Disentangling Degree-related Biases and Interest for Out-of-distribution Generalized Directed Network Embedding

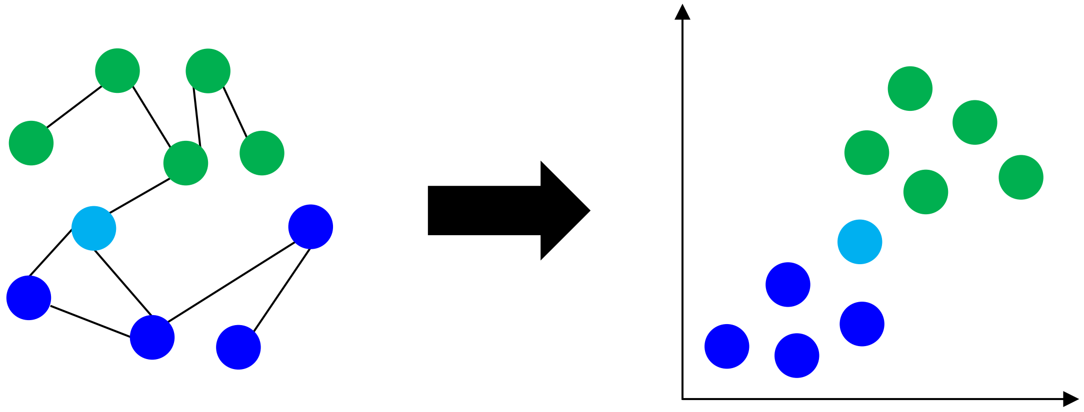
2023.02.02

Hyunsik Yoo

* This is a joint work with Prof. Sang-Wook Kim and Dr. Yeon-Chang Lee at **Hanyang Univ.**, and Prof. Kijung Shin at **KAIST**

Background: Network Embedding (NE)

- **Represents nodes in a given network as low-dimensional vectors that preserve the structural properties of the network**
  - *e.g.*, proximity between nodes

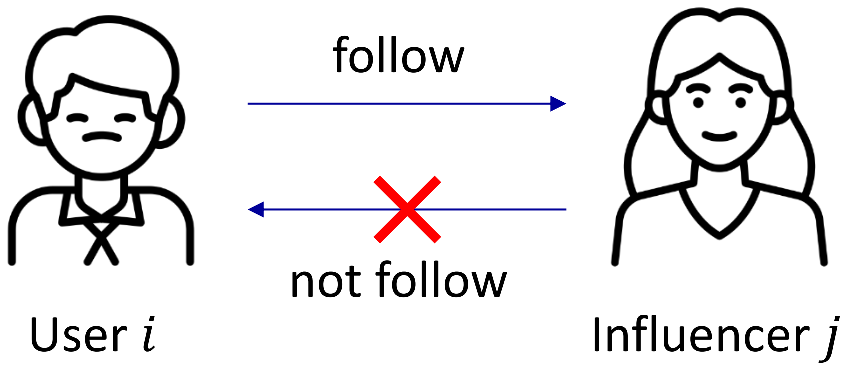


- **The learned embeddings can be used as informative features of nodes in various downstream network mining tasks**
  - **Link prediction → Our focus**
  - Node clustering/classification
  - Recommendation

Background: Directed Network

☐ A directed network

- A directed edge from node i to j expresses an asymmetric relationship (or proximities) between two nodes
- A toy example on Instagram

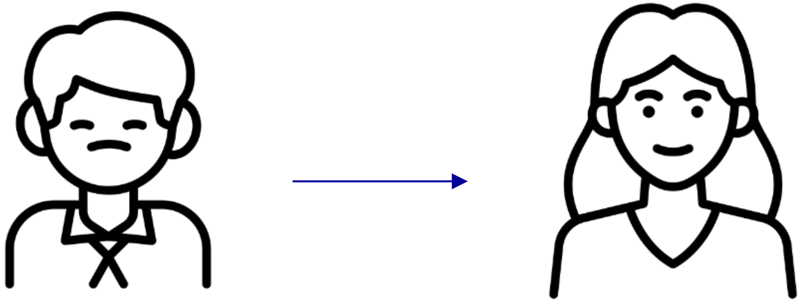


☐ To capture such asymmetric relationships accurately, various directed network embedding (DNE) methods have been proposed

- APP [AAAI'17]
- ATP [AAAI'19]
- NERD [ECML-PKDD'19]
- GVAE [CIKM'19]
- DiGCN [NeurIPS'20]
- MagNet [NeurIPS'21]
- DGGAN [AAAI'21]

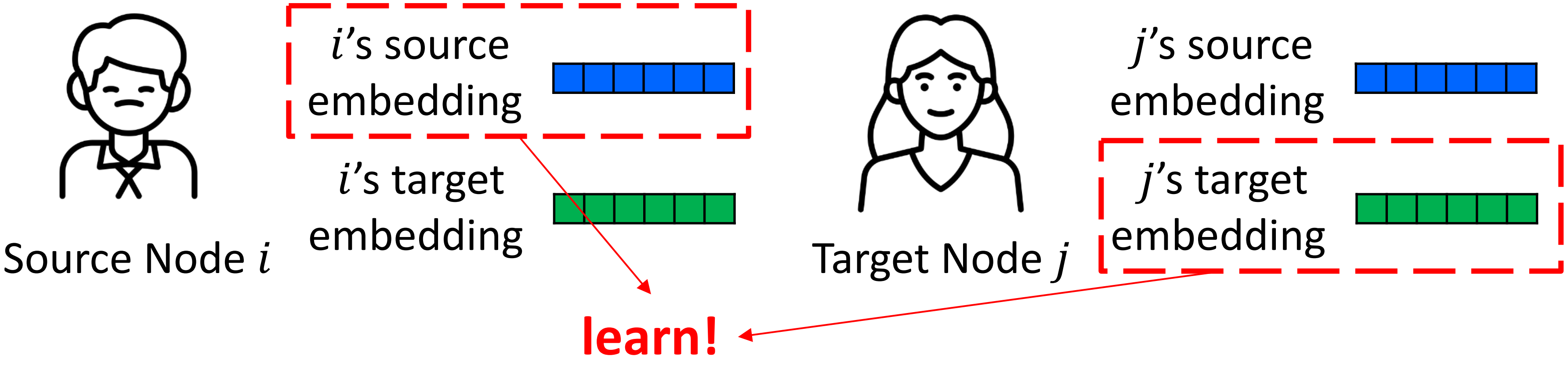
Background: Directed Network (cont'd)

- **Given a directed edge from i to j,**



User i Influencer j

- Distinguish the source node i and the target node j according to their roles
- Learn a source embedding and a target embedding, which preserve the node's properties as sources and targets

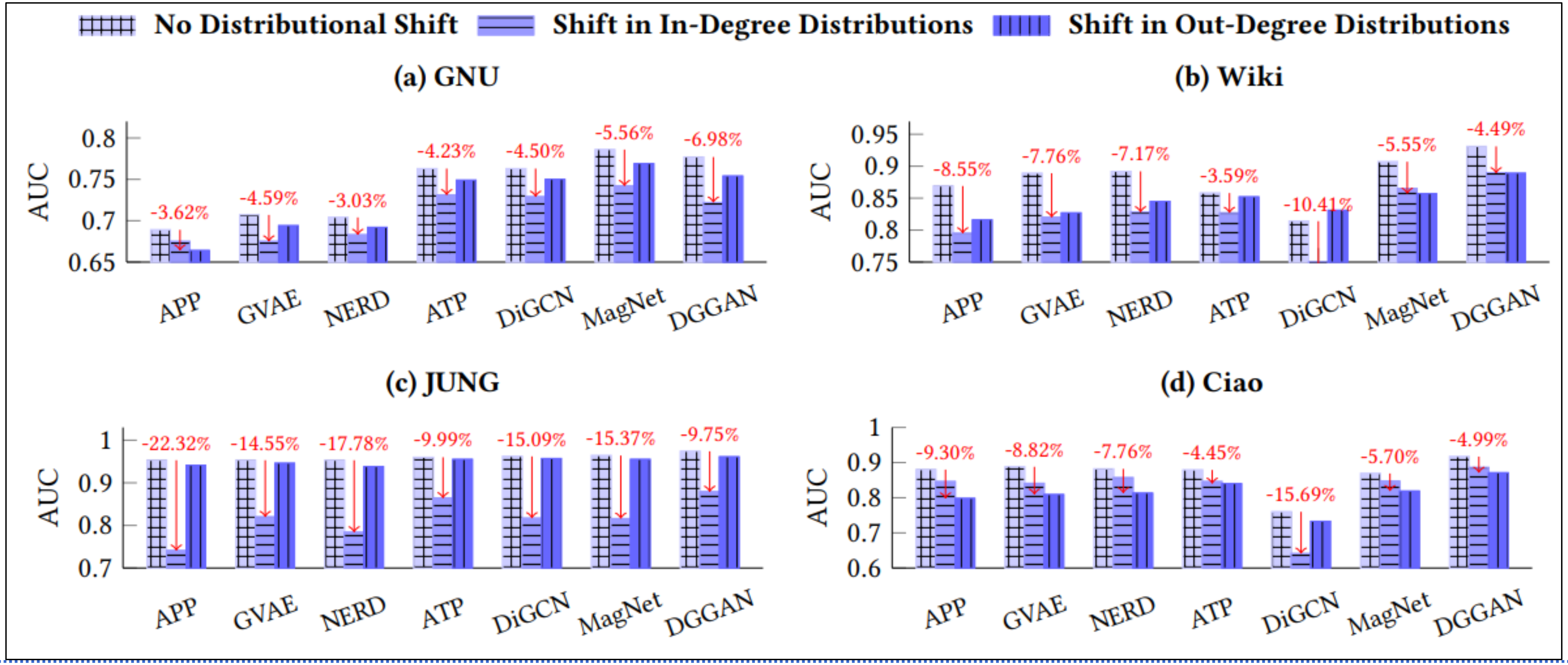


Motivation

- **Existing DNE methods lack out-of-distribution (OOD) generalization abilities against degree-related distributional shifts**
  - They assume that, in link prediction, the degree distribution of the training and test data are identical
- **However, in real-world scenarios, degree-related distributional shifts occur frequently → ruining the identical distribution (ID) assumption!**
  - Preferential attachment
  - Fitness model: it is also common that dominant hubs are overtaken by "new kids on the block" with higher fitness
    - *e.g.*, Google passed established search engines, such as Alta Vista

Motivation (cont'd)

- Link prediction accuracy of the existing methods in ID / non-ID settings
  - The accuracies of all methods significantly degrade in the non-ID settings compared to the ID settings

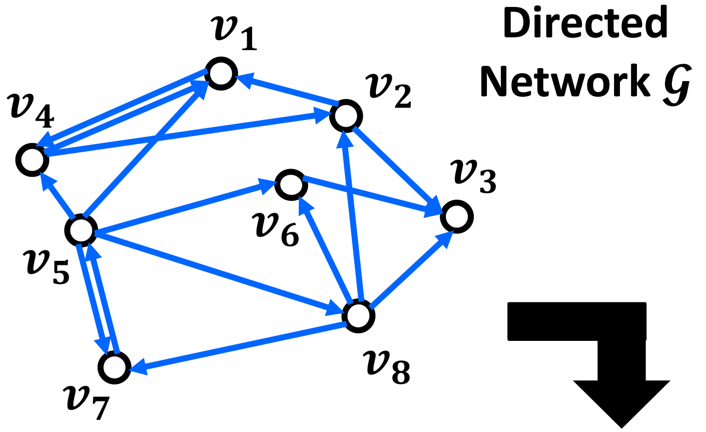


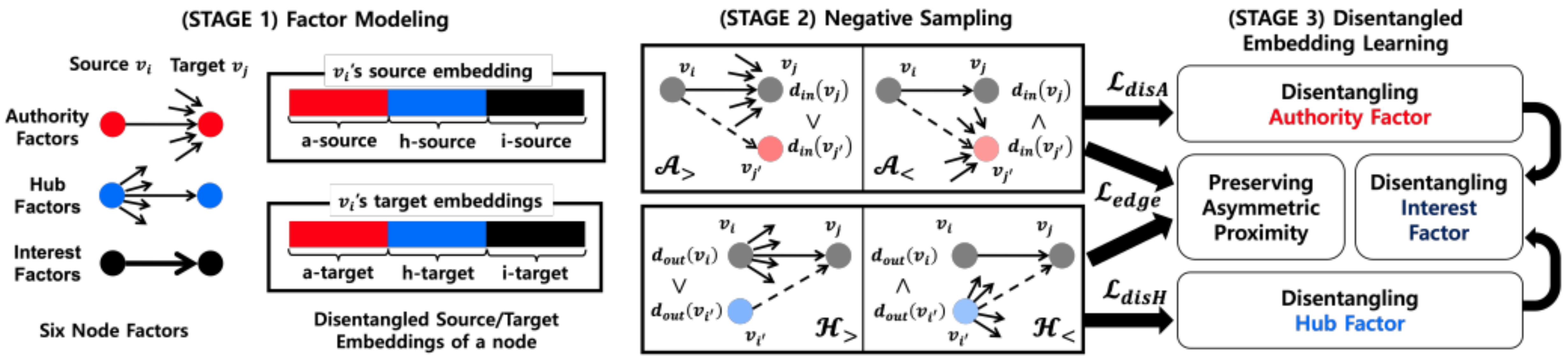
Proposed Method

- **Our idea: model and exploit biases related to node degrees for robustness against degree-related distributional shifts in DNE**

- **Propose ODIN (Out-of-Distribution Generalized Directed Network Embedding), which is designed to answer the following questions:**
  1. *How to model the formation of each directed edge?*
     - Define six node factors that can influence the formation of a directed edge from source to target
  2. *How to leverage such modeled factors for learning OOD generalized embeddings?*
     - Learn multiple factor embeddings, each of which preserves its desired factor

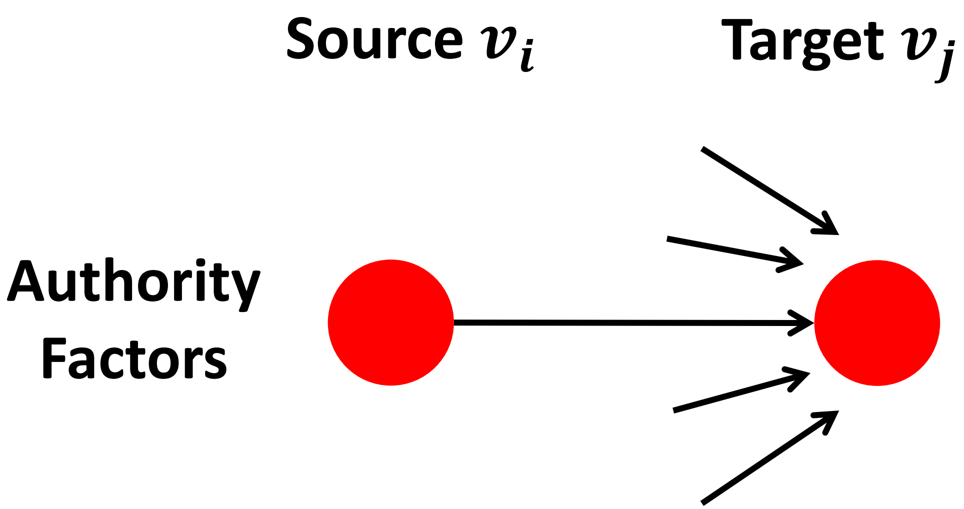
Overview of ODIN





(STAGE 1) Factor Modeling

- **Model the formation of each directed edge (v_i, v_j) based on six node factors grouped as follows:**
  1. **Authority factors**
     - (a) The target v_j's authority status **(a-target)** and (b) the source v_i's bias toward authorities **(a-source)**
     - They together model a bias related to target v_j's authority status (i.e., in-degree)

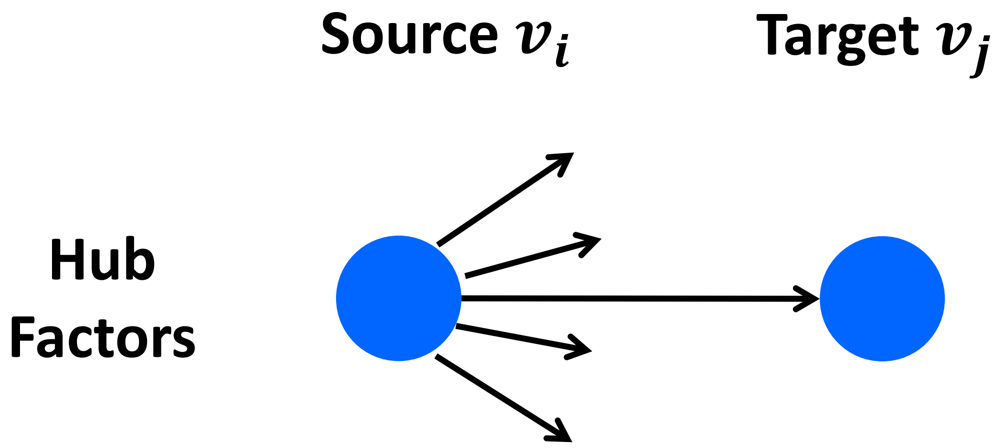


(STAGE 1) Factor Modeling (cont'd)

- **Model the formation of each directed edge (v_i, v_j) based on six node factors grouped as follows:**

  2. **Hub factors**

     - (a) The source v_i's hub status **(h-source)** and (b) the target v_j's bias toward hubs **(h-target)**
     - They together model a bias related to source v_i's hub status (i.e., out-degree)



(STAGE 1) Factor Modeling (cont'd)

- **Model the formation of each directed edge (v_i, v_j) based on six node factors grouped as follows:**

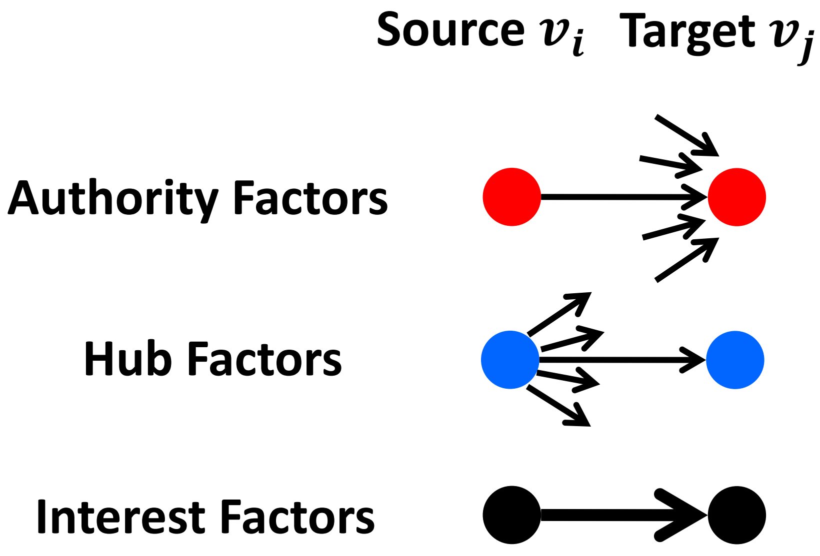
3. **Interest factors**
   - (a) The source v_i's intrinsic property as a source **(i-source)** and (b) the target v_j's intrinsic property as a target **(i-target)**
   - They together model the pure interest in forming an edge from v_i to v_j after removing degree-related biases

Source v_i **Target v_j**

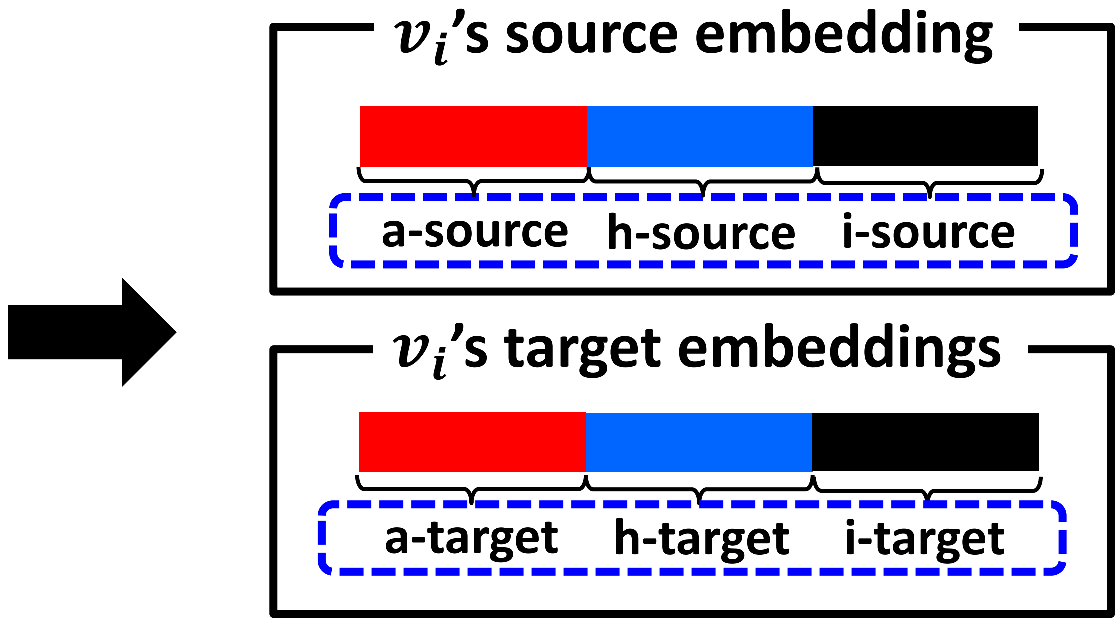
Interest Factors

(STAGE 1) Factor Modeling (cont'd)

- **Represent a node v_i as six factor sub-embeddings**
  - Source role: $\mathbf{a}_i^{src}, \mathbf{h}_i^{src}, \mathbf{i}_i^{src}$
  - Target role: $\mathbf{a}_i^{tar}, \mathbf{h}_i^{tar}, \mathbf{i}_i^{tar}$
- **Concatenate the three factor sub-embeddings as a source/target**
  - $\mathbf{s}_i = a_i^{src} \oplus h_i^{src} \oplus i_i^{src}$,
  - $\mathbf{t}_i = a_i^{tar} \oplus h_i^{tar} \oplus i_i^{tar}$



Six Node Factors



Disentangled Source/Target Embeddings

(STAGE 1) Factor Modeling (cont'd)

- **Compute three factor scores based on the six factor sub-embeddings**
  - Represent how much (a) the authority factor, (b) the hub factor, and (c) the interest factor affect the formation of the directed edge (v_i, v_j)

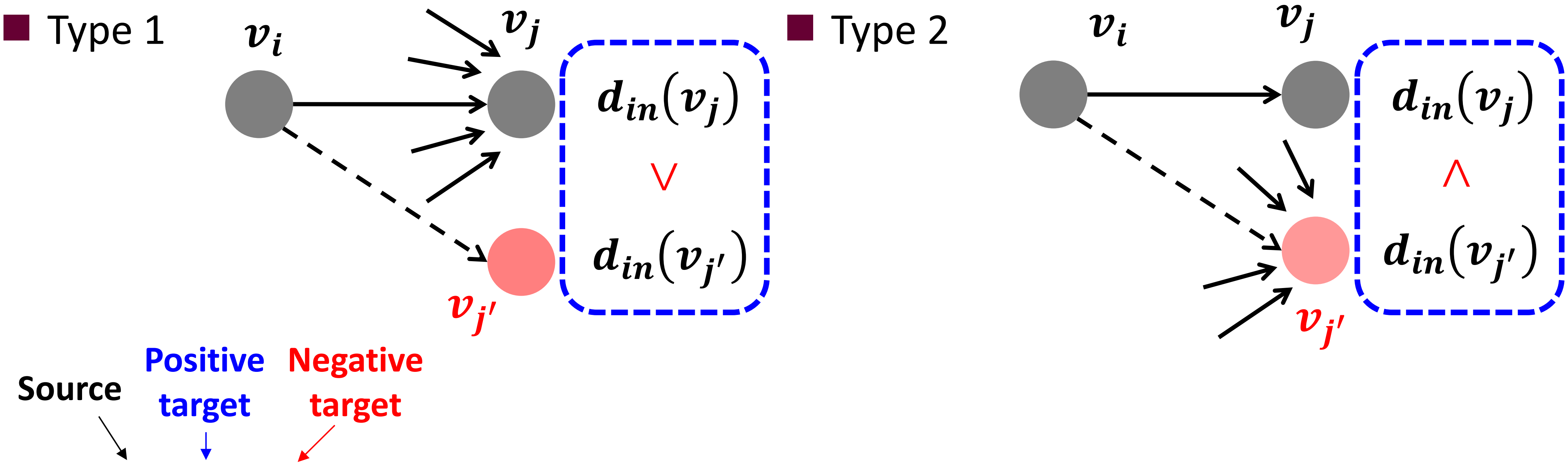
(a) $s_{ij}^{auth} = \mathbf{a}_i^{src} \circ \mathbf{a}_j^{tar}$ (b) $s_{ij}^{hub} = \mathbf{h}_i^{src} \circ \mathbf{h}_j^{tar}$ (c) $s_{ij}^{int} = \mathbf{i}_i^{src} \circ \mathbf{i}_j^{tar}$

- **Compute the overall edge score by adding the three factor scores**
  - Represent the likelihood of the formation of a directed edge (v_i, v_j)

$$s_{ij}^{edge} = s_{ij}^{auth} + s_{ij}^{hub} + s_{ij}^{int} (= \mathbf{s}_i \circ \mathbf{t}_i)$$

(STAGE 2) Negative Sampling

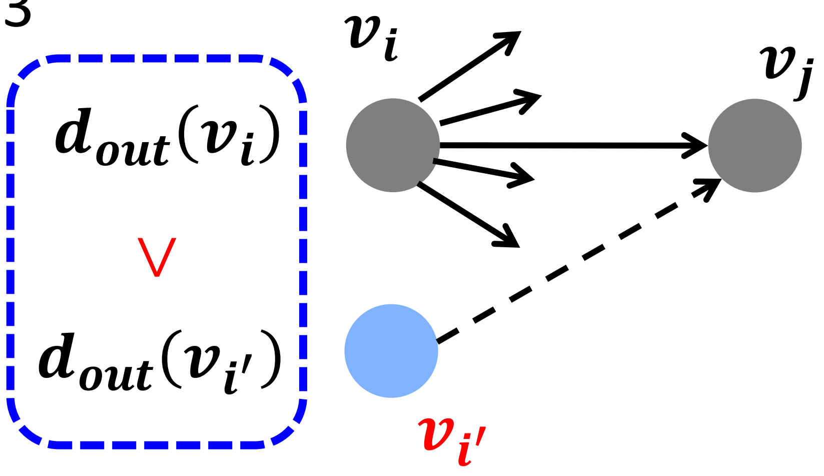
- For each existent edge (v_i, v_j), sample **different types of training instances (i.e., triplets)** for embedding learning
  - Type 1
  - Type 2



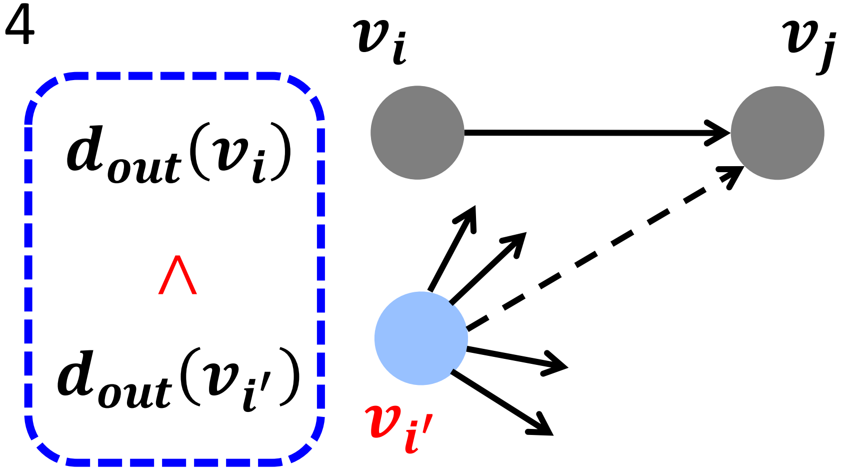
- **Add $(v_i, v_j, v_{j'})$ to the sets $A_>$ (for type 1) or $A_<$ (for type 2)**
  - Aid in capturing the influence of the bias related to the target's in-degree
  - $A = A_> \cup A_<$

(STAGE 2) Negative Sampling (cont'd)

- **For each existent edge (v_i, v_j), sample different types of training instances (i.e., triplets) for embedding learning**
  - Type 3



  - Type 4



Positive source

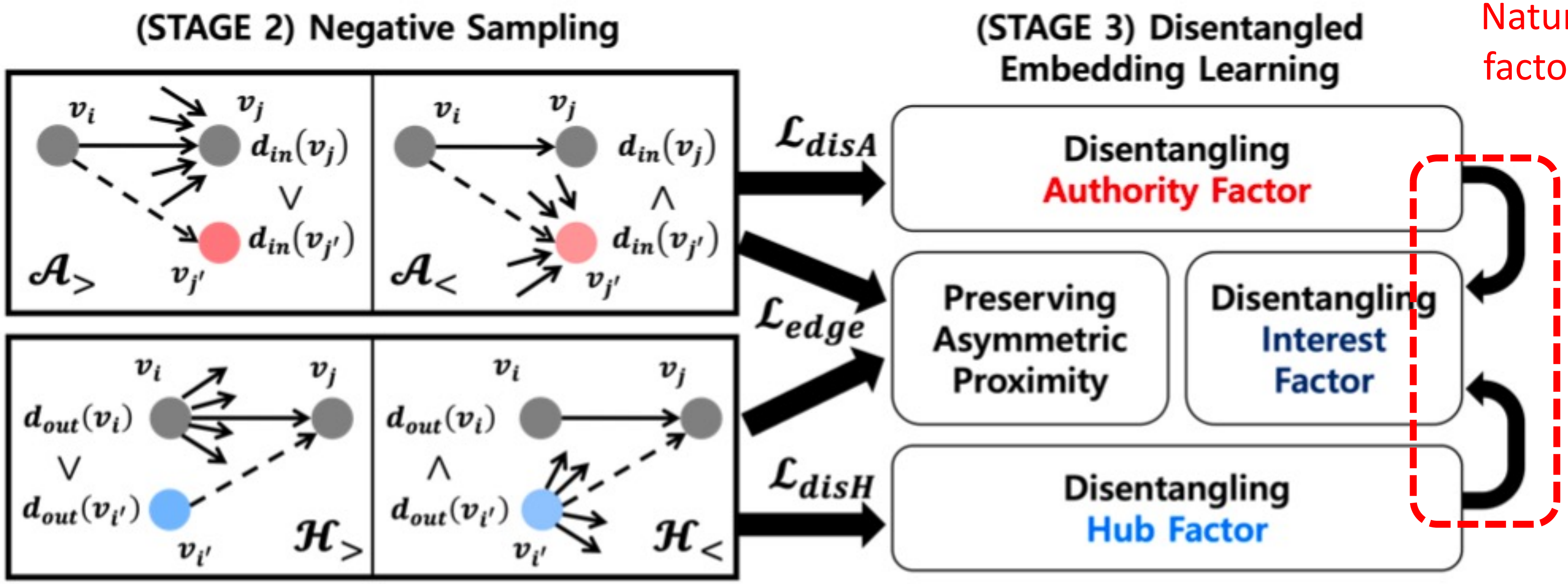
Target

Negative source

- **Add $(v_i, v_j, v_{i'})$ to the sets $H_>$ (for type 3) or $H_<$ (for type 4)**
  - Aid in capturing the influence of the bias related to the source's out-degree
  - $H = H_> \cup H_<$

(STAGE 3) Disentangled Embedding Learning

- **Learn the disentangled source and target embeddings of each node based on the sampled instances via the three objectives**
  1. L_{edge}: preserve asymmetric proximity between nodes in the input network
  2. L_{disA}: disentangle the authority factor from the other two factors
  3. L_{disH}: disentangle the hub factor from the other two factors



Naturally infer the interest factor through L_{disA}, L_{disH}

Loss Function: Multi-Objective Learning

$$L = \boxed{L_{edge}(A \cup H)} + \alpha\left(\boxed{L_{disA}(A)} + \boxed{L_{disH}(H)}\right)$$

① Preserving asymmetric proximities

② Disentangling the authority factor from the others

③ Disentangling the authority factor from the others

① $L_{edge}(A \cup H) =$

*$\mathrm{BPR}\left(s_{ij}^{edge}, s_{ij'}^{edge}\right) = -log\left(\sigma\left(s_{ij}^{edge} - s_{ij'}^{edge}\right)\right)$

$$\sum_{(v_i, v_j, v_{j'}) \in A} \mathrm{BPR}\left(s_{ij}^{edge}, s_{ij'}^{edge}\right) + \sum_{(v_i, v_j, v_{i'}) \in H} \mathrm{BPR}\left(s_{ij}^{edge}, s_{i'j}^{edge}\right)$$

➔ For $(v_i, v_j, v_{j'})$ in A,

$$s_{ij}^{edge} > s_{ij'}^{edge}$$

➔ For $(v_i, v_j, v_{i'})$ in H,

$$s_{ij}^{edge} > s_{i'j}^{edge}$$

However, L_{edge} alone does not contribute to preserving the desired factor in each factor sub-embedding

Loss Function: Multi-Objective Learning (cont'd)

$$L = \boxed{L_{edge}(A \cup B)} + \alpha\left(\boxed{L_{disA}(A)} + \boxed{L_{disH}(H)}\right)$$

① Preserving asymmetric proximities

② Disentangling the authority factor from the others

③ Disentangling the authority factor from the others

② $L_{disA}(A) = \underline{L_{auth}(A_{>})} + L_{auth}(A_{<}) + L_{hub+int}(A_{<})$

➔ For $(v_i, v_j, v_{j'})$ in $A_{>}$, $s_{ij}^{auth} > s_{ij'}^{auth}$

- The authority status (i.e., in-degree) of v_j is higher than that of $v_{j'}$
- Thus, if authority-factor scores capture the biases towards authorities, as desired, $s_{ij}^{auth} > s_{ij'}^{auth}$ **should hold!**

Loss Function: Multi-Objective Learning (cont'd)

$$L = \boxed{L_{edge}(A \cup B)} + \alpha(\boxed{L_{disA}(A)} + \boxed{L_{disH}(H)})$$

① Preserving asymmetric proximities

② **Disentangling the authority factor from the others**

③ Disentangling the authority factor from the others

$$② \ L_{disA}(A) = L_{auth}(A_{>}) + \underline{L_{auth}(A_{<})} + L_{hub+int}(A_{<})$$

➔ For $(v_i, v_j, v_{j'})$ in $A_{<}$, $s_{ij}^{auth} < s_{ij'}^{auth}$

- The authority status (i.e., in-degree) of v_j is lower than that of $v_{j'}$
- Thus, if authority-factor scores capture the biases towards authorities, as desired, $s_{ij}^{auth} < s_{ij'}^{auth}$ **should hold!**

Loss Function: Multi-Objective Learning (cont'd)

$$L = \boxed{L_{edge}(A \cup B)} + \alpha(\boxed{L_{disA}(A)} + \boxed{L_{disH}(H)})$$

① Preserving asymmetric proximities

② **Disentangling the authority factor from the others**

③ Disentangling the authority factor from the others

② $L_{disA}(A) = L_{auth}(A_>) + L_{auth}(A_<) + \underline{L_{hub+int}(A_<)}$

➔ For $(v_i, v_j, v_{j'})$ in $A_<$,

$$s_{ij}^{hub} + s_{ij}^{int} > s_{ij'}^{hub} + s_{ij'}^{int}$$

- Even though $s_{ij}^{auth} < s_{ij'}^{auth}$ holds, s_{ij}^{edge} should become higher than $s_{ij'}^{edge}$
- Thus, we can imply the following inequality $s_{ij}^{hub} + s_{ij}^{int} > s_{ij'}^{hub} + s_{ij'}^{int}$!

Loss Function: Multi-Objective Learning (cont'd)

$$L = \boxed{L_{edge}(A \cup B)} + \alpha(\boxed{L_{disA}(A)} + \boxed{L_{disH}(H)})$$

① Preserving asymmetric proximities

② Disentangling the authority factor from the others

③ Disentangling the hub factor from the others

③ $L_{disH}(H) = \underline{L_{hub}(H_{>})} + \underline{L_{hub}(H_{<})} + \underline{L_{auth+int}(H_{<})}$

For $(v_i, v_j, v_{i'})$ in $H_{<}$, $s_{ij}^{hub} < s_{i'j}^{hub}$

For $(v_i, v_j, v_{i'})$ in $H_{>}$,

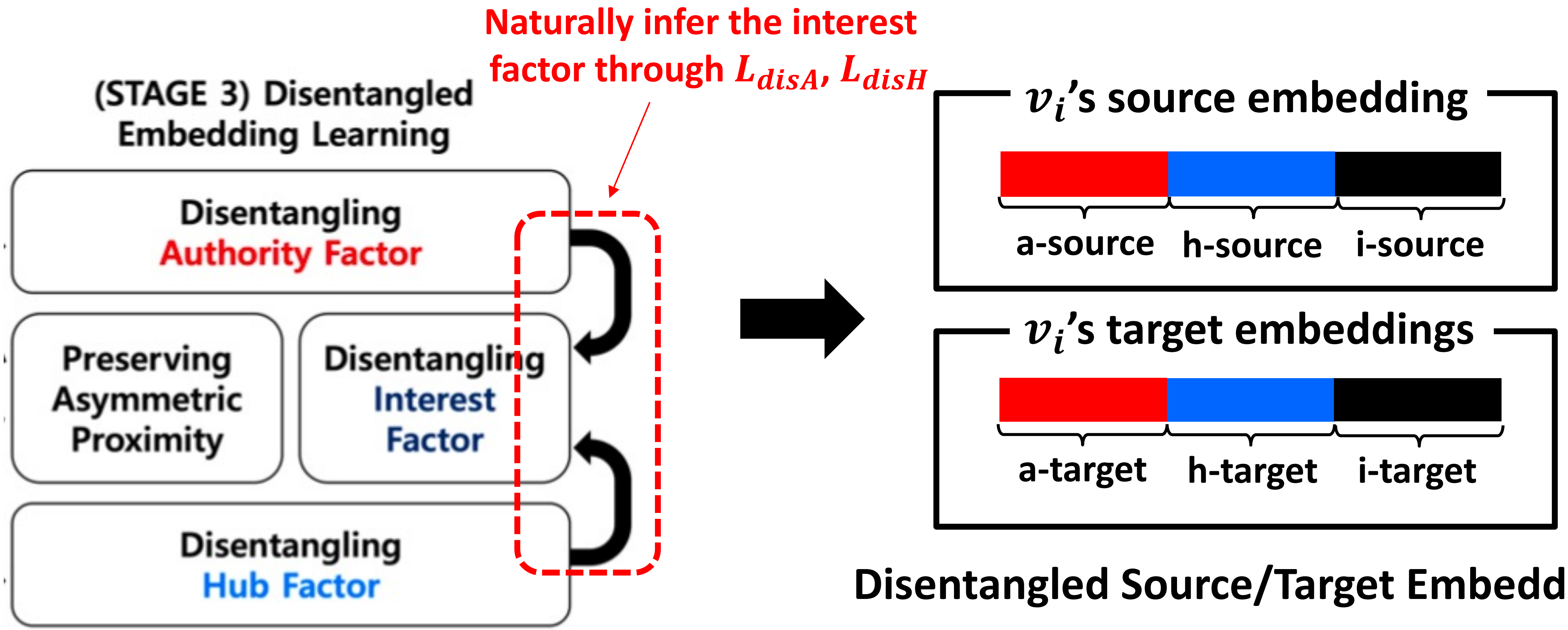
$$s_{ij}^{hub} > s_{i'j}^{hub}$$

For $(v_i, v_j, v_{i'})$ in $H_{<}$,

$$s_{ij}^{auth} + s_{ij}^{int} > s_{i'j}^{auth} + s_{i'j}^{int}$$

Final Embeddings

- ☐ Sub-embeddings capture **degree-related biases and interest separately**
- ☐ Thus, final embeddings are **robust to the shifts in degree distributions**



Disentangled Source/Target Embeddings

Experimental Settings

- Datasets

Datasets	GNU	Wiki	JUNG	Ciao
Nodes	6,301	7,115	6,120	4,658
Edges	20,777	103,689	50,535	40,133
Reciprocity	0.00%	5.64%	0.90%	34.90%
Types	P2P	Election	Software	Trust

- Nine competitors
  - 2 undirected NE methods
    - DeepWalk [KDD'14]
    - Node2Vec [KDD'16]
  - 7 directed NE methods
    - APP [AAAI'17]
    - ATP [AAAI'19]
    - NERD [ECML-PKDD'19]
    - GVAE [CIKM'19]
    - DiGCN [NeurIPS'20]
    - MagNet [NeurIPS'21]
    - DGGAN [AAAI'21]

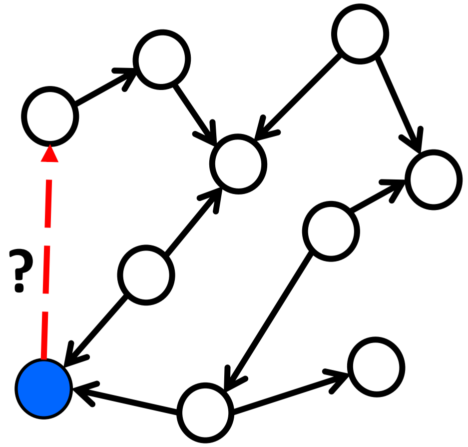
Non-ID Settings

- **Design two types of non-ID settings by splitting the edges in an input network into training and test sets with different degree distributions**
  1. Non-ID (in), where in-degree distributions are different
     - Each edge (v_i, v_j) is sampled into the test set with $p_{ij}^{in} \propto d_{in}(v_j)^k$
  2. Non-ID (out), where out-degree distributions are different
     - Each edge (v_i, v_j) is sampled into the test set with $p_{ij}^{out} \propto d_{out}(v_i)^k$

- **Control the level of distributional shifts by using k**
  - When k is 0, test edges are randomly sampled (i.e., ID setting)
  - When k is -1, test edges are sampled inversely proportional to the out-degree of the source nodes or in-degree of the target nodes (i.e., Shifts are strong)

Evaluation Task: Link Prediction (LP)

- **How accurately we can predict the directed edges removed from the input directed network?**



- **Evaluation protocol**
  - Consider the existent edges as positive examples
  - Perform two LP tasks, which depend on how we sample the negative examples
    - **Uniform LP** (U-LP): consider the non-existent edges sampled uniformly at random as negative examples
    - **Biased LP** (B-LP): consider the edges with the opposite directions to (unidirectional) positive examples as negative examples
  - Measure classification accuracy using *area under curve* (AUC)

Questions to Be Answered

- RQ1: Does ODIN outperform its competitors under distributional shifts in degree distributions?
- RQ2: How robust is ODIN under various levels of distributional shifts in degree distributions?
- RO3: Is factor disentanglement effective in ODIN?
- RQ4: How sensitive is ODIN to its hyperparameters?

Note: k is fixed to -1 for RQ1, RQ3, and RQ4 (in $p_{ij}^{in} \propto d_{in}(v_j)^k$ or $p_{ij}^{out} \propto d_{out}(v_i)^k$)

Results for RQ1

☐ Comparison with nine competitors

(a) Non-ID (in)

Datasets	Tasks	Undirected NE		Directed NE							ODIN
		DeepWalk	Node2Vec	APP	GVAE	NERD	ATP	DiGCN	MagNet	DGGAN	
GNU	U-LP	0.593±0.005	0.587±0.004	0.675±0.003	0.675±0.003	0.683±0.008	0.731±0.003	0.729±0.001	0.742±0.001	0.722±0.003	**0.760±0.004**
	B-LP	0.648±0.006	0.621±0.010	0.700±0.006	0.748±0.013	0.838±0.004	0.910±0.002	0.878±0.003	0.900±0.004	0.901±0.003	**0.924±0.001**
Wiki	U-LP	0.806±0.001	0.804±0.002	0.795±0.001	0.820±0.005	0.828±0.001	0.827±0.002	0.729±0.002	0.865±0.001	0.890±0.001	**0.905±0.001**
	B-LP	0.852±0.002	0.855±0.007	0.637±0.008	0.901±0.012	0.915±0.002	0.954±0.001	0.862±0.002	0.928±0.001	0.963±0.001	**0.973±0.001**
JUNG	U-LP	0.725±0.005	0.777±0.006	0.741±0.002	0.820±0.003	0.784±0.006	0.864±0.001	0.817±0.004	0.816±0.002	0.879±0.003	**0.884±0.002**
	B-LP	0.810±0.005	0.861±0.005	0.772±0.005	0.902±0.006	0.883±0.005	0.961±0.001	0.926±0.001	0.891±0.003	0.964±0.002	**0.969±0.001**
Ciao	U-LP	0.776±0.004	0.778±0.002	0.846±0.001	0.841±0.002	0.857±0.002	0.846±0.002	0.641±0.004	0.847±0.001	0.886±0.001	**0.892±0.001**
	B-LP	0.688±0.005	0.725±0.006	0.768±0.002	0.797±0.004	0.869±0.006	0.887±0.003	0.751±0.006	0.873±0.004	0.912±0.003	**0.914±0.003**

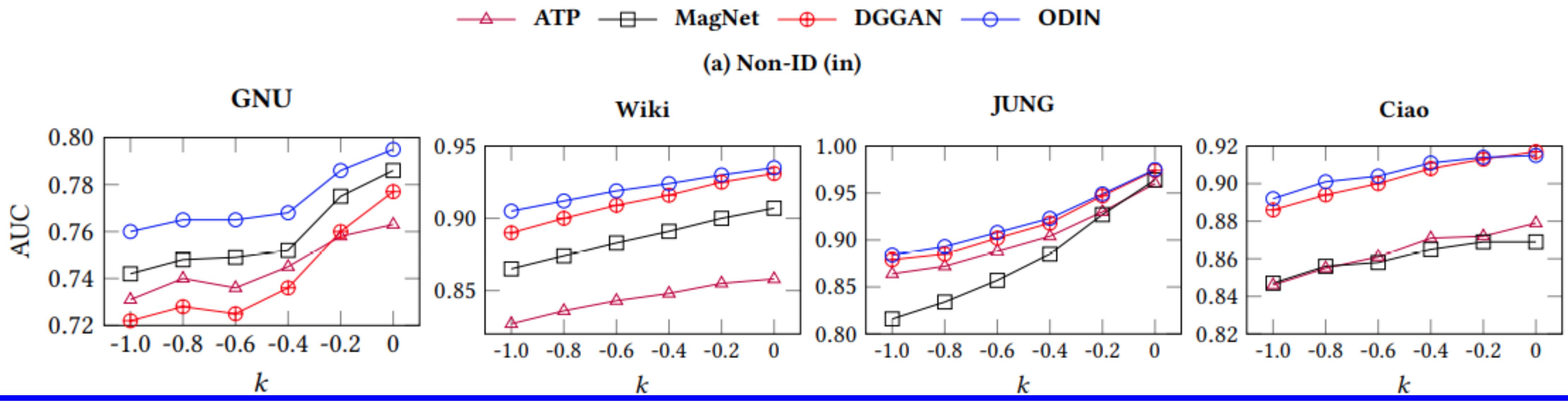
(b) Non-ID (out)

		Undirected NE		Directed NE							
	B-LP	0.635±0.011	0.695±0.004	0.597±0.004	0.684±0.008	0.750±0.004	0.867±0.003	0.836±0.003	0.827±0.003	0.871±0.002	**0.883±0.003**

- Best competitors change depending on tasks, datasets, and non-ID settings
- ODIN is effective compared to all the competitors in addressing the OOD generalization problem against degree-related distributional shifts on directed NE

Results for RQ2

☐ Effect of k on the link prediction performance



- In the ID setting (i.e., $k = 0$), the AUCs of ODIN are comparable to or higher than that of the strongest competitors
- ODIN shows the smallest accuracy degradation and, accordingly, the accuracy gain of ODIN against competitors steadily increases
- The results indicate that ODIN obtains OOD generalized embeddings robust to degree-related distributional shifts

Results for RQ3-1

☐ RQ3-1: Is each of two disentanglement losses effective in ODIN?

■ ODIN_A vs ODIN_{dis_A}

Datasets	Tasks	ODIN_A	ODIN_{disA}
GNU	U-LP	0.632±0.005	**0.763±0.004**
	B-LP	0.704±0.010	**0.927±0.001**
Wiki	U-LP	0.842±0.002	0.896±0.001
	B-LP	0.918±0.001	0.965±0.001
JUNG	U-LP	0.825±0.004	0.878±0.003
	B-LP	0.929±0.003	0.966±0.002
Ciao	U-LP	0.820±0.003	0.890±0.001
	B-LP	0.788±0.009	0.912±0.002

ODIN_A: only uses the *edge loss based on A*

ODIN_{disA}: additionally uses the *disA loss based on A*

■ ODIN_H vs ODIN_{dis_H}

Datasets	Tasks	ODIN_H	ODIN_{disH}
GNU	U-LP	0.604±0.010	0.678±0.001
	B-LP	0.669±0.015	0.820±0.010
Wiki	U-LP	0.793±0.007	0.898±0.001
	B-LP	0.863±0.011	0.968±0.001
JUNG	U-LP	0.714±0.006	**0.884±0.002**
	B-LP	0.830±0.004	**0.970±0.001**
Ciao	U-LP	0.853±0.001	0.886±0.001
	B-LP	0.867±0.005	0.909±0.002

ODIN_H: only uses the *edge loss based on H*

ODIN_{disH}: additionally uses the *disH loss based on H*

■ Each of the disentanglement losses is effective in obtaining embeddings robust to distributional shifts in degree distributions

Results for RQ3-2

RQ1-2: Is jointly using the both losses effective in ODIN?

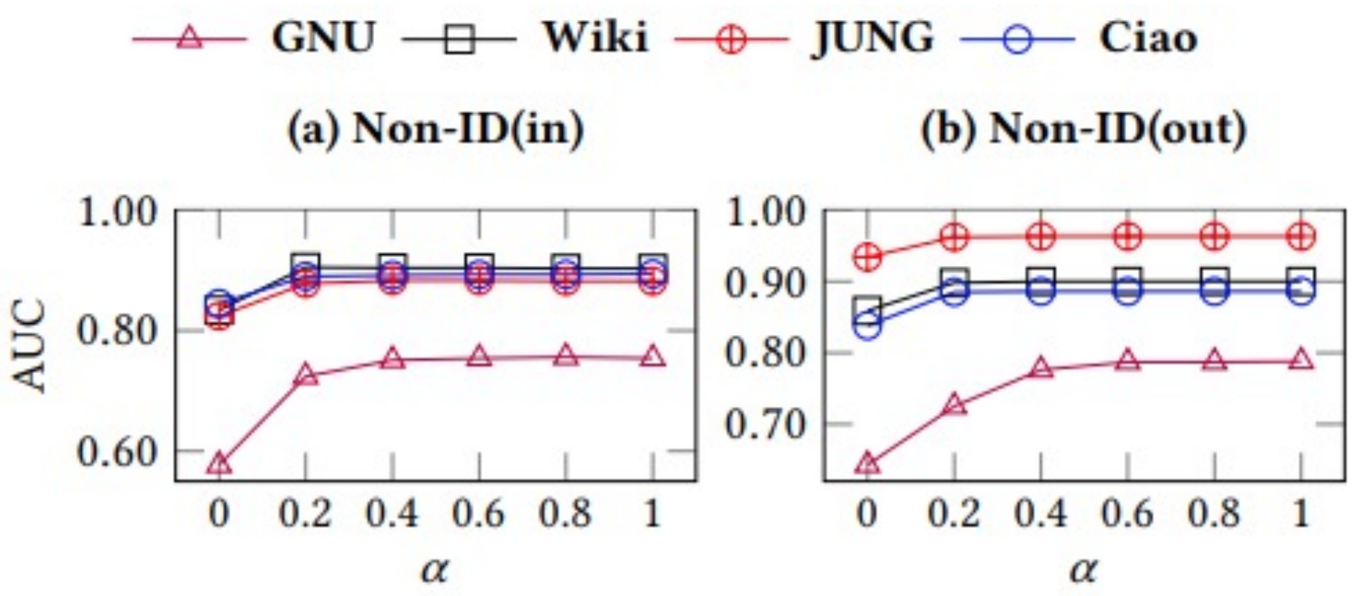
(a) Non-ID (in)

Datasets	Tasks	$ODIN_A$	$ODIN_{disA}$	$ODIN_H$	$ODIN_{disH}$	ODIN
GNU	U-LP	0.632±0.005	**0.763±0.004**	0.604±0.010	0.678±0.001	0.760±0.004
	B-LP	0.704±0.010	**0.927±0.001**	0.669±0.015	0.820±0.010	0.924±0.001
Wiki	U-LP	0.842±0.002	0.896±0.001	0.793±0.007	0.898±0.001	**0.905±0.001**
	B-LP	0.918±0.001	0.965±0.001	0.863±0.011	0.968±0.001	**0.973±0.001**
JUNG	U-LP	0.825±0.004	0.878±0.003	0.714±0.006	**0.884±0.002**	**0.884±0.002**
	B-LP	0.929±0.003	0.966±0.002	0.830±0.004	**0.970±0.001**	0.969±0.001
Ciao	U-LP	0.820±0.003	0.890±0.001	0.853±0.001	0.886±0.001	**0.892±0.001**
	B-LP	0.788±0.009	0.912±0.002	0.867±0.005	0.909±0.002	**0.914±0.003**

- Superiority between ODIN_{disA} and ODIN_{disH} varies depending on datasets
- ODIN outperforms ODIN_{disA} and ODIN_{disH} in most cases
  - That is, ODIN can selectively adopt the factor(s) beneficial in each dataset, thereby improving the robustness of embeddings in all datasets

Results for RQ4

☐ How the parameter α affects the accuracy of ODIN



$$L = L_{edge}(A \cup B)$$
$$+\boxed{\alpha}(L_{disA}(A) + L_{disH}(H))$$

- AUCs of ODIN steadily increase until α reaches 0.4 and then the AUCs converge
- ODIN is not highly sensitive to the weight for factor disentanglement

Conclusions

- We pointed out that the existing directed NE methods face difficulties in effectively addressing the OOD generalization problem
- We proposed ODIN, which models multiple factors in the formation of directed edges and learns nodes’ multiple factor sub-embeddings
- Through extensive experiments, we showed clearly the effectiveness of our strategies for factor modeling and disentangled embedding learning