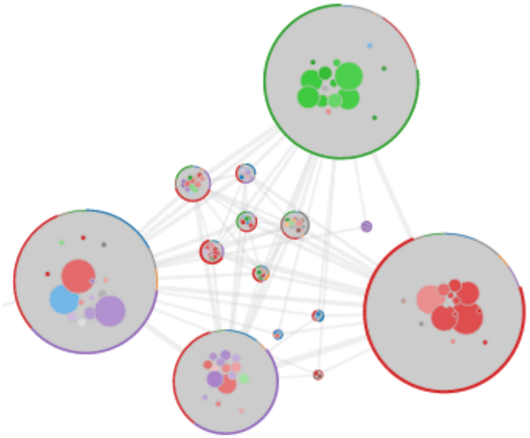
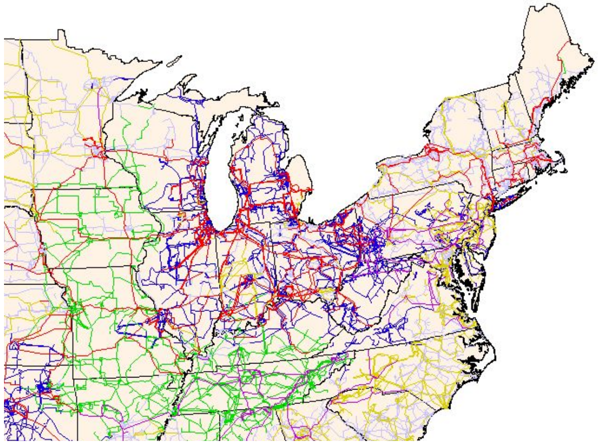
NetFair: Towards Fair Network Mining

Hanghang Tong
University of Illinois at Urbana-Champaign
htong@illinois.edu, http://tonghanghang.org

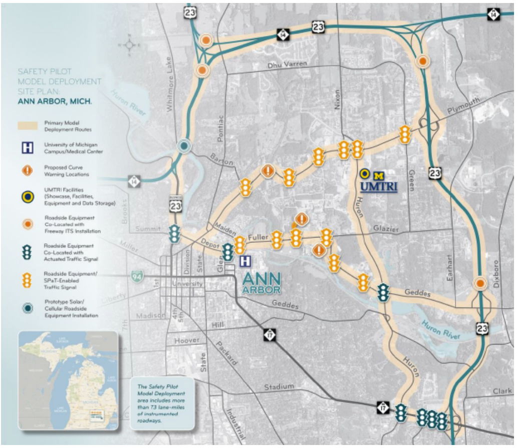
Observation: Networks & Graphs Are Everywhere!



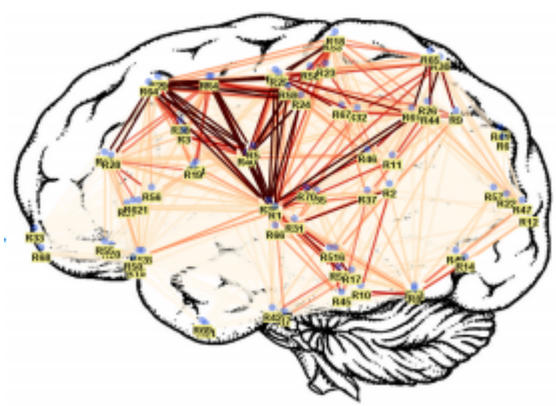
Collaboration Networks



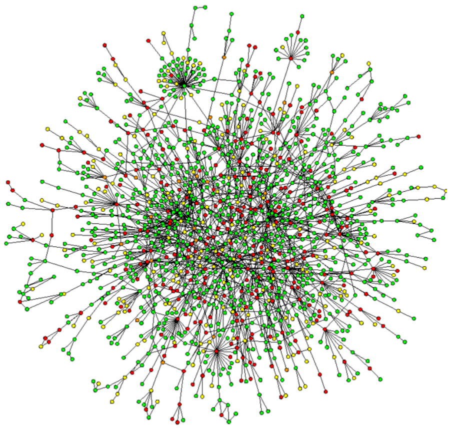
US Power Grid



Traffic Network



Brain Networks



Biological Networks

Hospital Networks

This Talk: Networks = Graphs

Research Theme: Understand and Utilize Networks

Applications

Social Networks | Bioinformatics | Neuroscience | Transportation | : | Commerce

Algorithms

Prediction
[KDD19, 15c,17d]
[KDD15d,14b,08b]
[TKDE17a, CIKM18]
[SDM15b, 16]
[WWW13,14,17a]
[KDD12a,11a]
[PKDD12, ICDm10a]
[IJCAI19a-b]

Inference
[WWW19b, AAAI19a
KDD17c,KDD16a]
[TKDE17b, CIKM16]
[WWW15,CIKM14a]
[SDM14b,IJCAI15]
[ICDM13b, NIPS12]
[KDD09, 11b]
[WCRE'11, WSDM19]

Control
[SAM'17, ICDM'15]
[KDD16c,18b,TKDD17]
[Bigdata18, SDM15a, d]
[SDM14a,13b]
[CIKM12a]
[ICDM10b]
[PKDD'10]
[Networking'10]

Detection
[WWW17b, 19a]
[KDD15b, 17b,d, 18a]
[SDM17,15c,12a-b]
[CIKM17b, 14b
[KDD12b,10,08b]
[TRB15, SAM12]
[CIKM12b,08]
[AAAI'11]

Querying
[KDD14a, WWW19a]
[KDD15a,16b,17a]
[BigData14]
[SDM13a,12b,11a]
[KDDa-c,07a-b,06]
[ICDM08,06]
[CIKM09,KAIS08]
[SDM08,SAM08]

Theories

Hardness
[TKDE16,TKDD16,NIPS12,
ICDM15, KDD11b,PKDD08
KDD18a]

Optimality
[NIPS19a, TKDE16,
ICDM10, CIKM12
KDD17d, 16a-b,08b, SDM08]

...

Complexity
[TKDE17, KDD17a-d, 16a-c
ICDM15,KDD14a,CIKM12
CIKM09,SDM08,ICDM06]

Equivalence
[NIPS19b, TKDD17,
KDD16b,SAM17,
SDM15, KDD11c,J.VLDB12]

Systems

Visualization
[TVCG17, CIKM17a]
[CHI15,17, IUI15]
[PacificViz14]
[ICDM14]
[ICDM13a]
[TKDE14]
[SDM13a]
[KDD12c]
[VLDB06]

Management
[SIGMOD17]
[AVI'16,ICDE'16]
[TKDE'15]
[J. VLDB12]
[P. IEEE12]
[KDD11c]

Results & Impacts

Publications
- 100+ papers
- 10,000+ citations
- 40+ h-index
- 20+ highly-cited
- 20+ patents

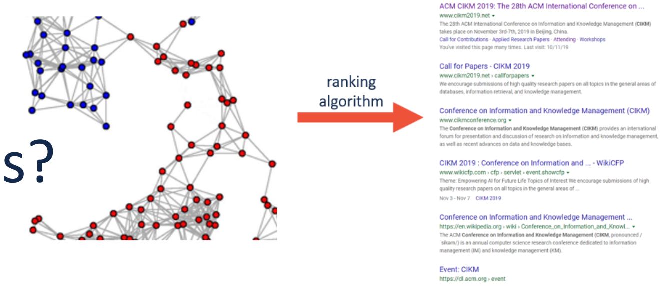
Awards
- test-of-time (ICDM'15)
- NSF CAREER (2017)
- SDM/IBM award (2018)
- Tao Li Award (2019)
- 4 best papers,7 finalists

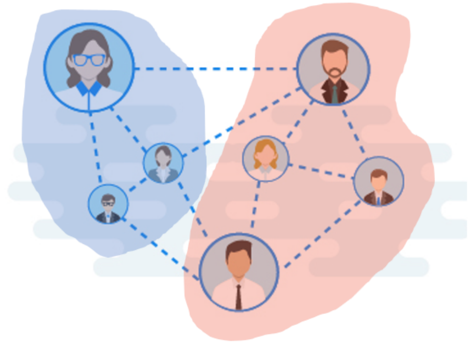
Grants
- $42M in total
- $27M since ASU
- $6M+ as PI
- $4M+ personal share
- 7 agencies

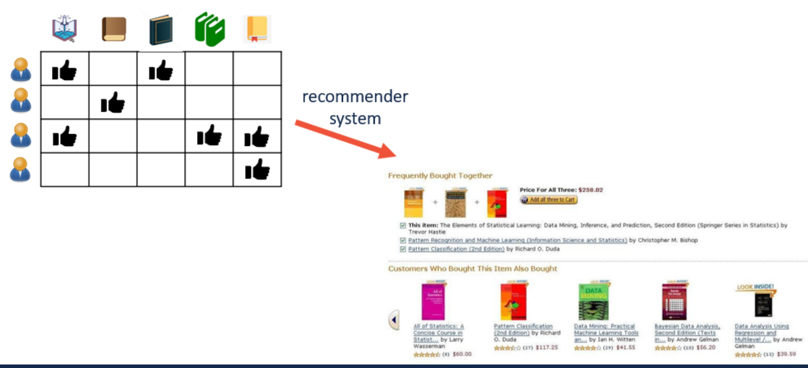
Tech. Transfer
- IBM (Gbase → System G. → Graphen.ai)
- G-Ray/MAGE
- MIT Lincoln Lab
- Lawrence Livermore Natl Lab
- DARPA SIGMA+

Network Mining: The Who & What Questions

- **Who** are in the same online community?
- **Who** is the key to bridge two academic areas?
- **Who** is the master criminal mind?
- **Who** started a misinformation campaign?
- **Which** items shall we recommend to a user?
- **Which** gene is most relevant to a given disease?
- **Which** webpage is most important?
- **Which** tweet is likely to go viral?
- **Which** transaction looks suspicious?
- …

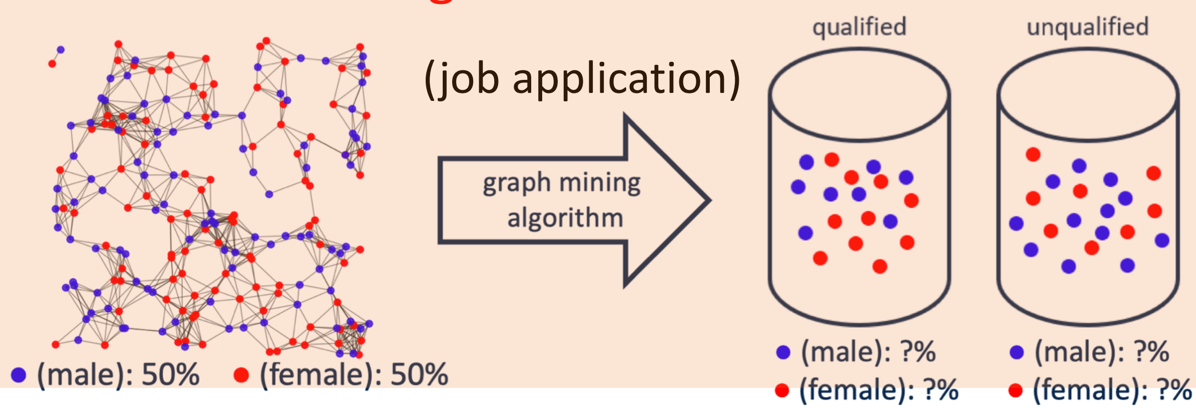






Network Mining: The Why & How Questions

- **How** to ensure the mining is fair?



This Talk

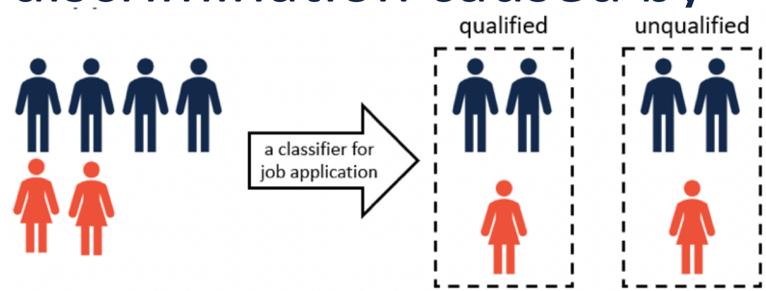
- **Why** do two seemingly different users are in the same community?
- **Why** is a particular tweet more likely to go viral than another?
- **Why** does the algorithm `think' a transaction looks suspicious?
- **How** does an influential researcher bridge two areas?
- **How** do fake review skew the recommendation results?
- **How** do the mining results relate to the input graph topology?

Roadmap

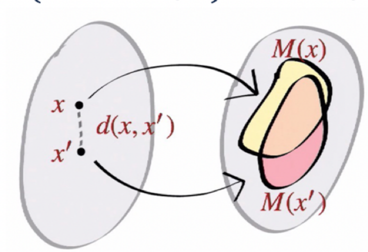
- ☑ Motivations
- ➡ InFoRM: Individual Fairness on Graph Mining
  - InFoRM Introduction
  - InFoRM Measures
  - InFoRM Algorithms
  - InFoRM Cost
- Some Other Work
- Future Directions

Algorithmic Fairness in Machine Learning

- **Goal:** minimize unintentional discrimination caused by machine learning algorithms
- **Existing Measures**
  - Group fairness
    - Disparate impact [1]
    - Statistical parity [2]
    - Equal odds [3]
  - Counterfactual fairness [4]
  - Individual fairness [5]



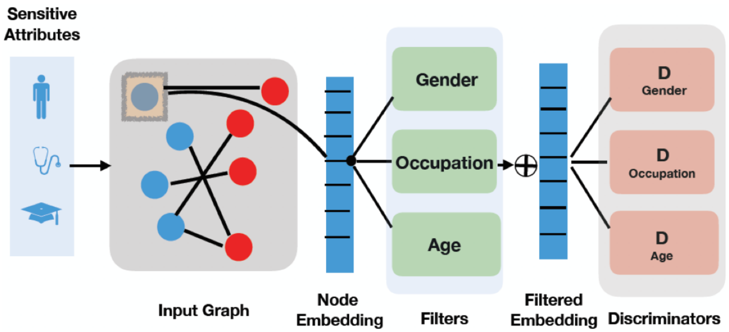
$$d_1(M(x), M(y)) \leq d_2(x, y)$$



- **Limitation:** IID assumption in traditional machine learning
  - Might be violated by the non-IID nature of graph data

[1] Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., & Venkatasubramanian, S.. Certifying and Removing Disparate Impact. KDD 2015.
[2] Chouldechova, A., & Roth, A.. The Frontiers of Fairness in Machine Learning. arXiv.
[3] Hardt, M., Price, E., & Srebro, N.. Equality of Opportunity in Supervised Learning. NIPS 2016.
[4] Kusner, M. J., Loftus, J., Russell, C., & Silva, R.. Counterfactual Fairness. NIPS 2017.
[5] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R.. Fairness through Awareness. ITCS 2012.

Algorithmic Fairness in Graph Mining

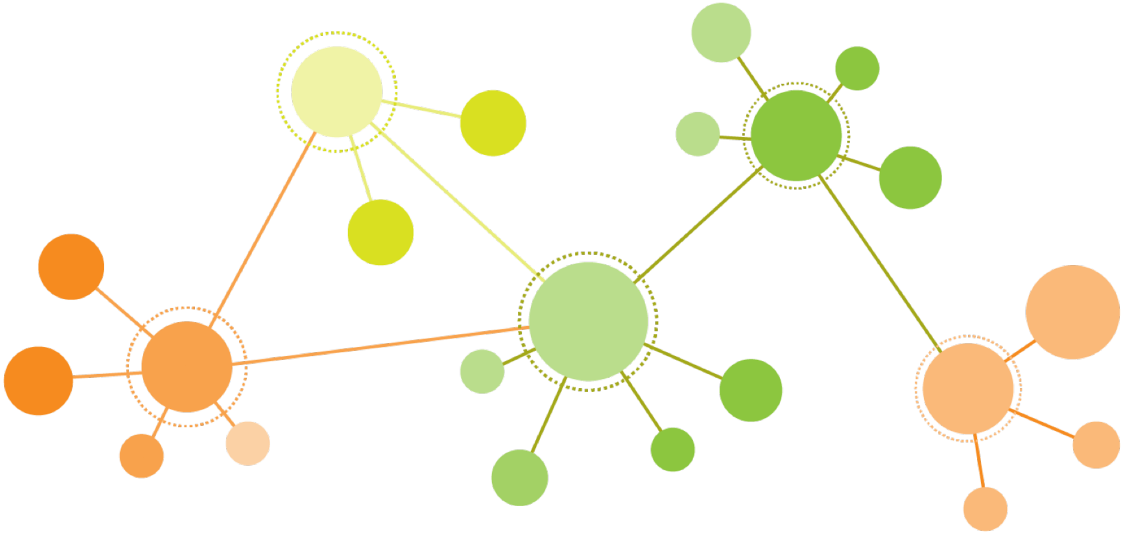


- **Fair Spectral Clustering** [1]
  - **Fairness notion:** disparate impact
- **Fair Graph Embedding**
  - Fairwalk [2], compositional fairness constraints [3]
    - **Fairness notion:** statistical parity
  - MONET [4]
    - **Fairness notion:** orthogonality of metadata and graph embedding
- **Fair Recommendation**
  - Information neural recommendation [5]
    - **Fairness notion:** statistical parity
  - Fairness for collaborative filtering [6]
    - **Fairness notion:** four metrics that measure the differences in estimation error between ground-truth and predictions across protected and unprotected groups
- **Observation:** all of them focus on group-based fairness!

[1] Kleindessner, M., Samadi, S., Awasthi, P., & Morgenstern, J.. Guarantees for Spectral Clustering with Fairness Constraints. ICML 2019.
[2] Rahman, T. A., Surma, B., Backes, M., & Zhang, Y.. Fairwalk: Towards Fair Graph Embedding. IJCAI 2019.
[3] Bose, A. J., & Hamilton, W. L.. Compositional Fairness Constraints for Graph Embeddings. ICML 2019.
[4] Palowitch, J., & Perozzi, B.. Monet: Debiasing Graph Embeddings via the Metadata-Orthogonal Training Unit. arXiv.
[5] Kamishima, T., Akaho, S., Asoh, H., & Sakuma, J.. Enhancement of the Neutrality in Recommendation. RecSys 2012 Workshop.
[6] Yao, S., & Huang, B.. Beyond Parity: Fairness Objectives for Collaborative Filtering. NIPS 2017.

InFoRM: Individual Fairness on Graph Mining

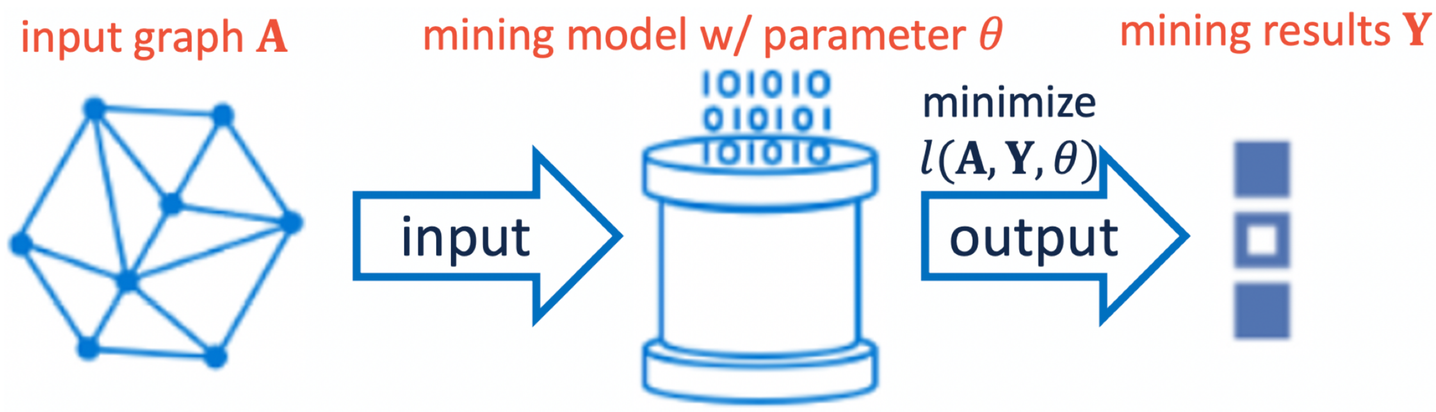
- **Research Questions**
  - **Q1. Measures:** how to quantitatively measure individual bias?
  - **Q2. Algorithms:** how to enforce individual fairness?
  - **Q3. Cost:** what is the cost of individual fairness?



J. Kang, J. He, R. Maciejewski and H. Tong: InFoRM: Individual Fairness on Graph Mining. KDD 2020

Graph Mining Algorithms

- **Graph Mining: An Optimization Perspective**



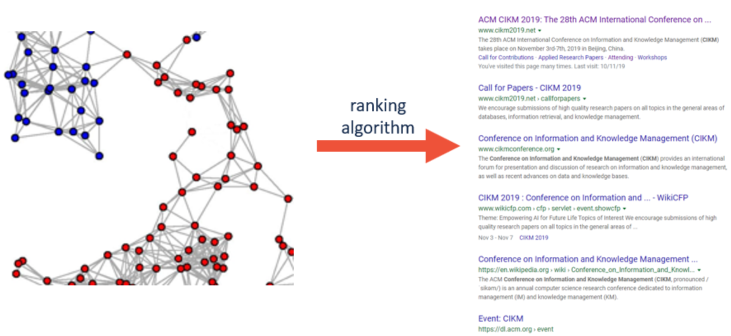
- **Input:**
  - Input graph $\mathbf{A}$
  - Model parameters θ
- **Output:** mining results $\mathbf{Y}$
  - Examples: ranking vectors, class probabilities, embeddings

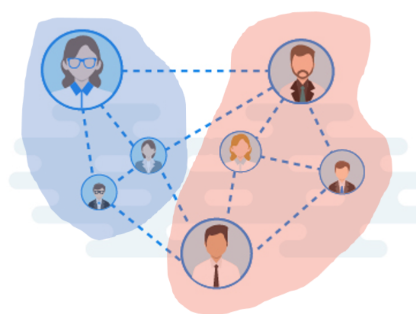
minimize task-specific loss function $l(\mathbf{A}, \mathbf{Y}, \theta)$

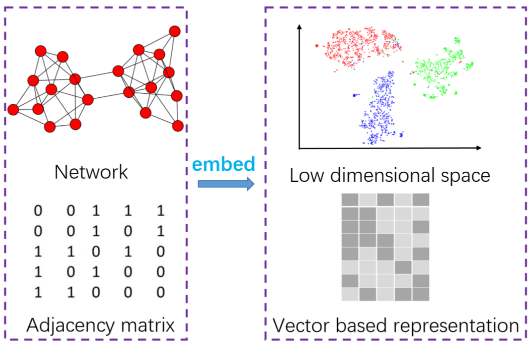
Classic Graph Mining Algorithms

Examples of Classic Graph Mining Algorithm

Mining Task	Task Specific Loss Function $l()$	Mining Result Y^*	Parameters
PageRank	$\min_{\mathbf{r}} c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\lVert\mathbf{r}-\mathbf{e}\rVert_F^2$	PageRank vector $\mathbf{r}$	damping factor c teleportation vector $\mathbf{e}$
Spectral Clustering	$\min_{\mathbf{U}} \mathrm{Tr}\,(\mathbf{U}'\mathbf{L}\mathbf{U})$ s.t. $\mathbf{U}'\mathbf{U}=\mathbf{I}$	eigenvectors $\mathbf{U}$	# clusters k
LINE (1st)	$\min_{\mathbf{X}} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{A}[i,j]\left(\log g(-\mathbf{X}[j,:]\mathbf{X}[i,:]')\right) + b\mathbb{E}_{j'\sim P_n}[\log g(-\mathbf{X}[j',:]\mathbf{X}[i,:]')]$	embedding matrix $\mathbf{X}$	embedding dimension d # negative samples b





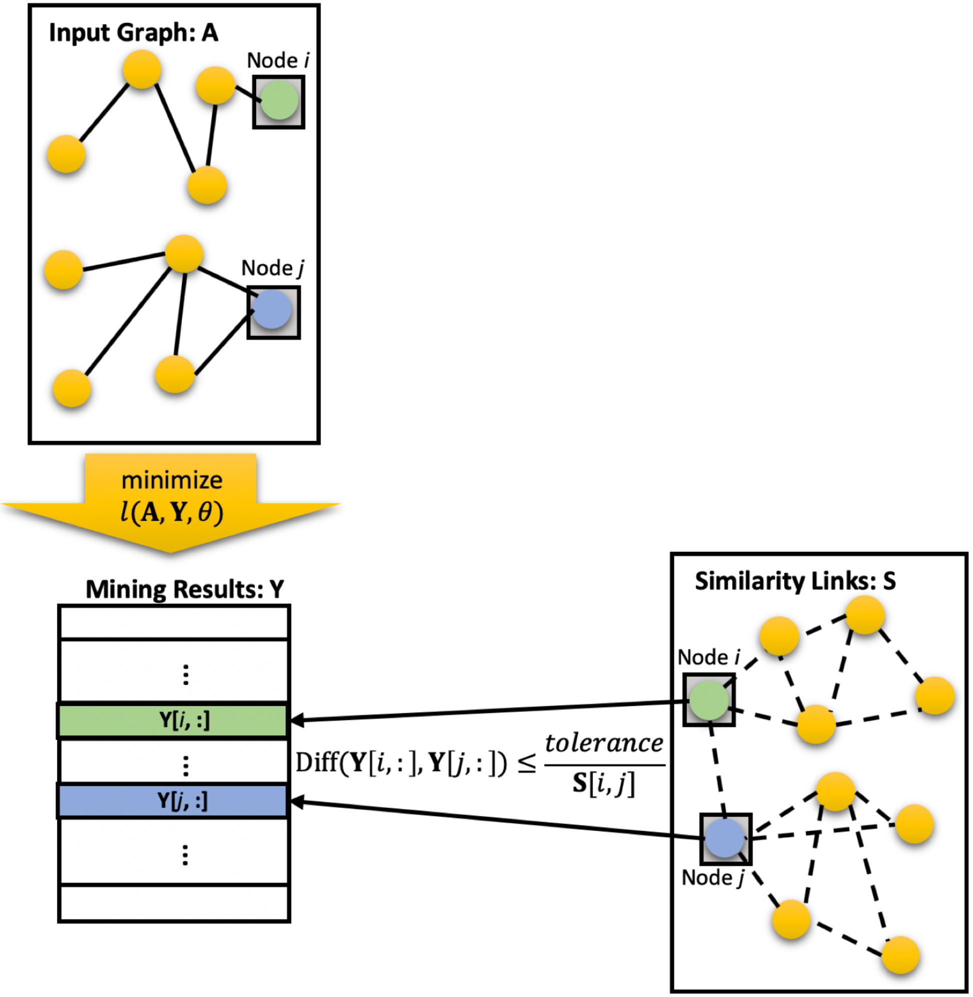


Roadmap

- ☑ Motivations
- InFoRM: Individual Fairness on Graph Mining
  - ☑ InFoRM Introduction
  - ➡ InFoRM Measures
  - InFoRM Algorithms
  - InFoRM Cost
- Some Other Work
- Future Directions

Problem Definition: InFoRM Measures

- **Questions**
  - How to determine if the mining results are fair?
  - How to quantitatively measure the overall bias?
- **Input**
  - Node-node similarity matrix $\mathbf{S}$
    - Non-negative, symmetric
  - Graph mining algorithm $l(\mathbf{A}, \mathbf{Y}, \theta)$
    - Loss function $l(\cdot)$
    - Additional set of parameters θ
  - Fairness tolerance parameter ϵ
- **Output**
  - binary decision on whether the mining results are fair
  - individual bias measure $\text{Bias}(\mathbf{Y}, \mathbf{S})$

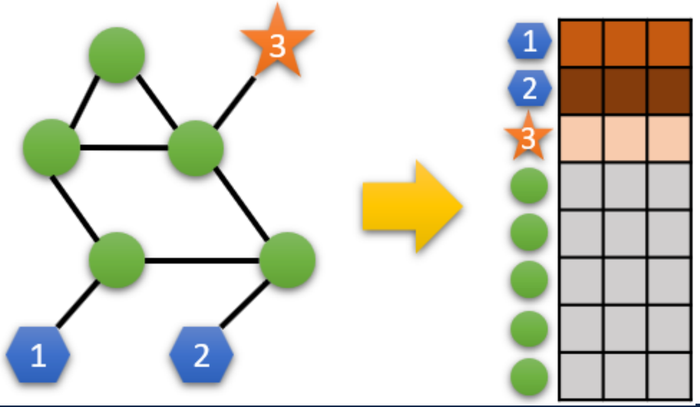


Measuring Individual Bias: Formulation

- **Principle:** similar nodes → similar mining results
- **Mathematical Formulation**

$$\|\mathbf{Y}[i,:]-\mathbf{Y}[j,:]\|_F^2 \leq \frac{\epsilon}{\mathbf{S}[i,j]} \quad \forall i,j=1,\ldots,n$$

  - **Intuition:** if $\mathbf{S}[i,j]$ is high, $\frac{\epsilon}{\mathbf{S}[i,j]}$ is small → push $\mathbf{Y}[i,:]$ and $\mathbf{Y}[j,:]$ to be more similar
  - **Observation:** Inequality should hold for *every* pairs of nodes i and j
    - **Problem:** too restrictive to be fulfilled
- **Relaxed Criteria:** $\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{Y}[i,:]-\mathbf{Y}[j,:]\|_F^2\mathbf{S}[i,j] = 2\mathrm{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y}) \leq m\epsilon = \delta$



Measuring Individual Bias: Solution

- **InFoRM (Individual Fairness on Graph Mining)**
  - Given (1) a graph mining results $\mathbf{Y}$, (2) a symmetric similarity matrix $\mathbf{S}$ and (3) a constant fairness tolerance δ
  - $\mathbf{Y}$ is individually fair w.r.t. $\mathbf{S}$ if it satisfies

$$\mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y}) \leq \frac{\delta}{2}$$

  - Overall individual bias is $\mathrm{Bias}(\mathbf{Y}, \mathbf{S}) = \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$

[1] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R.. Fairness through Awareness. ITCS 2012.

Lipschitz Property of Individual Fairness

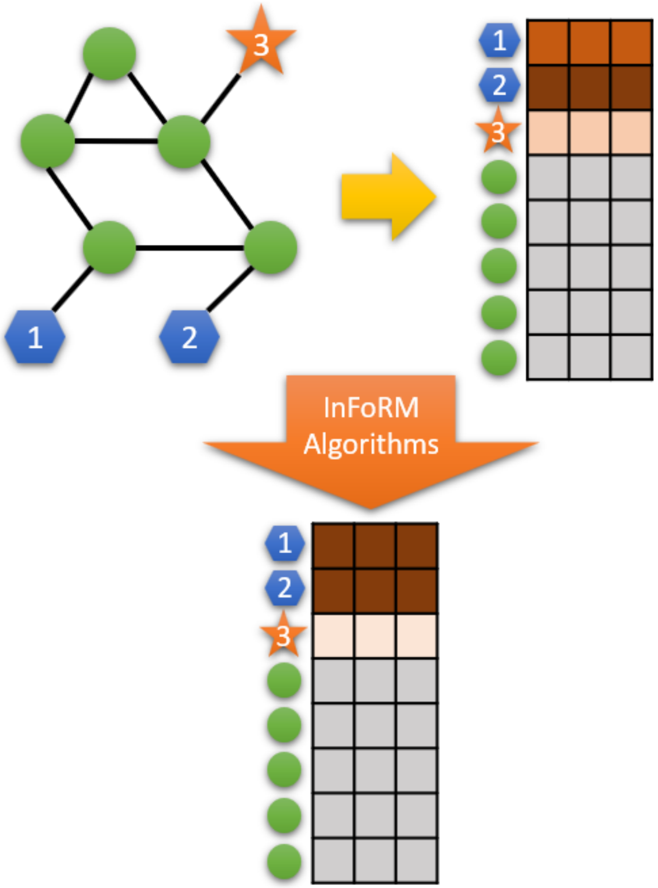
- **Connection to Lipschitz Property**
  - $(\boldsymbol{D_1}, \boldsymbol{D_2})$**-Lipschitz property [1]:** a function f is (D_1, D_2)-Lipschitz if it satisfies
    $$D_1\big(f(i), f(j)\big) \leq L D_2(i, j), \forall (x, y)$$
    - L is Lipschitz constant
  - InFoRM naturally satisfies (D_1, D_2)-Lipschitz property as long as
    - $f(i) = \mathbf{Y}[i, :]$
    - $D_1\big(f(i), f(j)\big) = \|\mathbf{Y}[i, :] - \mathbf{Y}[j, :]\|_2^2$, $D_2(i, j) = \frac{1}{\mathbf{S}[i, j]}$
  - Lipschitz constant of InFoRM is ϵ

Roadmap

- ☑ Motivations
- InFoRM: Individual Fairness on Graph Mining
  - ☑ InFoRM Introduction
  - ☑ InFoRM Measures
  - ➡ InFoRM Algorithms
  - InFoRM Cost
- Some Other Work
- Future Directions

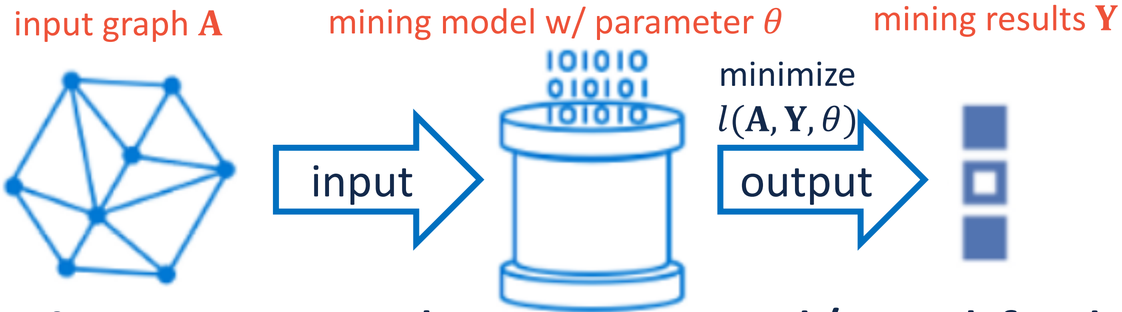
Problem Definition: InFoRM Algorithms

- **Question:** how to mitigate the bias of the mining results?
- **Input**
  - – Node-node similarity matrix $\mathbf{S}$
  - – Graph mining algorithm $l(\mathbf{A}, \mathbf{Y}, \theta)$
  - – Individual bias measure $\text{Bias}(\mathbf{Y}, \mathbf{S})$
    - Defined in the previous problem (InFoRM Measures)
- **Output:** revised mining results $\mathbf{Y}^*$ that minimizes
  - – Task-specific loss function $l(\mathbf{A}, \mathbf{Y}, \theta)$
  - – Individual bias measure $\text{Bias}(\mathbf{Y}, \mathbf{S})$



Mitigating Individual Bias: How To

- **Graph Mining Pipeline**



- **Observation:** Bias can be introduced/amplified in each component
  - **Solution:** bias can be mitigated in each part
- **Algorithmic Frameworks**
  - Debiasing the input graph
  - Debiasing the mining model
  - Debiasing the mining results

mutually complementary

Debiasing the Input Graph

- **Goal:** bias mitigation via a pre-processing strategy
- **Intuition:** learn a new topology of graph $\tilde{\mathbf{A}}$ such that
  - $\tilde{\mathbf{A}}$ is as similar to the original graph $\mathbf{A}$ as possible
  - Bias of mining results on $\tilde{\mathbf{A}}$ is minimized
- **Optimization Problem**

consistency in graph topology

$$\min_{\mathbf{Y}} \quad J = \left\|\tilde{\mathbf{A}} - \mathbf{A}\right\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$$

$$\text{s.t.} \quad \mathrm{Y} = \mathrm{argmin}_{\mathbf{Y}} \; l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta)$$

bias measure

- **Challenge:** bi-level optimization
  - **Solution:** exploration of KKT conditions [1, 2]

[1] Kang, J., & Tong, H.. N2N: Network Derivative Mining. CIKM 2019.
[2] Mei, S., & Zhu, X.. Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners. AAAI 2015.

Debiasing the Input Graph

- Considering the KKT conditions,

$$\min_{\mathbf{Y}} \quad J = \left\|\widetilde{\mathbf{A}} - \mathbf{A}\right\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L}_{\mathbf{S}}\mathbf{Y})$$

$$\text{s.t.} \quad \partial_{\mathbf{Y}} l\left(\widetilde{\mathbf{A}}, \mathbf{Y}, \theta\right) = 0$$

- **Proposed Method**
  - (1) Fix $\widetilde{\mathbf{A}}$ ($\widetilde{\mathbf{A}} = \mathbf{A}$ at initialization), find $\mathbf{Y}$ using current $\widetilde{\mathbf{A}}$
  - (2) Fix $\mathbf{Y}$, update $\widetilde{\mathbf{A}}$ by gradient descent
  - (3) Iterate between (1) and (2)
- **Problem:** how to calculate gradient w.r.t. $\widetilde{\mathbf{A}}$?

[1] Kang, J., & Tong, H.. N2N: Network Derivative Mining. CIKM 2019.
[2] Mei, S., & Zhu, X.. Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners. AAAI 2015.

Debiasing the Input Graph

- **Calculating Gradient**

key component to calculate

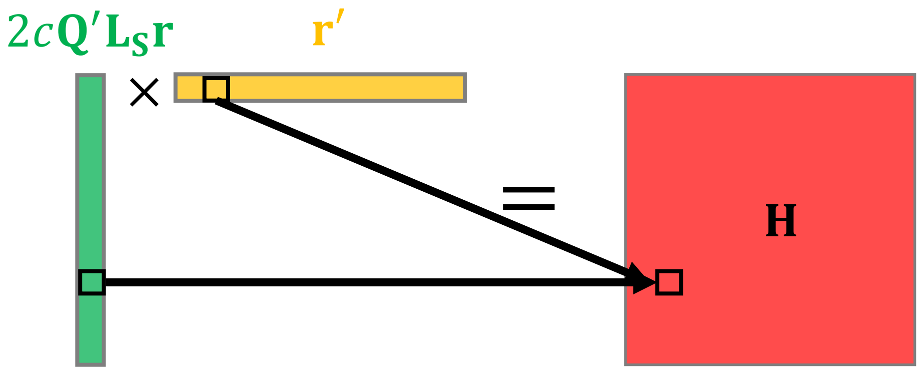
$$\frac{\partial J}{\partial \tilde{\mathbf{A}}} = 2(\tilde{\mathbf{A}} - \mathbf{A}) + \alpha \left[\mathrm{Tr}\left(2\tilde{\mathbf{Y}}\mathbf{L_S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]} \right) \right]$$

$$\frac{\mathrm{d}J}{\mathrm{d}\tilde{\mathbf{A}}} = \begin{cases} \frac{\partial J}{\partial \tilde{\mathbf{A}}} + (\frac{\partial J}{\partial \tilde{\mathbf{A}}})' - \mathrm{diag}\left(\frac{\partial J}{\partial \tilde{\mathbf{A}}}\right), & \text{if undirected} \\ \frac{\partial J}{\partial \tilde{\mathbf{A}}}, & \text{if directed} \end{cases}$$

  - $\tilde{\mathbf{Y}}$ satisfies $\partial_{\mathbf{Y}} l(\tilde{\mathbf{A}}, \mathbf{Y}, \theta) = 0$
  - $\mathbf{H} = \left[\mathrm{Tr}\left(2\tilde{\mathbf{Y}}\mathbf{L_S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]} \right) \right]$ is a matrix with $\mathbf{H}[i,j] = \mathrm{Tr}\left(2\tilde{\mathbf{Y}}\mathbf{L_S} \frac{\partial \tilde{\mathbf{Y}}}{\partial \tilde{\mathbf{A}}[i,j]} \right)$
- **Question:** how to efficiently calculate $\mathbf{H}$?

Instantiation #1: PageRank

- **Goal:** efficiently calculate $\mathbf{H}$ for PageRank
- **Mining Results $\mathbf{Y}$:** $\mathbf{r} = (1 - c)\mathbf{Q}\mathbf{e}$
- **Partial Derivatives $\mathbf{H}$:** $\mathbf{H} = 2c\mathbf{Q}'\mathbf{L}_{\mathbf{S}}\mathbf{r}\mathbf{r}'$
- **Remarks:** $\mathbf{Q} = (\mathbf{I} - c\mathbf{A})^{-1}$
- **Time Complexity**
  - Straightforward: $O(n^3)$
  - Ours: $O(m_1 + m_2 + n)$
    - $m_{\mathbf{A}}$: number of edges in $\mathbf{A}$
    - $m_{\mathbf{S}}$: number of edges in $\mathbf{S}$
    - n: number of nodes

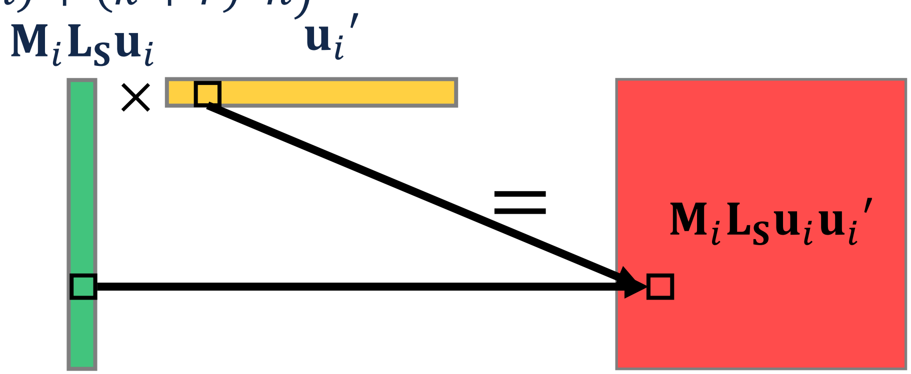


Instantiation #2: Spectral Clustering

- **Goal:** efficiently calculate $\mathbf{H}$ for spectral clustering
- **Mining Results $\mathbf{Y}$: $\mathbf{U}$** $=$ eigenvectors with k smallest eigenvalues
- **Partial Derivatives $\mathbf{H}$: $\mathbf{H}$** $= 2\sum_{i=1}^{k}\left(\mathrm{diag}(\mathbf{M}_i\mathbf{L_S}\mathbf{u}_i\mathbf{u}_i{}')\mathbf{1}_{n\times n} - \mathbf{M}_i\mathbf{L_S}\mathbf{u}_i\mathbf{u}_i{}'\right)$
- **Remarks:** $(\lambda_i, \mathbf{u}_i) = i$-th smallest eigenpair, $\mathbf{M}_i = (\lambda_i\mathbf{I} - \mathbf{L_A})^+$
- **Time Complexity**
  - Straightforward: $O(k^2(m+n) + k^3 n + kn^3)$
  - Ours: $O\big((k+r)(m_1+n) + k(m_2+n) + (k+r)^2 n\big)$
    - k: number of smallest eigenvalues
    - r: number of largest eigenvalues
    - m_1: number of edges in $\mathbf{A}$
    - m_2: number of edges in $\mathbf{S}$
    - n: number of nodes

low-rank

vectorize $\mathrm{diag}(\mathbf{M}_i\mathbf{L_S}\mathbf{u}_i\mathbf{u}_i{}')$ and stack it n times



Instantiation #3: LINE (1st)

- **Goal:** efficiently calculate $\mathbf{H}$ for LINE (1st)
- **Mining Results $\mathbf{Y}$:** $\mathbf{Y}[i,:]\mathbf{Y}[j,:]' = \log \frac{T(\widetilde{\mathbf{A}}[i,j]+\widetilde{\mathbf{A}}[j,i])}{d_i d_j^{3/4} + d_i^{3/4} d_j} - \log b$
  - d_i = outdegree of node i, $T = \sum_{i=1}^{n} d_i^{3/4}$ and b = number of negative samples
- **Partial Derivatives $\mathbf{H}$:** $\mathbf{H} = 2f\left(\widetilde{\mathbf{A}} + \widetilde{\mathbf{A}}'\right) \circ \mathbf{L_S} - 2\text{diag}(\mathbf{B}\mathbf{L_S})\mathbf{1}_{n\times n}$

element-wise in-place calculation

vectorize diag($\mathbf{BL_S}$) and stack it n times

- **Remarks**
  - $f()$ calculates Hadamard inverse, $\circ$ calculates Hadamard product
  - $\mathbf{B} = \frac{3}{4} f\left(\mathbf{d}^{5/4}(\mathbf{d}^{-1/4})' + \mathbf{d}\mathbf{1}_{n\times n}\right) + f\left(\mathbf{d}^{3/4}(\mathbf{d}^{1/4})' + \mathbf{d}\mathbf{1}_{n\times n}\right)$ with $\mathbf{d}^x[i] = d_i^x$

stack $\mathbf{d}$ n times

- **Time Complexity**
  - Straightforward: $O(n^3)$
  - Ours: $O(m_1 + m_2 + n)$
    - m_1: number of edges in $\mathbf{A}$
    - m_2: number of edges in $\mathbf{S}$
    - n: number of nodes

J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In WSDM. 459–467

Debiasing the Mining Model

- **Goal:** bias mitigation during model optimization
- **Intuition:** optimizing a regularized objective such that
  - Task-specific loss function is minimized
  - Bias of mining results as regularization penalty is minimized
- **Optimization Problem**

task-specific loss function

$$\min_{\mathbf{Y}} \; J = l(\mathbf{A}, \mathbf{Y}, \theta) + \alpha \mathrm{Tr}(\mathbf{Y}' \mathbf{L_S} \mathbf{Y})$$

bias measure

- **Solution**
  - **General:** solve by (stochastic) gradient descent $\frac{\partial J}{\partial \mathbf{Y}} = \frac{\partial l(\mathbf{A},\mathbf{Y},\theta)}{\partial \mathbf{Y}} + 2\alpha \mathbf{L_S}\mathbf{Y}$
  - **Task-specific:** solve by specific algorithm designed for the graph mining problem
- **Advantage**
  - Linear time complexity incurred in computing the gradient

Debiasing the Mining Model: Instantiations

- **PageRank**
  - **Objective Function:** $\min_{\mathbf{r}} c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\|\mathbf{r}-\mathbf{e}\|_F^2 + \alpha\mathbf{r}'\mathbf{L_S}\mathbf{r}$
  - **Solution:** $\mathbf{r}^* = c\left(\mathbf{A} - \frac{\alpha}{c}\mathbf{L_S}\right)\mathbf{r}^* + (1-c)\mathbf{e}$
    - PageRank on new transition matrix $\mathbf{A} - \frac{\alpha}{c}\mathbf{L_S}$
    - If $\mathbf{L_S} = \mathbf{I} - \mathbf{S}$, then $\mathbf{r}^* = \left(\frac{c}{1+\alpha}\mathbf{A} + \frac{\alpha}{1+\alpha}\mathbf{S}\right)\mathbf{r}^* + \frac{1-c}{1+\alpha}\mathbf{e}$
- **Spectral Clustering**
  - **Objective Function:** $\min_{\mathbf{U}} \mathrm{Tr}(\mathbf{U}'\mathbf{L_A}\mathbf{U}) + \alpha\mathrm{Tr}(\mathbf{U}'\mathbf{L_S}\mathbf{U}) = \mathrm{Tr}(\mathbf{U}'\mathbf{L_{A+\alpha S}}\mathbf{U})$
  - **Solution:** $\mathbf{U}^*$ = eigenvectors of $\mathbf{L_{A+\alpha S}}$ with k smallest eigenvalues
    - spectral clustering on an augmented graph $\mathbf{A} + \alpha\mathbf{S}$
- **LINE (1st)**
  - **Objective Function:** $\max_{\mathbf{x}_i,\mathbf{x}_j} \log g(\mathbf{x}_j\mathbf{x}_i') + b\mathbb{E}_{j'\in P_n}\left[\log g(-\mathbf{x}_{j'}\mathbf{x}_i')\right] - \alpha\|\mathbf{x}_i - \mathbf{x}_j\|_F^2\mathbf{S}[i,j]$
    $\forall i,j = 1,\ldots,n$
  - **Solution:** stochastic gradient descent

Debiasing the Mining Results

- **Goal:** bias mitigation via a post-processing strategy
- **Intuition:** no access to either the input graph or the graph mining model
- **Optimization Problem**

consistency of mining results, convex

$$\min_{\mathbf{Y}} \quad J = \|\mathbf{Y} - \bar{\mathbf{Y}}\|_F^2 + \alpha \mathrm{Tr}(\mathbf{Y}'\mathbf{L_S}\mathbf{Y})$$

bias measure, convex

  - $\bar{\mathbf{Y}}$ is the vanilla mining results
- **Solution:** $(\mathbf{I} + \alpha\mathbf{S})\mathbf{Y}^* = \bar{\mathbf{Y}}$
  - convex loss function as long as $\alpha \geq 0 \rightarrow$ global optima by $\frac{\partial J}{\partial \mathbf{Y}} = 0$
  - solve by conjugate gradient (or other linear system solvers)
- **Advantages**
  - No knowledge needed on the input graph
  - Model-agnostic

Experimental Settings

- **Questions:**
  - **RQ1.** What is the impact of individual fairness in graph mining performance?
  - **RQ2.** How effective are the debiasing methods?
  - **RQ3.** How efficient are the debiasing methods?
- **Datasets:** 5 publicly available real-world datasets

Name	Nodes	Edges
AstroPh	18,772	198,110
CondMat	23,133	93,497
Facebook	22,470	171,002
Twitter	7,126	35,324
PPI	3,890	76,584

- **Baseline Methods:** vanilla graph mining algorithm
- **Similarity Matrix:** Jaccard index, cosine similarity

Experimental Settings

- Metrics

	Metric		Definition
RQ1	$\text{Diff} = \frac{\lVert \mathbf{Y}^* - \bar{\mathbf{Y}} \rVert_F}{\lVert \bar{\mathbf{Y}} \rVert_F}$		difference between fair and vanilla graph mining results
	PageRank	$KL(\frac{\mathbf{Y}^*}{\lVert \mathbf{Y}^* \rVert_1} \,\vert\vert\, \frac{\bar{\mathbf{Y}}}{\lVert \bar{\mathbf{Y}} \rVert_1})$	KL divergence
		$Prec@50$	precision
		$NDCG@50$	normalized discounted cumulative gain
	spectral clustering	$NMI(\mathcal{C}_{\mathbf{Y}^*}, \mathcal{C}_{\bar{\mathbf{Y}}})$	normalized mutual information
	LINE	$ROC-AUC(\mathbf{Y}^*, \bar{\mathbf{Y}})$	area under ROC curve
		$F1(\mathbf{Y}^*, \bar{\mathbf{Y}})$	F1 score
RQ2	$Reduce = 1 - \frac{\text{Tr}\left((\mathbf{Y}^*)' \, \mathbf{L}_\mathbf{S} \mathbf{Y}^*\right)}{\text{Tr}(\bar{\mathbf{Y}}' \mathbf{L}_\mathbf{S} \bar{\mathbf{Y}})}$		degree of reduce in individual bias
RQ3	Running time in seconds		running time

Experimental Results

Table 1: Effectiveness results for PageRank. Lower is better in gray columns. Higher is better in the others.

Datasets	Jaccard Index						Cosine Similarity					
Debiasing the Input Graph												
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.109	5.37×10^{-4}	1.000	1.000	24.7%	564.9	0.299	5.41×10^{-3}	0.860	0.899	62.9%	649.3
PPI	0.185	1.90×10^{-3}	0.920	0.944	43.4%	584.4	0.328	8.07×10^{-3}	0.780	0.838	68.7%	636.8
Debiasing the Mining Model												
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.182	4.97×10^{-3}	0.940	0.958	62.0%	16.18	0.315	1.05×10^{-2}	0.940	0.957	73.9%	12.73
PPI	0.211	4.78×10^{-3}	0.920	0.942	50.8%	10.76	0.280	9.56×10^{-3}	0.900	0.928	67.5%	10.50
Debiasing the Mining Results												
	Diff	KL	Prec@50	NDCG@50	Reduce	Time	Diff	KL	Prec@50	NDCG@50	Reduce	Time
Twitch	0.035	9.75×10^{-4}	0.980	0.986	33.9%	0.033	0.101	5.84×10^{-3}	0.940	0.958	44.6%	0.024
PPI	0.045	1.22×10^{-3}	0.940	0.958	27.0%	0.020	0.112	6.97×10^{-3}	0.940	0.958	45.0%	0.019

- **Obs.:** effective in mitigating bias while preserving the performance of the vanilla algorithm with relatively small changes to the original mining results
  - Similar observations for spectral clustering and LINE (1st)

Roadmap

- ☑ Motivations
- InFoRM: Individual Fairness on Graph Mining
  - ☑ InFoRM Introduction
  - ☑ InFoRM Measures
  - ☑ InFoRM Algorithms
  - ➡ InFoRM Cost
- Some Other Work
- Future Directions

Problem Definition: InFoRM Cost

- **Question:** how to quantitatively characterize the cost of individual fairness?
- **Input**
  - Vanilla mining results $\bar{\mathbf{Y}}$
  - Debiased mining results $\mathbf{Y}^*$
    - Learned by the previous problem (InFoRM Algorithms)
- **Output:** an upper bound of $\|\bar{\mathbf{Y}} - \mathbf{Y}^*\|_F$
- **Debiasing Methods**
  - Debiasing the input graph
  - Debiasing the mining model
  - Debiasing the mining results

depend on specific graph topology/mining model

main focus of this paper

Cost of Debiasing the Mining Results

- **Given**
  - A graph with n nodes and adjacency matrix $\mathbf{A}$
  - A node-node similarity matrix $\mathbf{S}$
  - Vanilla mining results $\bar{\mathbf{Y}}$
  - Debiased mining results $\mathbf{Y}^* = (\mathbf{I} + \alpha\mathbf{S})^{-1}\bar{\mathbf{Y}}$
- If $\|\mathbf{S} - \mathbf{A}\|_F = \Delta$, we have

$$\|\bar{\mathbf{Y}} - \mathbf{Y}^*\|_F \leq 2\alpha\sqrt{n}\left(\Delta + \sqrt{rank(\mathbf{A})}\sigma_{\max}(\mathbf{A})\right)\|\bar{\mathbf{Y}}\|_F$$

- **Observation:** the cost of debiasing the mining results depends on
  - The number of nodes n (i.e. size of the input graph)
  - The difference Δ between $\mathbf{A}$ and $\mathbf{S}$
  - The rank of $\mathbf{A}$ → could be small due to low-rank structures in real-world graphs
  - The largest singular value of $\mathbf{A}$ → could be small if $\mathbf{A}$ is normalized

Cost of Debiasing the Mining Model: Case Study on PageRank

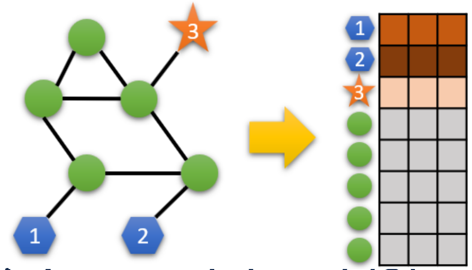
- **Given**
  - A graph with n nodes and symmetrically normalized adjacency matrix $\boldsymbol{A}$
  - A symmetrically normalized node-node similarity matrix $\mathbf{S}$
  - Vanilla PageRank vector $\bar{\mathbf{r}}$
  - Debiased PageRank vector $\mathbf{r}^* = (\mathbf{I} + \alpha\mathbf{S})^{-1}\bar{\mathbf{Y}}$
- If $\|\boldsymbol{S} - \mathbf{A}\|_F = \Delta$, we have

$$\|\bar{\mathbf{r}} - \mathbf{r}^*\|_F \leq \frac{2\alpha n}{1-c}\left(\Delta + \sqrt{rank(\mathbf{A})}\sigma_{\max}(\mathbf{A})\right)$$

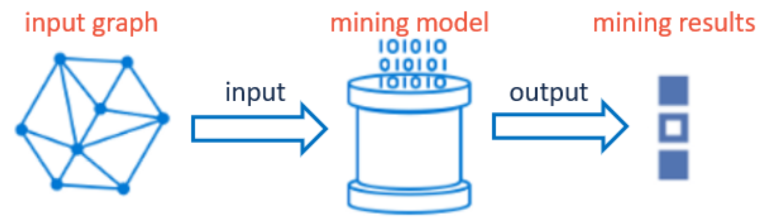
- **Observation:** the cost of debiasing PageRank depends on
  - The number of nodes n (i.e. size of the input graph)
  - The difference Δ between $\mathbf{A}$ and $\mathbf{S}$
  - The rank of $\mathbf{A}$ → could be small due to low-rank structures in real-world graphs
  - The largest singular value of $\boldsymbol{A}$ → upper bounded by 1

InFoRM Summary

- **Problem:** InFoRM (individual fairness on graph mining)
  - **fundamental questions:** measures, algorithms, cost



- **Solutions:**
  - **Measures:** $\mathrm{Bias}(\mathbf{Y}, \mathbf{S}) = \mathrm{Tr}(\mathbf{Y}'\mathbf{S}\mathbf{Y})$
  - **Algorithms:** debiasing (1) the input graph, (2) the mining model and (3) the mining results
  - **Cost:** the upper bound of $\|\bar{\mathbf{Y}} - \mathbf{Y}^*\|_F$
    - Upper bound on debiasing the mining results
    - Case study on debiasing PageRank algorithm



- **Results:** effective in mitigating individual bias in the graph mining results while maintaining the performance of vanilla algorithm

Debiasing the Input Graph								
Datasets	Jaccard Index				Cosine Similarity			
	Diff	NMI	Reduce	Time	Diff	NMI	Reduce	Time
Twitch	0.031	1.000	5.44%	1698	0.107	1.000	24.5%	1714
PPI	1.035	0.914	19.5%	829.3	0.933	0.849	24.1%	985.1

J. Kang, J. He, R. Maciejewski and H. Tong: InFoRM: Individual Fairness on Graph Mining. KDD 2020

Roadmap

- ☑ Motivations
- ☑ InFoRM: Individual Fairness on Graph Mining
- ➡ Some Other Work
  - Network Derivative Mining
  - Adversarial Multi-Network Mining
  - Discerning Edge Influence for Network Embedding
  - View Adversarial Network Embedding
  - Explainable Networked Prediction
  - Data Debugging in Collaborative Filtering
- Future Directions

N2N: Network Derivative Mining

Problem Dfn.

- **Given**: (1) a network *A*, (2) a mining model *L(A,Y, ϑ)*, & (3) a scalar function *l()*;
- **Find**: a derivative network *B:*

$$= \frac{\mathrm{d}l(\mathcal{Y}^*)}{\mathrm{d}\mathbf{A}} \quad \text{subject to } \mathcal{Y}^* \in \operatorname*{argmin}_{\mathcal{Y}} \mathcal{L}(\mathbf{A}, \mathcal{Y}, \theta)$$

- From What/Who to How/Why
- Bi-level opt. & Scalability

Potentials & Applications

- Explainable net mining
- Adversarial net mining
- Sensitivity analysis
- Active network mining
- Counterfactual analysis

Current Scope

Learning Tasks	Loss Function $\mathcal{L}$	Learning Results $\mathcal{Y}$	Additional Parameters θ	Scalar Function $l(\cdot)$
PageRank Ranking [1]	$\min_{\mathbf{r}} \; c\mathbf{r}'(\mathbf{I}-\mathbf{A})\mathbf{r} + (1-c)\|\|\mathbf{r}-\mathbf{e}\|\|_F^2$	PageRank vector $\mathbf{r}$	damping factor c	$l(\mathcal{Y}^*) = \|\|\mathbf{r}\|\|_F^2$
HITS Ranking [10]	$\min_{\mathbf{u},\mathbf{v}} \; \|\|\mathbf{A} - \mathbf{u}\mathbf{v}'\|\|_F^2$	hub vector $\mathbf{u}$ authority vector $\mathbf{v}$	none	$l(\mathcal{Y}^*) = \lambda_1 - \lambda_2$
Spectral Clustering [2]	$\min_{\mathbf{U}} \; \mathrm{Tr}(\mathbf{U}'\mathbf{L}\mathbf{U})$ subject to $\mathbf{U}'\mathbf{U} = \mathbf{I}$	matrix $\mathbf{U}$	number of clusters r	$l(\mathcal{Y}^*) = \sum_{i=1}^{r} \lambda_i$
Matrix Completion [11]	$\min_{\mathbf{U},\mathbf{V}} \; \|\|\mathrm{proj}_\Omega(\mathbf{A} - \mathbf{U}\mathbf{V}')\|\|_F^2 + \lambda_u\|\|\mathbf{U}\|\|_F^2 + \lambda_v\|\|\mathbf{V}\|\|_F^2$	user matrix $\mathbf{U}$ item matrix $\mathbf{V}$	latent dimensions r regularization parameters λ_u, λ_v	$l(\mathcal{Y}^*) = \|\|\mathbf{U}\mathbf{V}'\|\|_F^2$
Node Embedding [12]	$\min_{\mathbf{W}} \; -\log Pr(v_{i-w}, ..., v_{i-1}, v_{i+1}, ..., v_{i+w} \| \mathbf{W}(\mathbf{i},:))$	embedding matrix $\mathbf{W}$	embedding length d random walk length w	$l(\mathcal{Y}^*) = \|\|\mathbf{W}\|\|_F^2$

Main Results: a linear algorithm (in both time & space) for each task!

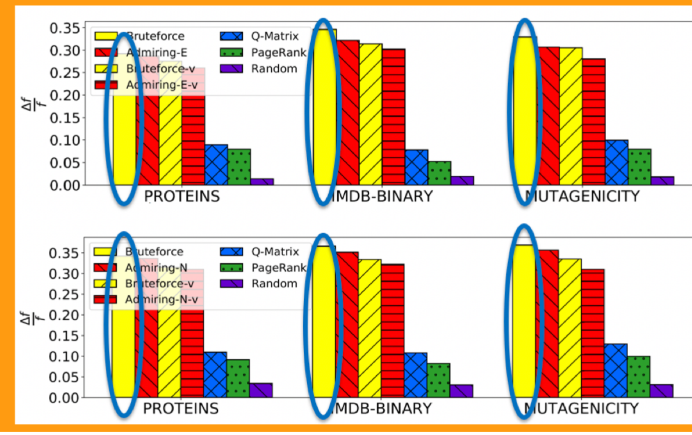
- M. Wang, J. Kang, N. Cao, Y. Xia, W. Fan, H. Tong: Graph Ranking Auditing: Problem Definitions and Fast Solutions. TKDE 2020
- J. Kang and H. Tong: N2N: Network Derivative Mining. CIKM 2019
- Y. Wang, Y. Yao, H. Tong, F. Xu and J. Lu :Discerning Edge Influence for Network Embedding. CIKM 2019
- J. Kang, M. Wang, N. Cao, Y. Xia, W. Fan, and H. Tong: AURORA: Auditing PageRank on Large Graphs. BigData 2018

Admiring: Adversarial Multi-Network Mining

Problem Dfn.

- **Given**: (1) two input attributed networks $\mathcal{G}_1$ and $\mathcal{G}_2$; and (2) a multi-network mining task;
- **Find**: a set of most influential elements:
  - Identify vulnerability
  - Improve robustness
  - Render explanability

Results

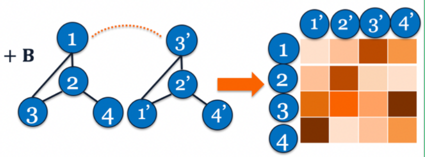


Proposed Method

- Generalized Sylvester Equation for multi-network mining

Given two attributed networks, $\mathcal{G}_1 = \{\mathbf{A}_1, \mathbf{N}_1\}$ and $\mathcal{G}_2 = \{\mathbf{A}_2, \mathbf{N}_2\}$,

$$\mathbf{X} = \sum_{l=1}^{d} c\mathbf{M}_l\mathbf{X}\mathbf{T}_l' + \mathbf{B}$$



$\mathbf{M}_l = \mathbf{N}_2^l\mathbf{A}_2$, $\mathbf{T}_l = \mathbf{N}_1^l\mathbf{A}_1$. By the Kronecker product property, $\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$

$$\mathbf{x} = c\mathbf{N}_\times\mathbf{A}_\times\mathbf{x} + \mathbf{b} \qquad \mathbf{A}_\times = \mathbf{A}_1 \otimes \mathbf{A}_2$$

The closed-form solution is given by $\mathbf{x} = (\mathbf{I} - c\mathbf{N}_\times\mathbf{A}_\times)^{-1}\mathbf{b}$

Multi-network Mining Tasks	Function $f(\cdot)$
Random walk graph kernel	$f(\mathbf{X}) = \mathbf{q}'_\times\mathrm{vec}(\mathbf{X})$
Network Alignment	$f(\mathbf{X}) = \mathbf{X}$ or $f(\mathbf{X}) = \mathrm{vec}(\mathbf{X})$
Cross-network node similarity	$f(\mathbf{X}) = \mathbf{X}(s,t)$
Subgraph Matching	$f(\mathbf{X}) = \mathrm{argmin}_{\mathbf{M}}\, g(\mathbf{M}, \mathbf{X})$

$$\max_{\mathcal{P}} \Delta f = \left(f(\mathbf{X}) - f(\mathbf{X}_{\mathcal{P}})\right)^2 \quad \text{s.t. } |\mathcal{P}| = k$$

new solution matrix after we perturb the network elements in set $\mathcal{P}$

Network Element Influence for a given mining task $f(\mathbf{X})$

- **Edge Influence**: the derivative of $f(\mathbf{X})$ w.r.t. this edge,

$$\mathcal{I}(\mathbf{A}_1(i,j)) = \frac{\partial f(\mathbf{X})}{\partial \mathbf{A}_1(i,j)}$$

rate of change

- **Node Influence**: the summation of influences of the incident edges,

$$\mathcal{I}(\mathbf{N}_1(i)) = \sum_{j|\mathbf{A}_1(i,j)=1} \mathcal{I}(\mathbf{A}_1(i,j))$$

- **Node Attribute Influence**: the derivative of $f(\mathbf{X})$ w.r.t. this attribute,

$$\mathcal{I}(\mathbf{N}_1^l(i,i)) = \frac{\partial f(\mathbf{X})}{\partial \mathbf{N}_1^l(i,i)}$$

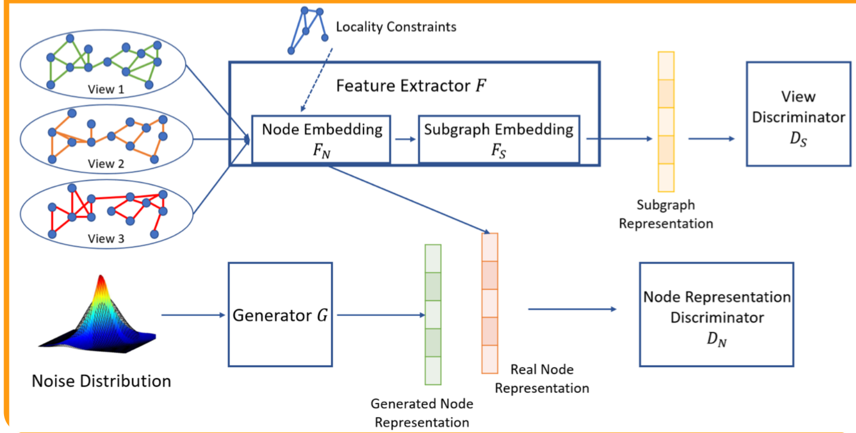
- Q. Zhou, L. Li, N. Cao, L. Ying and H. Tong: Admiring: Adversarial Multi-Network Mining. ICDM 2019
- B. Du and H. Tong FASTEN: Fast Sylvester Equation Solver for Graph Mining. KDD2018

View-Adversarial Network Embedding

Problem Dfn.

- **Given**: a multi-view network $G = (V, E_1, E_2, \dots, E_k)$;
- **Find**: the robust and consistent node representations across k different views $\{x_v\}_{v \in V} \in \mathcal{R}^d, d \ll |V|$

Overview Framework



Key Idea

- First adversarial game (F, D_S): enhances the comprehensiveness of the node representation
- Second adversarial game (G, F_N, D_N): improves the robustness of the node representation

alternatively

Results

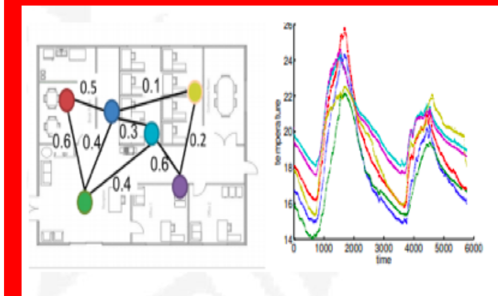
Methods	View	Accuracy (%) Node Classification	Link Prediction
DeepWalk	Follow	70.95±2.56	50.30
	Mention	69.64±5.46	50.27
	Retweet	73.78±5.18	52.24
	Combined	66.47±2.85	50.03
node2vec	Follow	79.52±4.42	65.45
	Mention	79.64±3.47	62.94
	Retweet	81.83±4.31	52.18
	Combined	80.59±2.75	60.61
GraphGAN	Follow	76.15±1.92	53.97
	Mention	71.95±2.74	51.88
	Retweet	39.20±2.42	50.21
	Combined	72.44±1.69	55.41
MNE	Follow	85.66±2.87	56.37
	Mention	84.70±3.45	74.66
	Retweet	85.06±3.42	76.15
MVE	All	83.76±4.90	68.85
VANE-RW	All	82.89±2.38	69.40
VANE-BRW	All	**90.60±2.57**	**85.36**

Table 2: Performance on Twitter-Rugby Dataset

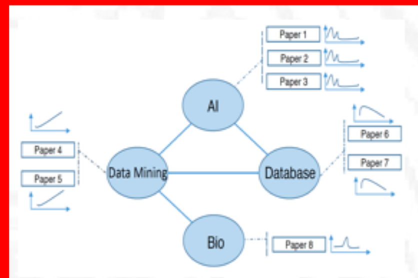
- D. Fu*, Z. Xu*, B. Li, H. Tong, and J. He: A View-Adversarial Framework for Multi-View Network Embedding. CIKM 2020

Explainable Networked Prediction

Goal: explain networked prediction



Network of Time Series



Scientific Impact Forecasting

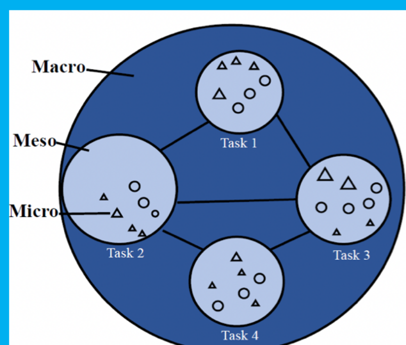
Solution

- **Multi-Aspect, Multi-Level Explanation**

Level / Aspect	Macro/System	Meso/Task	Micro/Test example
Training example x^t	Globally influential training sample ($I_G(x^t)$)	Task specific influential training sample ($I_s(x^t)$)	Test specific influential training sample ($I_{x^s_{test}}(x^t)$)
Learning task f_t	Globally influential task ($I_G(f_t)$)	Task specific influential task($I_s(f_t)$)	Test specific influential task($I_{x^s_{test}}(f_t)$)
Task Network A	Globally influential task connections ($I_G(A_{ij})$)	Task specific influential task connections ($I_s(A_{ij})$)	Test specific influential task connections ($I_{x^s_{test}}(A_{ij})$)

Key Challenges

- Multi-level: Macro, meso, micro
- Efficiency: Measure the influence w/o retraining

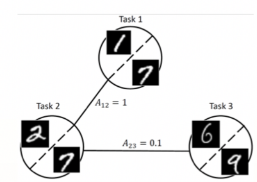


Networked Prediction Formulation:

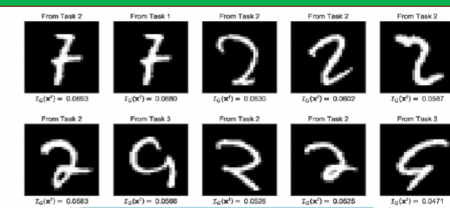
Task network

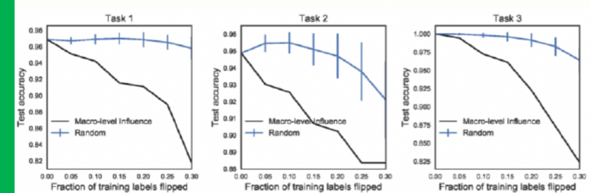
$$\min_{\theta_1,\dots,\theta_T} \sum_{t=1}^{T} \frac{1}{n_t} \sum_{i=1}^{n_t} L(f_t(x_i^t,\theta_t), y_i^t) + \lambda \sum_{i=1}^{T} \sum_{j=1}^{T} A_{ij} \|\theta_i - \theta_j\|^2$$

Results

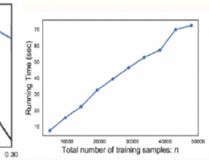


Setup on MNIST


Top-*10* globally influential samples



Fraction of training labels flipped vs. test accuracy



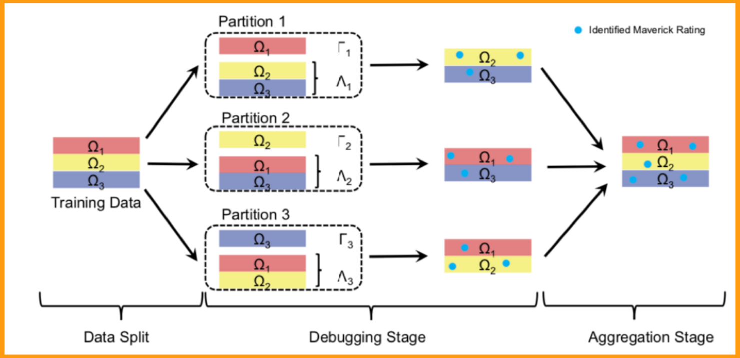
Scalability

- L. Li, H. Tong, H. Liu: Towards Explainable Networked Prediction. CIKM 2018
- 1. J. Kang, S. Freitas, H. Yu, Y. Xia, N. Cao, H. Tong: X-Rank: Explainable Ranking in Complex Multi-Layered Networks. CIKM 2018
- Q. Zhou, L. Li, N. Cao, N. and H. Tong: Extra: Explaining Team Recommendation in Networks. Recsys 2018

Data Debugging in Collaborative Filtering

- Research Questions:
- Q1. are all ratings helpful in collaborativ filtering, and if not,
- Q2. how can we mitigate harmful (i.e., overly personalized) ones to improve th overall recommendation accuracy?

- Solution



$$\Phi = \arg\min_{\Phi \subset \Lambda} L_{\Gamma}(\Theta(\Lambda - \Phi)), \quad s.t. \quad |\Phi| \leq K$$

- Results

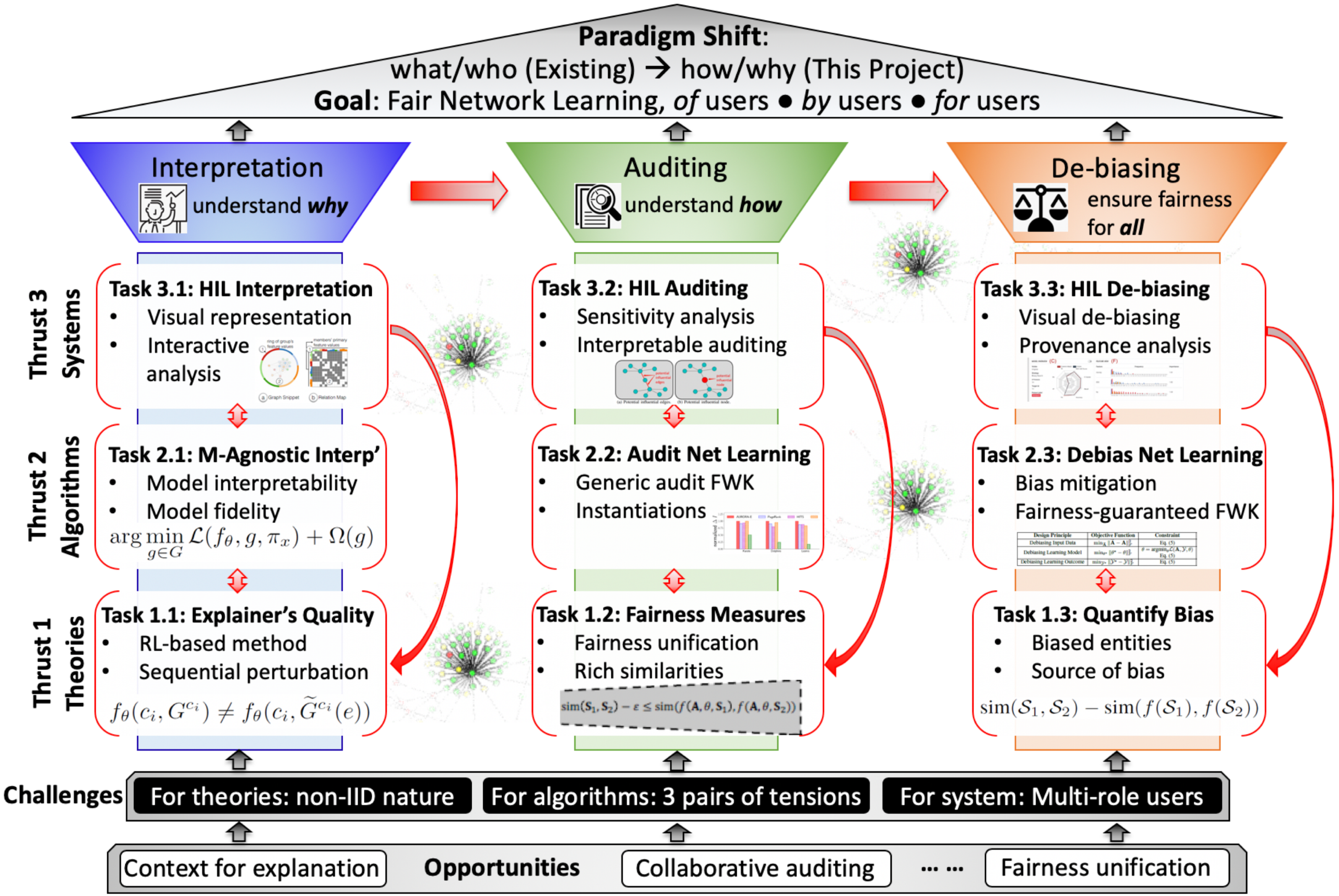
Method		eMF	NrMF	NoiseCorrection	CFDEBUG-full	CFDEBUG
modify ratings	0.1%	0.9134 ○	0.9137 ○	0.9126 ○	0.9052 ●	0.9071 ●
	0.2%	0.9140 ○	0.9137 ○	0.9142 ○	0.9011 ●	0.9037 ●
	0.5%	0.9148 *	0.9174 *	0.9146 *	0.8943 ●	0.8985 ●
	1%	0.9171 ○	0.9180 *	0.9176 ○	0.8880 ●	0.8926 ●
	2%	0.9198 ○	0.9218 *	0.9226 ○	0.8812 ●	0.8876 ●
	5%	0.9231 *	0.9334 *	0.9251 *	0.8735 ●	0.8810 ●
	10%	0.9246 *	0.9495 *	0.9310 *	0.8695 ●	0.8785 ●

- Long Chen, Yuan Yao, Feng Xu, Miao Xu, Hanghang Tong: Trading Personalization for Accuracy: Data Debugging in Collaborative Filtering. NeurIPS 2020

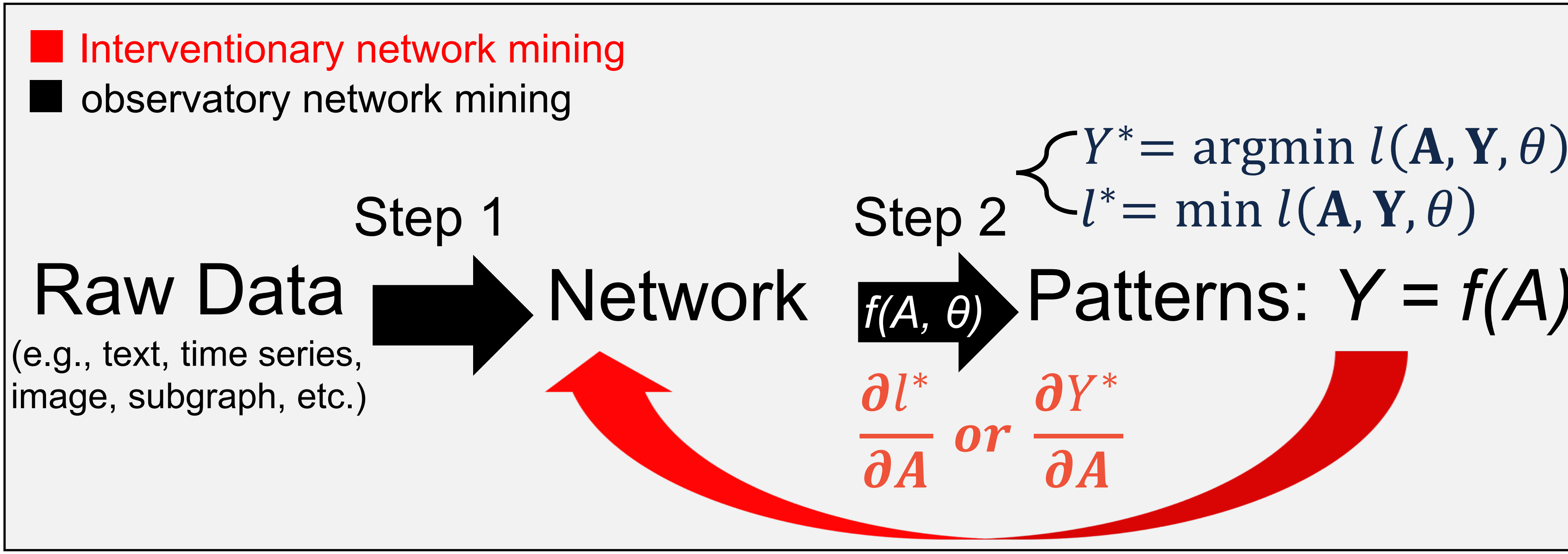
Roadmap

- ☑ Motivations
- ☑ InFoRM: Individual Fairness on Graph Mining
- ☑ Some Other Work
- ➡ Future Directions

NetFair: Fair Network Learning



Interventionary Network Mining



- **Implications**
  - InFoRM Algorithm
    - **(debiasing *A, this talk*)**
  - Explainable mining
  - Adversarial mining
  - Stability analysis
  - Learning w/ side info.
  - Active data collection
  - Debug data (optimal network)

- Key Challenge: How to compute a huge gradient matrix?
  - nested opt., implicit computation, scalability, compact representation