Hyper-relational Knowledge Graphs and Beyond:
From Representation Learning to Retrieval-Augmented Generation

Sangjun Ji (지상준)

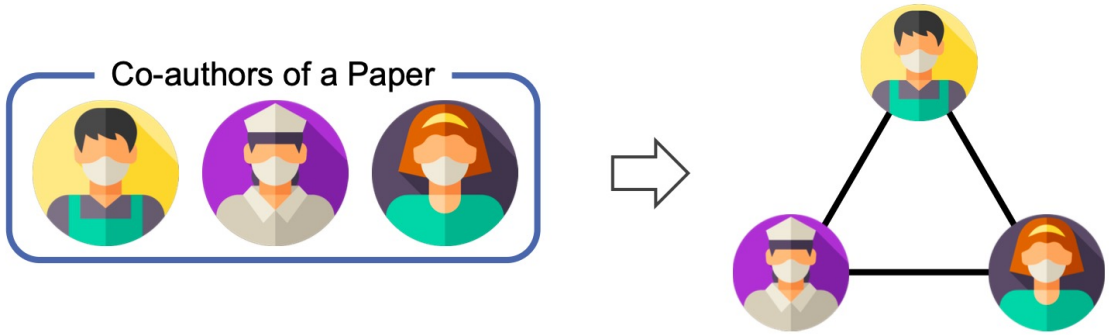
Department of Computer Science and Engineering
Konkuk University

2026. 07. 29.

CONTENTS

Graphs

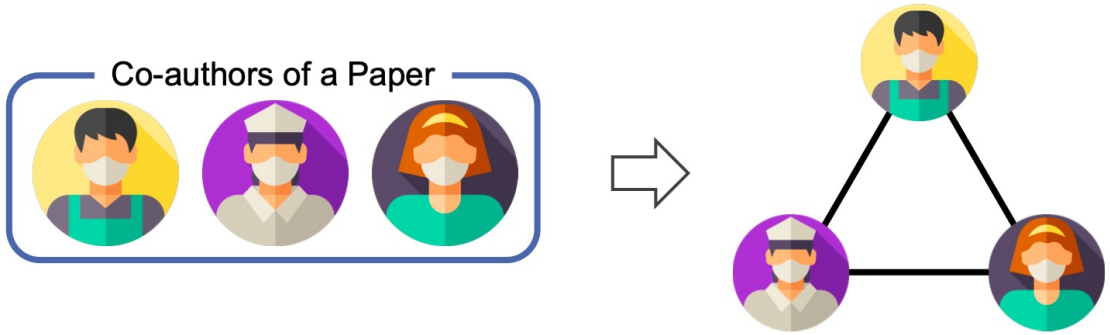
- **Graphs** can <u>model pairwise relations</u> by **edges**
  - ✓ Example: Co-authorship



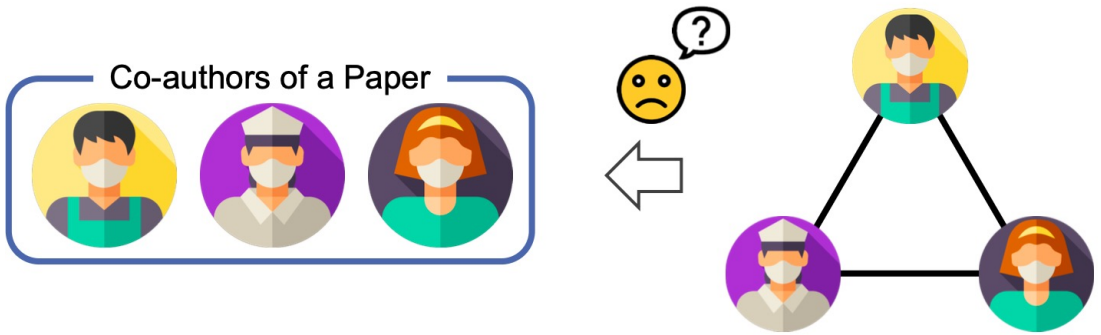
Mining of Real-world Hypergraphs: Patterns, Tools, and Generators (KDD'23)

Limitations of Graphs

- Graphs can only model pairwise relations by edges
  - ✓ Example: Co-authorship



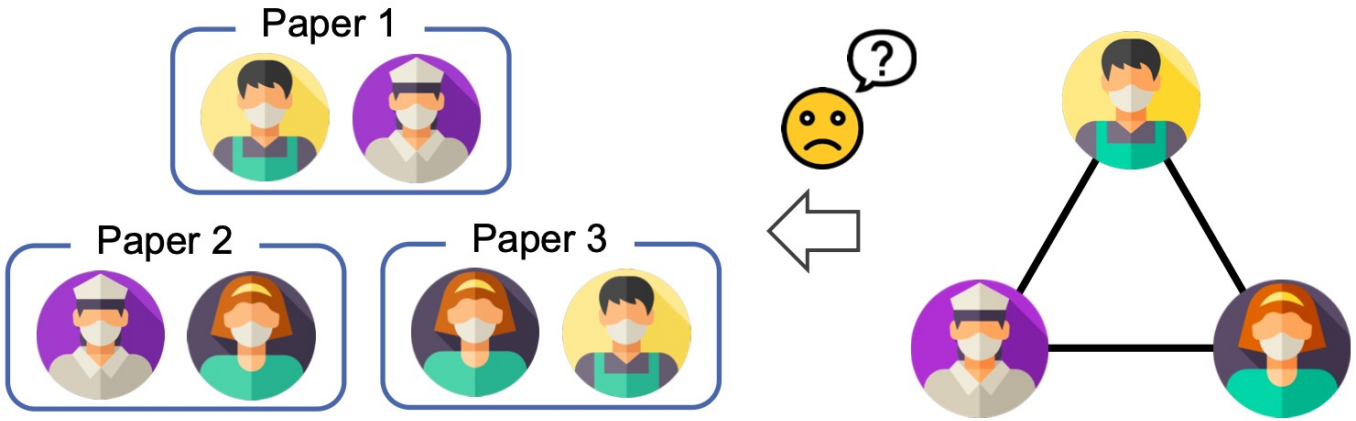
- Simple reduction to pairwise relations causes information loss
  - ✓ Example: Did the three authors co-work as a group?



Mining of Real-world Hypergraphs: Patterns, Tools, and Generators (KDD'23)

Limitations of Graphs (cont.)

- Example: The three authors may have never co-worked in the past.

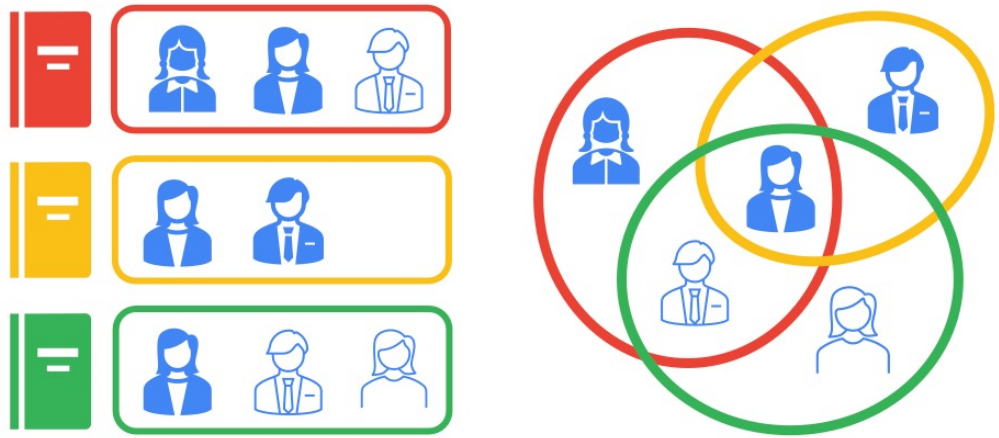


Mining of Real-world Hypergraphs: Patterns, Tools, and Generators (KDD'23)

What is Hypergraph?

- 기존의 simple graph와 달리,
  - ✓ hypergraph의 **hyperedge**는 두 개 이상의 node를 연결할 수 있음
  - ✓ Higher-order Interactions

- Ex) Co-authorship Relationship

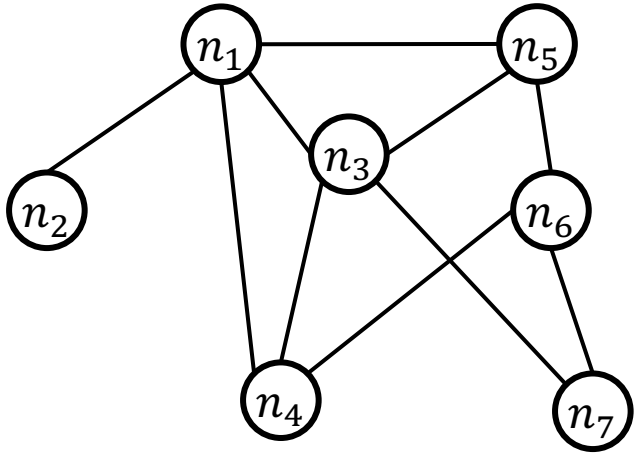


(a) Co-authors of publications (b) Hypergraph

A Survey on Hypergraph Neural Networks: An In-Depth and Step-By-Step Guide (KDD'24)

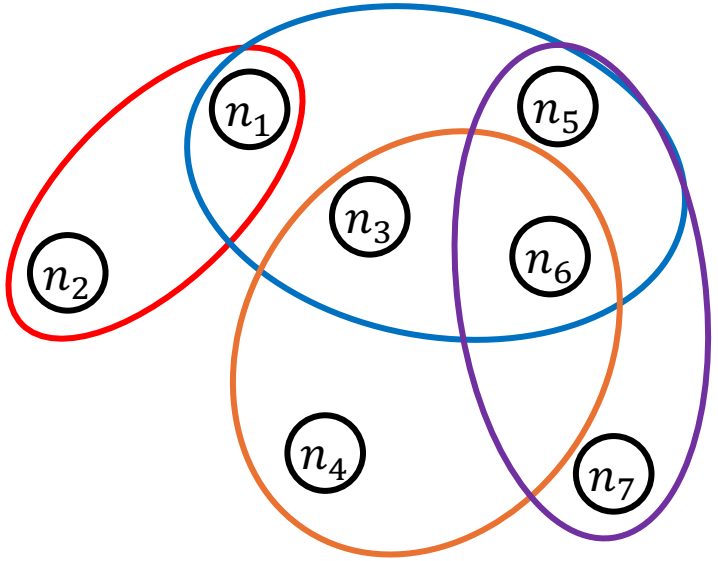
What is Hypergraph? (cont.)

- Graph
  - ✓ Node & Edge
  - ✓ Adjacency matrix A



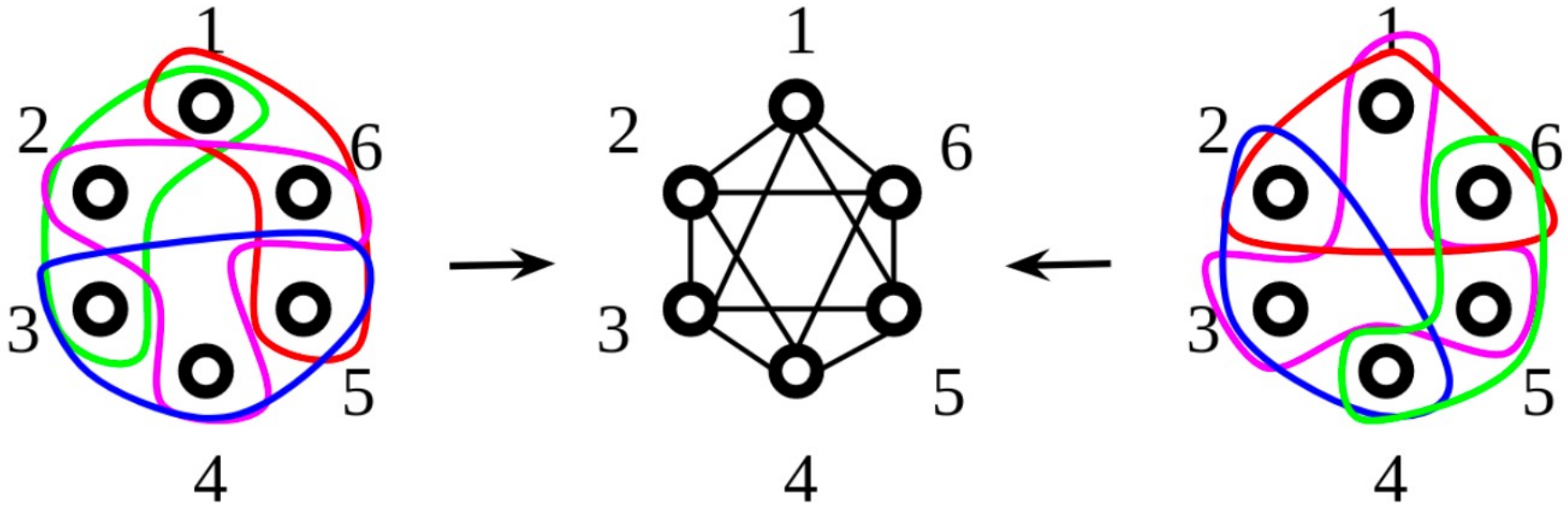
$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Hypergraph
  - ✓ Node & Hyperedge
  - ✓ Incidence matrix H



$$H = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Information Loss under Clique Expansion



- Clique Expansion을 통해 hypergraph 를 pairwise graph 로 변환 시
  - ✓ $E_1 = \{\{1,2,3\},\{1,5,6\},\{3,4,5\},\{2,4,6\}\}$
  - ✓ $E_2 = \{\{1,2,6\},\{1,3,5\},\{2,3,4\},\{4,5,6\}\}$
  - ✓ 두 hypergraph의 weighted clique expansion은 동일한 결과를 갖게 됨

Information Loss under Clique Expansion (cont.)

- 일반적인 graph 모델의 경우, edge가 두 node간의 pairwise 관계만을 나타낼 수 있음
  - ✓ 따라서 3개 이상의 node가 참여하는 higher-order interaction은 graph 구조에서 직접 표현될 수 없으며,
  - ✓ 이를 강제로 표현하려고 하면 필연적으로 정보 손실이 발생함

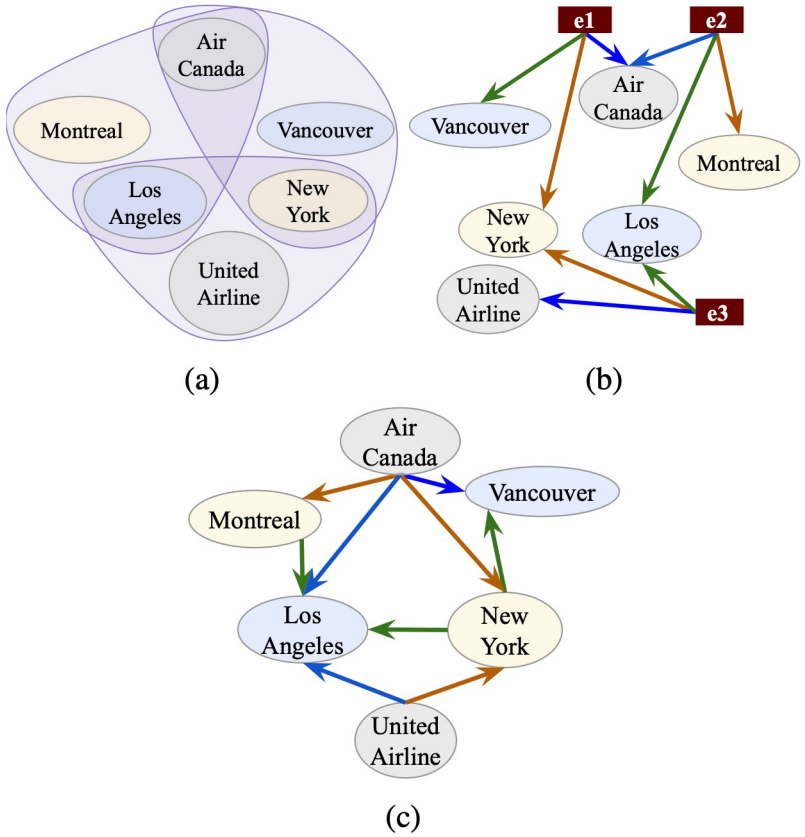
⇔ 하나의 higher-order interaction이 여러 개의 pairwise edge로 흩어지면서,

⇔ 원래 상호작용의 **그룹 구조**(어떤 노드들이 함께 연결되었는지에 대한 정보)가 소실됨

⇔ 즉, higher-order interactions을 일반적인 graph로 표현하면 topology 정보 손실이 발생할 수 있음!

Limitations of Knowledge Graphs

- 이는 Knowledge Graph에서도 마찬가지…
- Many facts in the real world are inherently n-ary (non-binary)

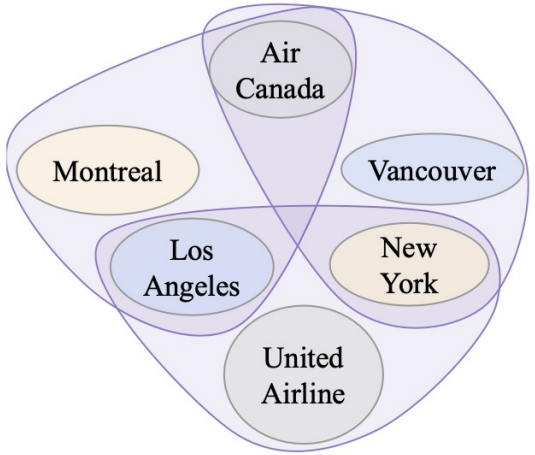


- (a)에서는 "**flies_between**"이라는 relation에 해당하는 세 개의 fact를 보여줌
- (b)는 **reification** 기법을 적용하여 binary relation을 가지도록 변환
- (c)는 **Star-to-clique** 방법을 원래의 hypergraph에 적용한 결과

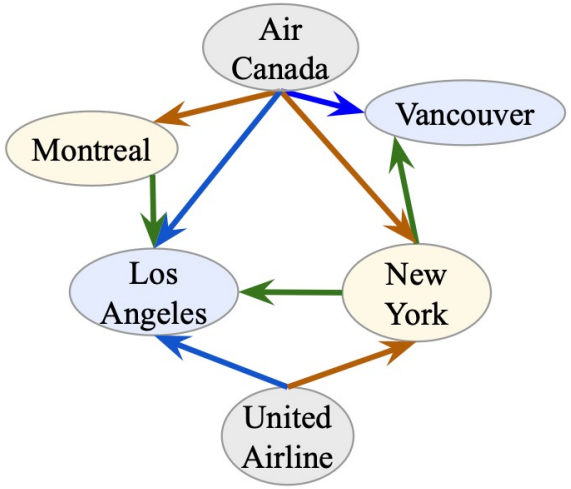
Knowledge Hypergraphs: Prediction Beyond Binary Relations (IJCAI'20)

Limitations of Knowledge Graphs (cont.)

- (c) Star-to-clique 방법을 사용하면,
  - ✓ 변환과정에서 정보 손실이 발생함
  - ✓ "flies_between(Air Canada, New York, Los Angeles)" 를 변환하면, 관련 entity들이 edge로 연결되어 있어 해당 사실이 참인 것처럼 해석될 수 있음
  - ✓ 그러나, 원래의 hypergraph를 살펴보면, Air Canada가 New York에서 Los Angeles로 비행하지 않는다는 사실이 명확!



(a)



(c)

Knowledge Hypergraphs: Prediction Beyond Binary Relations (IJCAI'20)

N-ary Fact Representations

- N-ary Relational Representations (NRRs)
- Knowledge Hypergraphs (KHGs)
- **Hyper-relational Knowledge Graphs (HKGs)**

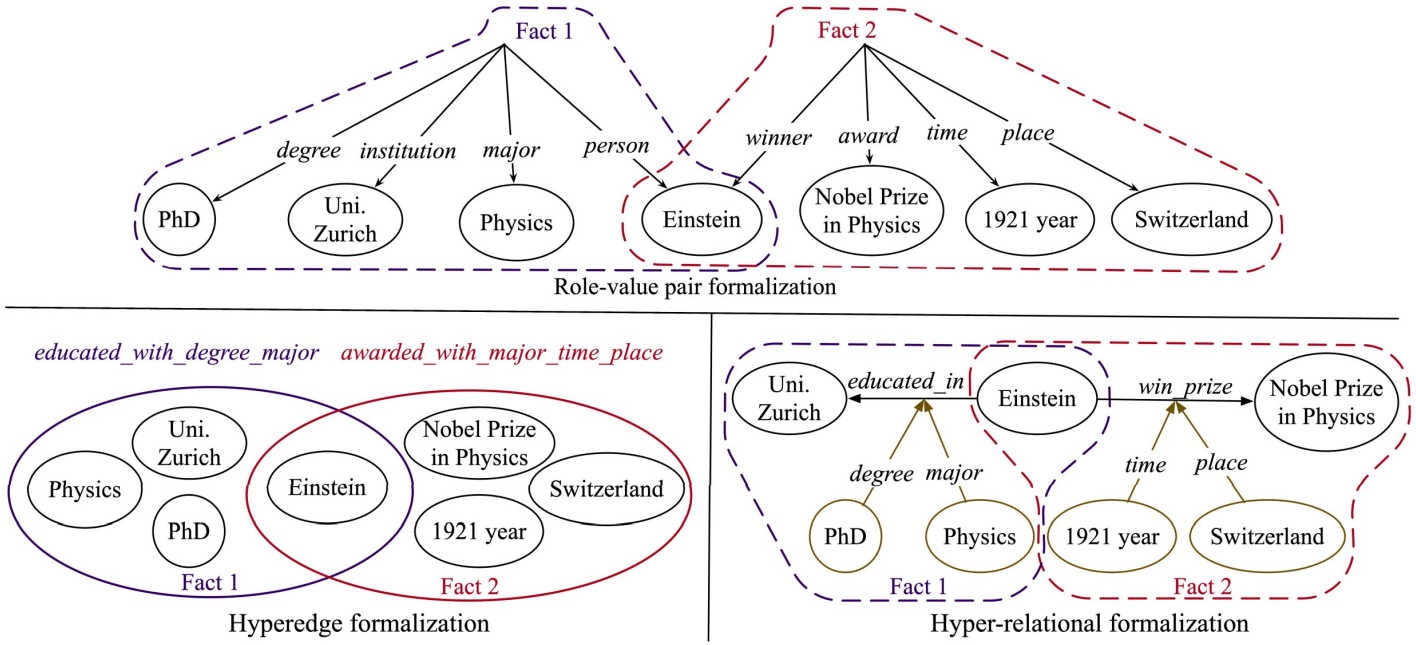
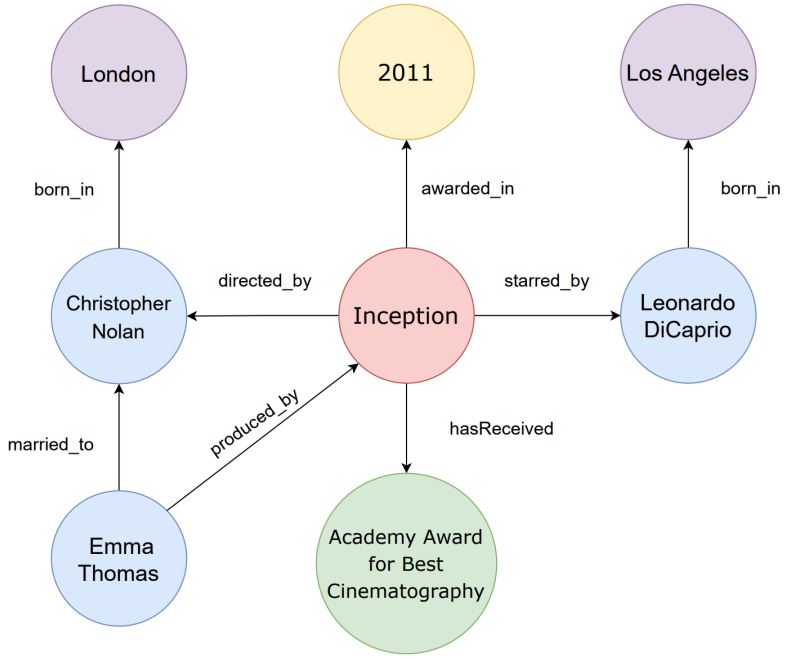


Figure 2: Examples of different formalizations of n-ary facts.

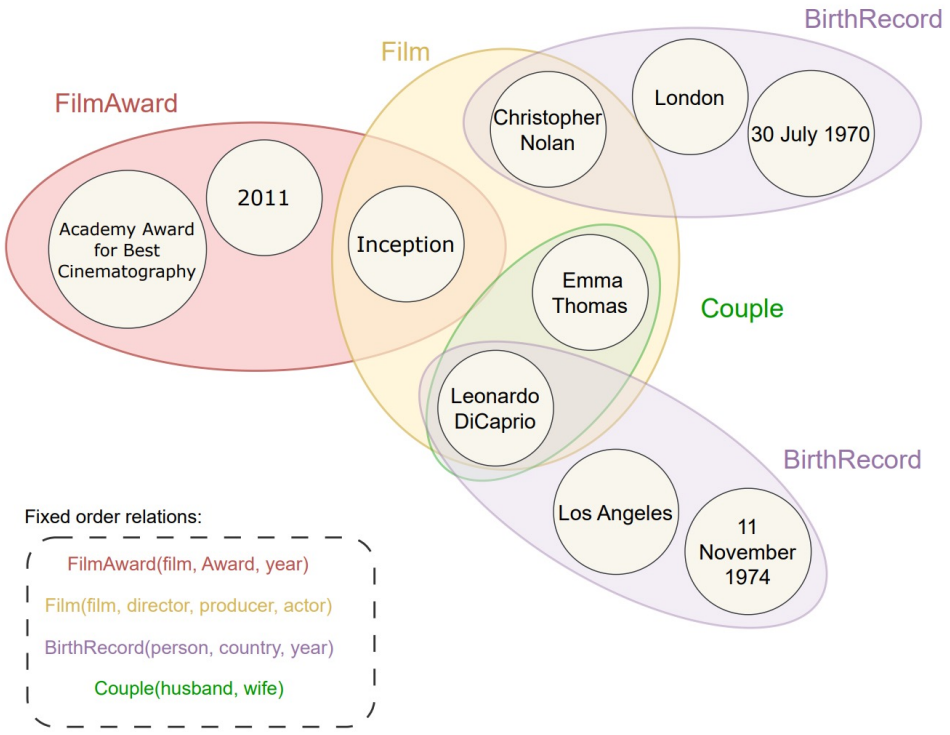
A Survey of Link Prediction in N-ary Knowledge Graphs (EMNLP'25)

Knowledge Hypergraphs

- Fixed order relation
  - ✓ relation(entity1, entity2, entity3, …)
  - ✓ relation 마다 position에 따른 entity role을 정의해줘야 함



(a) Knowledge Graph

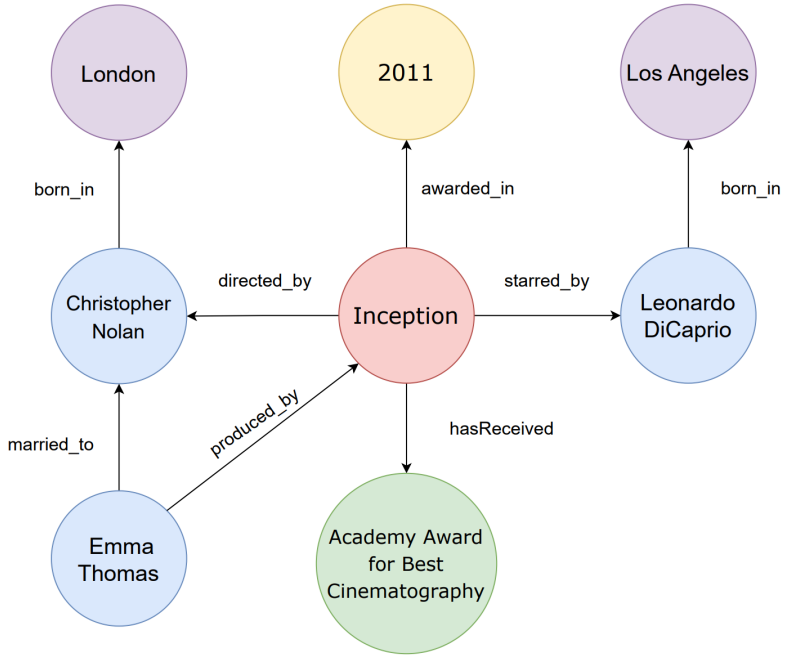


(b) Knowledge HyperGraph

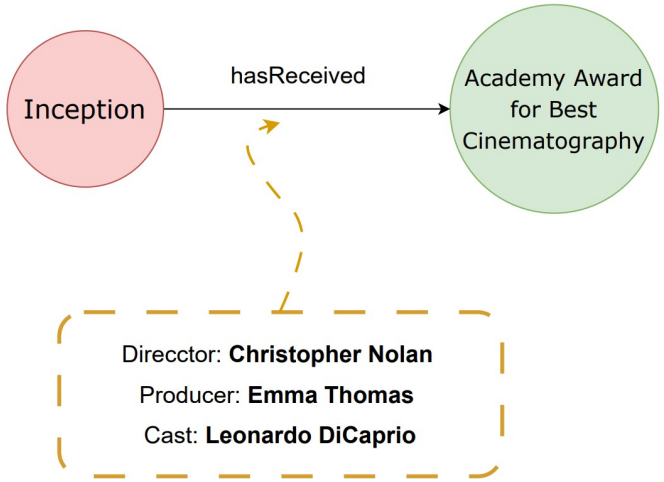
Two-dimensional Taxonomy for N-ary Knowledge Representation Learning Methods (arXiv'25)

Hyper-relational Knowledge Graphs

- Main triple - (subject, relation, object)
  - ✓ Qualifiers - (attribute, value)



(a) Knowledge Graph



(a) Hyper-relational Knowledge Graph

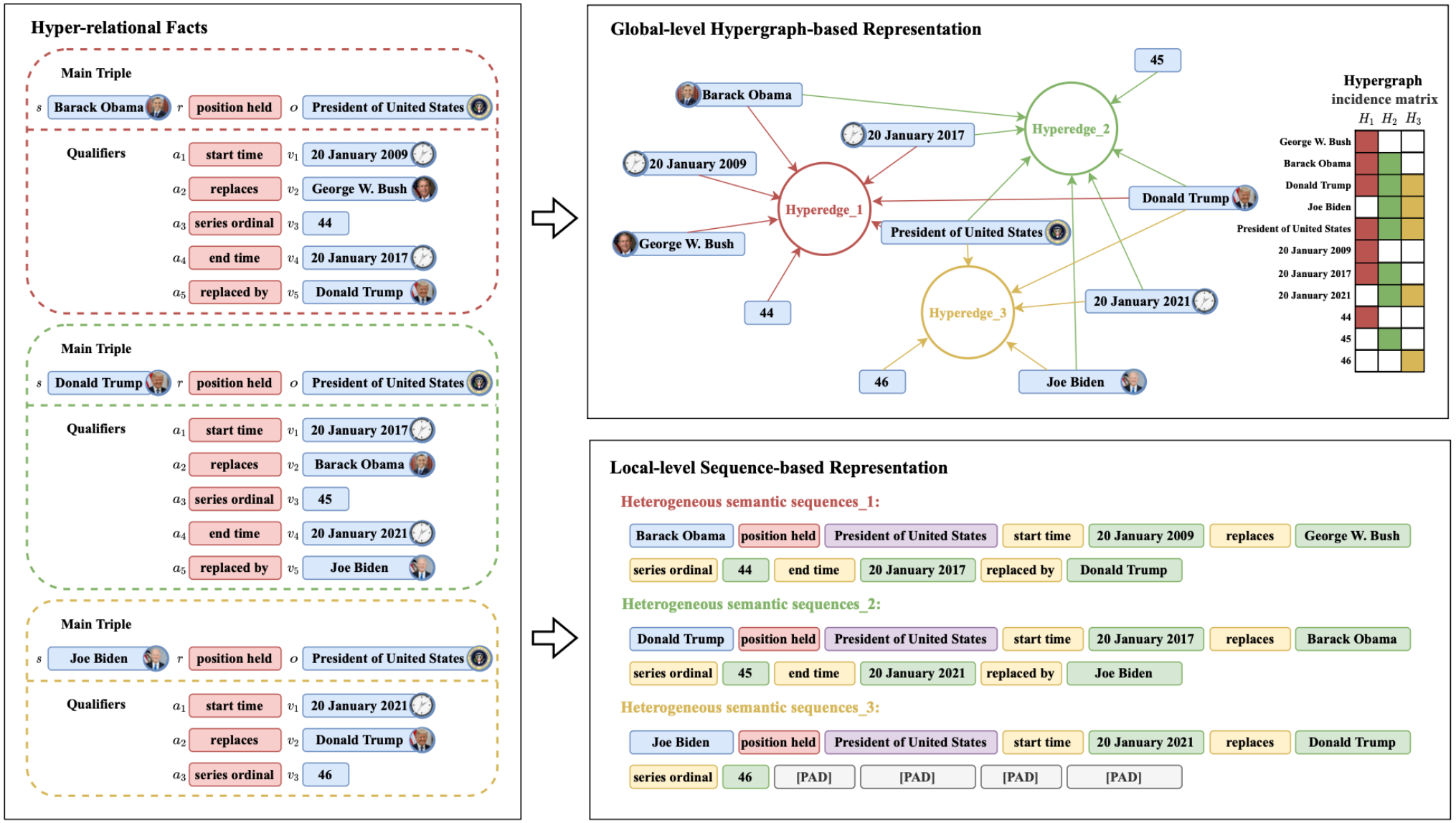
Two-dimensional Taxonomy for N-ary Knowledge Representation Learning Methods (arXiv'25)

Hyper-relational Knowledge Graphs (cont.)

- HKGs admit two complementary **views**:
  - ✓ **Globally**, a **hypergraph** captures cross-fact connections
  - ✓ **Locally**, **sequences** model intra-fact semantics

HAHE: Hierarchical Attention for Hyper-Relational Knowledge Graphs in Global and Local Level (ACL'23)

Hyper-relational Knowledge Graphs (cont.)



HAHE: Hierarchical Attention for Hyper-Relational Knowledge Graphs in Global and Local Level (ACL'23)

CONTENTS

Joint Global-Local Representations via Relation-Entity Pair Encoding for Hyper-Relational Knowledge Graphs

Sangjun Ji[1], Sangjune Kim[2], Youngho Lee[1], Bonyou Koo[1], Xiongnan Jin[3] and Byungkook Oh[1]*

[1]Konkuk University, Seoul, Republic of Korea
[2]KAIST, Daejeon, Republic of Korea
[3]Shenzhen University, Shenzhen, China

CONTENTS

Local-only HKG Representation Learning

- Early approaches predominantly model **intra-fact semantics with local encoders**

- **HINGE** encodes (main triple, qualifier) tuples with a CNN

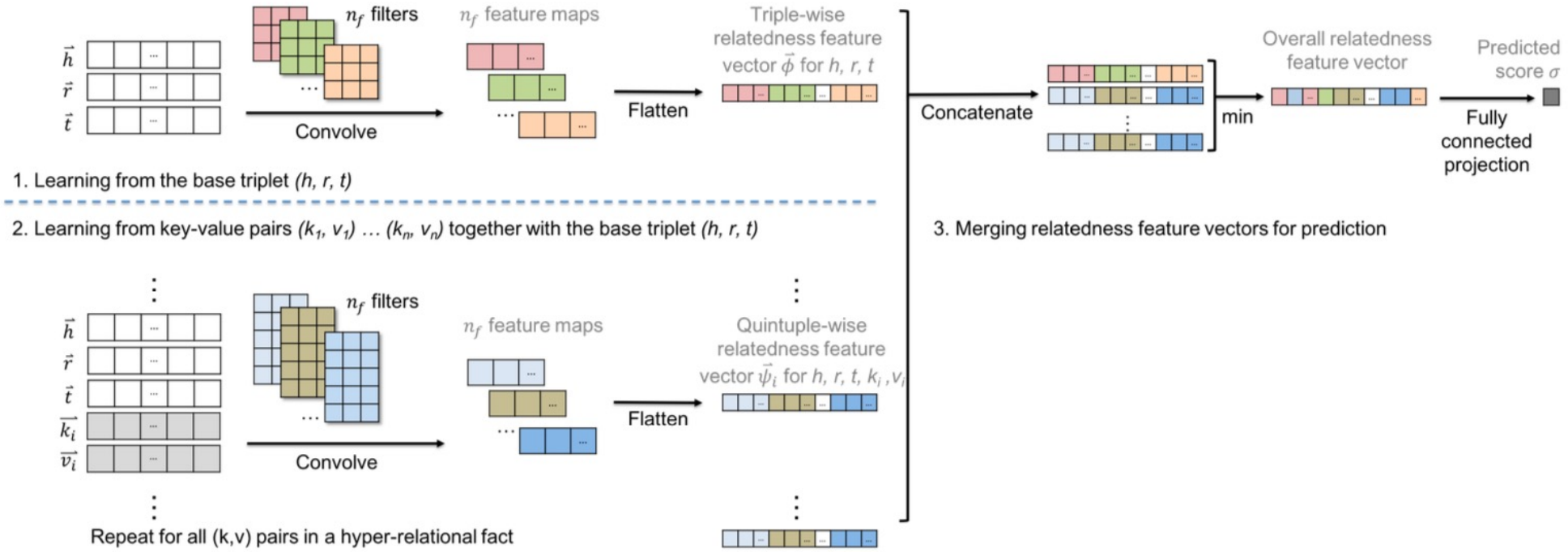


Figure 2: Overview of our proposed method HINGE

Beyond Triplets: Hyper-Relational Knowledge Graph Embedding for Link Prediction (WWW'20)

Local-only HKG Representation Learning (cont.)

- GRAN applies a transformer encoder over per-fact structures

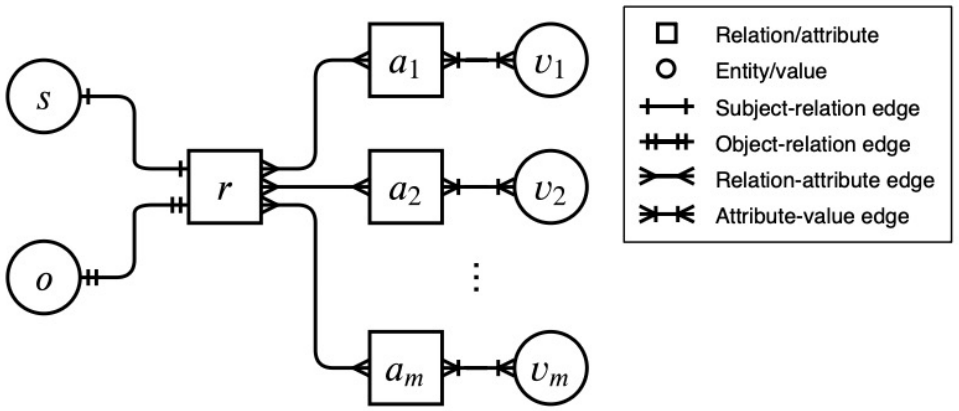


Figure 1: An n-ary fact as a heterogenous graph, with relations/attributes and entities/values as vertices, and four types of edges designed between the vertices.

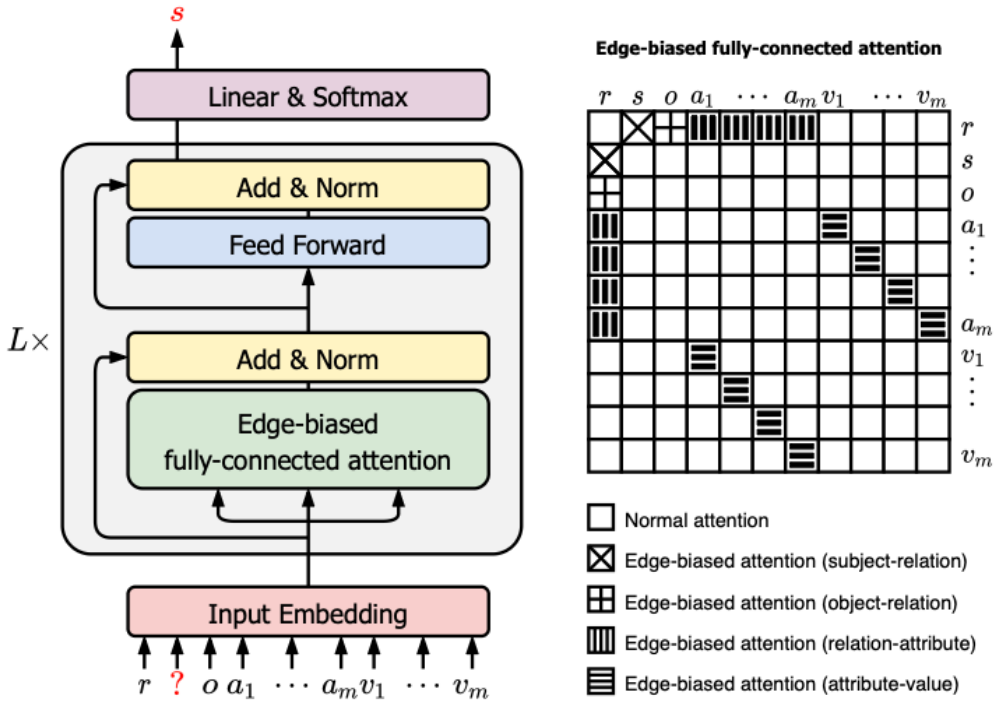


Figure 2: Overview of the graph learning process, with edge-biased fully-connected attention illustrated.

Link Prediction on N-ary Relational Facts: A Graph-based Approach (ACL-F'21)

Local-only* HKG Representation Learning (cont.)

- **HyperFormer** integrates one-hop neighbors via a sequence-based Entity Neighbor Aggregator
  - ✓ But still lacks multi-hop propagation

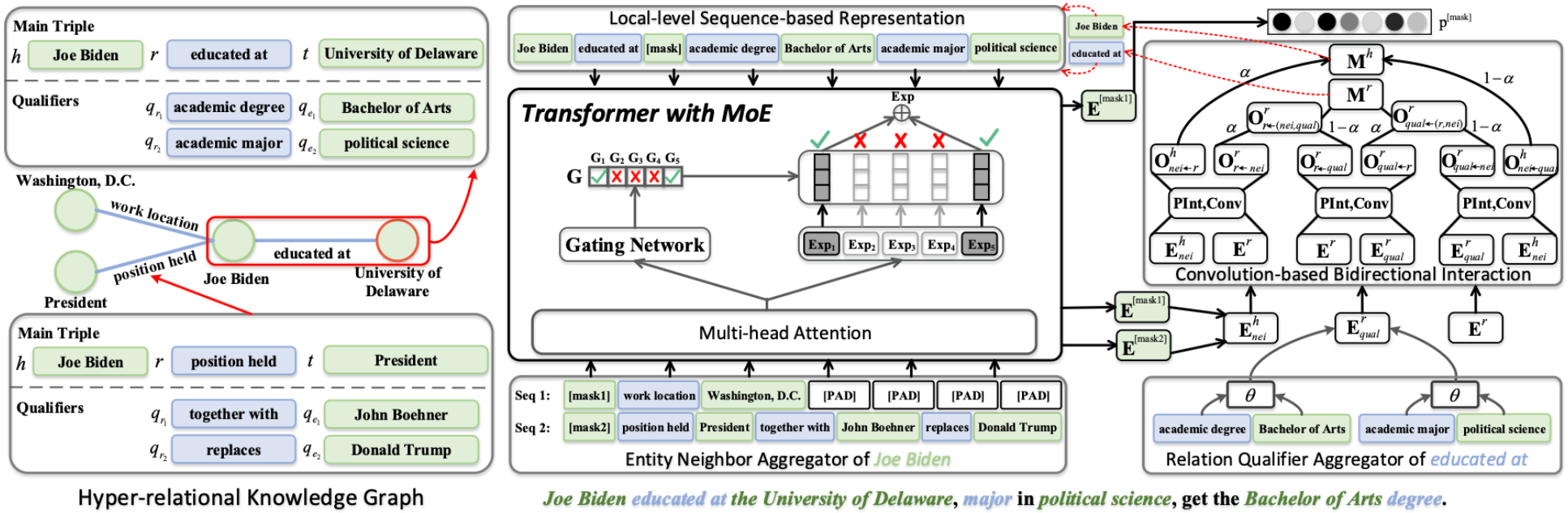


Figure 2: An overview of our HyperFormer model, containing three modules: Entity Neighbor Aggregator (§ 3.2.1), Relation Qualifier Aggregator (§ 3.2.2), and Convolution-based Bidirectional Interaction (§ 3.2.3).

HyperFormer: Enhancing Entity and Relation Interaction for Hyper-Relational Knowledge Graph Completion (CIKM'23)

Local-only HKG Representation Learning (cont.)

- Despite their strengths, these approaches tend to **treat each Hyper-relational fact (H-fact) in isolation**,
  - ✓ overlooking cross-fact structural connections.

- They may **miss higher-order dependencies** that only emerge through cross-fact connectivity.

- More recently, global encoders based on hypergraph message passing have incorporated qualifier-aware and/or multi-view structures.

Recent HKG Representation Learning

- StarE integrates qualifiers with main relations to form statement representations and propagates messages across the graph

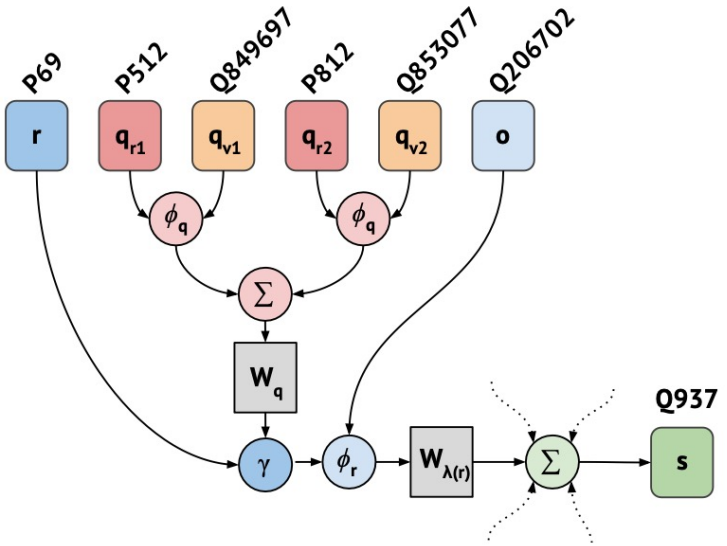
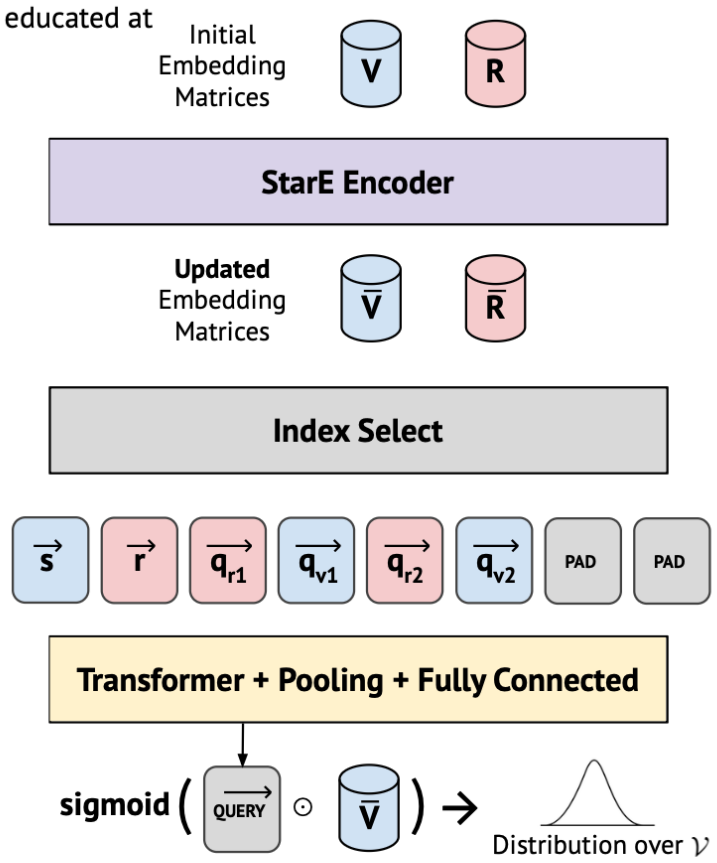


Figure 2: The mechanism in which StarE encodes a hyper-relational fact from Fig. 1.B. Qualifier pairs are passed through a composition function ϕ_q, summed and transformed by $\mathbf{W}_q$. The resulting vector is then merged via γ, and ϕ_r with the relation and object vector, respectively. Finally, node *Q937* aggregates messages from this and other hyper-relational edges.



Message Passing for Hyper-Relational Knowledge Graphs (EMNLP'20)

Recent HKG Representation Learning (cont.)

- HAHE constructs an entity-centric global hypergraph and applies hierarchical dual-attention
  - ✓ First updating entity embeddings globally,
  - ✓ Then transferring them to initialize local sequence encodings — a sequential cascade

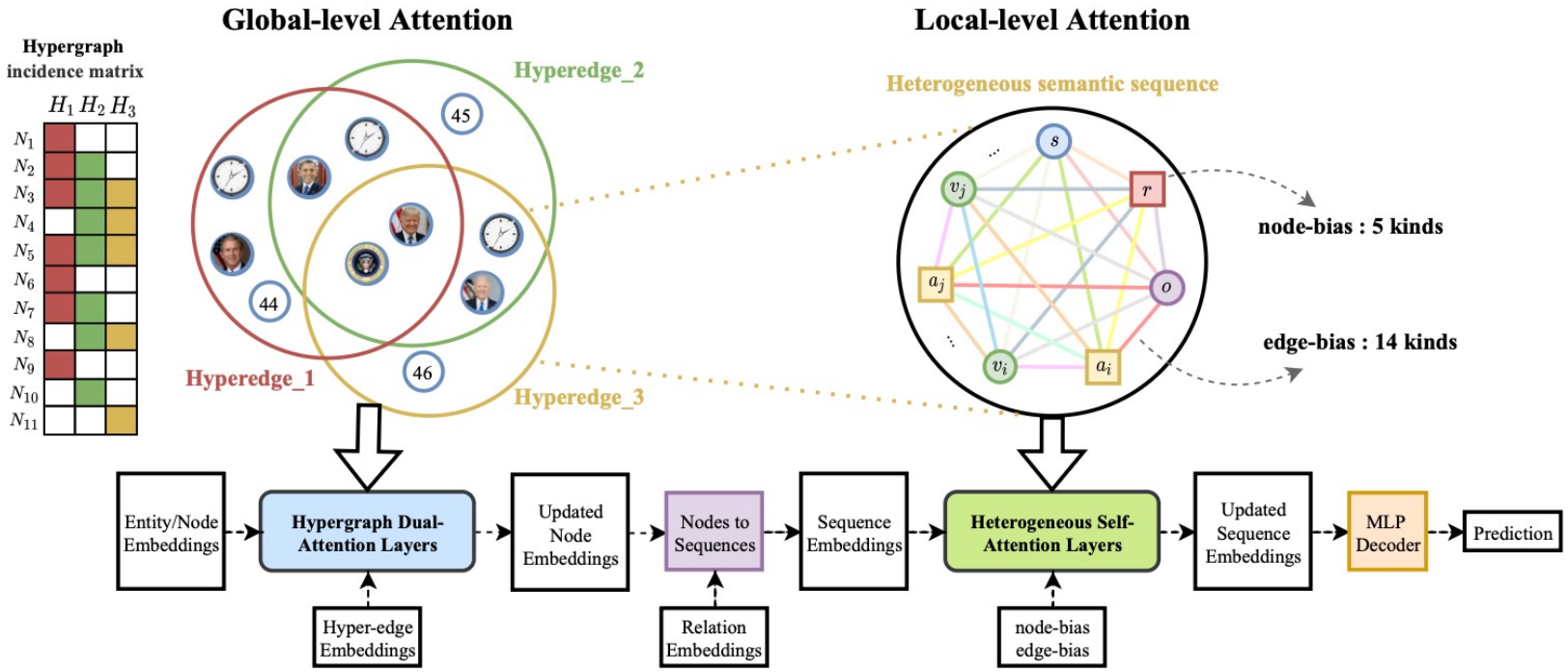
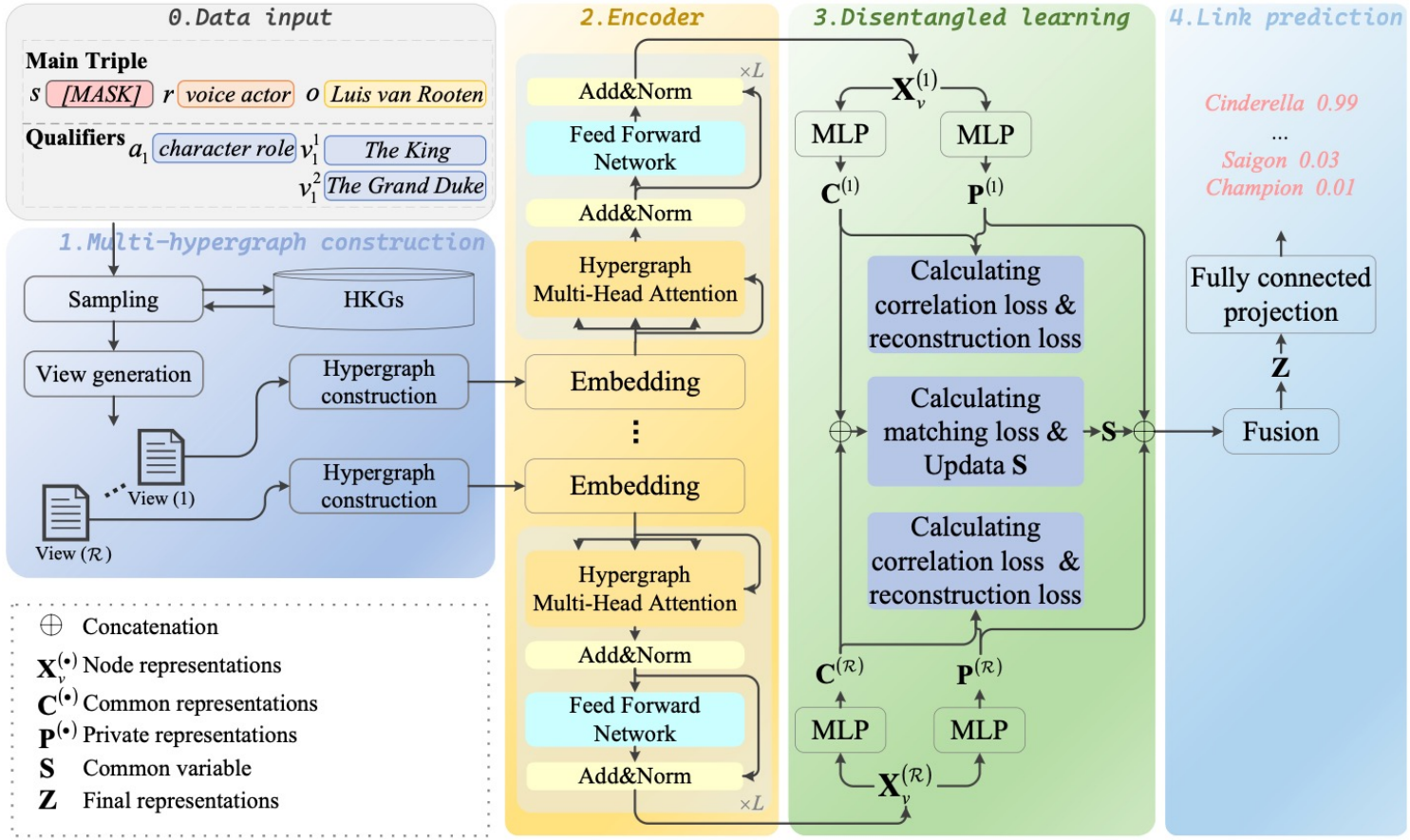


Figure 3: The overview of HAHE model for Global-level and Local-level Representation of HKGs.

HAHE: Hierarchical Attention for Hyper-Relational Knowledge Graphs in Global and Local Level (ACL'23)

Recent HKG Representation Learning (cont.)

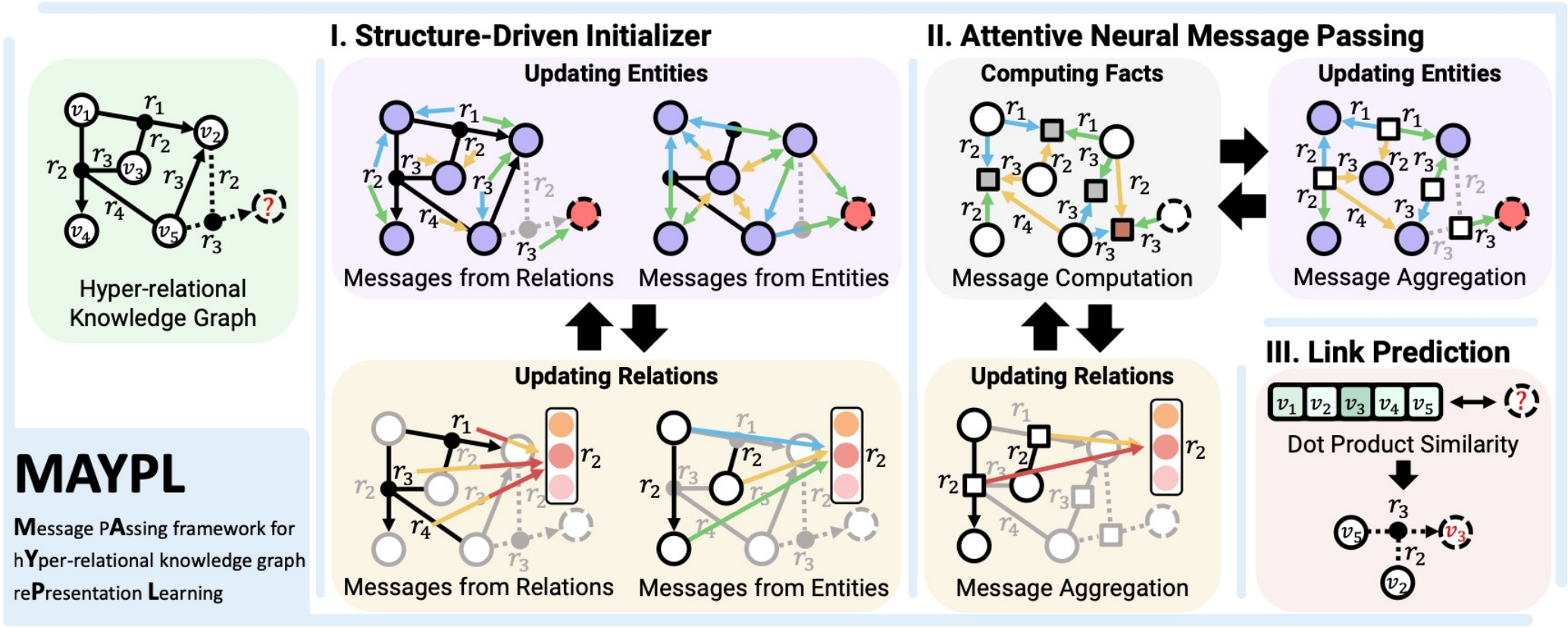
- MHD constructs multiple view-wise hypergraph perspectives
  - ✓ learns representations via **common-private disentanglement** to separate shared semantics from view-specific structure



Hyper-Relational Knowledge Representation Learning with Multi-Hypergraph Disentanglement (WWW'25)

Recent HKG Representation Learning (cont.)

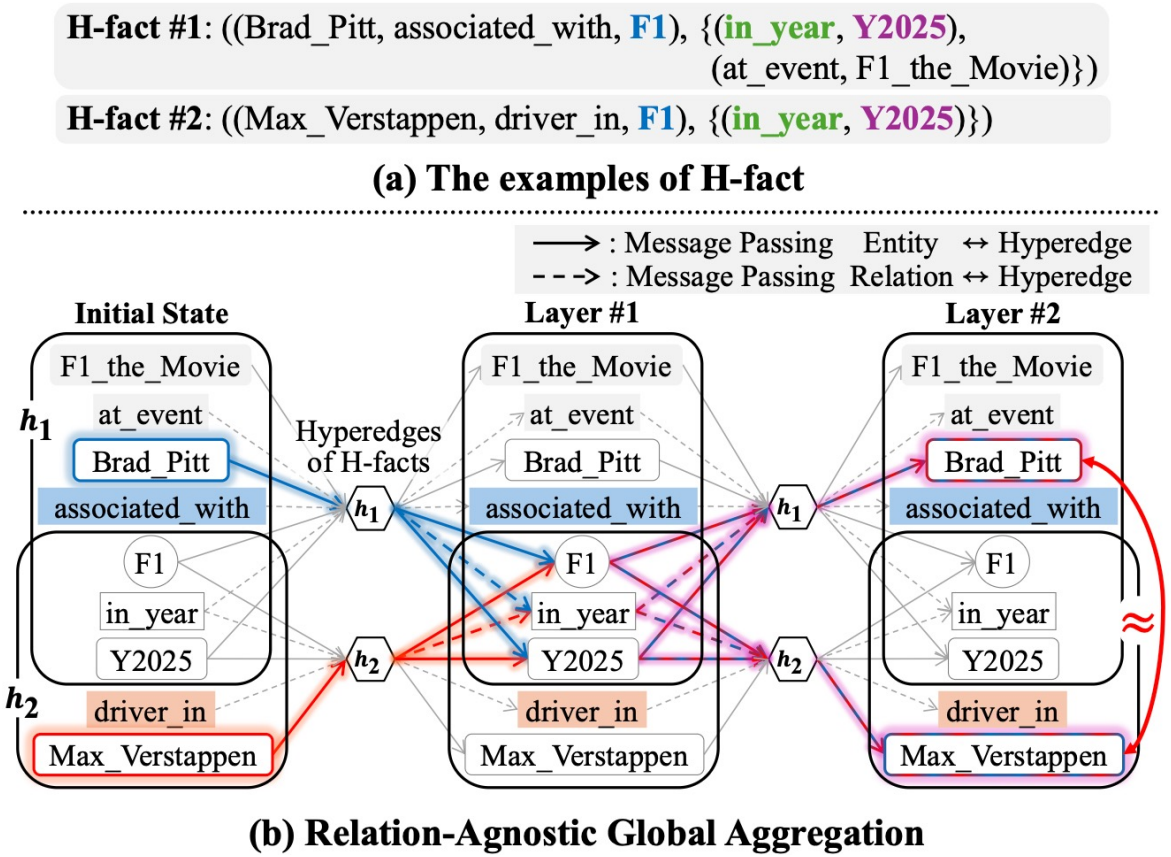
- MAYPL learns structural representations of HKGs through message passing over qualifier-induced structures



Structure Is All You Need: Structural Representation Learning on Hyper-Relational Knowledge Graphs (ICML'25)

Challenges - [C1] Relation-Agnostic Global Aggregation

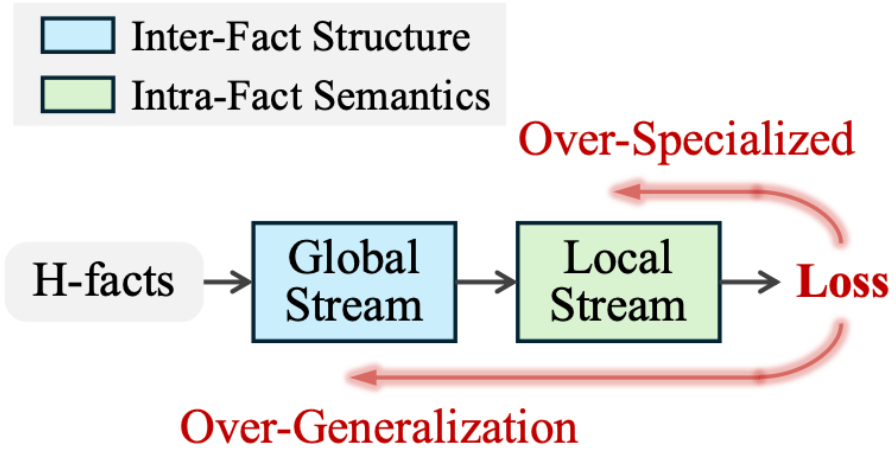
- A key issue arises from node-centric, relation/role-agnostic pooling:
  - ✓ Global encoders maintain a single representation per entity and pull messages from incident H-facts into the same node state,
    - Prematurely mixing heterogeneous evidence and blurring which relation and which role generated it.



Challenges - [C2] Insufficient Global-Local Fusion

- Existing methods typically
  - ✓ address only one of the two views or
  - ✓ combine them through a one-way global→local fusion scheme

- Because the two components interact mainly through one-way feature passing,
  - ✓ intra-fact constraints **cannot effectively modulate** global propagation,
  - ✓ leaving **limited feedback from local semantics to the global module** during training



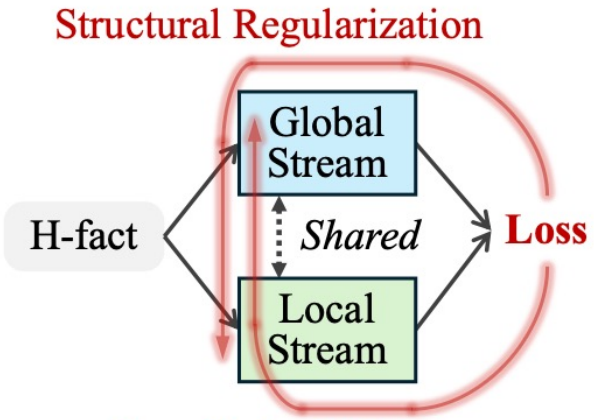
(c) Cascaded Fusion

GLoRE: Global-Local representations via Relation-Entity pair encoding

- **Relation-Entity pair-centric joint global-local model** for HKG representation learning

- To address [**C1:** Relation-Agnostic Global Aggregation],
  - ✓ GLoRE performs propagation and interaction at the **level of relation–entity pairs**. (rather than entities)

- To address [**C2:** Insufficient Global-Local Fusion],
  - ✓ GLoRE combines global hypergraph propagation with local intra-fact self-attention,
    - **both operating over relation–entity pairs** and **co-optimized end-to-end via a shared pair encoder**.



(d) Joint Fusion (GLoRE)

Comparison with previous studies

Method	Encoding Unit	Global–Local Fusion Strategy	Auxiliary Relation Graph	Global–Local Coupling
HINGE	Node	Local only	None	–
GRAN	Node	Local only	None	–
HyperFormer	Node	Local only (w/ 1-hop neighbors)	None	–
StarE	Node	Global only	None	–
HAHE	Node	Cascaded	None	X
MAYPL	Pair	Global only	None	–
GLoRE	**Pair**	**Joint**	**Yes**	**O**

CONTENTS

Preliminaries

- We define a Hyper-relational Knowledge Graph (HKG) as $G = (V, R, H)$, which consists of
  - ✓ a set of entities V,
  - ✓ a set of relations R,
  - ✓ and a set of hyper-relational facts (H-facts) H

- An individual H-fact combines a main triple with qualifiers which are auxiliary key-value pairs

main relation qualifier relation = attribute

$$h = \left((v_1, r_1, v_2), \{(r_{i+1}, v_{i+2})\}_{i=1}^{m}\right)$$

subject object value

H-fact #1: ((Brad_Pitt, associated_with, **F1**), {(**in_year**, **Y2025**), (at_event, F1_the_Movie)})

H-fact #2: ((Max_Verstappen, driver_in, **F1**), {(**in_year**, **Y2025**)})

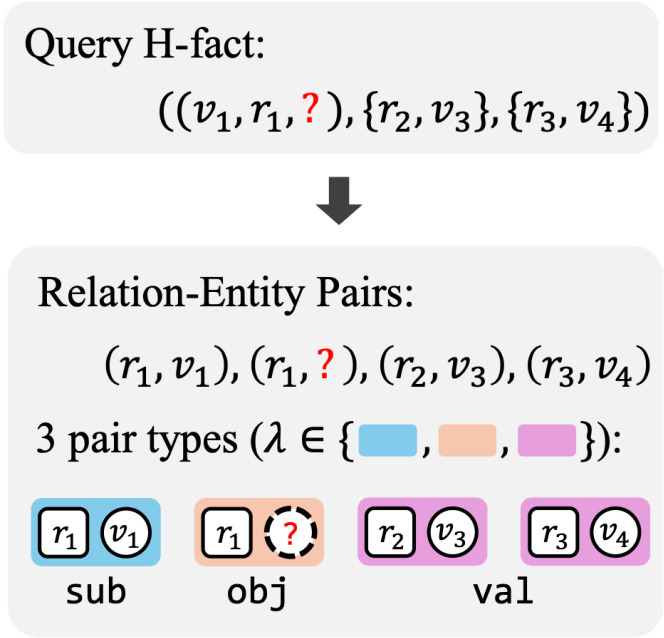
(a) The examples of H-fact

- **Task:** Link Prediction
  - ✓ Given an incomplete H-fact with one masked entity, rank all candidate entities to recover the missing entity

Relation-Entity Pair

- For a given hyper-relational fact, we construct a collection of relation-entity pairs:

- We classify each pair based on the role of its entity component
  - ✓ subject pair if the entity is the subject of the main triple
  - ✓ object pair if it is the object
  - ✓ qualifier pair for all remaining pairs

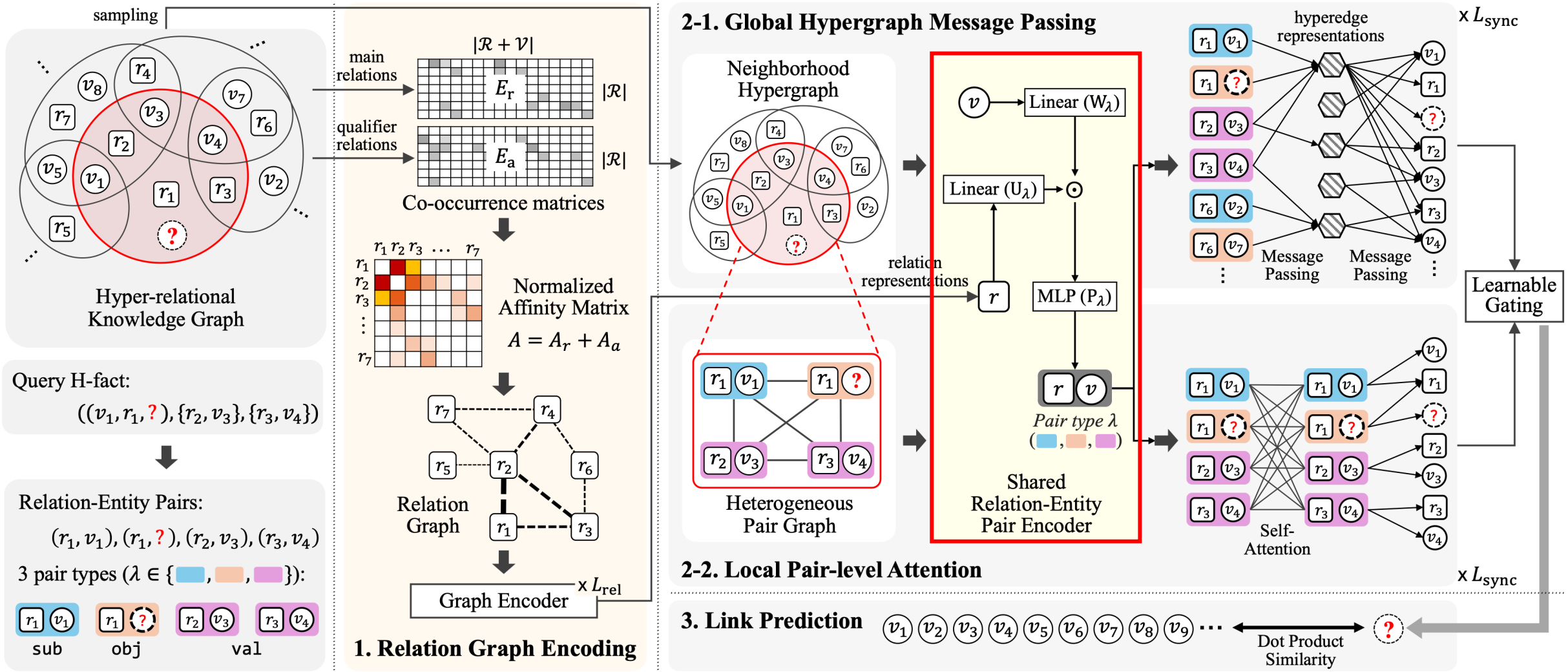


- **Relation-Entity Pair encoder**
  - ✓ Produces a role-conditioned representation

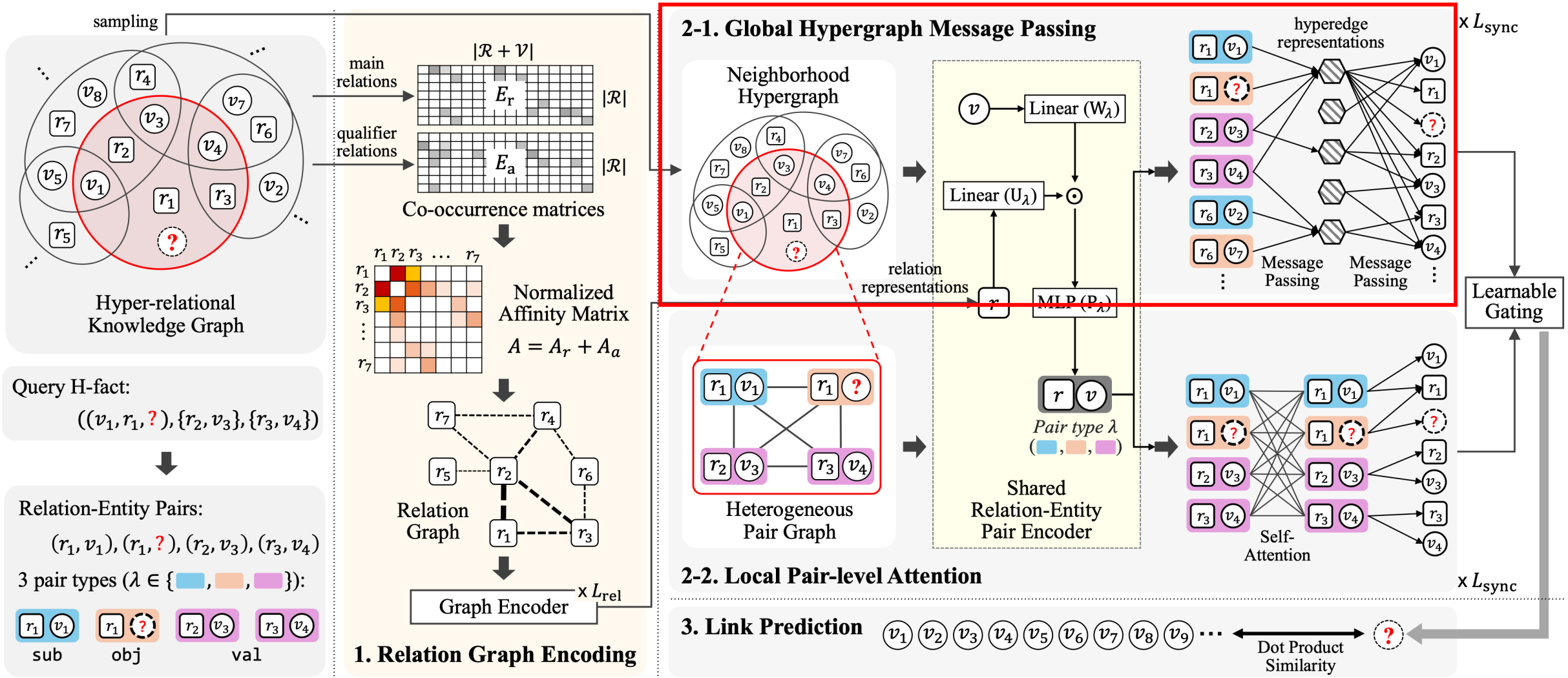
$$p_{(r,v)} = P_{\lambda_h(v)} \left(W_{\lambda_h(v)} \, \mathbf{v} \odot U_{\lambda_h(v)} \, \mathbf{r} \right) \qquad (1)$$

  - $\mathbb{v}, \mathbb{r}$ are entity and relation embeddings
  - ⊙ is the element-wise product
  - W, U are role-conditioned(=pair typed) linear maps

"Shared" Relation-Entity Pair Encoder



Global Hypergraph-based Message Passing



Global Hypergraph-based Message Passing (cont.)

- We treat each H-fact as a hyperedge in a hypergraph whose nodes are entities and relations

- Neighborhood Hypergraph
  - ✓ "적어도 하나의 entity를 공유하는 H-facts"를 neighbor로 정의

$$N(h) = \{ h' \in \mathcal{H} \mid h' \neq h,\ \mathcal{V}(h) \cap \mathcal{V}(h') \neq \emptyset \} \qquad (6)$$

- GLoRE encodes each hyperedge by aggregating relation–entity pair embeddings from the shared pair encoder
  - ✓ so cross-fact propagation preserves relation identity and entity-role information,
    - reducing role mixing compared to relation-agnostic node aggregation

Global Hypergraph-based Message Passing (cont.)

- Hyperedge Update
  - ✓ H-fact 하나를 구성하는 **relation-entity pair들을 바탕으로** hyperedge로 전달되는 message를 구성

$$m_h^{(l)} = \frac{1}{|\mathcal{P}_h|} \sum_{(r,v)\in\mathcal{P}_h} p_{(r,v)}^{(l)} \quad (7) \qquad \mathbf{h}^{(l)} = \text{LayerNorm}\left(\mathbf{h}^{(l-1)} + \text{act}\left(m_h^{(l)}\right)\right) \quad (8)$$

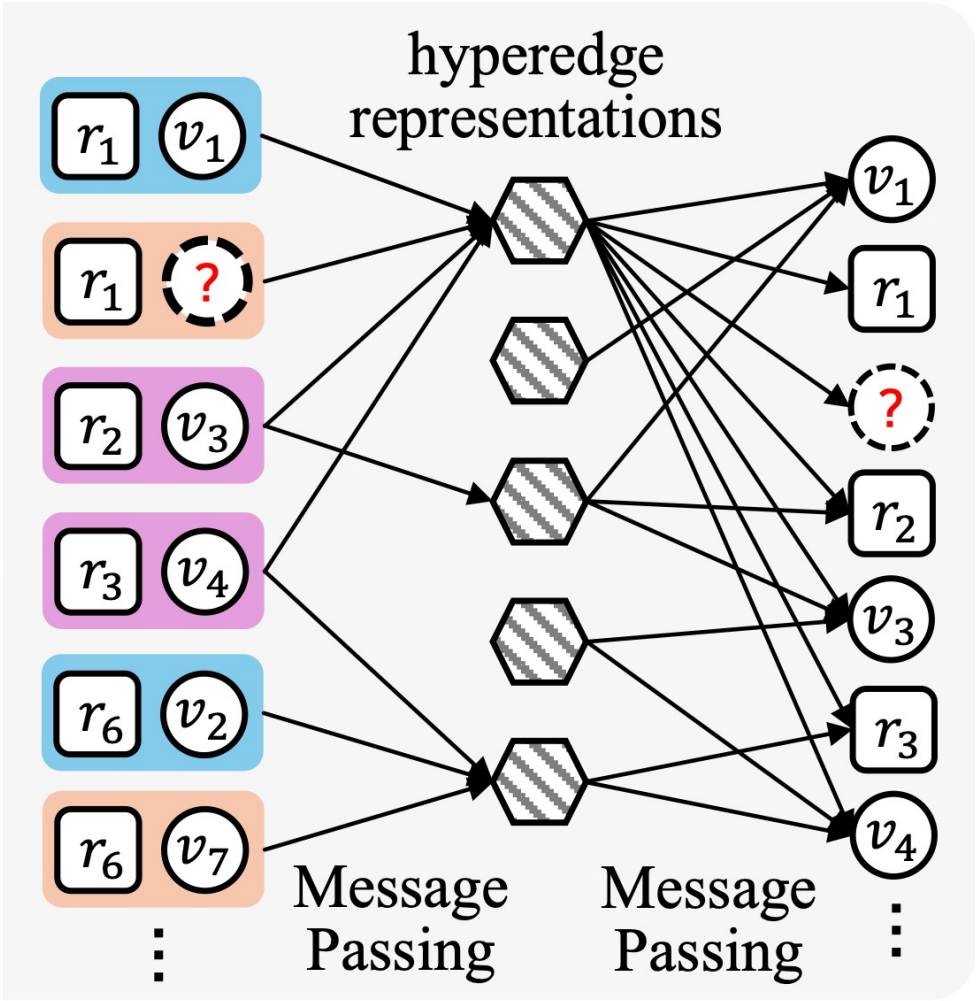
- Entity Update
  - ✓ 각 entity는 자신이 속한 hyperedge로부터 role-conditioned(=sub/obj/val) message를 수신해서 embedding을 갱신

$$q_{h\to v}^{(l)} = P'^{(l)}_{\lambda_h(v)}\left(W'^{(l)}_{\lambda_h(v)}\,\mathbf{h}^{(l)}\right) \quad (9) \qquad \mathbf{v}^{(l)} = \text{LayerNorm}\left(\mathbf{v}^{(l-1)} + \text{act}\left(\frac{1}{|H_v|}\sum_{h\in H_v} q_{h\to v}^{(l)}\right)\right) \quad (10)$$

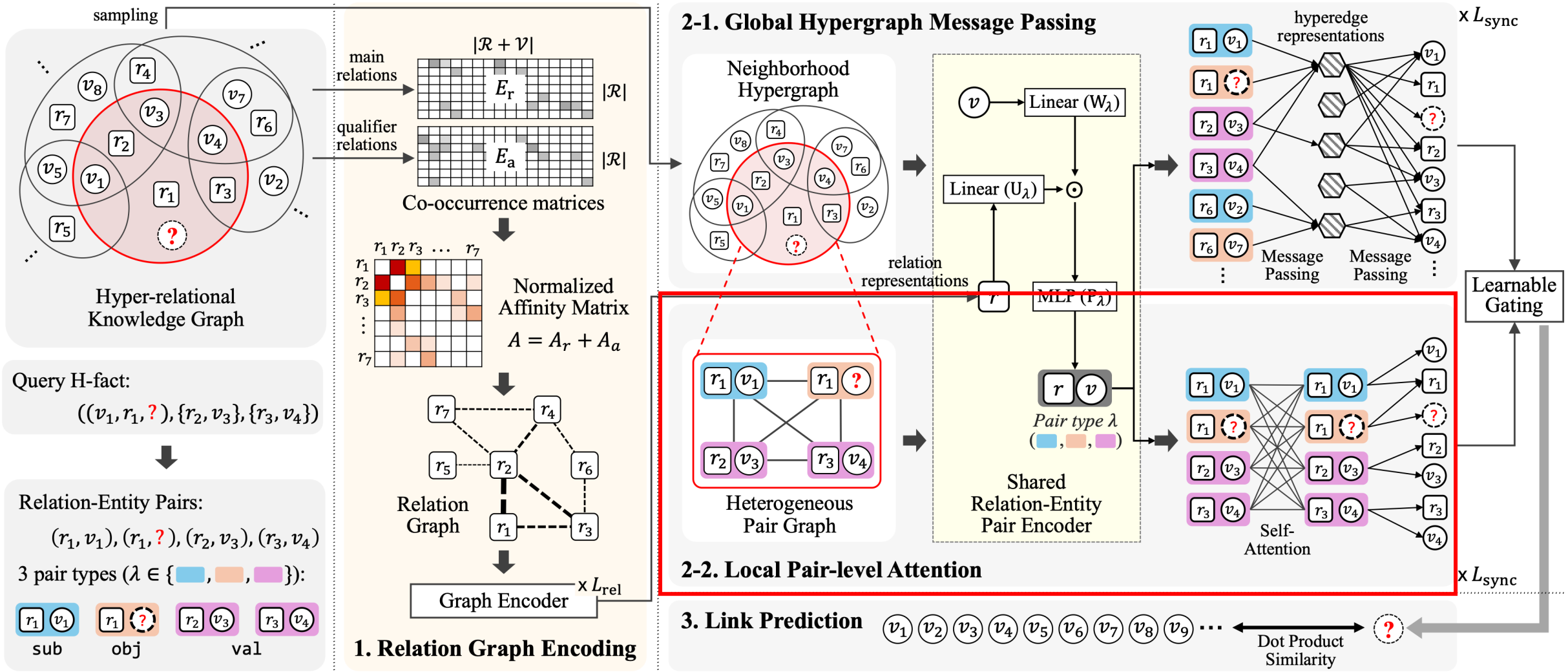
- Relation Update
  - ✓ 유사하게, 자신이 속한 hyperedge로부터 role-conditioned(=main/qual) message를 수신해서 embedding을 갱신

$$t_{h\to r}^{(l)} = P''^{(l)}_{\phi_h(r)}\left(W''^{(l)}_{\phi_h(r)}\,\mathbf{h}^{(l)}\right) \quad (11) \qquad \mathbf{r}^{(l)} = \text{LayerNorm}\left(\mathbf{r}^{(l-1)} + \text{act}\left(\frac{1}{|H_r|}\sum_{h\in H_r} t_{h\to r}^{(l)}\right)\right) \quad (12)$$

Global Hypergraph-based Message Passing (cont.)



Local Pair-level Self-Attention



Local Pair-level Self-Attention (cont.)

- Heterogeneous Pair Graph
  - ✓ We build a per-fact pair graph
    - whose nodes are relation-entity pairs each assigned a role {subj, obj, val}
  - ✓ Edges are typed by role combinations:

$$\ell_{ij} = \begin{cases} 0, & i = j \quad \text{(self)}, \\ 1, & \{\lambda_h(i), \lambda_h(j)\} = \{\text{sub}, \text{obj}\}, \\ 2, & \{\lambda_h(i), \lambda_h(j)\} = \{\text{sub}, \text{val}\}, \\ 3, & \{\lambda_h(i), \lambda_h(j)\} = \{\text{obj}, \text{val}\}, \\ 4, & \lambda_h(i) = \lambda_h(j) = \text{val}. \end{cases} \tag{13}$$

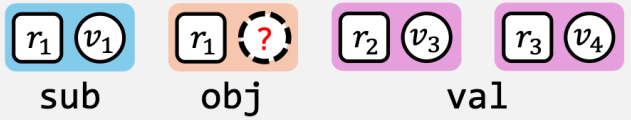
Query H-fact:

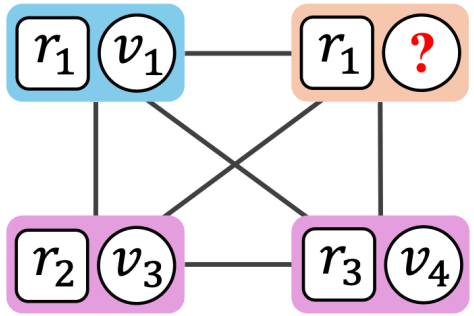
$$((v_1, r_1, ?), \{r_2, v_3\}, \{r_3, v_4\})$$

Relation-Entity Pairs:

$$(r_1, v_1), (r_1, ?), (r_2, v_3), (r_3, v_4)$$

3 pair types ($\lambda \in \{$, , $\}$):

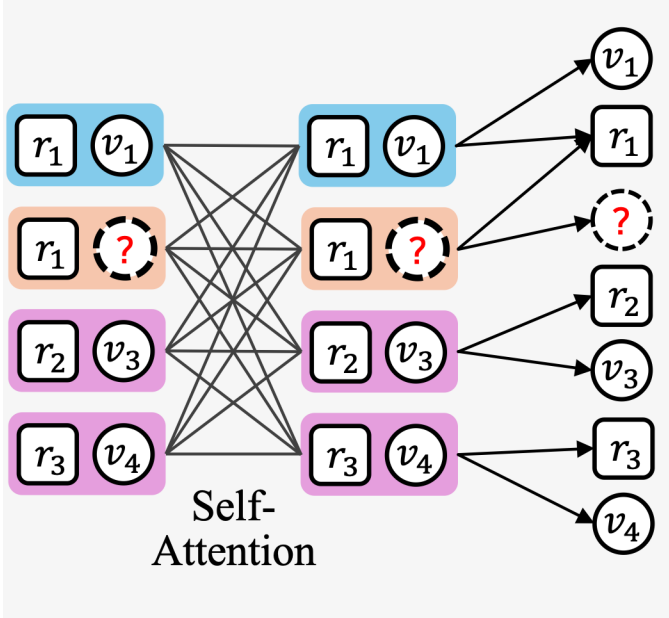




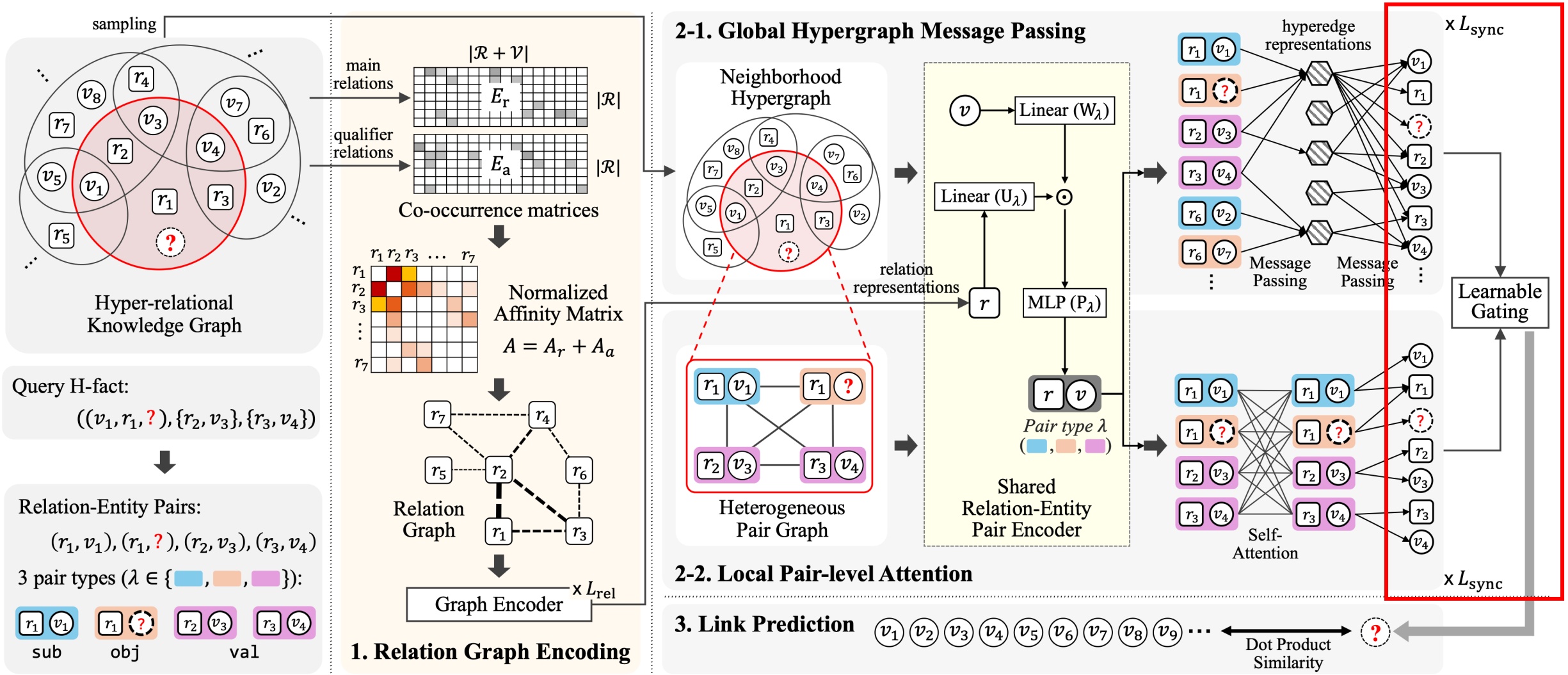
Local Pair-level Self-Attention (cont.)

- Intra-fact Pair-level Self-Attention
  - ✓ After constructing the pair graph, we obtain initial embedding for each pair using the Relation-Entity Pair encoder
  - ✓ Then, we update pair's representation by using self-attention

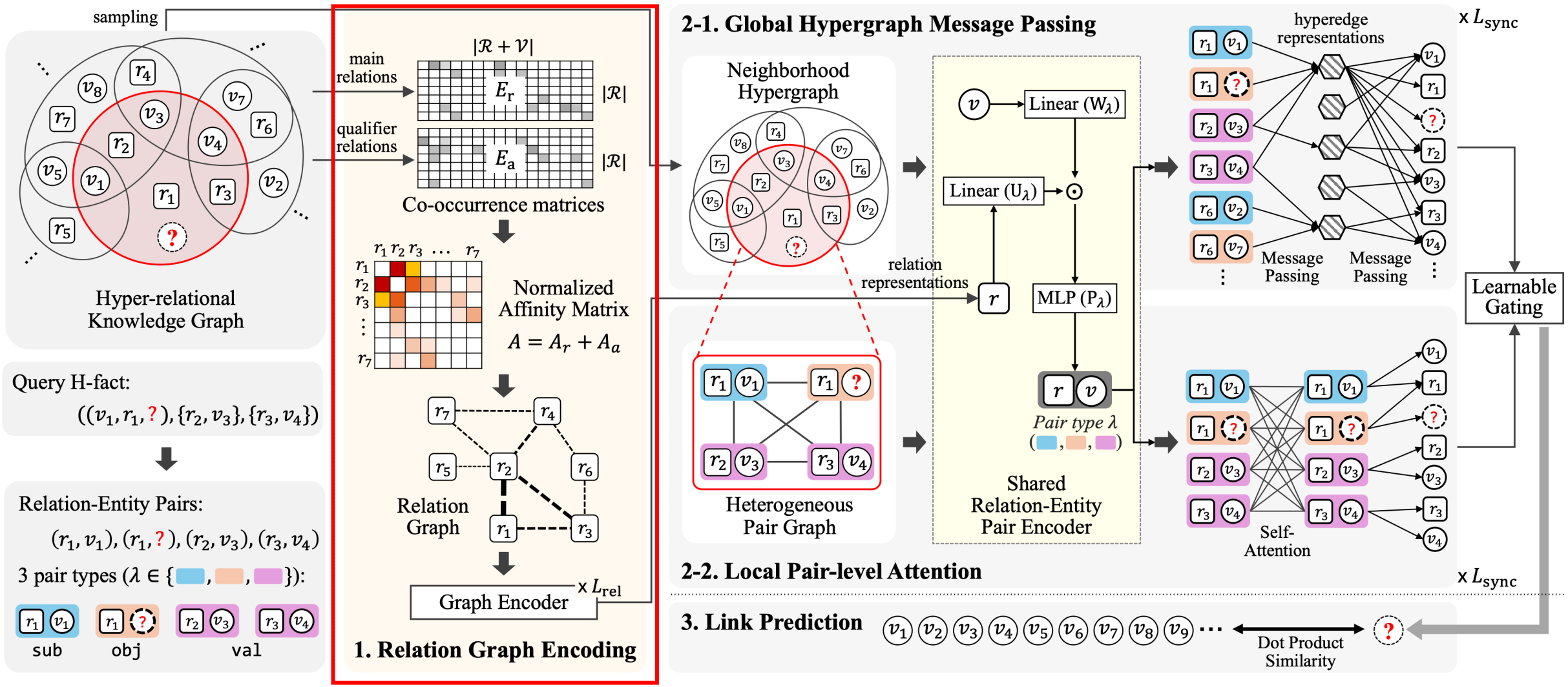
- Node Update from Pair Representations
  - ✓ In the local stream, both entity and relation nodes are updated from pair embeddings conditioned on pair types
    - By position-aware mechanism



Bridging Global and Local Representations

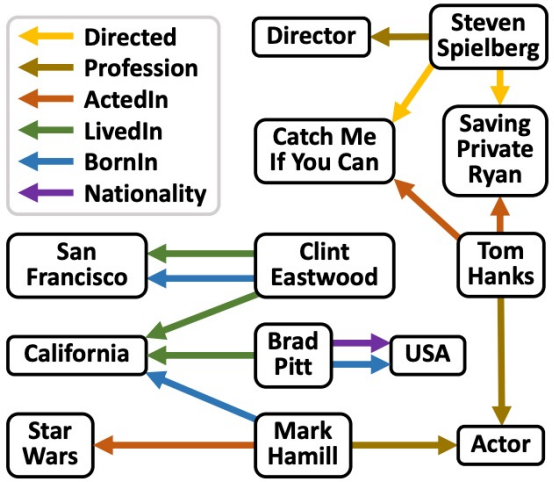


Relation Graph Encoding

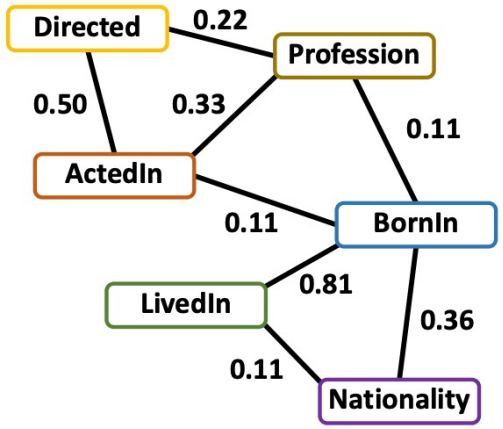


Relation Graph

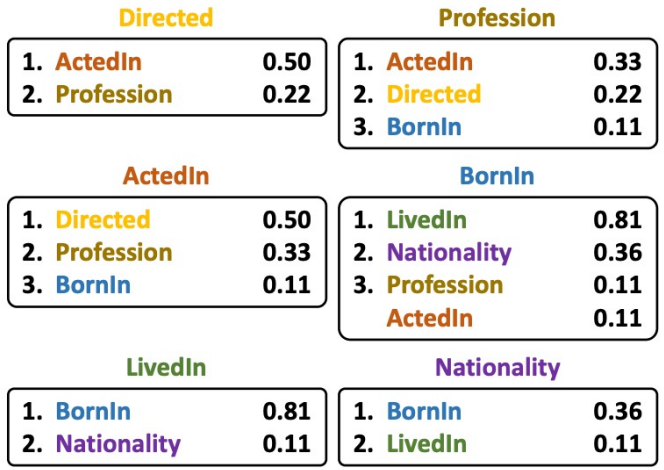
- InGram에 영감을 받아, Relation Graph를 구축
  - ✓ Binary Knowledge Graph와 달리,
  - ✓ HKG에서는 H-fact에 1개 이상의 relation, 2개 이상의 entity가 등장할 수 있으므로, 확장이 필요함



(a) Knowledge Graph



(b) Relation Graph



(c) Affinity scores of the relations

InGram: Inductive Knowledge Graph Embedding via Relation Graphs (ICML'23)

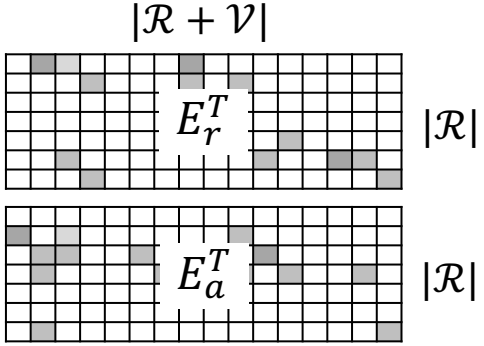
Relation Graph (cont.)

- We construct the relation affinity graph by computing relation-relation affinity based on the co-occurrence of relation-context pairs
  - ✓ Each relation co-occurs with
    - (i) its connected entities
    - (ii) other relations appearing in the same H-fact

H-fact #2: ((Max_Verstappen, driver_in, F1), {(in_year, Y2025)})

  - Main relation "driver_in" 은 entity "Max_Verstappen", "F1" 과, relation "in_year" 랑 co-occur
  - Qualifier relation "in_year"는 entity "Y2025"와. relation "driver_in"이랑 co-occur

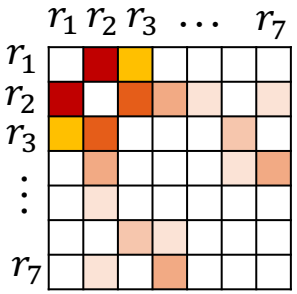
- To formalize this, we define two co-occurrence matrices E_r, E_a and derive the normalized affinity matrices



Co-occurrence matrices

$$A_r := E_r^\top D_r^{-2} E_r, \quad D_r[i,i] := \sum_j E_r[i,j] \qquad (2)$$

$$A_a := E_a^\top D_a^{-2} E_a, \quad D_a[i,i] := \sum_j E_a[i,j] \qquad (3)$$

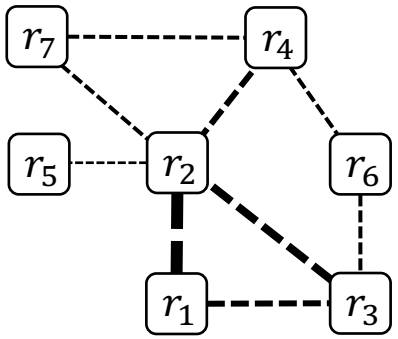


Normalized Affinity Matrix

$$A = A_r + A_a$$

Relation Graph Encoder

- Relation graph



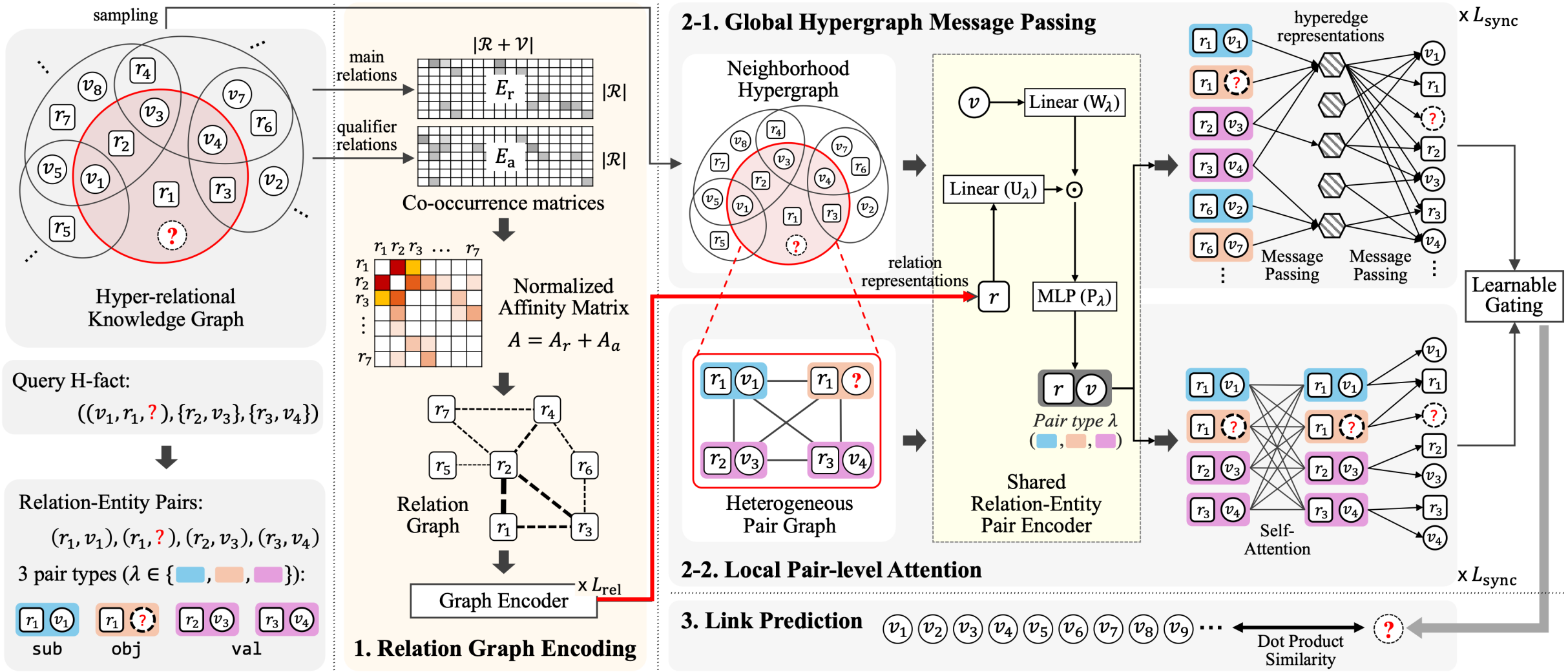
- Relation graph에서 attention-based message passing을 사용하여 relation에 대한 representation을 얻음

$$e_{ij}^{(\ell)} = \mathbf{u}^{(\ell)\top} \sigma\left(P^{(\ell)}\left[\, z_i^{(\ell)} \| \, z_j^{(\ell)} \,\right]\right) + c_{s(i,j)}^{(\ell)} \tag{4}$$

$$z_i^{(\ell+1)} = \sigma\left(\sum_{j \in \mathcal{N}_i} \frac{\exp\left(e_{ij}^{(\ell)}\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(e_{ik}^{(\ell)}\right)} W^{(\ell)} z_j^{(\ell)}\right) \tag{5}$$

- 이렇게 얻은 relation representation은 main GLoRE 모듈에서 사용하게 됨
  - ✓ 구체적으로는, shared relation-entity pair encoder의 입력으로 들어감

Relation Graph Encoder (cont.)



CONTENTS

Dataset

Dataset	H-Facts	H-Facts with Q(%)	Entities	Relations	Train	Valid	Test
JF17K	100,947	46,320 (45.9%)	28,645	501	76,379	–	24,568
WikiPeople$^-$	369,866	9,482 (2.6%)	34,825	178	294,439	37,715	37,712
WD50K	236,507	32,167 (13.6%)	47,156	532	166,435	23,913	46,159
WD50K(33)	102,107	31,866 (31.2%)	38,124	475	73,406	10,668	18,133
WD50K(66)	49,167	31,696 (64.5%)	27,347	494	35,968	5,154	8,045
WD50K(100)	31,314	31,314 (100%)	18,792	279	22,738	3,279	5,297

Table 2: Statistics of all datasets. "H-Facts with Q(%)" denotes the number and proportion of H-facts with ≥1 qualifier.

Research Questions

- **RQ1:** Does GLoRE outperform existing HKG embedding methods on standard link prediction benchmarks?
- **RQ2:** How does node vs. pair aggregation affect role mixing and embedding concentration?
- **RQ3:** How do global-local fusion designs (joint vs. cascaded fusion, use of shared pair encoder) affect performance?
- **RQ4:** What is the contribution of each key architectural component to GLoRE's performance?
- **RQ5:** What is the training and inference efficiency of GLoRE compared to representative HKG baselines?

Overall Performance (RQ1)

Model	JF17K						WikiPeople⁻						WD50K					
	main entities			all entities			main entities			all entities			main entities			all entities		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
m-TransH	0.206	0.206	0.462	0.102	0.069	0.168	0.063	0.063	0.300	–	–	–	–	–	–	–	–	–
RAE	0.215	0.215	0.466	0.310	0.219	0.504	0.059	0.059	0.306	0.172	0.102	0.320	–	–	–	–	–	–
NaLP	0.221	0.165	0.331	0.366	0.290	0.516	0.408	0.331	0.546	0.338	0.272	0.466	0.177	0.131	0.264	0.224	0.158	0.330
HINGE	0.431	0.342	0.611	0.517	0.436	0.675	0.342	0.272	0.463	0.350	0.282	0.467	0.243	0.176	0.377	0.232	0.164	0.343
GRAN	0.617	0.539	0.770	0.656	0.582	0.799	0.503	0.438	0.620	0.479	0.410	0.604	–	–	–	0.309	0.240	0.441
HyperFormer	0.664	0.601	0.787	–	–	–	0.473	0.361	0.646	–	–	–	0.366	0.288	0.514	–	–	–
StarE	0.572	0.493	0.725	0.542	0.454	0.685	0.491	0.398	0.592	0.378	0.265	0.542	0.349	0.271	0.496	–	–	–
HAHE	0.623	0.554	0.806	0.668	0.597	0.816	0.509	0.447	0.639	0.495	0.420	0.631	0.368	0.291	0.516	0.402	0.327	0.546
MAYPL	–	–	–	–	–	–	0.519	0.444	0.657	0.521	0.446	0.659	0.381	0.297	0.544	0.411	0.326	0.572
GLoRE (ours)	**0.700**	**0.616**	**0.853**	**0.719**	**0.632**	**0.877**	**0.554**	**0.451**	**0.743**	**0.555**	**0.451**	**0.746**	**0.478**	**0.370**	**0.684**	**0.512**	**0.401**	**0.718**
std.	*(±.005)*	*(±.007)*	*(±.002)*	*(±.001)*	*(±.004)*	*(±.004)*	*(±.002)*	*(±.004)*	*(±.003)*	*(±.002)*	*(±.004)*	*(±.004)*	*(±.007)*	*(±.006)*	*(±.008)*	*(±.006)*	*(±.005)*	*(±.007)*
Impv.	+5.4%	+2.5%	+5.8%	+7.6%	+5.9%	+7.5%	+6.7%	+0.8%	+13.1%	+6.5%	+1.1%	+13.2%	+25.5%	+24.6%	+25.7%	+24.6%	+22.6%	+25.5%

Aggregation Unit: Embedding Concentration and Role Mixing (RQ2)

- **NODE**: Global stream을 node-centric aggregation으로 수정한 variant

- [C1]의 Embedding Concentration and Role Mixing을 정말 잘 완화하는가?

Node-space Embedding Concentration			
Agg. Unit	CCIc (↓)	ERank/PR (↑)	Top-10 (↓)
PAIR (GLoRE)	**0.0022**	**150.91/44.37**	**0.251**
NODE	0.0383	29.48/4.05	0.564
Pair-space Role Separability			
Agg. Unit	RSS (↑)	RNCA(holdout) (↑)	ΔRNCA (↑)
PAIR (GLoRE)	**0.2210**	**0.698**	**0.296**
NODE	0.0521	0.236	0.040

Table 4: Aggregation unit diagnostics on WD50K (↑ better / ↓ better).

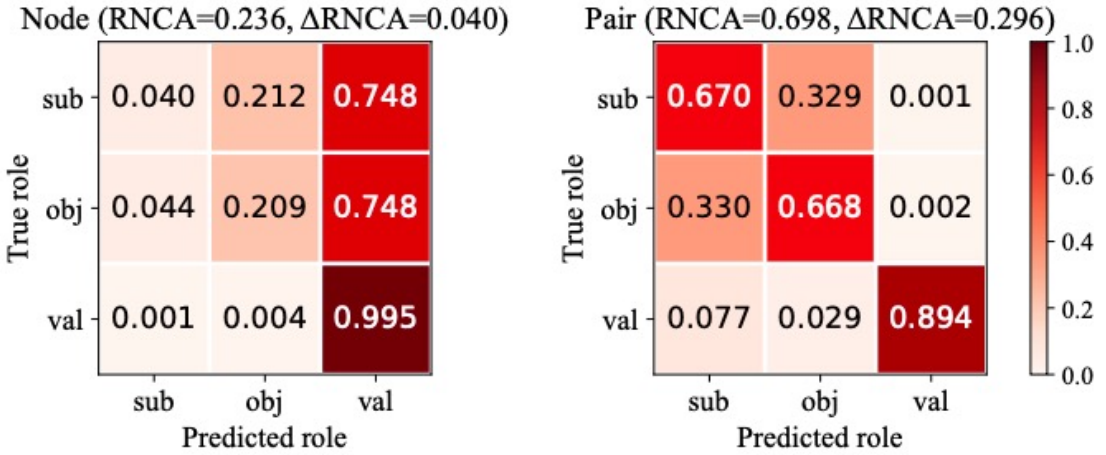
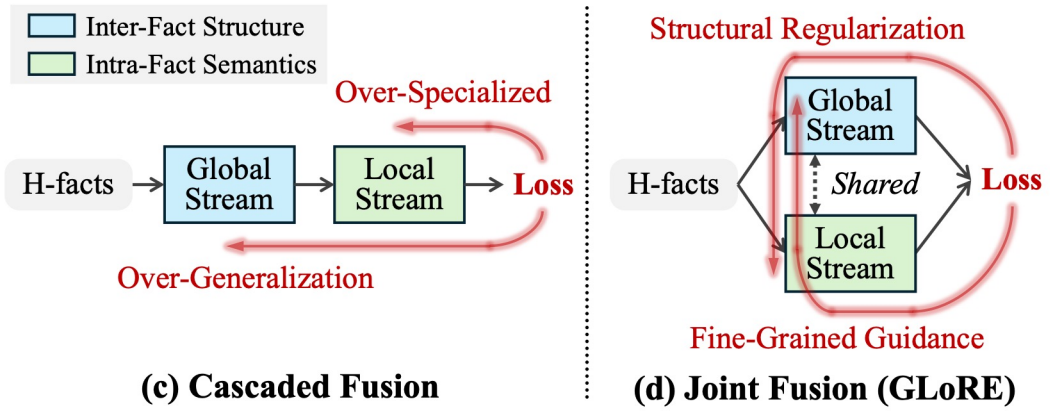


Figure 3: RNCA role confusion on WD50K. Row-normalized confusion matrices from nearest role-centroid classification: NODE exhibits strong cross-role confusion (sub/obj→val), while PAIR yields a sharper diagonal (higher RNCA/ΔRNCA).

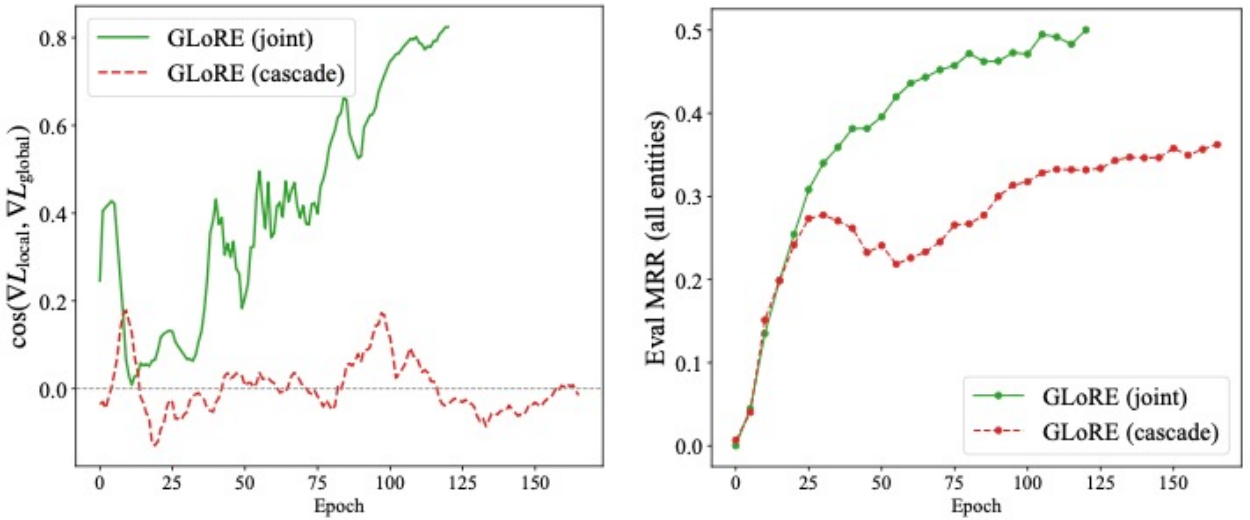
Global-Local Fusion Strategies (RQ3)

- [C2]의 Insufficient Global-Local Fusion을 잘 완화하는가?



	main entities			all entities		
	MRR	H@1	H@10	MRR	H@1	H@10
GLoRE (ours)	**0.478**	**0.370**	**0.684**	**0.512**	**0.401**	**0.718**
w/o sharing pair encoder	0.449	0.345	0.651	0.479	0.361	0.680
$\text{GLoRE}_{\text{cascaded}}$	0.332	0.241	0.508	0.364	0.268	0.546
w/o sharing pair encoder	0.352	0.260	0.530	0.383	0.289	0.567

Table 5: Global-Local fusion strategies on WD50K.

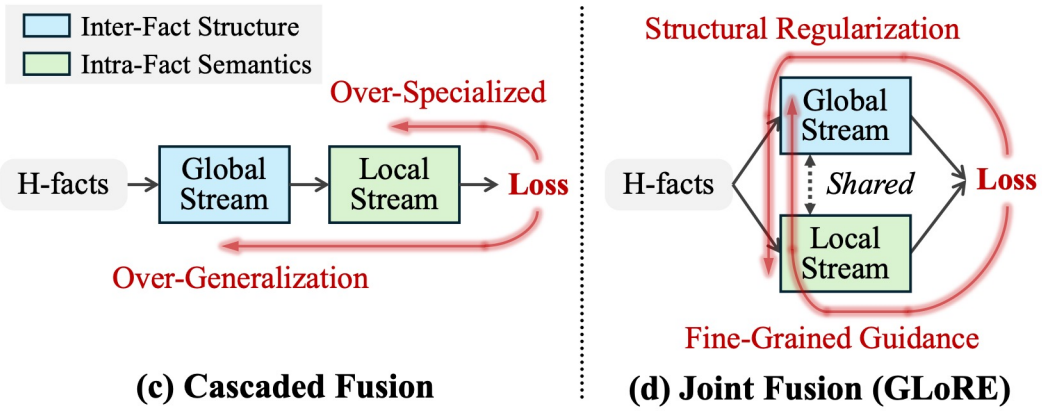


(a) Gradient alignment (b) Evaluation MRR

Figure 4: Optimization dynamics under *joint* vs. *cascaded* fusion (both w/ sharing pair encoder). (a) Gradient alignment on the shared pair encoder, measured as $\cos(\nabla_\theta \mathcal{L}_{\text{local}}, \nabla_\theta \mathcal{L}_{\text{global}})$. (b) Evaluation MRR at 5-epoch intervals.

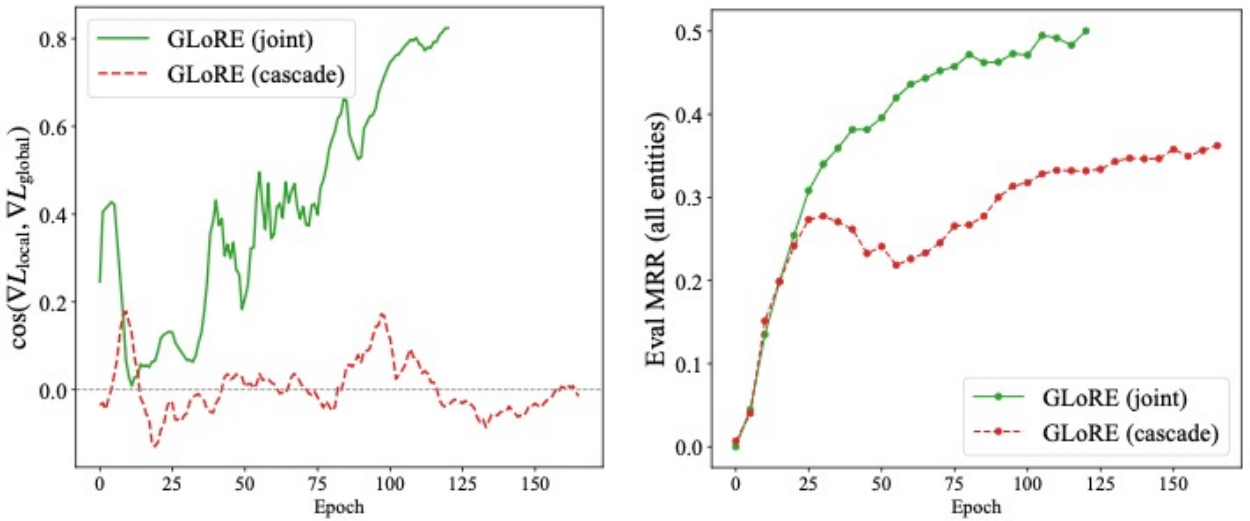
Global-Local Fusion Strategies (RQ3)

- Sharing relation-entity pair encoder의 역할?



	main entities			all entities		
	MRR	H@1	H@10	MRR	H@1	H@10
GLoRE (ours)	**0.478**	**0.370**	**0.684**	**0.512**	**0.401**	**0.718**
w/o sharing pair encoder	0.449	0.345	0.651	0.479	0.361	0.680
GLoRE$_{\text{cascaded}}$	0.332	0.241	0.508	0.364	0.268	0.546
w/o sharing pair encoder	0.352	0.260	0.530	0.383	0.289	0.567

Table 5: Global-Local fusion strategies on WD50K.



(a) Gradient alignment (b) Evaluation MRR

Figure 4: Optimization dynamics under *joint* vs. *cascaded* fusion (both w/ sharing pair encoder). (a) Gradient alignment on the shared pair encoder, measured as $\cos(\nabla_\theta \mathcal{L}_{\text{local}}, \nabla_\theta \mathcal{L}_{\text{global}})$. (b) Evaluation MRR at 5-epoch intervals.

Ablation Study (RQ4)

Model	JF17K						WikiPeople⁻						WD50K					
	main entities			all entities			main entities			all entities			main entities			all entities		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
GLoRE (ours)	**0.700**	**0.616**	**0.853**	**0.719**	**0.632**	**0.877**	**0.554**	**0.451**	**0.743**	**0.555**	**0.451**	**0.746**	**0.478**	**0.370**	**0.684**	**0.512**	**0.401**	**0.718**
w/o global stream	0.606	0.530	0.760	0.647	0.576	0.791	0.447	0.347	0.598	0.449	0.350	0.600	0.315	0.246	0.447	0.346	0.279	0.475
w/o relation graph	0.649	0.562	0.803	0.679	0.590	0.837	0.505	0.411	0.674	0.507	0.413	0.678	0.426	0.319	0.630	0.460	0.353	0.661
w/o sharing pair encoder	0.686	0.607	0.851	0.701	0.620	0.867	0.523	0.427	0.716	0.524	0.429	0.720	0.449	0.345	0.651	0.479	0.361	0.680
w/o learnable gating	0.695	0.612	0.846	0.713	0.630	0.868	0.549	0.447	0.737	0.550	0.447	0.739	0.468	0.359	0.671	0.498	0.388	0.702
w/o ONR	0.694	0.610	0.849	0.717	0.628	0.878	0.541	0.443	0.719	0.543	0.443	0.722	0.461	0.349	0.667	0.491	0.380	0.697
NODE (§5.3)	0.612	0.537	0.786	0.652	0.581	0.816	0.459	0.353	0.622	0.461	0.356	0.625	0.348	0.252	0.523	0.376	0.279	0.556

Table 9: Full results for the ablations in Table 6 and additional variants across all metrics and evaluation settings.

Further Analysis

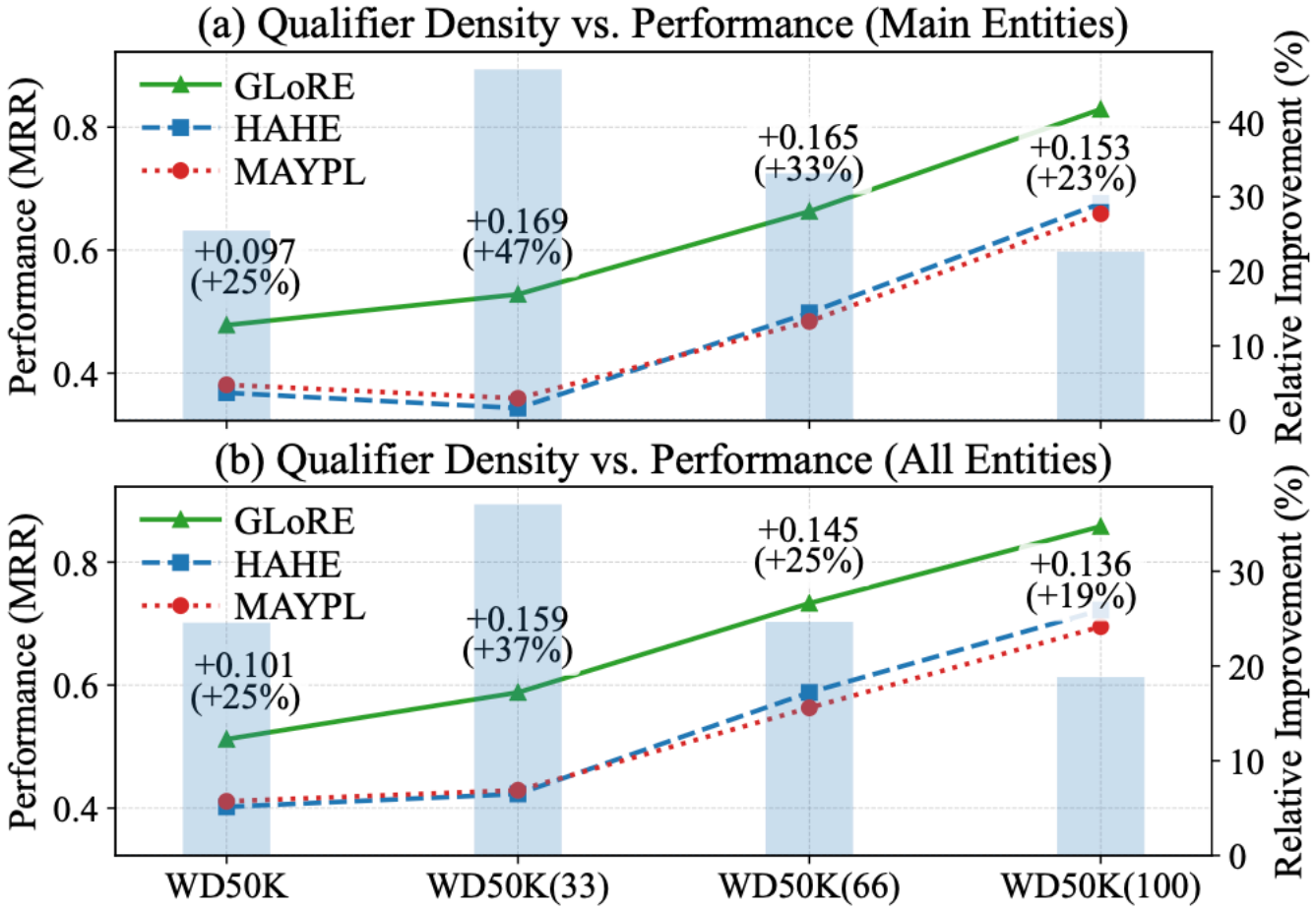


Figure 5: Robustness to Qualifier Density. GLoRE ranks first at every density level; baseline rankings shift. Bars show relative gain over the density-specific best baseline.

CONTENTS

Conclusion & Contribution

- We identify **two recurring failure modes** in HKG representation learning:
  - ✓ 1) entity-role evidence mixing and representation collapse under **node-centric global aggregation**
  - ✓ 2) **Insufficient global-local fusion** that limits interaction between global structure and intra-fact semantics

- We propose **GLoRE**, a **pair-centric joint global-local model**
  - ✓ that performs global hypergraph propagation and local intra-fact self-attention **over relation–entity pairs**,
  - ✓ and optimizes the two components end-to-end with a **shared pair encoder**

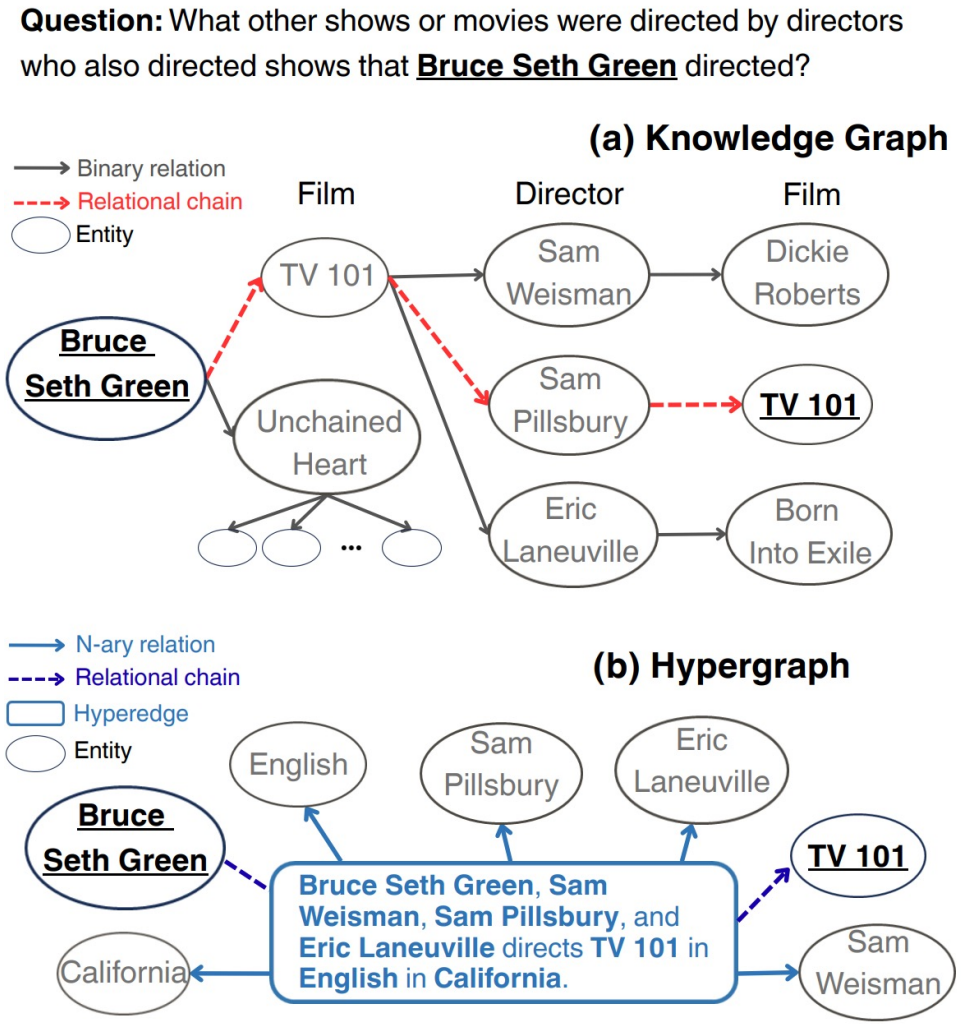
- We construct a weighted **relation graph from within-fact co-occurrences** (over relations and entities),
  - ✓ refining relation representations to provide relation affinity priors for pair encoding

- We conduct comprehensive experiments with ablations and analyses,
  - ✓ achieving top-ranked performance across all qualifier densities (up to +25.5% relative MRR on WD50K)

CONTENTS

Hypergraph-based RAG

- topic entity에 기반한 질문에 답하기 위해
  - ✓ standard KG에서는 **3-hop binary traversal**이 필요
- 반면 hypergraph에서는
  - ✓ 하나의 n-ary relation을 통해 **single-hop**으로 처리할 수 있음
- 따라서 hypergraph는 higher-order relational chain을 직접 modeling할 수 있으며,
  - ✓ Semantic Fragmentation을 완화하고 complex dependency를 포착하는 데 필요한 reasoning step을 줄일 수 있음



[2026][WWW][HyperRAG] HyperRAG: Reasoning N-ary Facts over Hypergraphs for Retrieval Augmented Generation

HyperGraphRAG (NIPS'25)

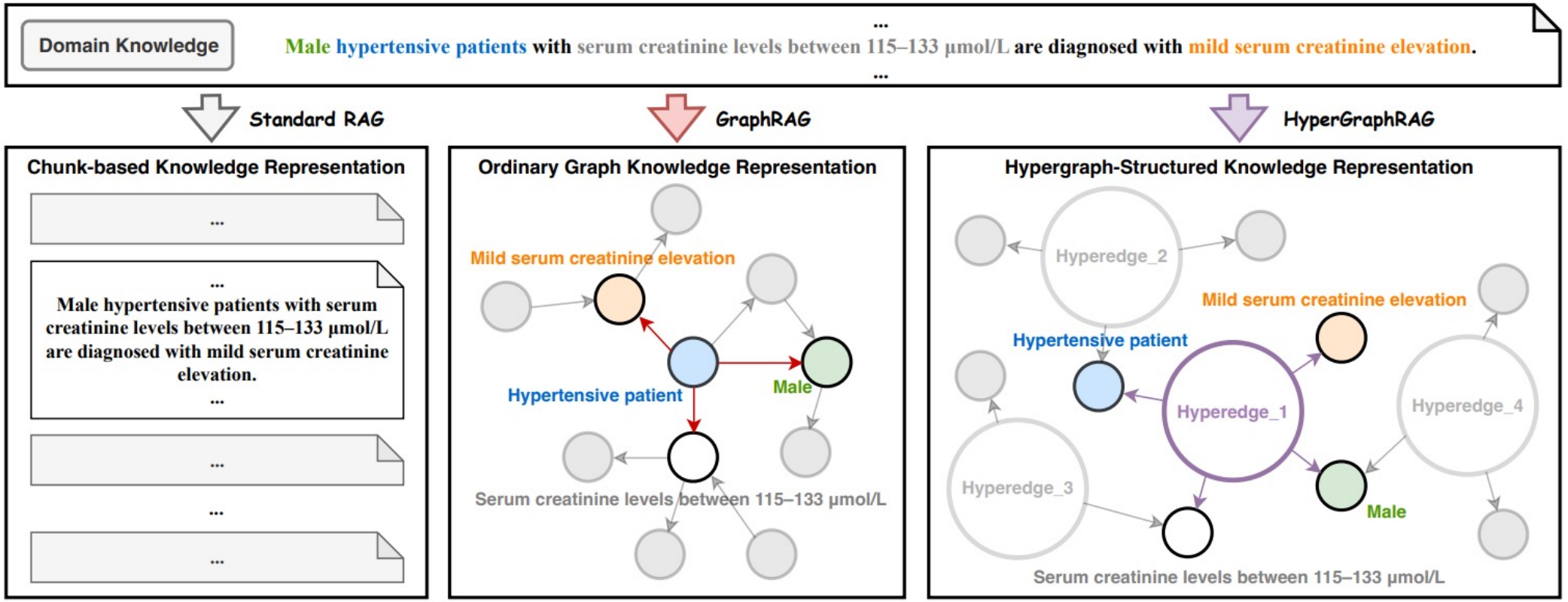
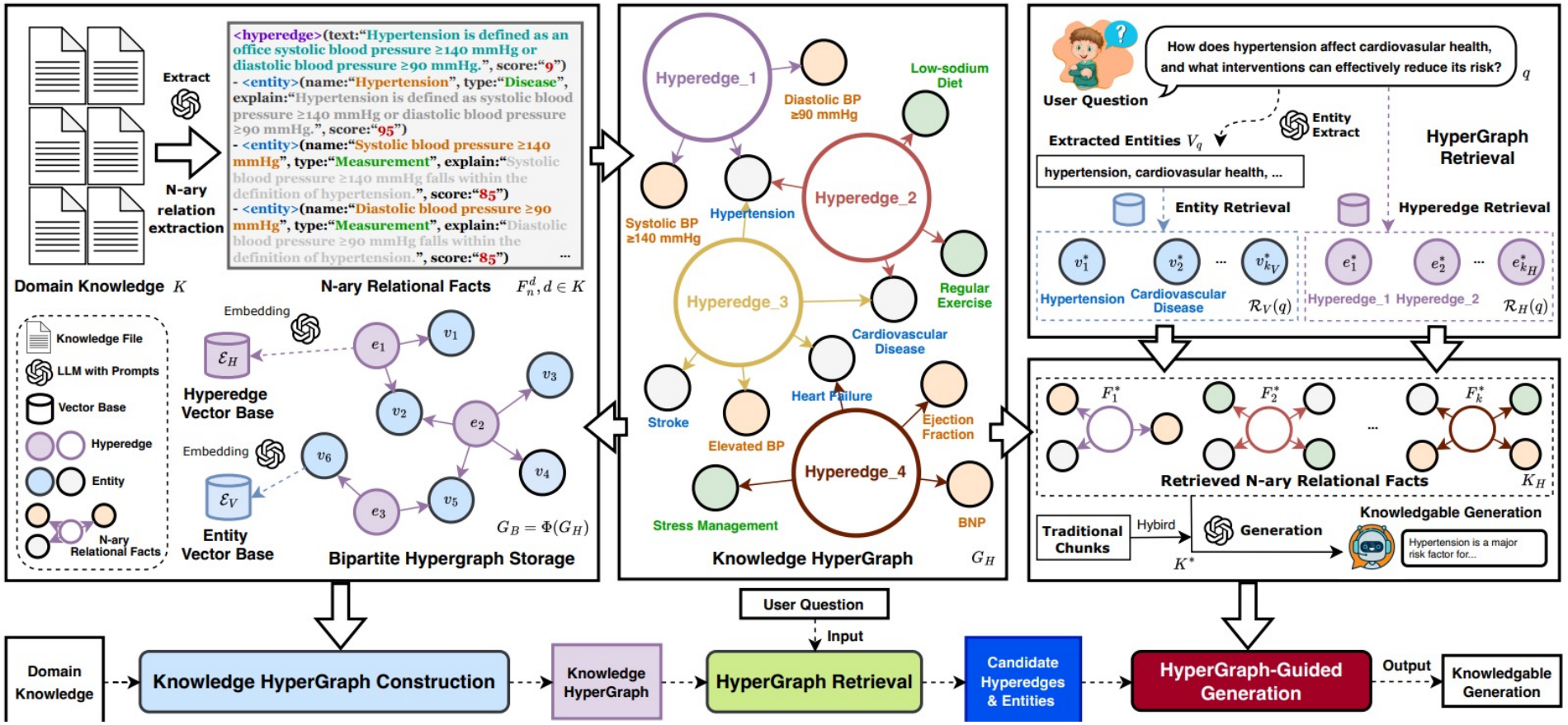


Figure 2: Comparison of knowledge representation: standard RAG uses chunks as units, GraphRAG captures binary relations with graphs, and HyperGraphRAG models n-ary relations with hyperedges.

[2025][NIPS][HyperGraphRAG] HyperGraphRAG: Retrieval-Augmented Generation with Hypergraph-Structured Knowledge Representation

HyperGraphRAG (NIPS'25)

- Document로부터 LLM으로 n-ary relational fact를 추출해 hyperedge로 저장하고,
  - ✓ entity retrieval과 hyperedge retrieval을 함께 수행한 뒤,
  - ✓ 확장된 hypergraph 지식과 기존 chunk를 결합해 답변을 생성하는 하이브리드 RAG framework



[2025][NIPS][HyperGraphRAG] HyperGraphRAG: Retrieval-Augmented Generation with Hypergraph-Structured Knowledge Representation

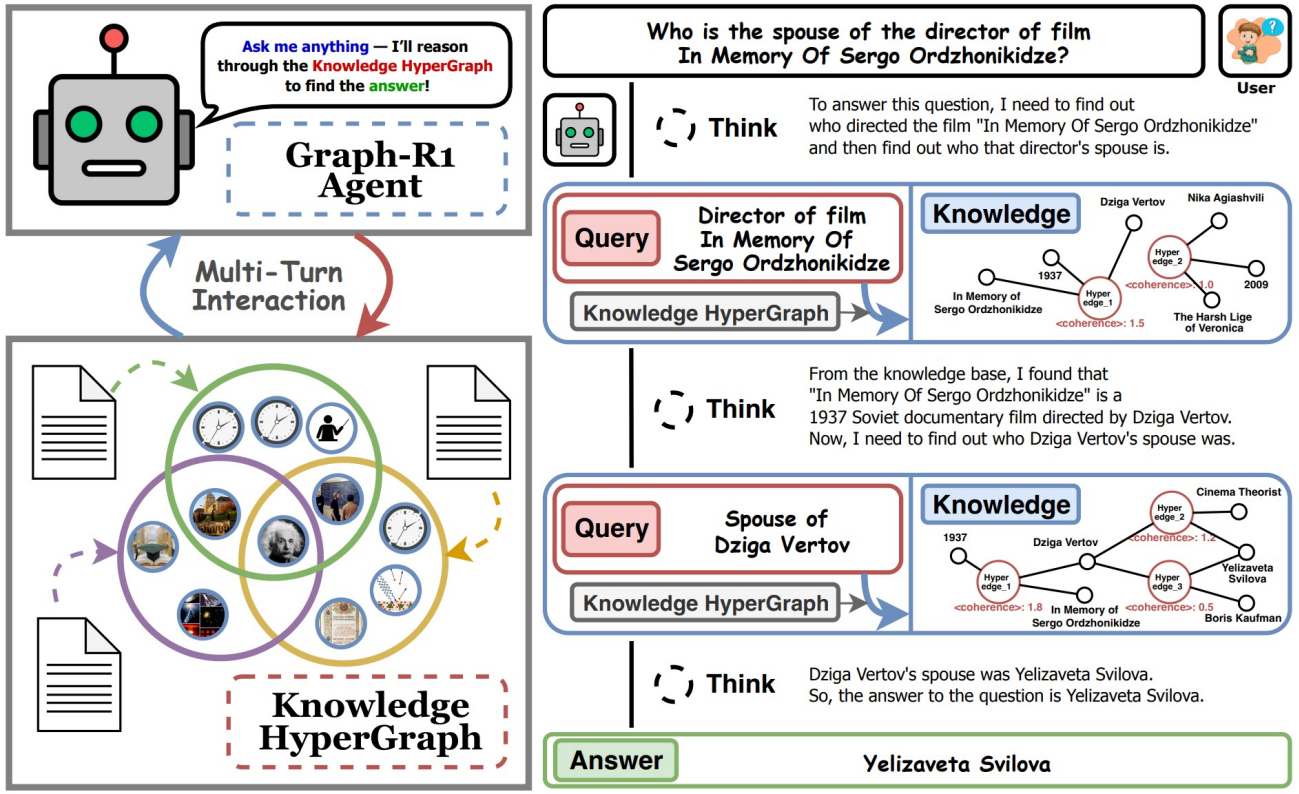
HyperGraphRAG (NIPS'25)

Table 2: Performance comparison across different domains. **Bold** indicates the best performance.

Method	Medicine			Agriculture			CS			Legal			Mix		
	F1	R-S	G-E	F1	R-S	G-E	F1	R-S	G-E	F1	R-S	G-E	F1	R-S	G-E
Binary Source															
NaiveGeneration	12.63	0.00	44.70	11.71	0.00	45.76	18.93	0.00	48.79	22.91	0.00	50.00	18.58	0.00	46.14
StandardRAG	26.87	61.08	56.24	28.31	42.69	57.58	28.87	49.44	57.10	37.19	52.21	59.85	47.57	46.79	67.42
GraphRAG	17.13	54.56	48.19	20.67	40.90	52.41	23.75	37.65	53.17	31.09	34.26	54.62	23.62	25.01	48.12
LightRAG	12.16	52.38	44.15	17.70	41.24	50.32	22.59	41.86	51.62	33.63	45.54	56.42	29.98	34.22	54.50
PathRAG	14.74	52.30	45.36	21.97	42.21	53.13	25.28	41.49	53.28	32.32	43.60	55.45	40.87	33.36	60.75
HippoRAG2	21.12	57.50	51.08	12.60	16.85	44.56	16.94	21.05	47.29	20.10	34.13	46.77	21.10	18.34	45.83
HyperGraphRAG (ours)	**36.45**	**69.91**	**60.65**	**34.80**	**61.97**	**59.99**	**31.60**	**60.94**	**57.54**	**44.42**	**60.87**	**63.53**	**51.51**	**67.34**	**68.76**
N-ary Source															
NaiveGeneration	13.15	0.00	41.83	13.78	0.00	47.93	18.37	0.00	48.94	20.37	0.00	48.09	15.29	0.00	45.16
StandardRAG	28.93	64.06	55.08	26.55	48.93	56.62	28.99	47.35	56.69	37.50	51.16	60.09	38.83	47.73	61.82
GraphRAG	18.07	57.22	47.09	21.90	41.27	53.49	22.90	39.97	53.76	29.12	34.11	53.76	14.93	24.32	42.32
LightRAG	13.43	54.67	41.86	18.78	42.44	50.92	22.85	41.19	52.20	29.64	44.47	54.65	24.08	33.22	50.83
PathRAG	15.14	54.08	42.77	20.64	42.53	51.83	28.18	42.29	54.97	30.27	44.47	55.26	33.27	34.11	57.47
HippoRAG2	21.56	61.54	48.06	12.66	20.32	45.14	17.75	26.92	48.44	16.95	34.72	45.09	21.95	18.49	46.87
HyperGraphRAG (ours)	**34.26**	**70.48**	**58.06**	**32.98**	**62.58**	**59.59**	**31.00**	**59.25**	**58.35**	**43.20**	**60.07**	**63.70**	**45.91**	**69.09**	**65.04**
Overall															
NaiveGeneration	12.89	0.00	43.27	12.74	0.00	46.85	18.65	0.00	48.87	21.64	0.00	49.05	16.93	0.00	45.65
StandardRAG	27.90	62.57	55.66	27.43	45.81	57.10	28.93	48.40	56.89	37.34	51.68	59.97	43.20	47.26	64.62
GraphRAG	17.60	55.89	47.64	21.28	41.08	52.95	23.33	38.81	53.47	30.11	34.18	54.19	19.27	24.67	45.22
LightRAG	12.79	53.52	43.00	18.24	41.84	50.62	22.72	41.53	51.91	31.64	45.00	55.53	27.03	33.72	52.67
PathRAG	14.94	53.19	44.06	21.30	42.37	52.48	26.73	41.89	54.13	31.29	44.03	55.36	37.07	33.73	59.11
HippoRAG2	21.34	59.52	49.57	12.63	18.58	44.85	17.34	23.99	47.87	18.53	34.42	45.93	21.53	18.42	46.35
HyperGraphRAG (ours)	**35.35**	**70.19**	**59.35**	**33.89**	**62.27**	**59.79**	**31.30**	**60.09**	**57.94**	**43.81**	**60.47**	**63.61**	**48.71**	**68.21**	**66.90**

Graph-R1 (ICML'26)

- 대부분의 GraphRAG는 one-shot retrieval에 가까워,
  - ✓ 복잡한 query에서 필요한 knowledge를 단계적으로 탐색하기 어렵다는 한계가 존재
- 이에, GraphRAG를 단순 retrieval pipeline이 아니라,
  - ✓ knowledge hypergraph environment 위에서 agent가 multi-turn으로 탐색하는 문제로 재정의

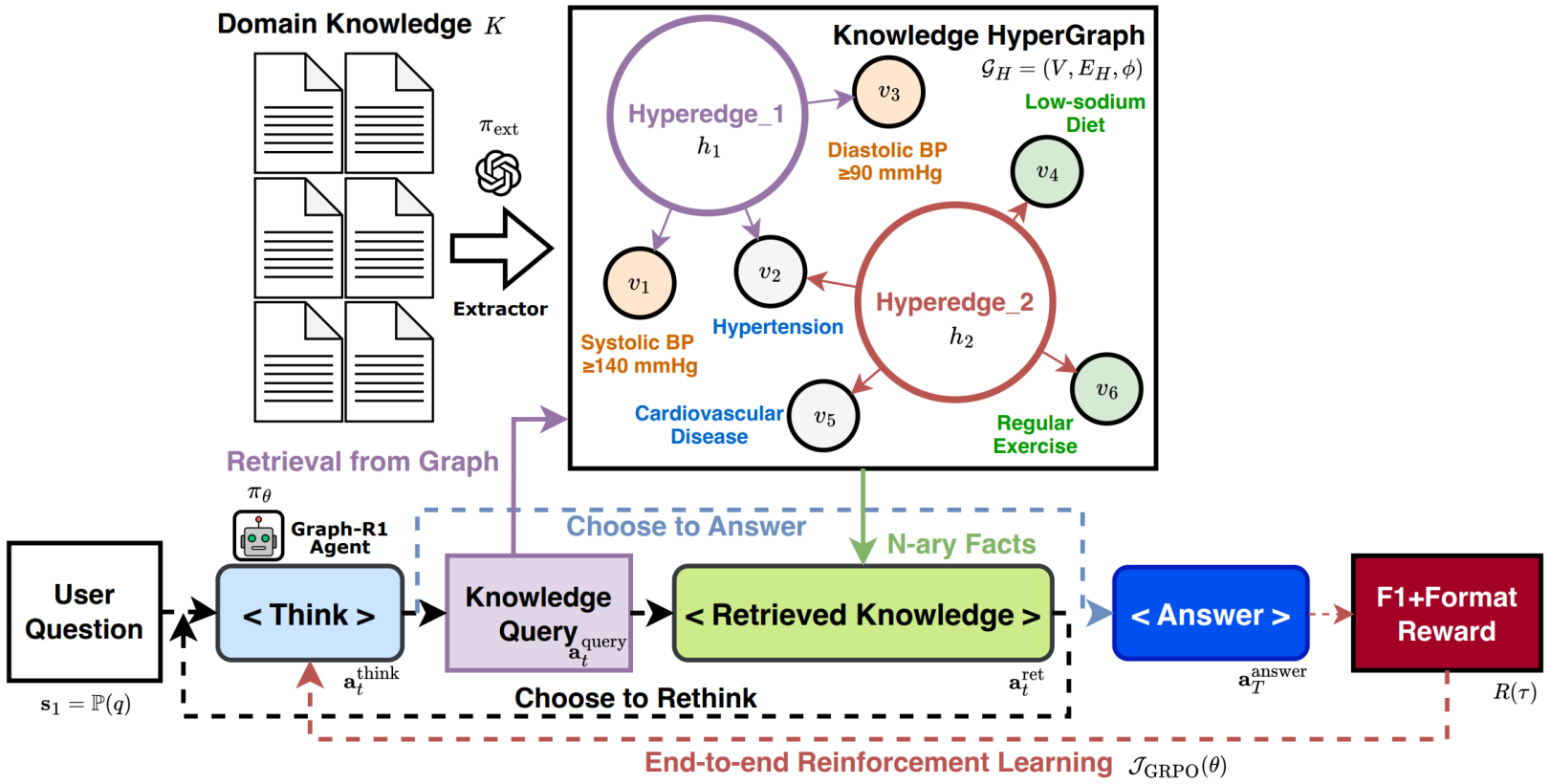


[2026][ICML][Graph-R1] Graph-R1: Towards Agentic GraphRAG Framework via End-to-end Reinforcement Learning

Graph-R1 (ICML'26)

- Document로부터 n-ary relational fact를 추출해 knowledge hypergraph를 만들고,
  - ✓ agent가 think → query → retrieve → rethink → answer 과정을 반복하도록 함

- GRPO 기반 RL을 사용해 agent가 언제 retrieval을 계속하고, 언제 answer를 생성할지 학습



[2026][ICML][Graph-R1] Graph-R1: Towards Agentic GraphRAG Framework via End-to-end Reinforcement Learning

Graph-R1 (ICML'26)

Table 2. Main results with best in **bold**. means prompt engineering, means training, means no knowledge interaction, means chunk-based knowledge, and means graph-based knowledge.

Method	2Wiki.		HotpotQA		Musique		NQ		PopQA		TriviaQA		Avg.			
	F1	G-E	F1	G-E	F1	G-E	F1	G-E	F1	G-E	F1	G-E	EM	F1	R-S	G-E
GPT-4o-mini																
NaiveGeneration	17.03	74.86	31.79	78.48	11.45	76.61	21.59	84.64	25.95	**72.75**	47.73	83.33	11.36	25.92	-	78.45
StandardRAG	**22.31**	73.02	**46.70**	**81.88**	17.31	74.93	**26.85**	84.55	**30.58**	69.42	**48.55**	84.63	**18.10**	**32.05**	52.68	78.07
GraphRAG	16.02	72.81	31.67	77.37	15.14	74.43	20.31	82.36	20.92	65.88	45.13	82.76	12.50	24.87	32.48	75.94
LightRAG	16.59	71.94	30.70	73.42	14.39	73.75	19.09	80.20	20.47	67.76	40.18	81.60	9.77	23.57	47.42	74.78
PathRAG	12.42	67.19	23.12	71.81	11.49	69.94	20.01	81.99	15.65	60.58	37.44	80.94	7.03	20.02	46.71	72.08
HippoRAG2	16.27	68.78	31.78	76.43	12.37	73.05	24.56	**84.65**	21.10	63.31	46.86	83.55	13.80	25.49	36.41	74.96
HyperGraphRAG	21.14	**76.76**	37.46	80.50	**20.40**	**79.29**	22.95	81.22	29.48	70.55	44.95	**85.20**	13.15	29.40	**61.82**	**78.92**
Qwen2.5-1.5B-Instruct																
NaiveGeneration	7.78	49.13	4.27	45.77	2.35	46.63	6.03	46.74	10.06	42.67	8.10	52.92	1.17	6.43	-	47.31
StandardRAG	11.46	55.38	9.93	52.91	3.18	39.46	11.39	59.73	13.08	50.29	17.43	60.52	5.73	11.08	52.84	53.05
SFT	13.26	34.72	13.61	38.93	5.14	28.50	11.56	46.61	15.61	31.35	26.18	46.66	9.83	14.23	-	37.80
R1	26.28	47.48	20.07	44.43	4.84	39.12	16.75	45.95	21.36	44.50	34.78	48.59	14.19	20.68	-	45.01
Search-R1	28.43	60.61	39.99	64.16	4.69	39.32	20.26	59.93	39.63	58.19	44.16	63.01	23.18	29.53	50.45	57.54
R1-Searcher	28.01	58.81	**41.50**	61.54	6.26	38.31	**36.86**	**60.79**	38.37	56.02	42.57	61.24	23.70	32.26	50.68	56.12
Graph-R1 (ours)	**35.13**	**65.73**	40.62	**65.30**	**28.28**	**58.82**	35.62	59.13	**43.55**	**66.46**	**57.36**	**70.83**	**31.90**	**40.09**	**59.35**	**64.38**
Qwen2.5-3B-Instruct																
NaiveGeneration	7.59	55.00	11.16	53.75	3.67	54.00	8.90	57.18	10.89	49.08	10.89	48.16	3.26	8.85	-	52.86
StandardRAG	12.52	60.01	15.41	62.51	2.92	50.40	10.69	65.13	14.70	57.25	21.92	68.43	3.39	13.03	52.69	60.62
SFT	12.40	52.31	16.48	51.35	5.04	51.31	11.23	58.20	16.95	46.42	33.02	59.98	9.64	15.85	-	53.26
R1	28.45	56.92	25.33	55.38	8.07	47.53	21.51	55.11	27.11	48.65	47.91	60.74	19.66	26.40	-	54.06
Search-R1	38.04	54.39	43.84	69.32	7.65	46.43	37.96	52.90	38.67	63.74	47.99	60.37	28.65	35.69	49.99	57.86
R1-Searcher	23.50	55.86	42.44	64.60	12.81	50.07	36.53	63.33	40.18	66.23	54.00	60.52	27.08	34.91	49.98	60.10
Graph-R1 (ours)	**57.56**	**76.45**	**56.75**	**77.46**	**40.51**	**67.84**	**44.75**	**69.92**	**45.65**	**71.27**	**62.31**	**75.01**	**42.45**	**51.26**	**60.19**	**72.99**
Qwen2.5-7B-Instruct																
NaiveGeneration	12.25	66.75	16.58	65.31	4.06	65.47	13.00	69.56	12.82	60.50	24.51	72.65	3.12	13.87	-	66.71
StandardRAG	12.75	60.06	21.10	66.13	4.53	59.84	15.97	70.49	16.10	60.86	24.90	73.71	5.34	15.89	52.67	65.18
SFT	20.28	63.85	27.59	65.65	10.02	63.50	19.02	68.19	27.93	56.31	39.21	70.25	15.57	24.01	-	64.63
R1	30.99	59.19	37.05	60.12	14.53	49.39	28.45	57.63	30.35	53.38	57.33	66.73	25.91	33.12	-	57.74
Search-R1	41.29	70.26	50.85	73.85	22.35	57.68	45.88	67.58	50.76	66.08	65.98	76.15	38.54	46.19	51.60	68.60
R1-Searcher	33.96	69.61	46.36	74.56	16.63	59.05	44.93	68.54	47.12	66.74	64.76	75.95	34.51	42.29	51.26	69.08
Graph-R1 (ours)	**65.04**	**82.42**	**62.69**	**80.03**	**46.17**	**71.42**	**49.87**	**70.97**	**51.22**	**73.43**	**71.93**	**79.11**	**48.57**	**57.82**	**60.40**	**76.23**

Thank you for listening

Sangjun Ji (지상준)

Department of Computer Science and Engineering
Konkuk University

📧Email: sangjunji02@gmail.com
🏠Homepage: https://approxy02.github.io/homepage/

2026. 07. 29.