ is joint feature map.
  - Intuition: given an action $\mathbf{a}$, the score of correct response graph $(\mathbf{a}, G_{\mathbf{a}})$ should be higher than any incorrect response graph $(\mathbf{a}, G'_{\mathbf{a}})$.

$$F(\mathbf{a}, G_{\mathbf{a}}, \mathbf{w}) > F(\mathbf{a}, G'_{\mathbf{a}}, \mathbf{w}), \quad \forall G'_{\mathbf{a}} \in \mathcal{H}(G)$$

- $\mathbf{w}$ is learned by solving structured output learning problem

$$\begin{aligned}
\min_{\mathbf{w},\xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{m}\xi_i \\
\textbf{s.t.} \quad & F(a_i, G_{\mathbf{a}_i}; \mathbf{w}) > \operatorname*{\textbf{argmax}}_{G'_{\mathbf{a}_i} \in \mathcal{H}(G)} (F(a_i, G'_{\mathbf{a}_i}, \mathbf{w}) \\
& + \ell_G(G_{\mathbf{a}_i}, G'_{\mathbf{a}_i})) - \xi_i, \xi_i \geq 0, \forall i \in \{1, \cdots, m\},
\end{aligned}$$

Inference Problem

- To solve the optimization, we have to solve similar inference problem appeared both in training and in prediction.
- In prediction phase:
  - Given the feature weight **w** and the complex network G.
  - To find out a network $H^* = (V_H, E_H)$ that gives the maximal compatibility score for a given action **a**

$$
\begin{aligned}
H^*(a) &= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} F(\mathbf{a}, H; \mathbf{w}) \\
&= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \langle \mathbf{w}, \phi(a) \otimes \psi(H) \rangle \\
&= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w})
\end{aligned}
$$

Inference Problem

- To solve the optimization, we have to solve similar inference problem appeared both in training and in prediction.
- In prediction phase:
  - Given the feature weight $\mathbf{w}$ and the complex network G.
  - To find out a network $H^* = (V_H, E_H)$ that gives the maximal compatibility score for a given action $\mathbf{a}$

$$
\begin{aligned}
H^*(a) &= \underset{H \in \mathcal{H}(G)}{\mathbf{argmax}}\, F(\mathbf{a}, H; \mathbf{w}) \\
&= \underset{H \in \mathcal{H}(G)}{\mathbf{argmax}}\, \langle \mathbf{w}, \phi(a) \otimes \psi(H) \rangle \\
&= \underset{H \in \mathcal{H}(G)}{\mathbf{argmax}} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w})
\end{aligned}
$$

Inference Problem

- To solve the optimization, we have to solve similar inference problem appeared both in training and in prediction.
- In prediction phase:
  - Given the feature weight **w** and the complex network G.
  - To find out a network $H^* = (V_H, E_H)$ that gives the maximal compatibility score for a given action **a**

$$
\begin{aligned}
H^*(a) &= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} F(\mathbf{a}, H; \mathbf{w}) \\
&= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \langle \mathbf{w}, \phi(a) \otimes \psi(H) \rangle \\
&= \operatorname*{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w})
\end{aligned}
$$

Inference Problem

- To solve the optimization, we have to solve similar inference problem appeared both in training and in prediction.
- In prediction phase:
  - Given the feature weight $\mathbf{w}$ and the complex network G.
  - To find out a network $H^* = (V_H, E_H)$ that gives the maximal compatibility score for a given action $\mathbf{a}$

$$
\begin{aligned}
H^*(a) &= \operatorname*{argmax}_{H \in \mathcal{H}(G)} F(\mathbf{a}, H; \mathbf{w}) \\
&= \operatorname*{argmax}_{H \in \mathcal{H}(G)} \langle \mathbf{w}, \phi(a) \otimes \psi(H) \rangle \\
&= \operatorname*{argmax}_{H \in \mathcal{H}(G)} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w})
\end{aligned}
$$

Inference Problem

- To solve the optimization, we have to solve similar inference problem appeared both in training and in prediction.
- In prediction phase:
  - Given the feature weight **w** and the complex network G.
  - To find out a network $H^* = (V_H, E_H)$ that gives the maximal compatibility score for a given action **a**

$$\begin{aligned} H^*(a) &= \mathop{\mathbf{argmax}}_{H \in \mathcal{H}(G)} F(\mathbf{a}, H; \mathbf{w}) \\ &= \mathop{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \langle \mathbf{w}, \phi(a) \otimes \psi(H) \rangle \\ &= \mathop{\mathbf{argmax}}_{H \in \mathcal{H}(G)} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w}) \end{aligned}$$

NP-hardness

$$H^*(a) = \underset{H \in \mathcal{H}(G)}{\mathbf{argmax}} \sum_{e \in E^H} s_{\mathbf{y}_e}(e, a, \mathbf{w}) \quad (1)$$

Lemma

Finding the graph that maximizes Eq. (1) is an $\mathcal{NP}$-hard problem.

Proof.

Reduction from MAX-CUT problem. □

Approximate Inference through SDP relaxation

- SDP inference:
  - We formulate the inference problem as *integer quadratic programming* (IQP).
    - Introduce for each node $u \in V$ a binary variable $x_u \in \{-1, +1\}$.
    - Introduce a special variable $x_0 \in \{-1, +1\}$ to distinguish activated node.

$$
\begin{aligned}
\textbf{max} \ \frac{1}{4} \sum_{(u,v)\in E} & [s_{pn}(u,v)(1 + x_0 x_u - x_0 x_v - x_u x_v) \\
& + s_{nn}(u,v)(1 - x_0 x_u - x_0 x_v + x_u x_v) \\
& + s_{pp}(u,v)(1 + x_0 x_u + x_0 x_v + x_u x_v)] \\
\textbf{s.t.} \quad & x_0, x_u, x_v \in \{-1, +1\}, \ \text{for all } u, v \in V,
\end{aligned}
$$

  - IQP is solved by *semidefinite programming relaxation* (SDP).
  - Optimization guarantee $E[Z] \geq (\alpha - \epsilon) Z_R$ with $\alpha > 0.796$, Z is objective achieved by SDP, Z_R is objective of IQP.

GREEDY Inference

- GREEDY inference:
  - We have shown that the inference problem can be expressed equivalently as nodes and their scores

$$H^*(\mathbf{a}) = \underset{H \in \mathcal{H}(G)}{\mathbf{argmax}} \sum_{v_i \in V_p^H} F_m(v_i).$$

  - The greedy algorithm iteratively maximizes the equation by adding one node into activited node set V_p^H in each iteration.
  - The time complexity for greedy inference algorithm is $\Theta(|E| \log |V|)$.
  - We are able to run GREEDY algorithm on network with upto 2000 nodes.

Experiment: Context-sensitive Prediction

- Experiment settings:
  - We assume action is known (e.g. bag-of-word of a tweet).
  - Task is to predict the response network given an action.
  - *Predicted Subgraph Coverage* (PSC) is the relative size of correctly predicted subgraph in terms of node labels.)
- Result:

Dataset	Node Accuracy			Node F_1 Score			Edge Acc		*PSC*		
	SVM	MMCRF	SPIN	SVM	MMCRF	SPIN	SVM	SPIN	SVM	MMCRF	SPIN
memeS	**73.4**	68.0	*72.2*	39.0	*39.8*	**47.1**	**62.7**	45.6	23.4	*25.3*	**33.6**
memeM	**82.1**	79.0	*81.5*	29.1	*30.1*	**38.0**	61.1	**68.8**	18.6	*18.8*	**28.3**
memeL	**89.9**	88.3	*89.8*	26.7	*27.1*	**35.0**	45.5	**80.0**	17.7	*18.9*	**27.6**
M700	91.9	**94.1**	*92.1*	*13.8*	7.3	**14.2**	26.3	**93.0**	*29.4*	23.9	**34.4**
M1k	94.1	**95.8**	*94.2*	**10.9**	3.5	*9.3*	26.6	**94.7**	*33.7*	16.6	**35.2**
M2k	*96.8*	**97.6**	96.7	**6.2**	1.4	*3.4*	25.3	**97.6**	**34.6**	9.6	*14.7*
L700	89.7	**92.4**	*89.7*	*16.2*	9.4	**17.3**	26.5	**90.4**	*9.5*	6.7	**12.5**
L1k	*92.4*	**94.4**	91.5	*12.4*	6.4	**13.9**	26.4	**92.3**	*6.1*	4.4	**8.4**
L2k	*92.5*	**94.5**	91.9	*12.3*	5.4	**12.7**	26.5	**93.2**	*6.0*	2.9	**7.2**
Geom.	85.5	*86.4*	**86.6**	*19.8*	12.6	**20.3**	32.6	**79.7**	*18.9*	14.2	**21.7**

Experiment: Context-free Prediction

- Experiment settings:
  - We assume action is unknown.
  - Task is to predict directed edges from a cascade of actions and compare the results against other influence network prediction methods.
  - The measure of success is *Precision@K*, where we ask for top-*K* percent edge predictions and compute the precision.
- Result:

Dataset	**Model**	**T** (10^3**s**)	Precision @ *K*					
			10%	20%	30%	40%	50%	60%
memeS	SPIN	*5.50*	**82.9**	**81.0**	**76.0**	**74.0**	**74.0**	**70.0**
	ICM-EM	**0.01**	60.3	63.5	65.1	62.0	62.0	61.5
	NETRATE	5.83	*76.2*	*73.8*	*70.4*	*68.7*	*68.7*	*66.8*
memeM	SPIN	*5.52*	**82.7**	**72.1**	**70.5**	**69.2**	**69.2**	**67.9**
	ICM-EM	**0.02**	56.3	55.3	56.8	57.4	57.4	56.3
	NETRATE	13.93	*61.2*	*64.6*	*62.9*	*62.5*	*62.5*	*62.4*
memeL	SPIN	*4.75*	**82.2**	**73.6**	**69.1**	**66.7**	**66.7**	**65.9**
	ICM-EM	**0.01**	52.1	55.7	54.2	56.5	56.5	56.7
	NETRATE	12.63	*56.5*	*57.8*	*60.0*	*59.3*	*59.3*	*59.4*

Conclusion

- We have developed a structured output learning approach for network response prediction problem.
- To outperform competing methods, our model takes the advantage of two sources of prior knowledge:
  - The context given by the action descriptions.
  - The structure of an underlying network (e.g. followership in Twitter, friendship in Facebook).
- The inference problem is $\mathcal{NP}$-hard, and is in practice tackled by two algorithms:
  - SDP relaxation with approximation guarantee.
  - a fast GREEDY heuristics.

- Poster No. 565 tonight.
- Thank you !