GCNs for Relation Extraction

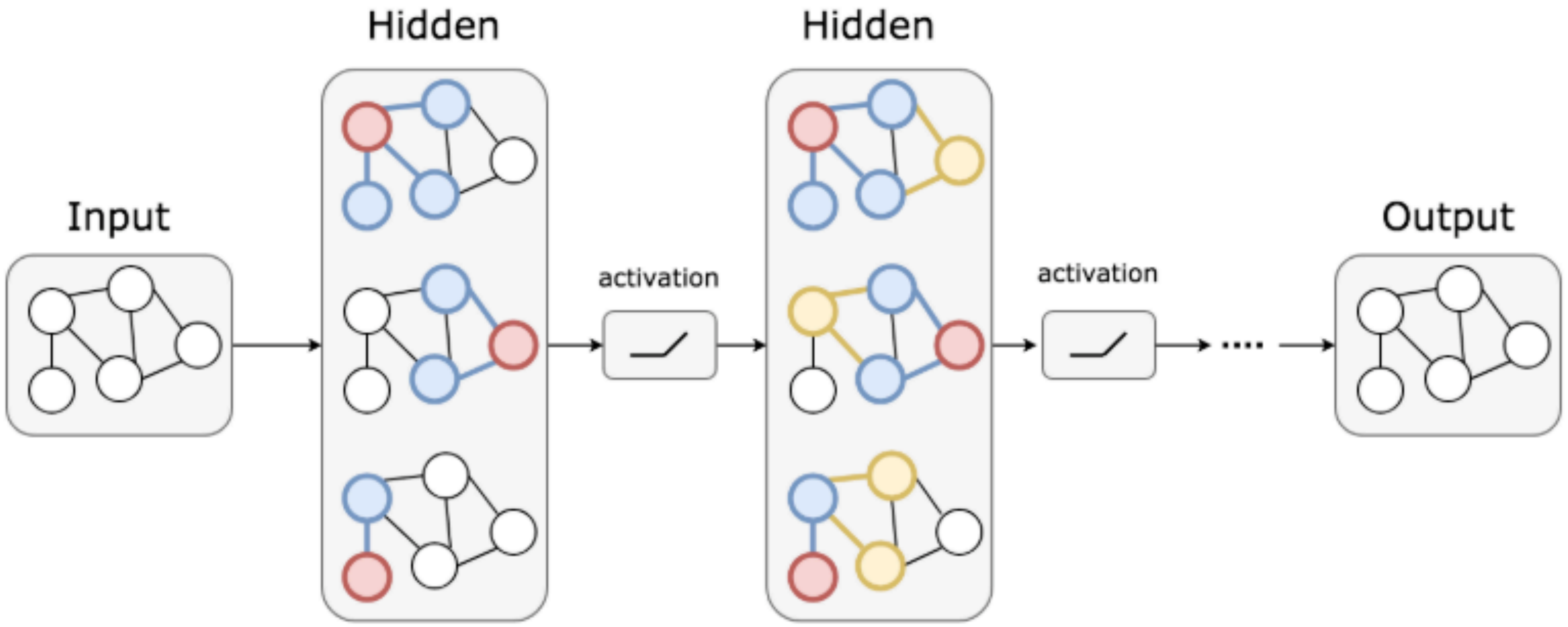
杨晰
xyang41@stu.ecnu.edu.cn

Outline

- Background
  - Graph Convolution Networks (GCN)
  - Relation Extraction (RE)
- Applications
  - Syntactic GCNs
  - RE-SIDE
  - GraphRel
- Conclusion

BGI: Graph Convolution Networks(GCN)

- 无向图G = (V, E)
  - V：节点集合
  - E：边集合

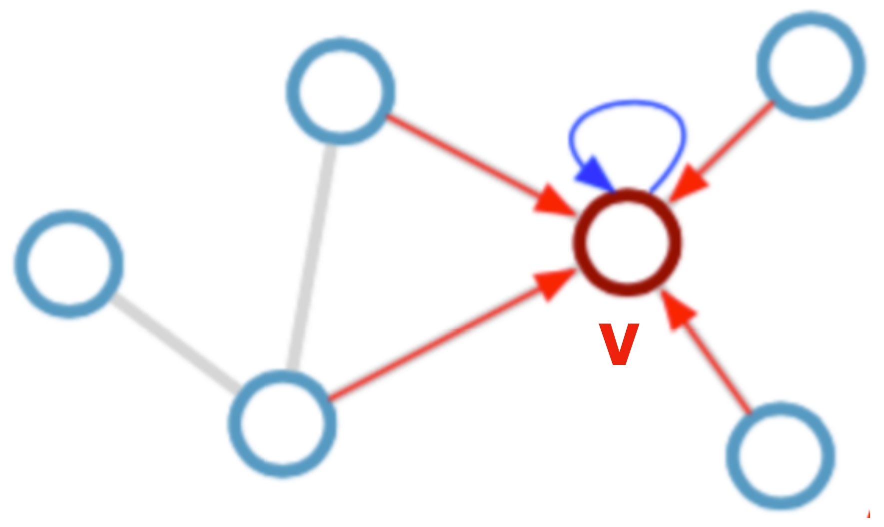


[ICLR2017]Semi-Supervised Classification with Graph Convolutional Networks

BGI: Graph Convolution Networks(GCN)

- 单层GCN

$$h_v = ReLU\left(\sum_{u \in \mathcal{N}(v)} (Wx_u + b)\right)$$



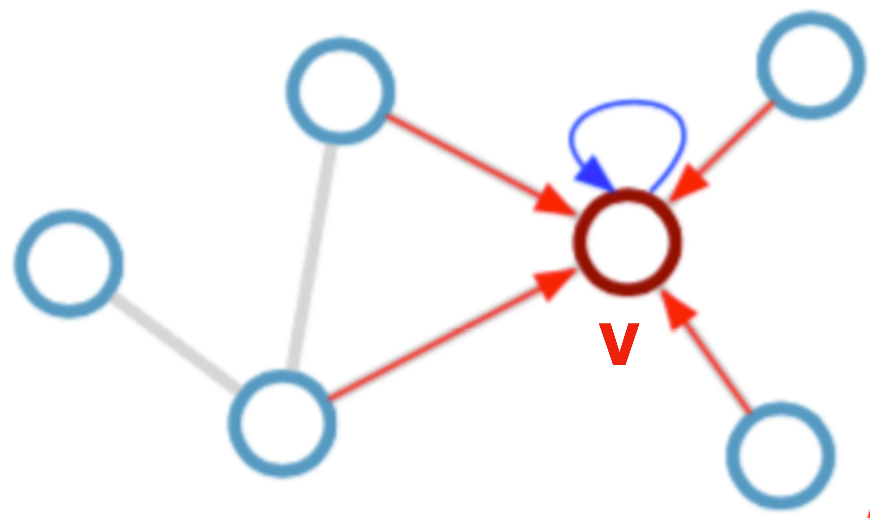
[ICLR2017]Semi-Supervised Classification with Graph Convolutional Networks

BGI: Graph Convolution Networks(GCN)

- 单层GCN

$$h_v = ReLU\left(\sum_{u\in\mathcal{N}(v)} (W \boxed{x_u} + b)\right)$$

初始节点特征



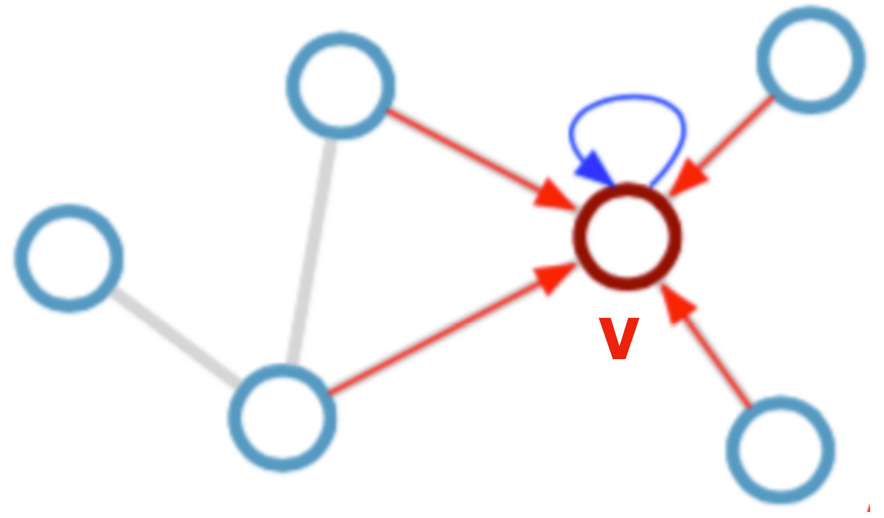
BGI: Graph Convolution Networks(GCN)

- 单层GCN

$$h_v = ReLU\left(\sum_{u \in \mathcal{N}(v)} (\boxed{W}\boxed{x_u} + \boxed{b})\right)$$

待学习参数

初始节点特征



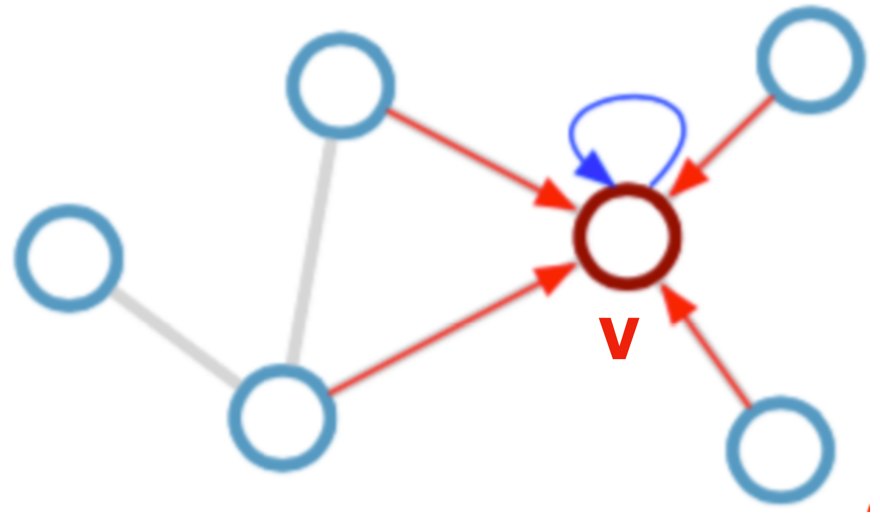
BGI: Graph Convolution Networks(GCN)

- 单层GCN

$$h_v = ReLU\left(\sum_{u\in\mathcal{N}(v)} \left(\boxed{W}\boxed{x_u}+\boxed{b}\right)\right)$$

待学习参数　　初始节点特征

节点v的邻居节点集合



BGI: Graph Convolution Networks(GCN)

- 单层GCN

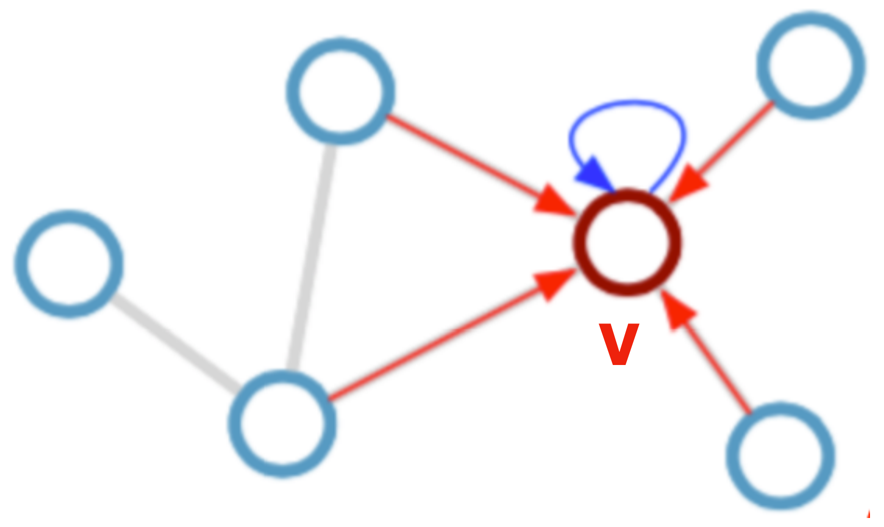
$$h_v = \boxed{ReLU}\left(\sum_{u\in\mathcal{N}(v)} (\boxed{W}\boxed{x_u}+\boxed{b})\right)$$

激活函数

待学习参数

初始节点特征

节点v的邻居节点集合



BGI: Graph Convolution Networks(GCN)

- 单层GCN

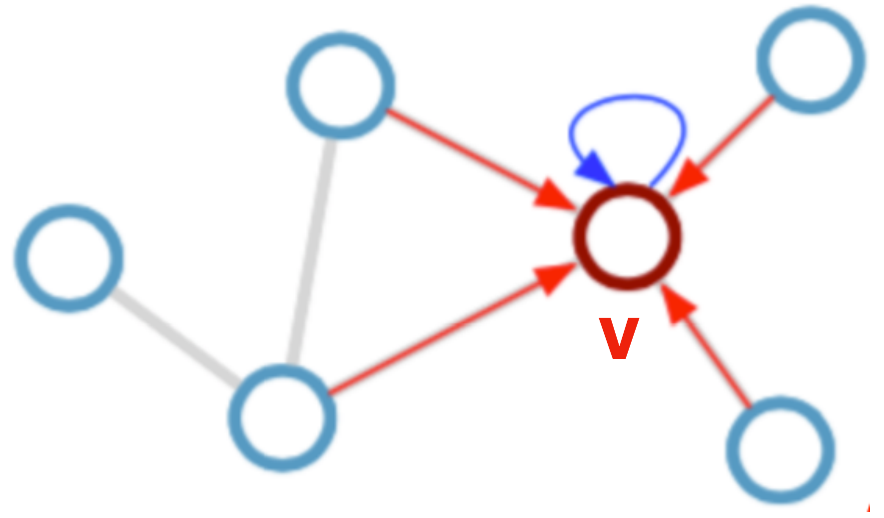
$$h_v = \boxed{ReLU}\left(\sum_{u \in \mathcal{N}(v)} (\boxed{W}\boxed{x_u} + \boxed{b})\right)$$

激活函数

节点v的邻居节点集合

待学习参数

初始节点特征

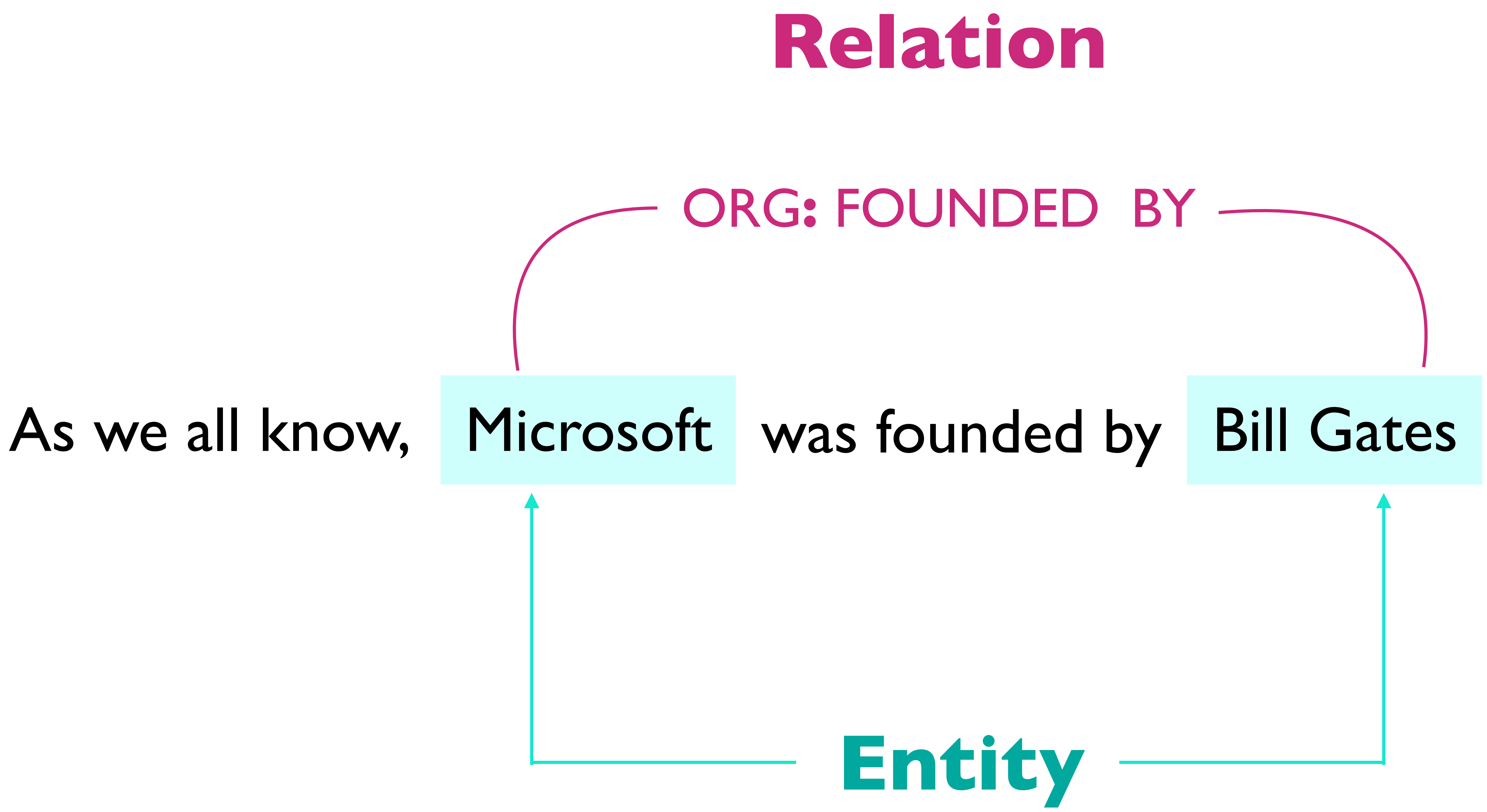


- K层GCN

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W^{(k)} \boxed{h_u^{(k)}} + b^{(k)}\right)$$

BGII: Relation Extraction(RE)

- Sentence-level RE Example



[ACL17] Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling

Diego Marcheggiani,[1] *Ivan Titov*[1,2]

[1] *ILLC, University of Amsterdam,* [2] *ILCC, School of Informatics, University of Edinburgh*

Contributions

- Syntactic GCN over syntactic dependency trees integrates syntax, context✅
- GCN, LSTM complement each other✅
- GCN-based SRL model

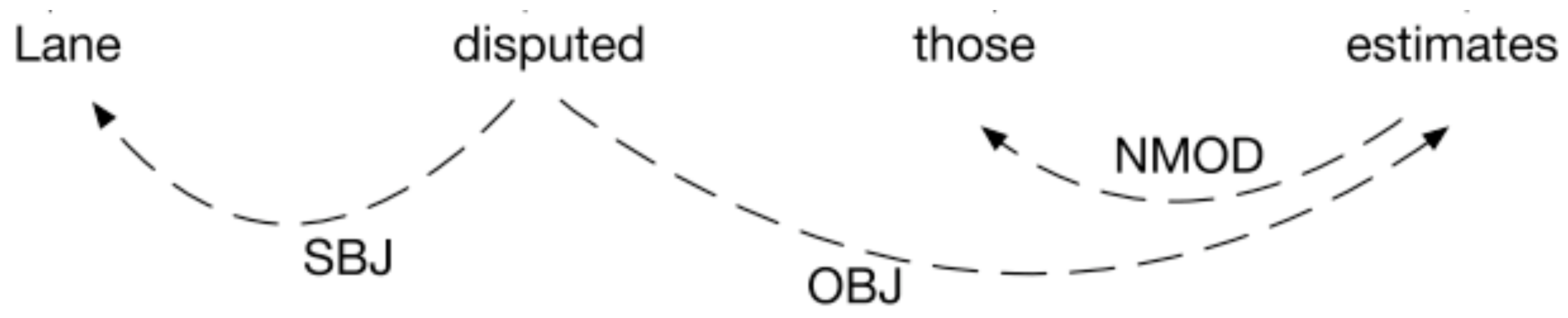
Syntactic GCNs

input sentece Lane disputed those estimates

Syntactic GCNs

input sentece

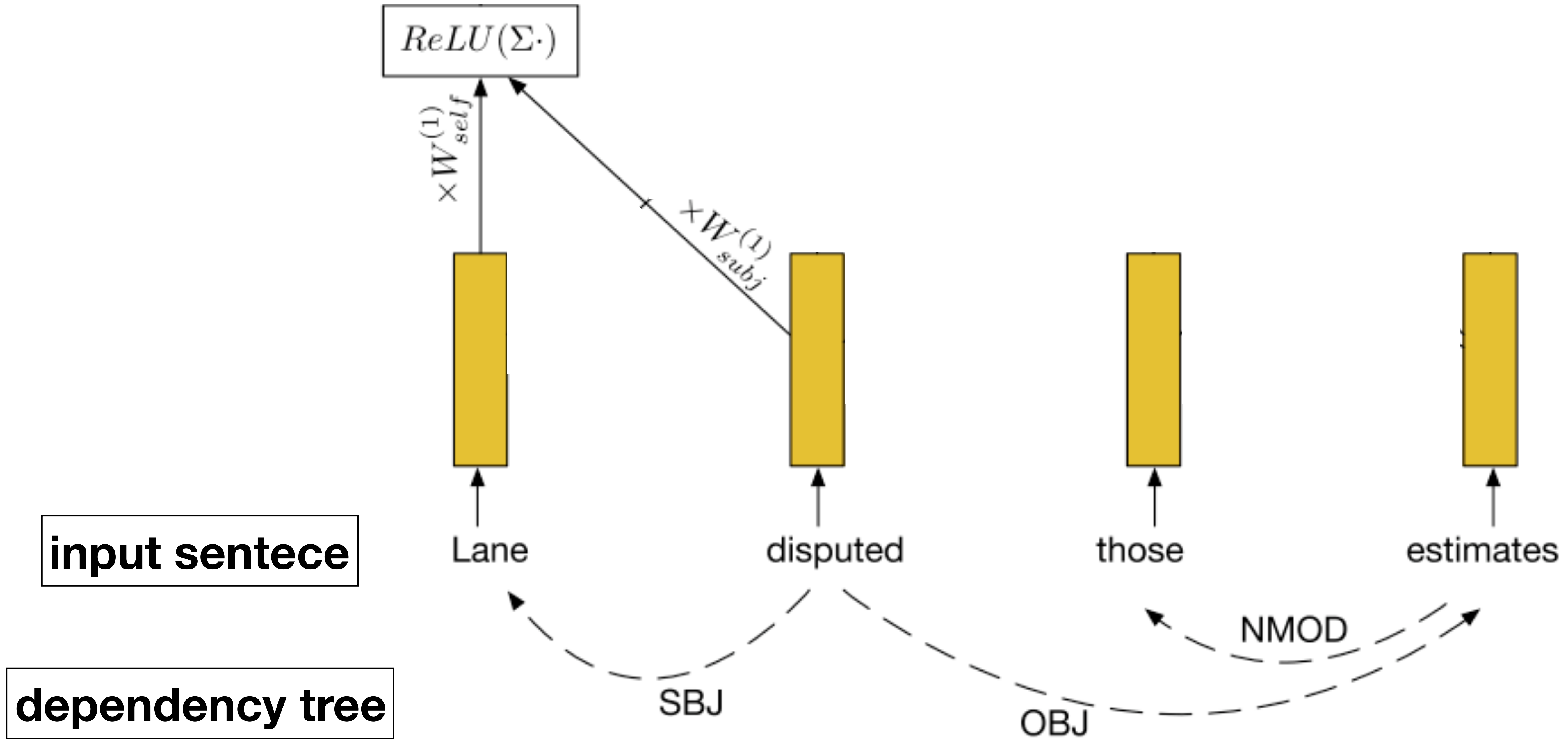
dependency tree



directed, labeled

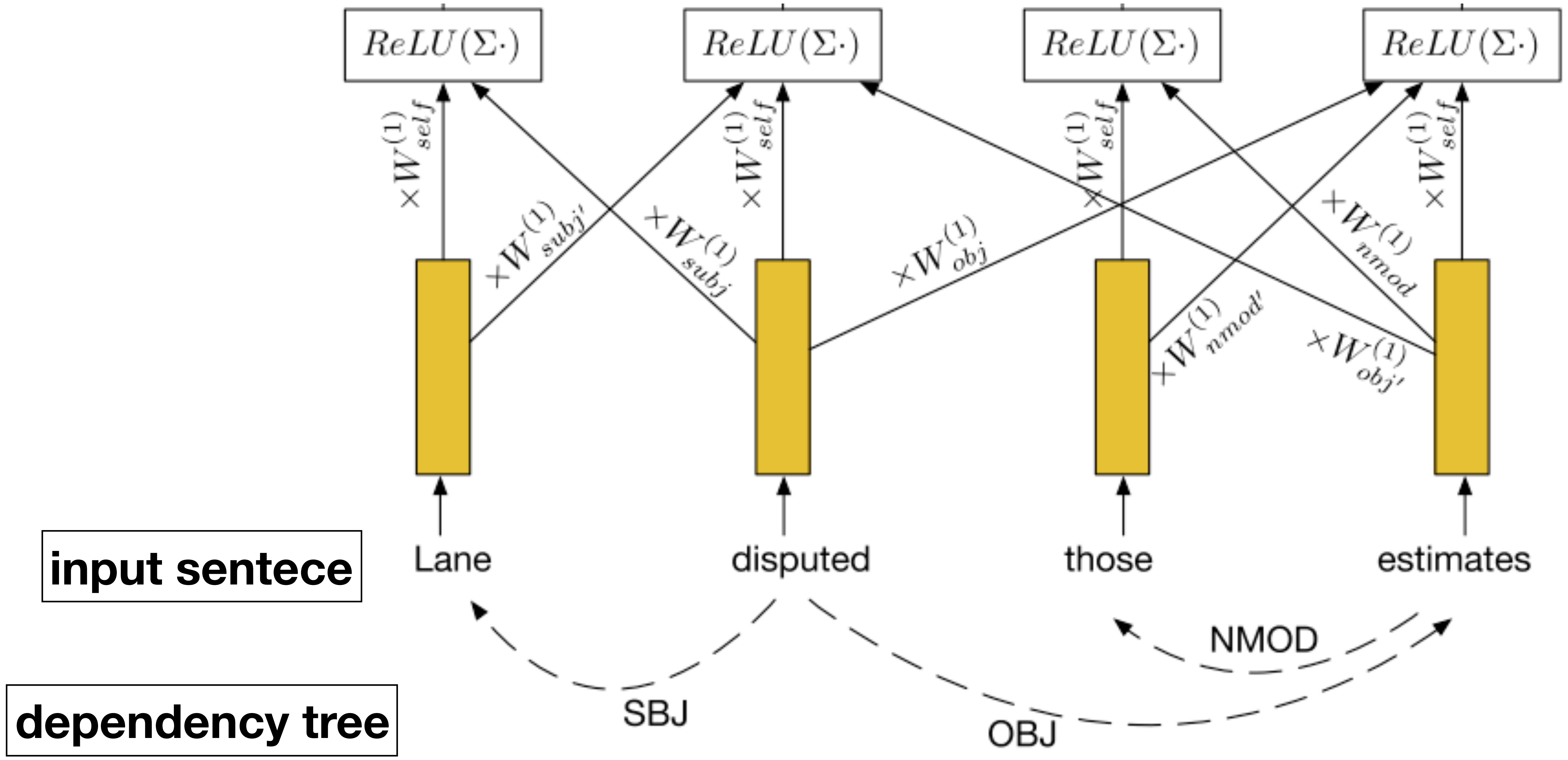
Syntactic GCNs

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)}\right)$$

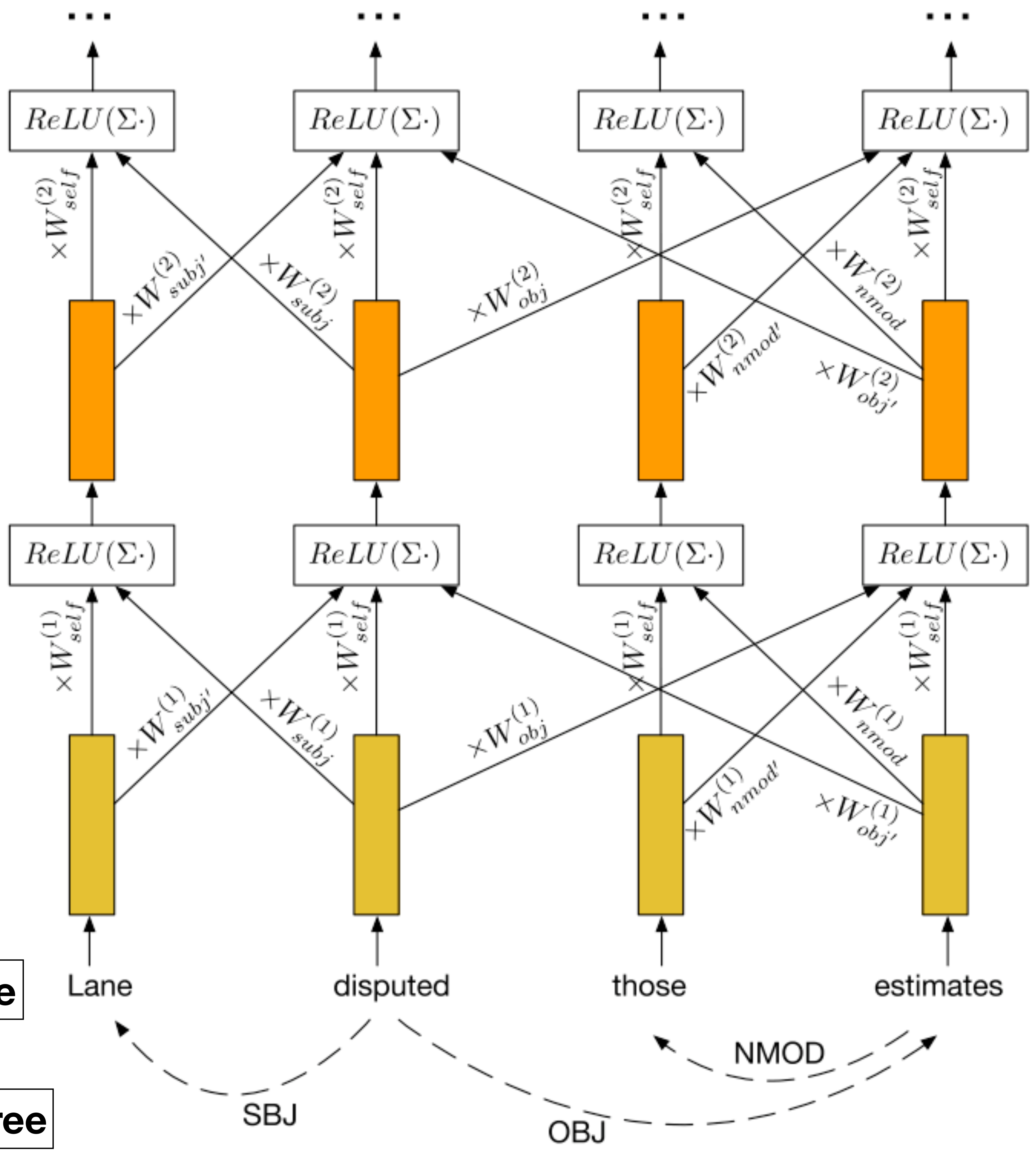


Syntactic GCNs

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)}\right)$$



Syntactic GCNs

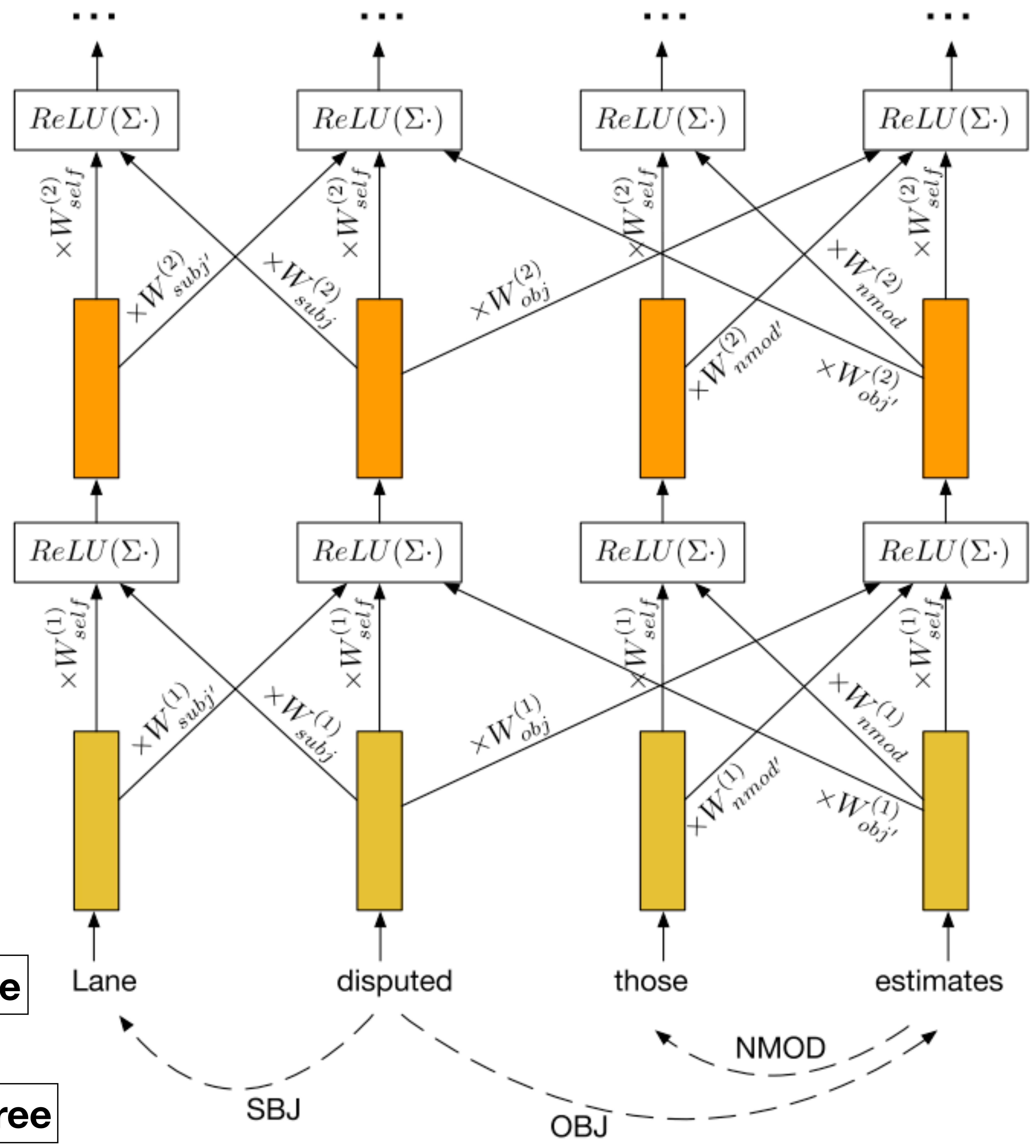


input sentece

dependency tree

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W^{(k)}_{L(u,v)} h_u^{(k)} + b^{(k)}_{L(u,v)}\right)$$

Syntactic GCNs



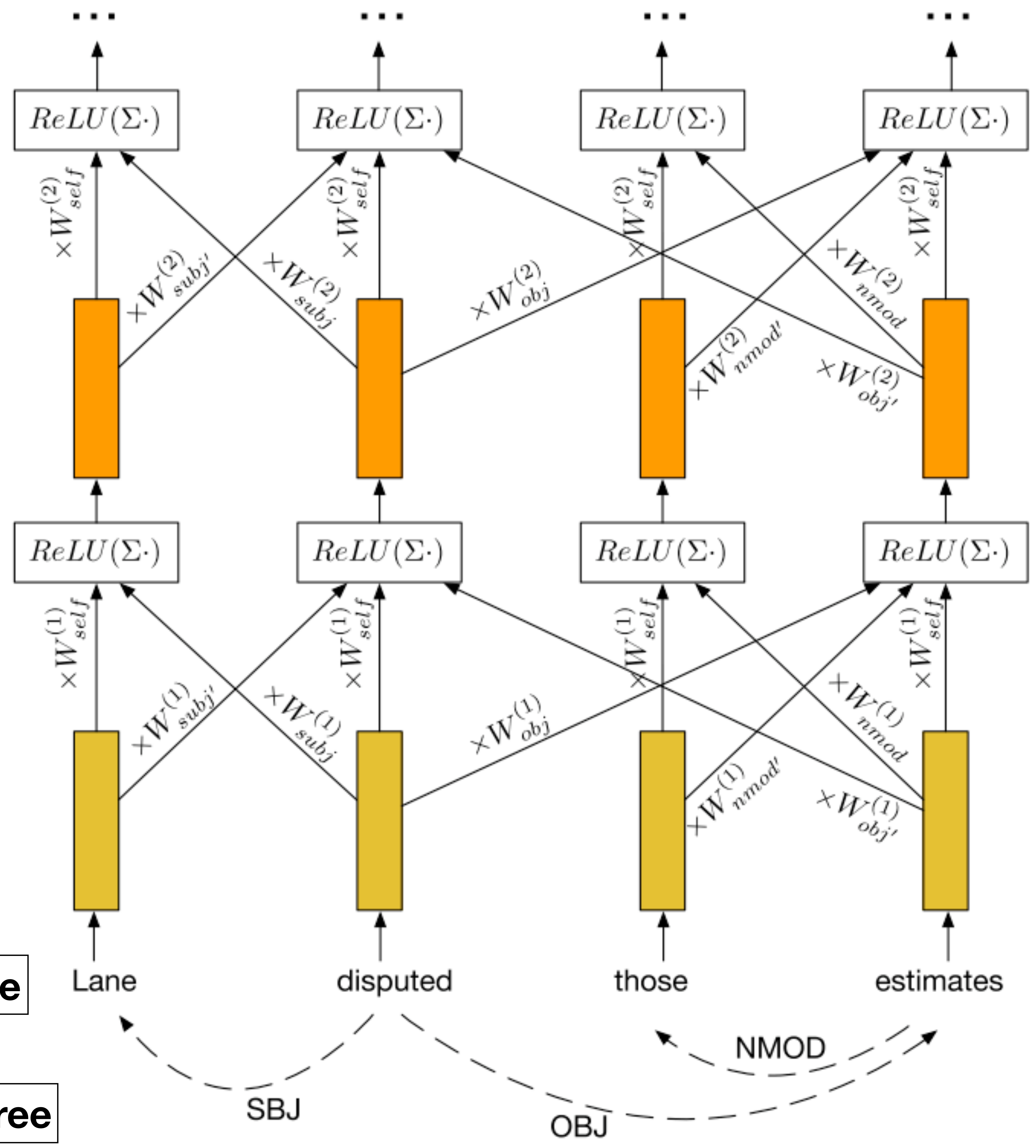
input sentece

dependency tree

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} \boxed{W_{L(u,v)}^{(k)}} h_u^{(k)} + \boxed{b_{L(u,v)}^{(k)}}\right)$$

over-parameterized

Syntactic GCNs



input sentece

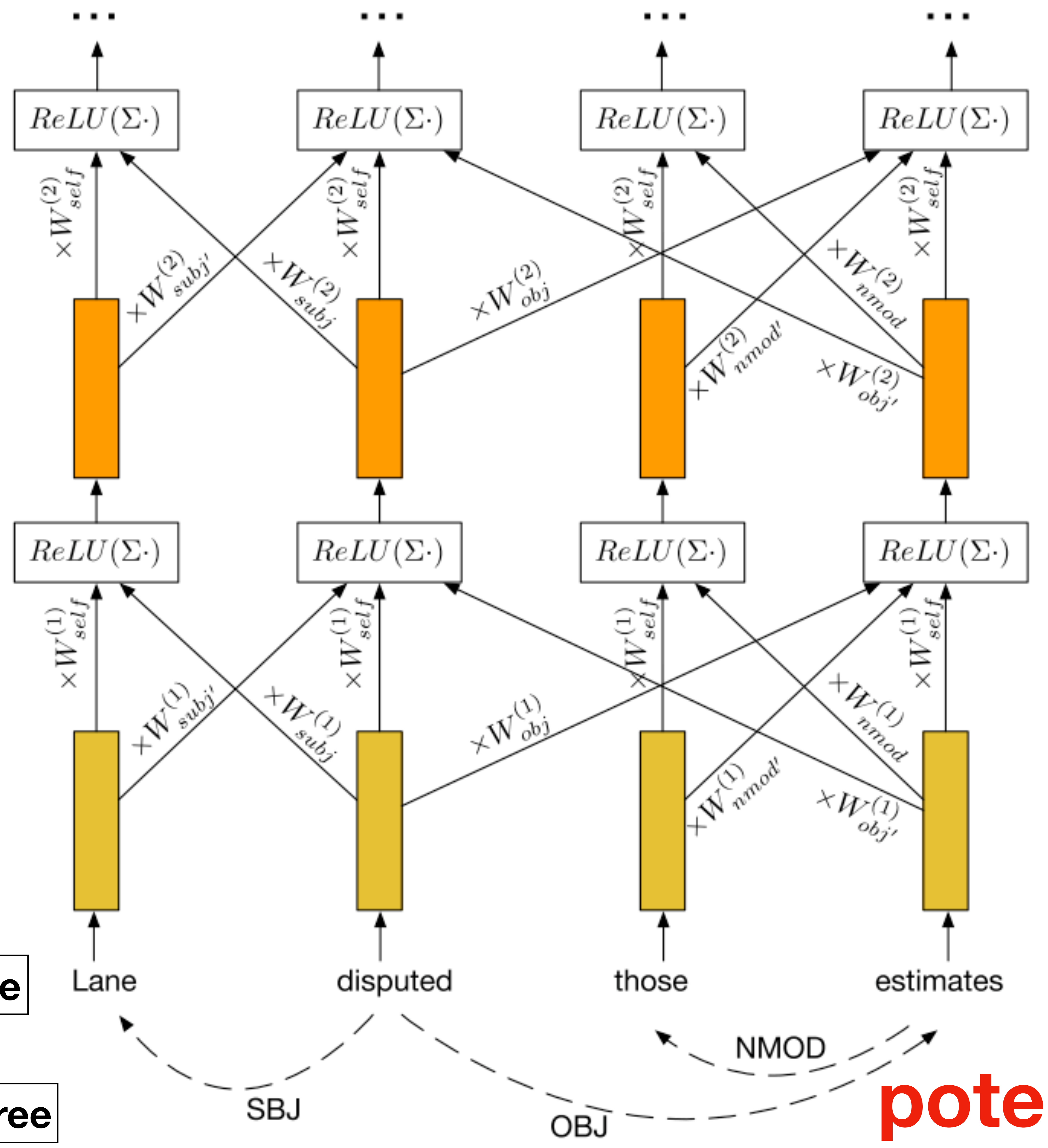
dependency tree

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W^{(k)}_{L(u,v)} h_u^{(k)} + b^{(k)}_{L(u,v)}\right)$$

$$W^{(k)}_{L(u,v)} = V^{(k)}_{dir(u,v)}$$

$$dir(u,v) = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

Syntactic GCNs



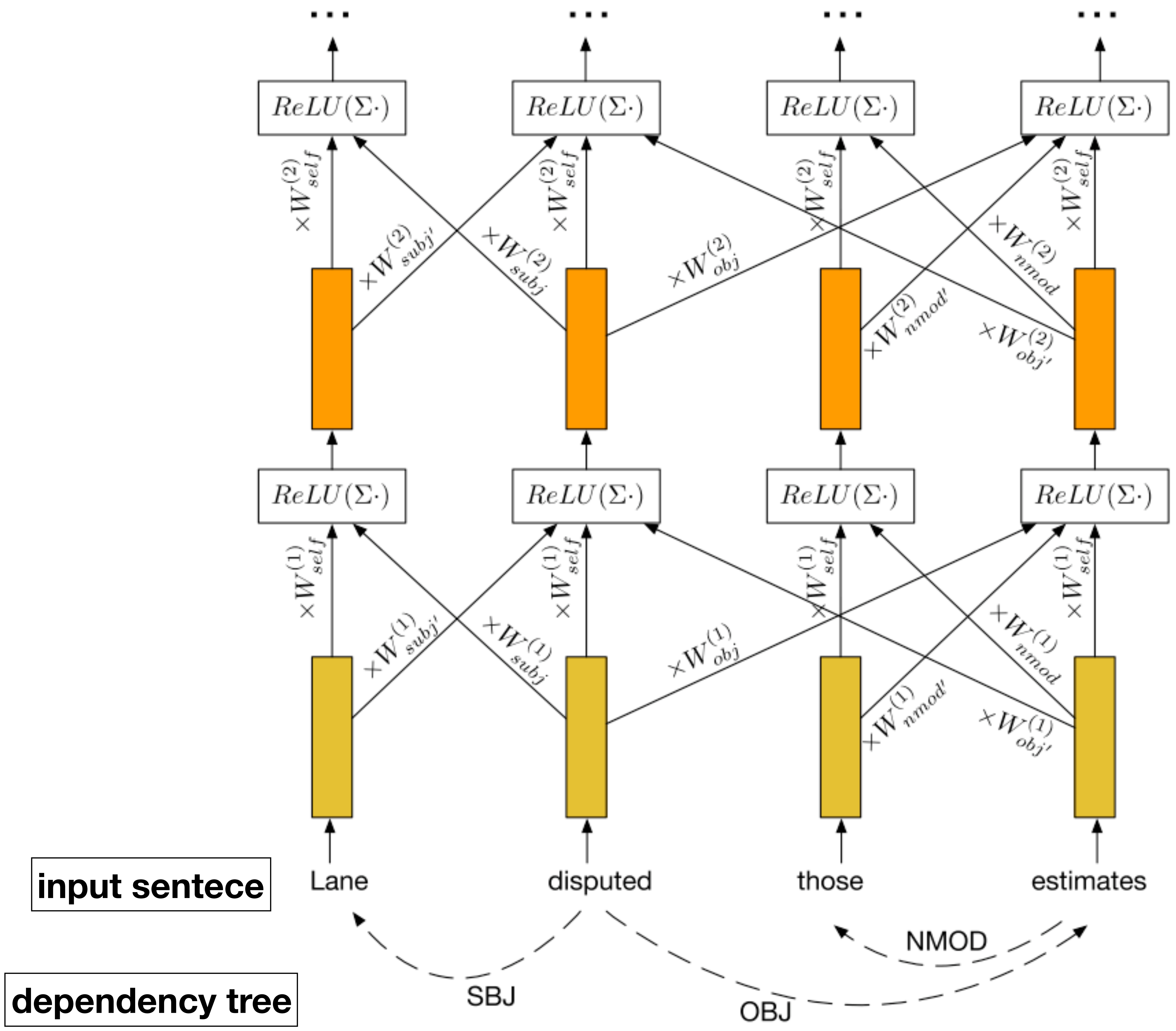
$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)}\right)$$

$$W_{L(u,v)}^{(k)} = V_{dir(u,v)}^{(k)}$$

$$dir(u,v) = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

potentially wrong edge

Syntactic GCNs



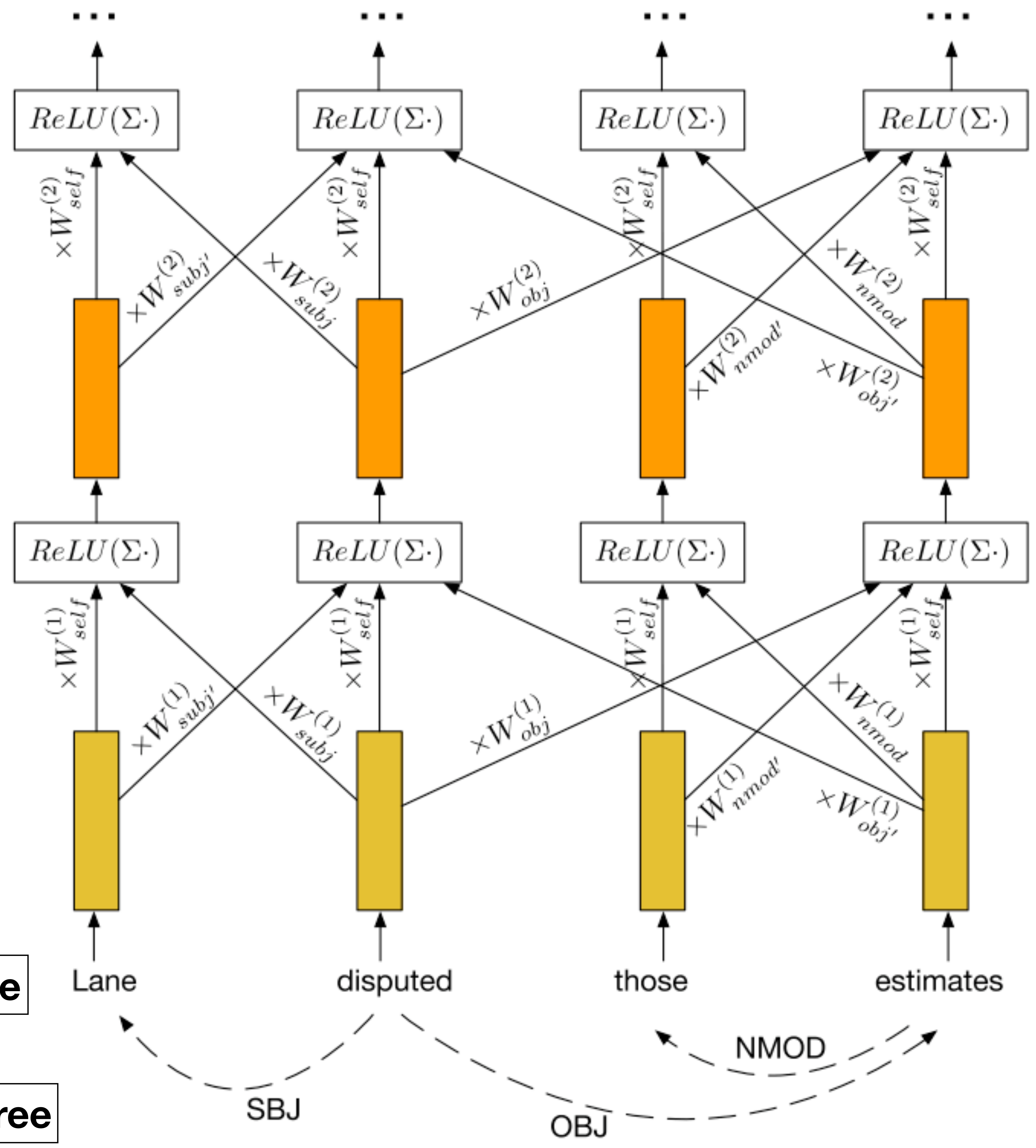
$$h_v^{(k+1)} = ReLU\left(\sum_{u\in\mathcal{N}(v)} W^{(k)}_{L(u,v)} h_u^{(k)} + b^{(k)}_{L(u,v)}\right)$$

$$W^{(k)}_{L(u,v)} = V^{(k)}_{dir(u,v)}$$

$$dir(u,v) = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

$$g^{(k)}_{u,v} = \sigma\left(h_u^{(k)} \cdot \hat{v}^{(k)}_{dir(u,v)} + \hat{b}^{(k)}_{L(u,v)}\right)$$ **Edge-wise gating**

Syntactic GCNs



input sentece

dependency tree

$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)}\right)$$

$$W_{L(u,v)}^{(k)} = V_{dir(u,v)}^{(k)}$$

$$dir(u,v) = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

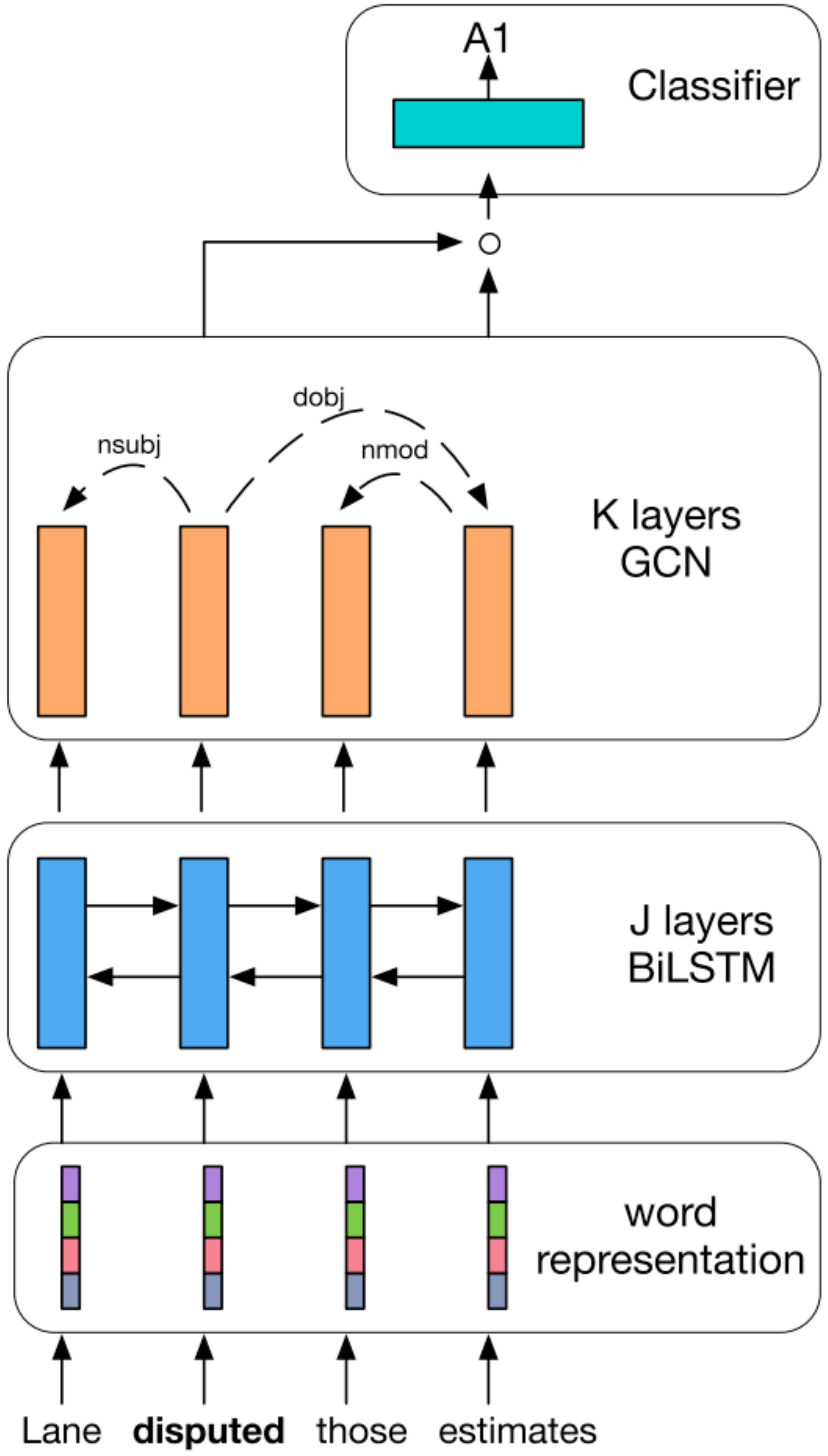
$$g_{u,v}^{(k)} = \sigma\left(h_u^{(k)} \cdot \hat{v}_{dir(u,v)}^{(k)} + \hat{b}_{L(u,v)}^{(k)}\right)$$ **Edge-wise gating**

$$\boxed{h_v^{(k+1)} = ReLU\Big(\sum_{u \in \mathcal{N}(v)} g_{v,u}^{(k)} (V_{dir(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)})\Big)}$$

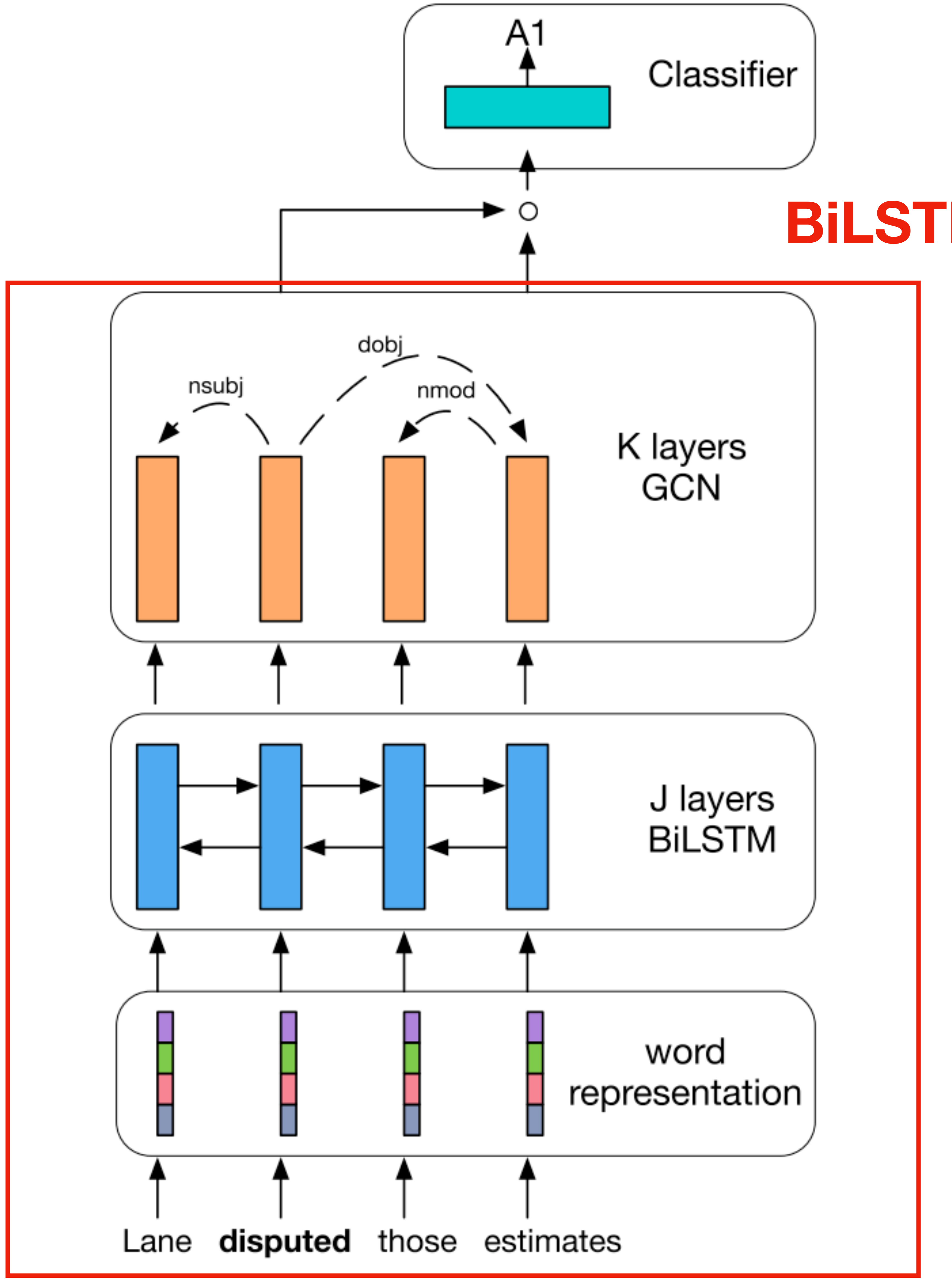
Complementarity of GCNs and LSTMs

- LSTM vs GCN
  - LSTM: extract sequential information
  - GCN: extract regional information, ’teleport’ even over a single (longest) syntactic dependency edge

LSTM-based SRL model



Syntax-Aware Neural SRL Encoder



BiLSTM + Dependency Tree ➡️ GCN

Experiments

System (English)	P	R	F_1
LSTMs	84.3	81.1	$\boxed{82.7}$
LSTMs + GCNs (K=1)	85.2	81.6	$\boxed{83.3}$
LSTMs + GCNs (K=2)	84.1	81.4	82.7
LSTMs + GCNs (K=1), no gates	84.7	81.4	83.0
GCNs (no LSTMs), K=1	79.9	70.4	74.9
GCNs (no LSTMs), K=2	83.4	74.6	78.7
GCNs (no LSTMs), K=3	83.6	75.8	79.5
GCNs (no LSTMs), K=4	82.7	76.0	79.2

Table 1: SRL results without predicate disambiguation on the English development set.

[ACL18] RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information

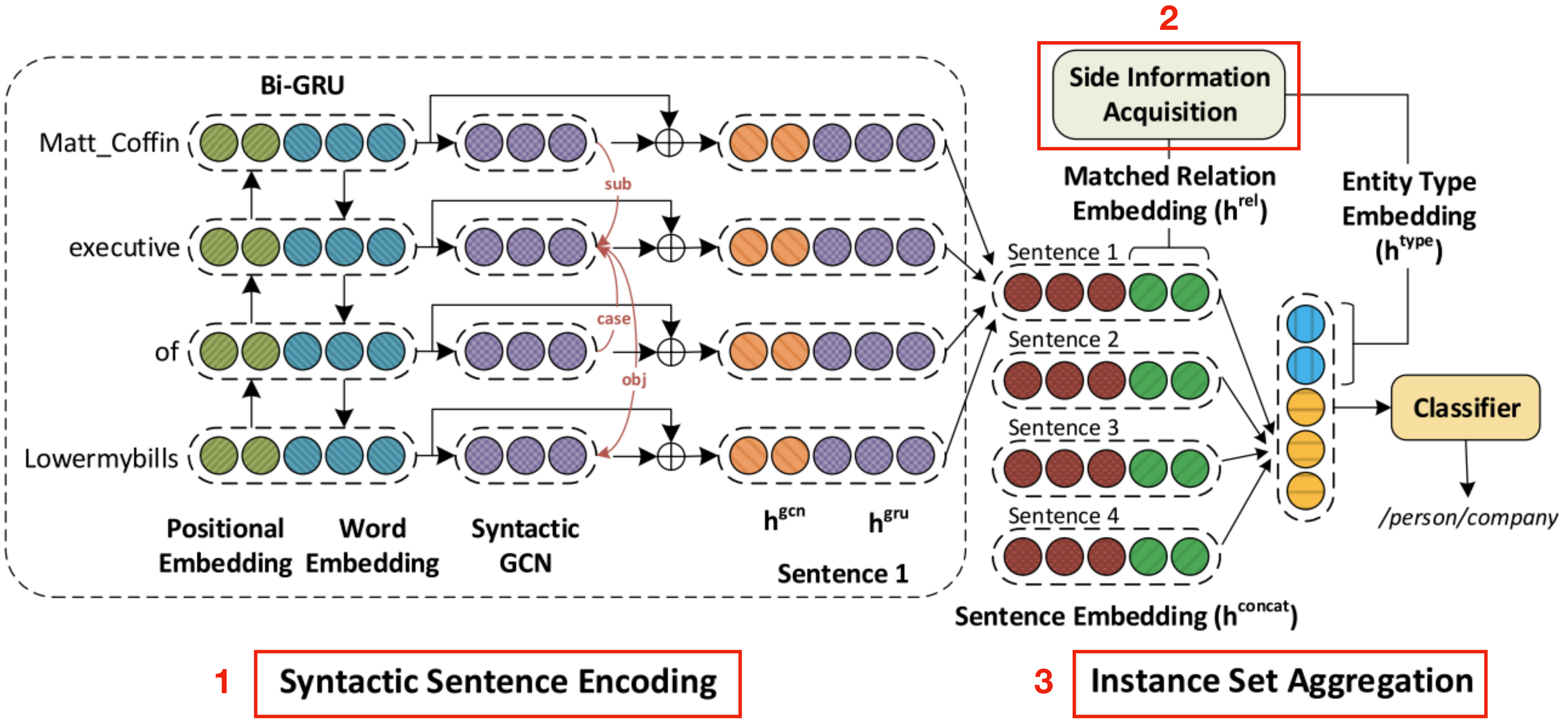
Shikhar Vashishth[1], *Rishabh Joshi*[2], *Sai Suman Prayaga*[1], *Chiranjib Bhattacharyya*[1], *Partha Talukdar*[1]

[1]*Indian Institute of Science*, [2]*Birla Institute of Technology and Science, Pilani*

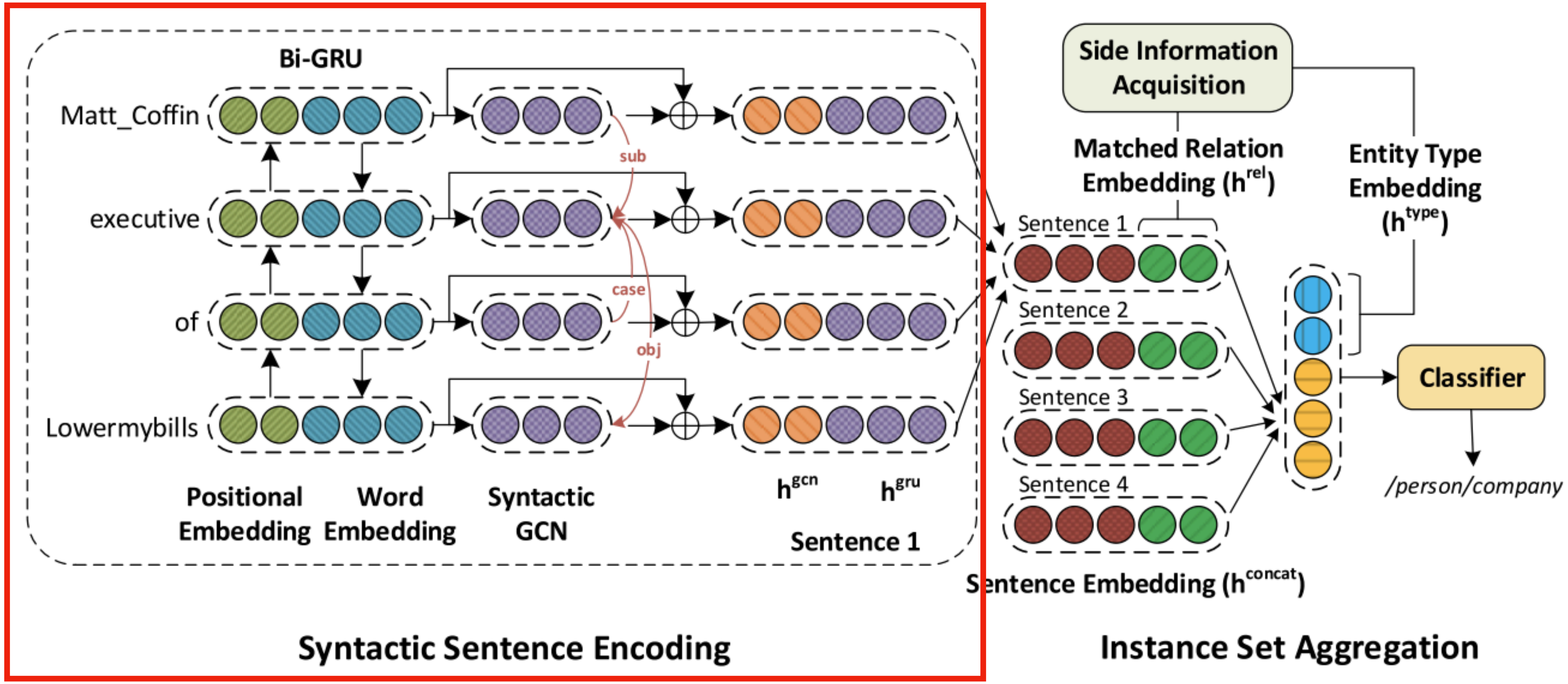
RESIDE

- Task
  - Multi-instance learning: given a bag of sentences (or instances) {s1, s2, ...sn} for a given entity pair, predict the relation between them.
- Motivation
  - Distantly-supervised Relation Extraction methods use relation instances in Knowledge Base (KB)
  - KBs often contain other relevant side information, such as aliases of relations

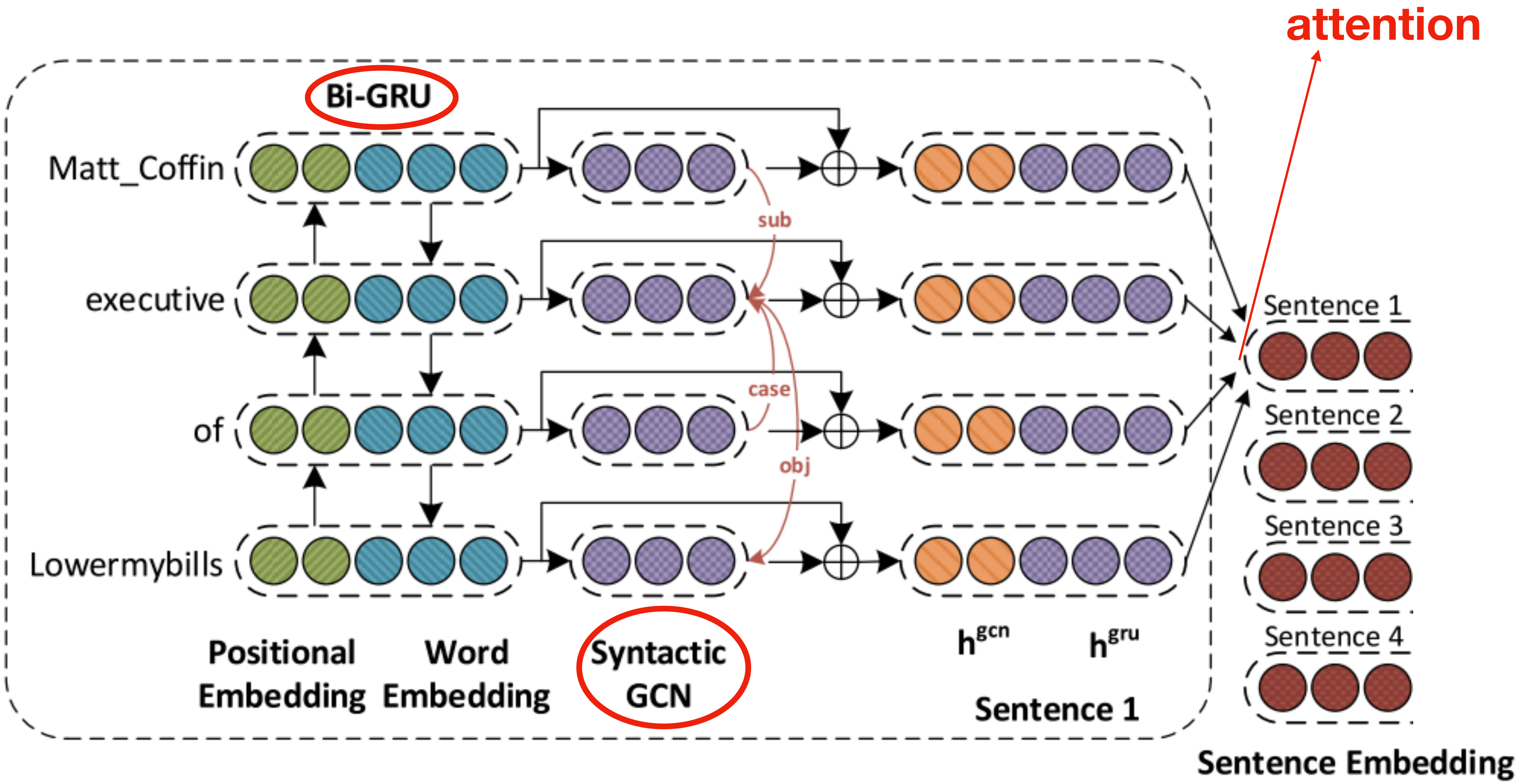
RESIDE Architecture



PartI: Syntactic Sentence Encoding

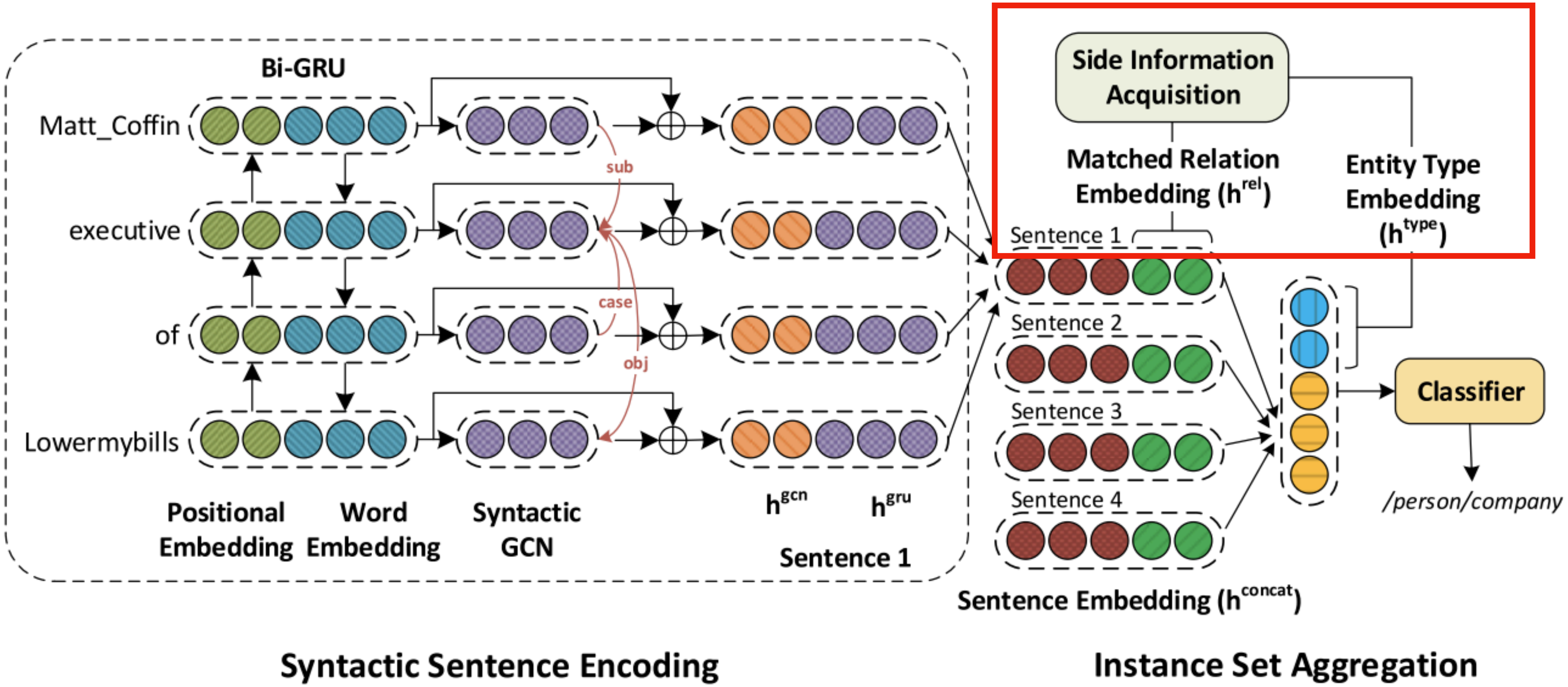


PartI: Syntactic Sentence Encoding

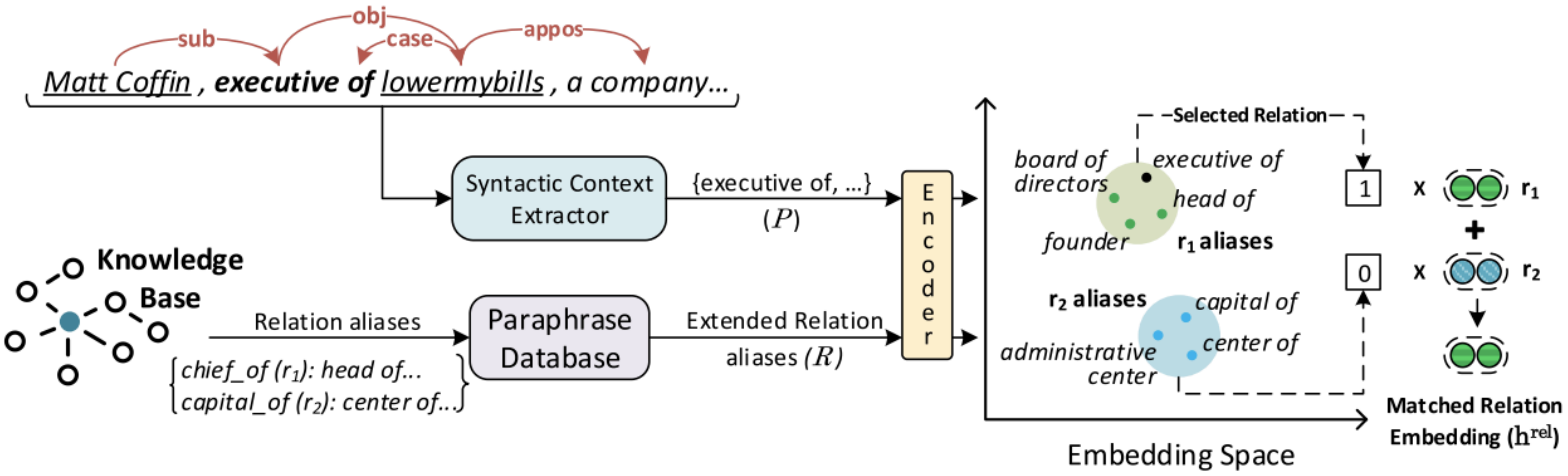


Syntactic Sentence Encoding

PartII: Side Information Acquisition

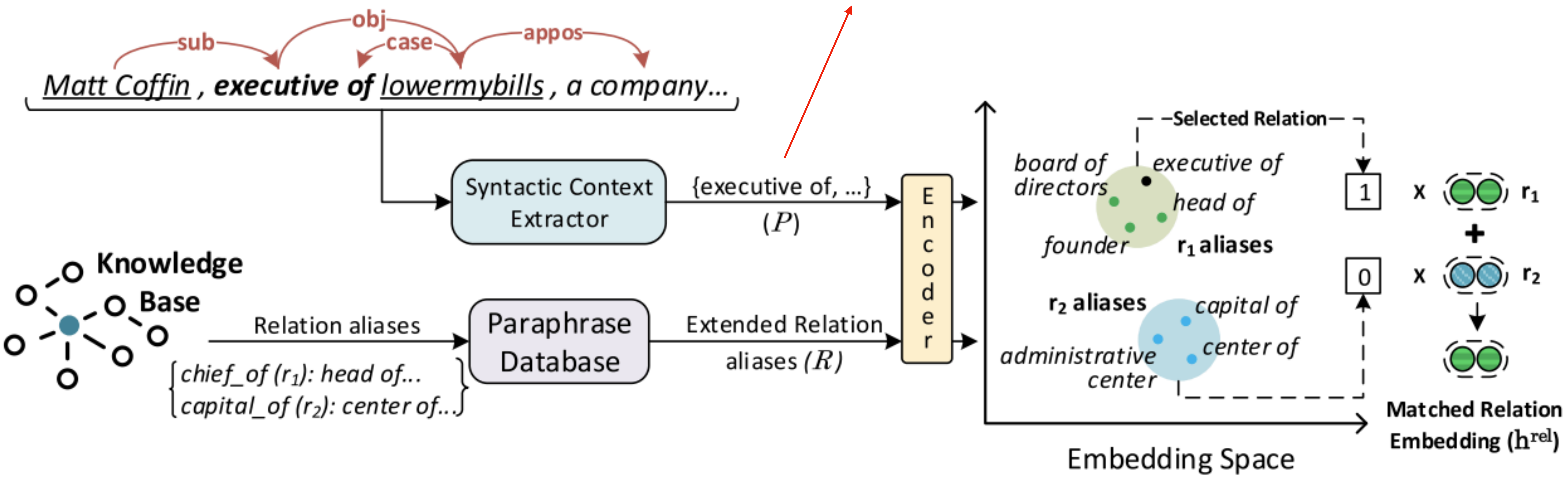


PartII: Relation Alias Side Information



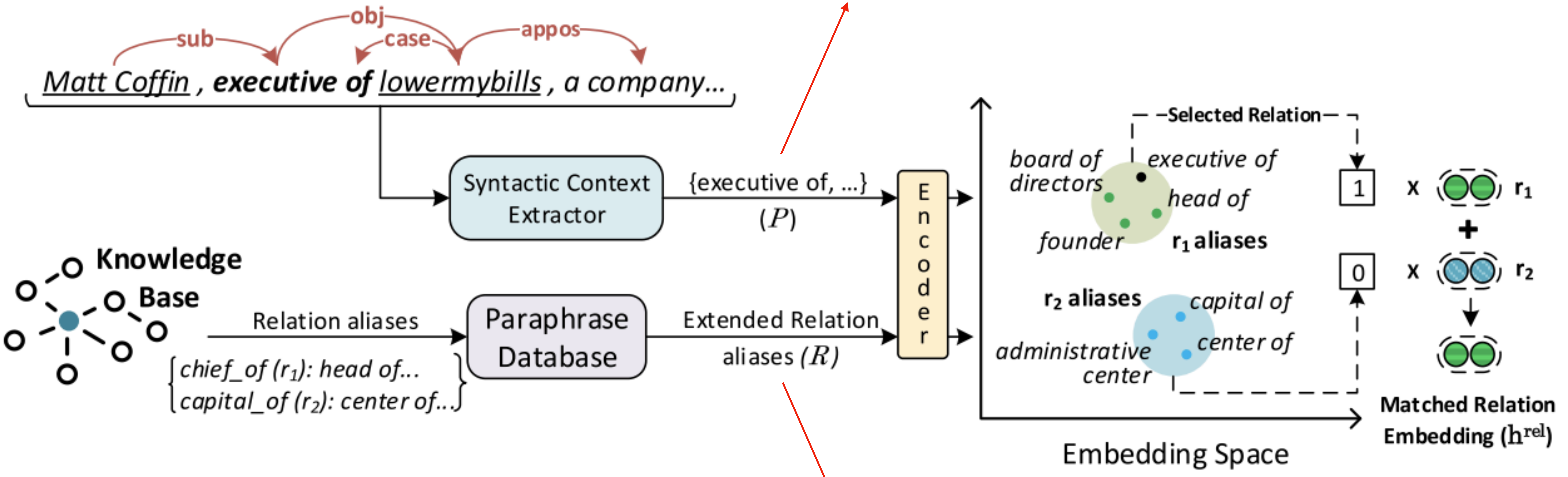
PartII: Relation Alias Side Information

relation phrases



PartII: Relation Alias Side Information

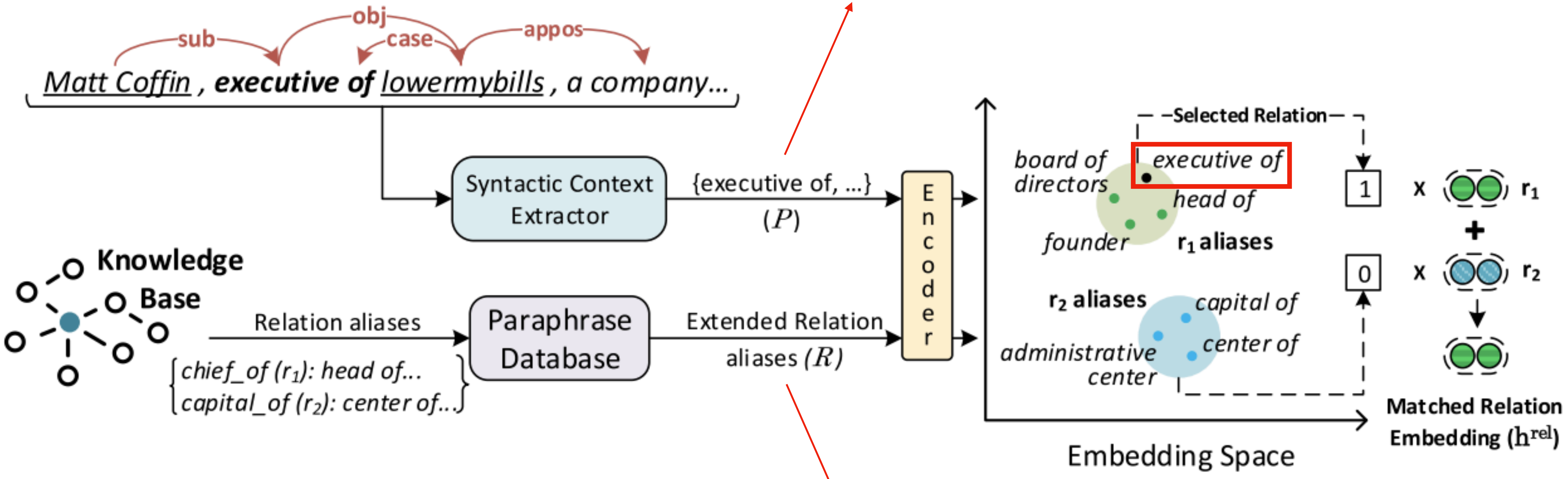
relation phrases



relation aliases

PartII: Relation Alias Side Information

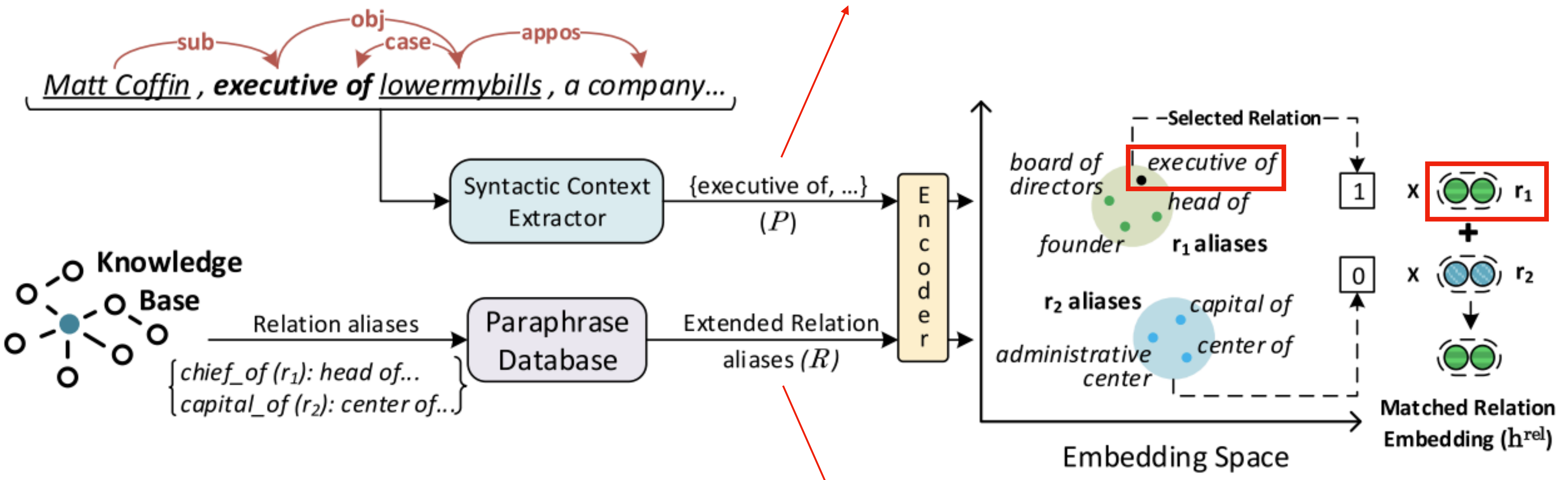
relation phrases



relation aliases

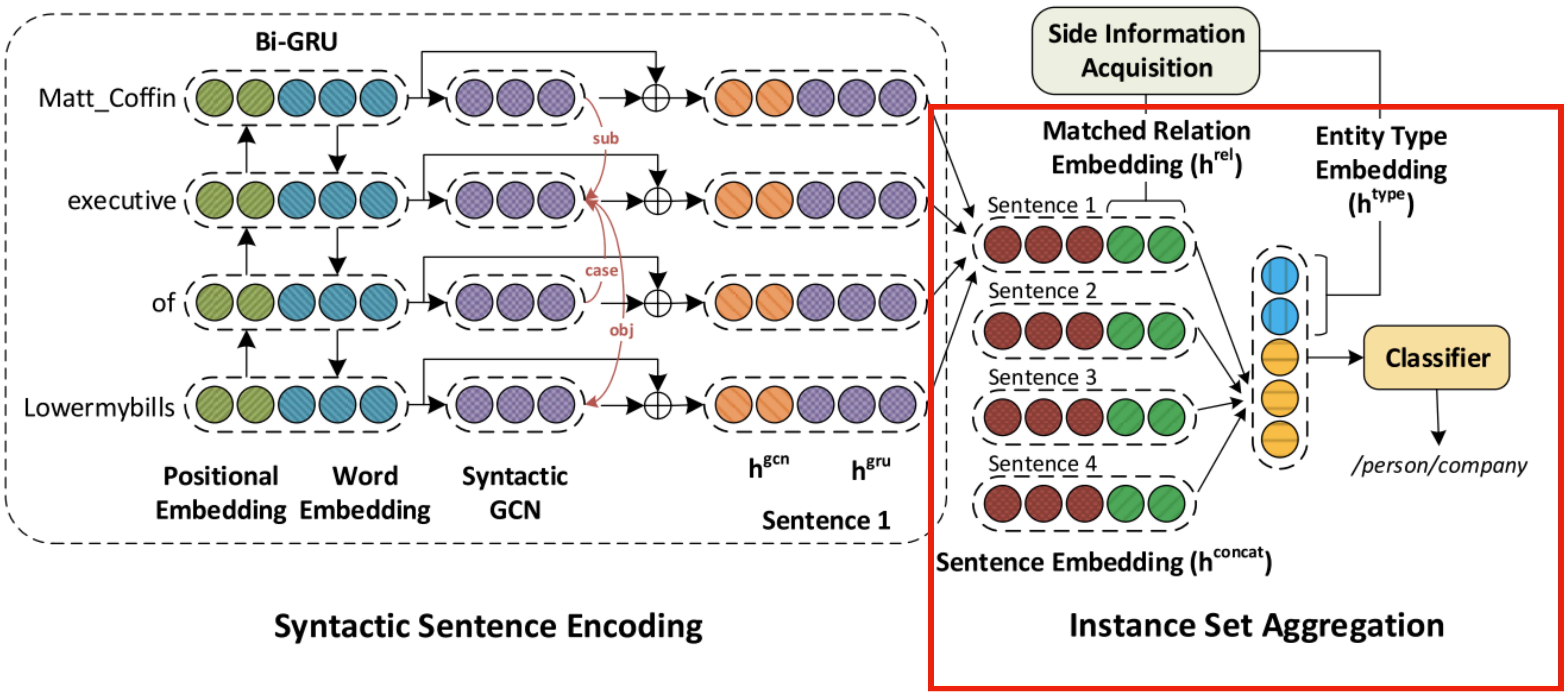
PartII: Relation Alias Side Information

relation phrases

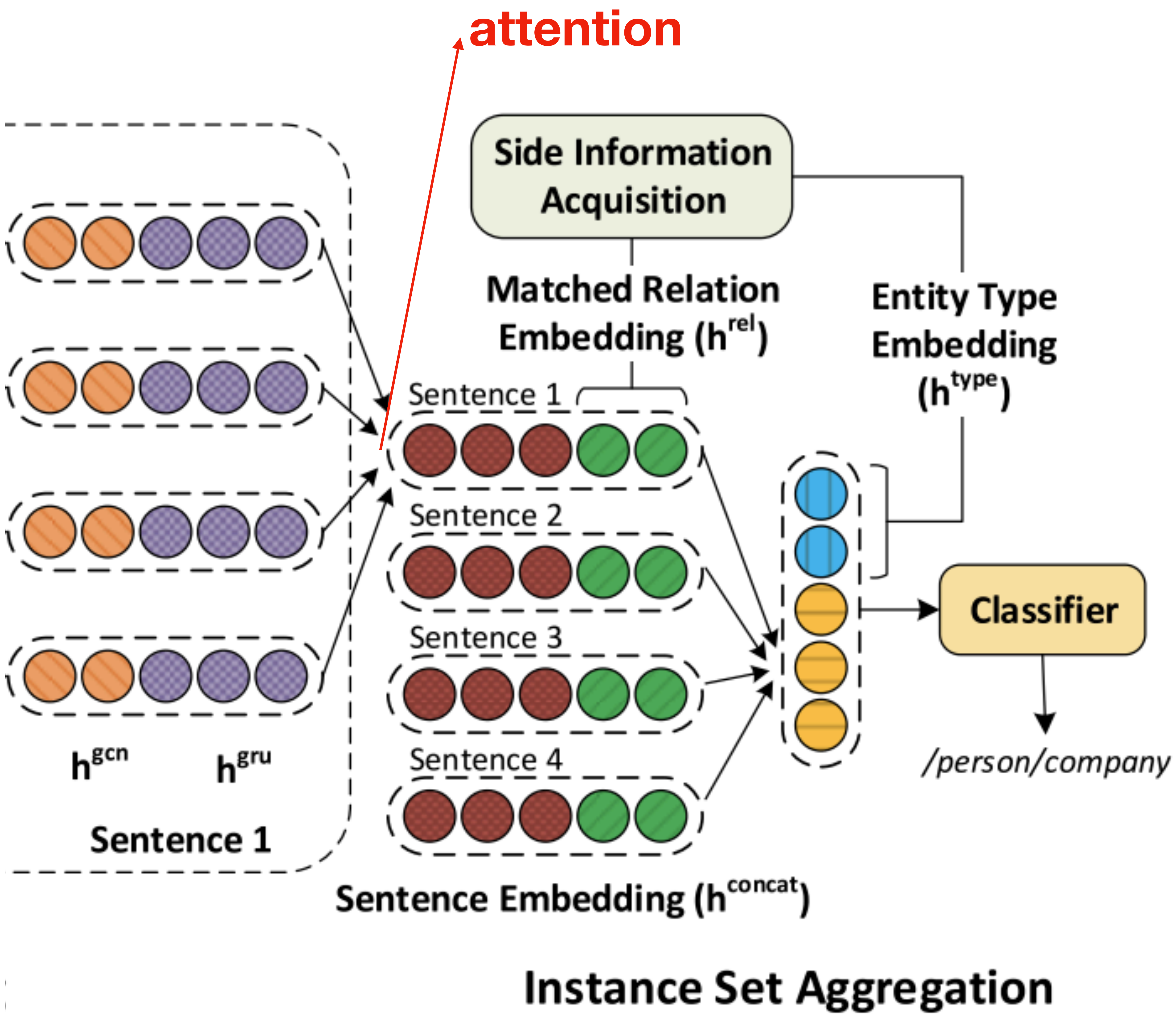


relation aliases

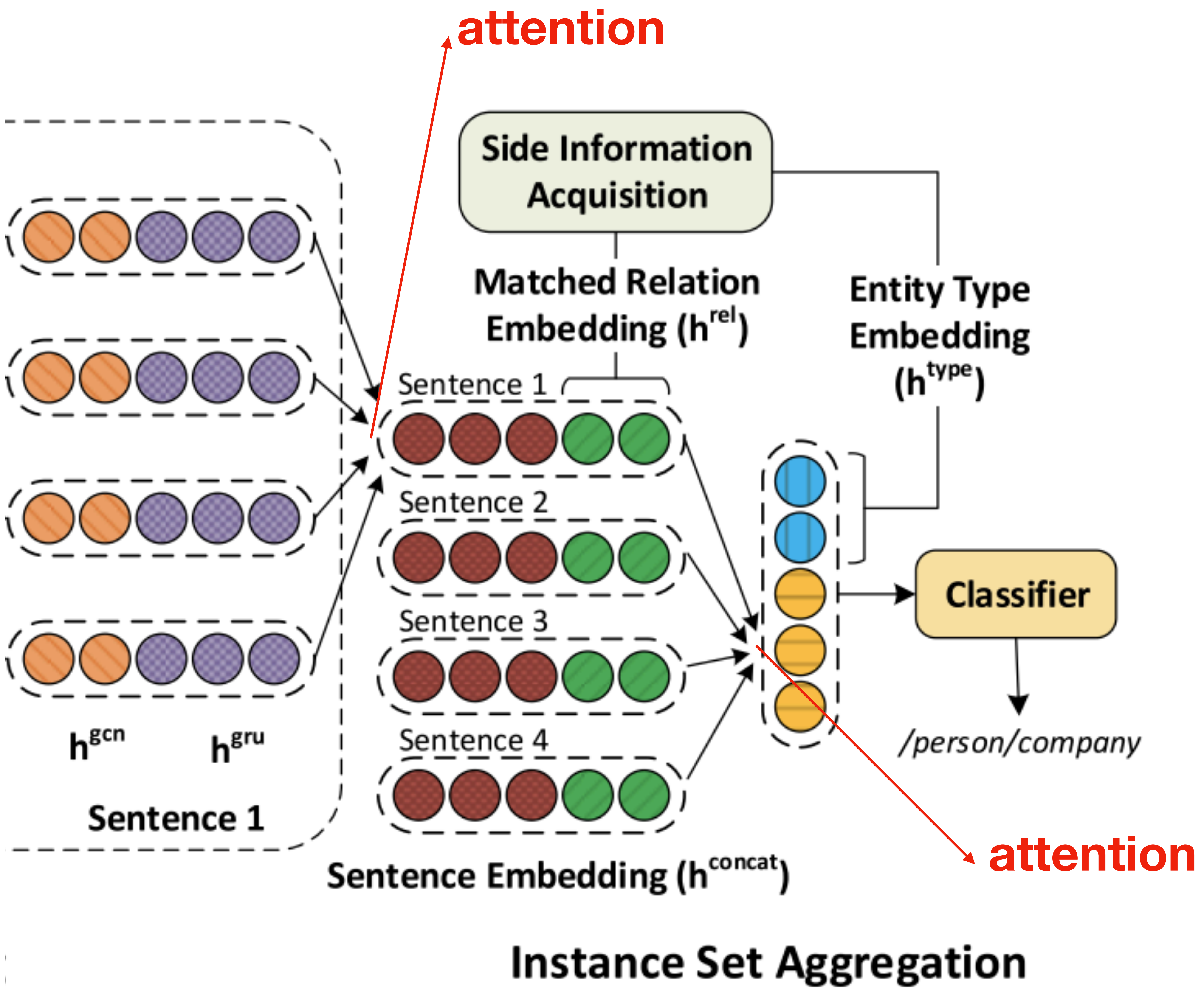
PartIII: Instance Set Aggregation



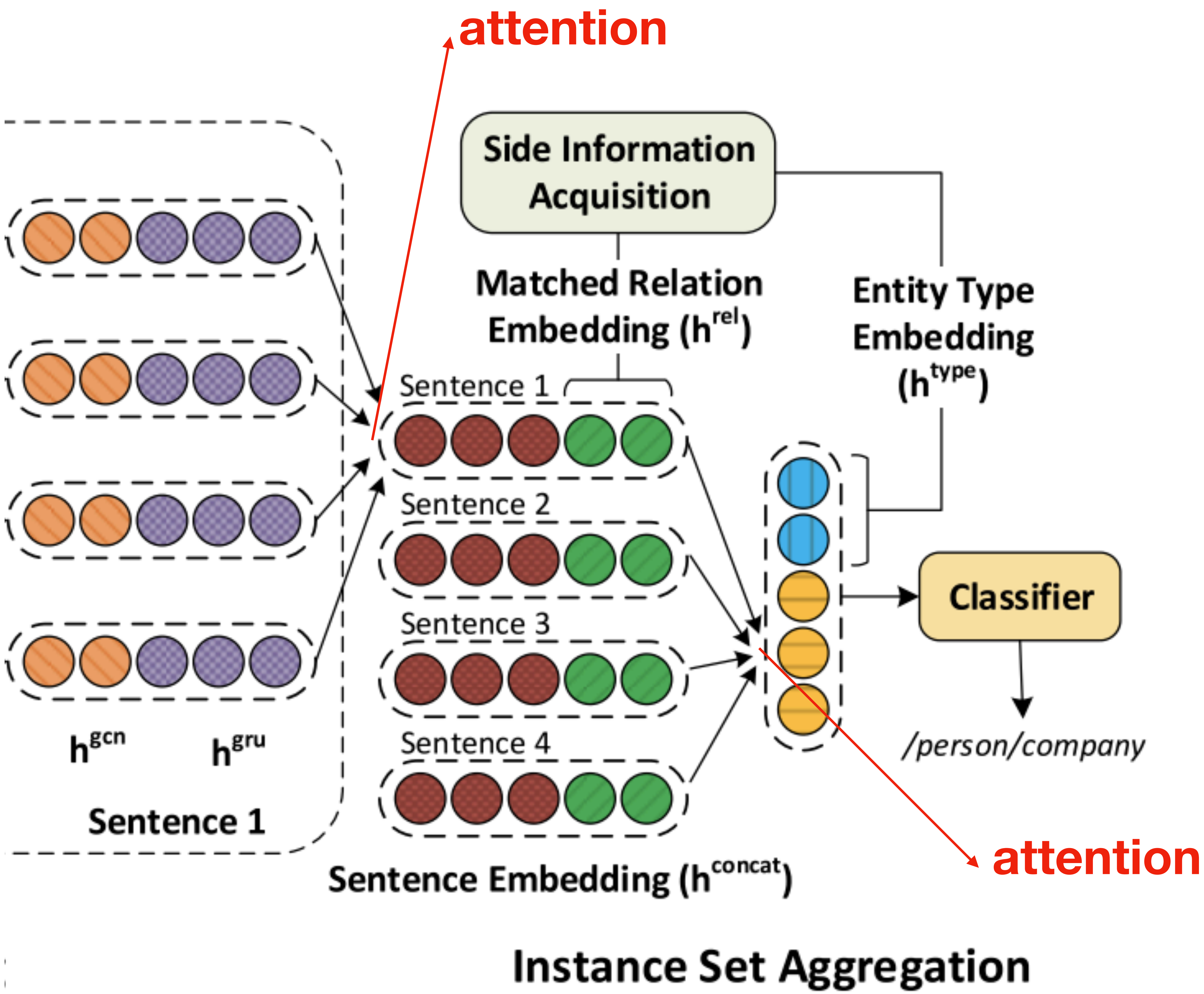
PartIII: Instance Set Aggregation



PartIII: Instance Set Aggregation



PartIII: Instance Set Aggregation



$$\hat{\mathcal{B}} = [\mathcal{B}; h_{sub}^{type}; h_{obj}^{type}] \text{ where, } \mathcal{B} = \sum_{i=1}^{n} \alpha_i \hat{s}_i$$

$$p(y) = \text{Softmax}(W \cdot \hat{\mathcal{B}} + b)$$

Experiments

	One			Two			All		
	P@100	P@200	P@300	P@100	P@200	P@300	P@100	P@200	P@300
PCNN	73.3	64.8	56.8	70.3	67.2	63.1	72.3	69.7	64.1
PCNN+ATT	73.3	69.2	60.8	77.2	71.6	66.1	76.2	73.1	67.4
BGWA	78.0	71.0	63.3	81.0	73.0	64.0	82.0	75.0	72.0
RESIDE	**80.0**	**75.5**	**69.3**	**83.0**	**73.5**	**70.6**	**84.0**	**78.5**	**75.6**

Table 2: P@N for relation extraction using variable number of sentences in bags (with more than one sentence) in Riedel dataset. Here, One, Two and All represents the number of sentences randomly selected from a bag. RESIDE attains improved precision in all settings.

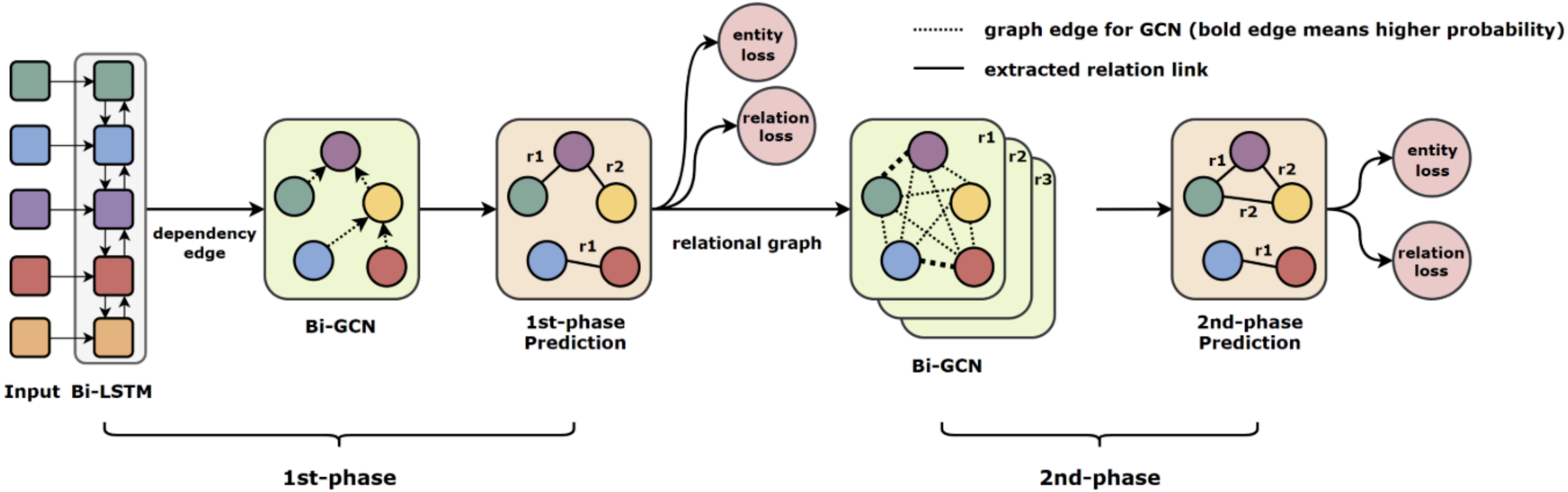
[ACL19] GraphRel: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction

Tsu-Jui Fu,[1] *Peng-Hsuan Li,*[1] *Wei-Yun Ma*[1]

[1] *Academia Sinica*

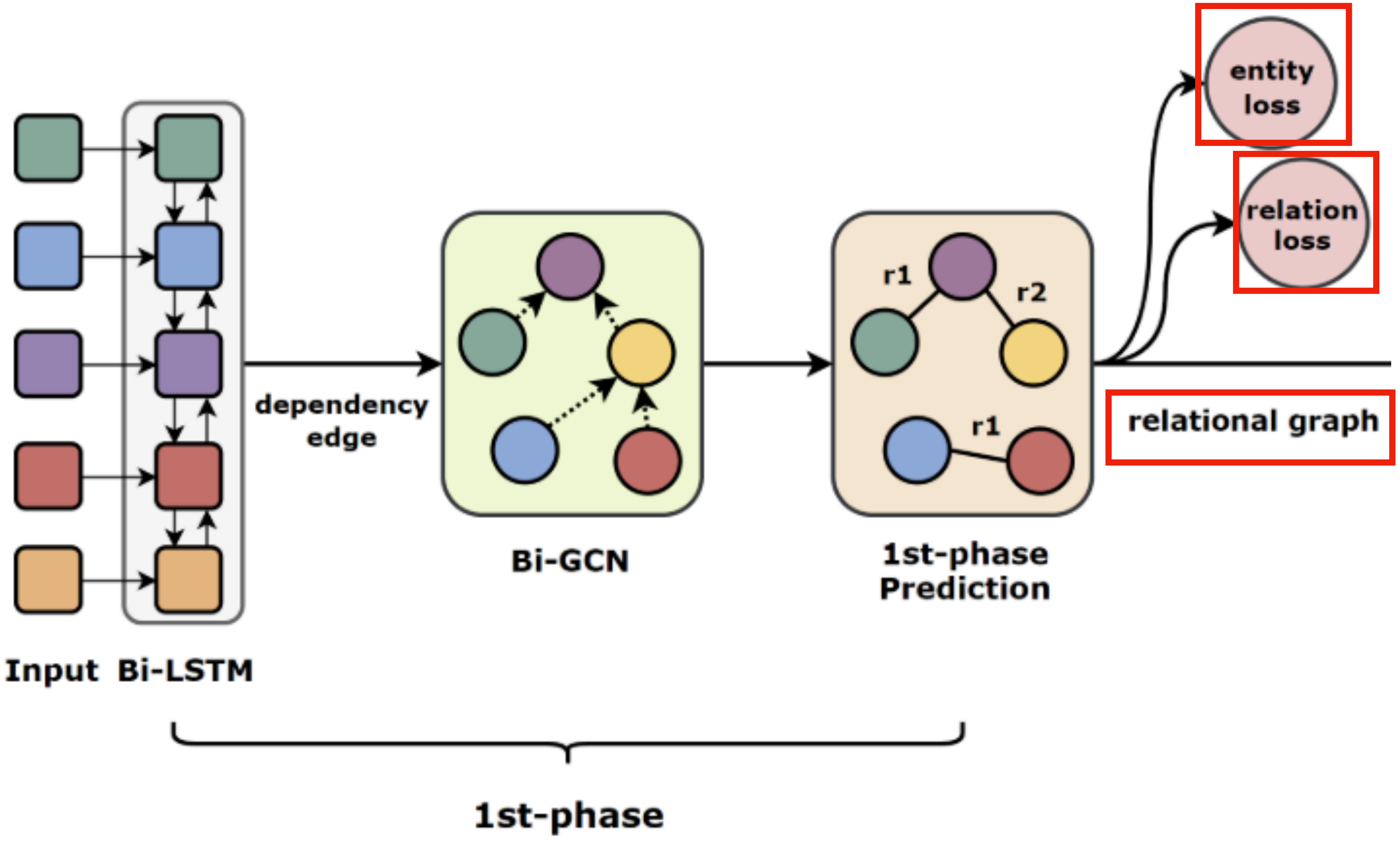
1

GraphRel Architecture



1st-phase Prediction

- Bi-LSTM + Dependency Edge ➡️ Bi-GCN ➡️ Prediction



1st-phase Prediction

- Bi-LSTM

$$h_u^0 = Word(u) \oplus POS(u)$$

1st-phase Prediction

- Bi-LSTM

$$h_u^0 = Word(u) \oplus POS(u)$$

- Bi-GCN

$$\overrightarrow{h}_u^{l+1} = ReLU\left(\sum_{v \in \overrightarrow{N}(u)} \left(\overrightarrow{W}^l h_v^l + \overrightarrow{b}^l\right)\right)$$

$$\overleftarrow{h}_u^{l+1} = ReLU\left(\sum_{v \in \overleftarrow{N}(u)} \left(\overleftarrow{W}^l h_v^l + \overleftarrow{b}^l\right)\right)$$

$$h_u^{l+1} = \overrightarrow{h}_u^{l+1} \oplus \overleftarrow{h}_u^{l+1},$$

1st-phase Prediction

- Bi-LSTM

$$h_u^0 = Word(u) \oplus POS(u)$$

- Bi-GCN

$$\overrightarrow{h}_u^{l+1} = ReLU\left(\sum_{v \in \overrightarrow{N}(u)} \left(\overrightarrow{W}^l h_v^l + \overrightarrow{b}^l\right)\right)$$

$$\overleftarrow{h}_u^{l+1} = ReLU\left(\sum_{v \in \overleftarrow{N}(u)} \left(\overleftarrow{W}^l h_v^l + \overleftarrow{b}^l\right)\right)$$

$$h_u^{l+1} = \overrightarrow{h}_u^{l+1} \oplus \overleftarrow{h}_u^{l+1},$$

- Prediction

$$S_{(w1,r,w2)} = W_r^3 ReLU\left(W_r^1 h_{w1} \oplus W_r^2 h_{w2}\right)$$

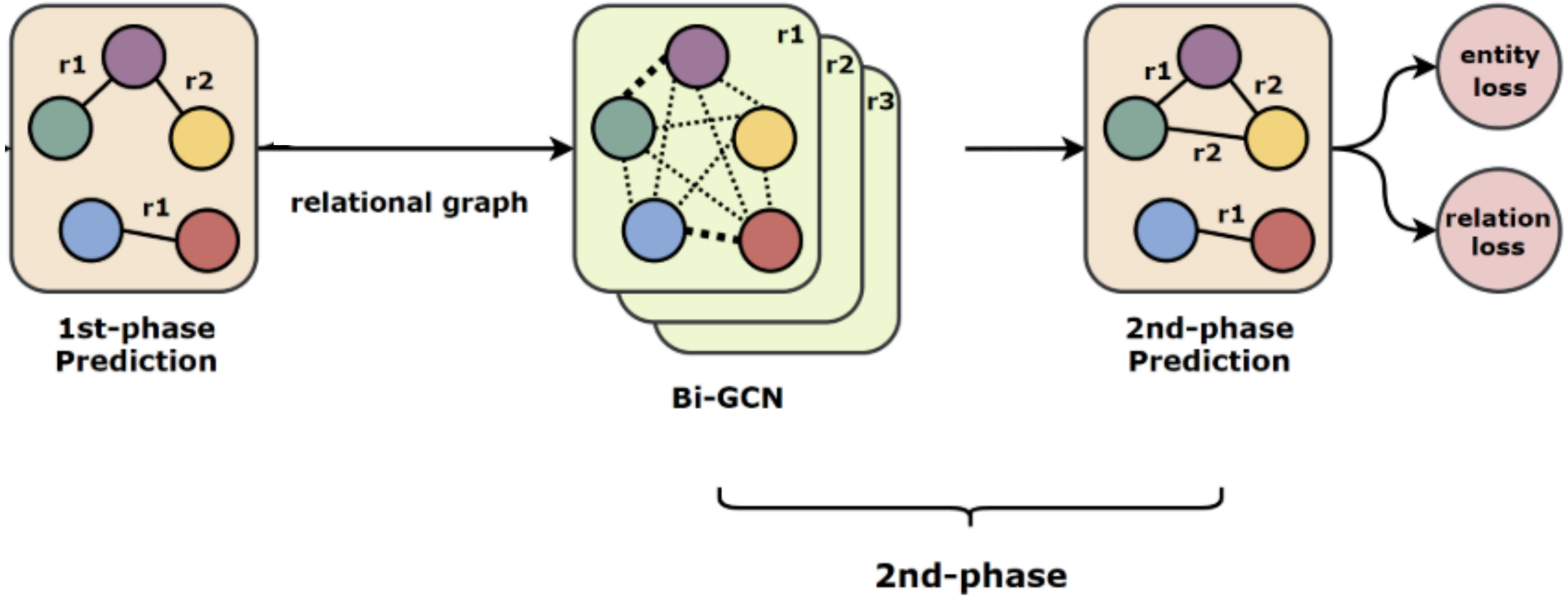
$$P_r(w1, w2) = \text{Softmax}(\ S_{(w1,r,w2)}\ )$$

1st-phase Prediction

- Categorical loss: eloss_1p, rloss_2p
- Relation Distribution of each word pair

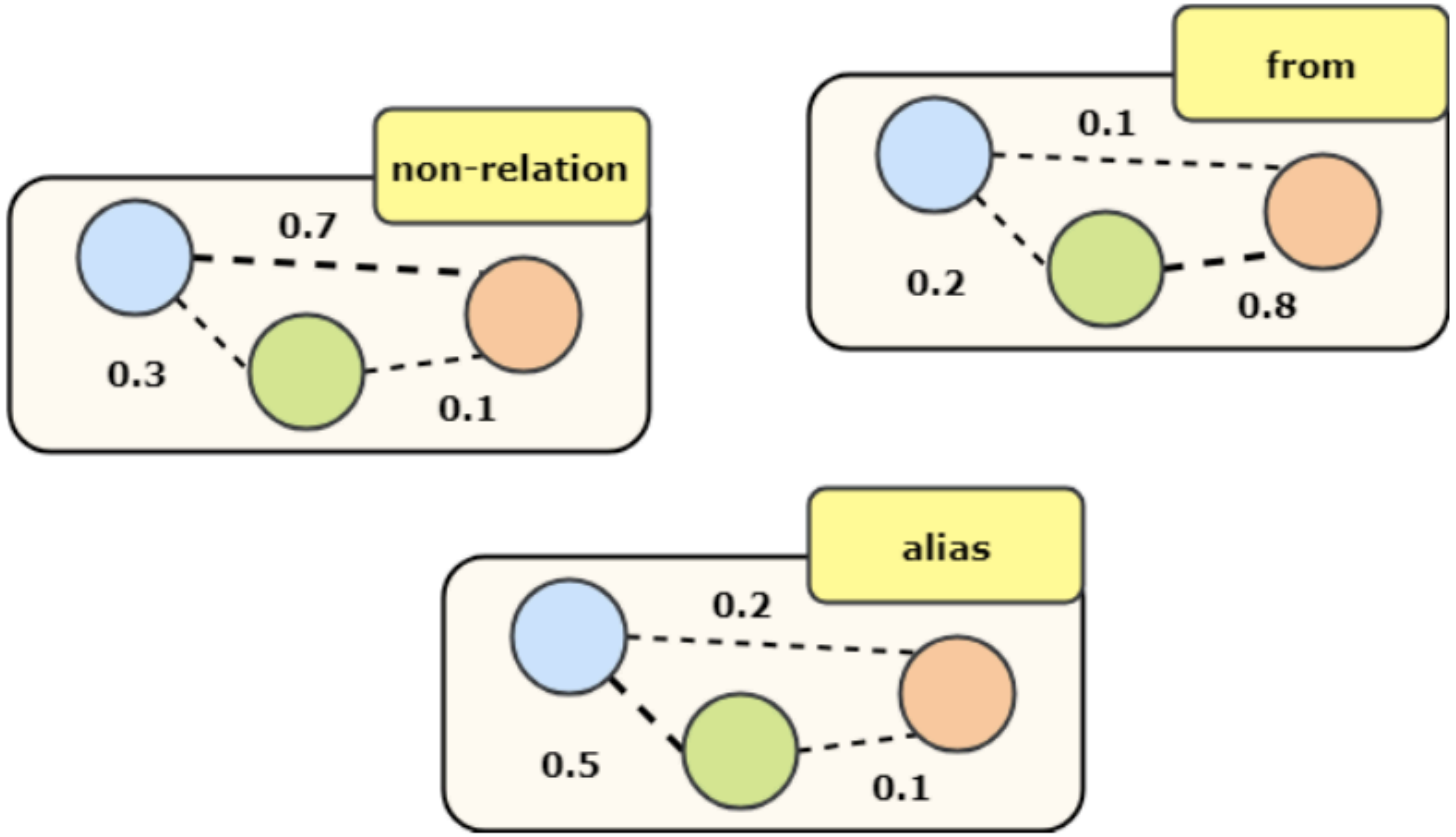
2nd-phase Prediction

- 1st-phase Node Embeddings + Relational Graph ➡️ Bi-GCN ➡️ Prediction



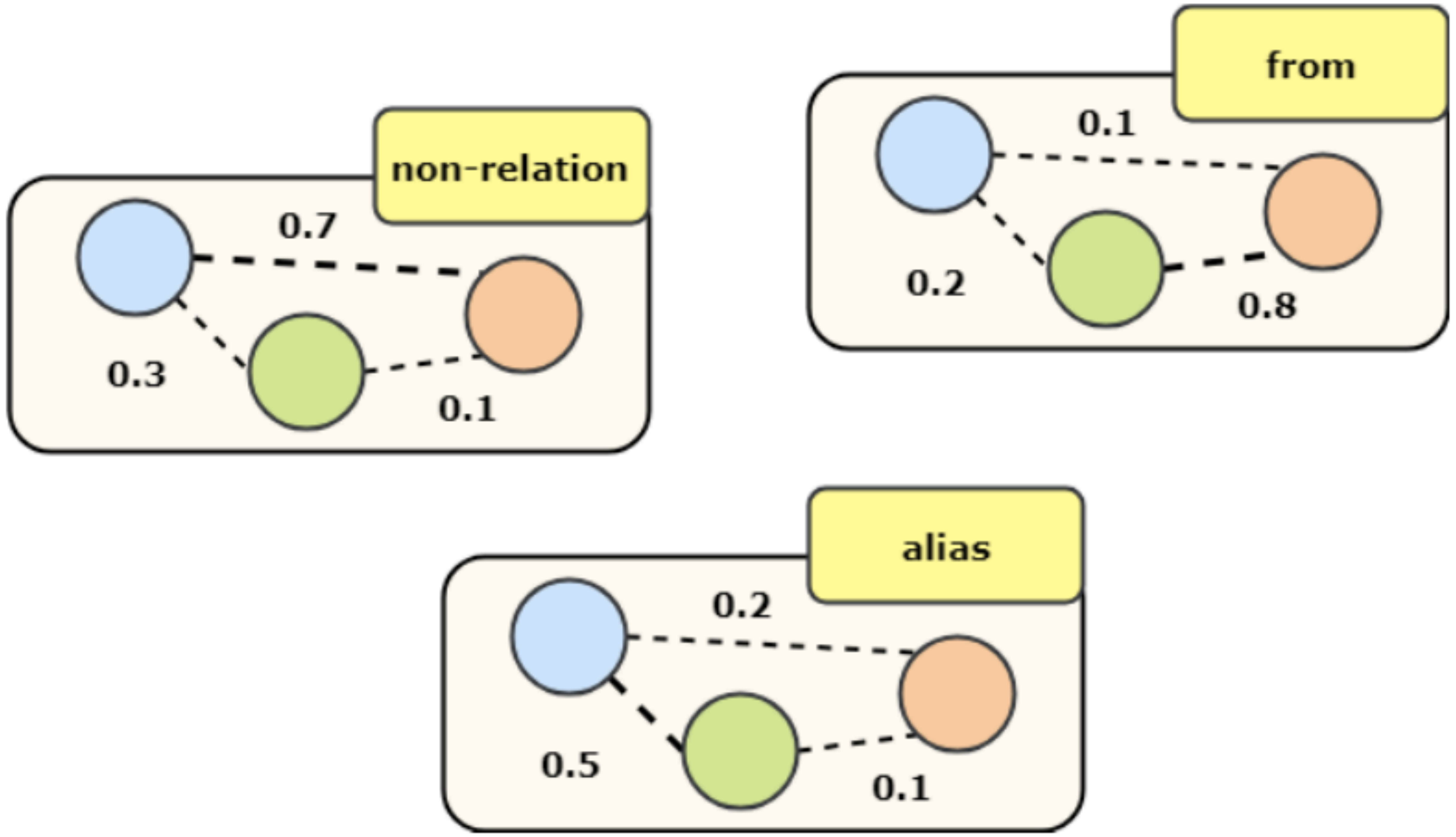
2nd-phase Prediction

- Relation-weighted Complete graph for each relation✅



2nd-phase Prediction

- Relation-weighted Complete graph for each relation✅



$$h_u^{l+1} = ReLU\left(\sum_{v \in V}\sum_{r \in R} P_r(u,v) \times \left(W_r^l h_v^l + b_r^l\right)\right) + h_u^l$$

Train & Relation Prediction

- Joint training

$$loss_{all} = (eloss_{1p} + rloss_{1p}) + \alpha\,(eloss_{2p} + rloss_{2p})$$

- Relation Prediction
  - Threshold inference: all word pairs of an entity mention pair are taken into account, choose the most probable class of them if proportion > threshold
  - Default: threshold = 0

Experiments

- Results on NYT and WebNLG datasets

Method	NYT			WebNLG		
	Precision	Recall	F1	Precision	Recall	F1
NovelTagging	62.4%	31.7%	42.0%	**52.5%**	19.3%	28.3%
OneDecoder	59.4%	53.1%	56.0%	32.2%	28.9%	30.5%
MultiDecoder	61.0%	56.6%	58.7%	37.7%	36.4%	37.1%
GraphRel_{1p}	62.9%	57.3%	60.0%	42.3%	39.2%	40.7%
GraphRel_{2p}	**63.9%**	**60.0%**	**61.9%**	44.7%	**41.1%**	**42.9%**

Experiments

- Case Study for GraphRel_1p and GraphRel_2p

Sentence	GraphRel_{1p}	GrapRel_{2p}
Agra Airport is in India where one of its leaders is Thakur.	(Agra Airport, location, India) (India, leader_name, Thakur)	(Agra Airport, location, India) (India, leader_name, Thakur)
In Italy, the capital is Rome and A.S. Gubbio 1910 is located there.	(Italy, captical, Rome)	(Italy, captical, Rome) (A.S. Gubbio 1910, ground, Italy)
Asam pedas (aka Asam padeh) is from the Sumatra and Malay Peninsula regions of Malaysia.	(Asam pedas, alias, Asam padeh) (Asam pedas, region, Malay Peninsula) (Asam pedas, country, Malaysia)	(Asam pedas, alias, Asam padeh) (Asam pedas, region, Malay Peninsula) (Asam padeh, region, Malay Peninsula) (Asam pedas, country, Malaysia) (Asam padeh, country, Malaysia)

Experiments

- Case Study for GraphRel_1p and GraphRel_2p

Sentence	GraphRel_{1p}	GrapRel_{2p}
Agra Airport is in India where one of its leaders is Thakur.	(Agra Airport, location, India) (India, leader_name, Thakur)	(Agra Airport, location, India) (India, leader_name, Thakur)
In Italy, the capital is Rome and A.S. Gubbio 1910 is located there.	(Italy, captical, Rome)	(Italy, captical, Rome) (A.S. Gubbio 1910, ground, Italy)
Asam pedas (aka Asam padeh) is from the Sumatra and Malay Peninsula regions of Malaysia.	(Asam pedas, alias, Asam padeh) (Asam pedas, region, Malay Peninsula) (Asam pedas, country, Malaysia)	(Asam pedas, alias, Asam padeh) (Asam pedas, region, Malay Peninsula) (Asam padeh, region, Malay Peninsula) (Asam pedas, country, Malaysia) (Asam padeh, country, Malaysia)

Conclusion

- RNNs and GCNs with dependency tree are complementary
  - RNNs: extract sequential features of text
  - GCNs: extract regional features of text
- GCNs are effective tool for exploiting graph structure in end-to-end learning
  - Graph structure is very important to GCNs

Q & A