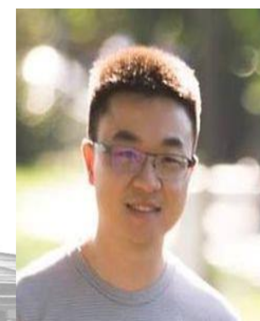
NetTrans: Neural Cross-Network Transformation

Si Zhang
(UIUC)

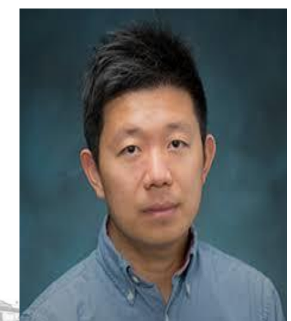
Hanghang Tong
(UIUC)

Yinglong Xia
(Facebook)

Liang Xiong
(Facebook)

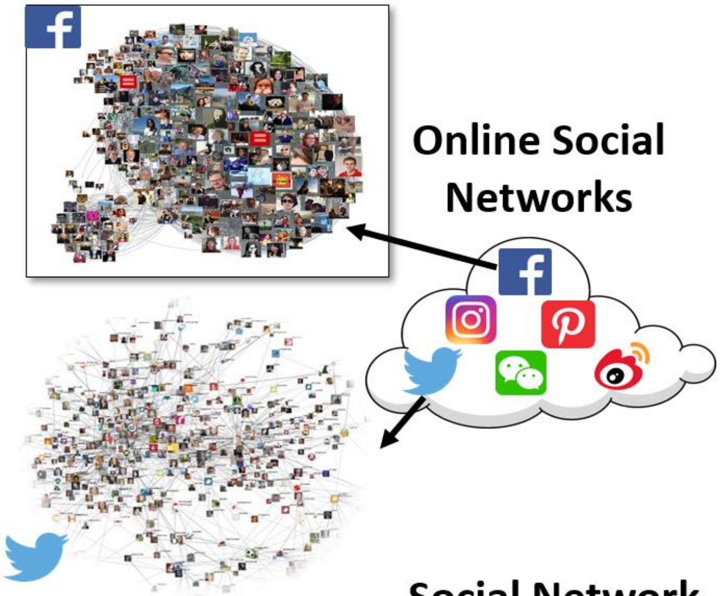


Jiejun Xu
(HRL Inc.)

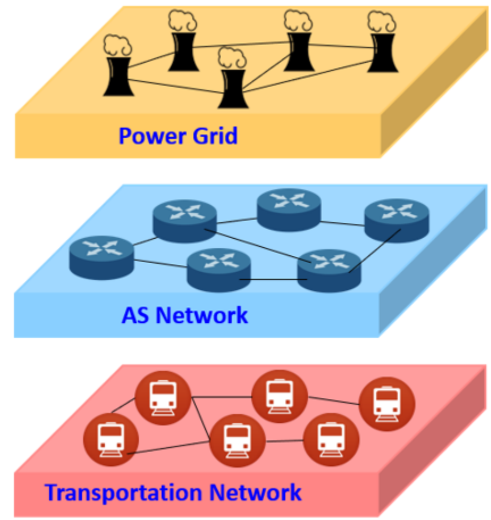


ILLINOIS

Networks Are Often Multi-Sourced



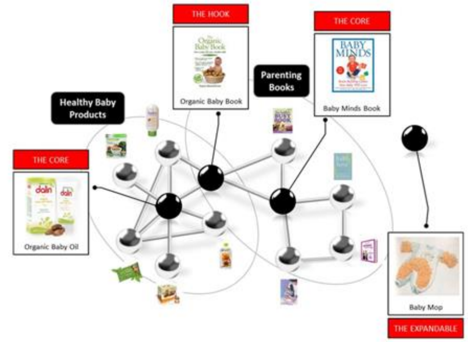
Online Social Networks



Infrastructure Networks

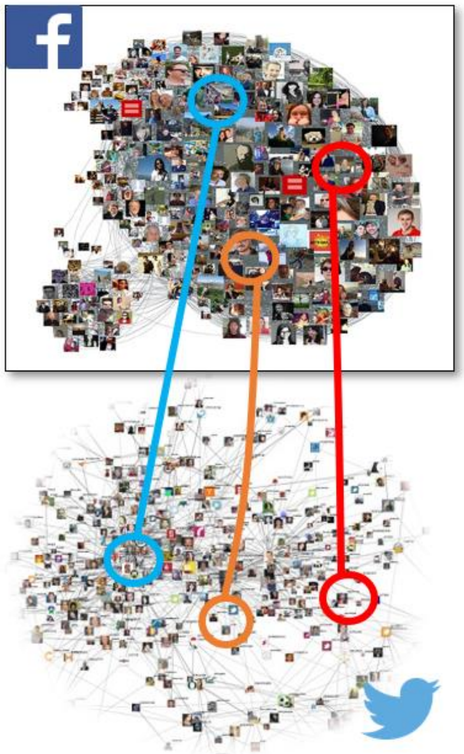
Social Network

Product Network

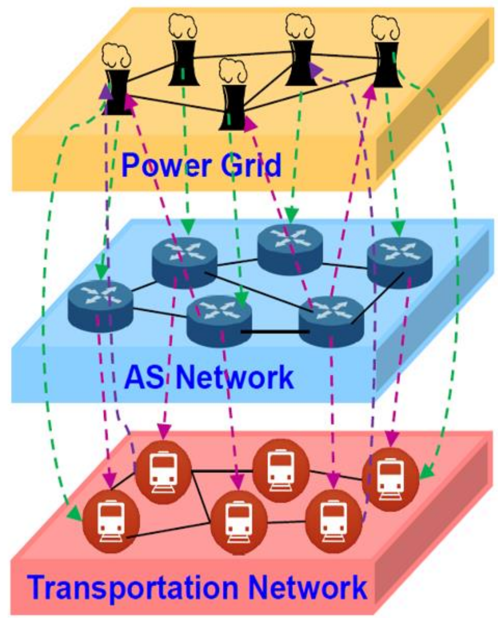


Cross-Network Node Associations

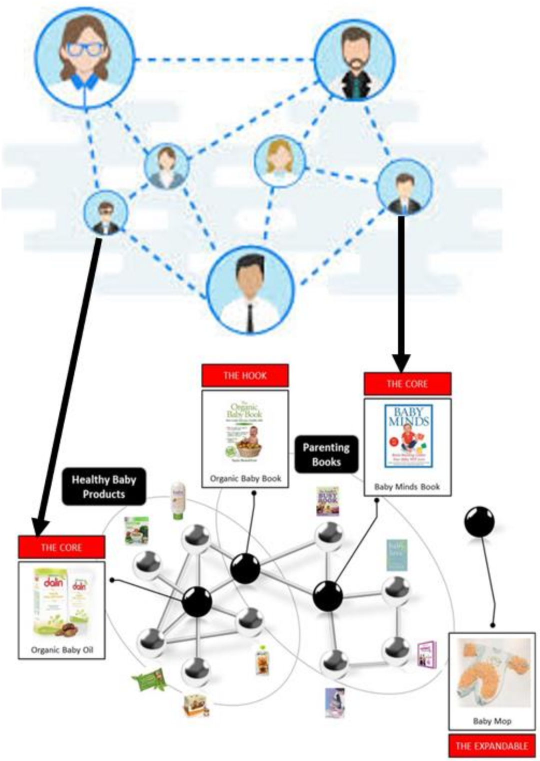
- To find node associations across different networks



Network alignment



Cross-layer dependency



Recommendation

Traditional Methods

- For network alignment – graph matching based [1]

$$\min \ \|\boldsymbol{B}_0 - \boldsymbol{P}\boldsymbol{A}_0\boldsymbol{P}^T\|_F^2$$

$$\min \ \left\|\text{vec}(\boldsymbol{B}_0) - \boxed{\widetilde{\boldsymbol{P}}\text{vec}(\boldsymbol{A}_0)}\right\|_2^2$$

Linear transformation

- For recommendation and cross-layer dependency [2,3]

$$\min \ \left\|\boldsymbol{R} - \boldsymbol{U}_1^T\boldsymbol{U}_2\right\|_F^2 + \alpha \sum_i \boxed{\text{Tr}\left(\boldsymbol{U}_i^T(\boldsymbol{D}_i - \boldsymbol{A}_i)\boldsymbol{U}_i\right)}$$

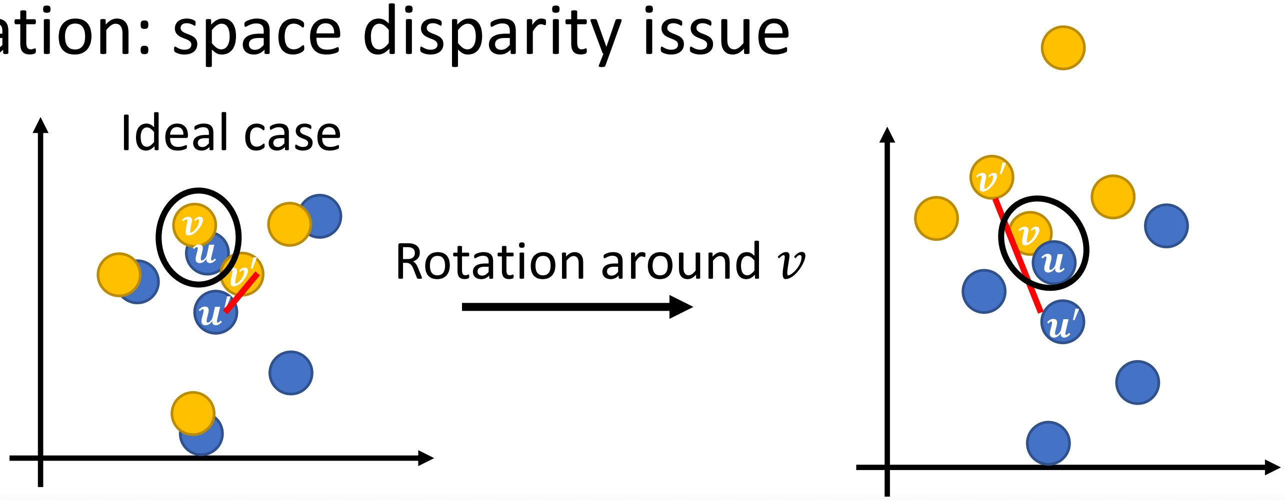
Network-based regularization

- Limitations: linear and/or consistency assumptions

[1] Umeyama, Shinji. "An eigendecomposition approach to weighted graph matching problems." *IEEE transactions on pattern analysis and machine intelligence* 10.5 (1988): 695-703.
[2] Yao, Yuan, et al. "Dual-regularized one-class collaborative filtering." *CIKM* 2014.
[3] Chen, Chen, et al. "FASCINATE: fast cross-layer dependency inference on multi-layered networks." *KDD* 2016.

Embedding Based Methods

- Existing methods
  - Network alignment [1,2]
    - Aligned nodes are closed in the embedding space
  - Cross-layer dependency [3]
    - Embeddings of different networks interact linearly
- Limitation: space disparity issue

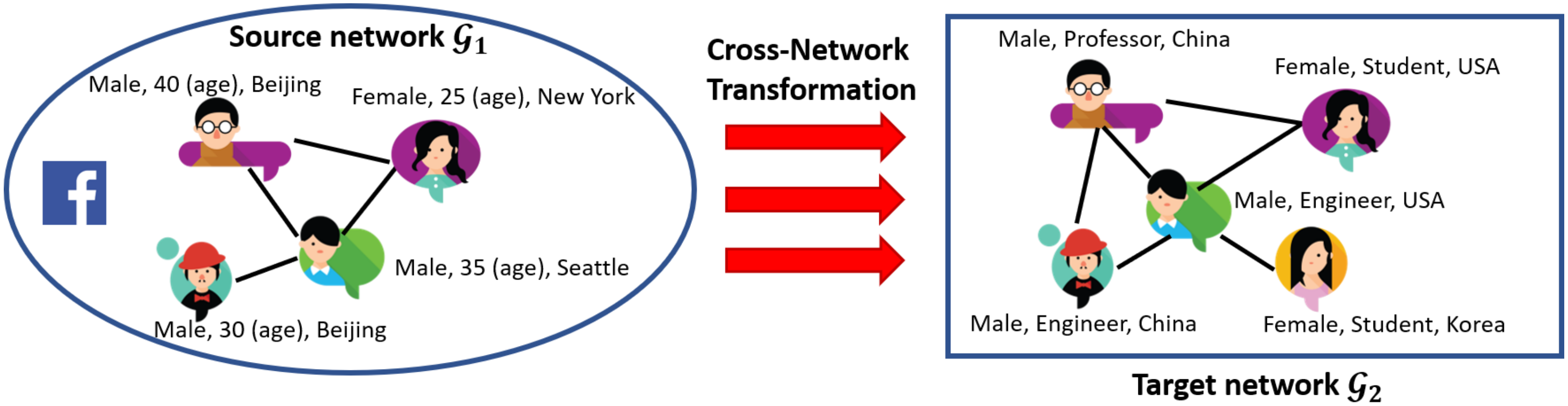


[1] Liu, Li, et al. "Aligning Users across Social Networks Using Network Embedding." *Ijcai*. 2016.
[2] Chu, Xiaokai, et al. "Cross-network embedding for multi-network alignment." *The World Wide Web Conference*. 2019.
[3] Li, Jundong, et al. "Multi-layered network embedding." *SDM*, 2018.

Cross-Net Node Assoc.: A New Angle

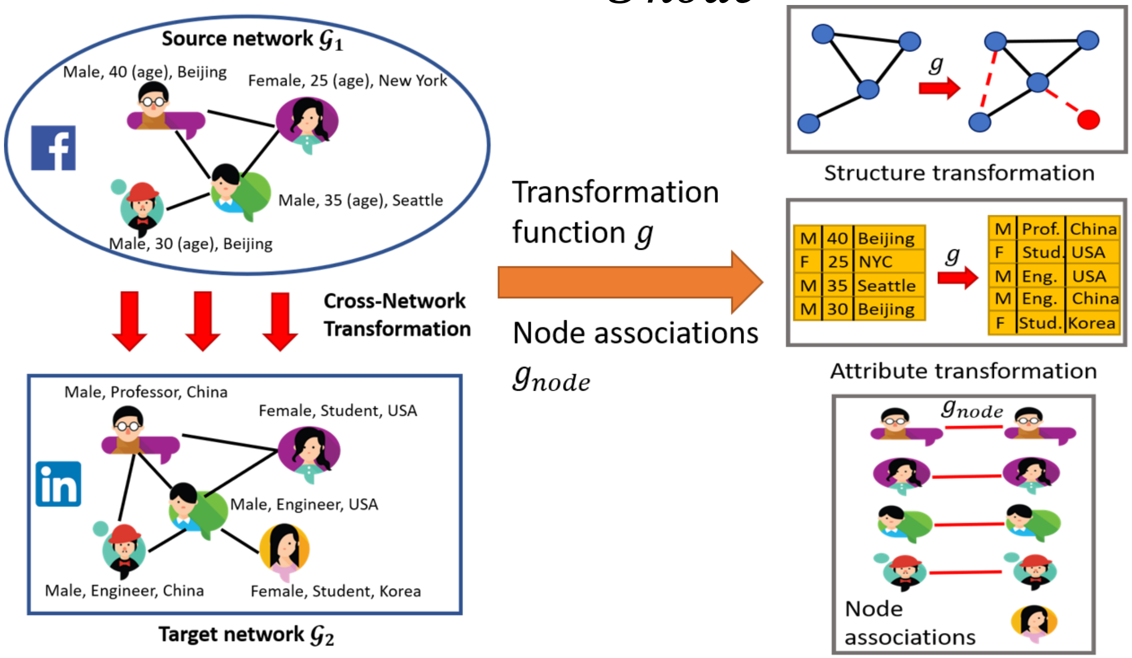
- A generic question:

Given two different networks, how can we transform one network to another?



Prob. Def.: Cross-Net Transformation

- **Given:** (1) source and target networks $\mathcal{G}_1 = \{\mathcal{V}_1, \boldsymbol{A}_0, \boldsymbol{X}_0\}$, $\mathcal{G}_2 = \{\mathcal{V}_2, \boldsymbol{B}_0, \boldsymbol{Y}_0\}$; (2) observed cross-network node associations $\boldsymbol{L}$
- **Output:** (1) cross-network transformation function g, s.t. $g(\mathcal{G}_1) \approx \mathcal{G}_2$; (2) node association function g_{node}



An Illustrative Example

- Graph matching based network alignment

$$\min \ \|\boldsymbol{B}_0 - \boldsymbol{P}\boldsymbol{A}_0\boldsymbol{P}^T\|_F^2 + \|\boldsymbol{Y}_0 - \boldsymbol{P}\boldsymbol{X}_0\|_F^2$$

$$\left\|\mathrm{vec}(\boldsymbol{B}_0) - \widetilde{\boldsymbol{P}}\mathrm{vec}(\boldsymbol{A}_0)\right\|_2^2 \text{ where } \widetilde{\boldsymbol{P}} = \boldsymbol{P} \otimes \boldsymbol{P}$$

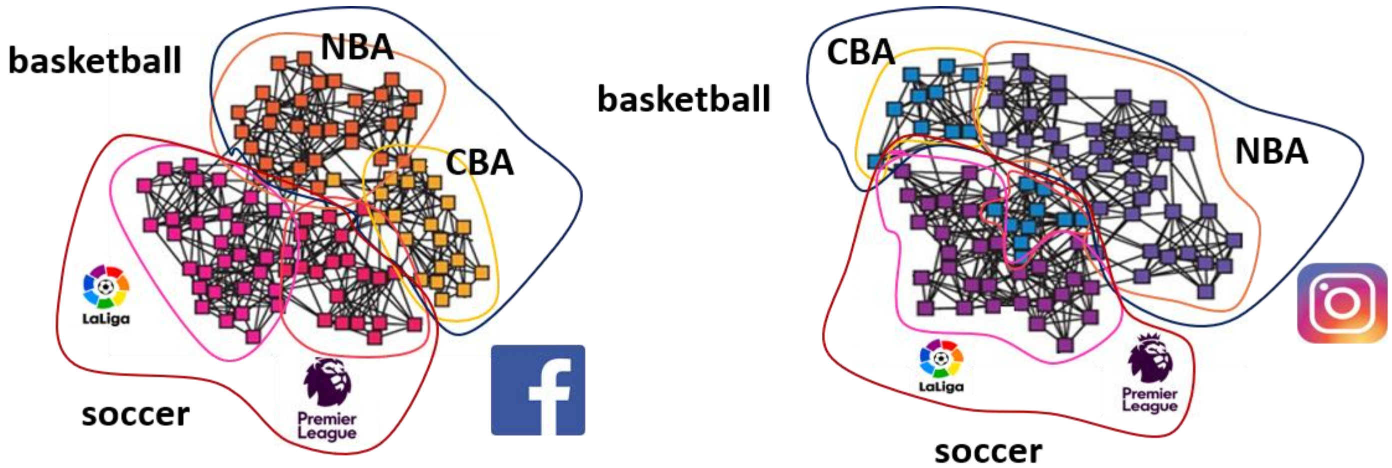
  - Objective: $\mathrm{vec}(\boldsymbol{B}_0) \approx \widetilde{\boldsymbol{P}}\mathrm{vec}(\boldsymbol{A}_0)$ and $\boldsymbol{Y}_0 \approx \boldsymbol{P}\boldsymbol{X}_0$
  - Transformation function: $g(\mathrm{vec}(\boldsymbol{A}_0), \boldsymbol{X}_0) = \left(\widetilde{\boldsymbol{P}}\mathrm{vec}(A_0), \boldsymbol{P}\boldsymbol{X}_0\right)$
  - Node association function: $g_{node}(u, v) = \boldsymbol{P}(v, u)$

Outline

- Motivations ✓
- NetTrans Model
  - Encoder: TransPool
  - Decoder: TransUnPool
- Experimental Results
- Conclusions

NetTrans – Model Overview

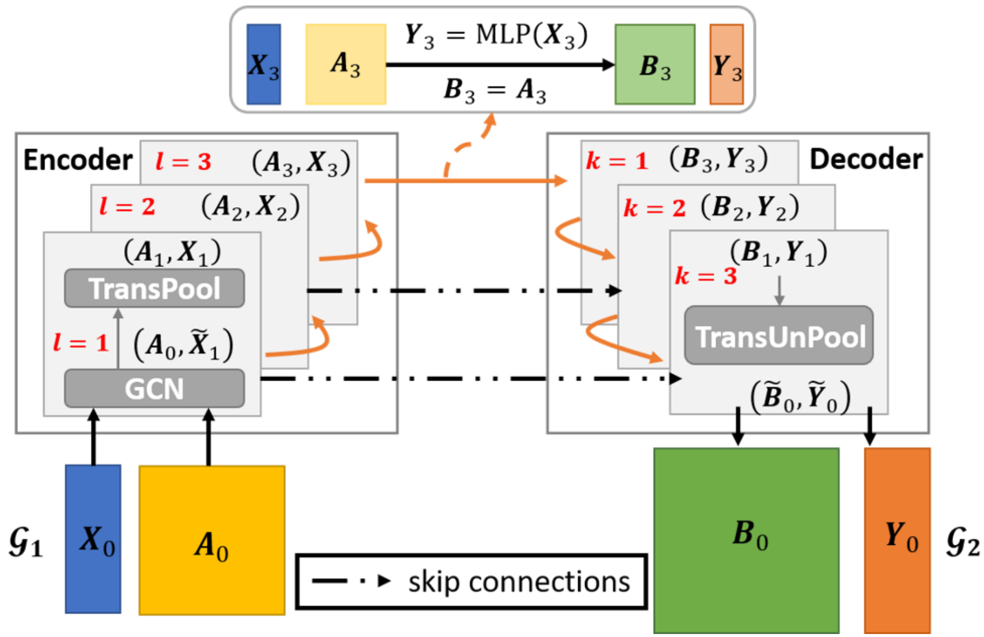
- Key idea #1: multi-resolution characteristic



  - Simplify network transformation at coarse resolutions
  - Assume same latent meanings, e.g., NBA (FB) vs. NBA (Ins)
  - Auxiliary associations info, e.g., NBA -> users who like NBA

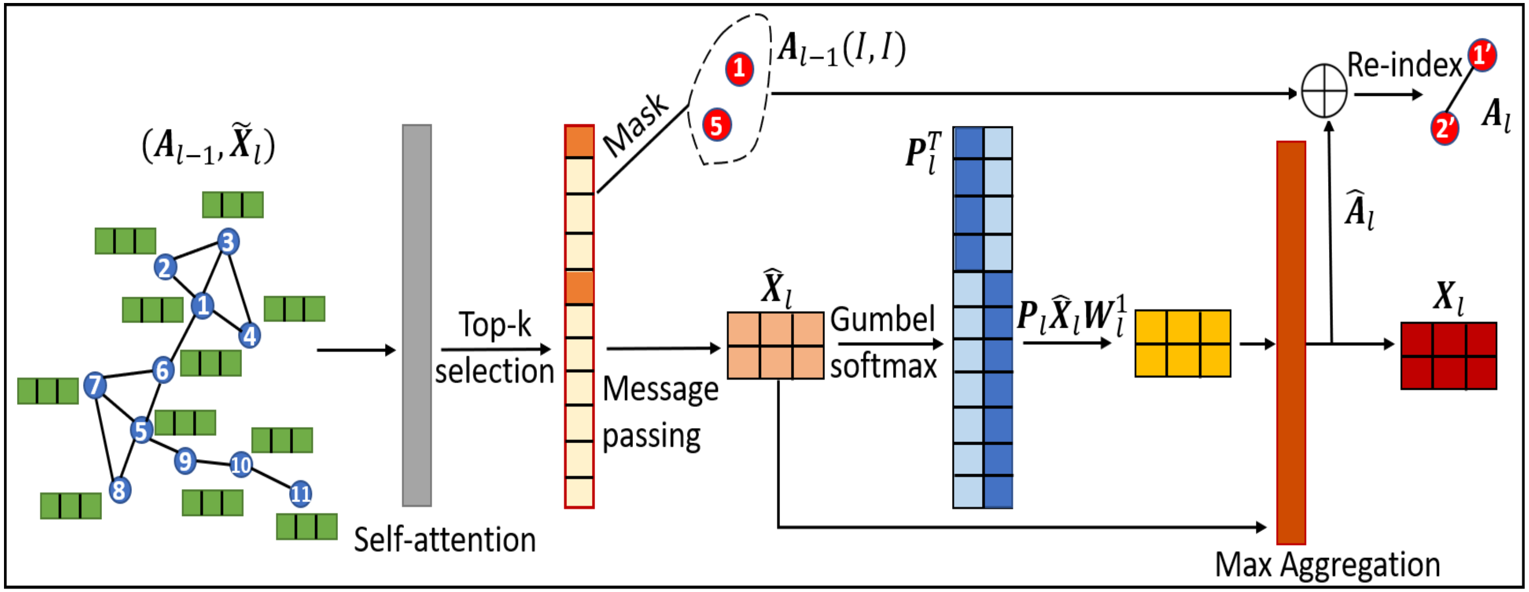
NetTrans – Model Overview (con't)

- Key idea #2: encoder-decoder architecture
  - Encoder: to coarsen source network at different resolutions
  - Decoder: to reconstruct target network at different resolutions



NetTrans – Encoder

- Goals:
  - To learn node representations and structure at different resolutions
  - To learn node-to-supernode assignments

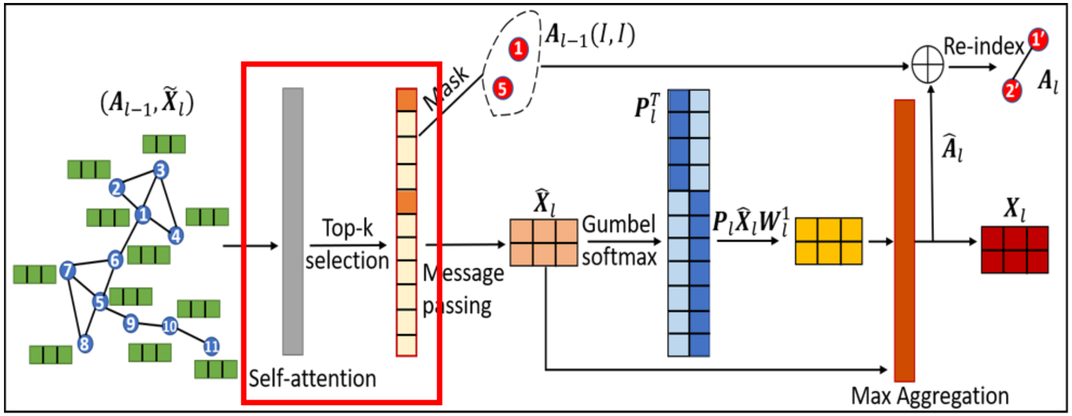


NetTrans Encoder: Part #1

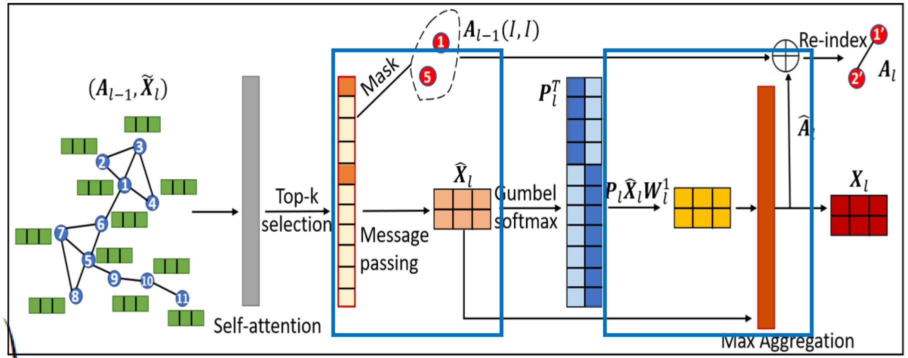
- Supernode selection
  - Self-attention based pooling [1]

$$\mathrm{z}_l = \sigma\left(\tilde{\mathrm{D}}_{l-1}^{-\frac{1}{2}}\tilde{\mathrm{A}}_{l-1}\tilde{\mathrm{D}}_{l-1}^{-\frac{1}{2}}\tilde{\mathrm{X}}_l\mathrm{W}_l^{\mathrm{self}}\right)$$

    - $\widetilde{\boldsymbol{A}}_{l-1} = \boldsymbol{A}_{l-1} + \boldsymbol{I}$ and $\widetilde{\boldsymbol{D}}_{l-1}$ is the degree matrix of $\widetilde{\boldsymbol{A}}_{l-1}$
  - Select nodes $I = \text{top}-\text{rank}(\boldsymbol{z}_l, n_l)$ as supernodes



NetTrans Encoder: Part #2



- Supernode representations
  - Message passing from 1-hop nodes to supernodes

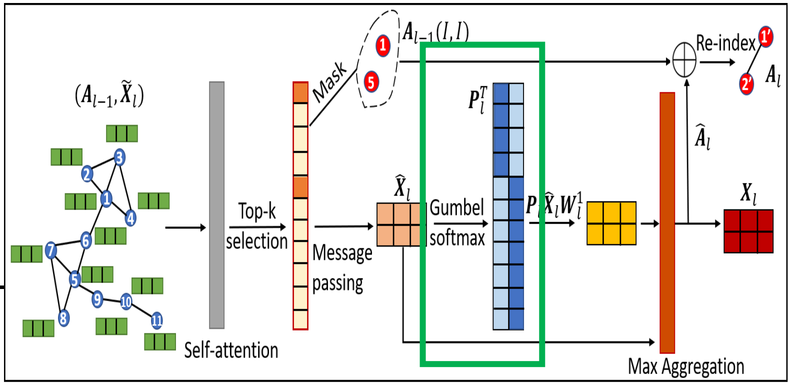
$$\hat{\mathrm{X}}_l(u', :) = \sigma\left(\tilde{\mathrm{X}}_l(\mathcal{I}_{u'}, :)\mathrm{W}_l^1 + \sum_{u \in \mathcal{N}_{u'}} \alpha_{u'u}\tilde{\mathrm{X}}_l(u, :)\mathrm{W}_l^1\right)$$

$$\alpha_{u'u} = \frac{\exp\left(\mathrm{a}_l^T\left[\tilde{\mathrm{X}}_l(\mathcal{I}_{u'}, :)\mathrm{W}_l^1 \| \tilde{\mathrm{X}}_l(u, :)\mathrm{W}_l^1\right]\right)}{\sum_{u_1 \in \mathcal{N}_{u'}} \exp\left(\mathrm{a}_l^T\left[\tilde{\mathrm{X}}_l(\mathcal{I}_{u'}, :)\mathrm{W}_l^1 \| \tilde{\mathrm{X}}_l(u_1, :)\mathrm{W}_l^1\right]\right)}$$

  - Aggregation from distant nodes to supernodes by $\boldsymbol{P}_l \widetilde{\boldsymbol{X}}_l \boldsymbol{W}_l^1$
  - Final supernode representations $\boldsymbol{X}_l = \mathrm{Aggr}\left(\widehat{\boldsymbol{X}}_l, \boldsymbol{P}_l \widetilde{\boldsymbol{X}}_l \boldsymbol{W}_l^1\right)$
- **Q:** How to learn node-to-supernode assignment $\boldsymbol{P}_l$?

NetTrans Encoder: Part #3

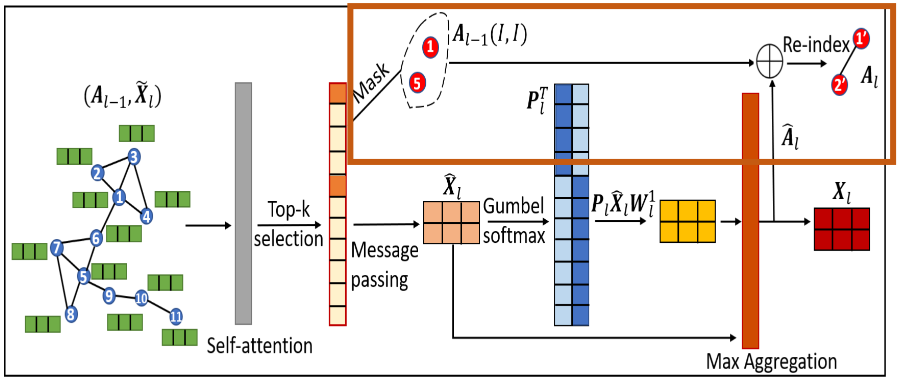
- Node-to-supernode assignment



$$\mathrm{P}_l(u', u) = \frac{\exp\left(\left[\log\left(\hat{\mathrm{X}}_l(u', :)\mathrm{W}_l^g \tilde{\mathrm{X}}_l^T\right) + g_{u'u}\right] / \tau\right)}{\sum_{c \in C(u)} \exp\left(\left[\log\left(\hat{\mathrm{X}}_l(c, :)\mathrm{W}_l^g \tilde{\mathrm{X}}_l^T\right) + g_{cu}\right] / \tau\right)}$$

- Gumbel softmax: approximation to discrete $\boldsymbol{P}_l$
- $\mathcal{C}(u)$: supernode candidates of node u
  - 1-hop: $\mathcal{C}(6) = \{1,5\}$
  - 2-hop: $\mathcal{C}(10) = \{5\}$
  - Others: all supernodes, i.e., $\mathcal{C}(11) = \{1,5\}$

NetTrans Encoder: Part #4



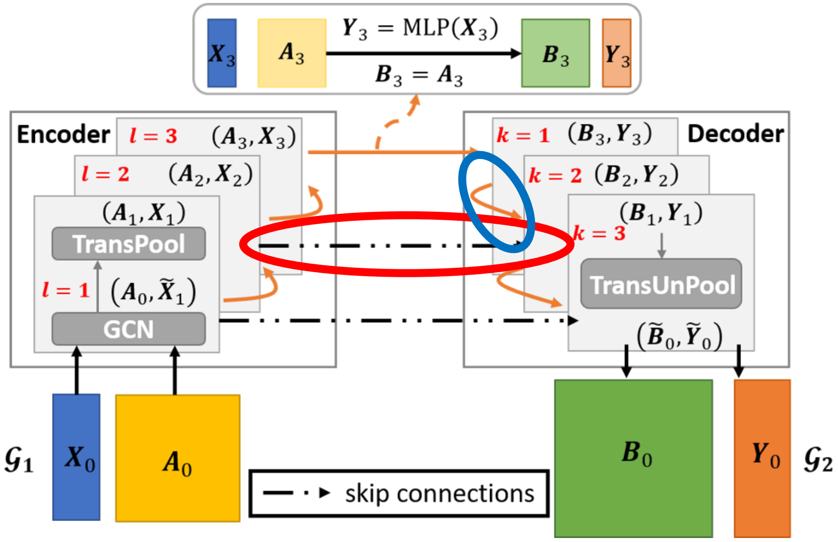
- Supernode connections
  - Use auxiliary connections $\widehat{\boldsymbol{A}}_l$

$$\boldsymbol{A}_l = \frac{1}{2}\left(\boldsymbol{A}_{l-1}(I, I) + \widehat{\boldsymbol{A}}_l\right)$$

  - $$\hat{\mathrm{A}}_l(u_1', u_2') = \begin{cases} 2\sigma_s\left(\mathrm{X}_l(u_1', :)\mathrm{X}_l(u_2', :)^T\right) & \text{if } u_1' \in \mathcal{I}_l^s \text{ or } u_2' \in \mathcal{I}_l^s \\ \sigma_s\left(\mathrm{X}_l(u_1', :)\mathrm{X}_l(u_2', :)^T\right) & \text{if } u_1' \notin \mathcal{I}_l^s \text{ and } u_2' \notin \mathcal{I}_l^s \\ 0 & \text{otherwise} \end{cases}$$
  - I_l^s: isolated supernodes, i.e., $A_{l-1}(I, I) = 0$

NetTrans – Decoder

- Goal: to reconstruct the target network
- Key idea: same latent meanings of supernodes
  - Part #1: leverage $\mathcal{G}_1$ by skip connections
  - Part #2: calibrate part #1 from supernodes to nodes
- Message passing
  - Part #1 -> Msg #1
  - Part #2 -> Msg #2



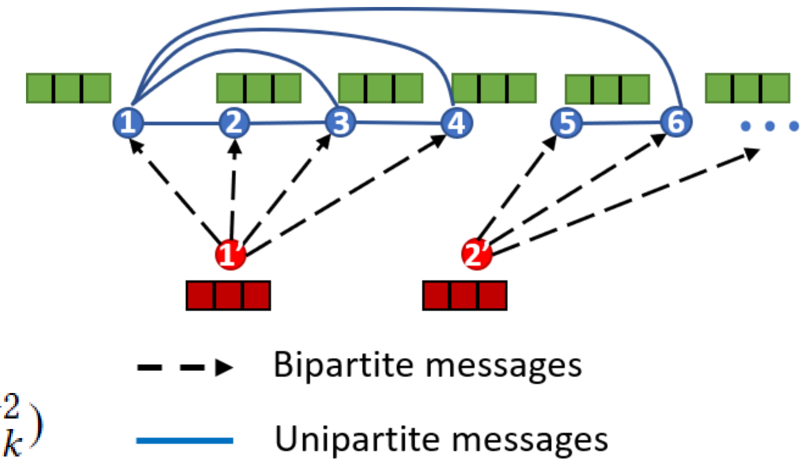
NetTrans Decoder – Message Passing

- Unipartite messages (Msg #1)

$$\mathbf{m}_{v_1 \to v}^{k} = \frac{1}{\sqrt{|\mathcal{N}_v|}\sqrt{|\mathcal{N}_{v_1}|}} \mathbf{X}_{L-k}(v_1, :)\mathbf{W}_k^3$$

- Bipartite messages (Msg #2)

$$\mathbf{m}_{v' \to v}^{k} = \mathbf{P}_{L-k+1}(v', v) \odot (\mathbf{Y}_{L-k+1}(v', :)\mathbf{W}_k^2)$$



- Node representations and network structure

$$\mathbf{Y}_{L-k}(v, :) = \sum_{\substack{v',\ s.t.\\ \mathbf{P}_{L-k+1}(v', v) > 0}} \mathbf{m}_{v' \to v}^{k} + \sum_{v_1 \in \mathcal{N}_v} \mathbf{A}_{L-k}(v_1, v) \odot \mathbf{m}_{v_1 \to v}^{k}$$

$$\mathbf{B}_{L-k}(v, v_1) = \frac{1}{2} \max\{0, \mathbf{A}_{L-k}(v, v_1) + \sigma_t(\mathbf{Y}_{L-k}(v, :)\mathbf{Y}_{L-k}(v_1, :)^T)\}$$

NetTrans – Loss Functions

- Structure reconstruction

$$\mathcal{L}_{\text{adj}} = -\frac{1}{|\mathcal{E}|}\sum_{(v,v_1)\in\mathcal{E}}[y_{v,v_1}\log p_{v,v_1} + (1-y_{v,v_1})\log(1-p_{v,v_1})]$$

- Attribute reconstruction

$$\mathcal{L}_{\text{attr}} = \frac{1}{m_0}\|\mathrm{Y}_0 - \mathrm{MLP}_2(\tilde{\mathrm{Y}}_0)\|_F^2$$

- Observed cross-network node associations
  - Network alignment: margin ranking loss

$$\mathcal{L}_{\text{rank}} = \frac{1}{|\mathcal{O}|}\sum_{(u,v,v_1)\in\mathcal{O}}\max\{0, \lambda - (g_{\text{node}}(u,v) - g_{\text{node}}(u,v_1))\}, \quad g_{\text{node}}(u,v) = \left[\mathrm{P}_1^T(\mathrm{Y}_1\tilde{\mathrm{Y}}_0^T)\right](u,v)$$

  - Recommendation: Bayesian personalized ranking loss

NetTrans – Variants & Generalizations

- Bi-directional cross-network transformation
  - Learn reverse direction as well, i.e., target → source network
- Graph-to-subgraph transformation
  - Source network: large data graph
  - Target network: small query graph

  → subgraph matching
- Dynamic network transformation
  - Source network: $\mathcal{G}^t$ at timestamp t
  - Target network: $\mathcal{G}^{t+1}$ at timestamp $t+1$

  → evolvement
- Single network auto-encoder
  - Source & target networks are same network

Outline

- Motivations ✓
- NetTrans Model ✓
  - Encoder: TransPool
  - Decoder: TransUnPool
- Experimental Results
- Conclusions

Experimental Setup

- Evaluation objectives
  - Effectiveness of learning cross-network node associations
  - Effectiveness of the proposed TransPool and TransUnPool
- Datasets

Tasks	Networks	# of nodes	# of edges	# of attributes
Network Alignment	Cora-1	2,708	5,806	1,433
	Cora-2	2,708	4,547	1,433
	ACM	9,872	39,561	17
	DBLP	9,916	44,808	17
	Foursquare	5,313	54,233	1
	Twitter	5,120	130,575	1
Recommendation	Ciao-user	3,719	65,213	1
	Ciao-product	4,612	49,136	28

- Baseline methods

Network alignment	FINAL-N	FINAL-P	REGAL
	IONE	CrossMNA	
Recommendation	NGCF	GraphRec	SamWalker
	wpZAN	BPR	

Experimental Results #1

Table 3: (Higher is better.) Effectiveness results on network alignment.

	Cora1-Cora2			ACM-DBLP			Foursquare-Twitter		
	Hits@10	Hits@30	Accuracy	Hits@10	Hits@30	Accuracy	Hits@10	Hits@30	Accuracy
NetTrans	**90.98%**	**97.51%**	**89.89%**	**84.09%**	**94.52%**	**58.21%**	**24.68%**	**34.58%**	**9.17%**
FINAL-N	88.73%	90.77%	87.58%	82.91%	90.71%	54.39%	24.09%	33.80%	8.47%
FINAL-P	62.28%	80.01%	54.34%	69.70%	83.12%	36.34%	24.09%	33.80%	8.47%
REGAL	60.90%	69.20%	46.26%	63.68%	71.80%	41.78%	0.15%	2.20%	0.11%
IONE	73.03%	79.92%	42.29%	58.93%	84.19%	33.00%	13.44%	28.17%	4.13%
CrossMNA	59.06%	68.62%	33.26%	42.54%	49.69%	21.04%	3.37%	14.79%	2.48%

Observation: NetTrans outperforms all other baselines for network alignment task

Experimental Results #2

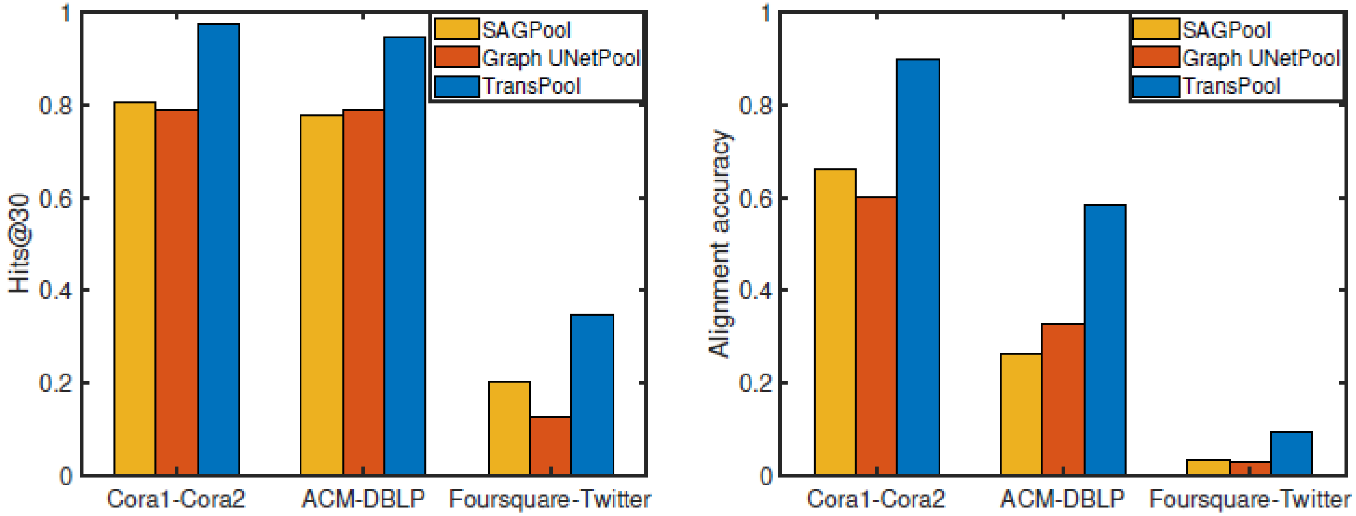
Table 4: (Higher is better.) Effectiveness results on social recommendation.

	Ciao-0.2			Ciao-0.3			Ciao-0.5		
	Prec@10	Rec@10	Rec@50	Prec@10	Rec@10	Rec@50	Prec@10	Rec@10	Rec@50
NetTrans	**13.87%**	**11.08%**	**29.90%**	**11.01%**	**13.23%**	**28.15%**	**10.87%**	**12.43%**	**39.02%**
BPR	1.37%	0.6%	20.25%	1.38%	0.62%	20.18%	1.00%	0.37%	14.97%
wpZAN	11.99%	9.19%	20.77%	9.88%	10.33%	23.22%	9.85%	11.64%	26.04%
GraphRec	8.65%	6.62%	17.56%	8.42%	6.60%	18.07%	6.94%	6.63%	18.08%
SamWalker	4.94%	1.97%	5.98%	4.39%	2.07%	5.67%	2.48%	1.58%	4.05%
NGCF	2.77%	1.21%	3.26%	2.77%	1.48%	3.61%	3.17%	1.99%	4.77%

Observation: NetTrans outperforms all other baselines for recommendation task

Experimental Results #3

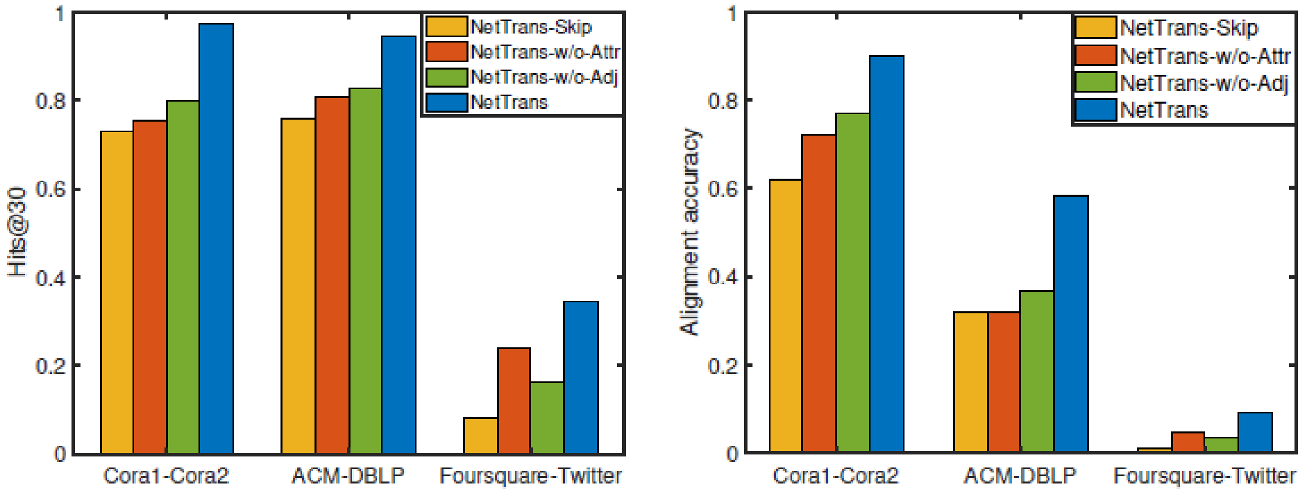
- Ablation study on TransPool layer



Observation: TransPool outperforms both Graph Unet pooling and SAGPool for learning cross-network node associations.

Experimental Results #4

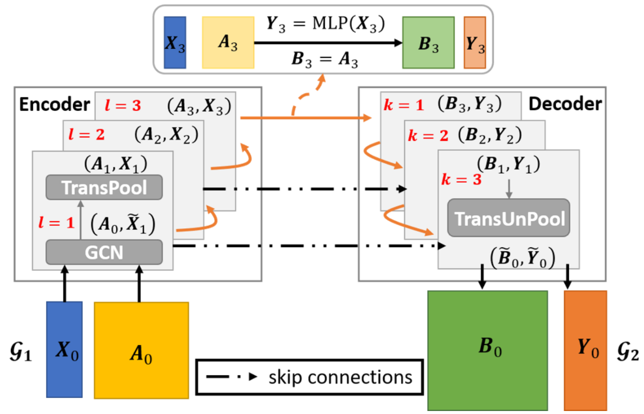
- Ablation study on TransUnPool layer



Observation: TransUnPool outperforms other variants indicating the importance of both structure and node representation calibrations.

Conclusions

- Cross-network transformation
  - Encoder-decoder model – NetTrans
  - Encoder – TransPool
  - Decoder – TransUnPool



- Results
  - NetTrans outperforms baseline methods in both tasks
  - TransPool and TransUnPool achieve better performance than other variants