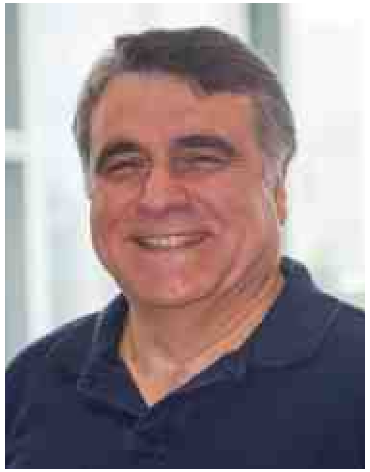
Carnegie Mellon University
Computer Science Department

amazon

Estimating Node Importance in Knowledge Graphs Using Graph Neural Networks

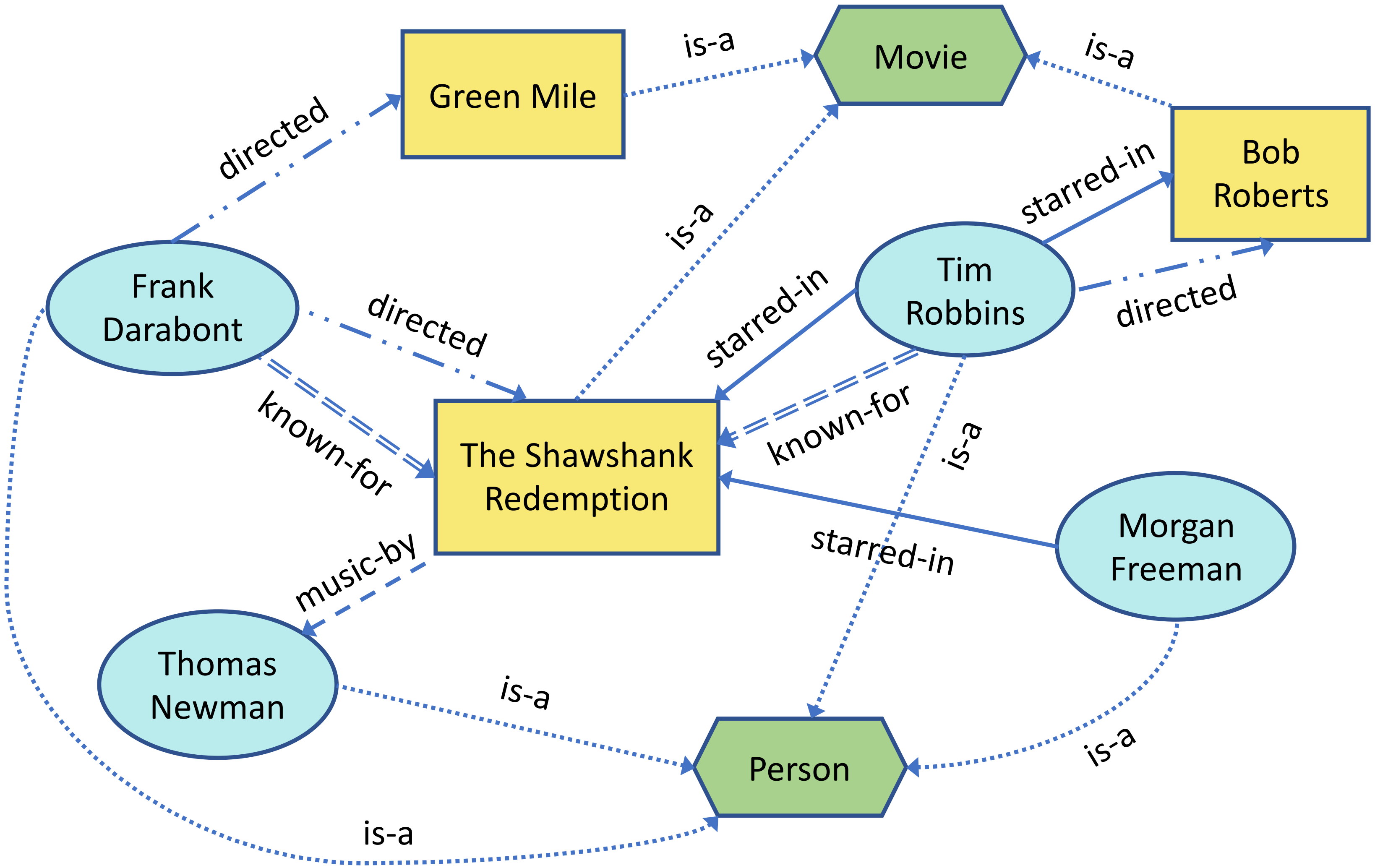
Namyong Park[1,2], Andrey Kan[2], Xin Luna Dong[2], Tong Zhao[2], Christos Faloutsos[1,2]
[1]Carnegie Mellon University, [2]Amazon

Roadmap

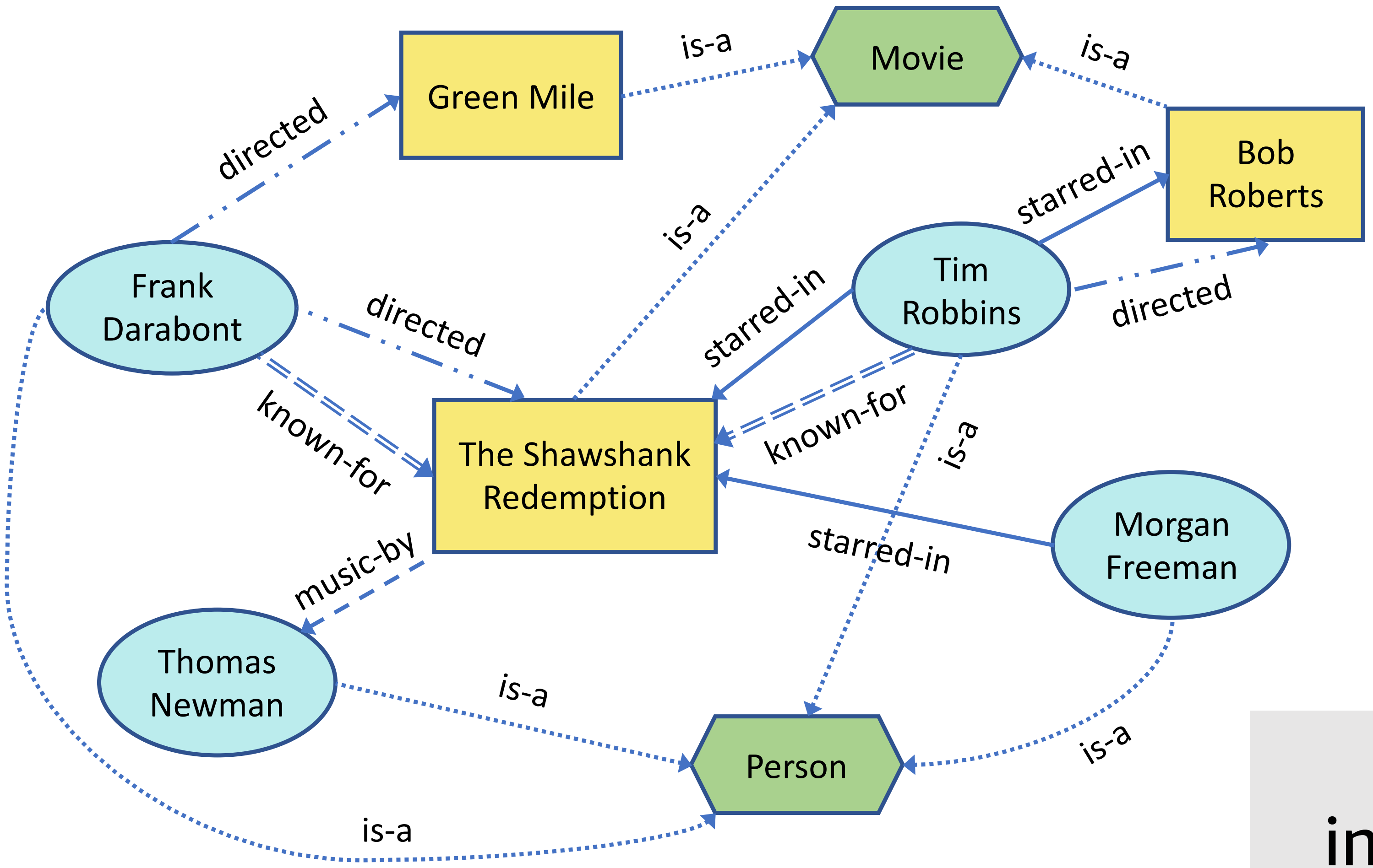
- **Introduction**
  - Knowledge Graph
  - Problem Definition
  - Existing Methods
- Proposed Method: GENI
- Experimental Results
- Conclusion

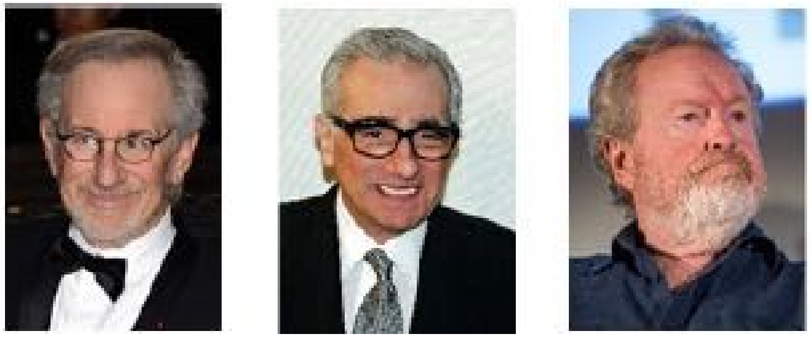
Knowledge Graph



An example knowledge graph on movies and related entities

Node Importance





Most important directors

What are the most important nodes in a KG?

Applications

Query disambiguation

"Beethoven"

Applications

- Search
- Information extraction

- Quality control for KGs

* Image source: www.freepik.com

Importance Score

- Often we can observe a signal that indicates node importance
- Examples

Total gross

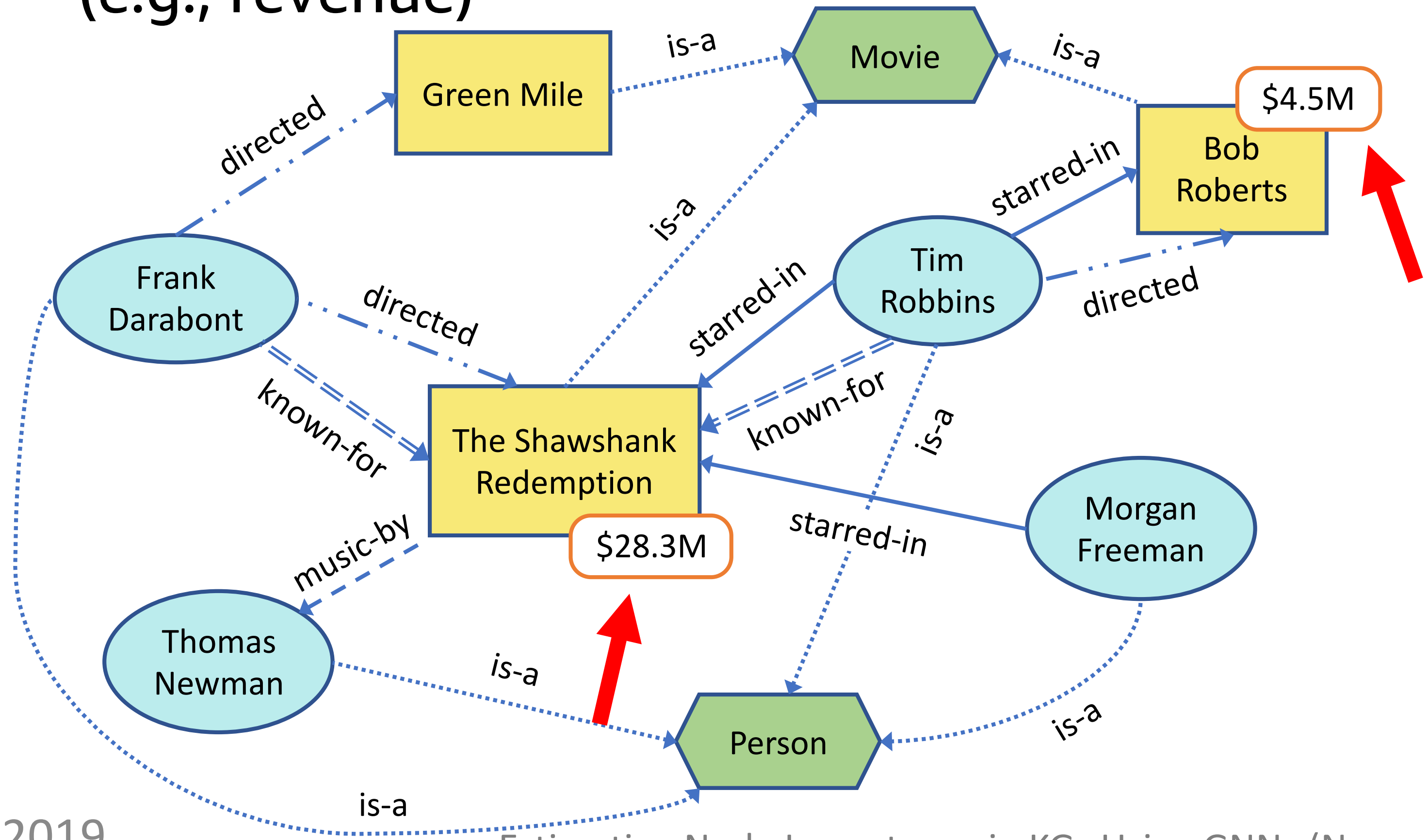
Number of votes

Number of pageviews

Node Importance Estimation

Input

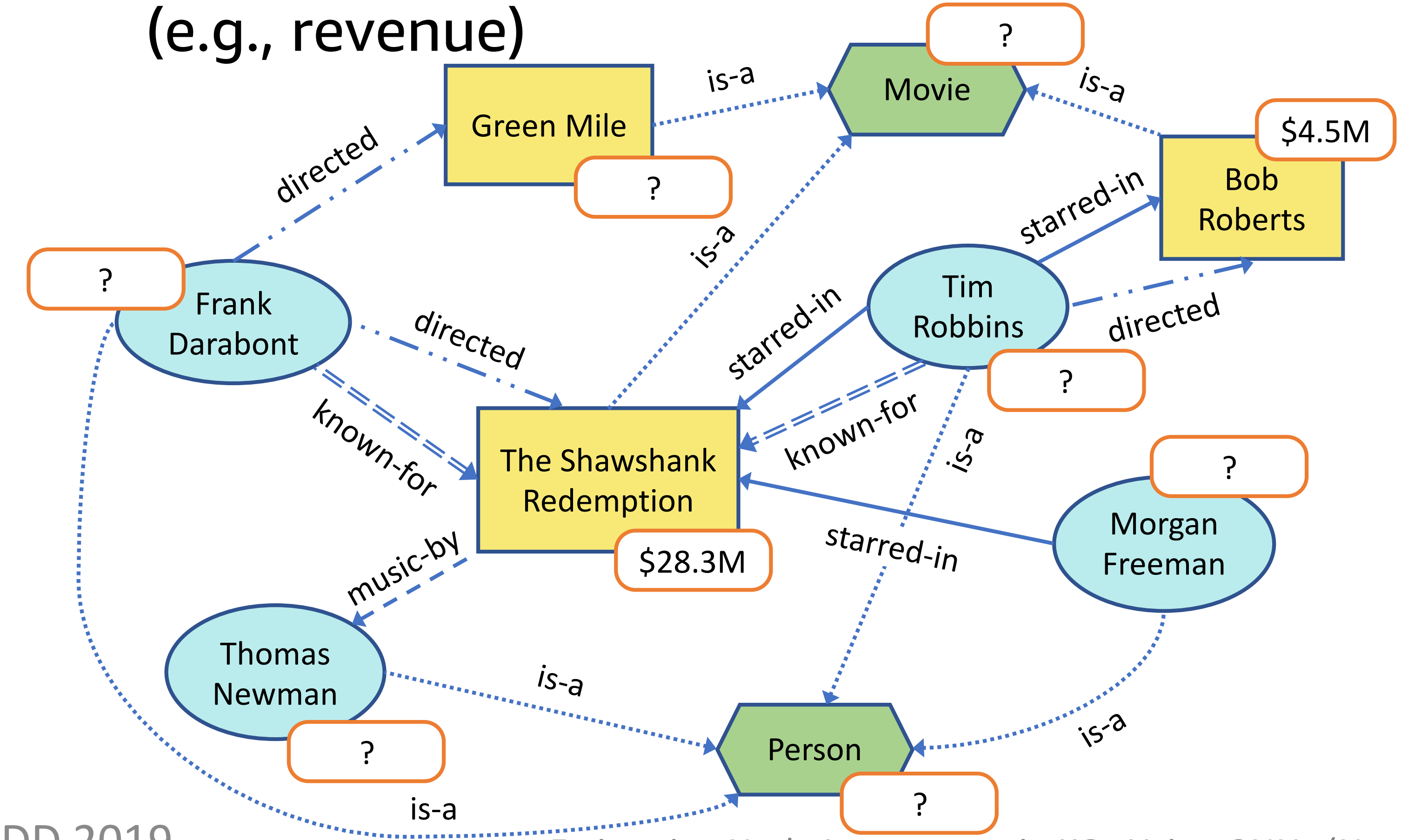
- A knowledge graph
- Input scores for some nodes
  (e.g., revenue)



Node Importance Estimation

Input

- A knowledge graph
- Input scores for some nodes (e.g., revenue)



Output

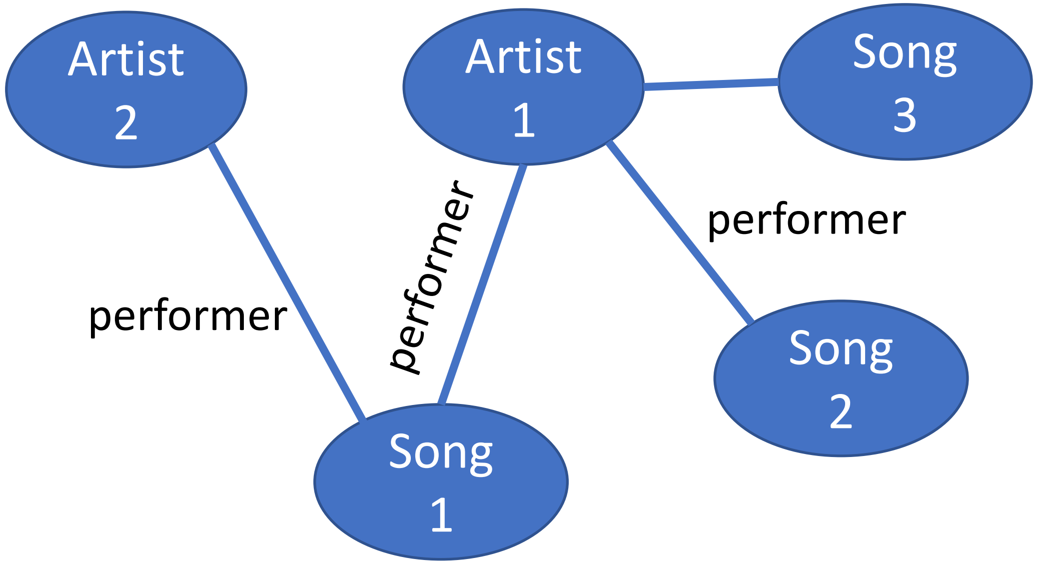
An importance score for each node

- Non-negative real value
- Reflects the popularity of a node
- Closely reconstructs input scores

Node Importance Estimation: Intuition

- Which artist is more important?
  - Artist 1? Artist 2?

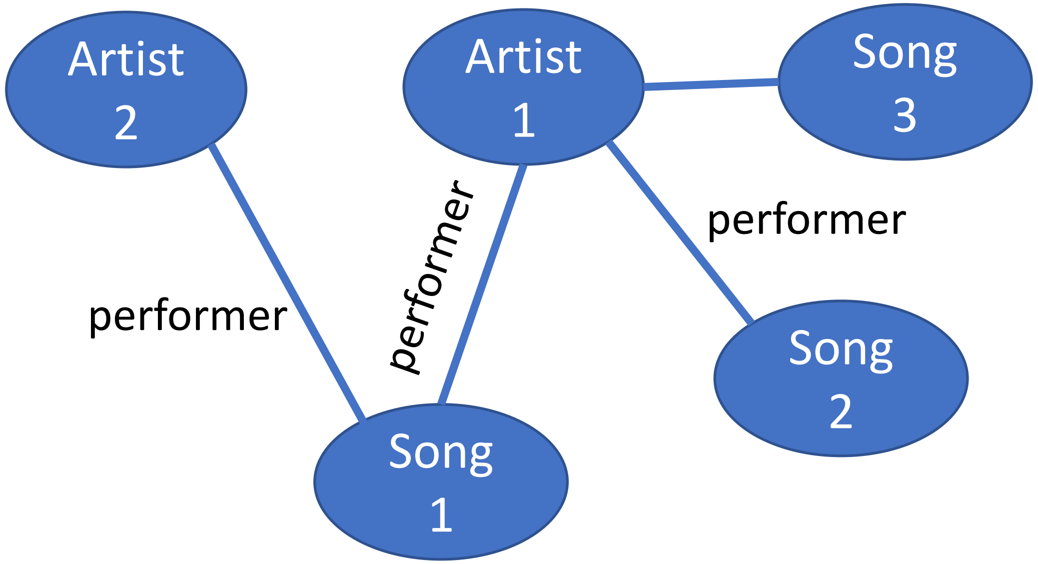
Requirements
Neighborhood Awareness



Node Importance Estimation: Intuition

- Which artist is more important?
  - Artist 1? Artist 2?

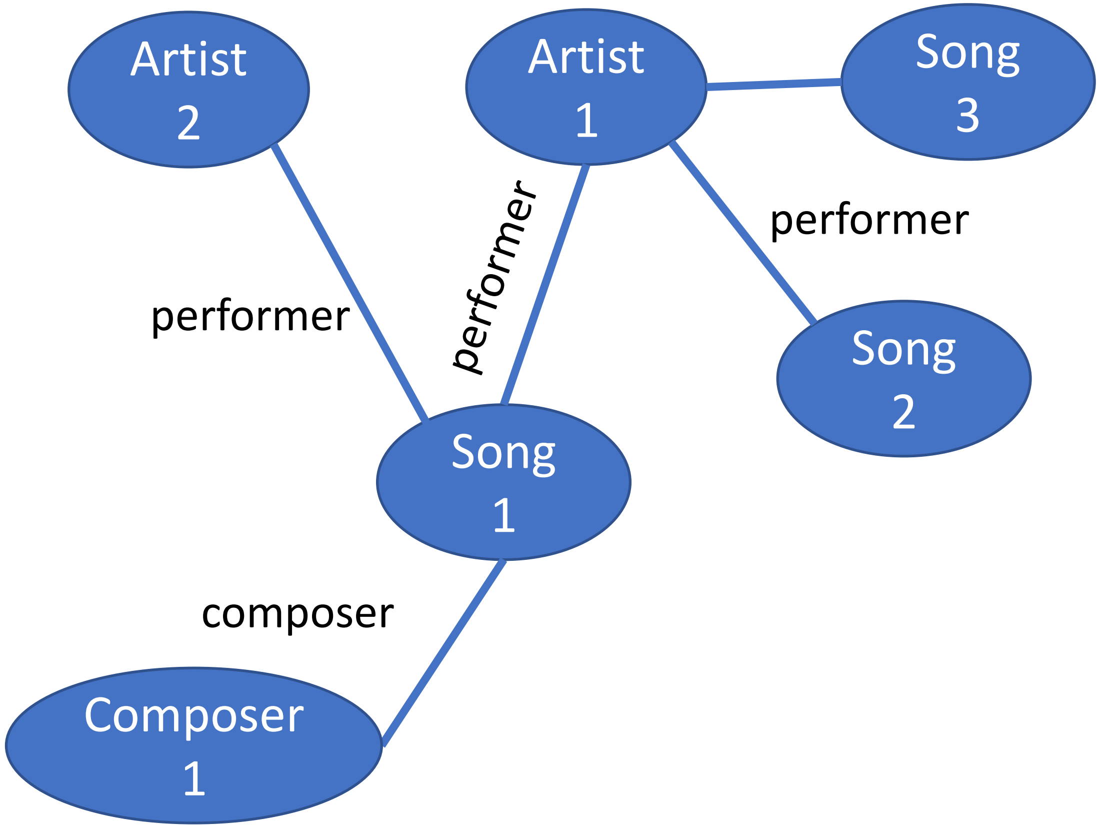
Requirements
Neighborhood Awareness
Centrality Awareness



Node Importance Estimation: Intuition

- What if there is a different predicate, e.g., a composer?

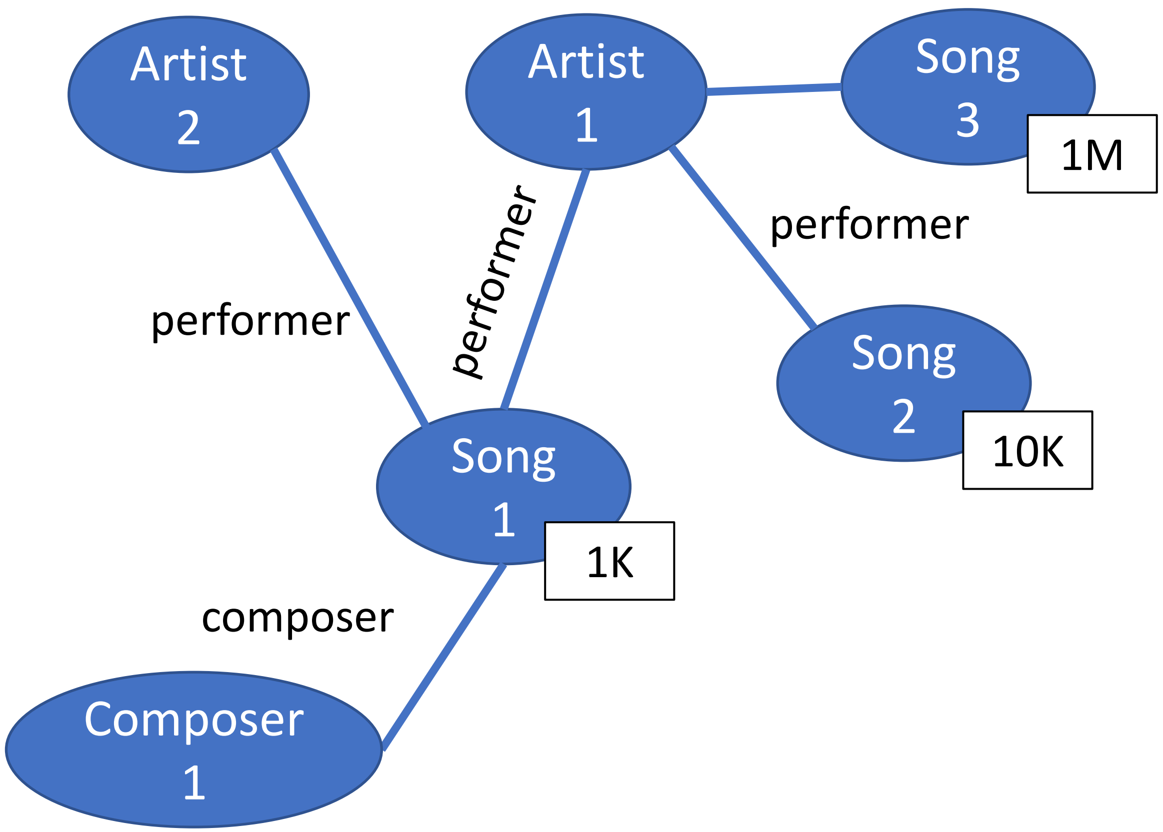
Requirements
Neighborhood Awareness
Centrality Awareness
Edge Type Awareness



Node Importance Estimation: Intuition

- We have access to importance scores for some nodes

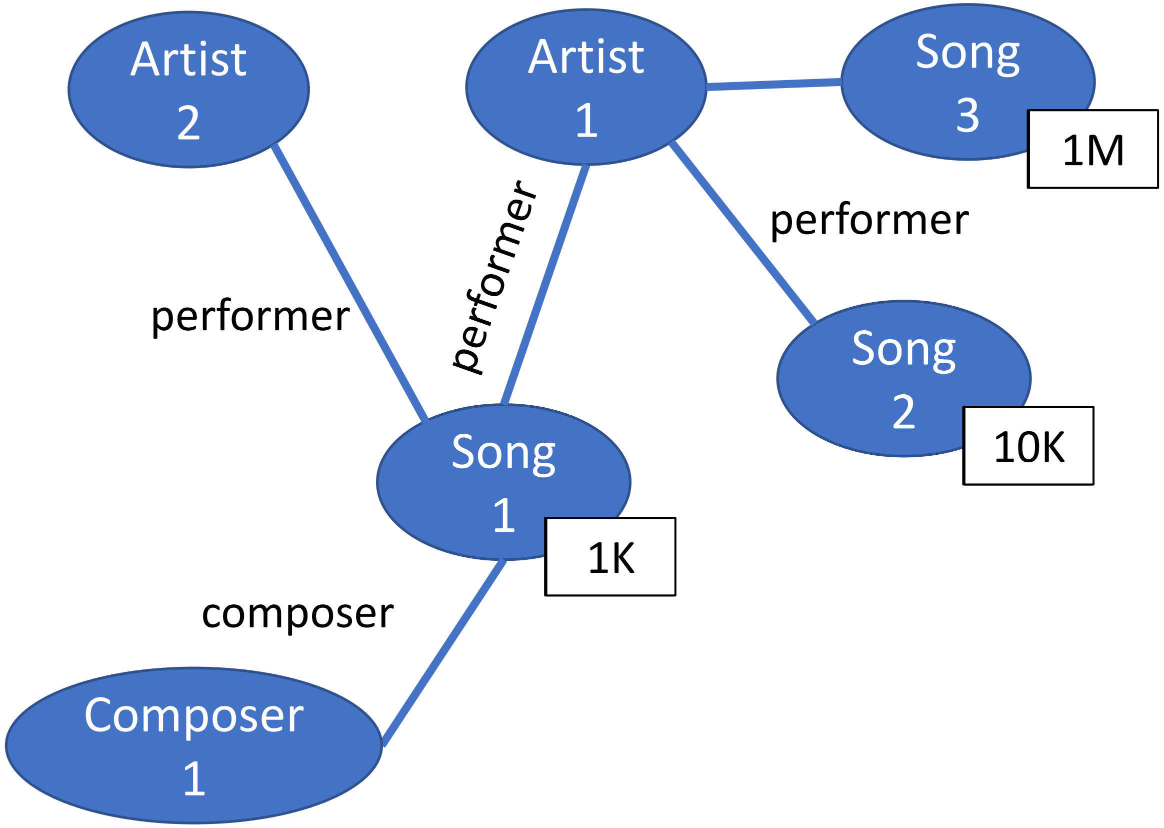
Requirements
Neighborhood Awareness
Centrality Awareness
Edge Type Awareness
Input Score Awareness



Node Importance Estimation: Intuition

- What if the score distribution changes?

Requirements
Neighborhood Awareness
Centrality Awareness
Edge Type Awareness
Input Score Awareness
Flexible Adaptation



Not a “Solved Problem”

Requirements	PageRank	Personalized PageRank	HAR
Neighborhood Awareness	✓	✓	✓
Centrality Awareness	✓	✓	✓
Input Score Awareness		✓	✓
Edge Type Awareness			✓
Flexible Adaptation			

Our Contributions

We explore supervised machine learning algorithms for this task

We present GENI, a GNN-based method

We provide empirical evidence and analysis of GENI on real-world KGs

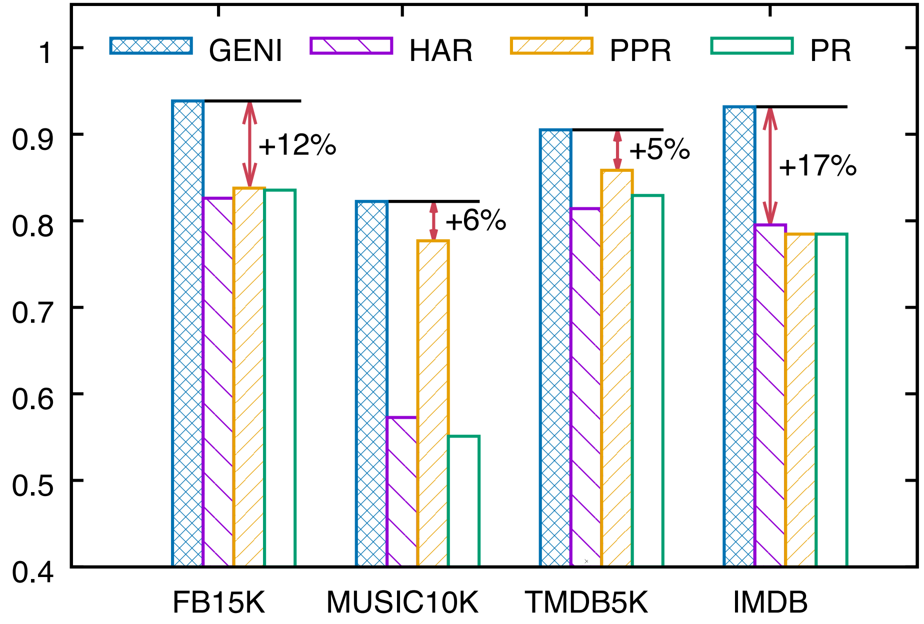
Linear regression

Random Forests

Neural Networks

Graph Neural Networks

Node Importance Estimation



Roadmap

- Introduction
- **Proposed Method: GENI**
  - Main Ideas
  - Model Architecture
- Experimental Results
- Conclusion

Proposed Method: GENI

- We propose GENI, a **G**raph neural network (GNN) for **E**stimating **N**ode **I**mportance in a KG

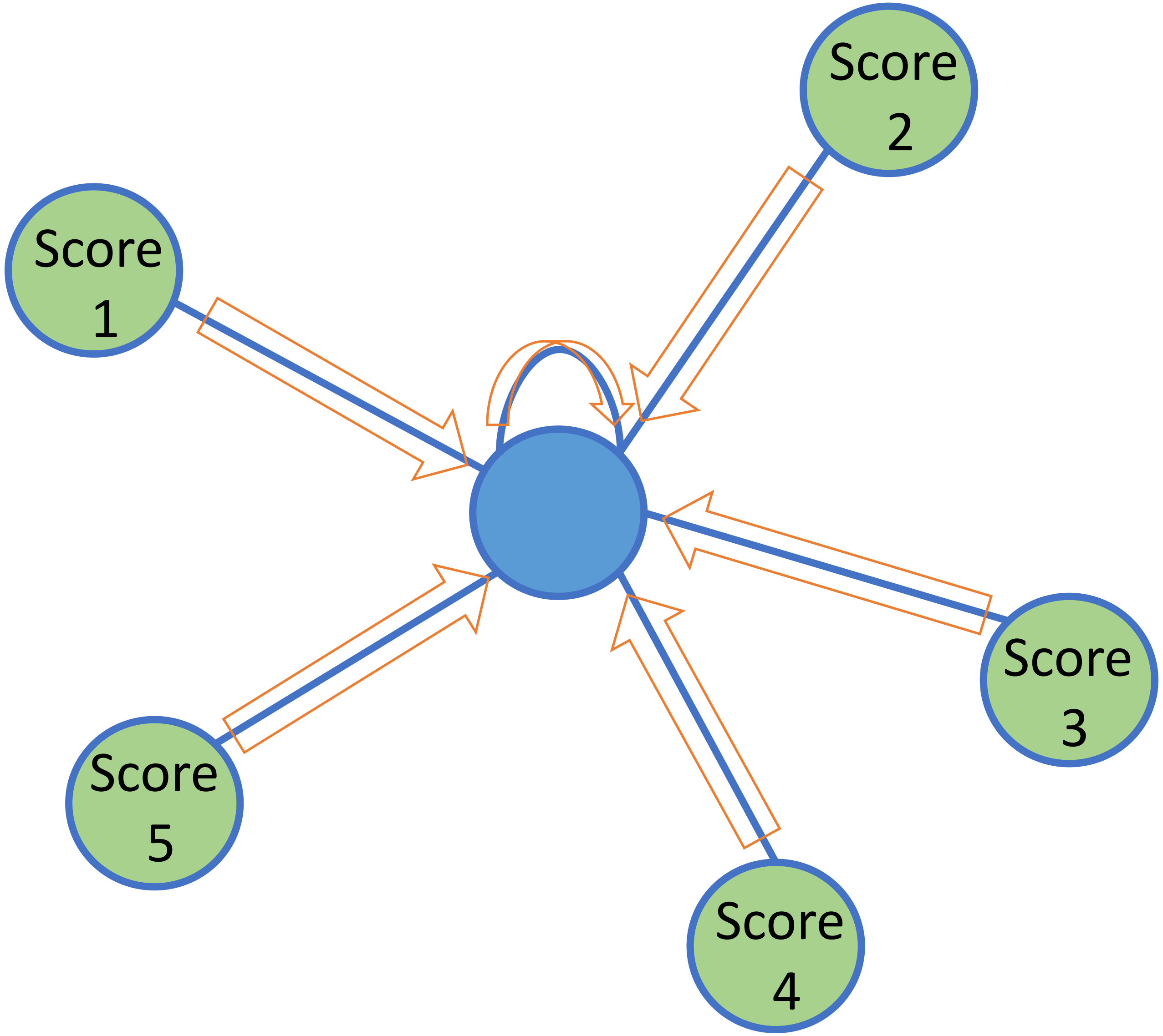
Requirements	PageRank	Personalized PageRank	HAR	GENI
Neighborhood Awareness	✓	✓	✓	✓
Centrality Awareness	✓	✓	✓	✓
Input Score Awareness		✓	✓	✓
Edge Type Awareness			✓	✓
Flexible Adaptation				✓

Proposed Method: GENI

- We propose GENI, a **G**raph neural network (GNN) for **E**stimating **N**ode **I**mportance in a KG

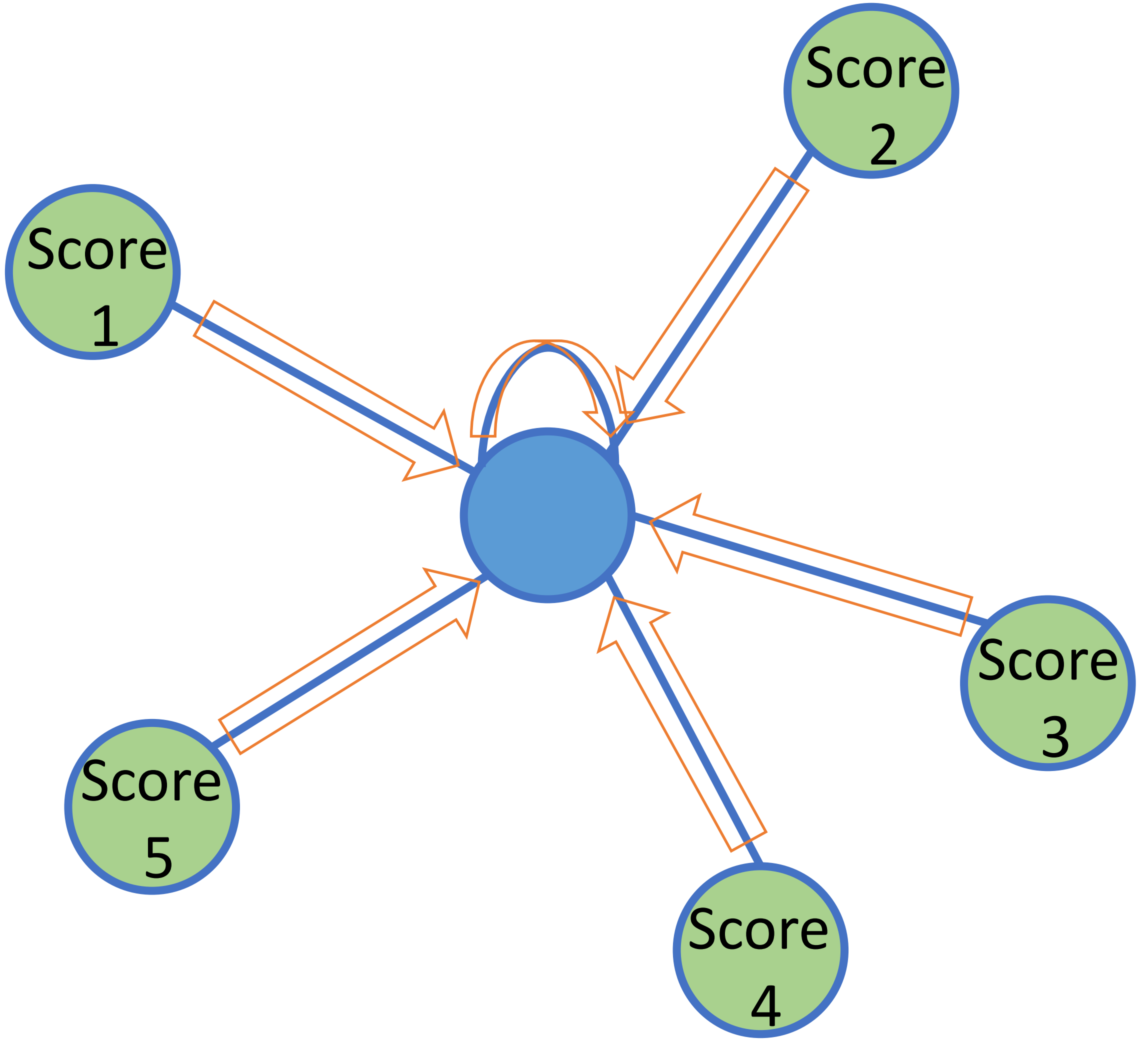
Requirements	Our Solution
Neighborhood Awareness	Score Aggregation
Edge Type Awareness	Predicate-Aware Attention
Centrality Awareness	Centrality Adjustment
Input Score Awareness	Supervised GNN framework
Flexible Adaptation	

Idea 1: Score Aggregation



Detail

Idea 1: Score Aggregation

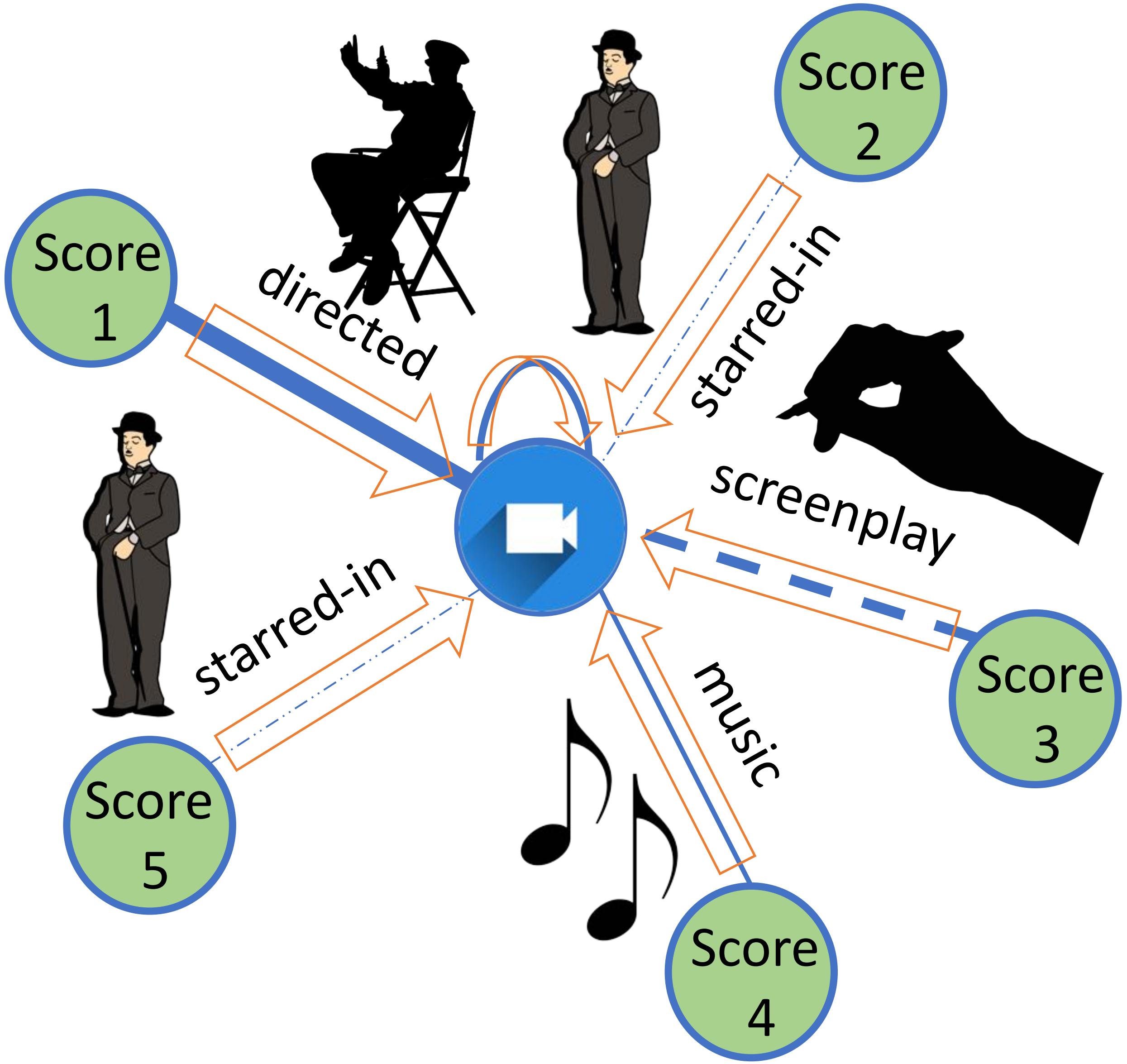


Score of a neighbor

$$s^{\ell}(i) = \sigma_s\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^{\ell} s^{\ell-1}(j)\right)$$

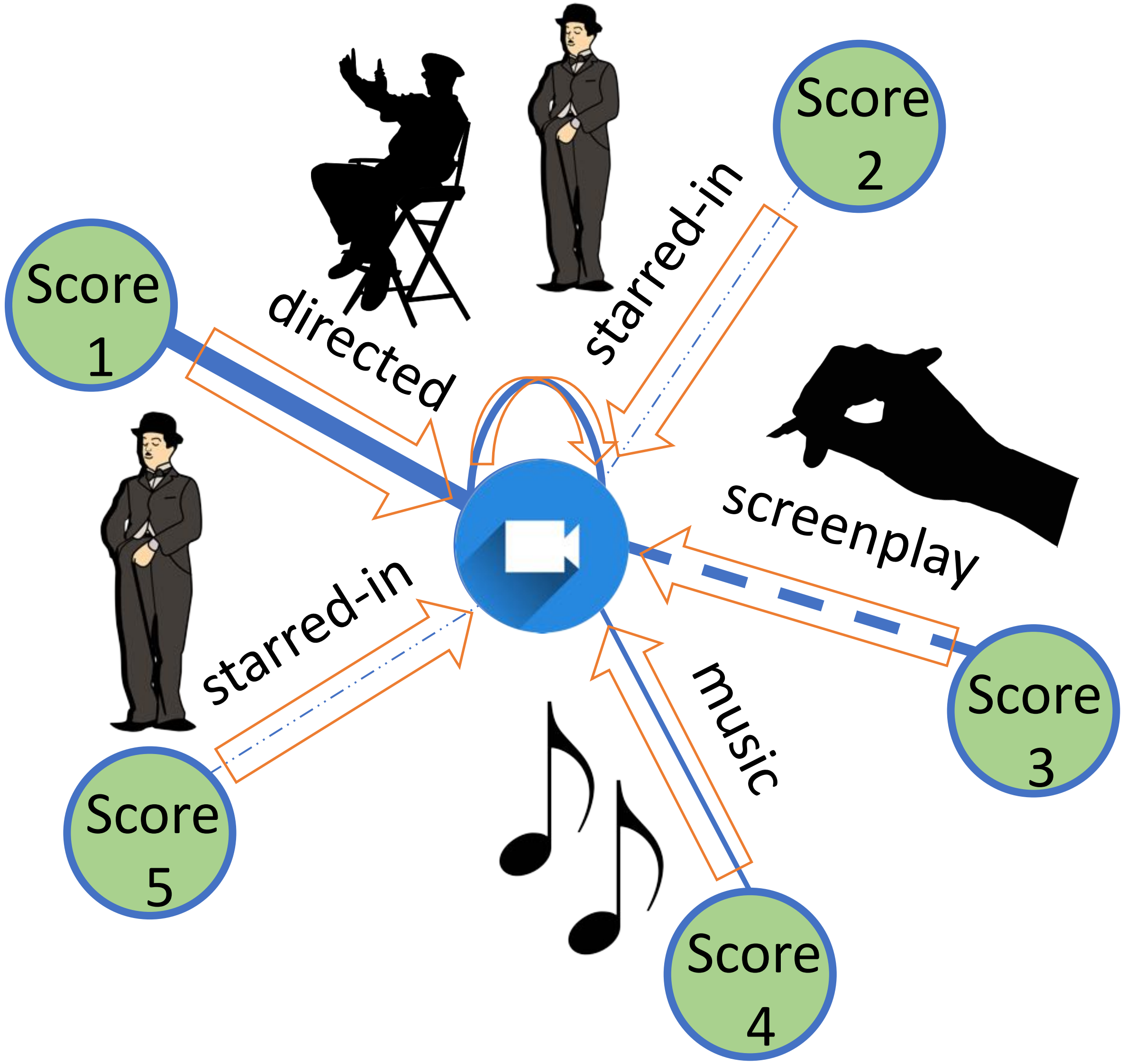
- ℓ: layer number
- $s^{\ell}(i)$: estimated score of node i
- $\mathcal{N}(i)$: neighbors of node
- α_{ij}^{ℓ}: node i's attention on node j

Idea 2: Predicate-Aware Attention



Idea 2: Predicate-Aware Attention

Detail



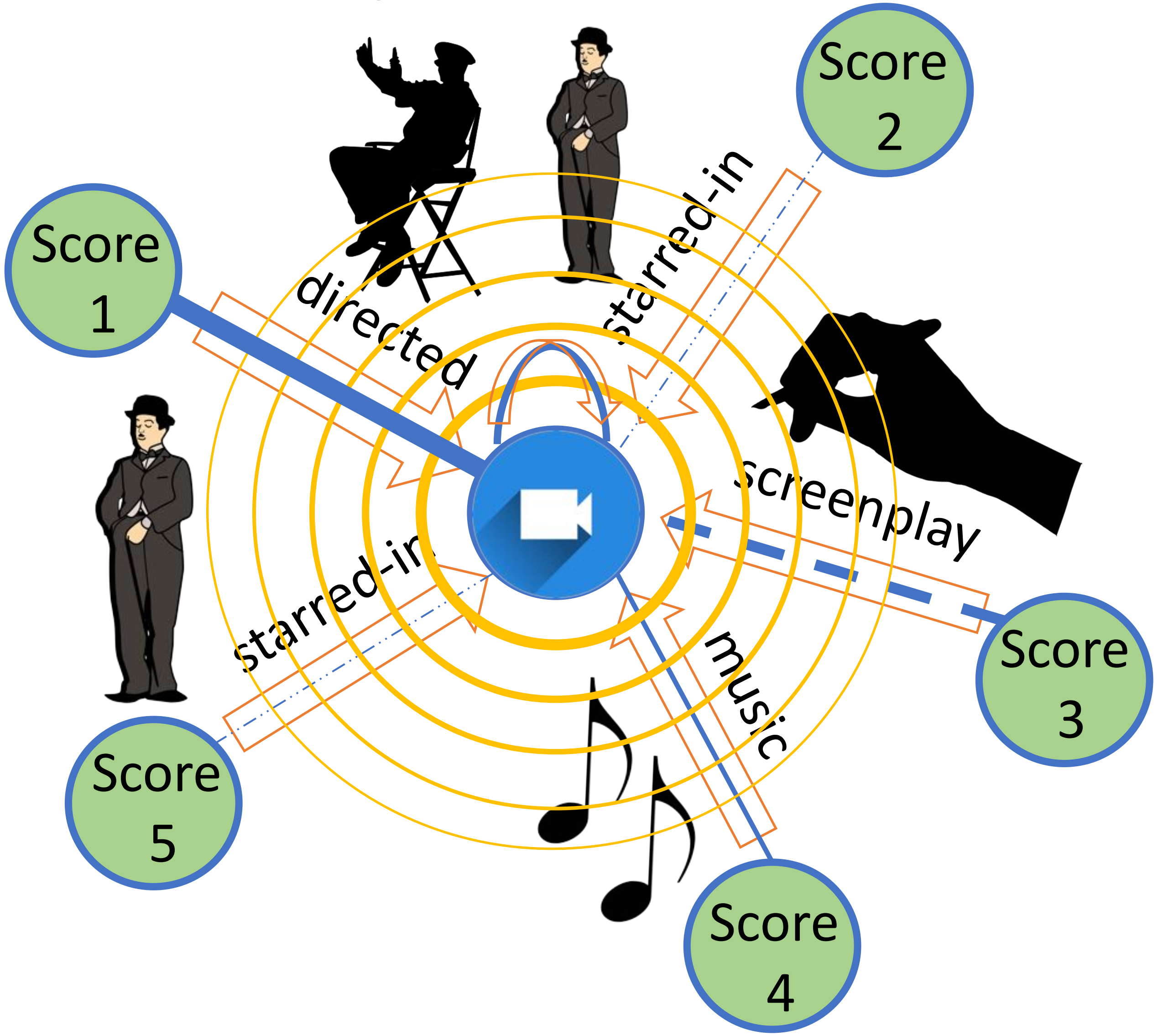
Attention considers predicate

$$s^{\ell}(i) = \sigma_s\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^{\ell} s^{\ell-1}(j)\right)$$

$$\alpha_{ij}^{\ell} = f\left(s^{\ell-1}(i), s^{\ell-1}(j), \vec{a}_{\ell}, \vec{p}_{ij}\right)$$

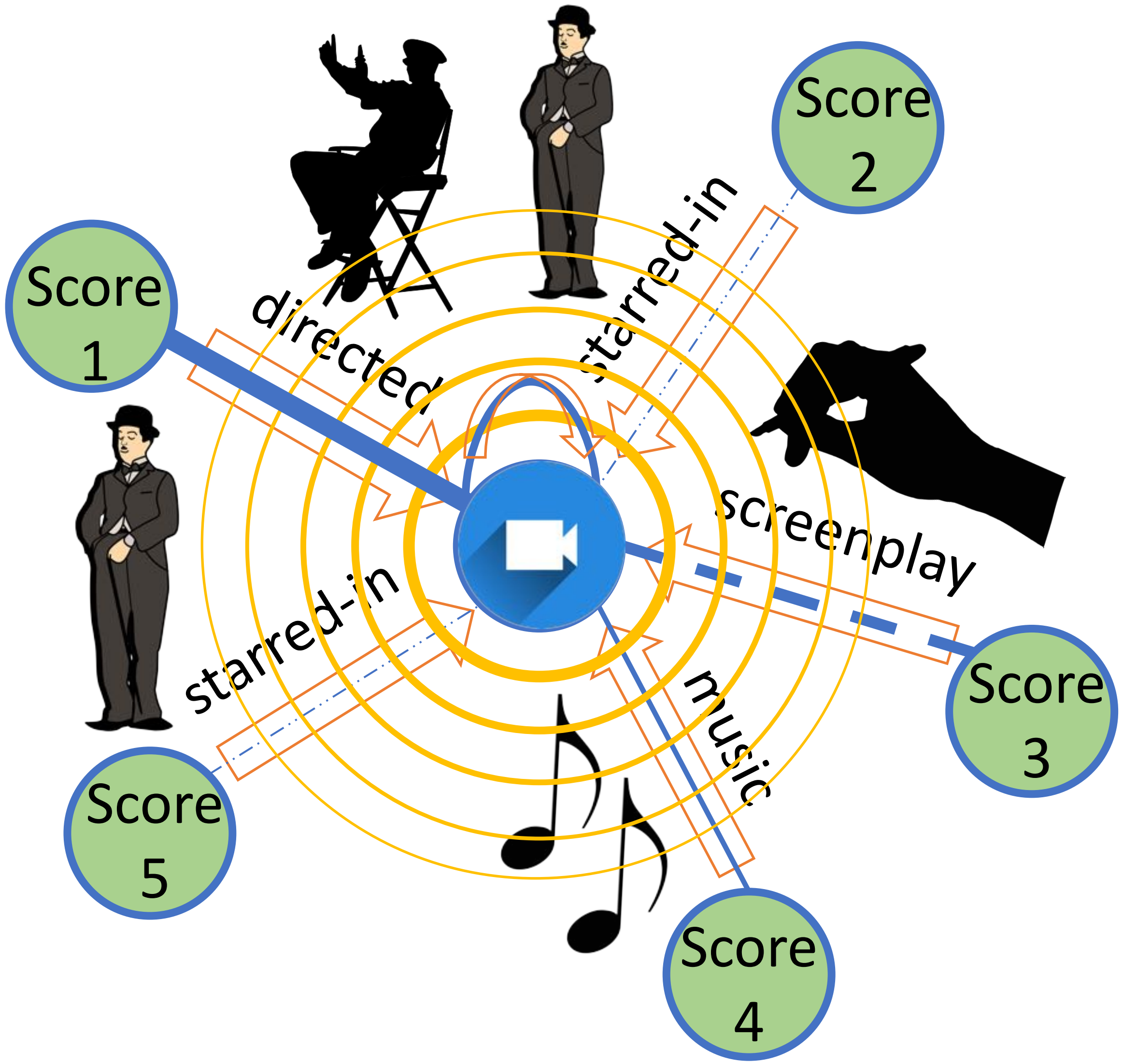
- α_{ij}^{ℓ}: node i's attention on node j computed by the ℓ-th layer
- $s^{\ell}(i)$: estimated score of node i
- $\vec{p}_{ij}$: predicates of between nodes i and j
- $\vec{a}_{\ell}$: attention parameters

Idea 3: Centrality Adjustment



Idea 3: Centrality Adjustment

Detail



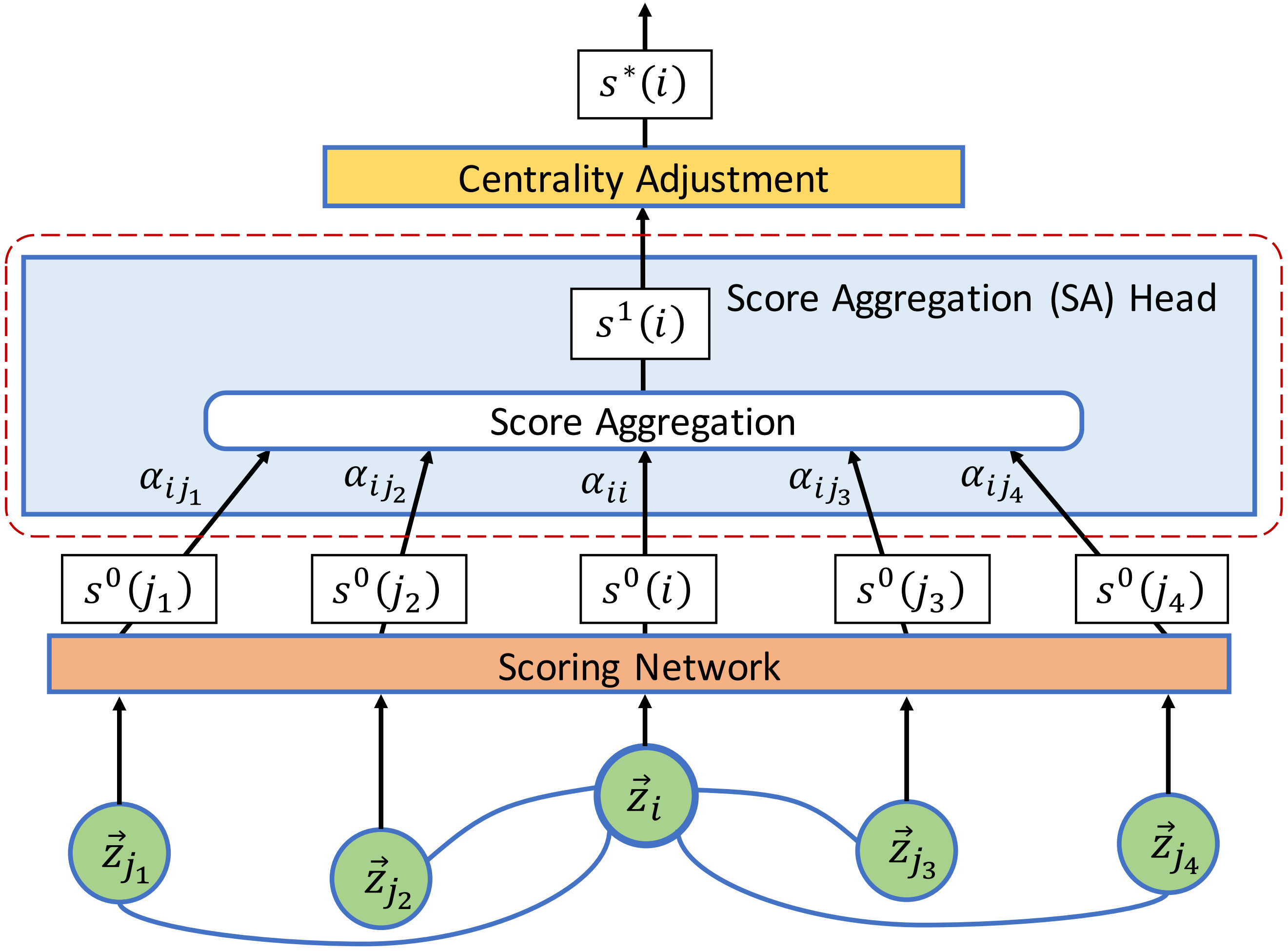
In-degree of a node

$$s^*(i) = \text{CentralityAdjustment}(s^L(i), d(i))$$

Estimated score before centrality adjustment

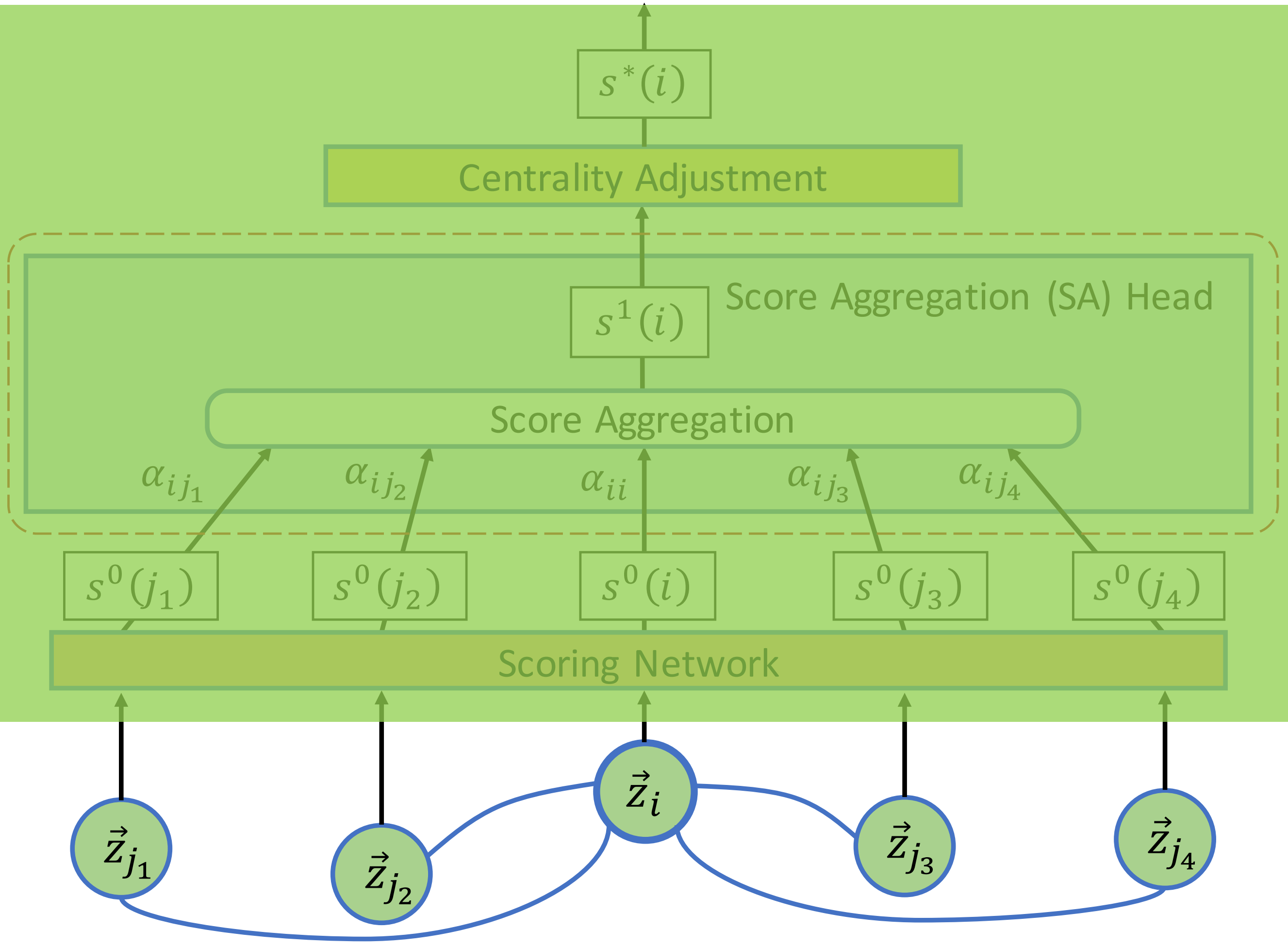
- $s^*(i)$: centrality-adjusted score estimation of node i
- $s^L(i)$: estimated score of node i before centrality adjustment
- $d(i)$: in-degree of node i
- L: final layer

Model Architecture: One Layer, One Head



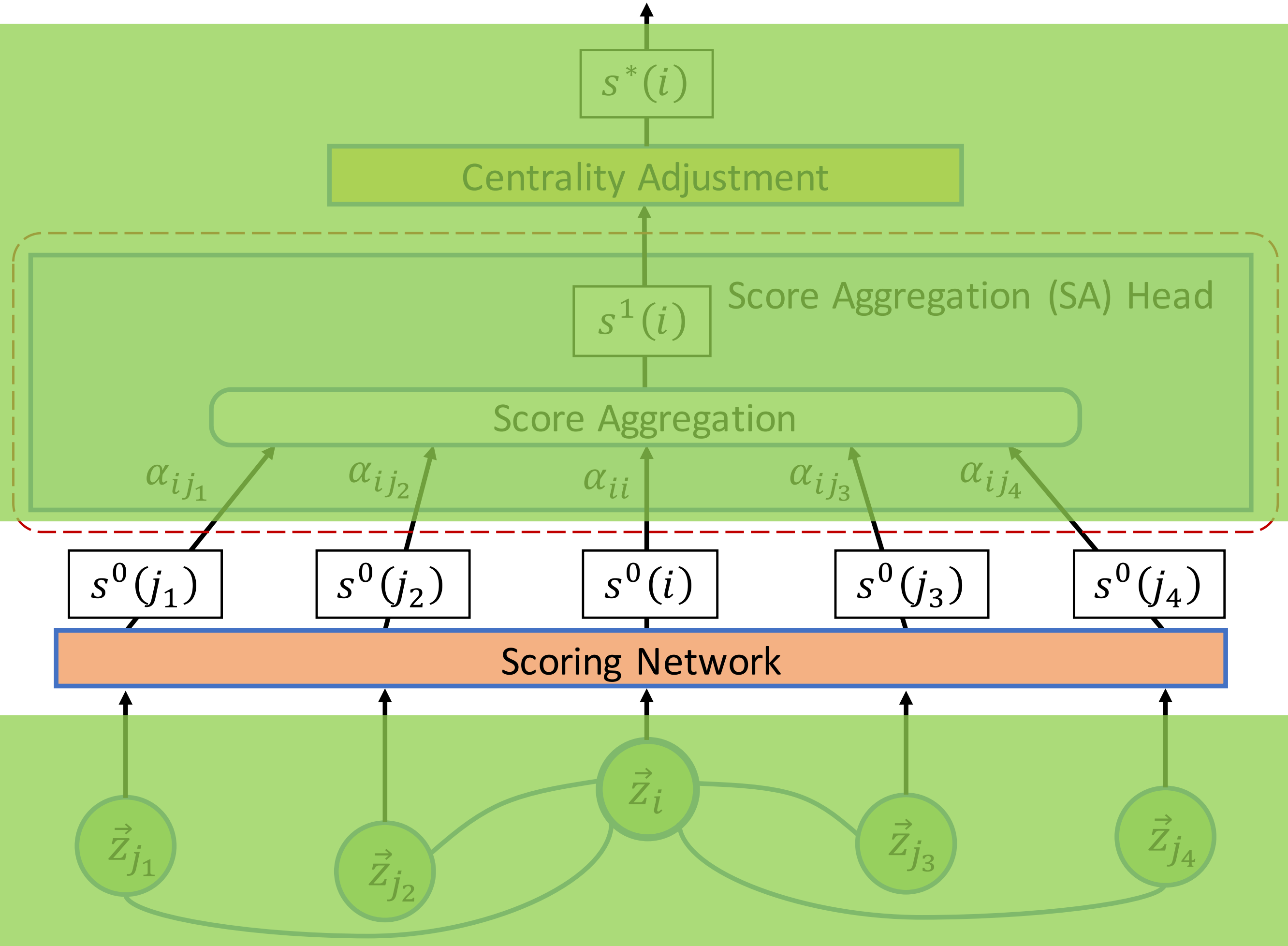
- $s^*(i) = \mathrm{Centr.Adj.}(s^1(i), d(i))$
- $s^1(i) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^1 s^0(j)\right)$
- $\alpha_{ij}^1 = f\left(s^0(i), s^0(j), \vec{a}_1, \vec{p}_{ij}\right)$
- $s^0(i) = \mathrm{ScoringNetwork}(\vec{z}_i)$
- $\vec{z}_i$: feature vector of node i

Model Architecture: One Layer, One Head



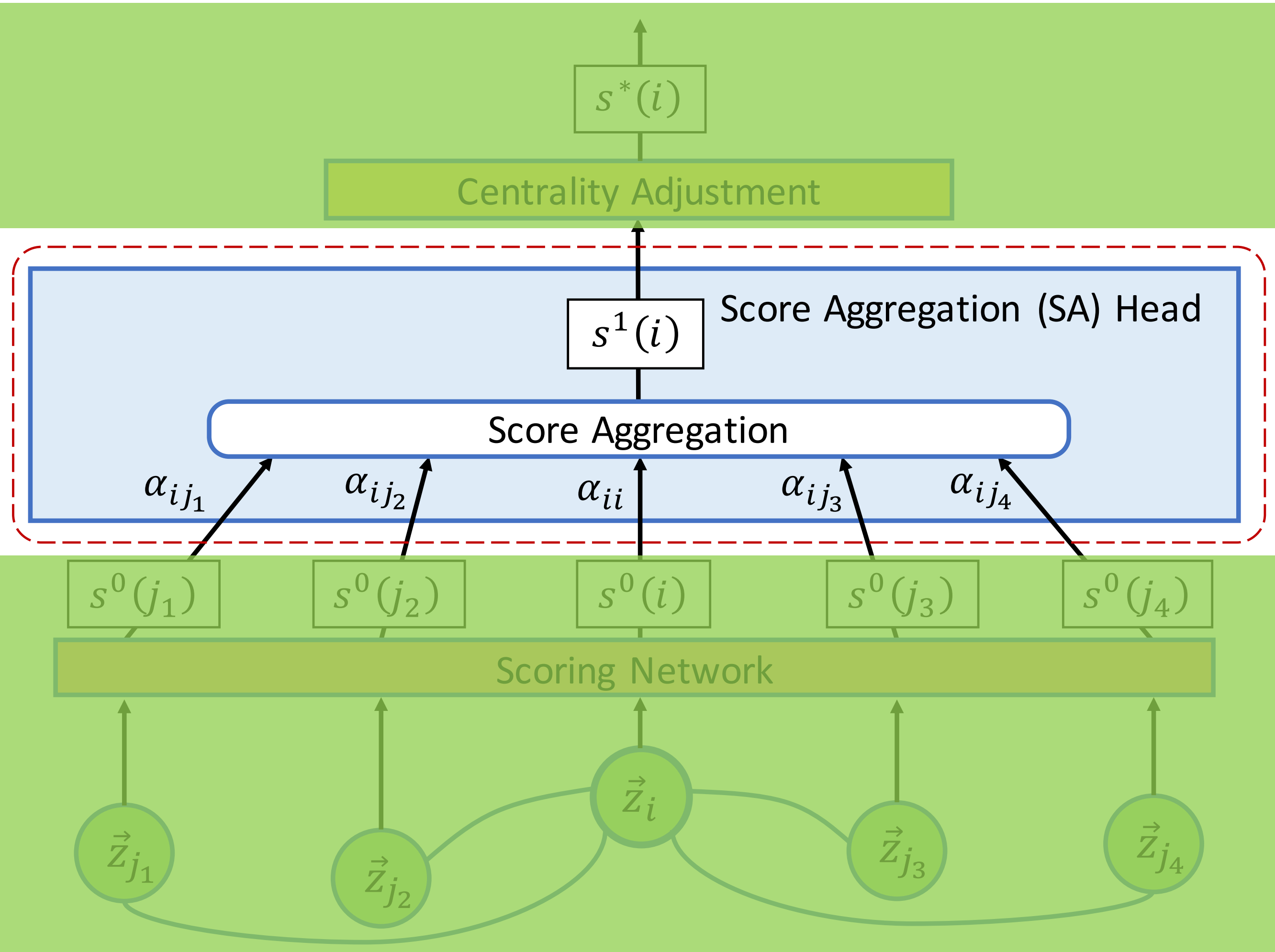
- $s^*(i) = \text{Centr. Adj.}\left(s^1(i), d(i)\right)$
- $s^1(i) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^1 s^0(j)\right)$
- $\alpha_{ij}^1 = f\left(s^0(i), s^0(j), \vec{a}_1, \vec{p}_{ij}\right)$
- $s^0(i) = \text{ScoringNetwork}(\vec{z}_i)$
- $\vec{z}_i$: feature vector of node i
  e.g., node2vec embeddings, distributed bag-of-words representation

Model Architecture: One Layer, One Head



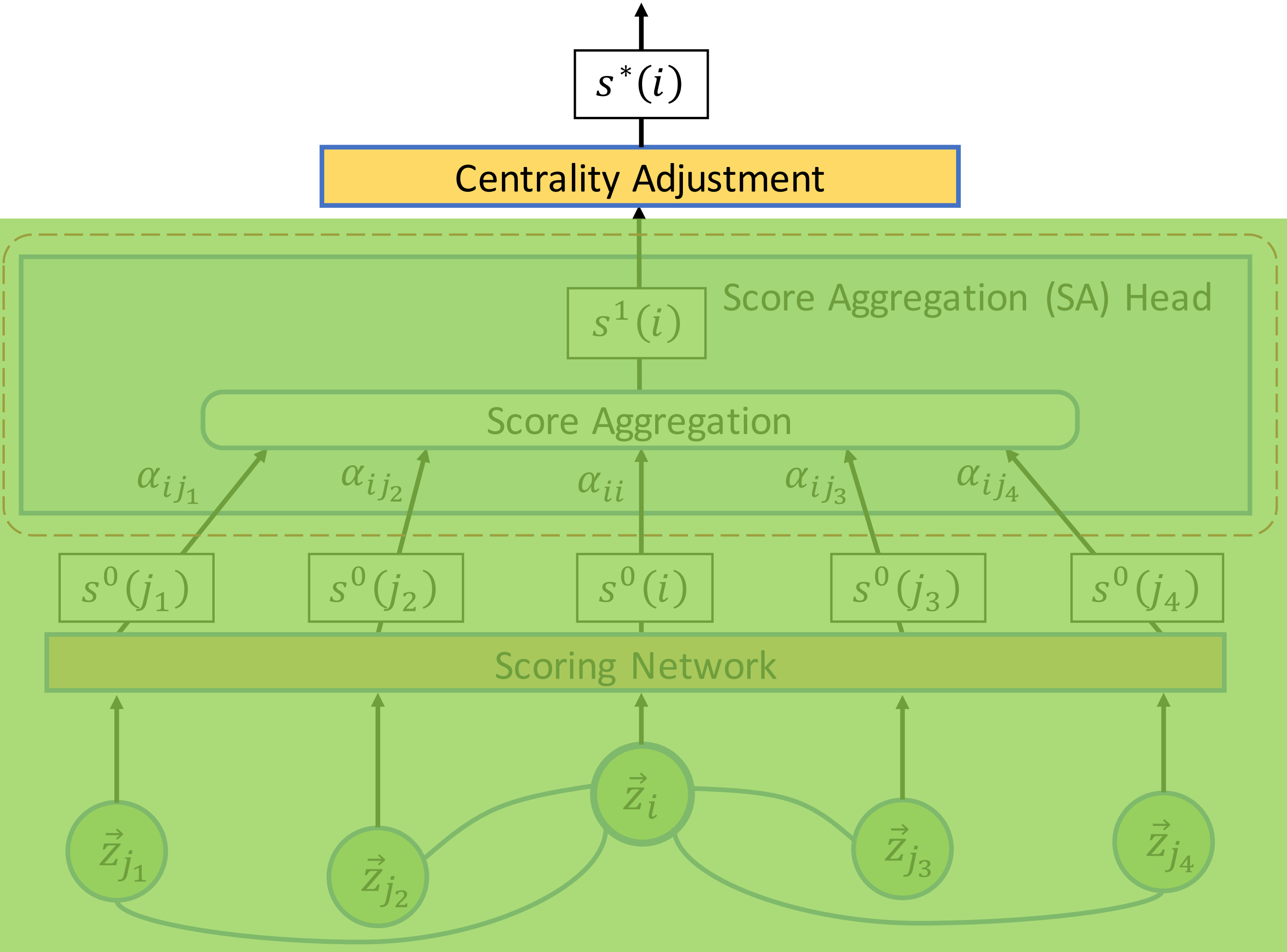
- $s^*(i) = \text{Centr. Adj.}\left(s^1(i), d(i)\right)$
- $s^1(i) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^1 s^0(j)\right)$
- $\alpha_{ij}^1 = f\left(s^0(i), s^0(j), \vec{a}_1, \vec{p}_{ij}\right)$
- $s^0(i) = \text{ScoringNetwork}(\vec{z}_i)$
- $\vec{z}_i$: feature vector of node i
  e.g., node2vec embeddings, distributed bag-of-words representation

Model Architecture: One Layer, One Head



- $s^*(i) = \text{Centr. Adj.}\left(s^1(i), d(i)\right)$
- $s^1(i) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^1 s^0(j)\right)$
- $\alpha_{ij}^1 = f\left(s^0(i), s^0(j), \vec{a}_1, \vec{p}_{ij}\right)$
- $s^0(i) = \text{ScoringNetwork}(\vec{z}_i)$
- $\vec{z}_i$: feature vector of node i
  e.g., node2vec embeddings, distributed bag-of-words representation

Model Architecture: One Layer, One Head

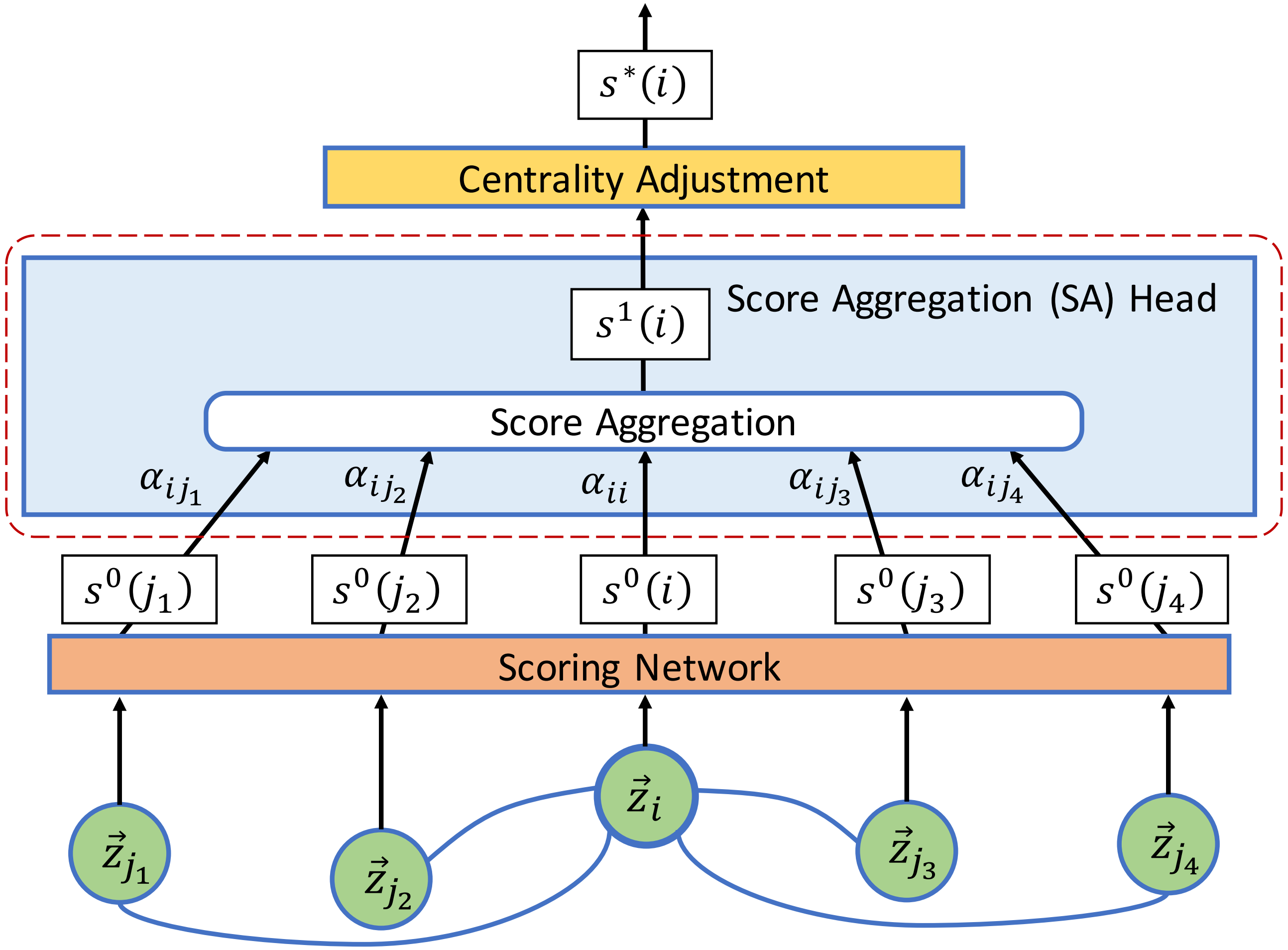


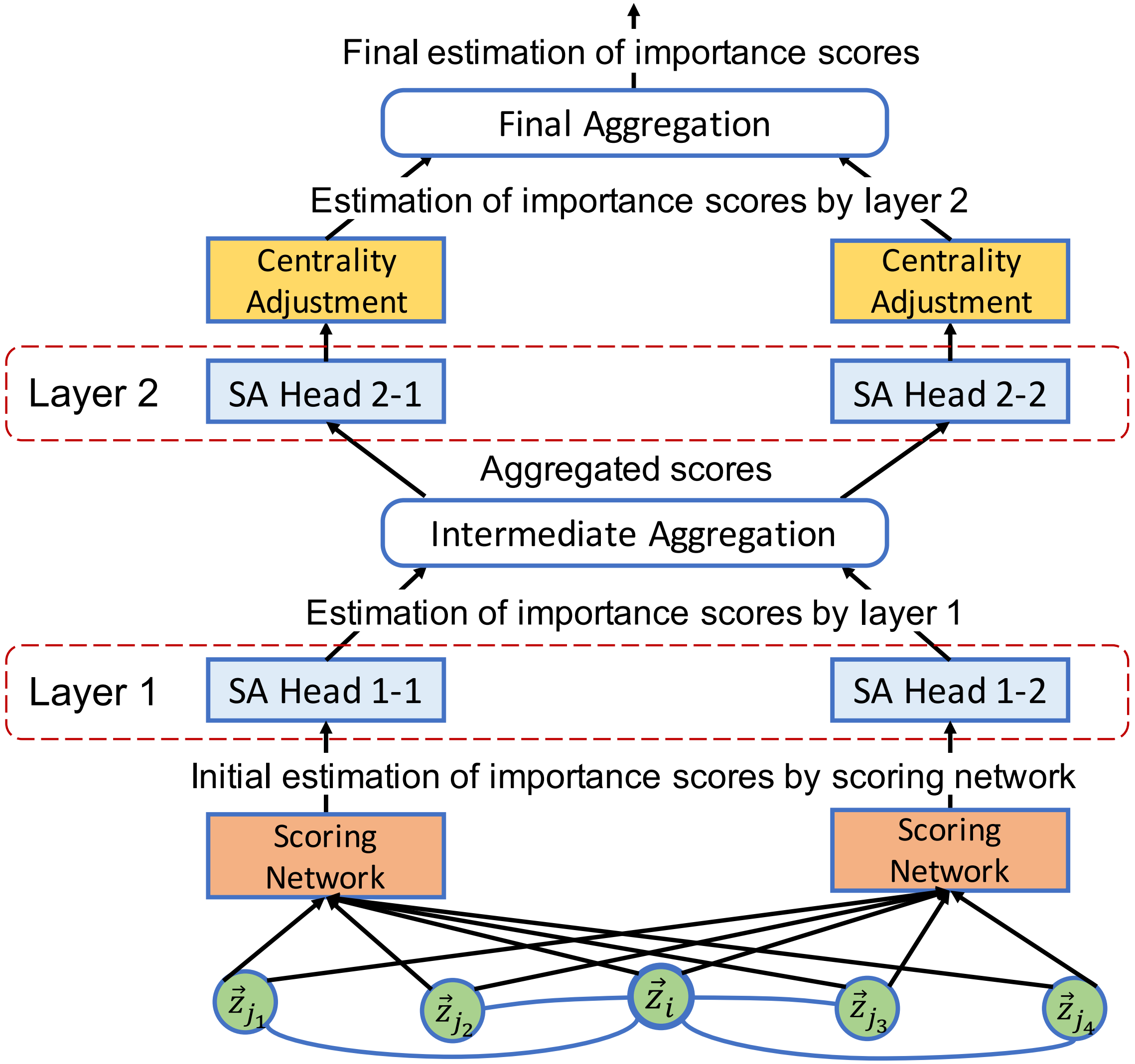
- $s^*(i) = \text{Centr. Adj.}\left(s^1(i), d(i)\right)$
- $s^1(i) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij}^1 s^0(j)\right)$
- $\alpha_{ij}^1 = f\left(s^0(i), s^0(j), \vec{a}_1, \vec{p}_{ij}\right)$
- $s^0(i) = \text{ScoringNetwork}(\vec{z}_i)$
- $\vec{z}_i$: feature vector of node i

  e.g., node2vec embeddings, distributed bag-of-words representation

Model Architecture: Multi Layer, Multi Head

Detail





Roadmap

- Introduction
- Proposed Method: GENI
- **Experimental Results**
- Conclusion

Experiments: Baselines

- Non-trainable approaches
  - PageRank (PR)
  - Personalized PageRank (PPR)
  - Hub, Authority, and Relevance score (HAR)
- Supervised approaches
  - Linear regression (LR)
  - Random forests (RF)
  - Neural networks (NN)
  - Graph attention networks (GAT)

Experiments: Datasets

Name	# Nodes	# Edges	# Predicates	Input Score Type	# Nodes w/ Scores
FB15K	14,951	592,213	1,345	# Pageviews	14,108 (94%)
MUSIC10K	24,830	71,846	10	Song hotttnesss	4,214 (17%)
TMDB5K	123,906	532,058	22	Movie popularity	4,803 (4%)
IMDB	1,567,045	14,067,776	28	# Votes for movies	215,769 (14%)

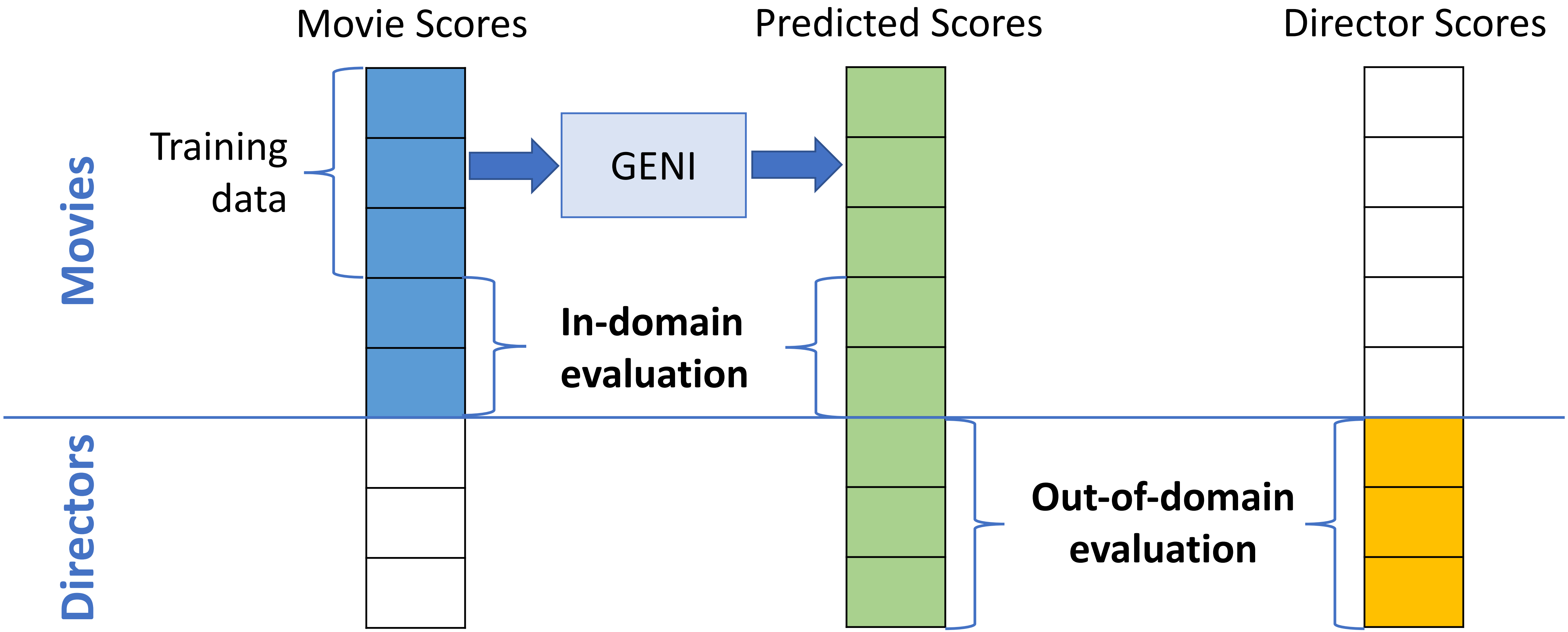
Experiments: Evaluation Strategies

We answer the following questions

[Q1] How well does each method estimate node importance w.r.t. the given input score type?
→ "In-domain" evaluation

[Q2] How well does the estimation of each method generalize to the node of unseen types?
→ "Out-of-domain" evaluation

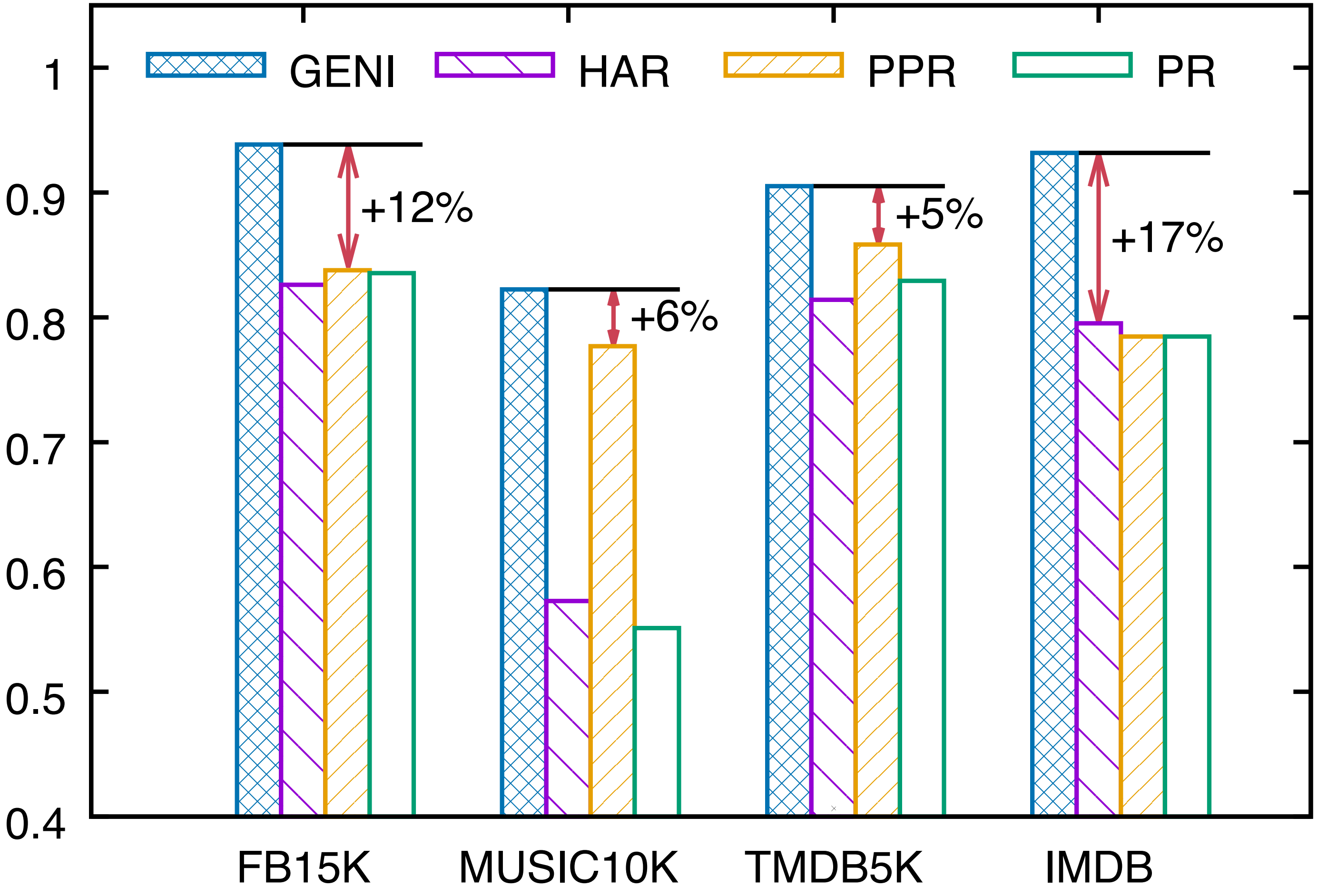
Experiments: Evaluation Strategies



In-Domain Evaluation

GENI (leftmost) outperforms baselines

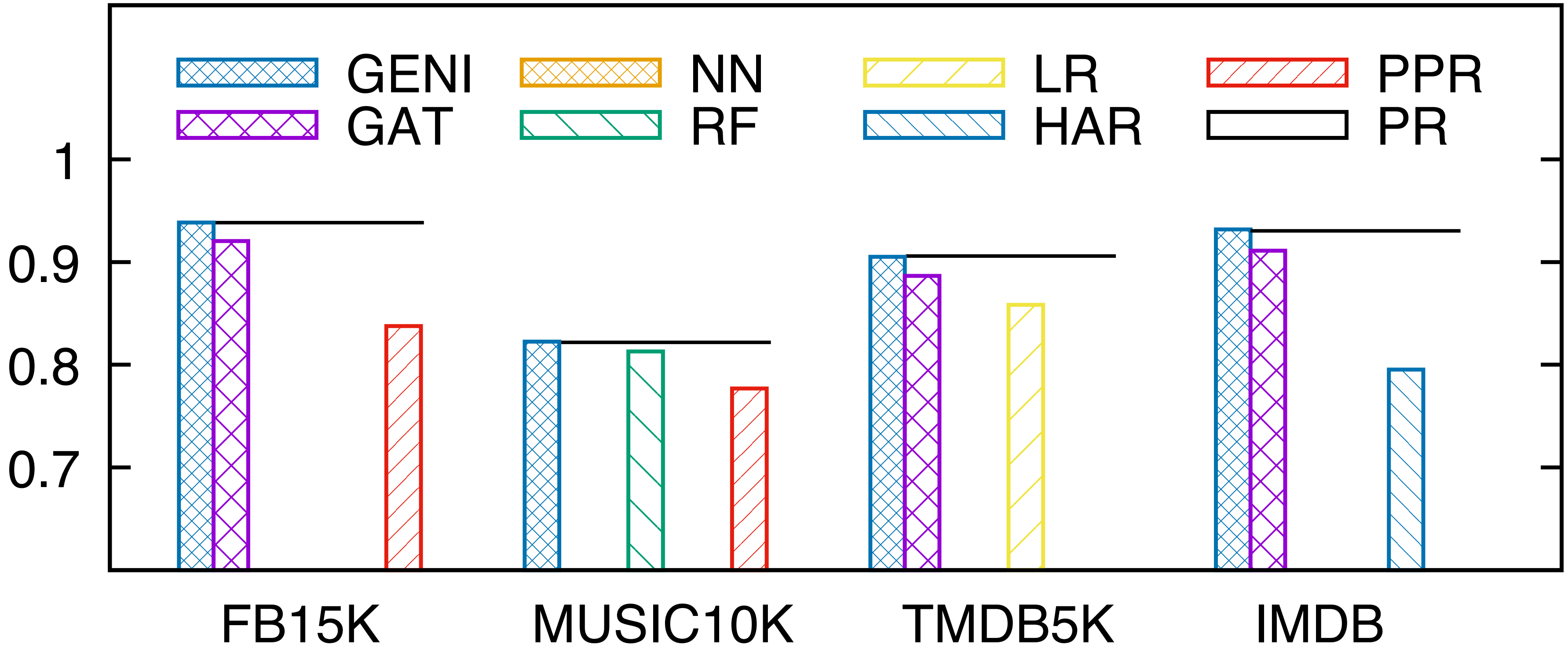
NDCG@100, higher is better



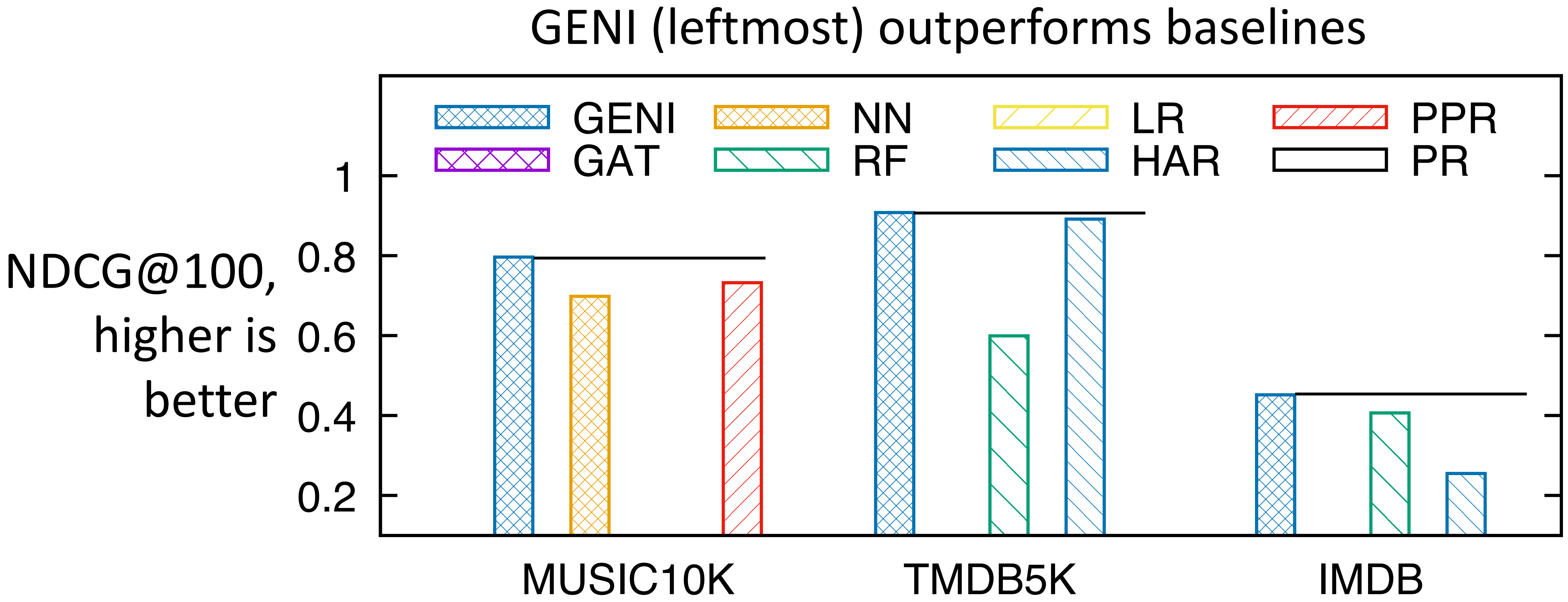
In-Domain Evaluation

GENI (leftmost) outperforms baselines

NDCG@100, higher is better



Out-Of-Domain Evaluation

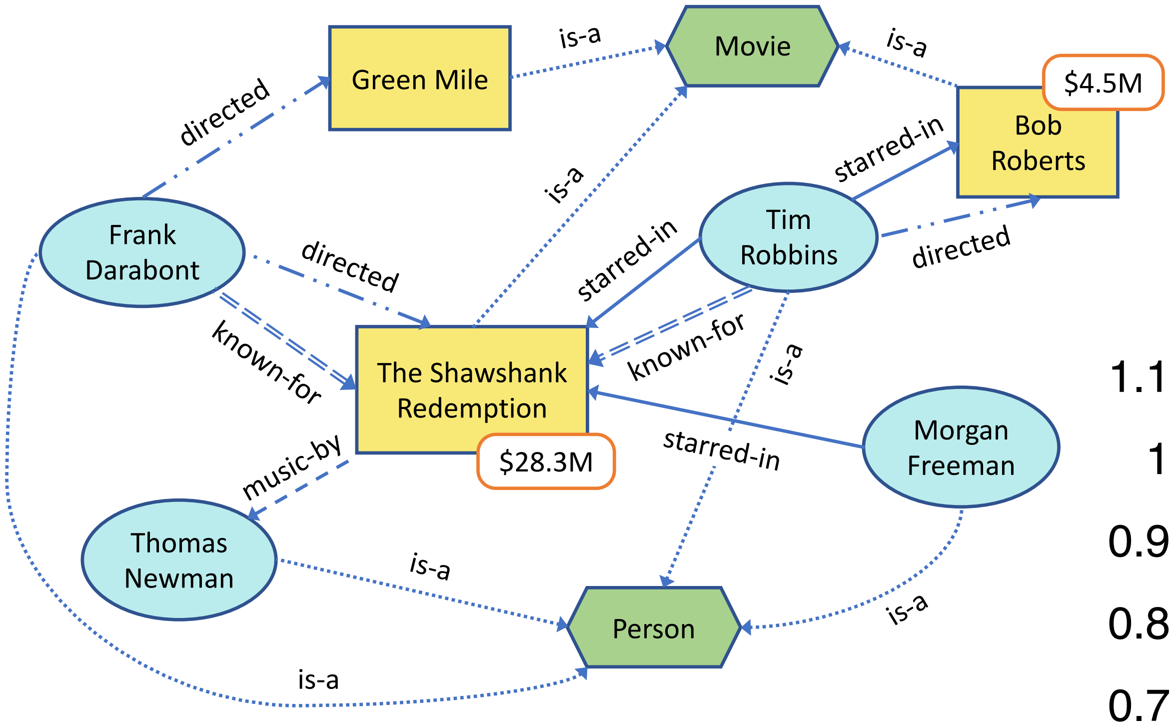


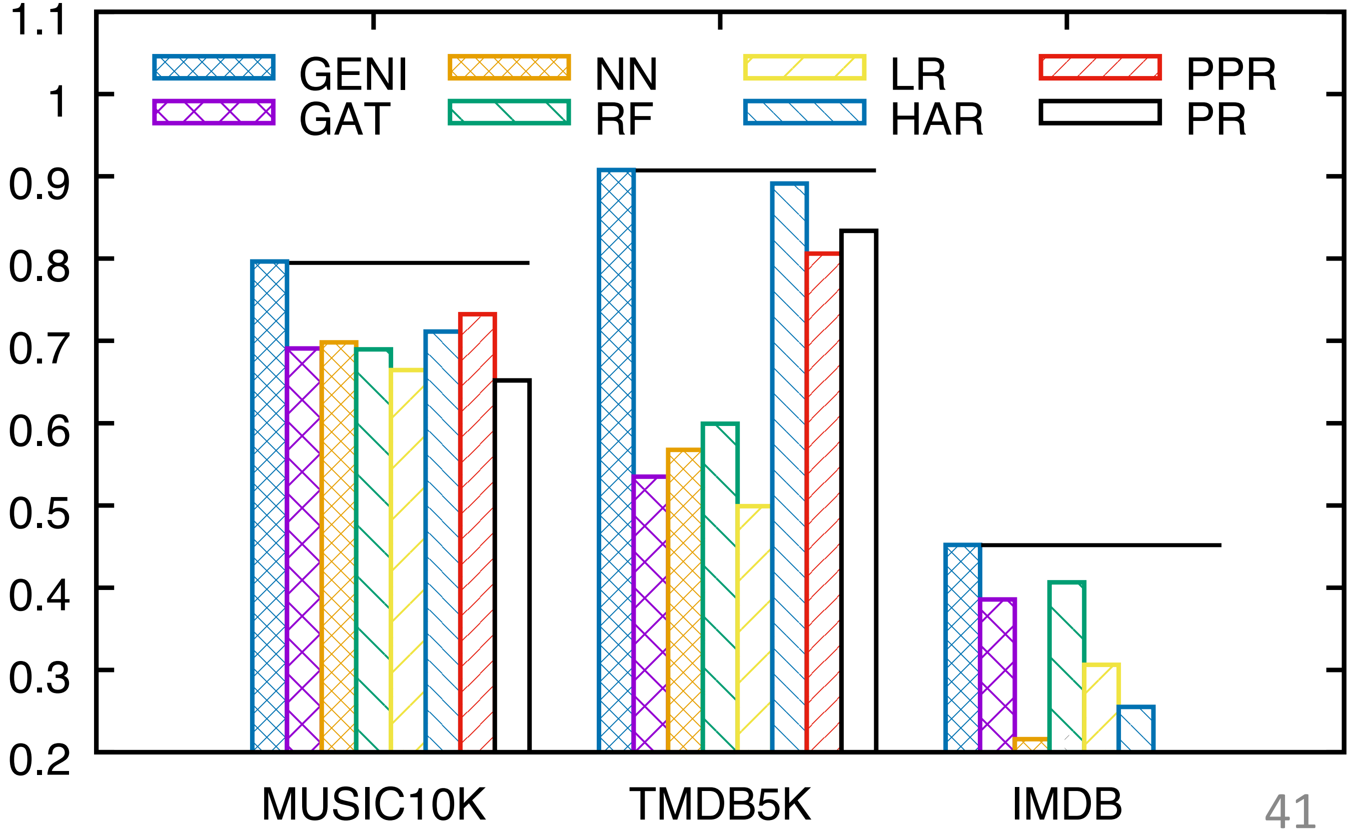
	MUSIC10K	TMDB5K	IMDB
Input Scores	Song hotttnesss	Movie popularity	# Votes for Movies
Out-Of-Domain Scores	Artist hotttnesss	Director ranking	Director ranking

Roadmap

- Introduction
- Proposed Method: GENI
- Experimental Results
- **Conclusion**

Conclusion





- Our method GENI outperforms baselines in estimating node importance in a KG
- Future directions include considering multiple importance scores as input signals

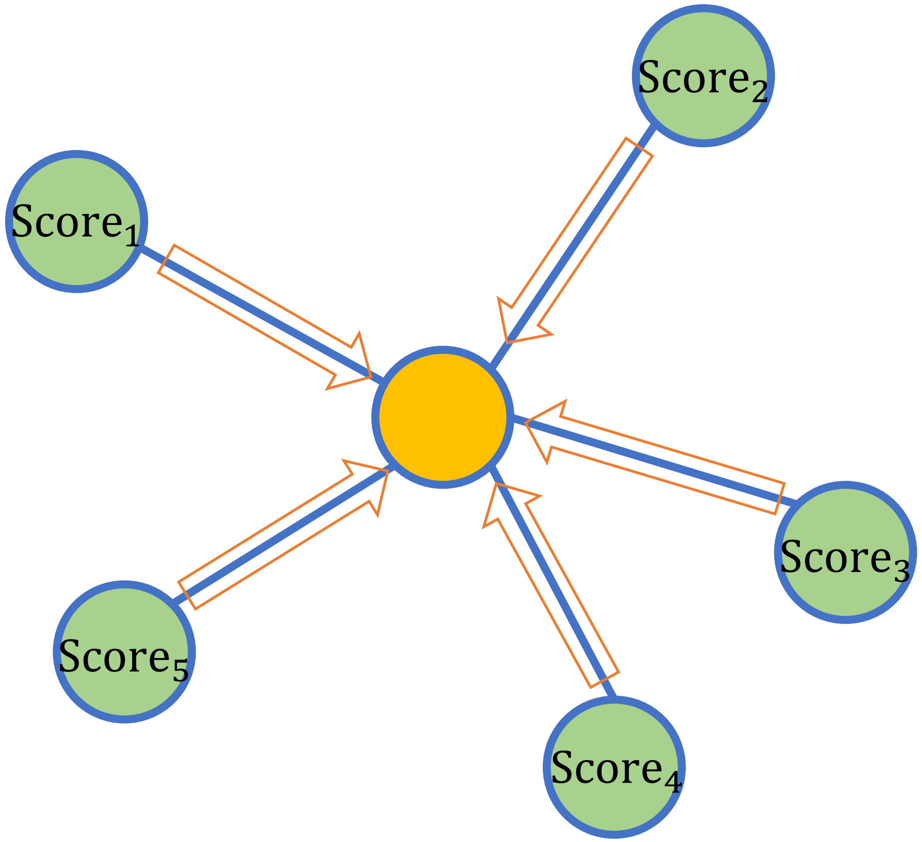
Thank you!

Appendix

Detail

Idea 1: Score Aggregation

- Initial scores $s^0(\cdot)$ are computed by $\mathrm{ScoringNetwork}$, a feed forward NN trained jointly with the rest of GENI



$$s^0(i) = \mathrm{ScoringNetwork}(\vec{z}_i)$$

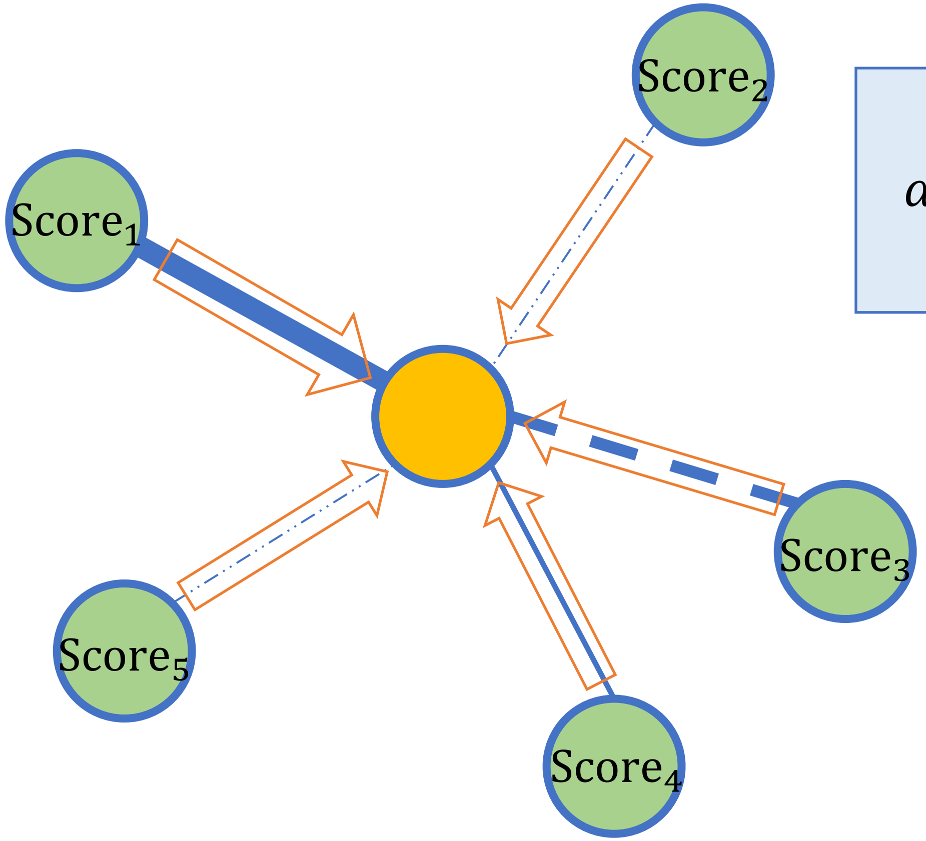
$$s^\ell(i) = \sigma_s\left(\sum_{j\in\mathcal{N}(i)\cup\{i\}} \alpha_{ij}^\ell s^{\ell-1}(j)\right)$$

- $\vec{z}_i$: feature vector of node i
- $s^0(i)$: initial score estimation of node i
- $s^\ell(i)$: estimated score of node i
- $\mathcal{N}(i)$: neighbors of node i
- α_{ij}^ℓ: node i's attention on node j

Detail

Idea 2: Predicate-Aware Attention

- Model how predicates affect the importance of neighboring entities by using shared self-attention mechanism

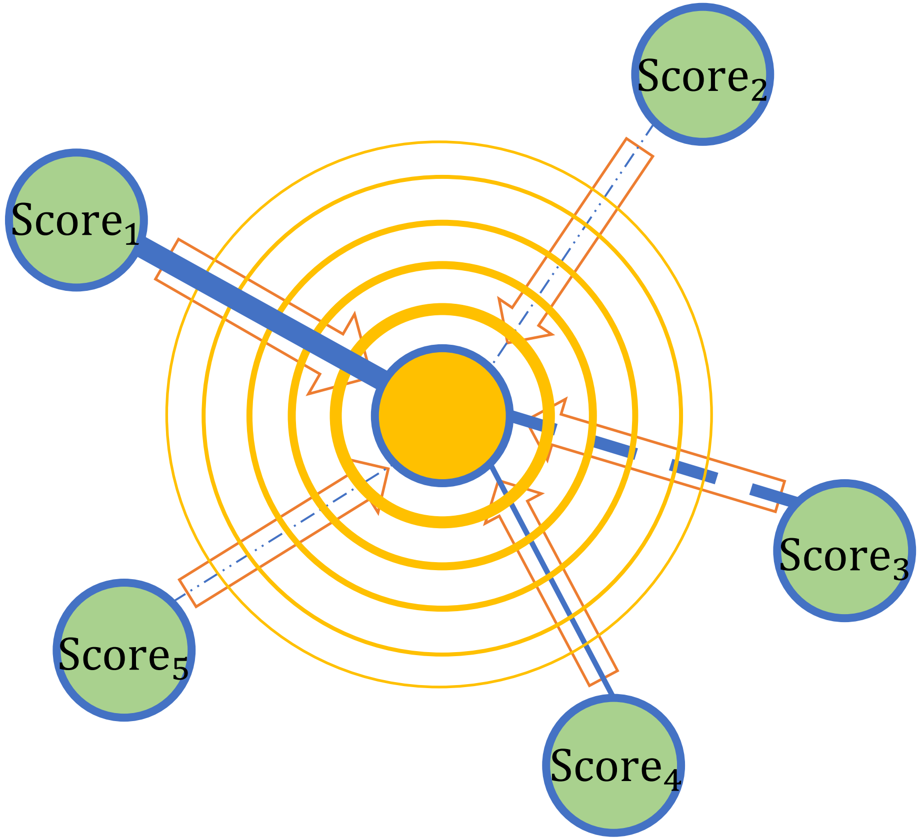


$$\alpha_{ij}^{\ell} = \frac{\exp\left(\sigma_a\left(\sum_m \vec{a}_\ell^T [s^{\ell-1}(i) || \phi(p_{ij}^m) || s^{\ell-1}(j)]\right)\right)}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} \exp\left(\sigma_a\left(\sum_m \vec{a}_\ell^T [s^{\ell-1}(i) || \phi(p_{ik}^m) || s^{\ell-1}(k)]\right)\right)}$$

- α_{ij}^{ℓ}: node i's attention on node j computed by the ℓ-th layer
- p_{ik}^{m}: predicate of m-th edge between nodes i and j
- $\phi(\cdot)$: mapping from a predicate to its embedding
- $s^{\ell}(i)$: estimated score of node i
- $\mathcal{N}(i)$: neighbors of node i

Detail

Idea 3: Centrality Adjustment



$$c(i) = \log(d(i) + \epsilon)$$

$$c^*(i) = \gamma \cdot c(i) + \beta$$

$$s^*(i) = \sigma_s(c^*(i) \cdot s^L(i))$$

- $d(i)$: in-degree of node i
- $c(i)$: initial centrality of node i
- $c^*(i)$: scaled and shifted centrality of node i
- $s^*(i)$: centrality-adjusted score estimation of node i

Experiments: Evaluation Metrics

- Ranking quality
  - **NDCG** (Normalized Discounted Cumulative Gain)
  - **Spearman** correlation coefficient
- Regression quality
  - **RMSE** (Root-Mean-Squared Error)

Experiments: Datasets

Name	# Nodes	# Edges	# Predicates	Input Score Type	# Nodes w/ Scores	Data for OOD Evaluation
FB15K	14,951	592,213	1,345	# Pageviews	14,108 (94%)	N/A
MUSIC10K	24,830	71,846	10	Song hotttnesss	4,214 (17%)	Artist hotttnesss
TMDB5K	123,906	532,058	22	Movie popularity	4,803 (4%)	Director ranking
IMDB	1,567,045	14,067,776	28	# Votes for movies	215,769 (14%)	Director ranking

Detail

Experiments: Evaluation Metrics

- Ranking quality
  - **NDCG** (Normalized Discounted Cumulative Gain)

$$\mathrm{DCG@k} = \sum_{i=1}^{k} \frac{r_i}{\log_2(i+1)} \quad / \quad \mathrm{NDCG@k} = \frac{\mathrm{DCG@k}}{\mathrm{IDCG@k}}$$ where IDCG@k is an ideal DCG at position k

  - **Spearman** correlation coefficient

$$\mathrm{Spearman} = \frac{\sum_i (g_{r_i} - \bar{g}_r)(s_{r_i} - \bar{s}_r)}{\sqrt{(g_{r_i} - \bar{g}_r)^2}\sqrt{(s_{r_i} - \bar{s}_r)^2}}$$

- r_i: graded relevance of node at position i
- $\vec{g}, \vec{s}$: ground truth scores and predicted scores
- $\vec{g}_r, \vec{s}_r$: rankings induced from $\vec{g}$ and $\vec{s}$
- $\bar{g}_r, \bar{s}_r$: mean of $\vec{g}_r$ and $\vec{s}_r$

- Regression quality
  - **RMSE** (Root-Mean-Squared Error)

$$\mathrm{RMSE} = \frac{1}{|V_s|} \sum_{i \in V_s} (s(i) - g(i))^2$$

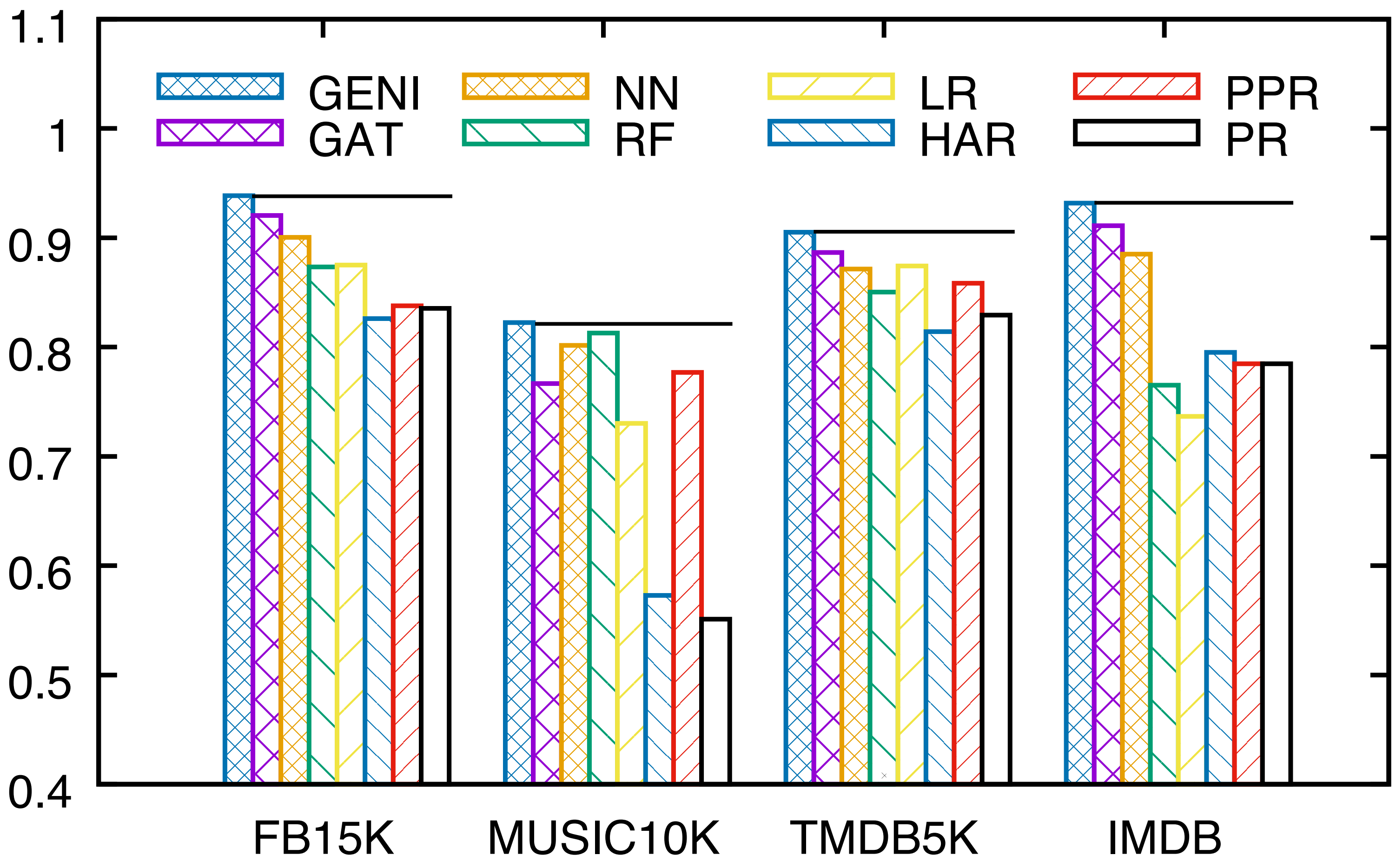
- $s(i)$: predicted score of node i
- $g(i)$: ground truth score of node i
- V_s: a set of nodes with importance scores

Detail

In-Domain Evaluation

GENI (blue) outperforms baselines

NDCG@100, higher is better

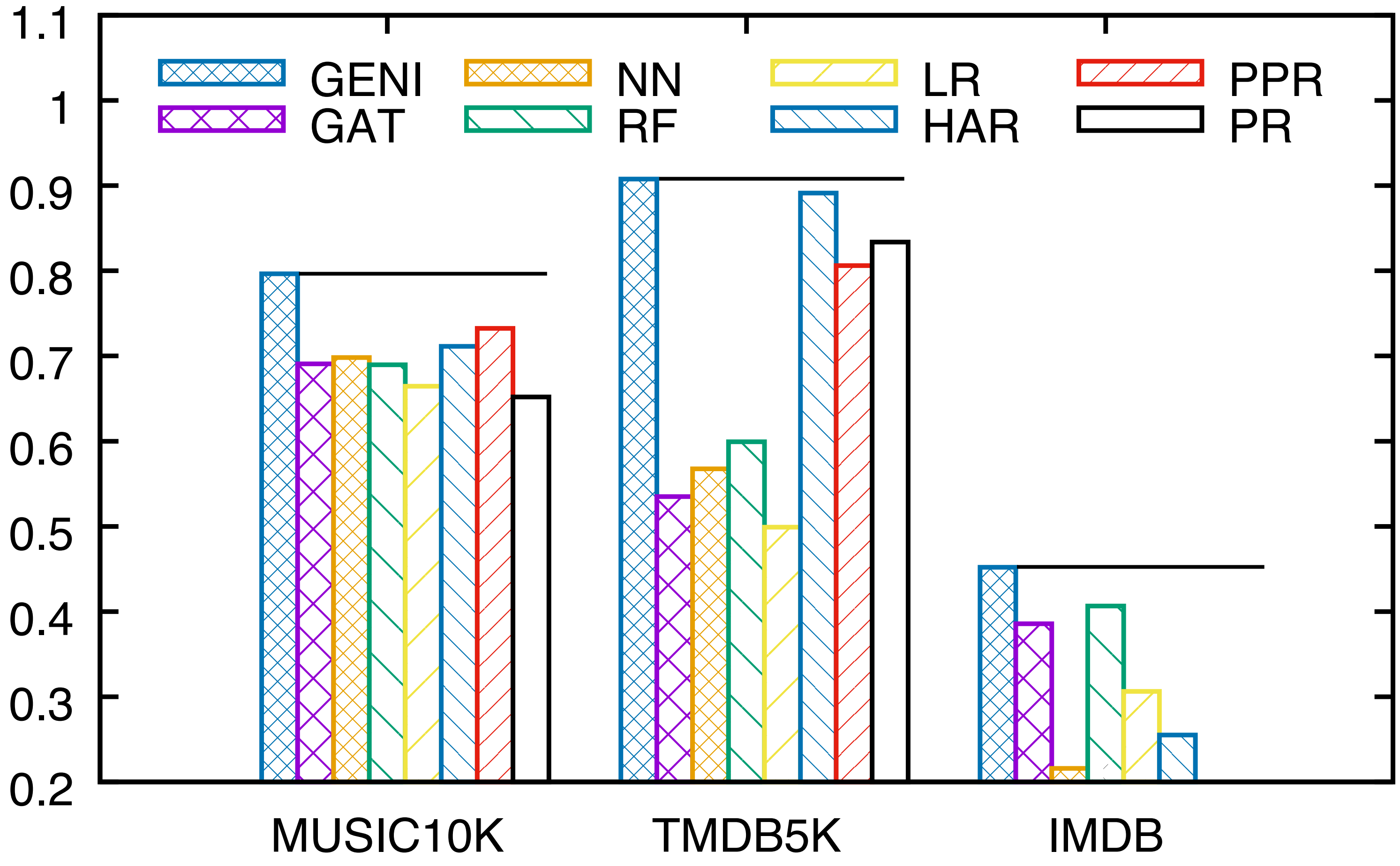


Out-Of-Domain Evaluation

Detail

GENI (blue) outperforms baselines

NDCG@100, higher is better



Detail

Experiments: In-Domain Prediction

Method	FB15K		MUSIC10K		TMDB5K		IMDB	
	NDCG@100	SPEARMAN	NDCG@100	SPEARMAN	NDCG@100	SPEARMAN	NDCG@100	SPEARMAN
PR	0.8354 ± 0.016	0.3515 ± 0.015	0.5510 ± 0.021	−0.0926 ± 0.034	0.8293 ± 0.026	0.5901 ± 0.011	0.7847 ± 0.048	0.0881 ± 0.004
PPR	0.8377 ± 0.015	0.3667 ± 0.015	0.7768 ± 0.009	0.3524 ± 0.046	0.8584 ± 0.013	<u>0.7385 ± 0.010</u>	0.7847 ± 0.048	0.0881 ± 0.004
HAR	0.8261 ± 0.005	0.2020 ± 0.012	0.5727 ± 0.017	0.0324 ± 0.044	0.8141 ± 0.021	0.4976 ± 0.014	0.7952 ± 0.036	0.1318 ± 0.005
LR	0.8750 ± 0.005	0.4626 ± 0.019	0.7301 ± 0.023	0.3069 ± 0.032	0.8743 ± 0.015	0.6881 ± 0.013	0.7365 ± 0.009	0.5013 ± 0.002
RF	0.8734 ± 0.005	0.5122 ± 0.019	<u>0.8129 ± 0.012</u>	<u>0.4577 ± 0.012</u>	0.8503 ± 0.016	0.5959 ± 0.022	0.7651 ± 0.010	0.4753 ± 0.005
NN	0.9003 ± 0.005	0.6031 ± 0.012	0.8015 ± 0.017	0.4491 ± 0.027	0.8715 ± 0.006	0.7009 ± 0.009	0.8850 ± 0.016	0.5120 ± 0.008
GAT	<u>0.9205 ± 0.009</u>	<u>0.7054 ± 0.013</u>	0.7666 ± 0.016	0.4276 ± 0.023	<u>0.8865 ± 0.011</u>	0.7180 ± 0.010	<u>0.9110 ± 0.011</u>	<u>0.7060 ± 0.007</u>
GENI	**0.9385 ± 0.004**	**0.7772 ± 0.006**	**0.8224 ± 0.018**	**0.4783 ± 0.009**	**0.9051 ± 0.005**	**0.7796 ± 0.009**	**0.9318 ± 0.005**	**0.7387 ± 0.002**

- GENI performs the best for all datasets
- Supervised models mostly outperform non-trainable ones
- Directly utilizing network connectivity further enhances performance

Detail

Experiments: Out-Of-Domain Prediction

Method	MUSIC10K		TMDB5K		IMDB	
	NDCG@100	NDCG@2000	NDCG@100	NDCG@2000	NDCG@100	NDCG@2000
PR	0.6520 ± 0.000	0.8779 ± 0.000	0.8337 ± 0.000	0.8079 ± 0.000	0.0000 ± 0.000	0.1599 ± 0.000
PPR	0.7324 ± 0.006	0.9118 ± 0.002	0.8060 ± 0.041	0.7819 ± 0.022	0.0000 ± 0.000	0.1599 ± 0.000
HAR	0.7113 ± 0.004	0.8982 ± 0.001	0.8913 ± 0.010	0.8563 ± 0.007	0.2551 ± 0.019	0.3272 ± 0.005
LR	0.6644 ± 0.006	0.8667 ± 0.001	0.4990 ± 0.013	0.5984 ± 0.002	0.3064 ± 0.007	0.2755 ± 0.003
RF	0.6898 ± 0.022	0.8796 ± 0.003	0.5993 ± 0.040	0.6236 ± 0.005	0.4066 ± 0.145	0.3719 ± 0.040
NN	0.6981 ± 0.017	0.8836 ± 0.005	0.5675 ± 0.023	0.6172 ± 0.009	0.2158 ± 0.035	0.3105 ± 0.019
GAT	0.6909 ± 0.009	0.8834 ± 0.003	0.5349 ± 0.016	0.5999 ± 0.007	0.3858 ± 0.065	0.4209 ± 0.016
GENI	**0.7964 ± 0.007**	**0.9121 ± 0.002**	**0.9078 ± 0.004**	**0.8776 ± 0.002**	**0.4519 ± 0.051**	**0.4962 ± 0.025**

- Prediction is done for entities of some type $\mathcal{T}$, which is not used for training.
- GENI achieves the best results for all KGs
- Non-trainable methods achieves better results on MUSIC10K and TMDB5K

Detail

Experiments: Case Study

Top-10 movies (In-domain estimation)

	GENI		HAR		GAT	
1	The Dark Knight Rises	11	Jason Bourne	63	The Dark Knight Rises	11
2	The Lego Movie	70	The Wolf of Wall Street	21	Clash of the Titans	103
3	Spectre	10	Rock of Ages	278	Ant-Man	4
4	Les Misérables	94	Les Misérables	94	The Lego Movie	68
5	The Amazing Spider-Man	22	The Dark Knight Rises	7	Jack the Giant Slayer	126
6	Toy Story 2	39	V for Vendetta	27	Spectre	7
7	V for Vendetta	26	Now You See Me 2	81	The Wolf of Wall Street	16
8	Clash of the Titans	97	Spectre	5	The 5th Wave	67
9	Ant-Man	-2	Austin Powers in Goldmember	140	The Hunger Games: Mockingjay - Part 2	-4
10	Iron Man 2	29	Alexander	141	X-Men: First Class	767

Top-10 directors (Out-of-domain estimation)

	GENI		HAR		GAT	
1	Steven Spielberg	0	Steven Spielberg	0	Noam Murro	N/A
2	Tim Burton	9	Martin Scorsese	44	J Blakeson	N/A
3	Ridley Scott	6	Ridley Scott	6	Pitof	N/A
4	Martin Scorsese	42	Clint Eastwood	19	Paul Tibbitt	N/A
5	Francis Ford Coppola	158	Woody Allen	112	Rupert Sanders	N/A
6	Peter Jackson	-4	Robert Zemeckis	1	Alan Taylor	145
7	Robert Rodriguez	127	Tim Burton	4	Peter Landesman	N/A
8	Gore Verbinski	8	David Fincher	40	Hideo Nakata	N/A
9	Joel Schumacher	63	Oliver Stone	105	Drew Goddard	N/A
10	Robert Zemeckis	-3	Ron Howard	-2	Tim Miller	N/A

- The top-10 movies predicted by GENI is qualitatively better than others
- The top-10 directors by GENI and HAR are similar in quality, having five common directors
- GAT's estimation on directors is much worse than the two others

Detail

Experiments: In-Domain Regression

RMSE of In-Domain Prediction for Supervised Methods

Method	FB15K	MUSIC10K	TMDB5K	IMDB
LR	1.3536 ± 0.017	0.1599 ± 0.002	0.8431 ± 0.028	1.7534 ± 0.005
RF	1.2999 ± 0.024	0.1494 ± 0.002	0.9223 ± 0.015	1.8181 ± 0.011
NN	1.2463 ± 0.015	0.1622 ± 0.009	0.8496 ± 0.012	2.0279 ± 0.033
GAT	1.0798 ± 0.031	0.1635 ± 0.007	0.8020 ± 0.010	1.2972 ± 0.018
GENI	$\mathbf{0.9471 \pm 0.017}$	$\mathbf{0.1491 \pm 0.002}$	$\mathbf{0.7150 \pm 0.003}$	$\mathbf{1.2079 \pm 0.011}$

- GENI performs better than other supervised baselines
- Overall, the regression performance of supervised methods follows a similar trend to their performance in terms of ranking measures

Detail

Experiments: Flexibility for Centrality Adjustment

Method	FB15K		TMDB5K	
	NDCG@100	SPEARMAN	NDCG@100	SPEARMAN
PR	0.835 ± 0.02	0.352 ± 0.02	0.829 ± 0.03	0.590 ± 0.01
Log In-Degree	0.810 ± 0.02	0.300 ± 0.03	0.852 ± 0.02	0.685 ± 0.02
GENI-Fixed CA	0.868 ± 0.01	0.613 ± 0.01	0.899 ± 0.01	0.771 ± 0.01
GENI-Flexible CA	$\mathbf{0.938 \pm 0.00}$	$\mathbf{0.777 \pm 0.01}$	$\mathbf{0.905 \pm 0.01}$	$\mathbf{0.780 \pm 0.01}$

- GENI with fixed CA estimates $s^*(i) = \sigma_s\big(c(i) \cdot s^L(i)\big)$
- When node centrality correlates well with input scores, fixed CA works well
- When node centrality does not agree with input scores, flexible CA performs much better than fixed CA

Experiments: Effect of Considering Predicates

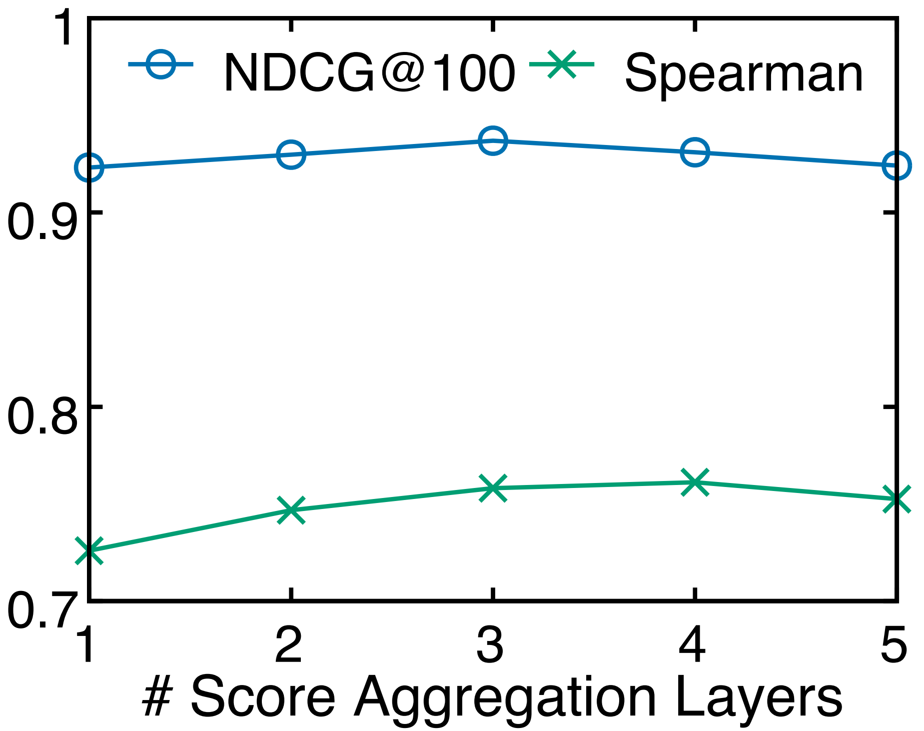
Detail

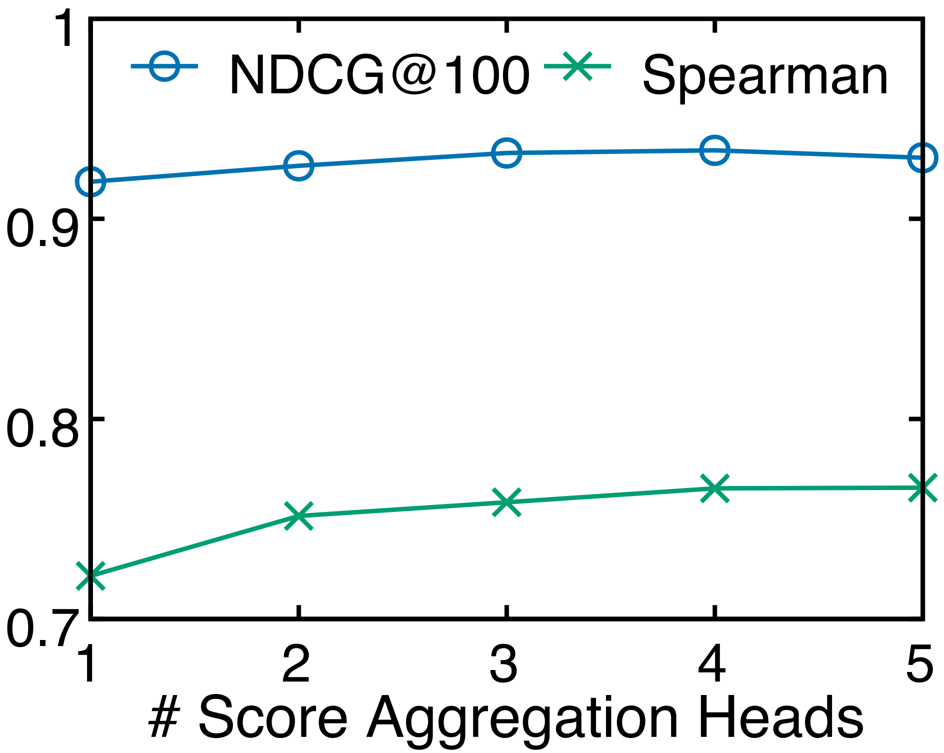
Metric	Shared Embedding	Distinct Embedding
NDCG@100	0.9062 ± 0.008	$\mathbf{0.9385 \pm 0.004}$
SPEARMAN	0.6894 ± 0.007	$\mathbf{0.7772 \pm 0.006}$

- Using "shared embedding" forces GENI to lose the ability to distinguish between different predicates
- Results show that GENI makes an effective use of predicates for modeling the relation between node importance

Experiments: Parameter Sensitivity

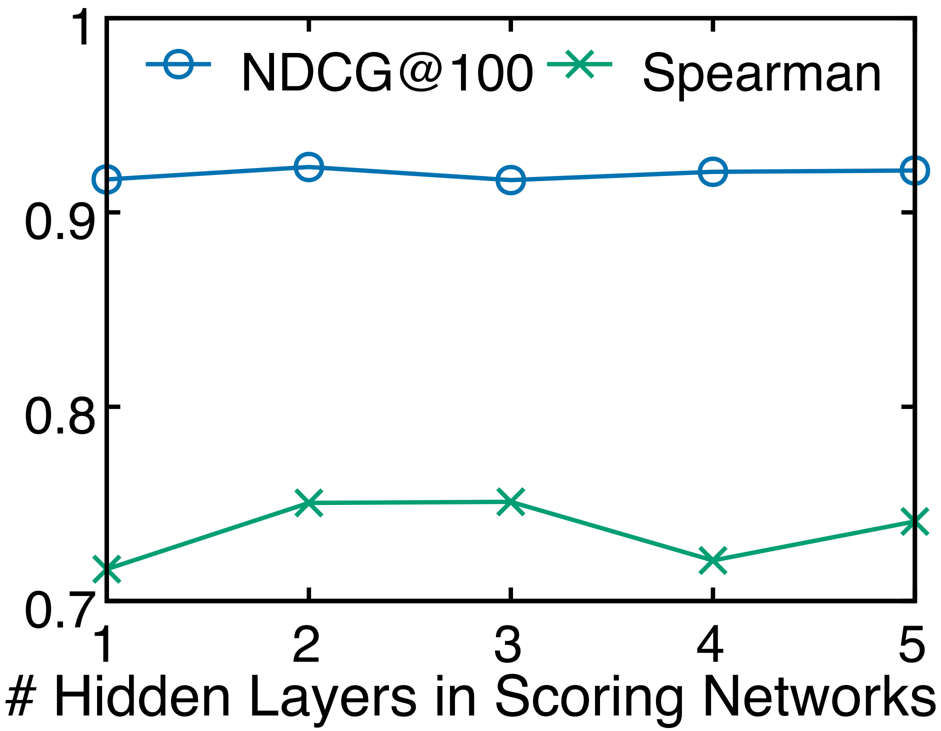
Detail

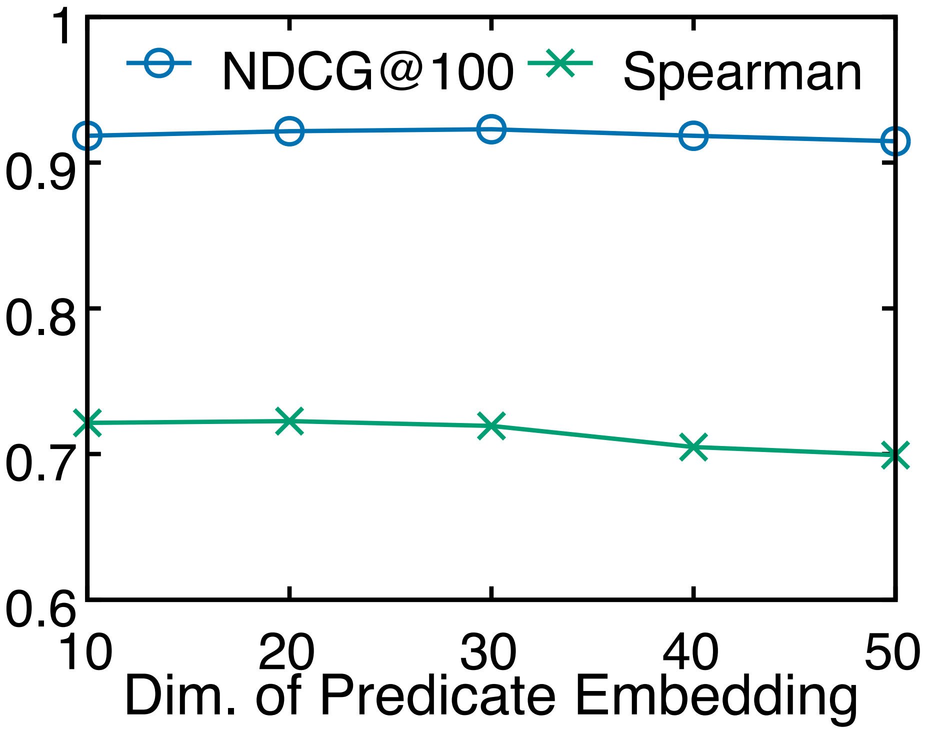




- Model performance improves as we use a greater number of SA layers and SA heads

Experiments: Parameter Sensitivity





- Model performance tends to improve as we use a greater number of hidden layers in scoring networks
- Increasing the dimension of predicate embedding beyond an appropriate value negatively affects model performance