Lecture#2: Inductive Reasoning on Knowledge Graphs

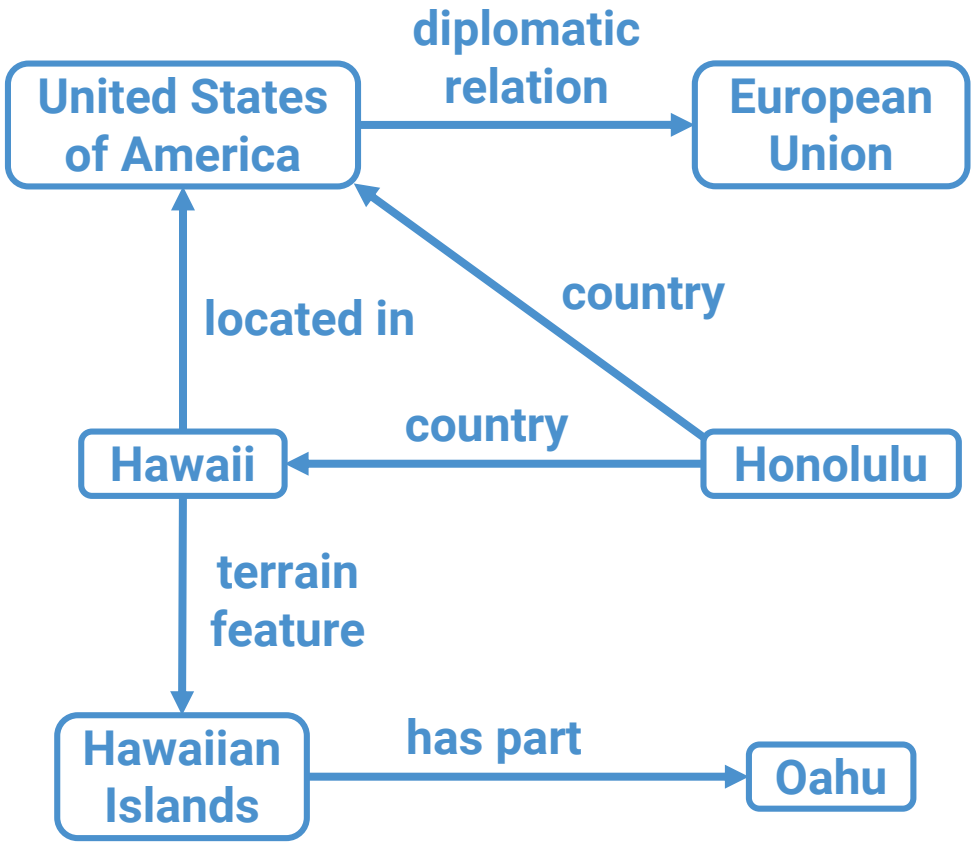
Joyce Jiyoung Whang
School of Computing, KAIST

Key Facets in Modern Knowledge Graph Representation Learning (KeyKGRL), ISWC 2025 Tutorial

https://bdi-lab.kaist.ac.kr

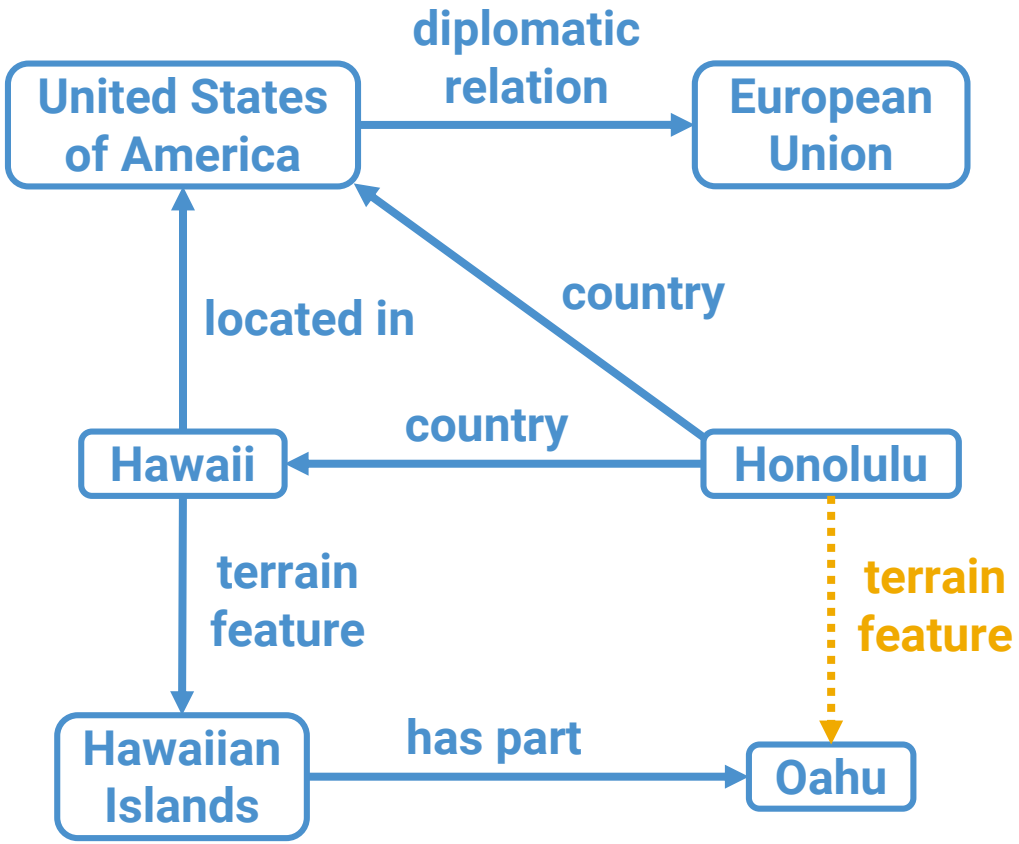
KAIST BDILab
BIG DATA INTELLIGENCE

01 Knowledge Graph



Training Graph

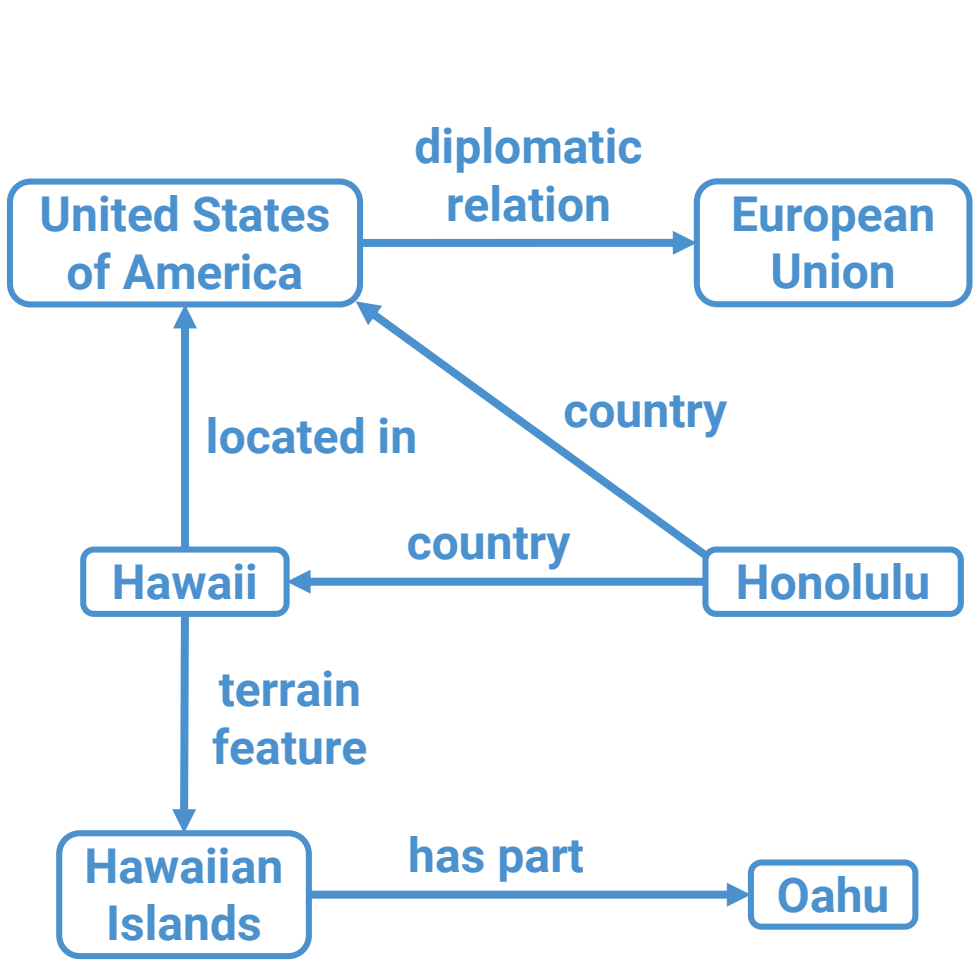
01 Transductive Knowledge Graph Completion



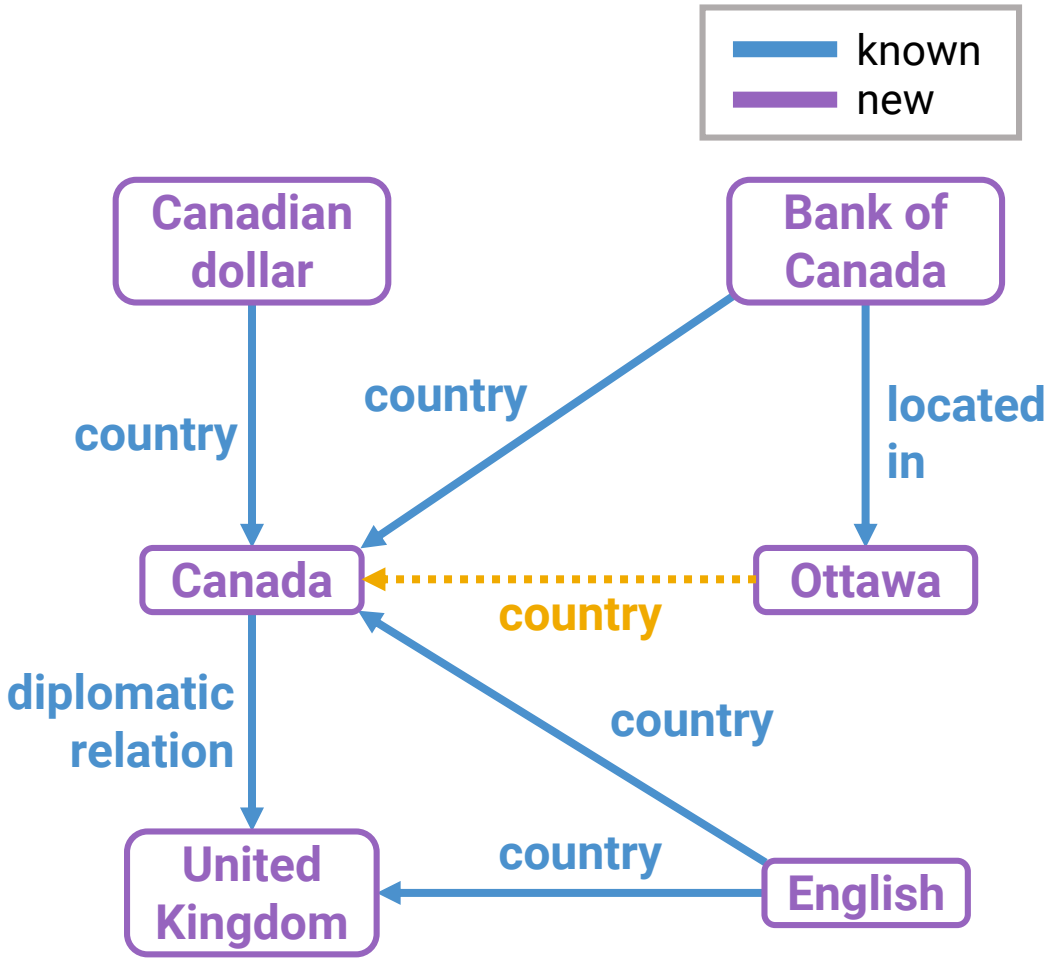
Training Graph

(Honolulu, terrain feature, ?)

01 Inductive Inference for Entities



Training Graph

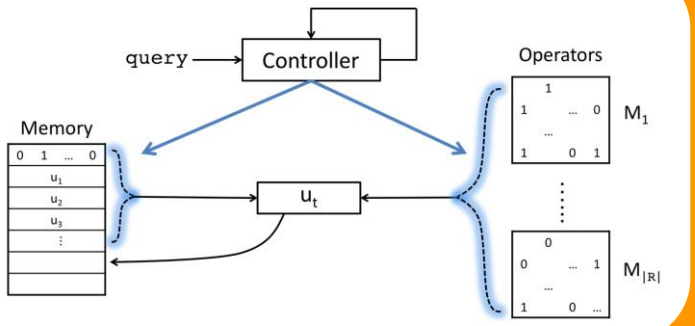


Inference Graph

01 Inductive Reasoning Methods

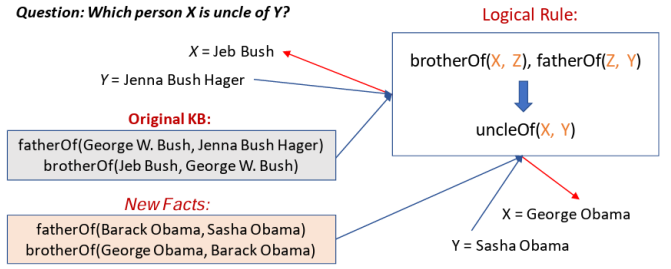
NeuralLP (NIPS 2017)

- Learns probabilistic first-order logical rules for reasoning on knowledge graphs
  - Requires learning the parameters in a continuous space and the structure in a discrete space
- Learns the parameter and structure of first-order logical rules



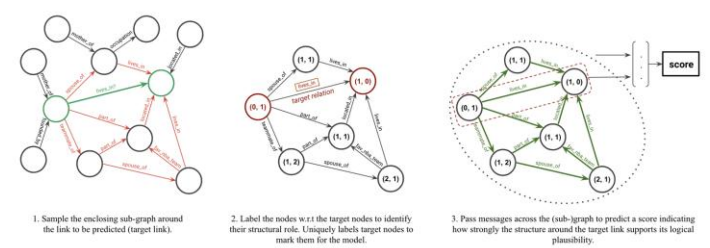
DRUM (NeurIPS 2019)

- Learns probabilistic first-order logical rules for inductive and interpretable link prediction
- Connects the learning of confidence scores for each rule and low-rank tensor approximation
- Uses bidirectional RNNs to share information across relations



GraIL (ICML 2020)

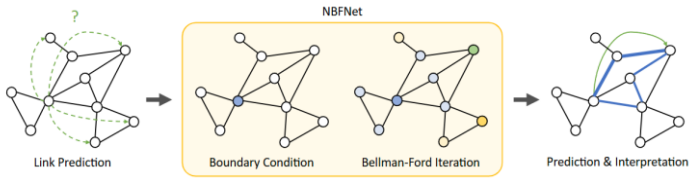
- Proposes a GNN-based KGC framework that reasons over local subgraph structures
  - Has an inductive bias to learn entity-independent relational semantics
- GraIL can represent a useful subset of first-order logic



01 Inductive Reasoning Methods

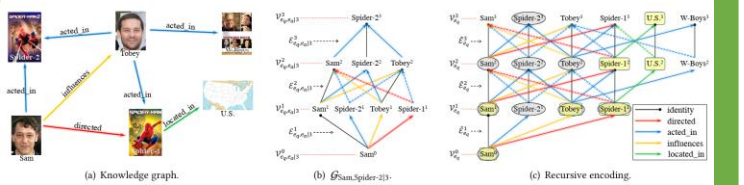
NBFNet (NeurIPS 2021)

- Defines the representation of a node pair as the generalized sum of all path representation between them
- Defines the representation of each path as the generalized product of the edge representations in the path
- Parameterizes the generalized Bellman-Ford algorithm



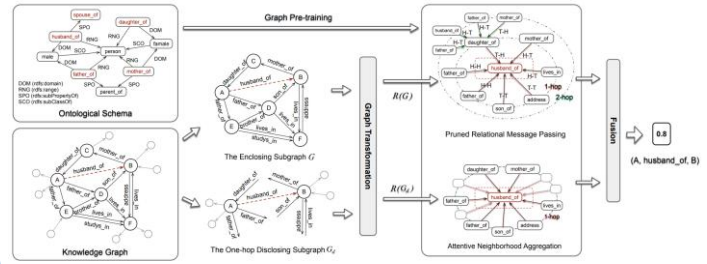
RED-GNN (TheWebConf 2022)

- Proposes relational directed graph (r-digraph), which is composed of overlapped relational paths
  - Designed to capture the local evidence of KGs
- Uses dynamic programming to recursively encode multiple r-digraphs with shared edges



RMPI (ICDE 2023)

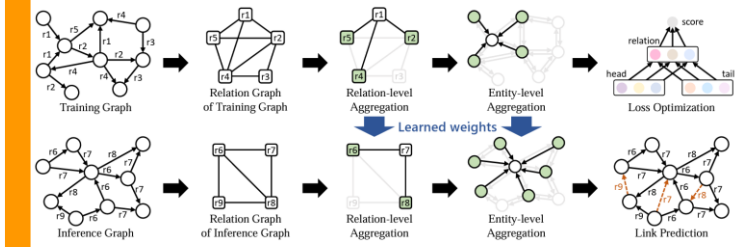
- Passes messages directly between relations to use relation patterns for subgraph reasoning
  - Proposes techniques for relational message passing
- RMPI can utilize the relation semantics defined in the KG's ontological schema



01 Inductive Reasoning Methods

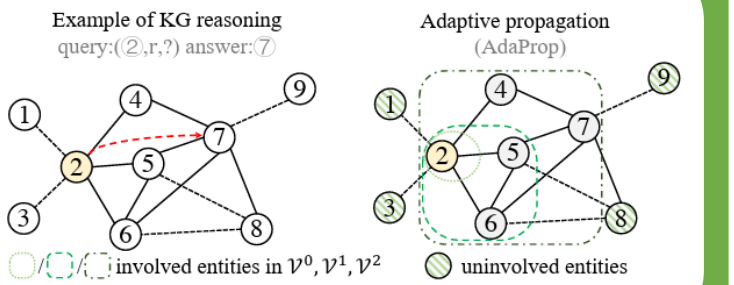
InGram (ICML 2023)

- Defines a relation graph as a weighted graph of relations and the affinity weights between them
- Learns how to aggregate neighbor embeddings to generate relation and entity embeddings
  - Utilizes the relation graph and the original knowledge graph



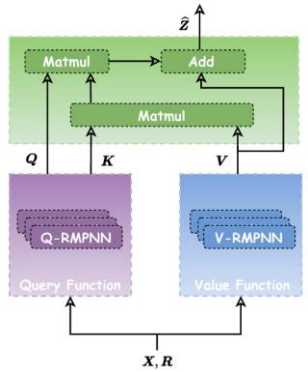
AdaProp (KDD 2023)

- Learns an adaptive propagation path to filter out irrelevant entities while preserving promising targets
- Designs an incremental sampling mechanism with linear complexity
- Designs a learning-based sampling distribution to identify the semantically related entities



KnowFormer (ICML 2024)

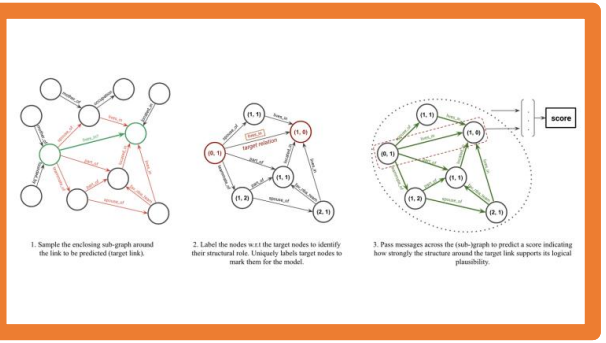
- Utilizes a transformer architecture to perform reasoning on KGs from the message-passing perspective
- Defines the attention computation based on the query prototype of knowledge graph reasoning
- Introduces structure-aware modules to calculate query, key, and value



Inductive Relation Prediction by Subgraph Reasoning

Komal K. Teru*, Etienne G. Denis, and William L. Hamilton

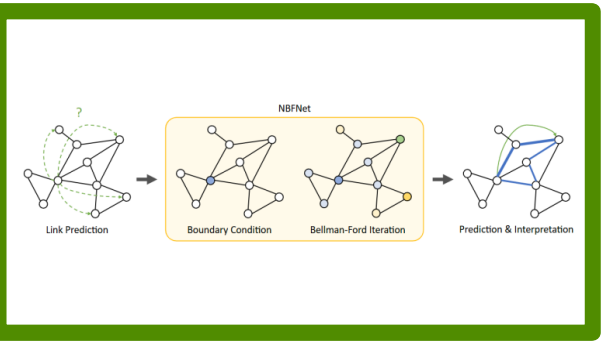
ICML 2020



Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction

Zhaocheng Zhu, Zuobai Zhang,
Louis-Pascal Xhonneux, and Jian Tang

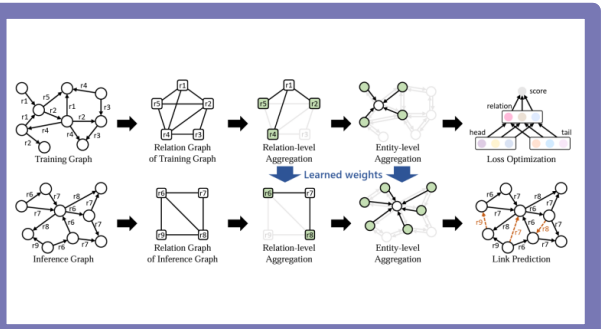
NeurIPS 2021



InGram: Inductive Knowledge Graph Embedding via Relation Graphs

Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang*

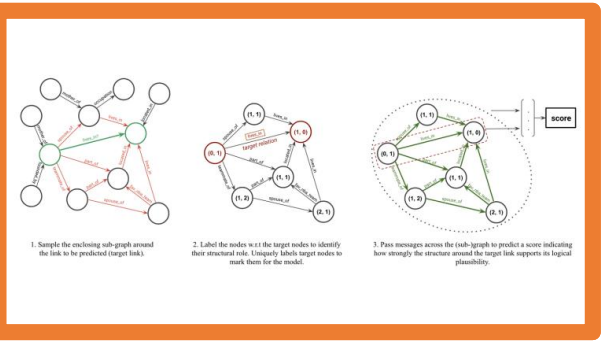
ICML 2023



Inductive Relation Prediction by Subgraph Reasoning

Komal K. Teru*, Etienne G. Denis, and William L. Hamilton

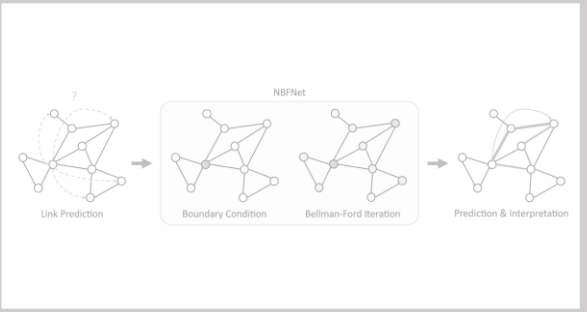
ICML 2020



Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction

Zhaocheng Zhu, Zuobai Zhang,
Louis-Pascal Xhonneux, and Jian Tang

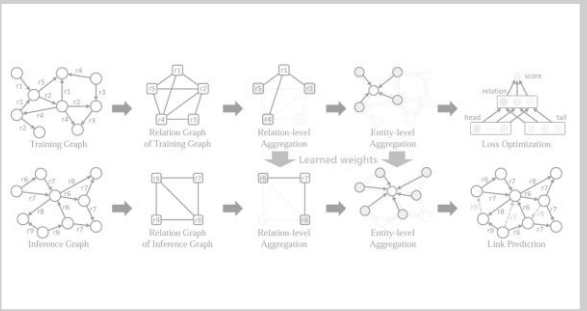
NeurIPS 2021



InGram: Inductive Knowledge Graph Embedding via Relation Graphs

Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang*

ICML 2023



02 Motivation

- The most dominant paradigm in KGC is to learn and operate on latent representations of entities and relations
  - Not clear if these embedding-based KGC methods **effectively capture the logical rules** that hold among the relations underlying the KG

- KGC can be viewed as a logical induction problem, where one seeks to derive probabilistic logical rules underlying a given KG
  - By learning entity-independent relational semantics, one can **generalize to unseen entities**

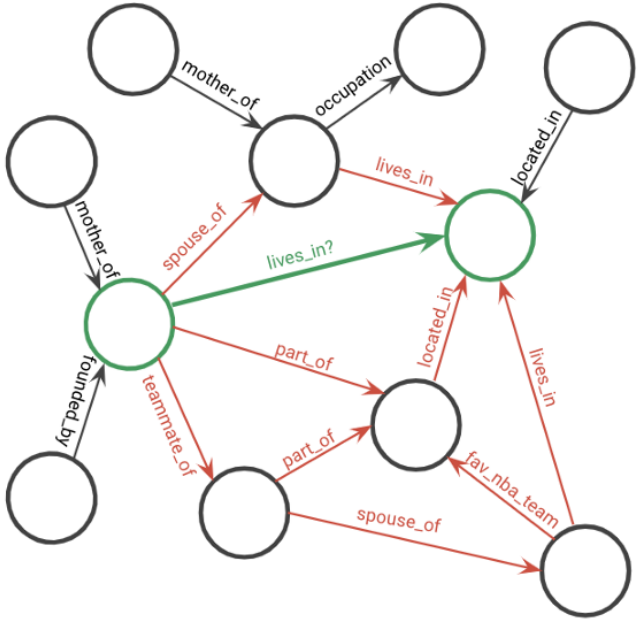
- In contrast, embedding-based KGC methods **assume a fixed set of entities** in a graph
  - Many real-world KGs are evolving, with new entities being added over time
  - The **ability to make predictions on new entities** without expensive re-training is essential for production-ready machine learning models

02 Contributions

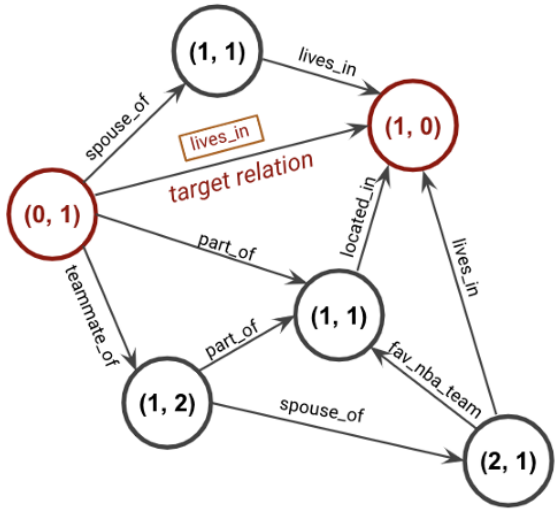
- Present a GNN framework **GraIL** (Graph Inductive Learning), that has an inductive bias to learn **entity-independent relational semantics**
  - GraIL can represent a useful subset of first-order logic
  - Naturally generalizes to unseen nodes, as the model learns to reason over subgraph structures independent of any particular node identities

- Introduce a series of **benchmark tasks for inductive KGC**
  - To test models with inductive capabilities for entities, new inductive datasets are constructed by sampling subgraphs from KG datasets

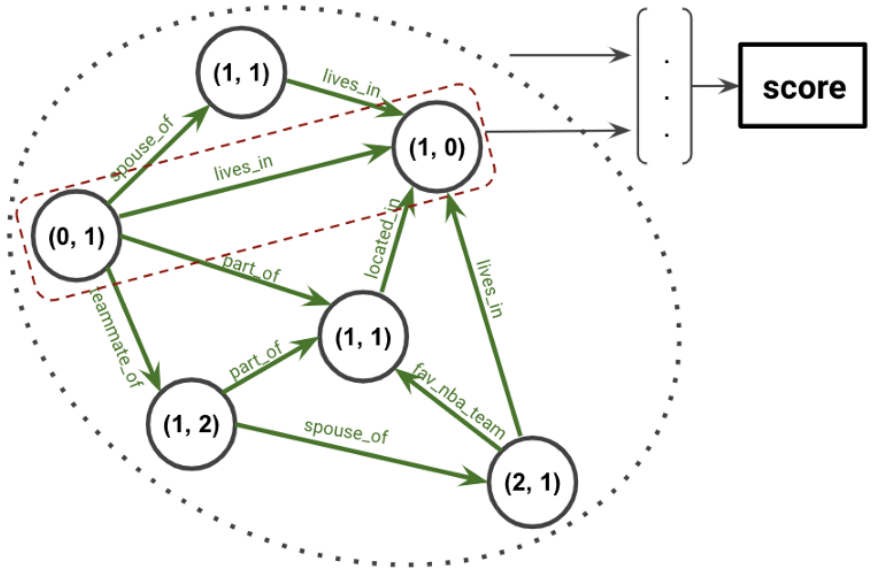
02 Overview of GraIL



1. Sample the enclosing sub-graph around the link to be predicted (target link).



2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.

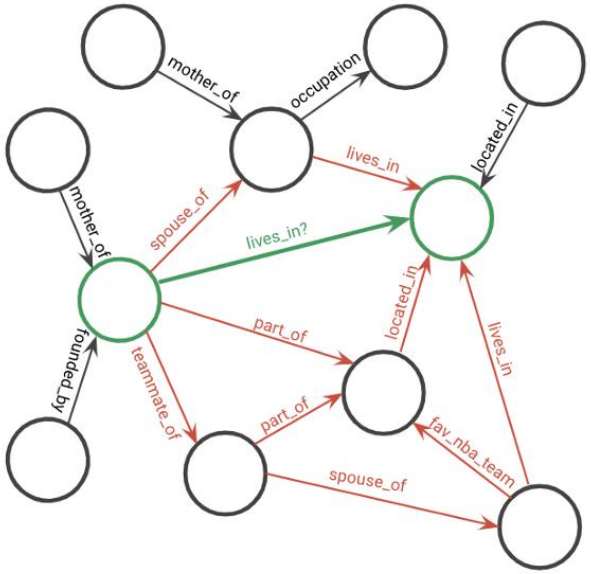


3. Pass messages across the (sub-)graph to predict a score indicating how strongly the structure around the target link supports its logical plausibility.

02 Subgraph Extraction

- Assume that local graph neighborhood of a particular triplet in the KG will **contain the logical evidence** needed to deduce the relation between the target nodes
  - Assume that the paths connecting two target nodes contain the information that could imply the target relation

- Extract the **enclosing subgraph** around the target nodes
  - Intersection of k-hop neighborhood of the two target nodes



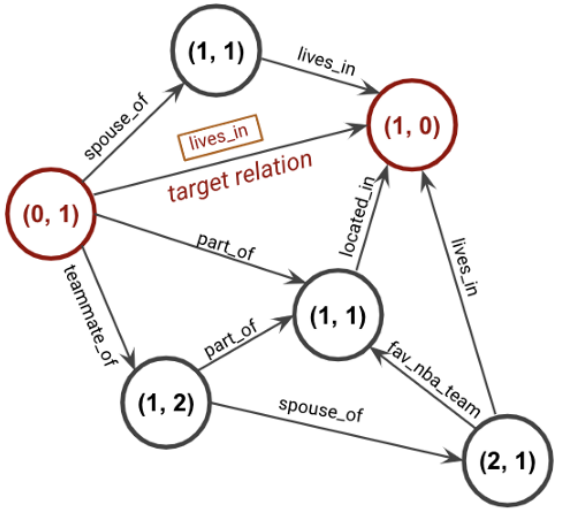
1. Sample the enclosing sub-graph around the link to be predicted (target link).

02 Node Labeling

- Designed to **capture the topological position** of each node with respect to the target nodes and reflect its structural role in the subgraph

- Each node i in the subgraph around nodes u and v is labeled with $\big(d(i,u), d(i,v)\big)$
  - $d(i,u)$ denotes the **shortest distance** between nodes i and u without counting any path through v

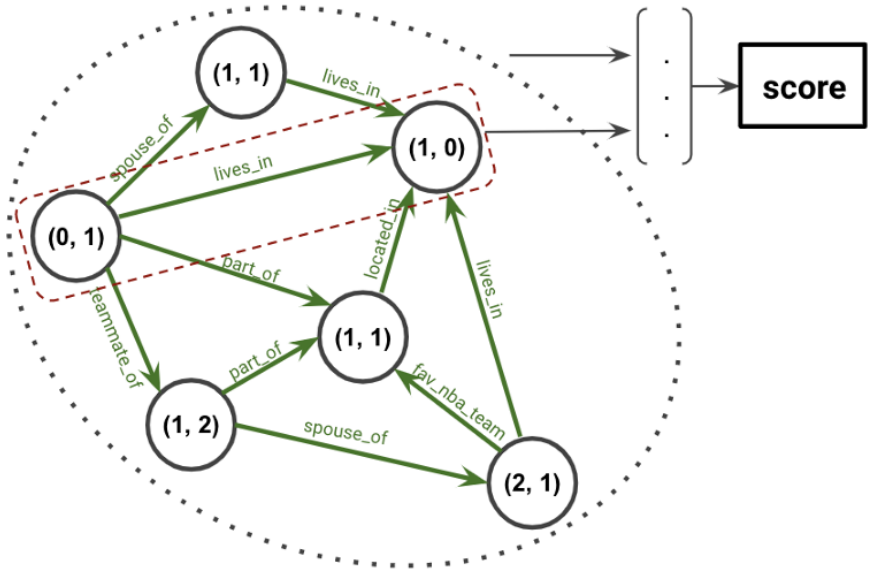
- The dimension of node features is **bounded by the number of hops** considered while extracting the enclosing subgraph



2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.

02 GNN Scoring

- Use a GNN to **score the likelihood** of the target triplet using the **extracted and labeled subgraph** around the target nodes
  - A node representation is iteratively updated by combining it with aggregation of it's neighbors' representations
- Use representations from the intermittent layers and the last layer
  - Allows flexible neighborhood ranges for each node



3. Pass messages across the (sub-)graph to predict a score indicating how strongly the structure around the target link supports its logical plausibility.

02 Experiments

Table 1. Inductive results (AUC-PR)

	WN18RR				FB15k-237				NELL-995			
	v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
Neural-LP	86.02	83.78	62.90	82.06	69.64	76.55	73.95	75.74	64.66	83.61	87.58	85.69
DRUM	86.02	84.05	63.20	82.06	69.71	76.44	74.03	76.20	59.86	83.99	87.71	85.94
RuleN	90.26	89.01	76.46	85.75	75.24	88.70	91.24	91.79	84.99	88.40	87.20	80.52
GraIL	**94.32**	**94.18**	**85.80**	**92.72**	**84.69**	**90.57**	**91.68**	**94.46**	**86.05**	**92.62**	**93.34**	**87.50**

Table 2. Inductive results (Hits@10)

	WN18RR				FB15k-237				NELL-995			
	v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
Neural-LP	74.37	68.93	46.18	67.13	52.92	58.94	52.90	55.88	40.78	78.73	82.71	**80.58**
DRUM	74.37	68.93	46.18	67.13	52.92	58.73	52.90	55.88	19.42	78.55	82.71	**80.58**
RuleN	80.85	78.23	53.39	71.59	49.76	77.82	**87.69**	85.60	53.50	81.75	77.26	61.35
GraIL	**82.45**	**78.68**	**58.43**	**73.41**	**64.15**	**81.80**	82.83	**89.29**	**59.50**	**93.25**	**91.41**	73.19

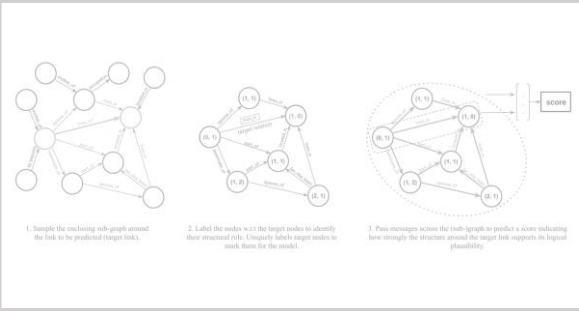
02 Conclusion

- Propose a GNN-based framework, **GraIL**, for inductive reasoning for entities
  - Unlike embedding-based approaches, GraIL is able to **predict links between nodes that were unseen during training**
  - GraIL brings an inductive bias complementary to the state-of-the-art KGC methods
- Provide **theoretical insights into the expressive power of GNNs** in encoding a useful subset of logical rules
- Open a new direction on **inductive reasoning** in context of knowledge graphs

Inductive Relation Prediction by Subgraph Reasoning

Komal K. Teru*, Etienne G. Denis, and William L. Hamilton

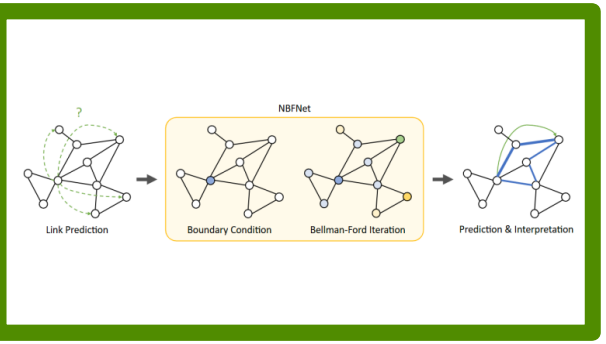
ICML 2020



Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction

Zhaocheng Zhu, Zuobai Zhang,
Louis-Pascal Xhonneux, and Jian Tang

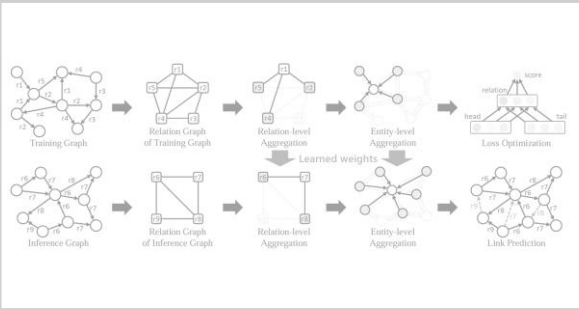
NeurIPS 2021



InGram: Inductive Knowledge Graph Embedding via Relation Graphs

Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang*

ICML 2023



03 Motivation

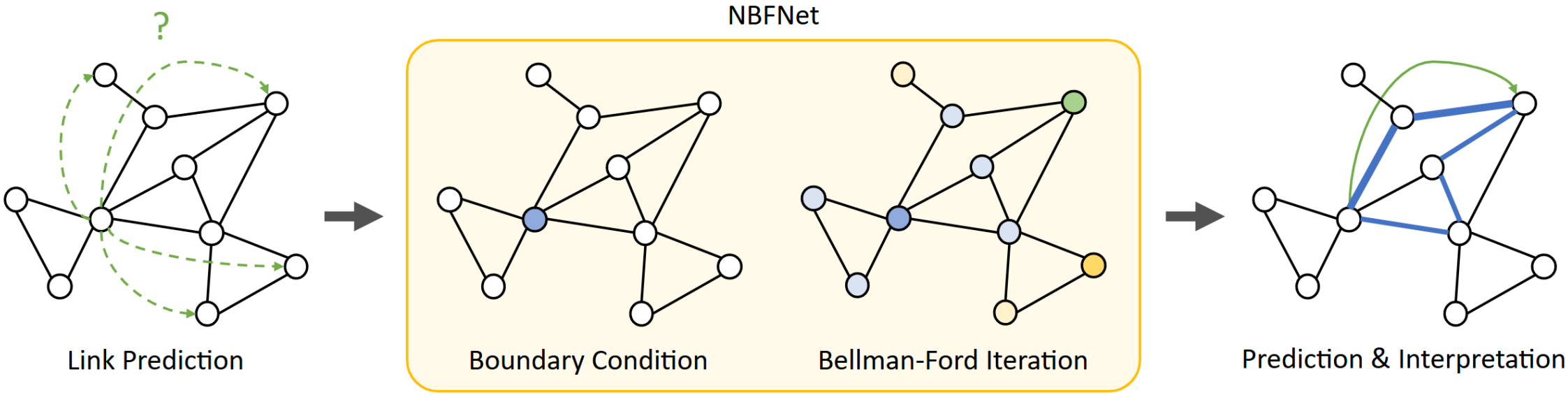
- Traditional methods of link prediction (e.g., Katz, PPR) can be **directly applied to new graphs**
  - Also enjoy good interpretability and scale up to large graphs
  - However, they **may not be optimal** for link prediction on real-world graphs

- Recent link prediction methods adopt GNNs to automatically extract important features from local neighborhoods for link prediction
  - However, these methods can only be applied to **predict new links on the training graph**
  - Some methods support inductive setting, but the **scalability of these methods is compromised**

03 Contributions

- Propose an approach that enjoys the advantages of **both traditional path-based methods** and **recent approaches based on GNNs**
  - Generalization in the inductive setting, interpretability, high model capacity and scalability
- Propose a **path formulation** that generalizes many link prediction methods and graph theory algorithms
  - This formulation can be **efficiently solved via the generalized Bellman-Ford algorithm**
- Propose Neural Bellman-Ford Networks (**NBFNet**), a GNN that solves the proposed path formulation with learned operators in the generalized Bellman-Ford algorithm
  - **Parameterizes the handcrafted operators** of generalized Bellman-Ford algorithm

03 Overview of NBFNet



03 Path Formulation

- Formulate the representation of a pair of nodes u and v as a **generalized sum of path representations** between u and v with a commutative summation operator $\oplus$
  - $\boldsymbol{h}_q(u,v) = \boldsymbol{h}_q(P_1) \oplus \boldsymbol{h}_q(P_2) \oplus \ldots \oplus \boldsymbol{h}_q\left(P_{|\mathcal{P}_{uv}|}\right)|_{P_i \in \mathcal{P}_{uv}}$
- Each path representation is defined as a **generalized product of the edge representations** in the path with the multiplicator operator $\otimes$
  - $\boldsymbol{h}_q\left(P = \left(e_1, e_2, \ldots, e_{|P|}\right)\right) = \boldsymbol{w}_q(e_1) \otimes \boldsymbol{w}_q(e_2) \otimes \cdots \otimes \boldsymbol{w}_q\left(e_{|P|}\right)$
- Such a formulation is capable of modeling several traditional link prediction methods and graph theory algorithms
  - Katz index, Personalized PageRank, graph distance, widest path, most reliable path

03 Generalized Bellman-Ford Algorithm

- The path formulation is computationally expensive since the **number of paths grows exponentially** with the path length

- A scalable solution is to use the generalized Bellman-Ford algorithm
  - computes the pair representation $h_q(u, v)$ for a given entity u, a given query relation q, and **all $v \in \mathcal{V}$ in parallel**
  - Reduces the total computation by the distributive property of multiplication over summation
  - $\boldsymbol{h}_q^{(0)}(u, v) \leftarrow \mathbf{1}_q(u = v)$
  - $\boldsymbol{h}_q^{(t)}(u, v) \leftarrow \left(\oplus_{(x,r,v)\in\mathcal{E}(v)} \boldsymbol{h}_q^{(t-1)}(u, x) \otimes \boldsymbol{w}_q(x, r, v)\right) \oplus \boldsymbol{h}_q^{(0)}(u, v)$

03 Neural Bellman-Ford Networks

- Parameterize the generalized Bellman-Ford algorithm with **neural functions**
  - Indicator function $\mathbf{1}_q(u = v)$ is replaced by INDICATOR function
  - Binary multiplication operator $\otimes$ is replaced by MESSAGE function
  - N-ary summation operator $\oplus$ is replaced by AGGREGATE function
  - $\boldsymbol{h}_q^{(0)}(u, v) \leftarrow \text{INDICATOR}(u, v, q)$
  - $\boldsymbol{h}_q^{(t)}(u, v) \leftarrow \text{AGGREGATE}\left(\left\{\text{MESSAGE}\left(\boldsymbol{h}_q^{(t-1)}(u, x), \boldsymbol{w}_q(x, r, v)\right) | (x, r, v) \in \mathcal{E}(v)\right\} \cup \left\{\boldsymbol{h}_q^{(0)}(u, v)\right\}\right)$

- NBFNet can be interpreted as a **GNN framework for learning pair representations**

03 Comparison of Operators

Class	Method	MESSAGE $\boldsymbol{w}_q(e_i) \otimes \boldsymbol{w}_q(e_j)$	AGGREGATE $\boldsymbol{h}_q(P_i) \oplus \boldsymbol{h}_q(P_j)$	INDICATOR $⓪_q, ①_q$	Edge Representation $\boldsymbol{w}_q(e)$
Traditional Link Prediction	Katz Index [30]	$\boldsymbol{w}_q(e_i) \times \boldsymbol{w}_q(e_j)$	$\boldsymbol{h}_q(P_i) + \boldsymbol{h}_q(P_j)$	$0, 1$	βw_e
	Personalized PageRank [42]	$\boldsymbol{w}_q(e_i) \times \boldsymbol{w}_q(e_j)$	$\boldsymbol{h}_q(P_i) + \boldsymbol{h}_q(P_j)$	$0, 1$	$\alpha w_{uv} / \sum_{v' \in \mathcal{N}(u)} w_{uv'}$
	Graph Distance [37]	$\boldsymbol{w}_q(e_i) + \boldsymbol{w}_q(e_j)$	$\min(\boldsymbol{h}_q(P_i), \boldsymbol{h}_q(P_j))$	$+\infty, 0$	w_e
Graph Theory Algorithms	Widest Path [4]	$\min(\boldsymbol{w}_q(e_i), \boldsymbol{w}_q(e_j))$	$\max(\boldsymbol{h}_q(P_i), \boldsymbol{h}_q(P_j))$	$-\infty, +\infty$	w_e
	Most Reliable Path [4]	$\boldsymbol{w}_q(e_i) \times \boldsymbol{w}_q(e_j)$	$\max(\boldsymbol{h}_q(P_i), \boldsymbol{h}_q(P_j))$	$0, 1$	w_e
Logic Rules	NeuralLP [69] / DRUM [46]	$\boldsymbol{w}_q(e_i) \times \boldsymbol{w}_q(e_j)$	$\boldsymbol{h}_q(P_i) + \boldsymbol{h}_q(P_j)$	0, 1	Weights learned by LSTM [23]
	NBFNet	Relational operators of knowledge graph embeddings [6, 68, 52]	Learned set aggregators [9]	Learned indicator functions	Learned relation embeddings

03 Experiments

Class	Method	FB15k-237					WN18RR				
		MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
Path-based	Path Ranking [35]	3521	0.174	0.119	0.186	0.285	22438	0.324	0.276	0.360	0.406
	NeuralLP [69]	-	0.240	-	-	0.362	-	0.435	0.371	0.434	0.566
	DRUM [46]	-	0.343	0.255	0.378	0.516	-	0.486	0.425	0.513	0.586
Embeddings	TransE [6]	357	0.294	-	-	0.465	3384	0.226	-	-	0.501
	DistMult [68]	254	0.241	0.155	0.263	0.419	5110	0.43	0.39	0.44	0.49
	ComplEx [58]	339	0.247	0.158	0.275	0.428	5261	0.44	0.41	0.46	0.51
	RotatE [52]	177	0.338	0.241	0.375	0.553	3340	0.476	0.428	0.492	0.571
	HAKE [76]	-	0.346	0.250	0.381	0.542	-	0.497	0.452	0.516	0.582
	LowFER [1]	-	0.359	0.266	0.396	0.544	-	0.465	0.434	0.479	0.526
GNNs	RGCN [48]	221	0.273	0.182	0.303	0.456	2719	0.402	0.345	0.437	0.494
	GraIL [55]	2053	-	-	-	-	2539	-	-	-	-
	NBFNet	**114**	**0.415**	**0.321**	**0.454**	**0.599**	**636**	**0.551**	**0.497**	**0.573**	**0.666**

Class	Method	FB15k-237				WN18RR			
		v1	v2	v3	v4	v1	v2	v3	v4
Path-based	NeuralLP [16]	0.529	0.589	0.529	0.559	0.744	0.689	0.462	0.671
	DRUM [46]	0.529	0.587	0.529	0.559	0.744	0.689	0.462	0.671
	RuleN [39]	0.498	0.778	0.877	0.856	0.809	0.782	0.534	0.716
GNNs	GraIL [55]	0.642	0.818	0.828	0.893	0.825	0.787	0.584	0.734
	NBFNet	**0.834**	**0.949**	**0.951**	**0.960**	**0.948**	**0.905**	**0.893**	**0.890**

03 Conclusion

- Present a **representation learning framework based on paths** for link prediction
  - Generalizes several traditional methods, and can be efficiently solved via the generalized Bellman-Ford algorithm

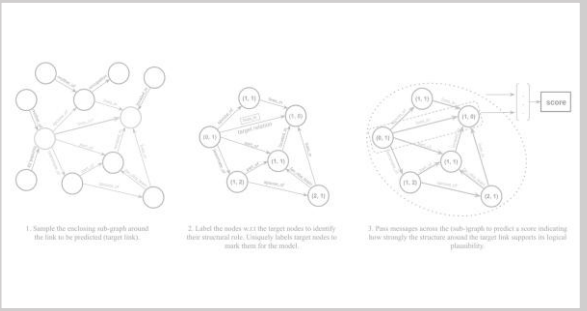
- To improve the capacity of the path formulation, **NBFNet** is proposed, which parameterizes the generalized Bellman-Ford algorithm with **learned functions**

- Experiments on knowledge graphs show that NBF outperforms a wide range of methods in **both transductive and inductive settings**

Inductive Relation Prediction by Subgraph Reasoning

Komal K. Teru*, Etienne G. Denis, and William L. Hamilton

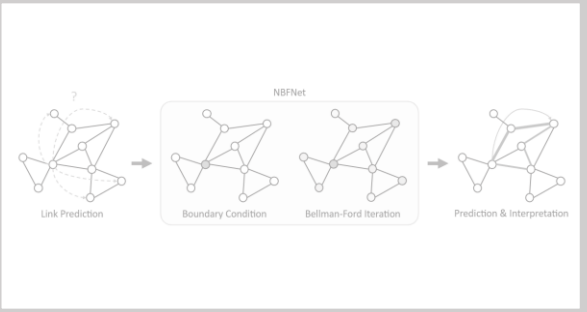
ICML 2020



Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction

Zhaocheng Zhu, Zuobai Zhang,
Louis-Pascal Xhonneux, and Jian Tang

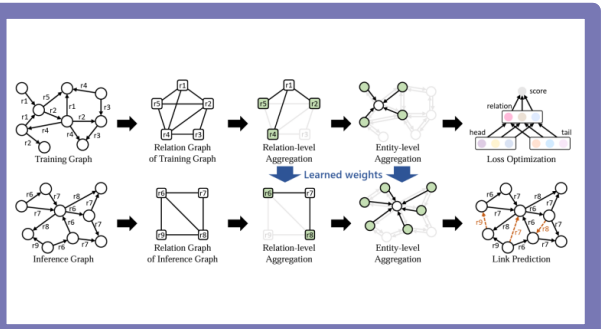
NeurIPS 2021



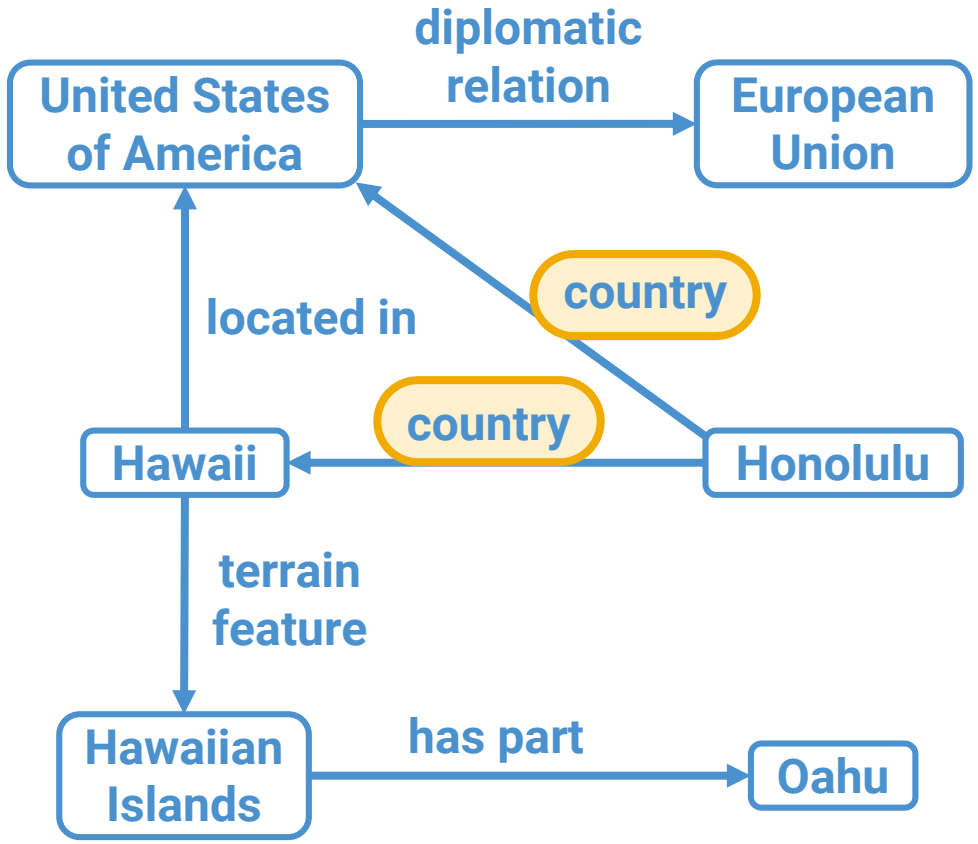
InGram: Inductive Knowledge Graph Embedding via Relation Graphs

Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang*

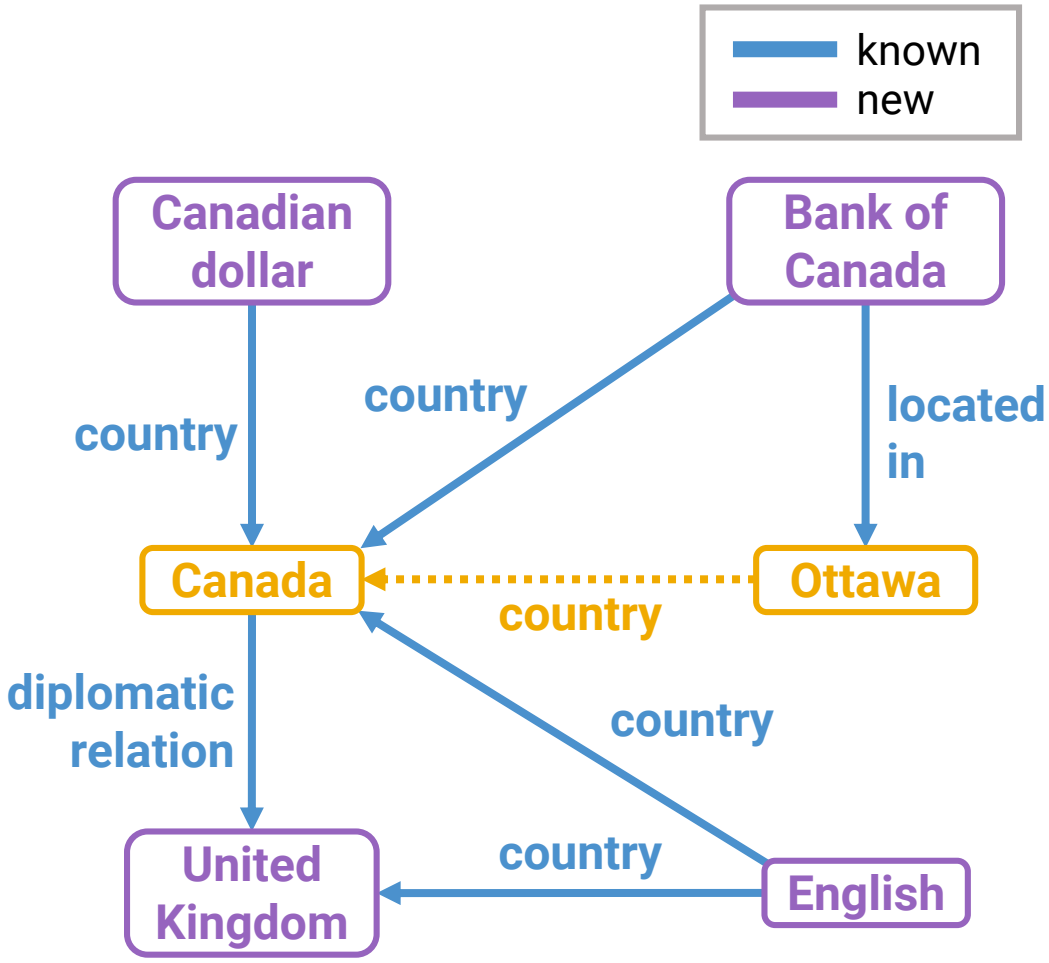
ICML 2023



04 Transductive Inference for Relations

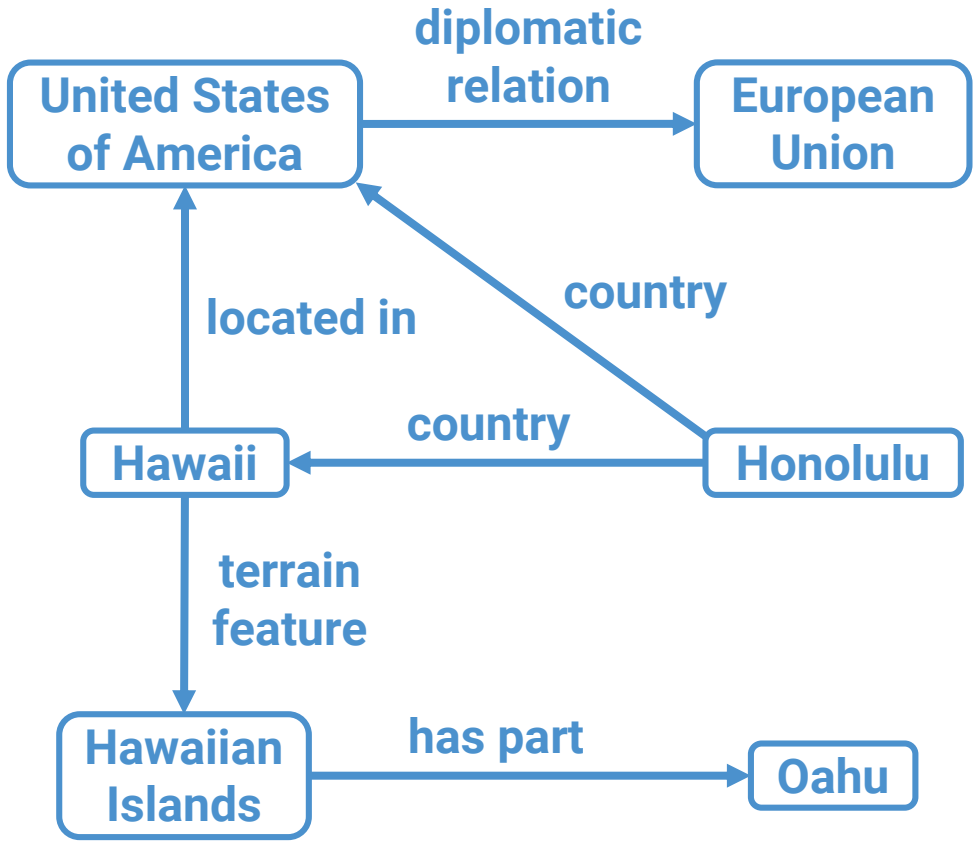


Training Graph

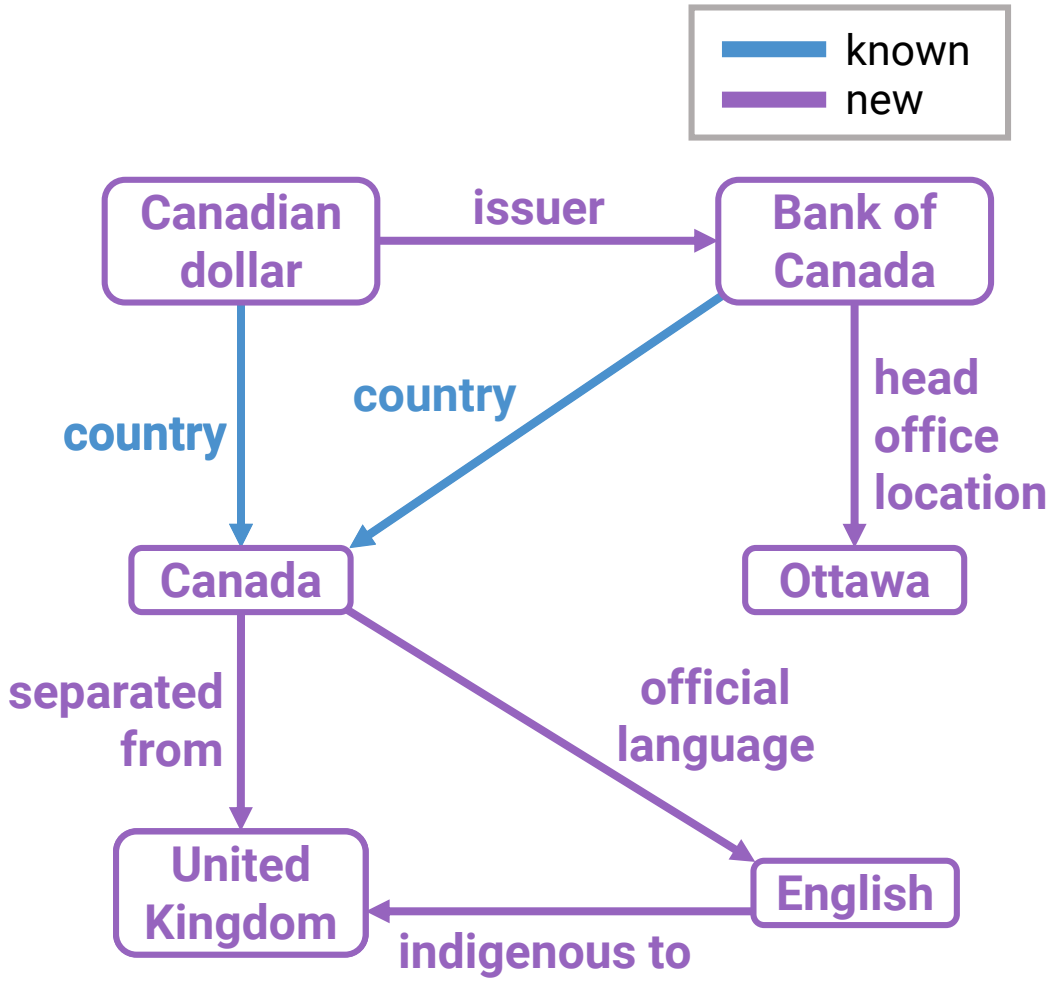


Inference Graph

04 Semi-Inductive Inference for Relations

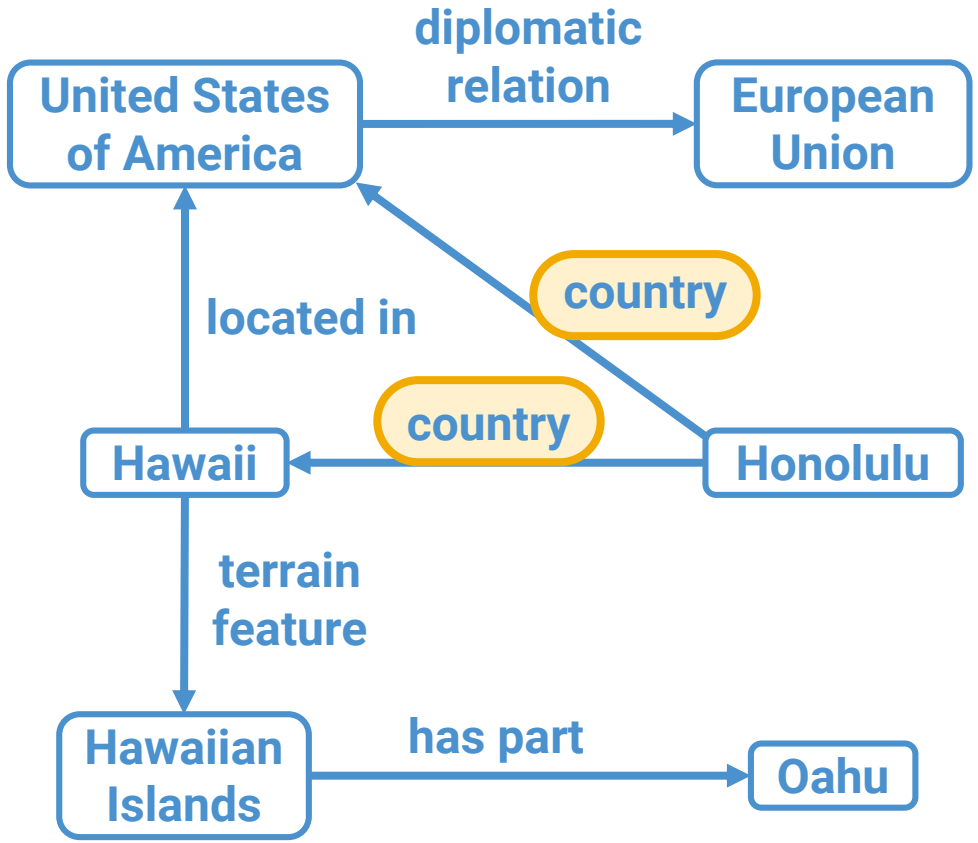


Training Graph

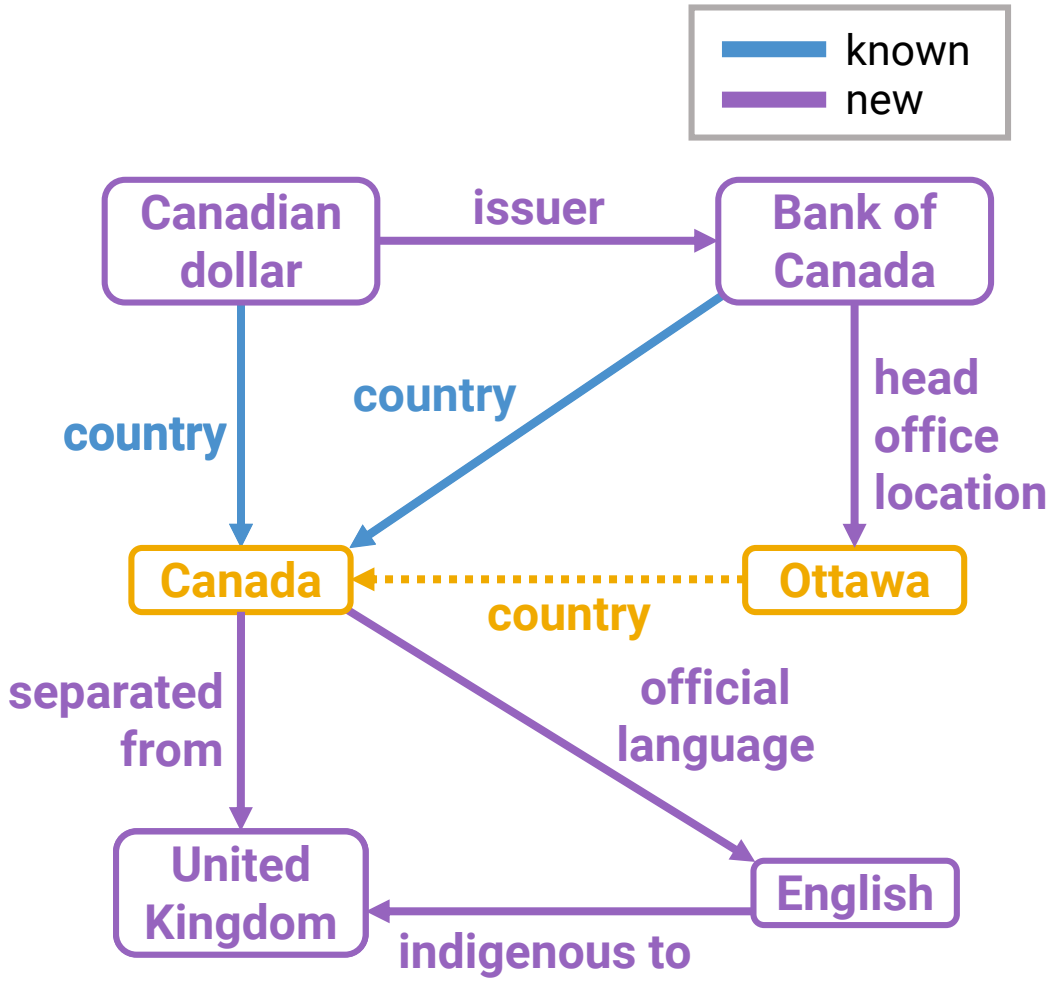


Inference Graph

04 Semi-Inductive Inference for Relations

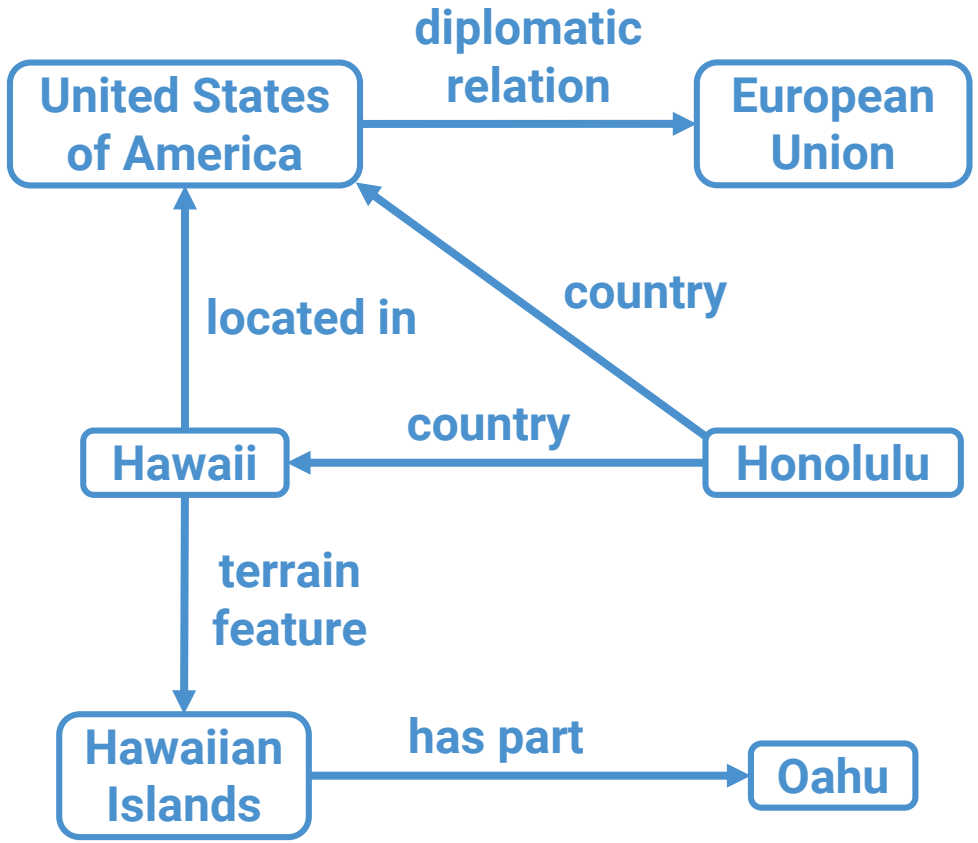


Training Graph

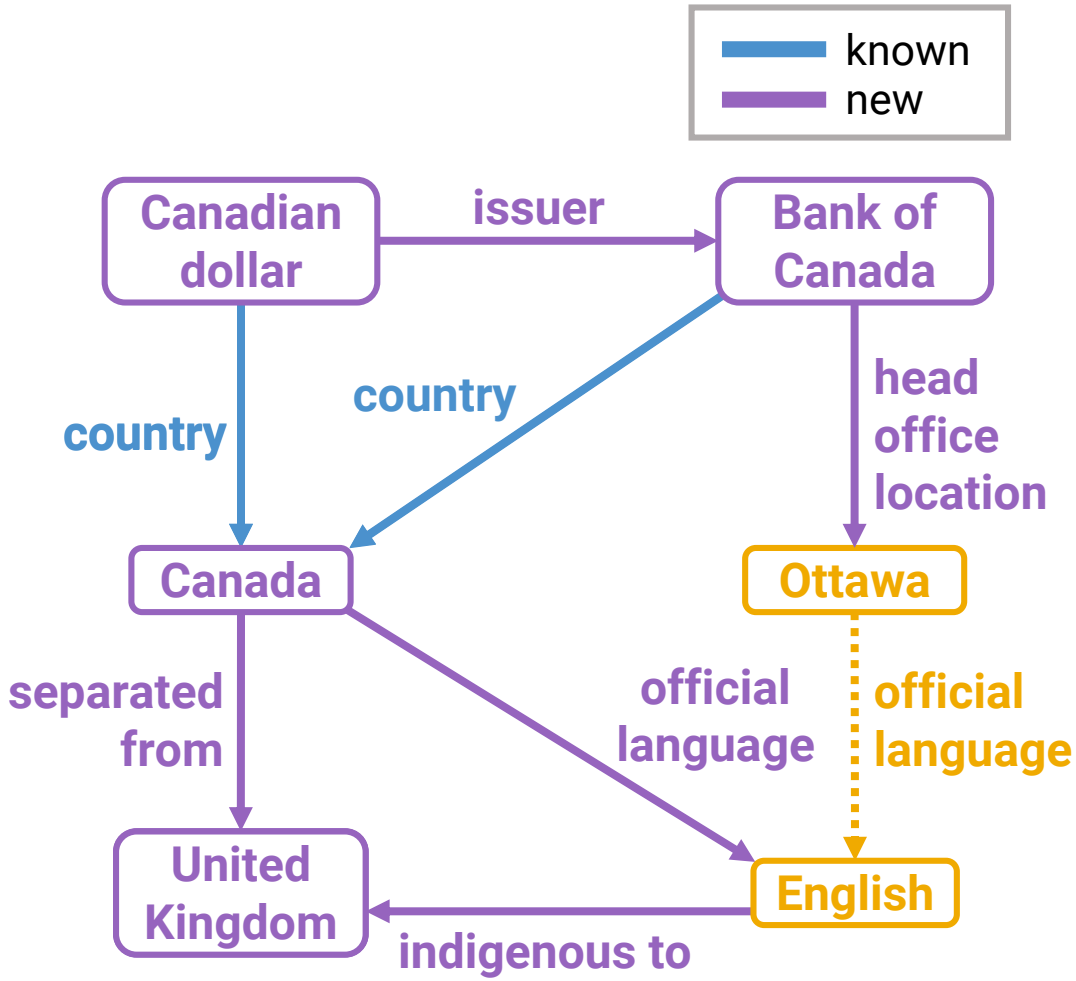


Inference Graph

04 Semi-Inductive Inference for Relations

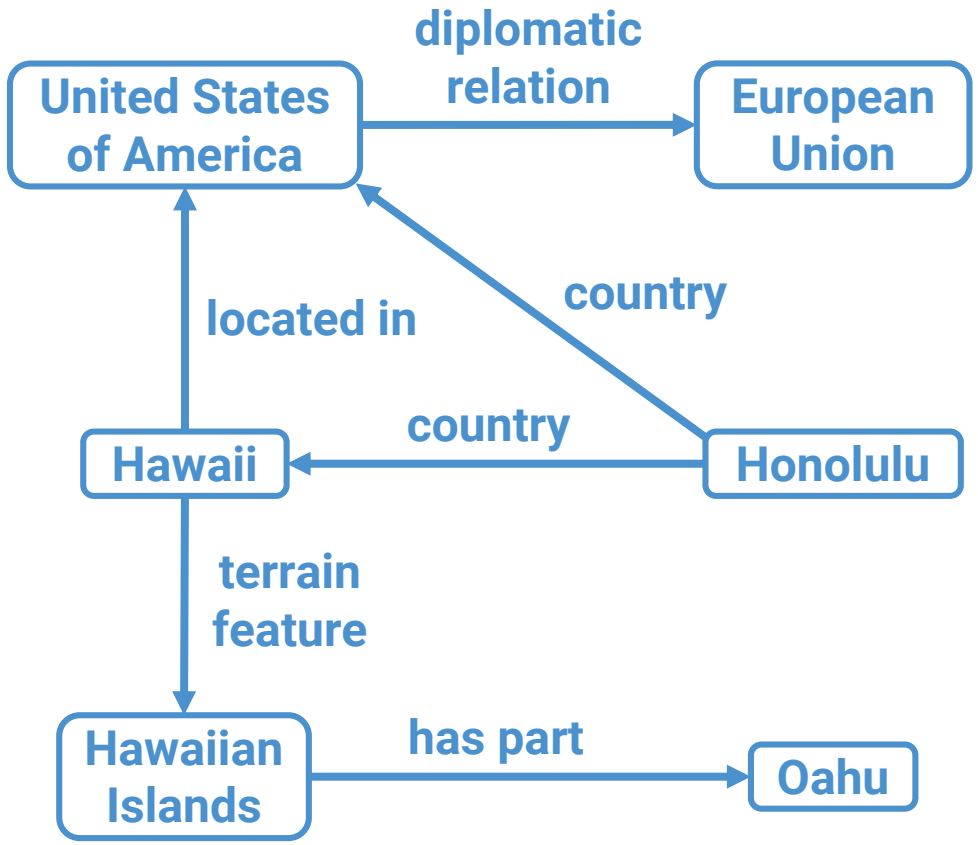


Training Graph

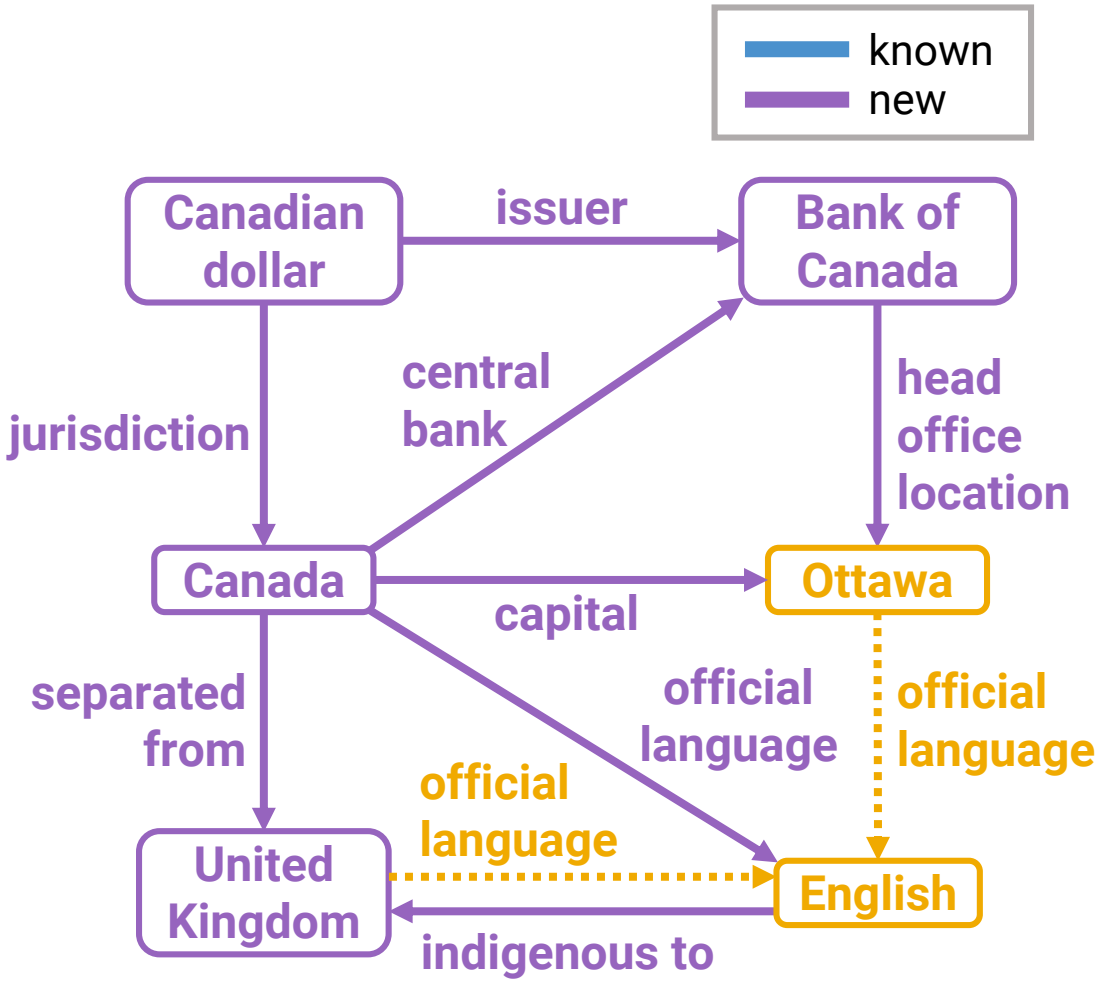


Inference Graph

04 Inductive Inference for Relations



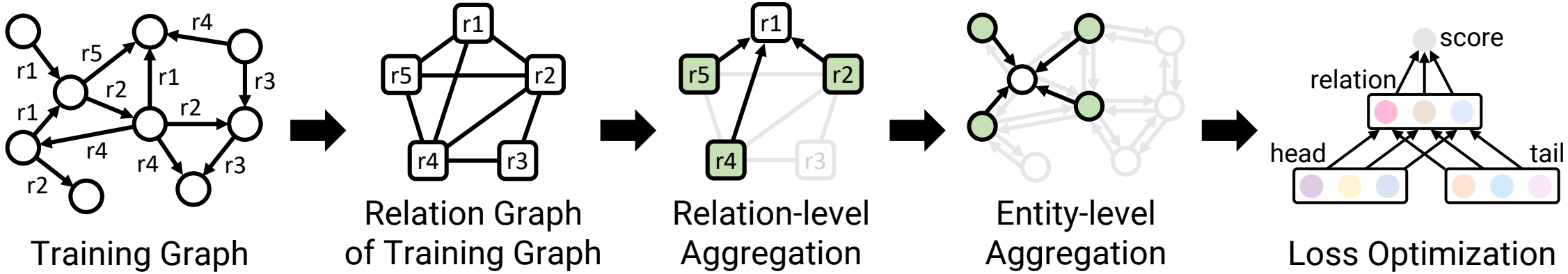
Training Graph



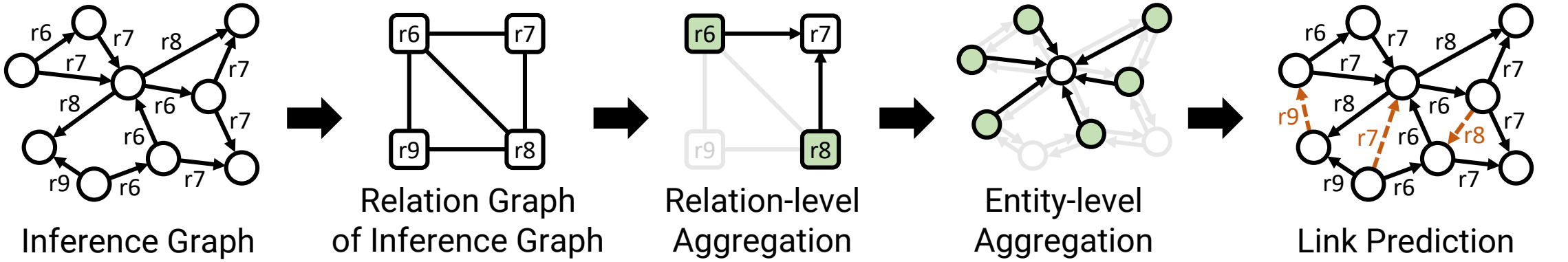
Inference Graph

04 Overview of InGram

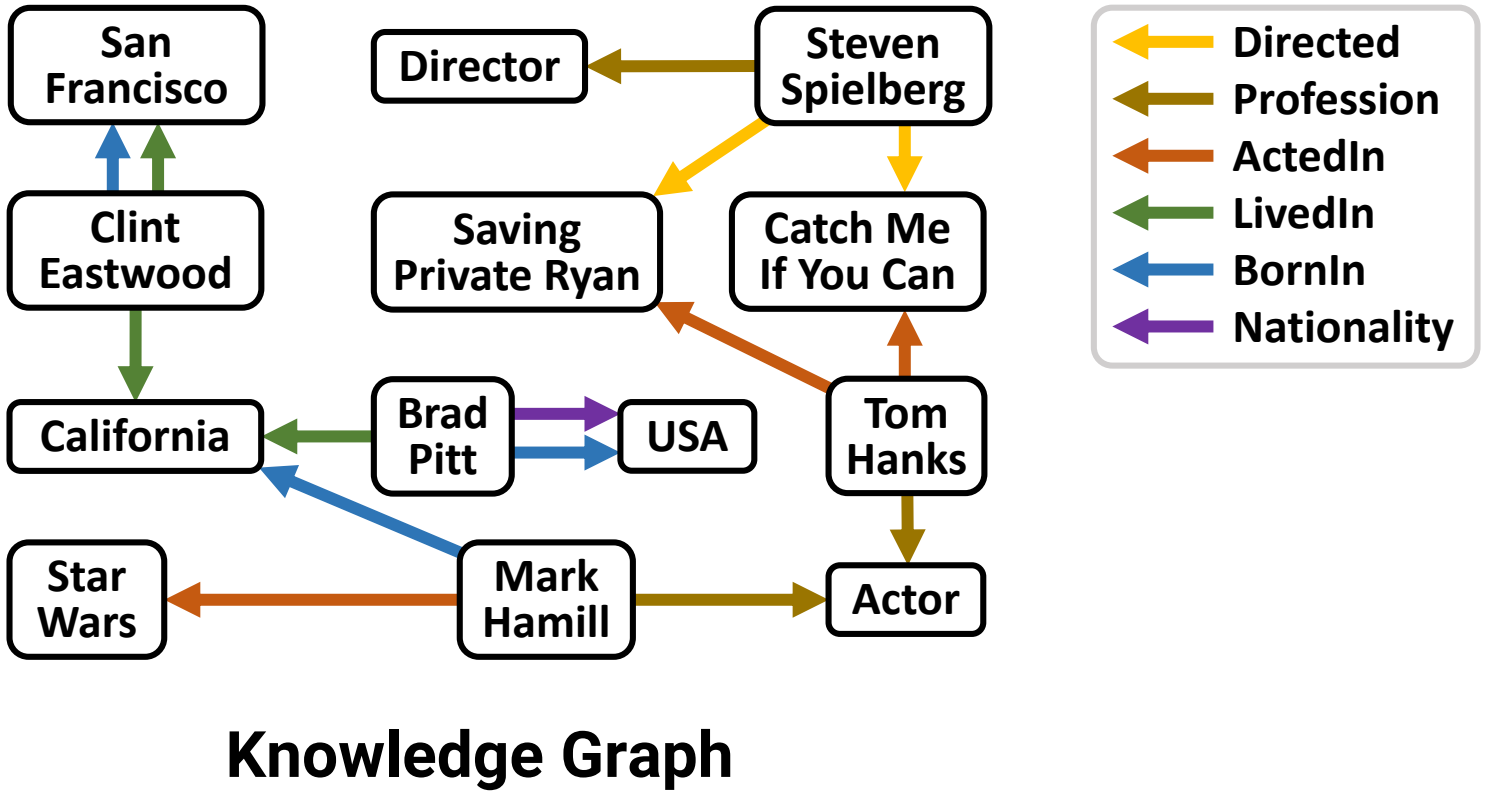
Training Time:



Inference Time:

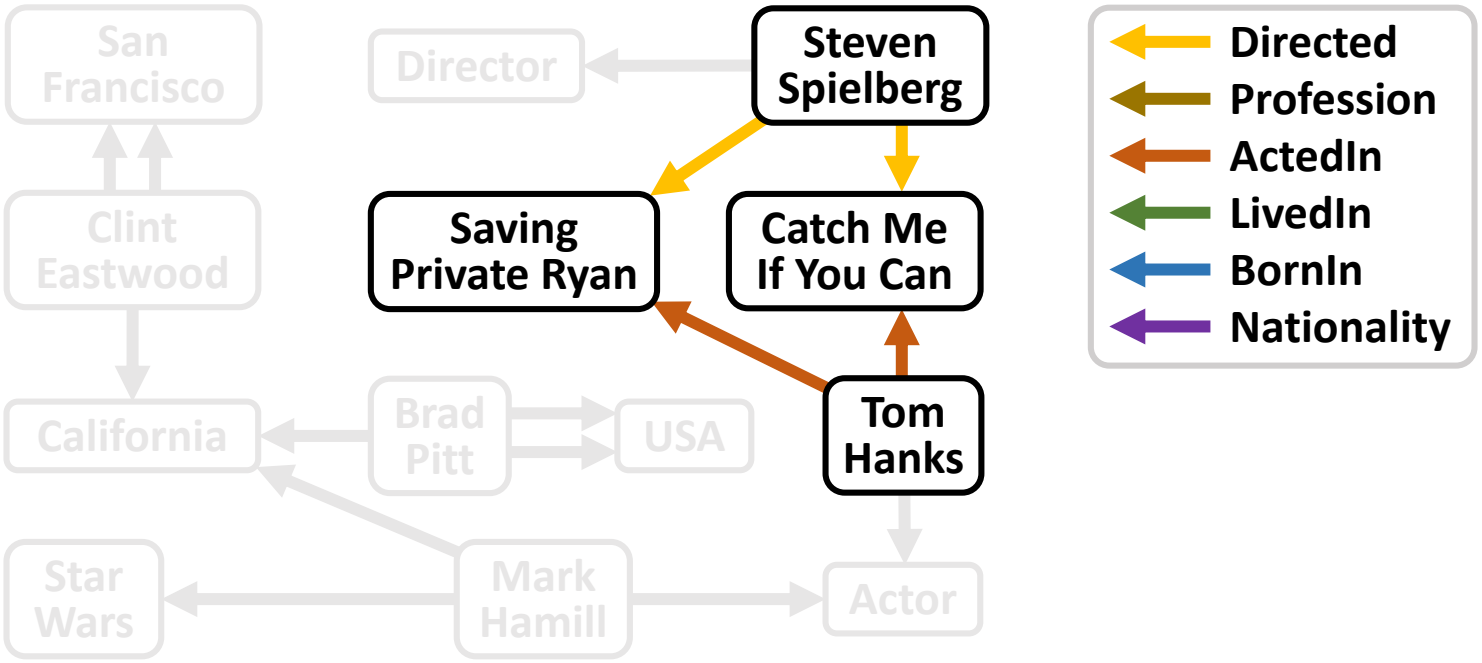


04 Relation Graph



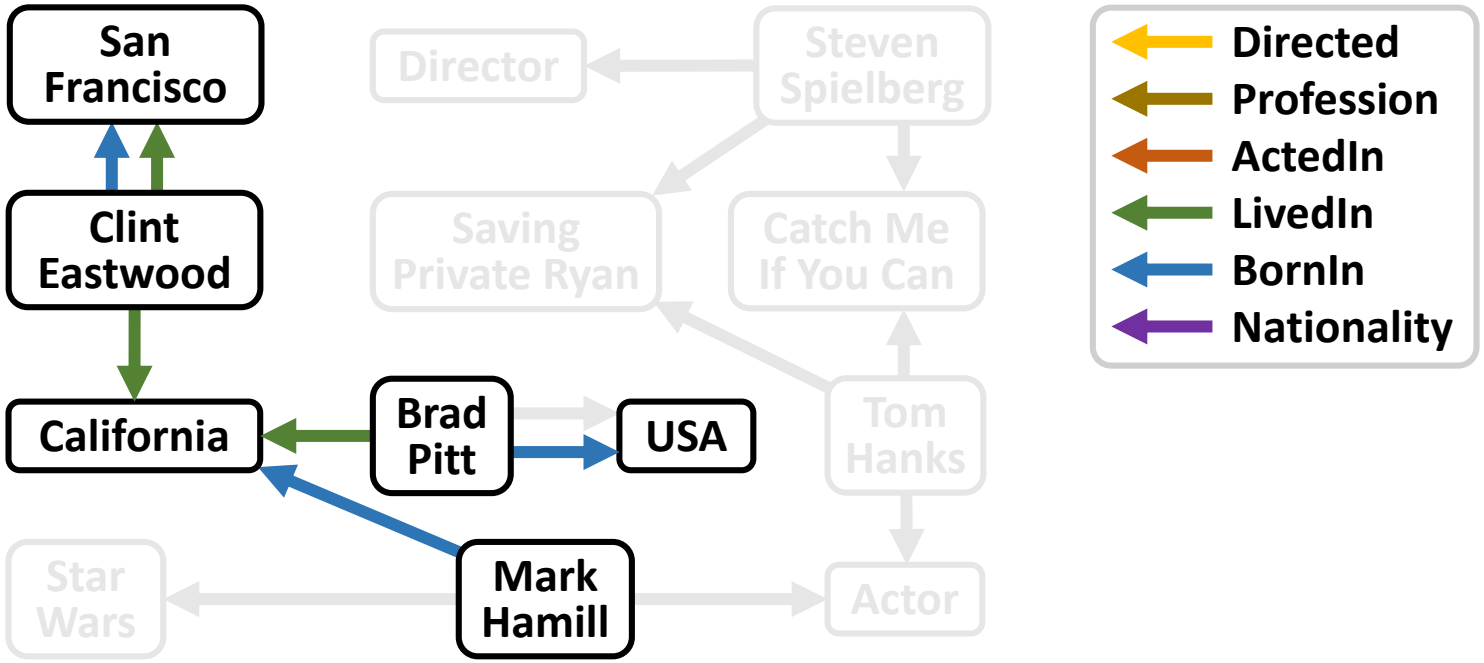
Knowledge Graph

04 Relation Graph



Knowledge Graph

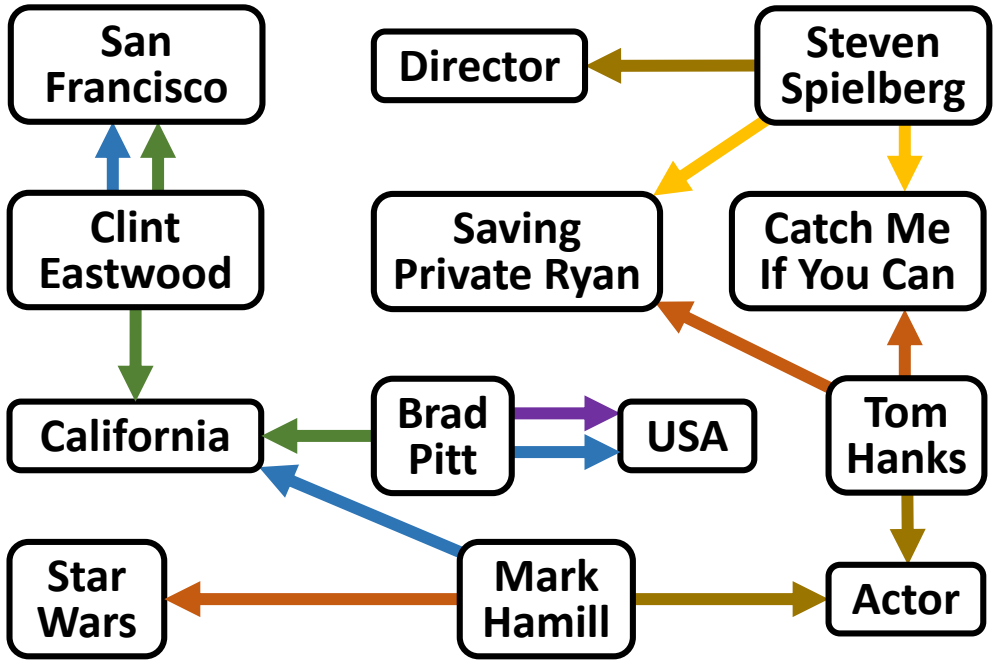
04 Relation Graph



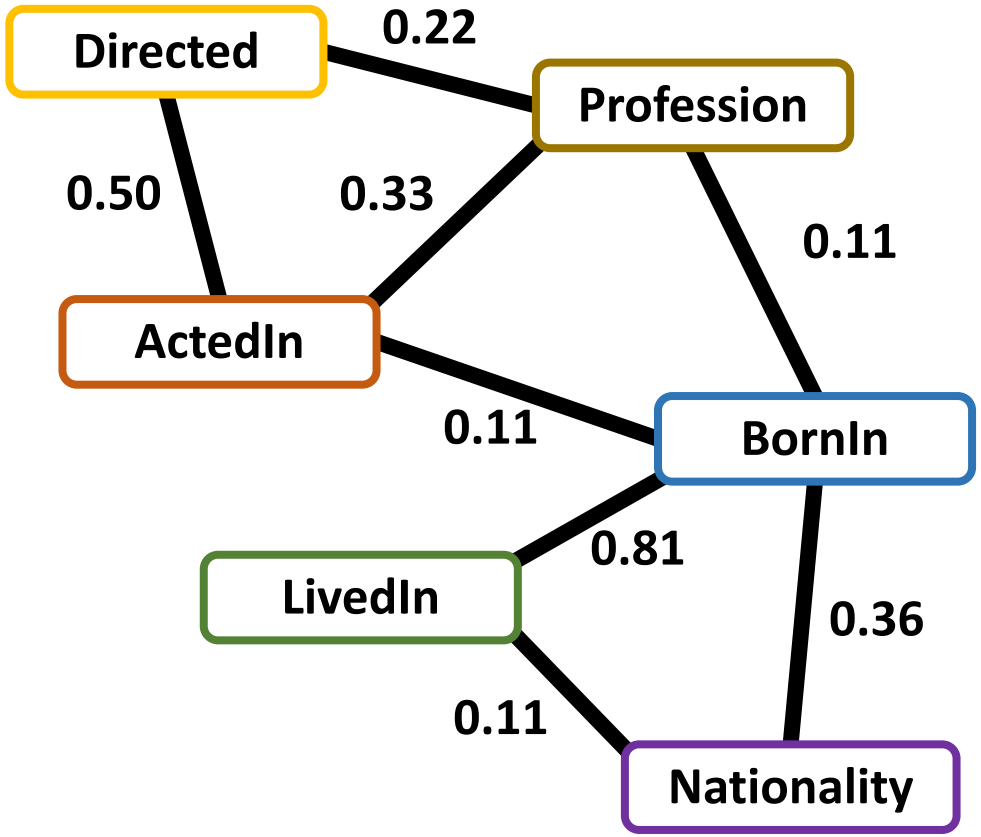
Knowledge Graph

04 Relation Graph

$$A = E_h^\top D_h^{-2} E_h + E_t^\top D_t^{-2} E_t$$

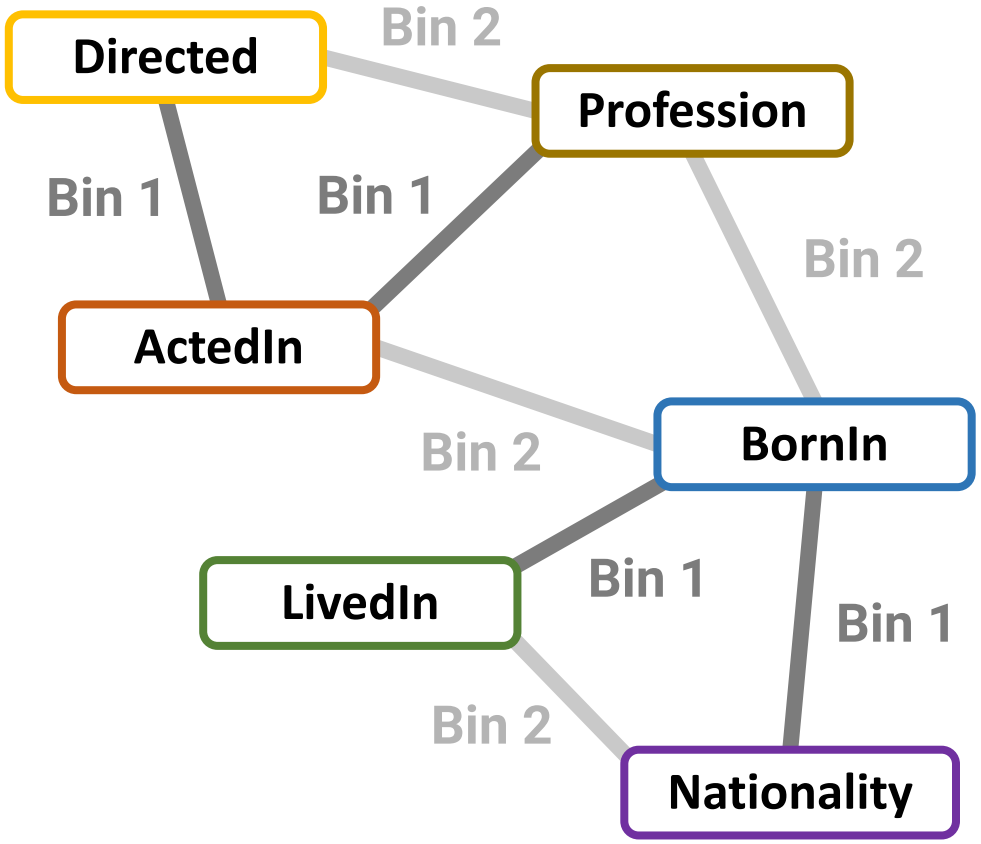


Knowledge Graph



Relation Graph

04 Relation Graph with Binning



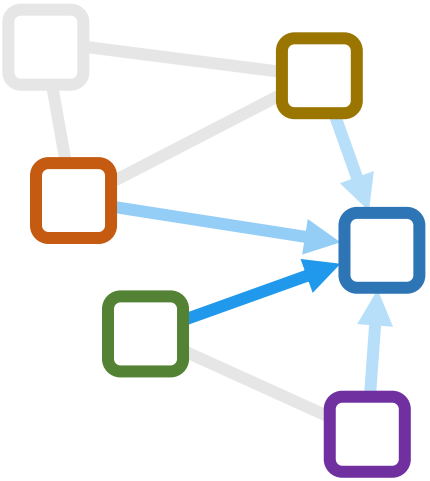
Relation Graph with Binning

Relation Pairs	Weights	
(BornIn, LivedIn)	0.81	Bin 1
(ActedIn, Directed)	0.50	
(BornIn, Nationality)	0.36	
(ActedIn, Profession)	0.33	
(Directed, Profession)	0.22	Bin 2
(ActedIn, BornIn)	0.11	
(BornIn, Profession)	0.11	
(LivedIn, Nationality)	0.11	

Relation Pairs Sorted by Weights

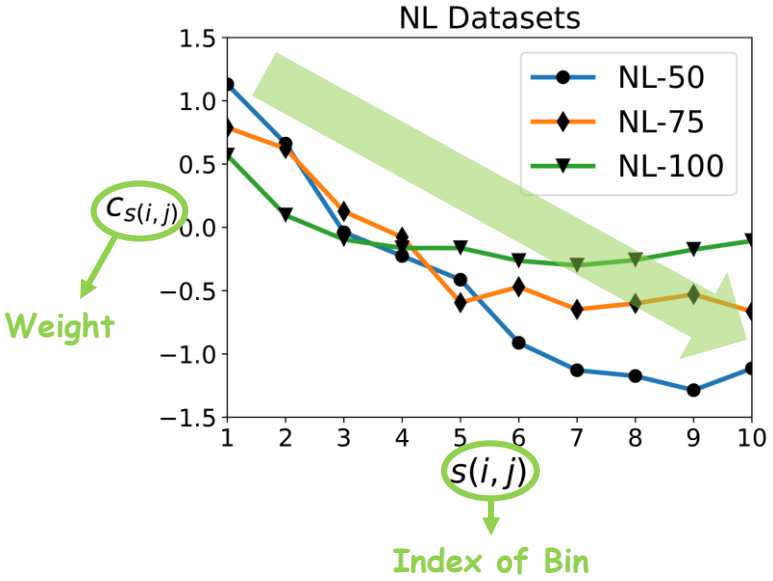
04 Relation-level Aggregation

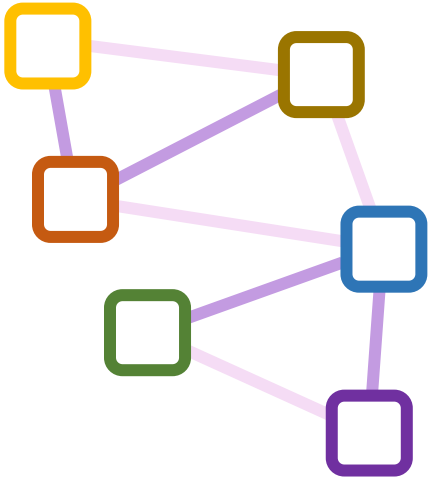
- Aggregate neighboring relations' embedding vectors
  - $\boldsymbol{z}_i^{(l+1)} = \sigma\left(\sum_{r_j \in \mathcal{N}_i} \alpha_{ij}^{(l)} \boldsymbol{W}^{(l)} \boldsymbol{z}_j^{(l)}\right)$



Local structure

- Consider the **relative importance** and the **affinity weight**
  - $\alpha_{ij}^{(l)} = \frac{\exp\left(\psi^{(l)}\left(\left[\boldsymbol{z}_i^{(l)} \| \boldsymbol{z}_j^{(l)}\right]\right) + c_{s(i,j)}^{(l)}\right)}{\sum_{r_{j'} \in \mathcal{N}_i} \exp\left(\psi^{(l)}\left(\left[\boldsymbol{z}_i^{(l)} \| \boldsymbol{z}_{j'}^{(l)}\right]\right) + c_{s(i,j')}^{(l)}\right)}$
  - $\psi^{(l)}(\boldsymbol{x}) = \boldsymbol{y}^{(l)} \sigma\left(\boldsymbol{P}^{(l)} \boldsymbol{x}\right)$

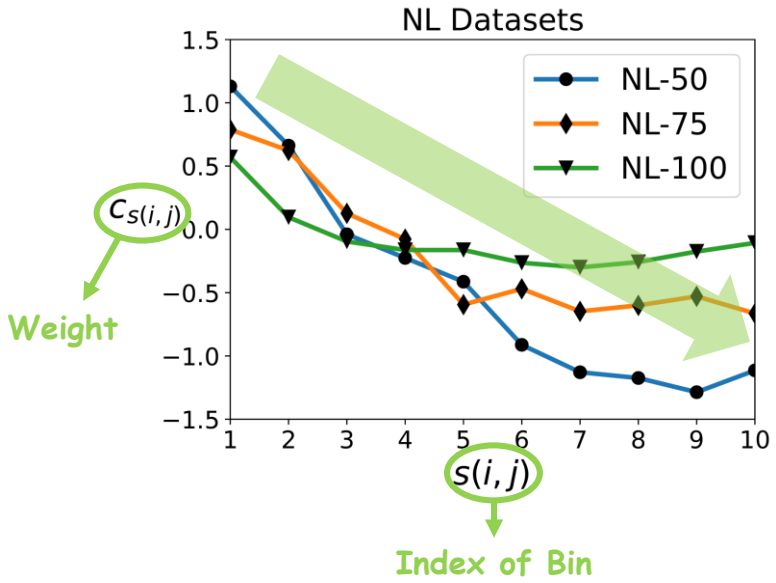


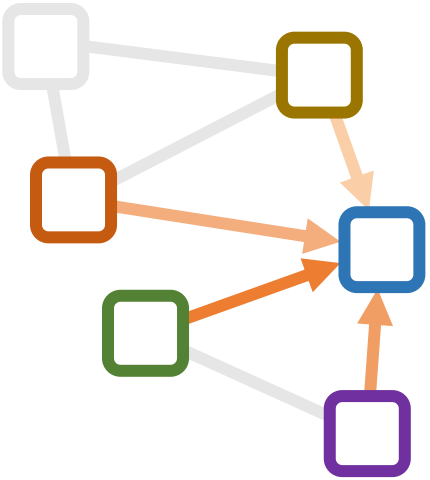


Global level of affinity

04 Relation-level Aggregation

- Aggregate neighboring relations' embedding vectors
  - $\boldsymbol{z}_i^{(l+1)} = \sigma\left(\sum_{r_j \in \mathcal{N}_i} \alpha_{ij}^{(l)} \boldsymbol{W}^{(l)} \boldsymbol{z}_j^{(l)}\right)$
- Consider the **relative importance** and the **affinity weight**
  - $\alpha_{ij}^{(l)} = \frac{\exp\left(\psi^{(l)}\left(\left[\boldsymbol{z}_i^{(l)} \| \boldsymbol{z}_j^{(l)}\right]\right) + c_{s(i,j)}^{(l)}\right)}{\sum_{r_{j'} \in \mathcal{N}_i} \exp\left(\psi^{(l)}\left(\left[\boldsymbol{z}_i^{(l)} \| \boldsymbol{z}_{j'}^{(l)}\right]\right) + c_{s(i,j')}^{(l)}\right)}$
  - $\psi^{(l)}(\boldsymbol{x}) = \boldsymbol{y}^{(l)} \sigma\left(\boldsymbol{P}^{(l)} \boldsymbol{x}\right)$

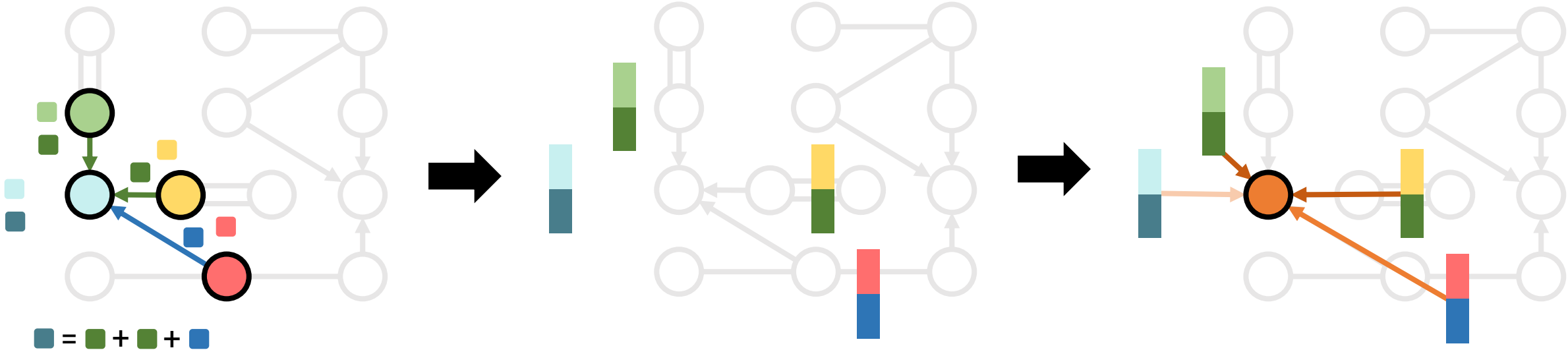




04 Entity-level Aggregation

- Compute an entity embedding by considering its **own vector**, its **neighbors' embeddings**, and its **adjacent relations**

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\beta_{ii}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right] + \sum_{v_j \in \widehat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \beta_{ijk}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)$$



04 Entity-level Aggregation

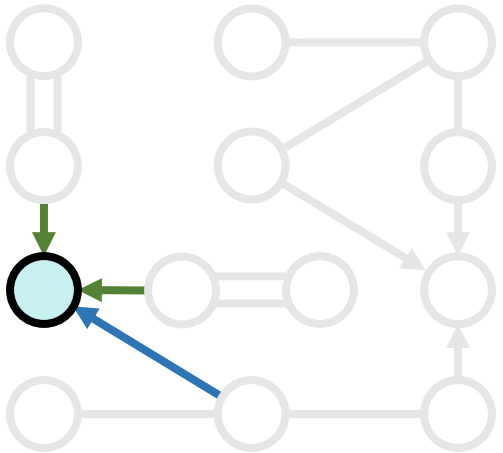
- Consider the **entity itself** and its **adjacent relations**

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\beta_{ii}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right] + \sum_{v_j \in \hat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \beta_{ijk}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)$$

$$\bar{\boldsymbol{z}}_i^{(L)} = \sum_{v_j \in \hat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \frac{\boldsymbol{z}_k^{(L)}}{\sum_{v_j \in \hat{\mathcal{N}}_i}|\mathcal{R}_{ji}|}$$

$$\beta_{ii}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right)/\lambda$$

$$\beta_{ijk}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)\right)/\lambda$$



$$\hat{\psi}^{(l)}(\boldsymbol{x}) = \hat{\boldsymbol{y}}^{(l)}\sigma\left(\widehat{\boldsymbol{P}}^{(l)}\boldsymbol{x}\right)$$

$$\lambda = \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right) + \sum_{v_{j'} \in \hat{\mathcal{N}}_i}\sum_{r_{k'} \in \mathcal{R}_{j'i}} \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_{j'}^{(l)} \| \boldsymbol{z}_{k'}^{(L)}\right]\right)\right)$$

04 Entity-level Aggregation

- Consider **neighbors' embedding vectors** and their adjacent **relations**

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\beta_{ii}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right] + \sum_{v_j \in \widehat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \beta_{ijk}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)$$

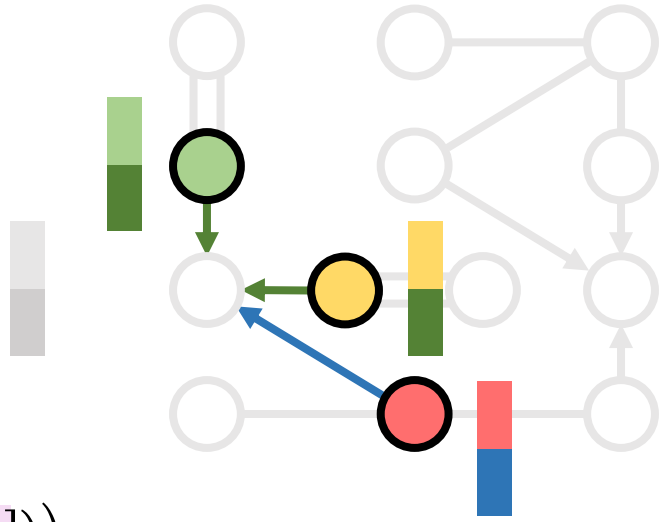
$$\bar{\boldsymbol{z}}_i^{(L)} = \sum_{v_j \in \widehat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \frac{\boldsymbol{z}_k^{(L)}}{\sum_{v_j \in \widehat{\mathcal{N}}_i}|\mathcal{R}_{ji}|}$$

$$\beta_{ii}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right)/\lambda$$

$$\beta_{ijk}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)\right)/\lambda$$

$$\hat{\psi}^{(l)}(\boldsymbol{x}) = \hat{\boldsymbol{y}}^{(l)}\sigma\left(\widehat{\boldsymbol{P}}^{(l)}\boldsymbol{x}\right)$$

$$\lambda = \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right) + \sum_{v_{j'} \in \widehat{\mathcal{N}}_i}\sum_{r_{k'} \in \mathcal{R}_{j'i}} \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_{j'}^{(l)} \| \boldsymbol{z}_{k'}^{(L)}\right]\right)\right)$$



04 Entity-level Aggregation

- Consider **the entity itself**, **its neighbors**, and the **relations**

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\beta_{ii}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right] + \sum_{v_j \in \widehat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \beta_{ijk}^{(l)}\widehat{\boldsymbol{W}}^{(l)}\left[\boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)$$

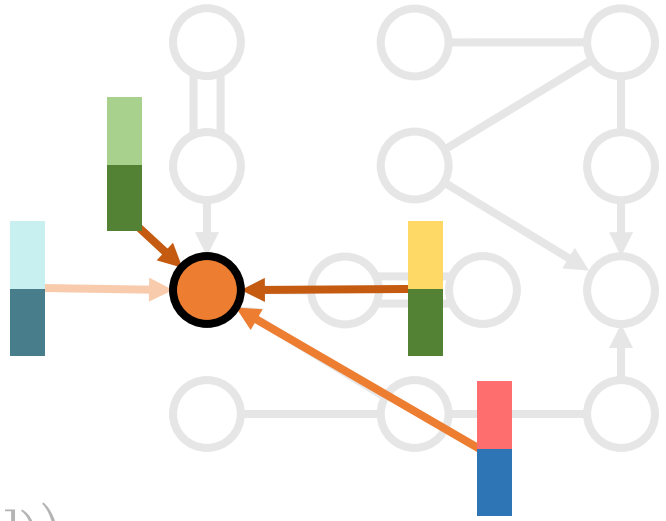
$$\bar{\boldsymbol{z}}_i^{(L)} = \sum_{v_j \in \widehat{\mathcal{N}}_i}\sum_{r_k \in \mathcal{R}_{ji}} \frac{\boldsymbol{z}_k^{(L)}}{\sum_{v_j \in \widehat{\mathcal{N}}_i}|\mathcal{R}_{ji}|}$$

$$\beta_{ii}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right)/\lambda$$

$$\beta_{ijk}^{(l)} = \exp\left(\hat{\psi}^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_j^{(l)} \| \boldsymbol{z}_k^{(L)}\right]\right)\right)/\lambda$$

$$\hat{\psi}^{(l)}(\boldsymbol{x}) = \hat{\boldsymbol{y}}^{(l)}\sigma\left(\widehat{\boldsymbol{P}}^{(l)}\boldsymbol{x}\right)$$

$$\lambda = \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_i^{(l)} \| \bar{\boldsymbol{z}}_i^{(L)}\right]\right)\right) + \sum_{v_{j'} \in \widehat{\mathcal{N}}_i}\sum_{r_{k'} \in \mathcal{R}_{j'i}} \exp\left(\psi^{(l)}\left(\left[\boldsymbol{h}_i^{(l)} \| \boldsymbol{h}_{j'}^{(l)} \| \boldsymbol{z}_{k'}^{(L)}\right]\right)\right)$$



04 Modeling Relation-Entity Interactions

- Final embedding vectors computation

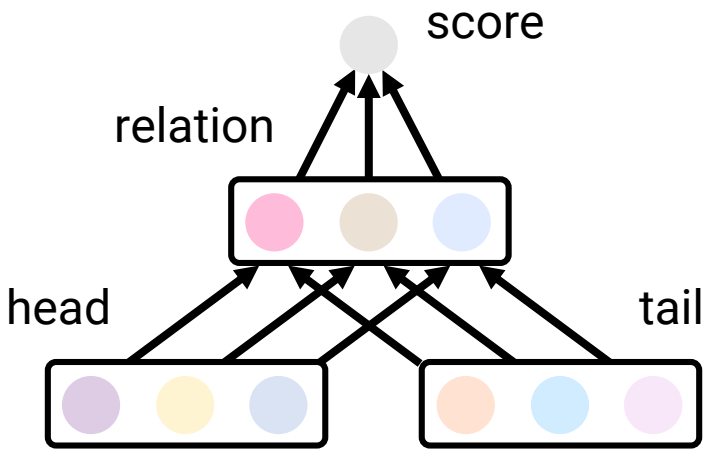
$$\boldsymbol{z}_k = \boldsymbol{M}\boldsymbol{z}_k^{(L)} \text{ and } \boldsymbol{h}_i = \widehat{\boldsymbol{M}}\boldsymbol{h}_i^{(\hat{L})}$$

- Scoring function

$$f(v_i, r_k, v_j) = \boldsymbol{h}_i^{\top} \operatorname{diag}(\overline{\boldsymbol{W}}\boldsymbol{z}_k)\boldsymbol{h}_j$$

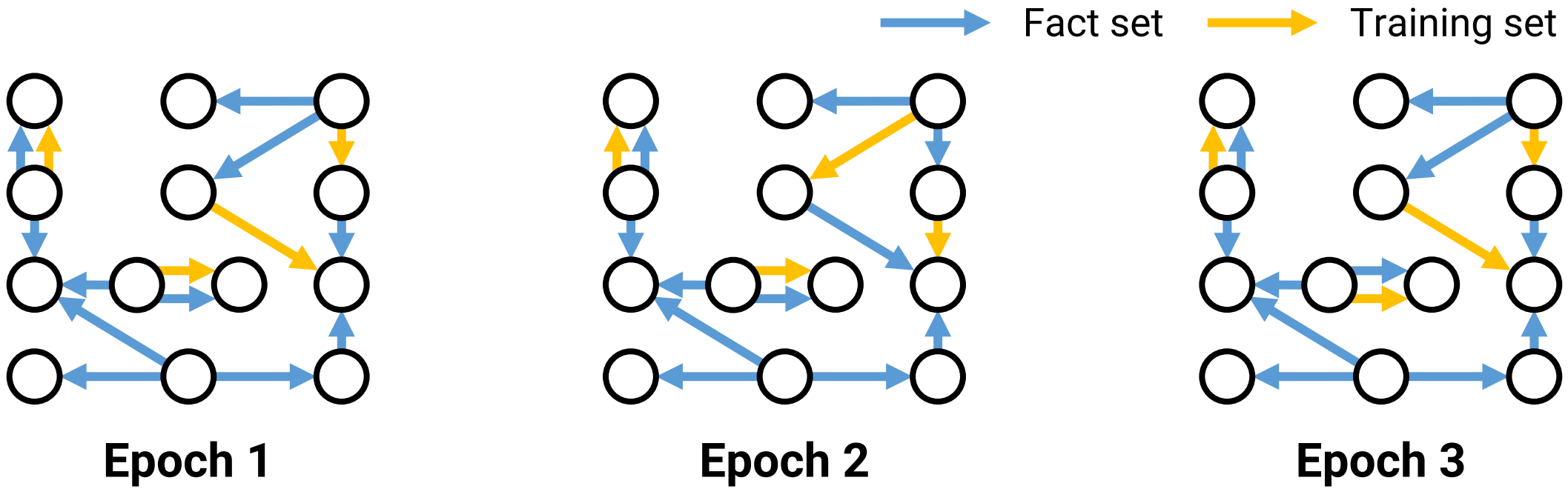
- Loss

$$\sum_{(v_i, r_k, v_j) \in \mathcal{T}_{\text{tr}}} \sum_{(\mathring{v}_i, r_k, \mathring{v}_j) \in \mathring{\mathcal{T}}_{\text{tr}}} \max\left(0, \gamma - f(v_i, r_k, v_j) + f\left(\mathring{v}_i, r_k, \mathring{v}_j\right)\right)$$



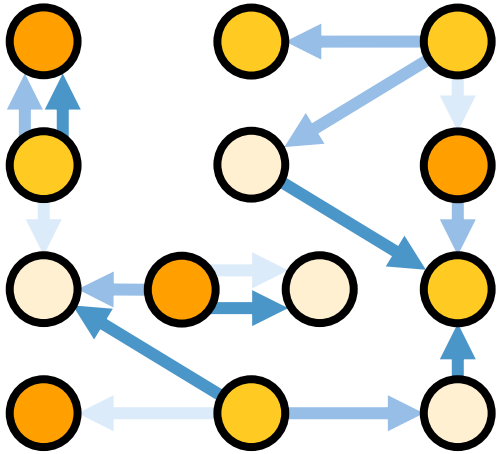
04 Dynamic split

- Randomly **re-split** the fact set and the training set
  - Fact set: used for aggregating neighboring embeddings
  - Training set: used for calculating the loss

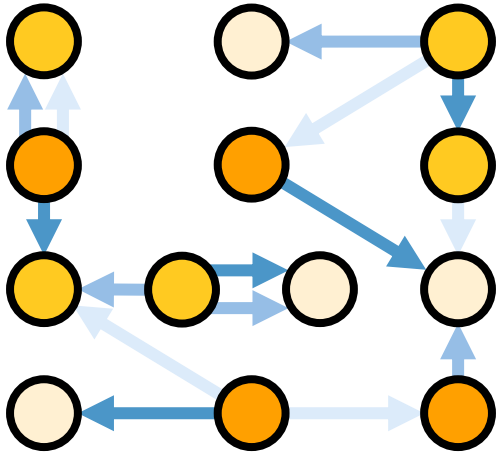


04 Re-initialization

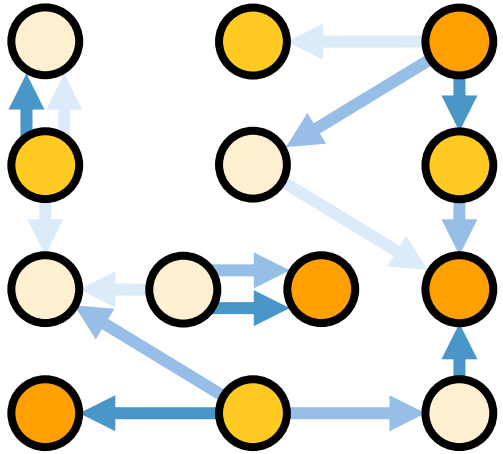
- Randomly **re-initialize** all feature vectors of entities and relations
  - Learns how to compute embedding vectors using random feature vectors
  - Related to the expressive power of GNNs



Epoch 1



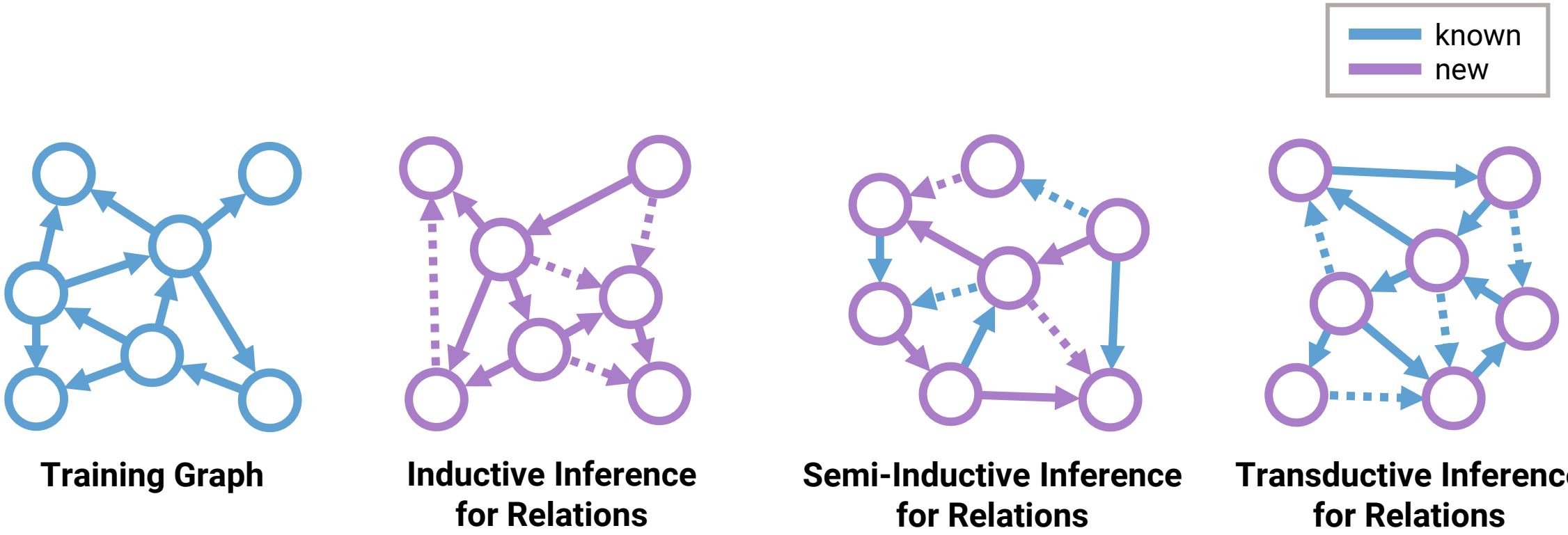
Epoch 2



Epoch 3

04 Experimental Results

- Datasets
  - Based on NELL, Wikidata, and Freebase
  - Create **13 real-world datasets** with various inductive settings

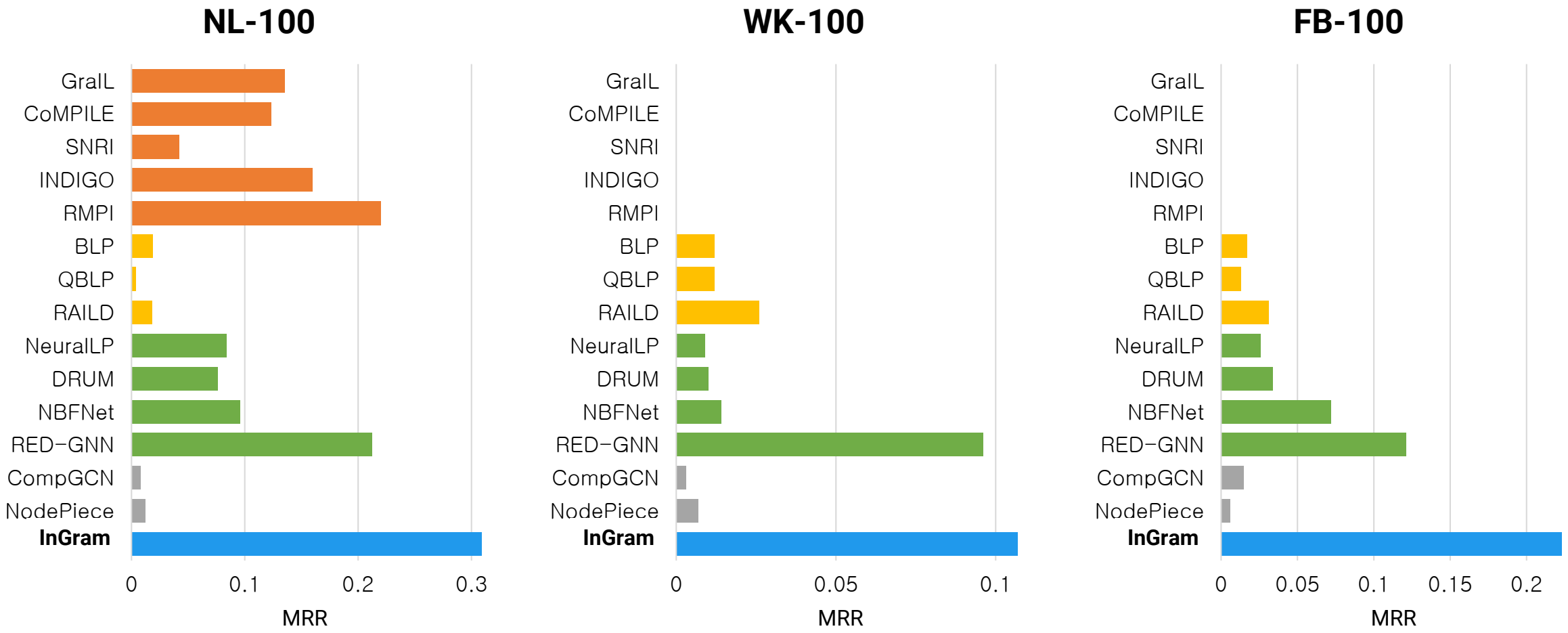


04 Experimental Results

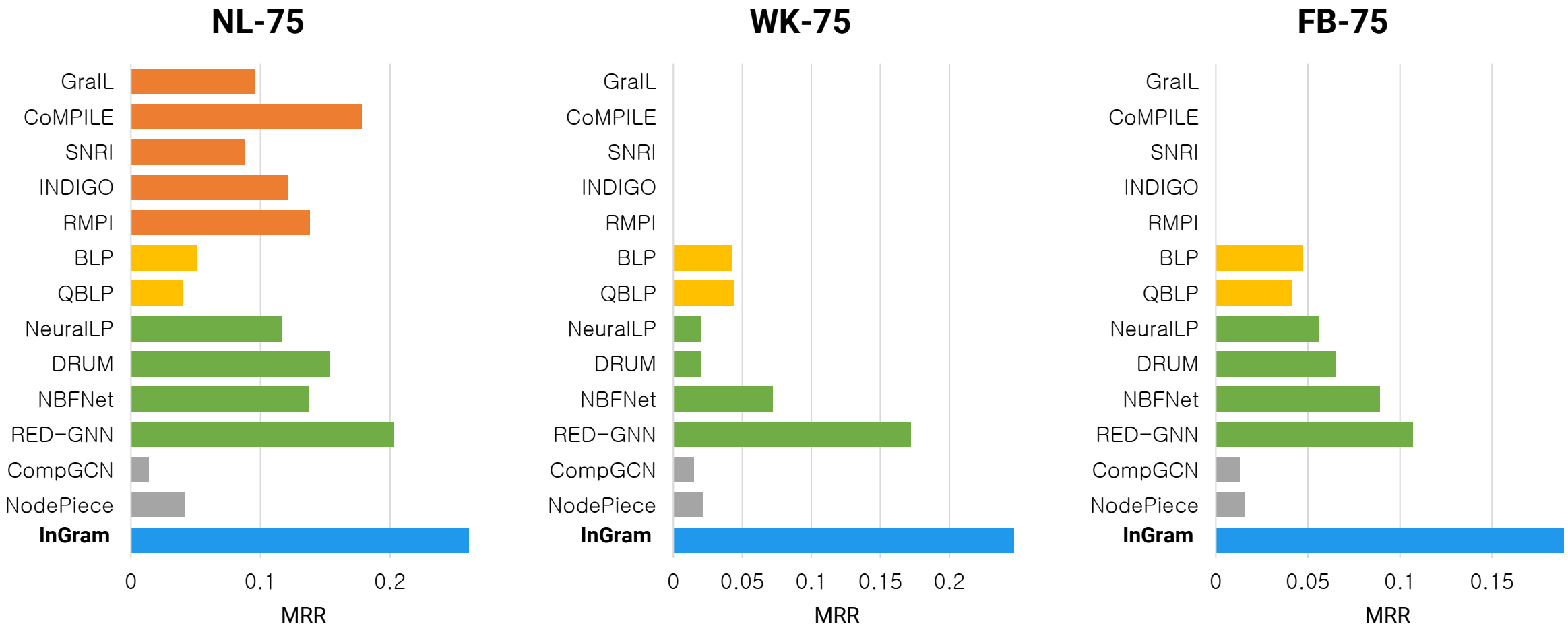
- Datasets
  - Based on NELL, Wikidata, and Freebase
  - Create **13 real-world datasets** with various inductive settings

- Comparison with **14 baselines**
  - Subgraph sampling: GraIL(ICML 2020), CoMPILE(AAAI 2021), SNRI(IJCAI 2022), INDIGO(NeurIPS 2021), RMPI(ICDE 2023)
  - BERT-based: BLP(WWW 2021), QBLP(ISWC 2021), RAILD(IJCKG 2022)
  - Rule-based: NeuralLP(NIPS 2017), DRUM(NeurIPS 2019), NBFNet(NeurIPS 2021), RED-GNN(WWW 2022)
  - Others: CompGCN(ICLR 2020), NodePiece(ICLR 2022)

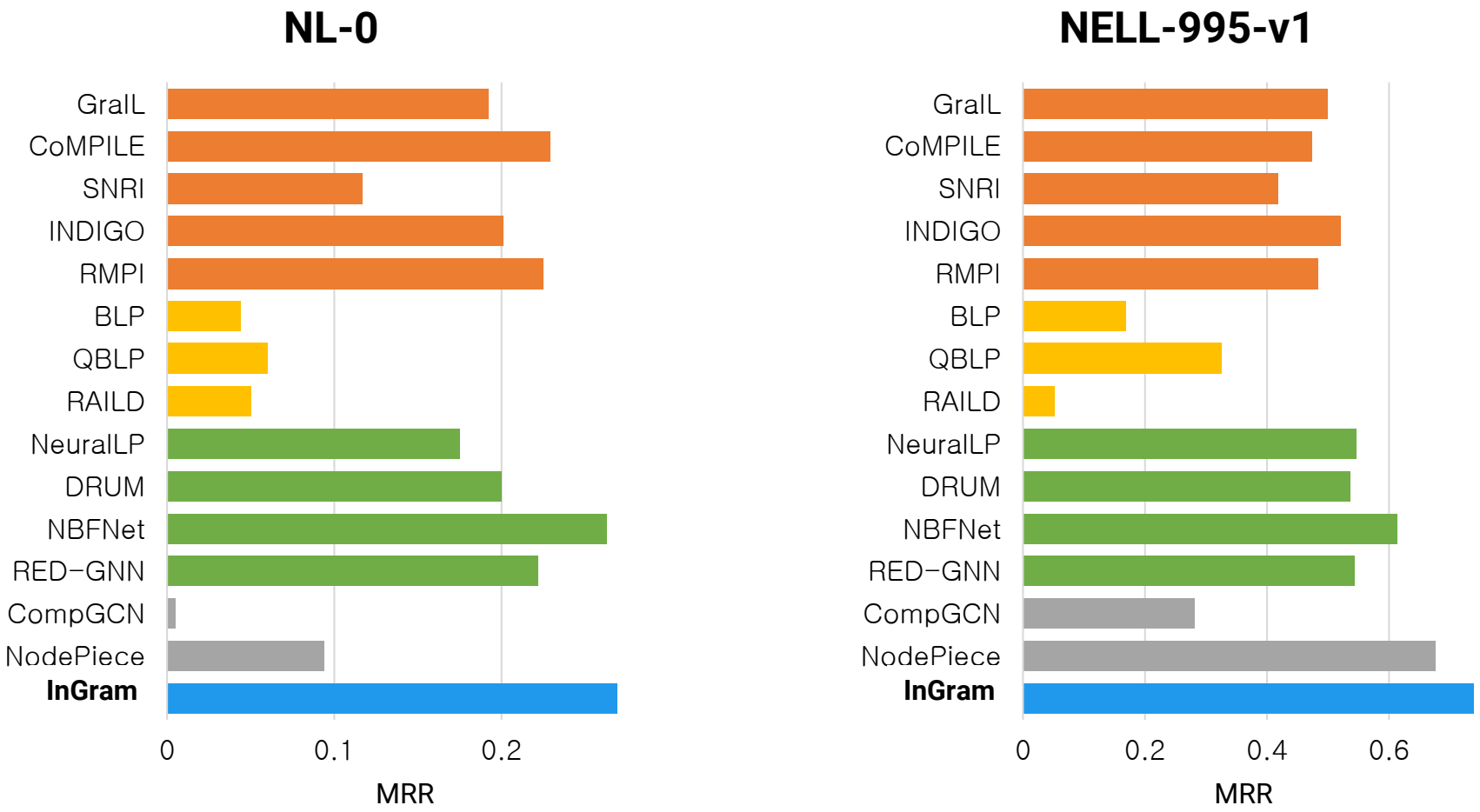
04 Inductive Inference for Relations



04 Semi-Inductive Inference for Relations



04 Transductive Inference for Relations



04 Conclusion

- Explore various **inductive settings**
- Define the **relation graph** to handle new relations at inference time
- Propose **InGram**, which learns to generate embeddings solely based on the structure of a given knowledge graph
- InGram significantly outperforms state-of-the-art methods for **inductive**, **semi-inductive**, and **transductive** inferences for relations

05 References

- Some slides are made based on the following references.
  - F. Yang et al., “Differentiable Learning of Logical Rules for Knowledge Base Reasoning”, NIPS, 2017.
  - A. Sadeghian et al., “DRUM:End-To-End Differentiable Rule Mining On Knowledge Graphs”, NeurIPS, 2019.
  - K. K. Teru et al., “Inductive Relation Prediction by Subgraph Reasoning”, ICML, 2020.
  - Z. Zhu et al., “Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction”, NeurIPS, 2021.
  - Y. Zhang and Q. Yao, “Knowledge Graph Reasoning with Relational Digraph”, TheWebConf, 2022.
  - Y. Geng et al., “Relational Message Passing for Fully Inductive Knowledge Graph Completion”, ICDE, 2023.
  - J. Lee et al., “InGram: Inductive Knowledge Graph Embedding via Relation Graphs”, ICML, 2023.
  - Y. Zhang et al., “AdaProp: Learning Adaptive Propagation for Graph Neural Network based Knowledge Graph Reasoning”, KDD, 2023.
    J. Liu et al., “KnowFormer: Revisiting Transformers for Knowledge Graph Reasoning”, ICML, 2024.