GNN Advanced
-- GNN in NLP

Zhaoxuan Tan

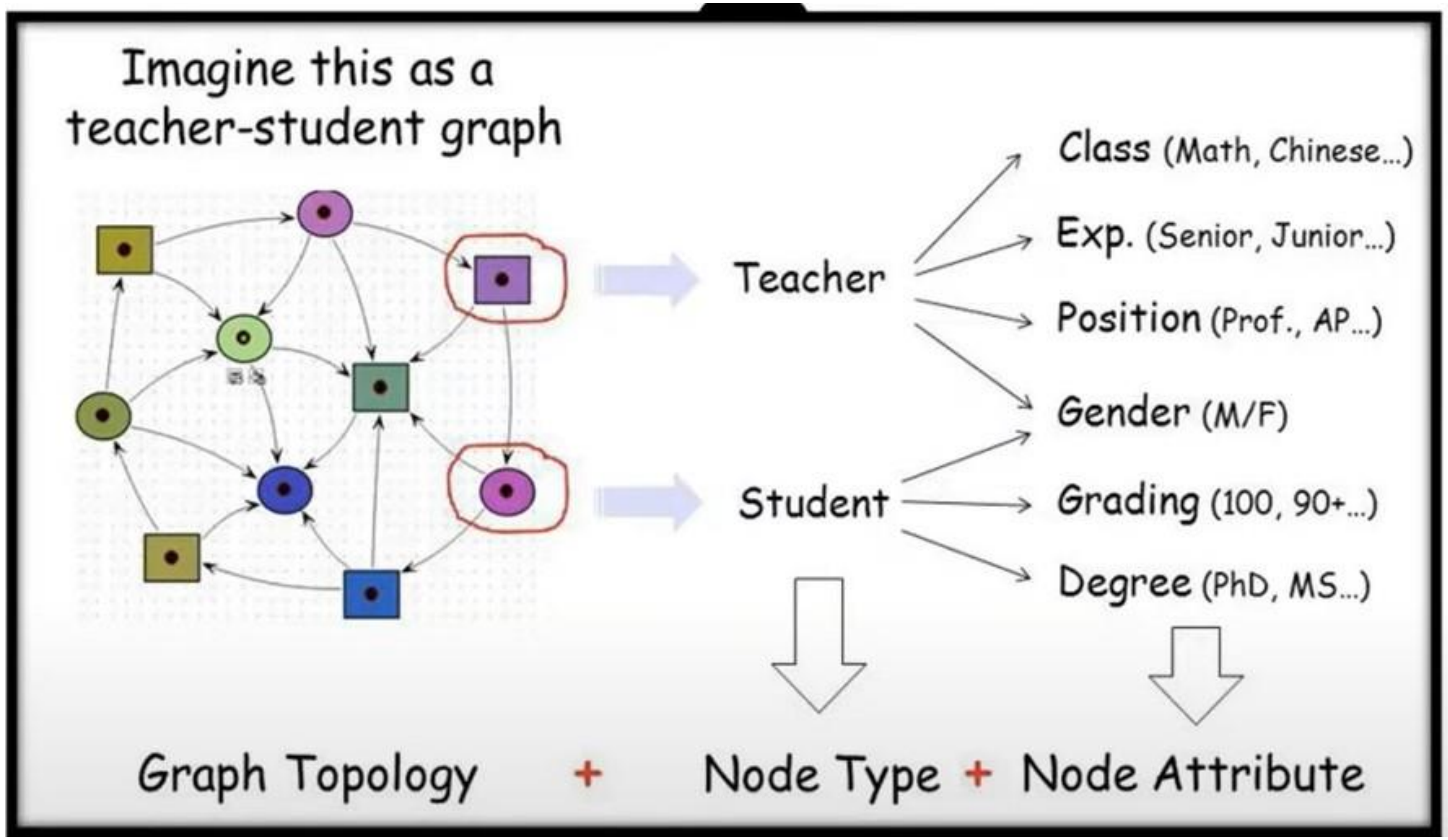
Xi'an Jiaotong University

March 5, 2022

Table of Contents

Why Graph?

- Graphs are a general language for describing and modeling complex system

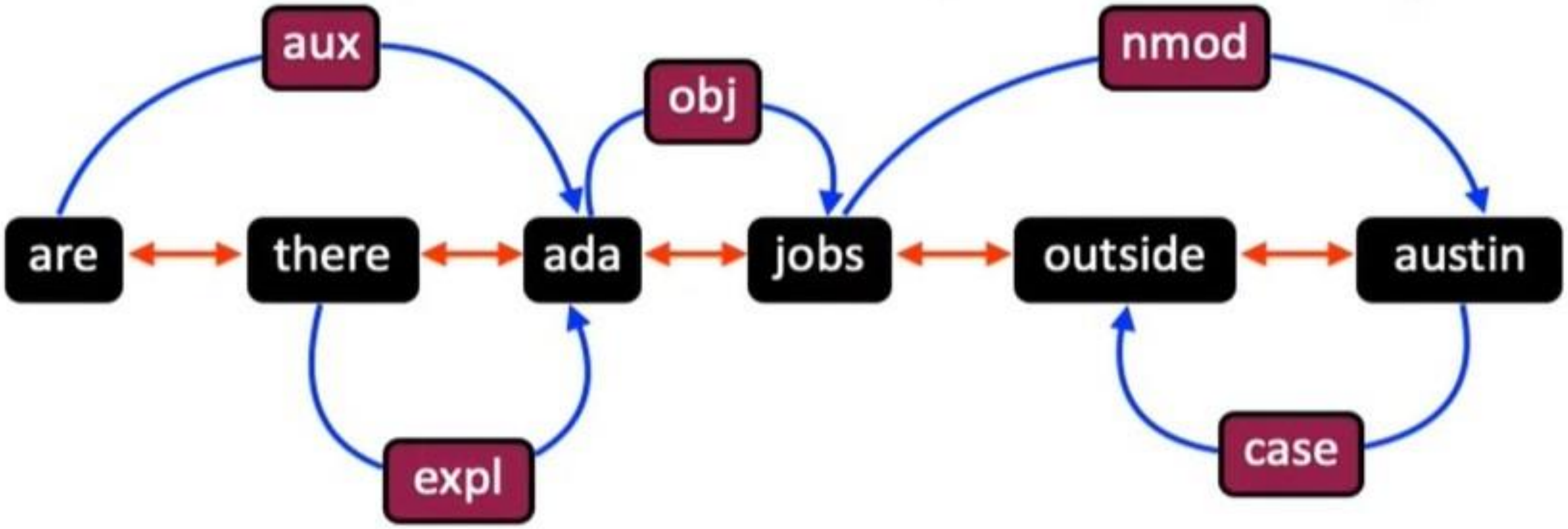


Graph Construction: Static Method

- Problem Setting:
  - Input: raw text (sentence, paragraph, document etc.)
  - Output: graph
- Conducting during preprocessing by augmenting text with domain knowledge

Static Method: Dependency Graph

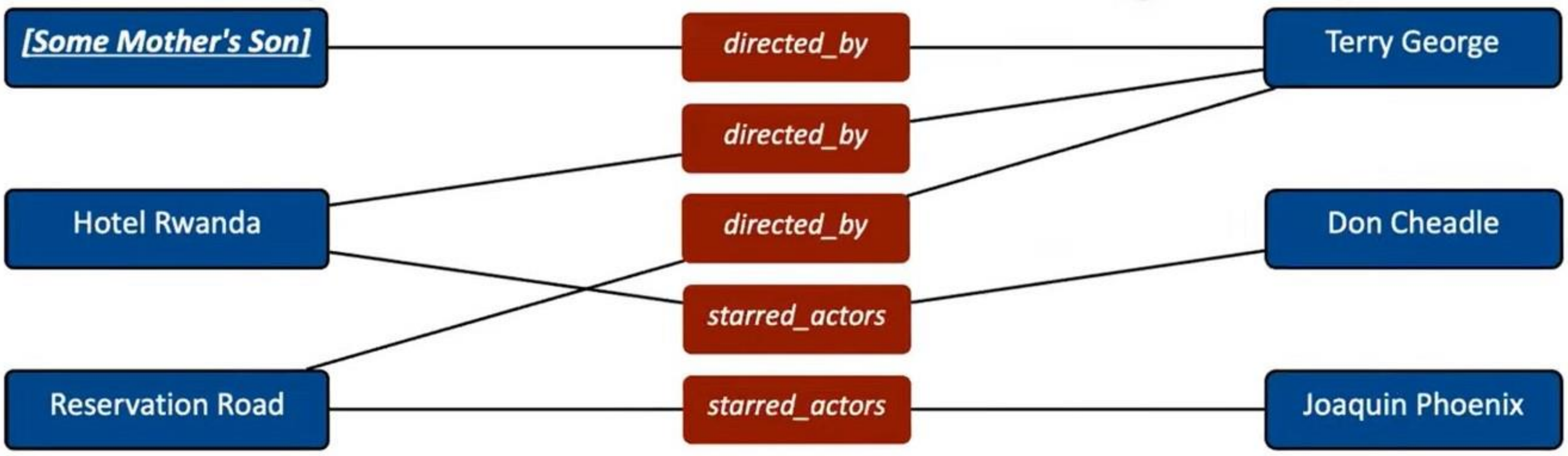
- Text: are there ada jobs outside Austin?



- Add sequential edge
  - reserve sequential information in raw text
  - Connection multiple dependency graphs in a paragraph

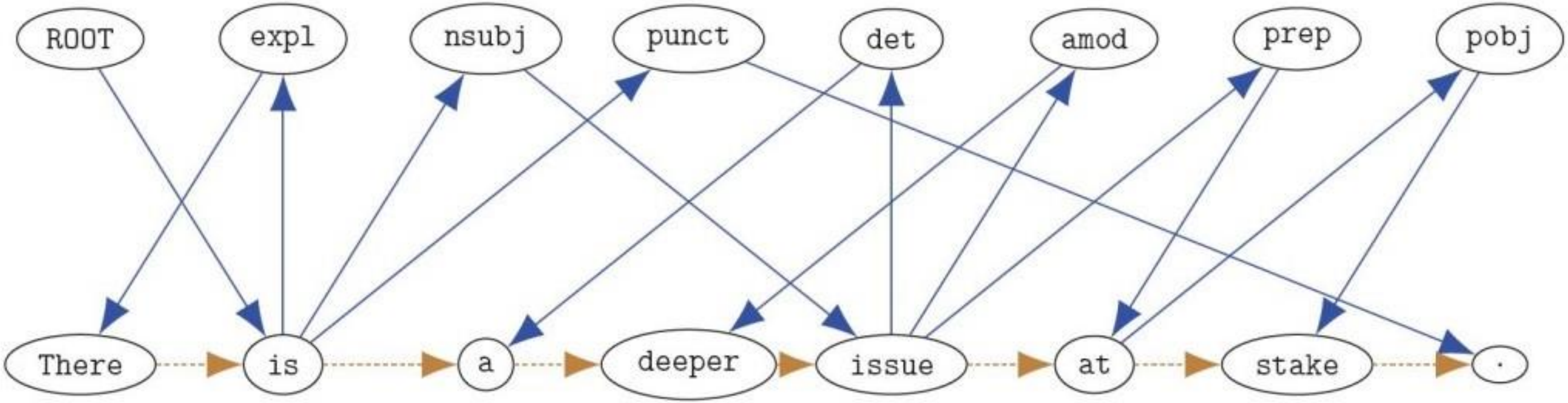
Static Method: Knowledge Graph

- Question: who acted in the movie directed by the director of [*Some Mother's Son*]
- Answer: Don Cheadle, Joaquin Phoenix

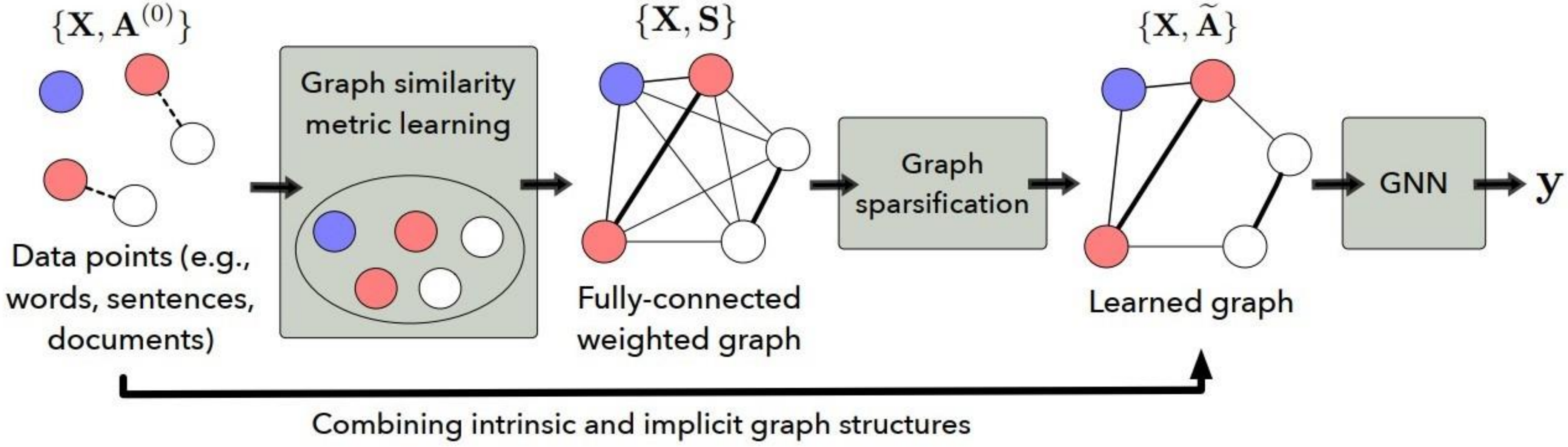


Static Method: Levi Graph

- An edge for every (node, edge)
- Edge label have hidden embedding



Graph Construction: Dynamic Method



Dynamic Method: Attention-based node similarity

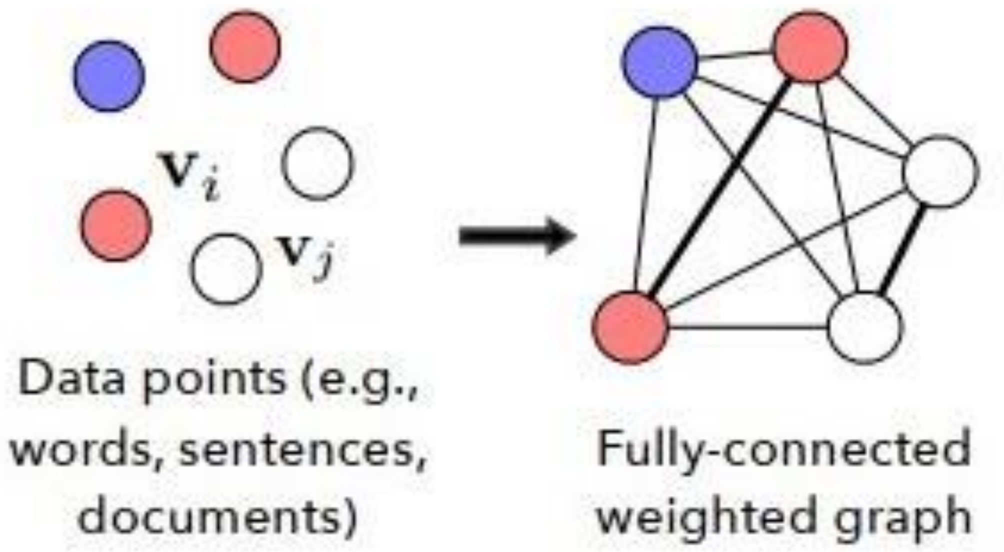
$$S_{i,j} = (\mathbf{v}_i \odot \mathbf{u})^T \mathbf{v}_j$$

Node feature vector

Non-negative learnable weight vector

$$S_{i,j} = \mathrm{ReLU}(\mathbf{W}\mathbf{v}_i)^T \mathrm{ReLU}(\mathbf{W}\mathbf{v}_j)$$

Learnable weight matrix



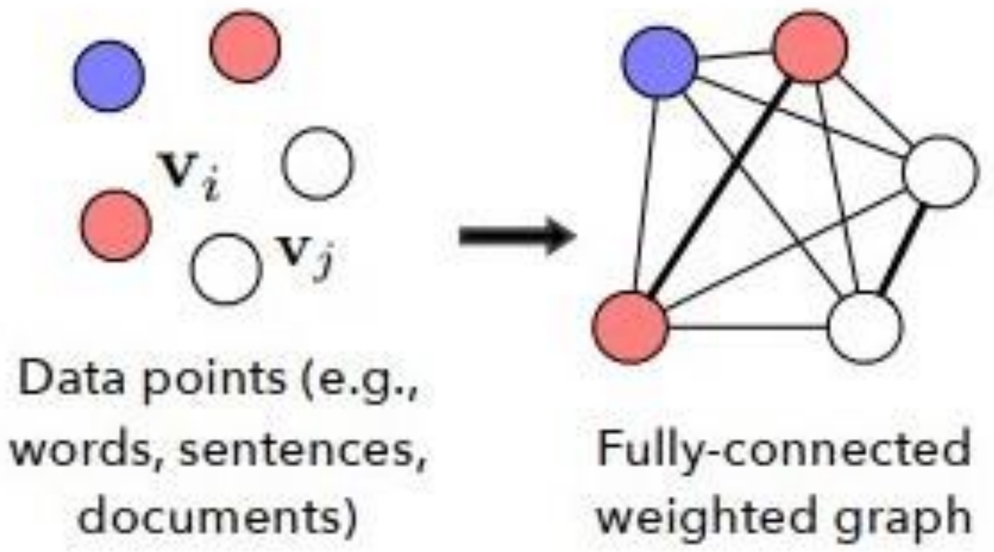
Dynamic Method: Cosine-based node similarity

$$S_{i,j}^{p} = \cos(\mathbf{w}_p \odot \mathbf{v}_i, \mathbf{w}_p \odot \mathbf{v}_j)$$

Learnable weight vector

$$S_{i,j} = \frac{1}{m}\sum_{p=1}^{m} S_{ij}^{p}$$

Multi-head similarity scores



Dynamic Method: Graph Sparsification

- KNN-style Sparsification

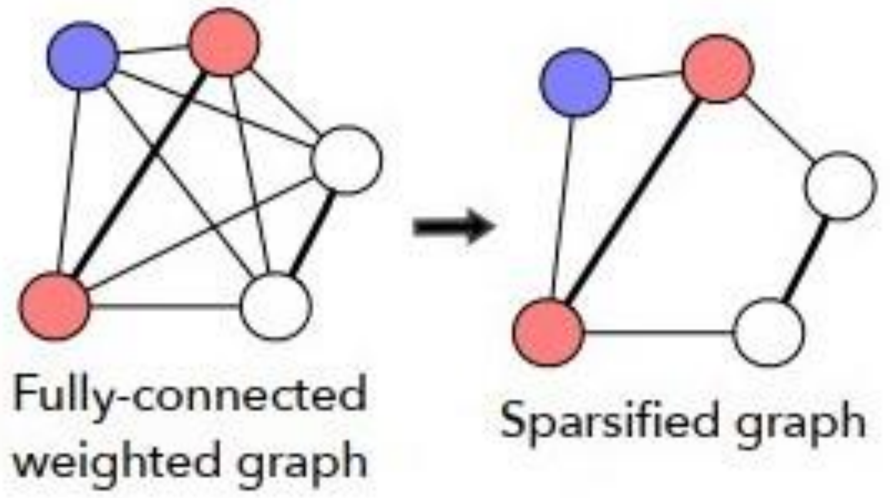
$$\mathbf{A}_{i,:} = \text{topk}(\mathbf{S}_{i,:})$$

- Epsilon-neighborhood Sparsification

$$A_{i,j} = \begin{cases} S_{i,j} & S_{i,j} > \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

- Graph Regularization

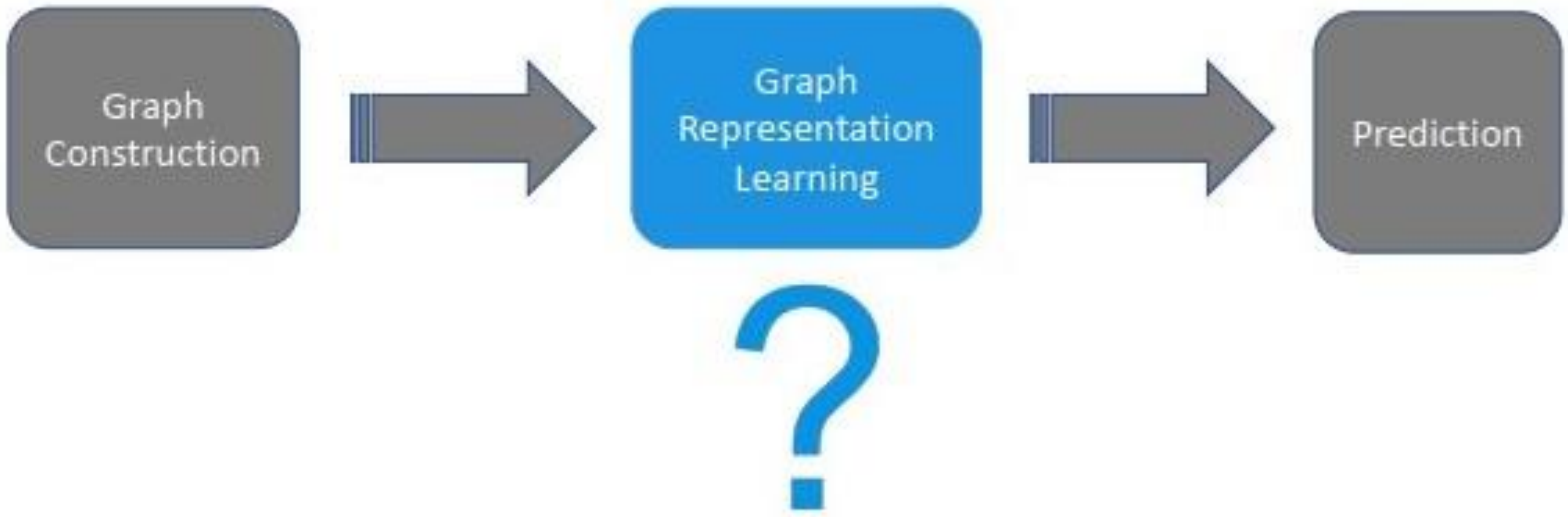
$$\frac{1}{n^2}||A||_F^2$$



Static vs. Dynamic Graph Construction

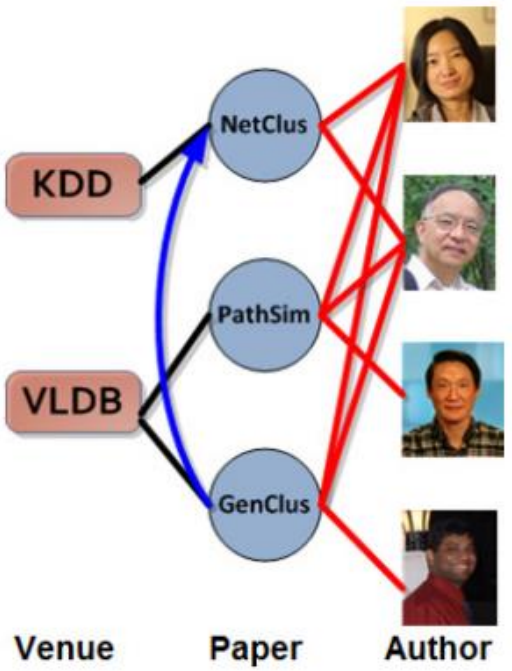
Static graph construction	Dynamic graph construction
Pros	Pros
prior knowledge	no domain expertise
	joint graph structure & representation learning
Cons	Cons
extensive domain expertise	scalability
• error-prone (e.g., noisy, incomplete) • sub-optimal	explainability
• disjoint graph structure & representation learning • error accumulation	

After constructing graph? -> Representation!

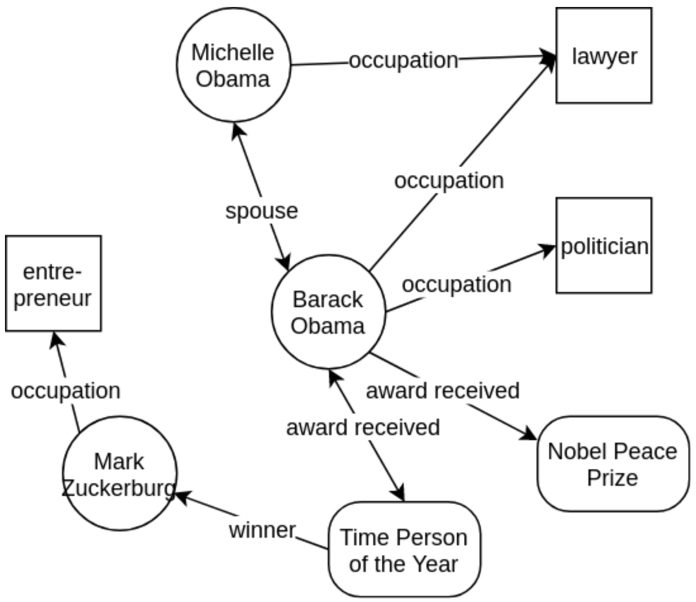


What is Heterogeneous GNN?

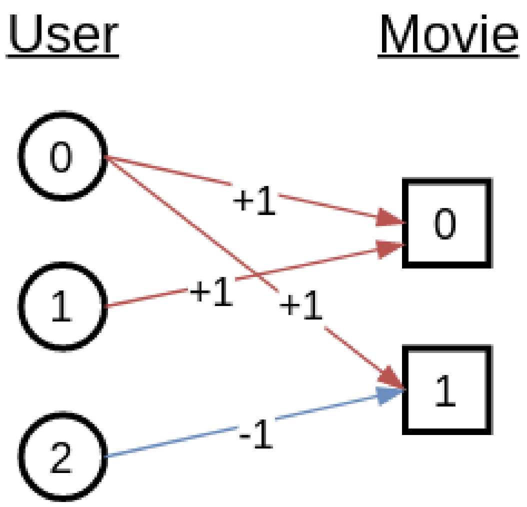
- Citation Graph



- Knowledge Graph



- Recommender System



R-GCN

- Message Passing Scheme

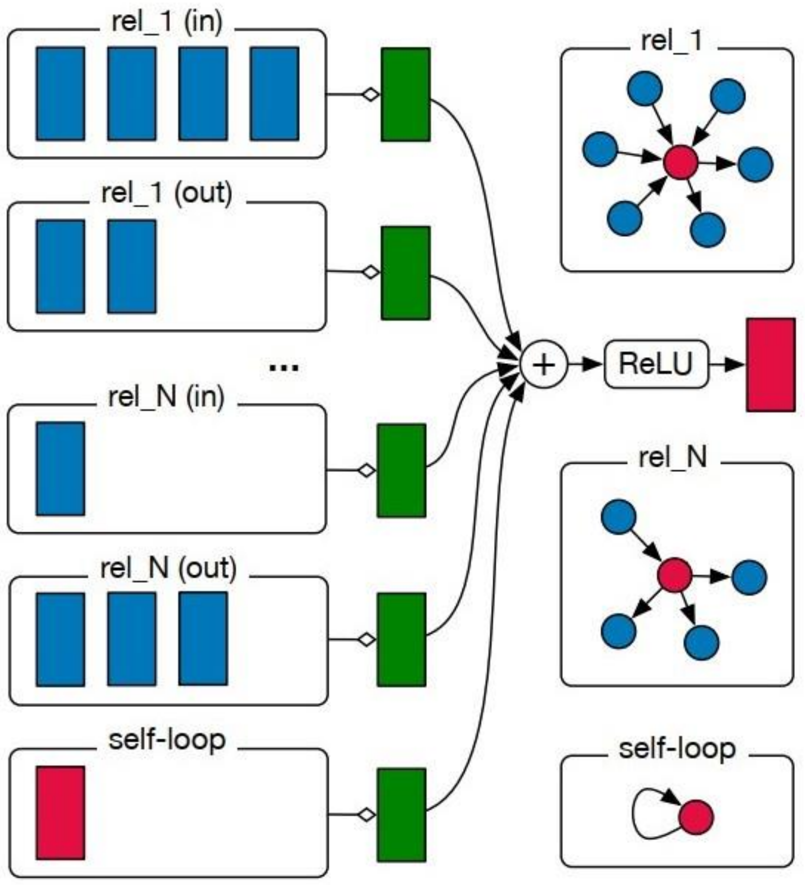
$$h_i^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l)} + W_0^{(l)}h_i^{(l)}\right)$$

- Regularization: avoid over-parameterization
  - Basis decomposition

$$W_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} V_b^{(l)}$$

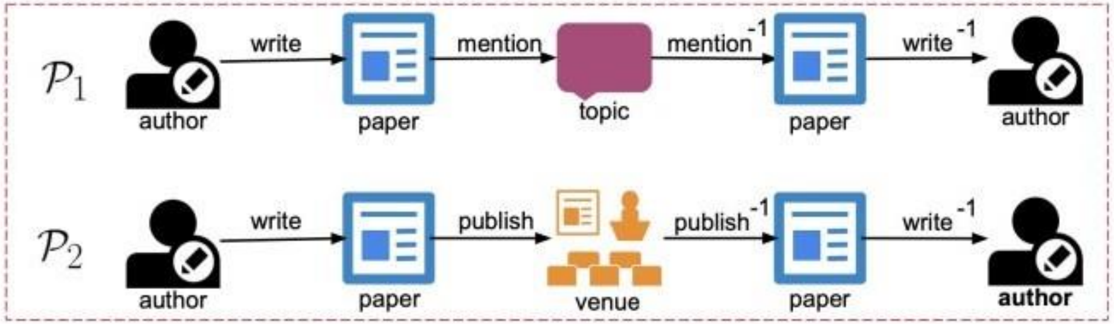
  - Block decomposition

$$W_r^{(l)} = \bigoplus_{b=1}^{B} Q_{br}^{(l)}.$$

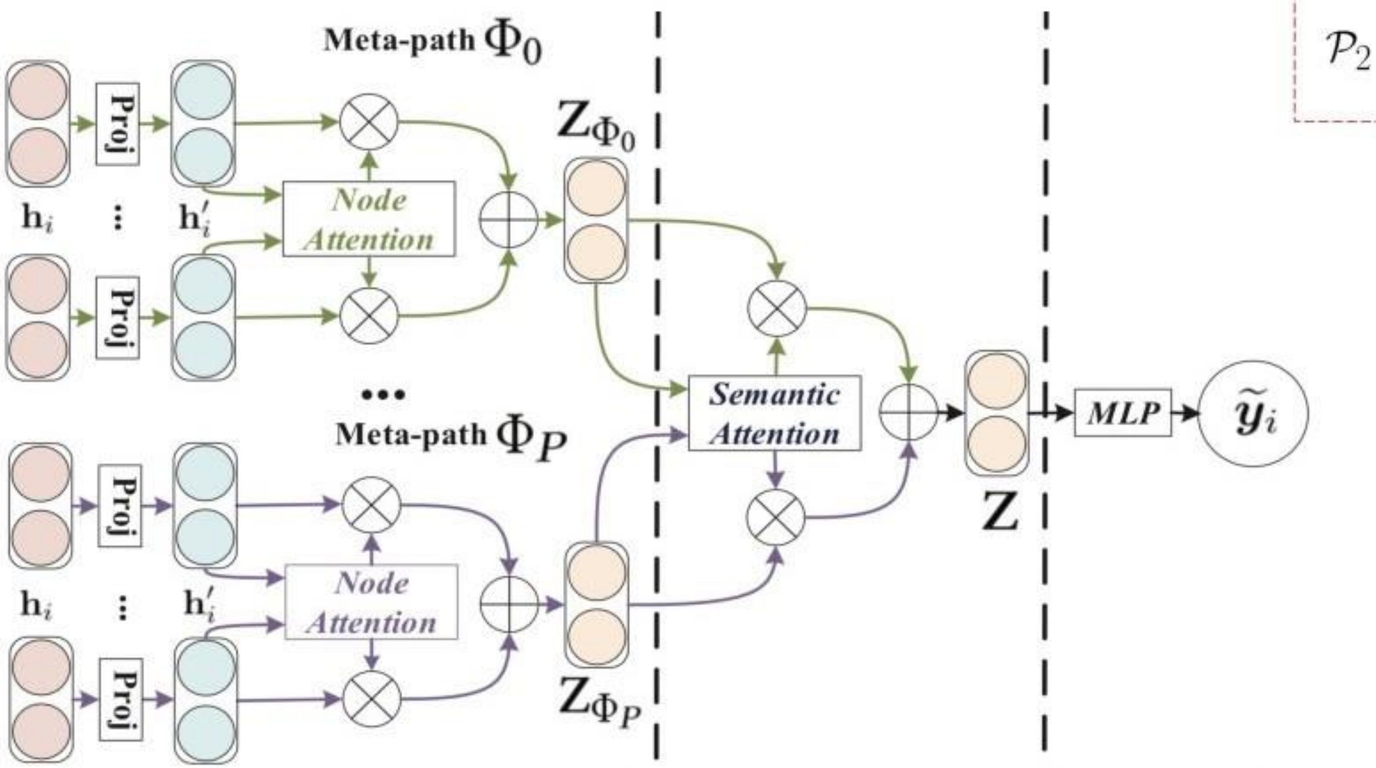


HAN

- Meta path



Meta paths among author nodes



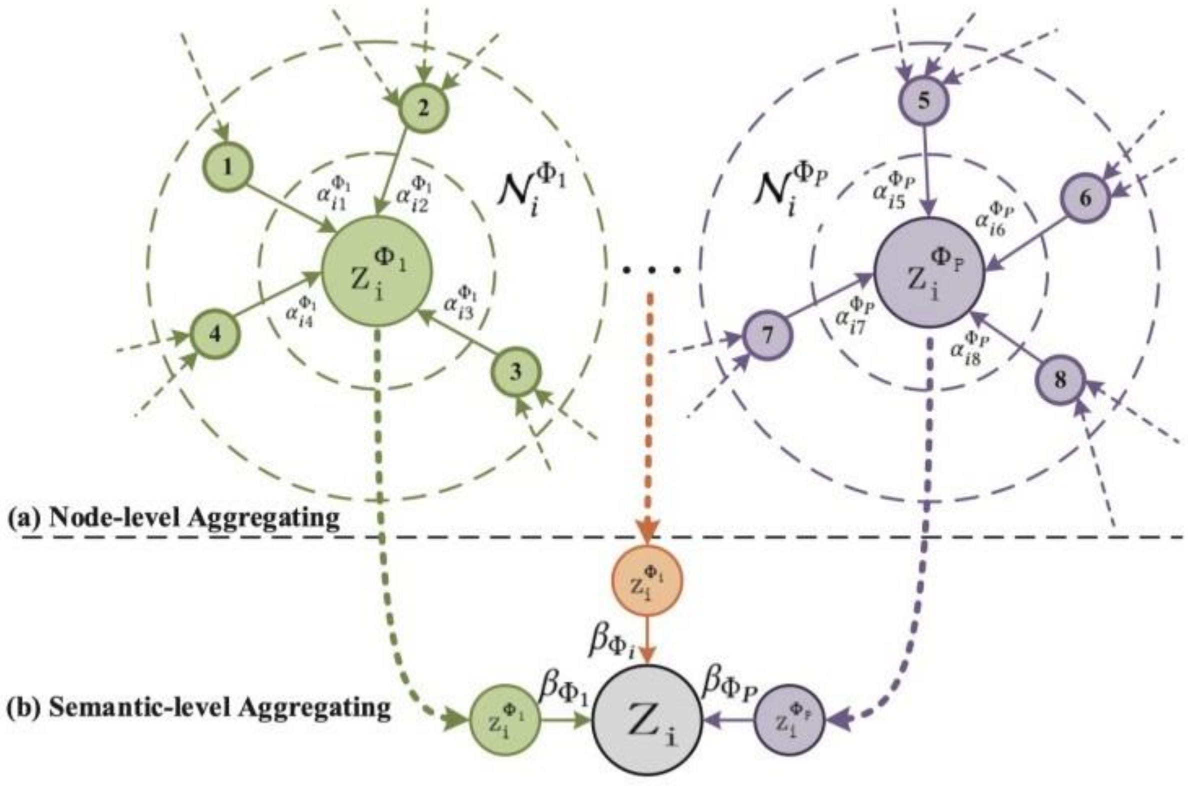
(a) Node-Level Attention (b) Semantic-Level Attention (c) Prediction

- Node Level Attention

$$\alpha_{ij}^{\Phi} = softmax_j(e_{ij}^{\Phi}) = \frac{\exp\left(\sigma(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot [\mathbf{h}'_i \| \mathbf{h}'_j])\right)}{\sum_{k \in \mathcal{N}_i^{\Phi}} \exp\left(\sigma(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot [\mathbf{h}'_i \| \mathbf{h}'_k])\right)}$$

$$\mathbf{z}_i^{\Phi} = \Big\|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}'_j\right).$$

HAN



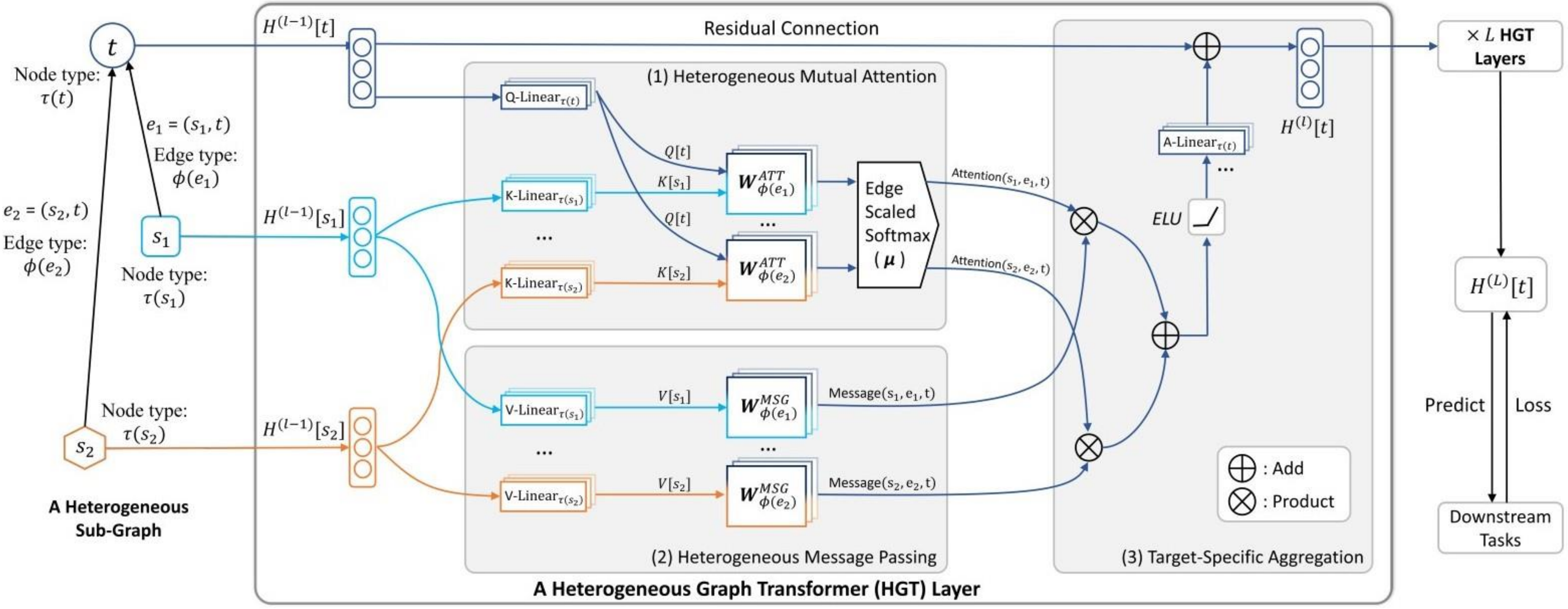
- Semantic Level Attention

$$w_{\Phi_p} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{q}^{\mathrm{T}} \cdot \tanh(\mathbf{W} \cdot \mathbf{z}_i^{\Phi_p} + \mathbf{b})$$

$$\beta_{\Phi_p} = \frac{\exp(w_{\Phi_p})}{\sum_{p=1}^{P} \exp(w_{\Phi_p})}$$

$$\mathbf{Z} = \sum_{p=1}^{P} \beta_{\Phi_p} \cdot \mathbf{Z}_{\Phi_p}$$

HGT



$$H^l[t] \leftarrow \underset{\forall s\in N(t), \forall e \in E(s,t)}{\textbf{Aggregate}}\left(\textbf{Attention}(s,t)\cdot\textbf{Message}(s)\right)$$

$$\textbf{Attention}_{HGT}(s,e,t) = \underset{\forall s\in N(t)}{\text{Softmax}}\left(\underset{i\in[1,h]}{\Big\|} ATT\text{-}head^i(s,e,t)\right) \qquad (3)$$

$$ATT\text{-}head^i(s,e,t) = \left(K^i(s)\ W^{ATT}_{\phi(e)}\ Q^i(t)^T\right)\cdot\frac{\mu_{\langle \tau(s),\phi(e),\tau(t)\rangle}}{\sqrt{d}}$$

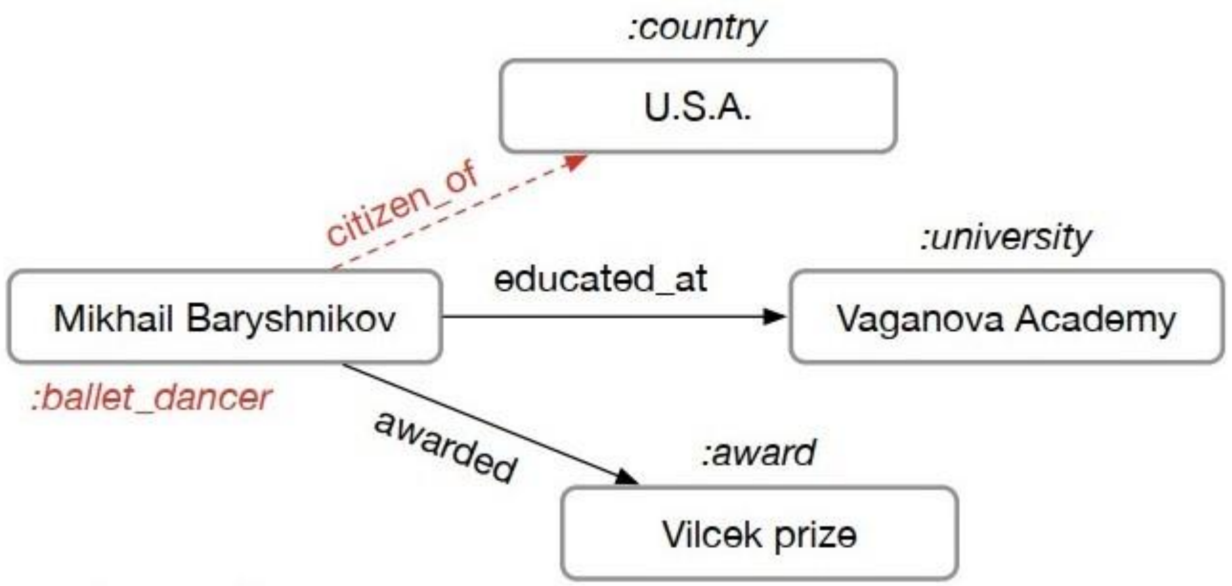
$$K^i(s) = \text{K-Linear}^i_{\tau(s)}\left(H^{(l-1)}[s]\right)$$

$$Q^i(t) = \text{Q-Linear}^i_{\tau(t)}\left(H^{(l-1)}[t]\right)$$

$$\textbf{Message}_{HGT}(s,e,t) = \underset{i\in[1,h]}{\Big\|} MSG\text{-}head^i(s,e,t)$$

$$MSG\text{-}head^i(s,e,t) = \text{M-Linear}^i_{\tau(s)}\left(H^{(l-1)}[s]\right)\ W^{MSG}_{\phi(e)}$$

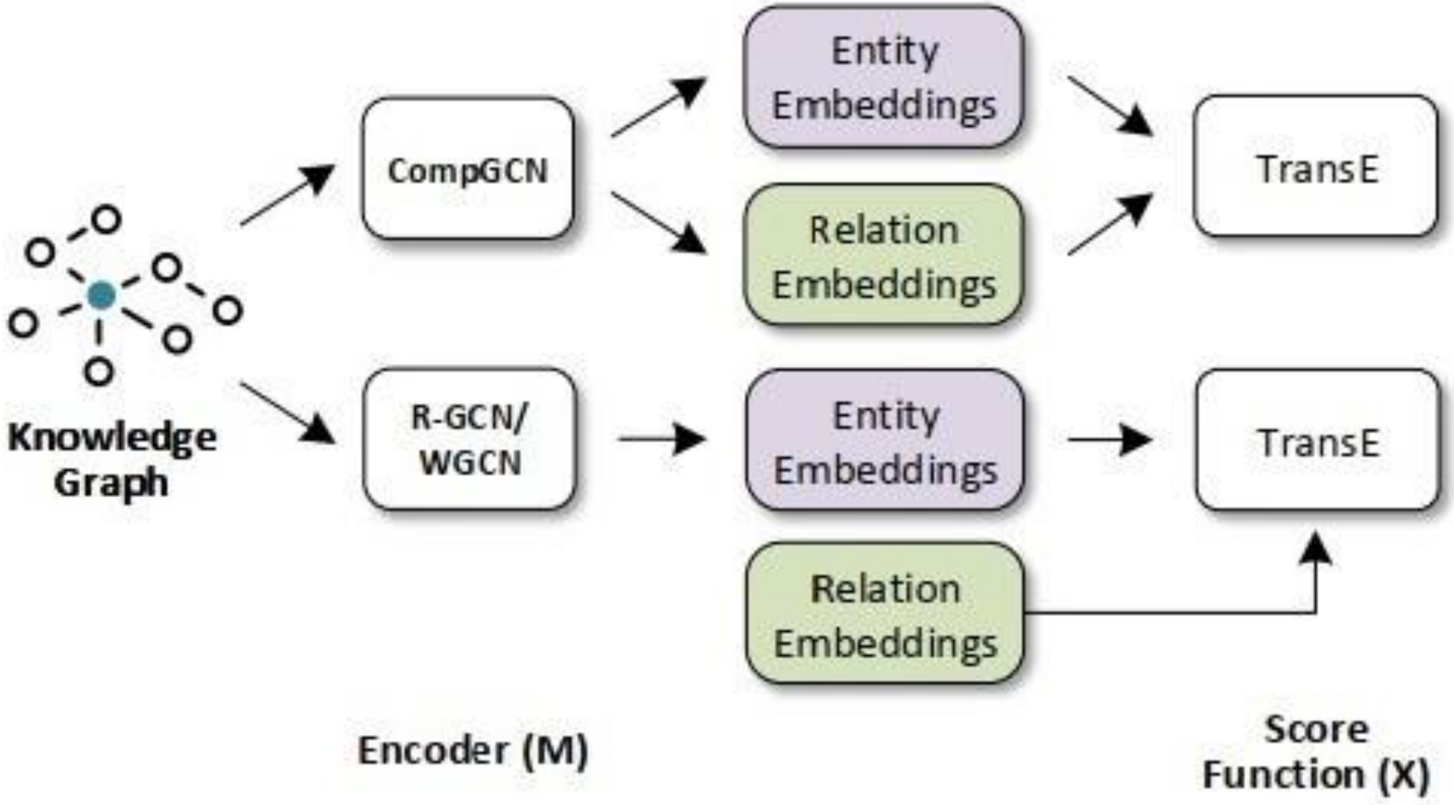
GNN for KG Embedding



$$h_s = \sigma\left(\sum_{r\in\mathcal{R}} \sum_{o\in\mathcal{N}_r(s)} f\left(s,r,o\right)\right)$$

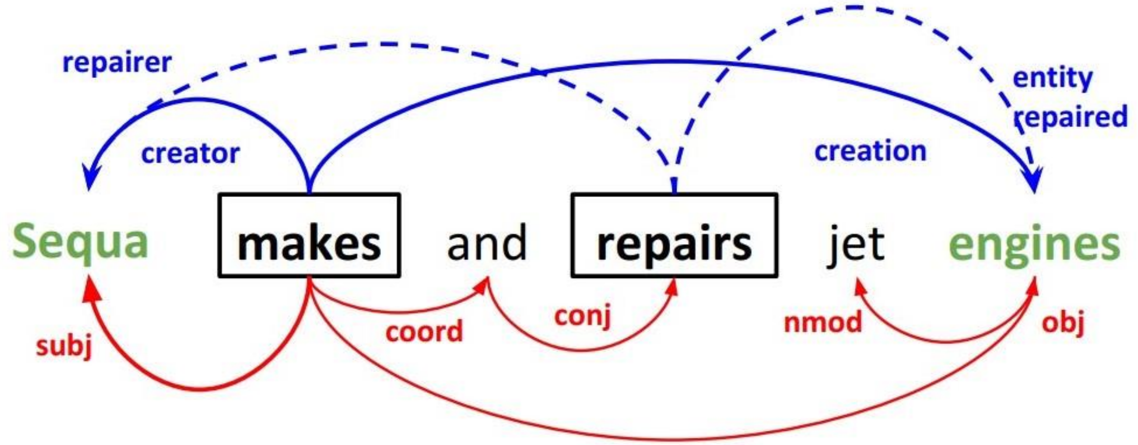
Method	$f\left(s,r,o\right)$
R-GCN	$W_r h_o$
SACN	$\alpha_r W h_o$
KBGAT	$\alpha_{sro} W h_{sro}$
CompGCN	$W_{\lambda(r)}\phi(s,r)$

GNN for KG Embedding: Encoder-Decoder for LP



Application: Syntactic GCN for SRL

- Task: discover who did what to whom.
- Syntax mirrors semantics
- Exploit syntax using convolution



$$h_v = ReLU\left(\sum_{u \in \mathcal{N}(v)} g_{u,v}\left(W_{d(u,v)}\, h_u + b_{l(u,v)} \right)\right)$$

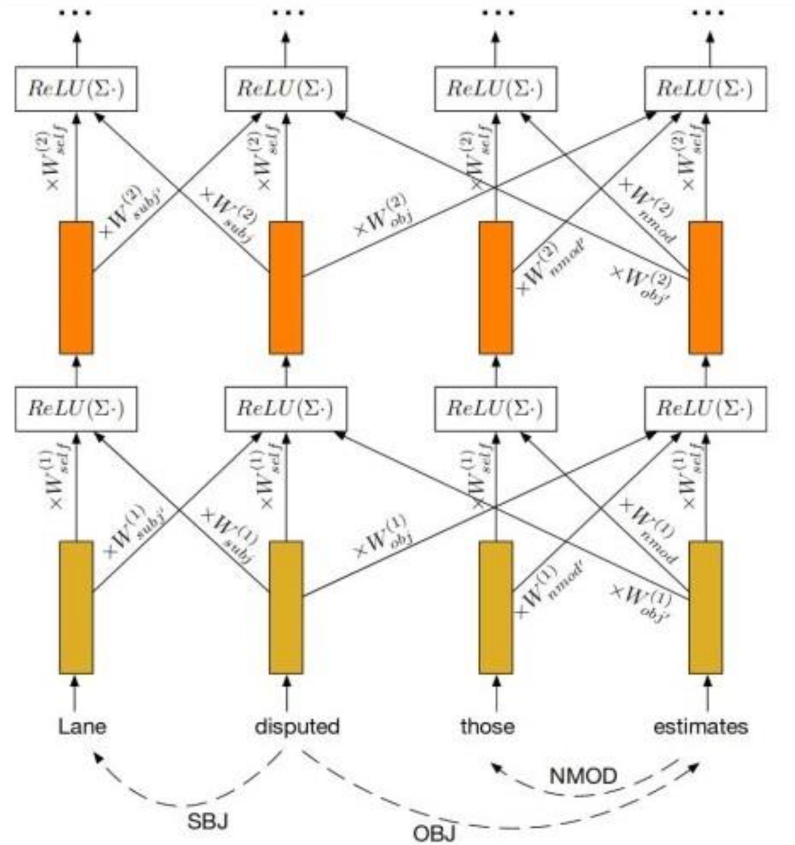
word emb of v

weight of direction

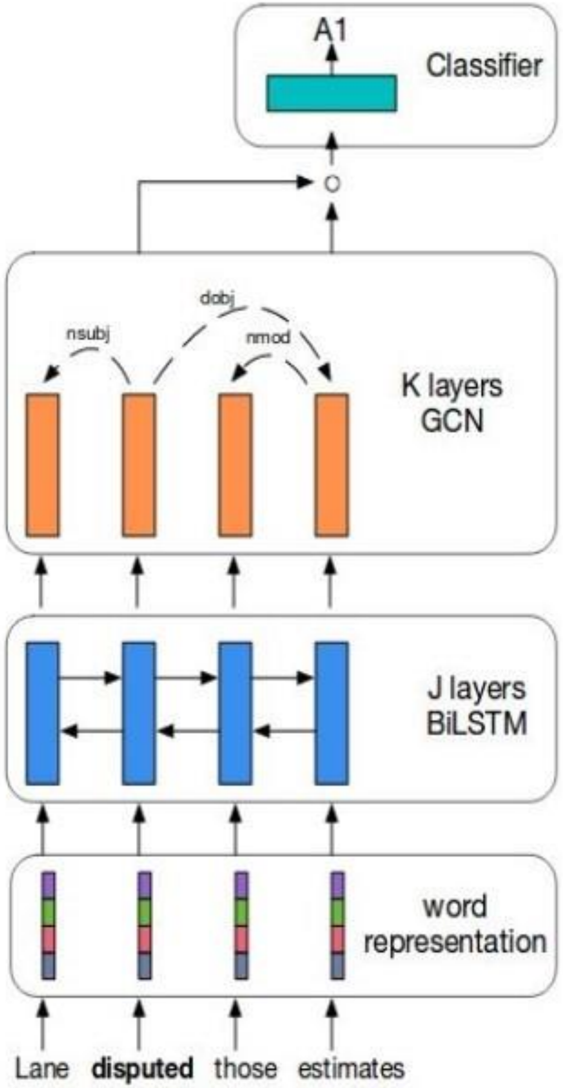
bias of label + direction

edge-wise gating

$$g_{u,v} = \sigma\left(\hat{w}_{d(u,v)}\, h_u + \hat{b}_{l(u,v)} \right)$$



Application: Syntactic GCN for SRL



- Trained with cross-entropy loss

F1 on CoNLL-2009

BiLSTM	82.7
BiLSTM + GCN	**83.3**

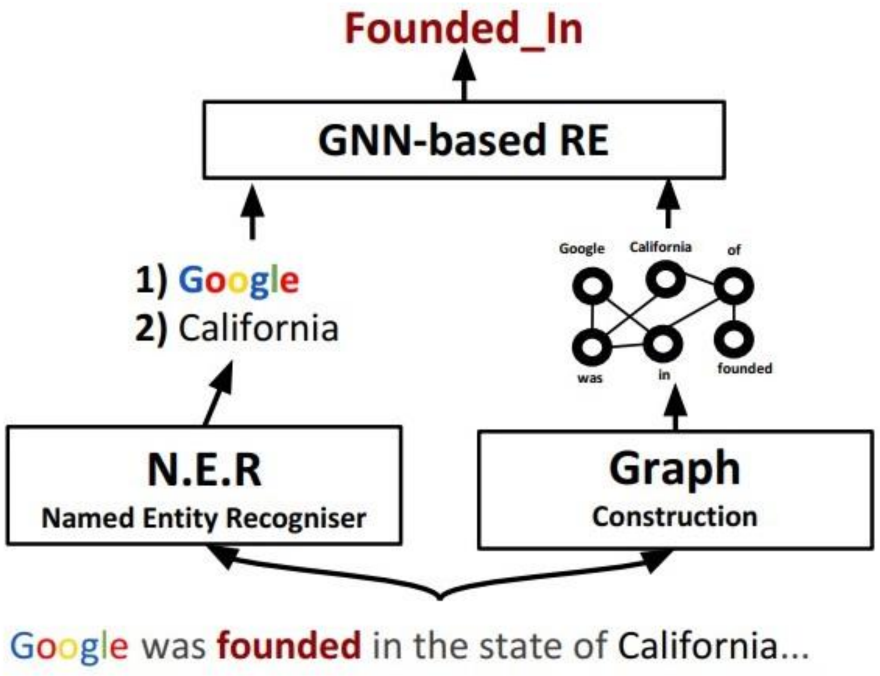
- GCN integrates syntax, context
- GCN, LSTM complement each other

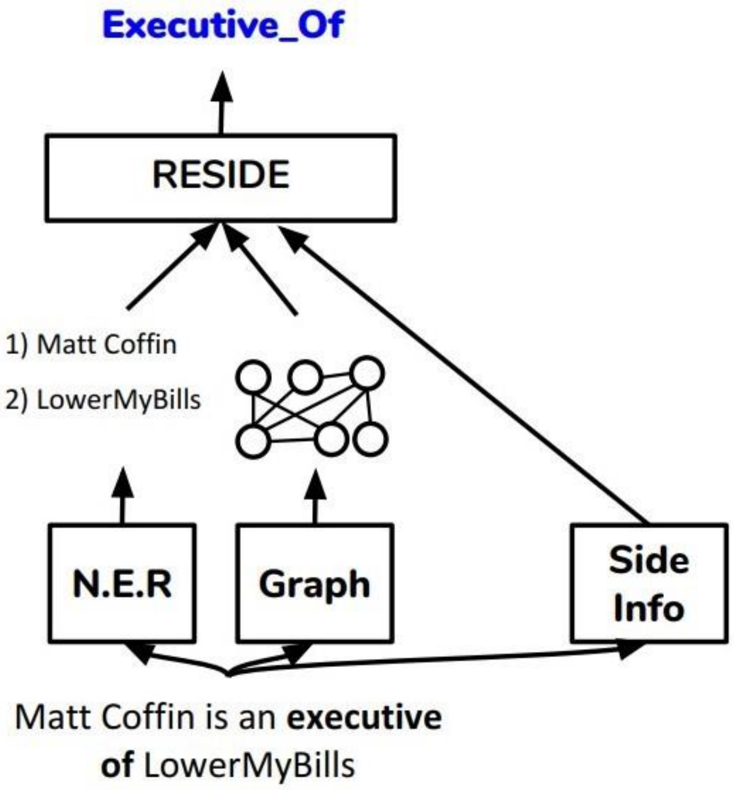
Application: GNNs for Relation Extraction

- Identify relation between entities

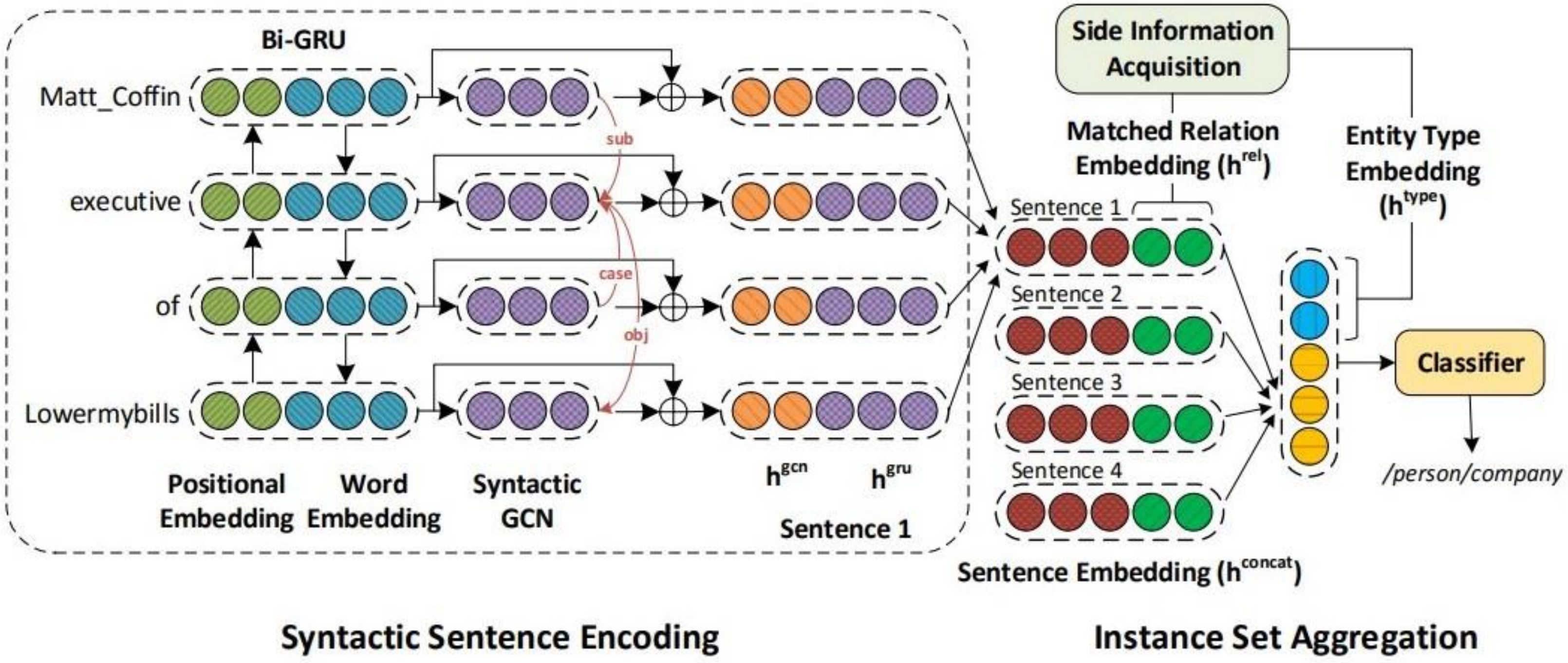
Google was **founded** in California in 1998.

- **Founding-year** (Google, 1998)
- **Founding-location** (Google, California)





GNNs for Relation Extraction: RE-SIDE



- Side information Acquisition
  - Relation Alias side information

    Open IE, PPDB -> GloVe Embeddings -> find the closest relation for each phase (h^{rel})

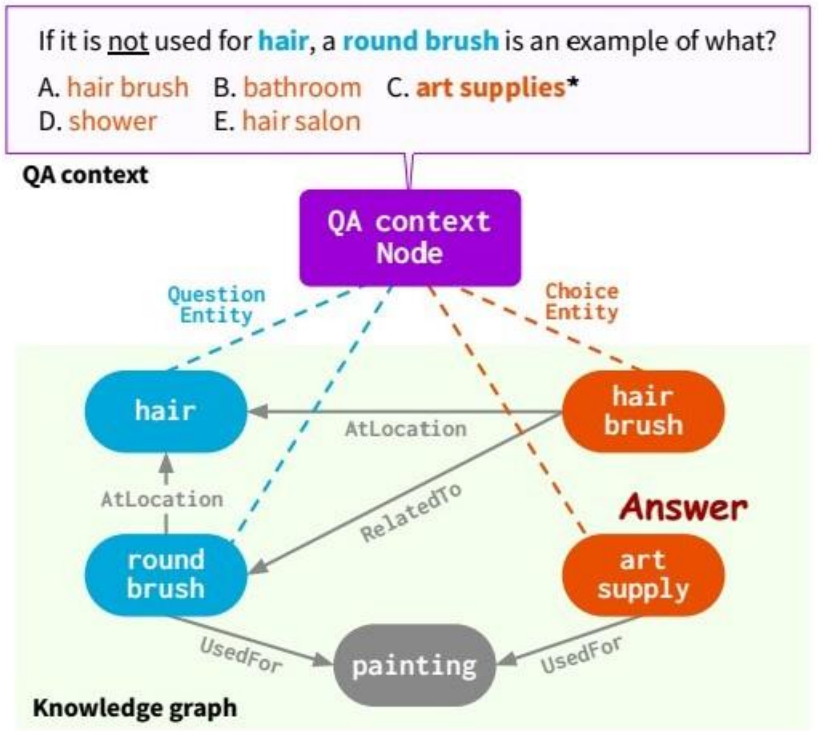
  - Entity Type side information

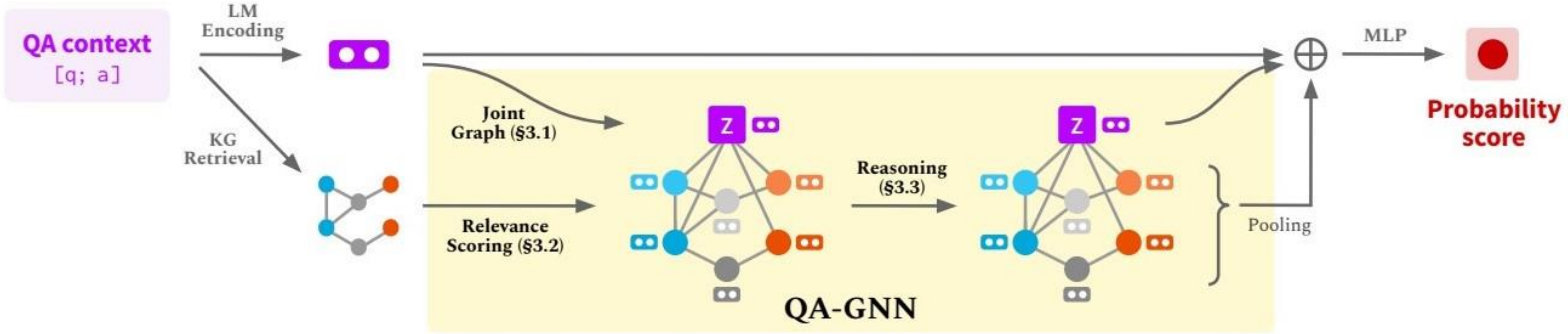
    E.g. Paris: government, location -> average type embeddings (h^{type})

- Side info improves performance!

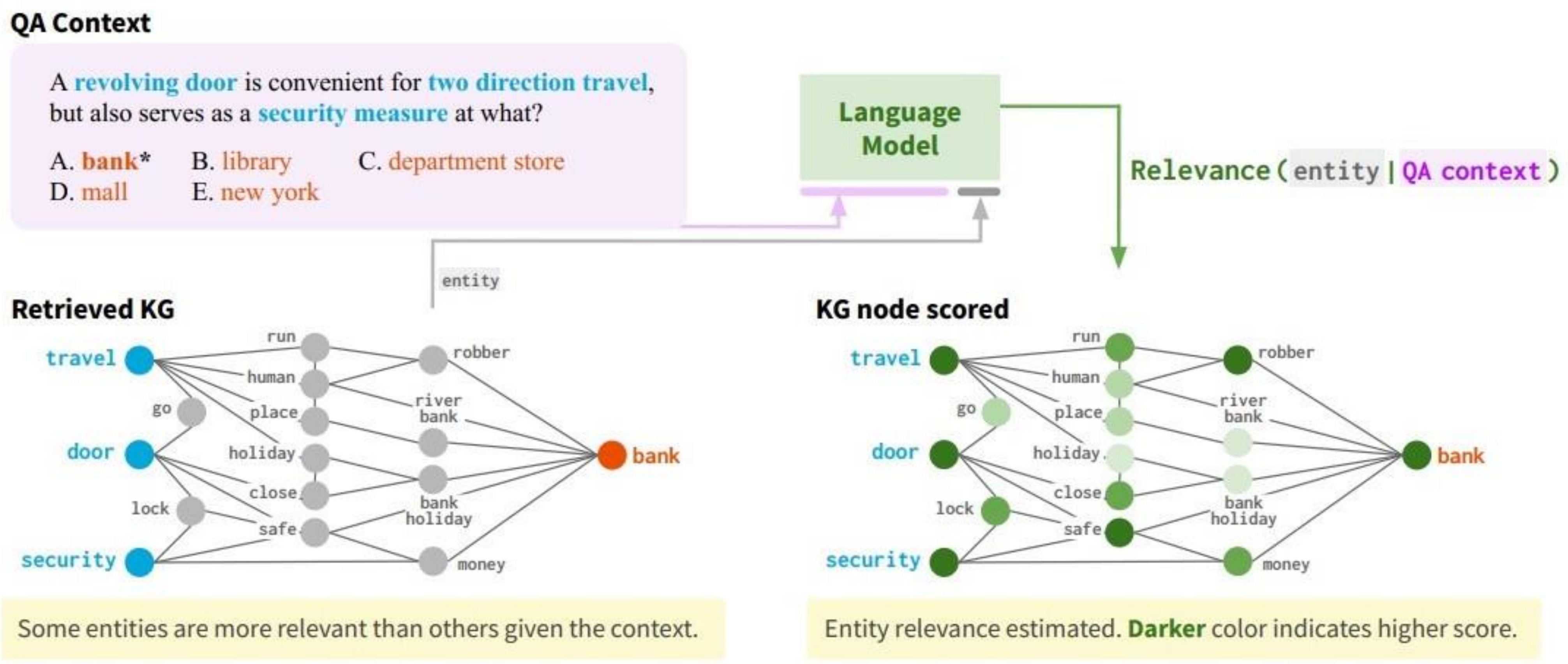
Application: QA-GNN

- Identify relevant knowledge from KG -> **relevance scoring**
- Performa **joint reasoning** over the QA context and KG



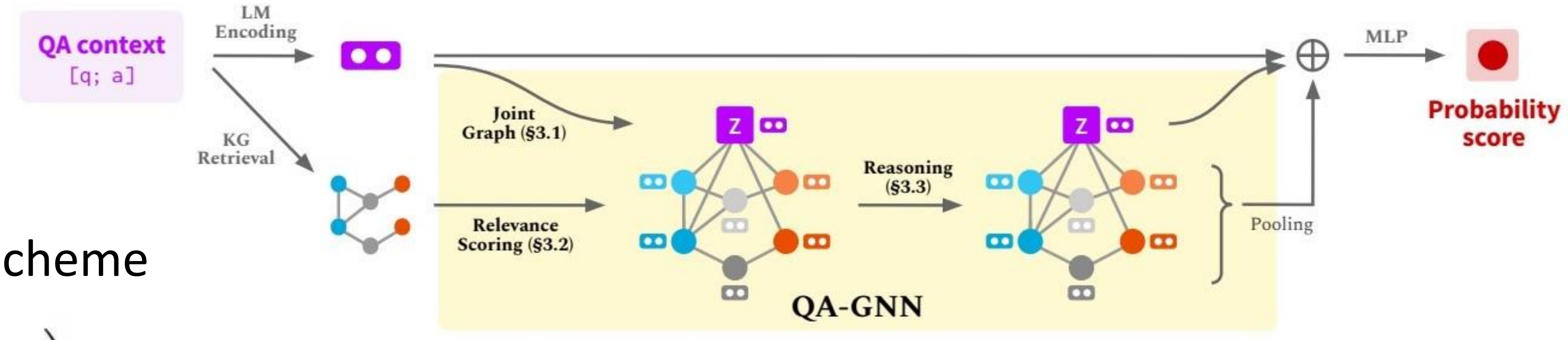


QA-GNN: relevance scoring



$$\rho_v = f_{\text{head}}(f_{\text{enc}}([\texttt{text}(z); \texttt{text}(v)])),$$

QA-GNN: GNN architecture



- Message passing scheme

$$h_t^{(\ell+1)} = f_n\left(\sum_{s\in\mathcal{N}_t\cup\{t\}} \alpha_{st} \boldsymbol{m}_{st}\right) + \boldsymbol{h}_t^{(\ell)}$$

- Message

$$\boldsymbol{m}_{st} = f_m(\boldsymbol{h}_s^{(\ell)}, \boldsymbol{u}_s, \boldsymbol{r}_{st})$$

$$\boldsymbol{u}_t = f_u(\mathrm{u}_t), \quad \boldsymbol{r}_{st} = f_r(\mathrm{e}_{st}, \mathrm{u}_s, \mathrm{u}_t)$$

- Attention weight

$$\boldsymbol{\rho}_t = f_\rho(\rho_t)$$

$$\boldsymbol{q}_s = f_q(\boldsymbol{h}_s^{(\ell)}, \boldsymbol{u}_s, \boldsymbol{\rho}_s),$$

$$\boldsymbol{k}_t = f_k(\boldsymbol{h}_t^{(\ell)}, \boldsymbol{u}_t, \boldsymbol{\rho}_t, \boldsymbol{r}_{st})$$

$$\alpha_{st} = \frac{\exp(\gamma_{st})}{\sum_{t'\in\mathcal{N}_s\cup\{s\}} \exp(\gamma_{st'})}, \quad \gamma_{st} = \frac{\boldsymbol{q}_s^\top \boldsymbol{k}_t}{\sqrt{D}}$$

Thx for Attention

Zhaoxuan Tan

Xi’an Jiaotong University

March 5, 2022