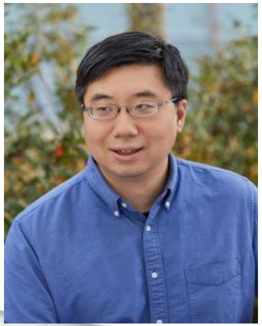
Balancing Consistency and Disparity in Network Alignment

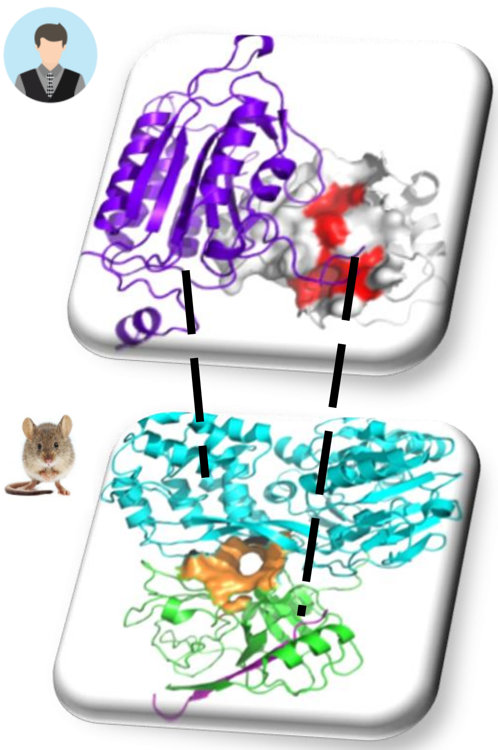
Si Zhang Hanghang Tong Long Jin Yinglong Xia Yunsong Guo



Network Alignment

- Goal: To find node correspondence across networks
- An example:

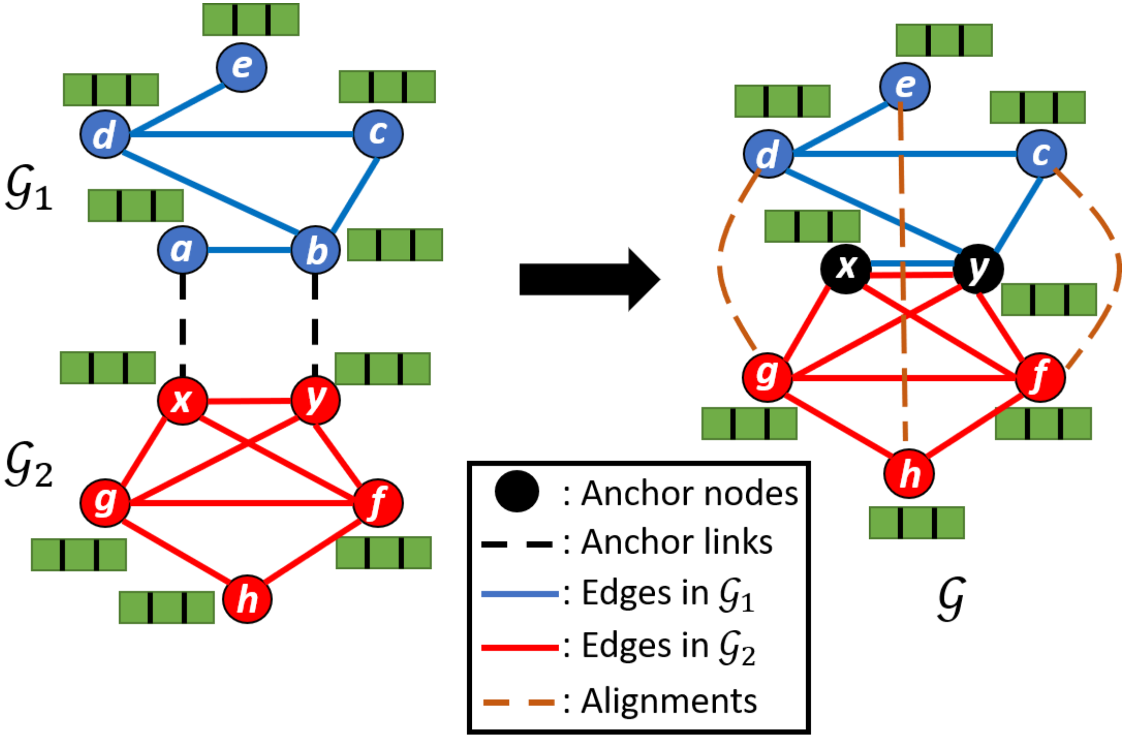


❖ **Evolutionary relationship discovery**

Organism	CHIMP	MOUSE	CHICKEN	FRUIT FLY
Gene Conservation with Humans (%)	99.5	88	75	60

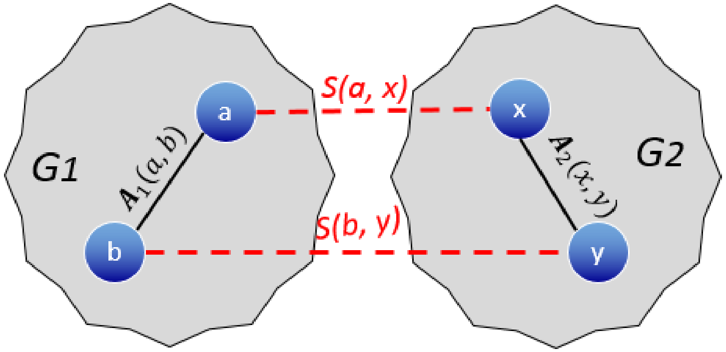
Problem Definition

- **Given:** (1) undirected networks $\mathcal{G}_1 = \{\mathcal{V}_1, \boldsymbol{A}_1, \boldsymbol{X}_1\}$, $\mathcal{G}_2 = \{\mathcal{V}_2, \boldsymbol{A}_2, \boldsymbol{X}_2\}$; (2) a set of anchor links $\mathcal{L}$
- **Output:** alignment matrix $\boldsymbol{S}$



Existing Methods

- Optimization-based methods
  - Key idea: To encourage alignment consistency among neighbors
  - Example formulation (FINAL [1]):
    - Intuition: similar node pairs tend to have similar neighboring node pairs



    - Math:

$$\min_{\mathbf{S}} \sum_{a,b,x,y} \left[\frac{\mathbf{S}(a,x)}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_2(x)|}} - \frac{\mathbf{S}(b,y)}{\sqrt{|\mathcal{N}_1(b)||\mathcal{N}_2(y)|}} \right]^2 \mathbf{A}_1(a,b)\mathbf{A}_2(x,y)$$

alignment differences

neighborhood

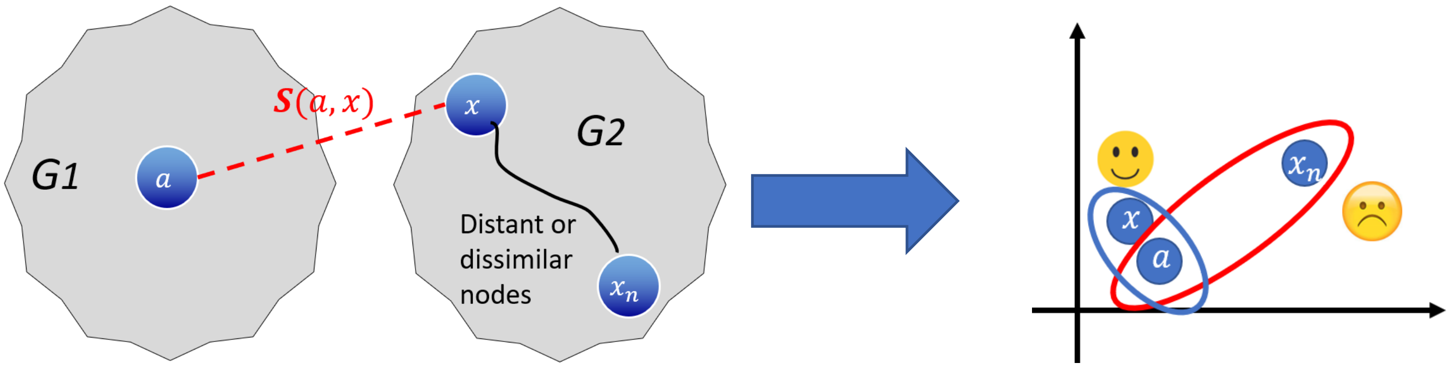
[1] Zhang, Si, and Hanghang Tong. "Final: Fast attributed network alignment." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

Existing Methods (Con't)

- Embedding-based methods
  - Key idea: To learn node embeddings w/ negative sampling
  - Example formulation [1]:
    - Intuition: Nodes that are close in embedding space are more likely to be aligned
    - Math:

Encourage negative samples x_n not to be aligned with a

$$\log p(x|a) \propto \log \sigma(\boldsymbol{x}^T \boldsymbol{a}) + \sum_{m=1}^{K} E_{x_n \sim p_n(x)} \log \sigma(-\boldsymbol{x}_n^T \boldsymbol{a})$$



Limitation #1: Alignment Consistency

- Alignment over-smoothness issue
  - Given an anchor link (a, x), i.e., they are aligned apriori

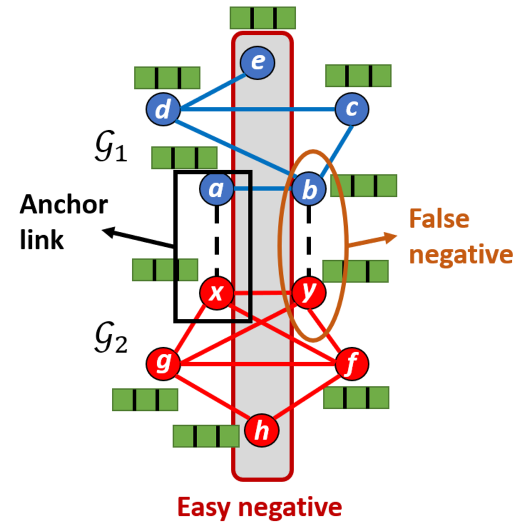
$$\min_{\mathbf{S}} \sum_{a,b,x,y} \left[\frac{\boldsymbol{S}(a,x)}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_2(x)|}} - \frac{\boldsymbol{S}(b,y)}{\sqrt{|\mathcal{N}_1(b)||\mathcal{N}_2(y)|}} \right]^2 \boldsymbol{A}_1(a,b)\boldsymbol{A}_2(x,y)$$

    - Anchor link (a, x) → High $\boldsymbol{S}(a, x)$
    - Minimizing alignment difference → High $\boldsymbol{S}(b, y)$ for **all** neighboring node pairs
    - Cannot distinguish correct alignments from misleading ones
  - Equivalently, neighboring node pairs (b, y) are used as positive samples of (a, x)

Limitation #2: Alignment Disparity

- Negative sampling → disparity → reduce over-smoothness
- Competing sampling strategies

	Alignment consistency	Meaningful disparity	Example negative of anchor (a, x)
Positive correlation [1]	✗	✓	Node pair (b, y)
Negative correlation [2]	✓	✗	Node pair (e, h)
Degree-based sampling [3]	?	?	Node pair (d, x)



[1] Yang, Zhen, et al. "Understanding negative sampling in graph representation learning." KDD. 2020.
[2] Maruf, M., and Anuj Karpatne. "Maximizing Cohesion and Separation in Graph Representation Learning: A Distance-aware Negative Sampling Approach." SDM, 2021.
[3] Liu, Li, et al. "Aligning Users across Social Networks Using Network Embedding." IJCAI. 2016.

Balancing Consistency & Disparity

- Key question:

  What are the intrinsic relationships behind alignment consistency and disparity?

  - Q1: How to design model architecture to encode alignment consistency?
  - Q2: How to sample negative node pairs to distinguish correct alignments from misleading ones?
    - Target #1: Should not violate overall alignment consistency
    - Target #2: Should learn meaningful node embeddings

Outline

- Motivations ✓
- NeXtAlign Model
  - Model Design
  - Model Training
- Experimental Results
- Conclusions

Alignment Consistency by GCNs

- Unsupervised FINAL [1]

$$\min_{S} \sum_{a,b,x,y} \left[\frac{S(a,x)}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_2(x)|}} - \frac{S(b,y)}{\sqrt{|\mathcal{N}_1(b)||\mathcal{N}_2(y)|}} \right]^2 A_1(a,b) A_2(x,y)$$

Fixed-point solution

$$S^t = \widetilde{A}_1 S^{t-1} \widetilde{A}_2$$

- Relationship with GCNs

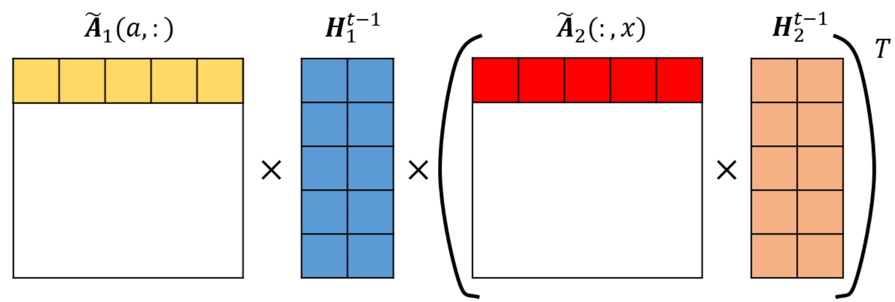
Suppose $S^t = (H_1^t)' \, H_2^t$

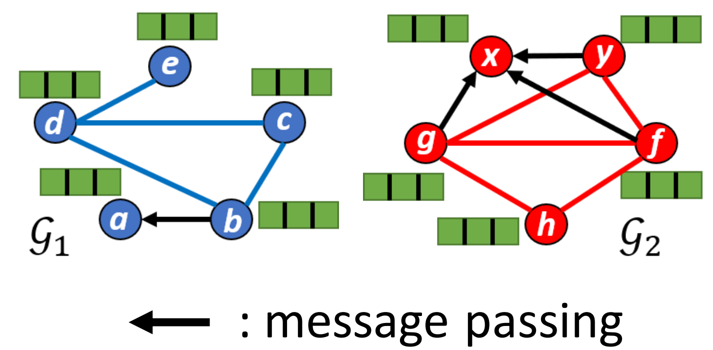
$$S^t(a,x) = (a^t)' x^t = \widetilde{A}_1(a,:) S^{t-1} \widetilde{A}_2(:,x)$$

$$S^t(a,x) = \left(\sum_{b \in \mathcal{N}_1(a)} \frac{b^{t-1}}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_1(b)|}} \right)' \sum_{y \in \mathcal{N}_2(x)} \frac{y^{t-1}}{\sqrt{|\mathcal{N}_2(x)||\mathcal{N}_2(y)|}}$$

Update by GCN w/o parameters

a^t — Inner product — x^t





[1] Zhang, Si, and Hanghang Tong. "Final: Fast attributed network alignment." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

Alignment Consistency by GCNs (Con't)

- Alignment consistency – semi-supervised [1]

$L(a,x) = 1$ if $(a,x) \in \mathcal{L}$

$$\min_{\mathbf{S}} \alpha \sum_{a,b,x,y} \left[\frac{\mathbf{S}(a,x)}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_2(x)|}} - \frac{\mathbf{S}(b,y)}{\sqrt{|\mathcal{N}_1(b)||\mathcal{N}_2(y)|}} \right]^2 \mathbf{A}_1(a,b)\mathbf{A}_2(x,y) + (1-\alpha)\|\mathbf{S}-\mathbf{L}\|_F^2$$

Fixed-point solution

$$\mathbf{S}^t = \alpha \widetilde{\mathbf{A}}_1 \mathbf{S}^{t-1} \widetilde{\mathbf{A}}_2 + (1-\alpha)\mathbf{L}$$

- Message passing w/o parameters

$$\mathbf{u}^t = \sqrt{\alpha} \sum_{b \in \mathcal{N}_1(u)} \frac{\mathbf{b}^{t-1}}{\sqrt{|\mathcal{N}_1(u)||\mathcal{N}_1(b)|}} + \sqrt{1-\alpha}\,\mathbf{u}^{t-1}$$

$$\mathbf{v}^t = \sqrt{\alpha} \sum_{y \in \mathcal{N}_2(v)} \frac{\mathbf{y}^{t-1}}{\sqrt{|\mathcal{N}_2(v)||\mathcal{N}_2(y)|}} + \sqrt{1-\alpha}\,\mathbf{v}^{t-1}$$

Non-anchor

$$\mathbf{a}^t = \mathbf{x}^t = \sqrt{\alpha} \sum_{b \in \mathcal{N}_1(a)} \frac{\mathbf{b}^{t-1}}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_1(b)|}} + \sqrt{1-\alpha}\,\mathbf{x}^{t-1} + \sqrt{\alpha} \sum_{y \in \mathcal{N}_2(x)} \frac{\mathbf{y}^{t-1}}{\sqrt{|\mathcal{N}_2(x)||\mathcal{N}_2(y)|}}$$

$\mathbf{S}^0 = \mathbf{L}$

$\mathbf{a}^0 = \mathbf{x}^0 = \mathbf{e}_i$

Alignment consistency

$$\mathbf{S}(u,v) = \alpha \widetilde{\mathbf{A}}_1(u,:)\mathbf{L}\widetilde{\mathbf{A}}_2(:,v) + (1-\alpha)\mathbf{L}(u,v)$$

$$\mathbf{S}(u,x) = \alpha \widetilde{\mathbf{A}}_1(u,:)\mathbf{L}\widetilde{\mathbf{A}}_2(:,x) + (1-\alpha)\mathbf{L}(u,x) + \alpha \mathbf{S}_1(u,a) + \sqrt{\alpha(1-\alpha)} \frac{\mathbf{A}_1(u,a)}{\sqrt{|\mathcal{N}_1(u)||\mathcal{N}_1(a)|}}$$

$$\mathbf{S}(a,x) = 2\alpha \widetilde{\mathbf{A}}_1(a,:)\mathbf{L}\widetilde{\mathbf{A}}_2(:,x) + (1-\alpha)\mathbf{L}(a,x) + \alpha\big(\mathbf{S}_1(a,a) + \mathbf{S}_2(x,x)\big)$$

Within-network proximity

[1] Zhang, Si, and Hanghang Tong. "Final: Fast attributed network alignment." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

RelGCN – Relational GCN for Alignment

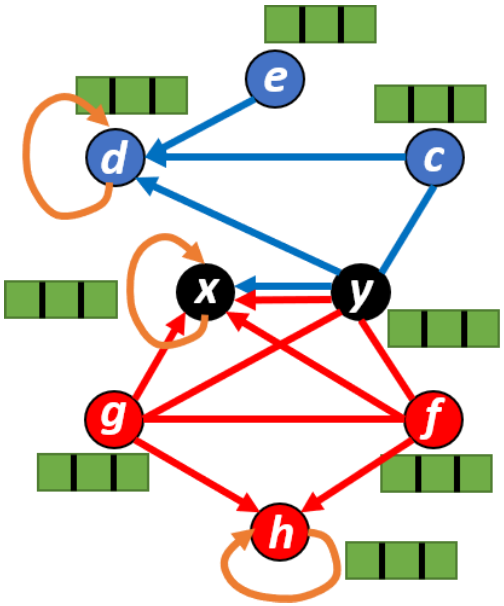
- Message passing w/ parameters

$$u^t = \sqrt{\alpha}\sum_{b\in\mathcal{N}_1(u)}\frac{W_1^t b^{t-1}}{\sqrt{|\mathcal{N}_1(u)||\mathcal{N}_1(b)|}} + \sqrt{1-\alpha}W_0^t u^{t-1}$$

$$v^t = \sqrt{\alpha}\sum_{y\in\mathcal{N}_2(v)}\frac{W_2^t y^{t-1}}{\sqrt{|\mathcal{N}_2(v)||\mathcal{N}_2(y)|}} + \sqrt{1-\alpha}W_0^t v^{t-1}$$

$$a^t = x^t = \sqrt{\alpha}\sum_{b\in\mathcal{N}_1(a)}\frac{W_1^t b^{t-1}}{\sqrt{|\mathcal{N}_1(a)||\mathcal{N}_1(b)|}} + \sqrt{1-\alpha}W_0^t x^{t-1}$$

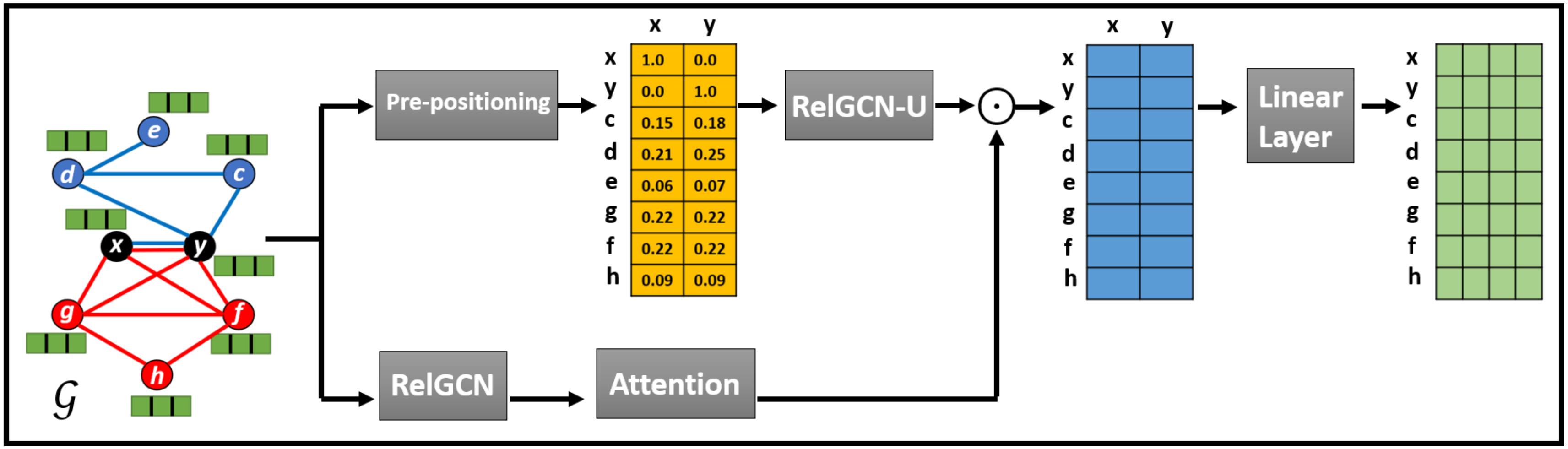
$$+\sqrt{\alpha}\sum_{y\in\mathcal{N}_2(x)}\frac{W_2^t y^{t-1}}{\sqrt{|\mathcal{N}_2(x)||\mathcal{N}_2(y)|}}$$



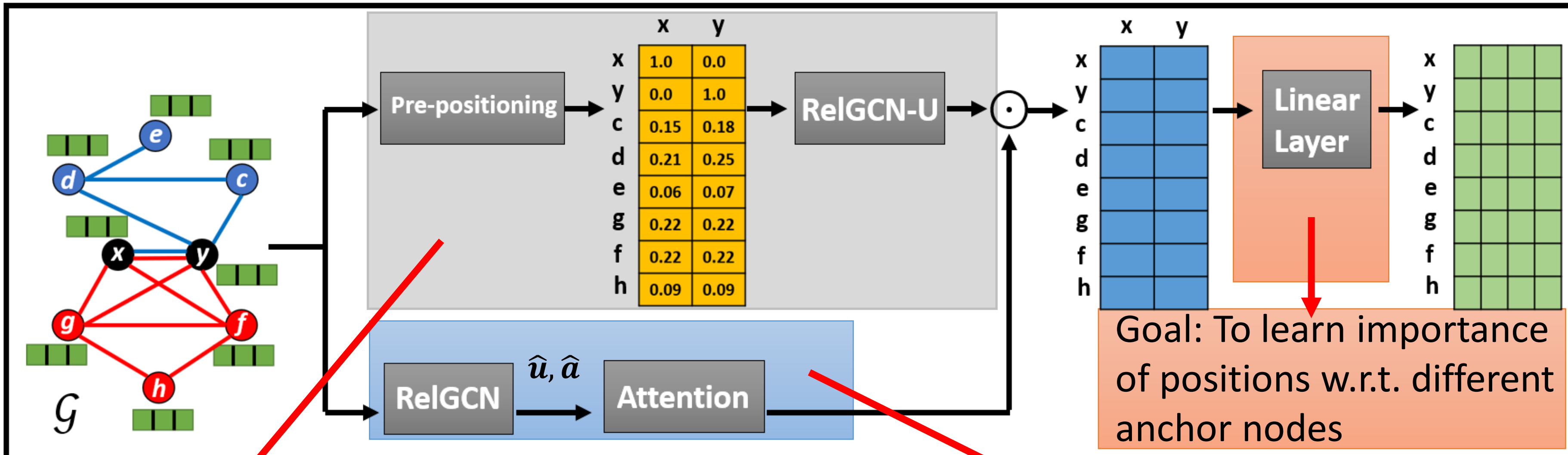
- W_0^t, W_1^t, W_2^t: parameters at the t-th layer
- RelGCN-U: variant w/o parameters

NeXtAlign – Model Design

- Key idea:
  - Use RelGCNs to compute relative positions w.r.t. anchor nodes
  - Feed to a linear layer to compute final embeddings
- Model architecture



Model Design Details



- Goal: To use RelGCN-U to encode alignment consistency
- Pre-positioning:
  - Anchor nodes: $\boldsymbol{a}^0 = \boldsymbol{x}^0 = \boldsymbol{e}_i$
  - Non-anchor nodes: RWR scores w.r.t. anchor nodes [1,2]

- Goal: To mitigate over-smoothness of RelGCN-U
- RelGCN w/ attention to rescale positions

$$c_{ua} = \frac{\exp(\boldsymbol{w}_c'[\hat{\boldsymbol{u}}||\hat{\boldsymbol{a}}])}{\sum_{b \in \mathcal{L}_1} \exp(\boldsymbol{w}_c'[\hat{\boldsymbol{u}}||\hat{\boldsymbol{b}}])}$$

[1] Tong, Hanghang, Christos Faloutsos, and Jia-Yu Pan. "Fast random walk with restart and its applications." Sixth international conference on data mining (ICDM'06). IEEE, 2006.
[2] Yan, Yuchen, Si Zhang, and Hanghang Tong. "BRIGHT: A Bridging Algorithm for Network Alignment." Proceedings of the Web Conference 2021. 2021.

Outline

- Motivations ✓
- NeXtAlign Model
  - Model Design ✓
  - Model Training
- Experimental Results
- Conclusions

NeXtAlign – Model Training

- Loss functions

$$J_a = -\sum_{b \in \mathcal{V}_1} [p_d(b|a) \log \sigma(\boldsymbol{b}'\boldsymbol{a}) + k p_n(b|a) \log \sigma(-\boldsymbol{b}'\boldsymbol{a})]$$

$$J_x = -\sum_{y \in \mathcal{V}_2} [p_d(y|x) \log \sigma(\boldsymbol{y}'\boldsymbol{x}) + k p_n(y|x) \log \sigma(-\boldsymbol{y}'\boldsymbol{x})]$$

Link prediction loss in $\mathcal{G}_1, \mathcal{G}_2$

$$J_{ax} = -\sum_{b \in \mathcal{V}_1} [p_{dc}(b|x) \log \sigma(\boldsymbol{b}'\boldsymbol{x}) + k p_{nc}(b|x) \log \sigma(-\boldsymbol{b}'\boldsymbol{x})] - \sum_{y \in \mathcal{V}_2} [p_{dc}(y|a) \log \sigma(\boldsymbol{y}'\boldsymbol{a}) + k p_{nc}(y|a) \log \sigma(-\boldsymbol{y}'\boldsymbol{a})]$$

Anchor link prediction loss

$$J = \sum_{(a,x) \in \mathcal{L}} J_{a,x} = \sum_{(a,x) \in \mathcal{L}} J_a + J_x + J_{ax}$$

  - p_d, p_n: within-network positive, negative sampling distributions
  - p_{dc}, p_{nc}: cross-network positive, negative sampling distributions
- Question: How to design sampling distributions?

Sampling Strategy

- An intuitive design
  - p_d: similar nodes are likely to co-occur in the context [1]
  - p_n: samples distant/dissimilar nodes [2]
  - p_{dc}: high-similarity node pairs preserve alignment consistency
  - p_{nc}: high-similarity node pairs → hard negative alignment pairs [3] → alignment disparity

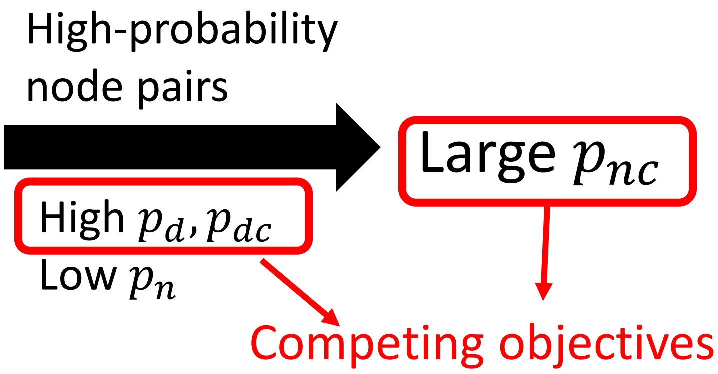
LEMMA *Denote $\Delta\theta_b = \theta_b^B - \theta_b^*$ and $\Delta\theta_y = \theta_y^B - \theta_y^*$. The mean square errors for nodes $b \in \bar{\mathcal{L}}_1$ and $y \in \bar{\mathcal{L}}_2$ can be formulated by*

$$\mathbb{E}\left[\Delta\theta_b^2\right] = \frac{1}{B}\left[\frac{1}{p_d(b|a) + p_{dc}(b|x)} + \frac{1}{kp_n(b|a) + kp_{nc}(b|x)} - C\right]$$

$$\mathbb{E}[\Delta\theta_y^2] = \frac{1}{B}\left[\frac{1}{p_d(y|x) + p_{dc}(y|a)} + \frac{1}{kp_n(y|x) + kp_{nc}(y|a)} - C\right]$$

For nodes $b \in \mathcal{L}_1$ and $y \in \mathcal{L}_2$, the mean square error is computed by

$$\mathbb{E}\left[\Delta\theta_b^2\right] = \mathbb{E}[\Delta\theta_y^2] = \frac{1}{B}\left[\frac{1}{p_1} + \frac{1}{kp_2} - C\right]$$



[1] Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." KDD. 2014.
[2] Maruf, M., and Anuj Karpatne. "Maximizing Cohesion and Separation in Graph Representation Learning: A Distance-aware Negative Sampling Approach." SDM, 2021.
[3] Yang, Zhen, et al. "Understanding negative sampling in graph representation learning." KDD. 2020.

Sampling Strategy (Con't)

- Denote $\boldsymbol{b} = [\boldsymbol{b}_{(1)} || \boldsymbol{b}_{(2)}], \boldsymbol{x} = [\boldsymbol{x}_{(1)} || \boldsymbol{x}_{(2)}]$
  - $b_{(1)}$: captures local information of node-b in $\mathcal{G}_1$
  - $b_{(2)}$: captures how node-b posits in $\mathcal{G}_2$
- A new scoring function → instead of plain inner product

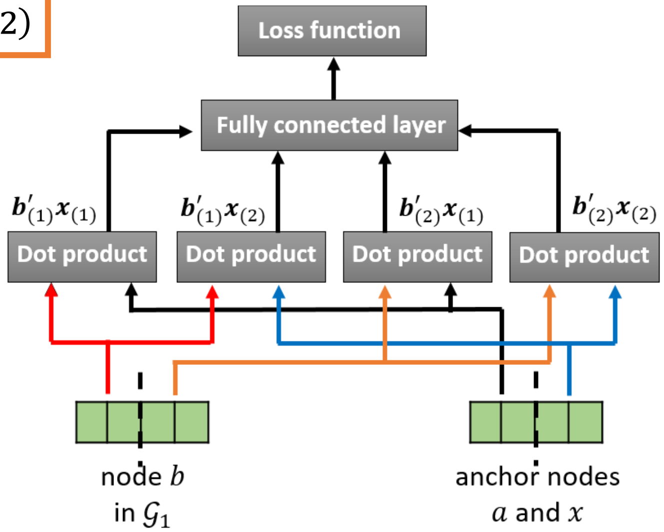
$$\boldsymbol{b} \star \boldsymbol{x} = w_1 \boldsymbol{b}'_{(1)} \boldsymbol{x}_{(1)} + w_2 \boldsymbol{b}'_{(1)} \boldsymbol{x}_{(2)} + w_3 \boldsymbol{b}'_{(2)} \boldsymbol{x}_{(1)} + w_4 \boldsymbol{b}'_{(2)} \boldsymbol{x}_{(2)}$$

$\boldsymbol{a} = \boldsymbol{x}$

Intra-network proximity → p_d, p_n

Node interaction similar as recommendation → p_{nc}

Node interaction in the context of $\mathcal{G}_2$ → p_{dc}



Outline

- Motivations ✓
- NeXtAlign Model
  - Model Design ✓
  - Model Training ✓
- Experimental Results
- Conclusions

Experimental Setup

- Evaluation objectives
  - How accurate is NeXtAlign for network alignment?
  - Effectiveness of different components
- Datasets

Scenarios	Networks	# of nodes	# of edges	# of attributes
S1	ACM	9,872	39,561	17
	DBLP	9,916	44,808	17
S2	Foursquare	5,313	54,233	0
	Twitter	5,120	130,575	0
S3	Phone	1,000	41,191	0
	Email	1,003	4,627	0

- Baseline methods
  - Bright [1], NetTrans [2], FINAL [3], IONE [4], CrossMNA [5]

[1] Yan, Yuchen, Si Zhang, and Hanghang Tong. "BRIGHT: A Bridging Algorithm for Network Alignment." WWW. 2021.
[2] Zhang, Si, et al. "NetTrans: Neural Cross-Network Transformation." KDD. 2020.
[3] Zhang, Si, and Hanghang Tong. "Final: Fast attributed network alignment." KDD. 2016.
[4] Liu, Li, et al. "Aligning Users across Social Networks Using Network Embedding." IJCAI. 2016.
[5] Chu, Xiaokai, et al. "Cross-network embedding for multi-network alignment." WWW. 2019.

Experimental Results #1

Results with 20% training data w/o node attributes.

	ACM-DBLP		Foursquare-Twitter		Phone-Email	
	Hits@10	Hits@30	Hits@10	Hits@30	Hits@10	Hits@30
NeXtAlign	**0.8417±0.0032**	**0.9011±0.0081**	**0.2956±0.0096**	**0.4174±0.0066**	**0.3926±0.0168**	**0.6748±0.0105**
Bright	0.7904±0.0041	0.8669±0.0041	0.2500±0.0154	0.3206±0.0097	0.2570±0.0091	0.5344±0.0086
NetTrans	0.7925±0.0065	0.8356±0.0082	0.2468±0.0036	0.3458±0.0098	0.2650±0.0025	0.5325±0.0075
FINAL	0.6768±0.0080	0.8237±0.0098	0.2357±0.0091	0.3457±0.0091	0.2203±0.0151	0.4586±0.0184
IONE	0.7476±0.0125	0.8453±0.0097	0.1624±0.0109	0.2918±0.0209	0.3779±0.0131	0.6444±0.0084
CrossMNA	0.6532±0.0042	0.7900±0.0041	0.0236±0.0172	0.0751±0.0384	0.1542±0.0041	0.4045±0.0115

Observations:

- Our method NeXtAlign significantly outperforms other baseline methods.
- More improvements on Foursquare-Twitter and Phone-Email whose network structures are disparate (i.e., consistency may not work well).

Experimental Results #2

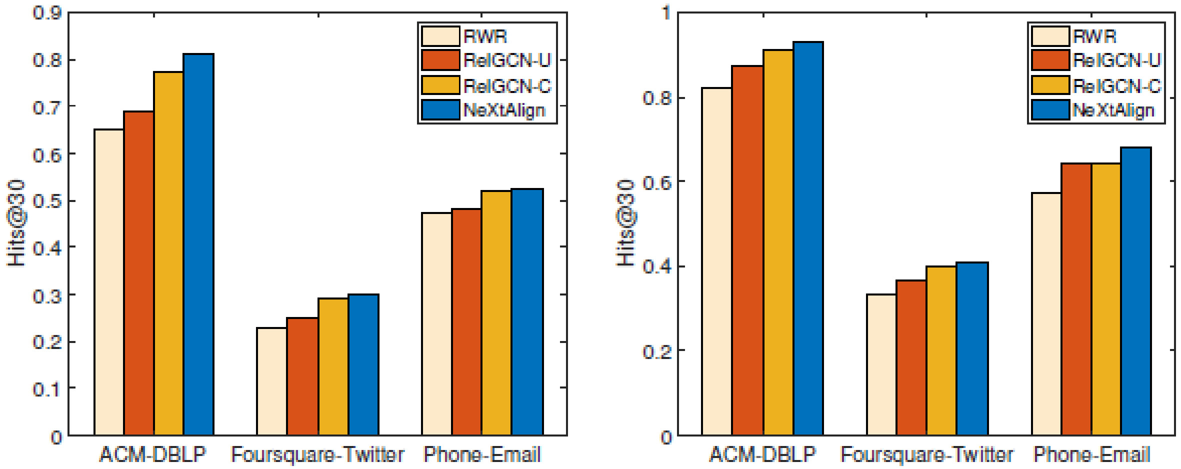
Results with node attributes.

	10% training data		20% training data	
	Hits@10	Hits@30	Hits@10	Hits@30
NeXtAlign	**0.785±0.010**	**0.871±0.009**	**0.872±0.016**	**0.942±0.003**
Bright	0.781±0.004	0.862±0.003	0.797±0.004	0.870±0.006
NetTrans	0.708±0.004	0.846±0.009	0.841±0.010	0.916±0.013
FINAL	0.651±0.013	0.817±0.009	0.825±0.008	0.916±0.006

Observation: Our method NeXtAlign still outperforms other baseline methods.

Experimental Results #3

- Ablation study on model design
  - (1) RWR scores, (2) RelGCN-U: uses output of RelGCN-U, (3) RelGCN-C: uses re-scaled relative positions



(a) 10% training data. (b) 20% training data.

Observation: All components are necessary to achieve the best performance.

Experimental Results #4

- Ablation study on negative sampling strategies

Hits@30 of different negative sampling strategies.

	ACM-DBLP	Foursquare-Twitter	Phone-Email
NeXtAlign	0.9277	0.4103	0.6813
Uniform	0.8975	0.3924	0.6525
Degree	0.9093	0.3923	0.6637
Positive	0.9097	0.4040	0.6650

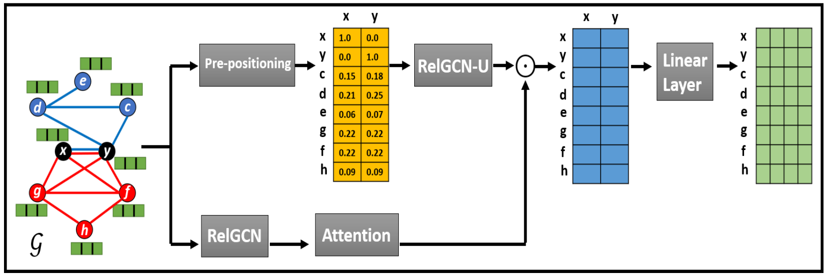
Observation: The proposed negative sampling method achieves a better performance than sampling hard negatives.

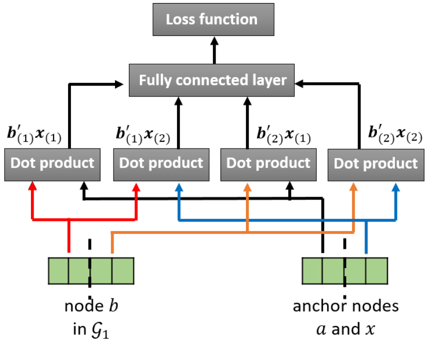
Outline

- Motivations ✓
- NeXtAlign Model ✓
  - Model Design
  - Model Training
- Experimental Results ✓
- Conclusions

Conclusions

- Goal: To strike a balance of alignment consistency and disparity in semi-supervised network alignment
- Method:
  - Model design
    - Connect GCNs with FINAL
    - RelGCN for alignment consistency
  - Model training
    - New sampling method for disparity





- Results
  - NeXtAlign significantly outperforms baseline methods
  - The proposed sampling method achieves better performance


Thank you

Embedding Mean Square Errors

- Empirical risk $J^B_{(a,x)}$
  - Sample B nodes by p_d, p_n, p_{dc}, p_{nc}
- Denote $\boldsymbol{\theta} = [\boldsymbol{b}'_1\boldsymbol{x}, \cdots, \boldsymbol{b}'_{n_1}\boldsymbol{x}, \boldsymbol{y}'_1\boldsymbol{x}, \cdots, \boldsymbol{y}'_{n_2}\boldsymbol{x}]$
  - θ^*, θ^B: optimal embedding to $J_{(a,x)}, J^B_{(a,x)}$

$$J^B_{(a,x)} = -\frac{1}{B}\sum_{i_1,i_2,j_1,j_2}(\log\sigma(\boldsymbol{b}'_{i_1}\boldsymbol{x}) + \log\sigma(\boldsymbol{b}'_{i_2}\boldsymbol{x}) + \log\sigma(\boldsymbol{y}'_{j_1}\boldsymbol{x}) + \log\sigma(\boldsymbol{y}'_{j_2}\boldsymbol{x})) - \frac{1}{B}\sum_{i_3,i_4,j_3,j_4}(\log\sigma(-\boldsymbol{b}'_{i_3}\boldsymbol{x}) + \log\sigma(-\boldsymbol{b}'_{i_4}\boldsymbol{x}) + \log\sigma(-\boldsymbol{y}'_{j_3}\boldsymbol{x}) + \log\sigma(-\boldsymbol{y}'_{j_4}\boldsymbol{x}))$$

LEMMA *Denote $\Delta\theta_b = \theta^B_b - \theta^*_b$ and $\Delta\theta_y = \theta^B_y - \theta^*_y$. The mean square errors for nodes $b \in \bar{\mathcal{L}}_1$ and $y \in \bar{\mathcal{L}}_2$ can be formulated by*

$$\mathbb{E}\left[\Delta\theta_b^2\right] = \frac{1}{B}\left[\frac{1}{p_d(b|a) + p_{dc}(b|x)} + \frac{1}{kp_n(b|a) + kp_{nc}(b|x)} - C\right]$$

$$\mathbb{E}[\Delta\theta_y^2] = \frac{1}{B}\left[\frac{1}{p_d(y|x) + p_{dc}(y|a)} + \frac{1}{kp_n(y|x) + kp_{nc}(y|a)} - C\right]$$

For nodes $b \in \mathcal{L}_1$ and $y \in \mathcal{L}_2$, the mean square error is computed by

$$\mathbb{E}\left[\Delta\theta_b^2\right] = \mathbb{E}[\Delta\theta_y^2] = \frac{1}{B}\left[\frac{1}{p_1} + \frac{1}{kp_2} - C\right]$$

[1] Yang, Zhen, et al. "Understanding negative sampling in graph representation learning." KDD. 2020.