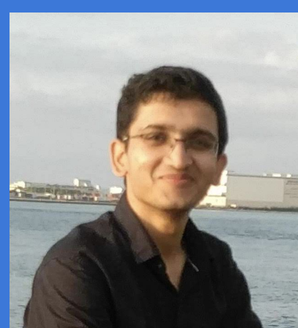
EMNLP 2019, Hong Kong

Tutorial Homepage: **github/svjan5/GNNs-for-NLP**

A Tutorial on **Graph Neural Networks for Natural Language Processing**



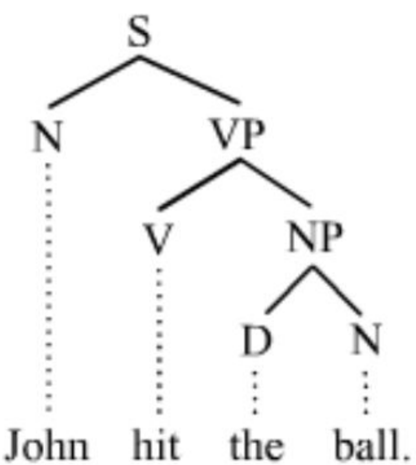
Shikhar Vashishth[1] shikhar@iisc.ac.in

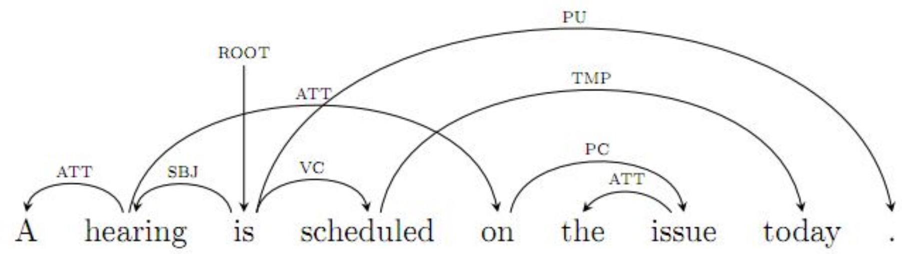
Y. Naganand[1] naganand@iisc.ac.in

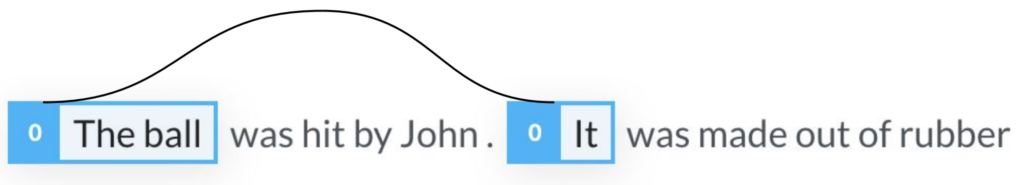
Partha Talukdar[1,2] ppt@iisc.ac.in

[1]Indian Institute of Science, Bangalore **[2]KENOME**

Graphs are everywhere in NLP







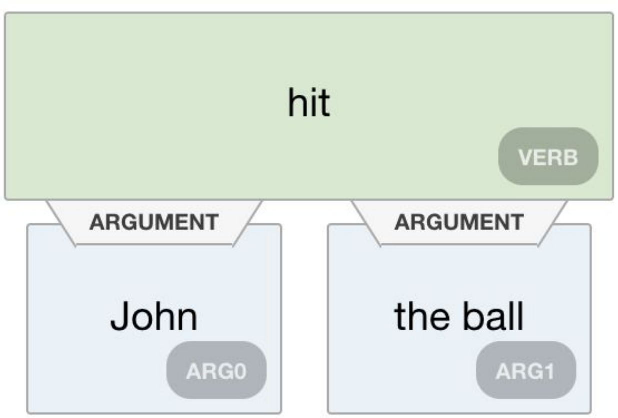


Image credits: Wikipedia, Allen NLP

Graphs are extensively used in NLP

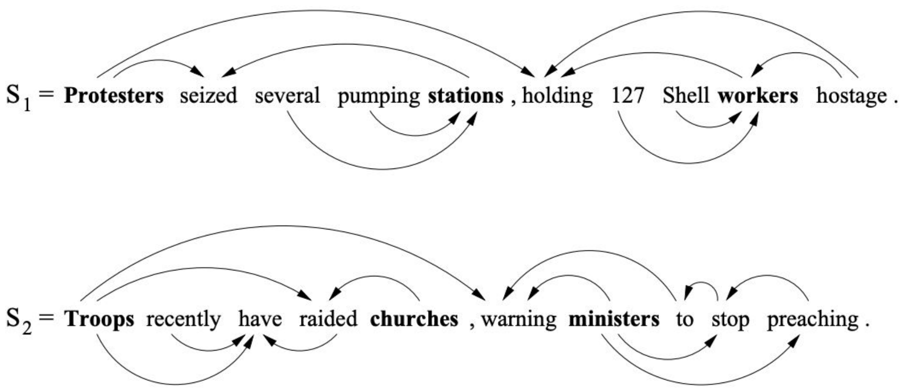
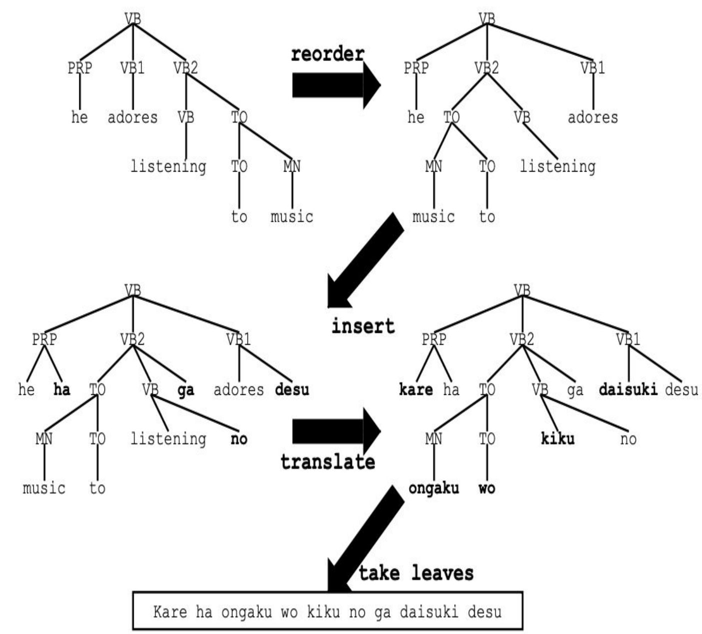


Figure 1: Sentences as dependency graphs.

Relation Instance	Shortest Path in Undirected Dependency Graph
S_1: protesters AT stations	**protesters** $\longrightarrow$ seized $\longleftarrow$ **stations**
S_1: workers AT stations	**workers** $\longrightarrow$ holding $\longleftarrow$ protesters $\longrightarrow$ seized $\longleftarrow$ **stations**
S_2: troops AT churches	**troops** $\longrightarrow$ raided $\longleftarrow$ **churches**
S_2: ministers AT churches	**ministers** $\longrightarrow$ warning $\longleftarrow$ troops $\longrightarrow$ raided $\longleftarrow$ **churches**

Table 1: Shortest Path representation of relations.

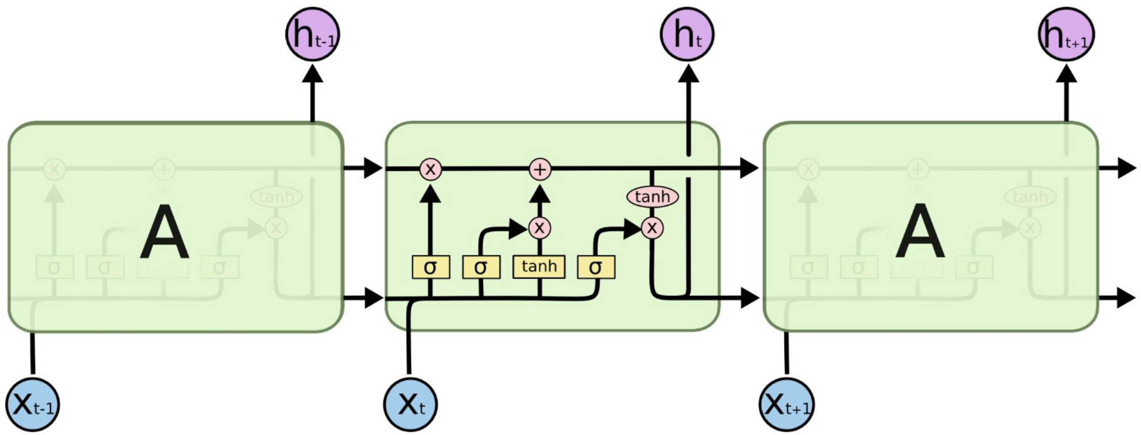
[Bunescu and Mooney, 2005]



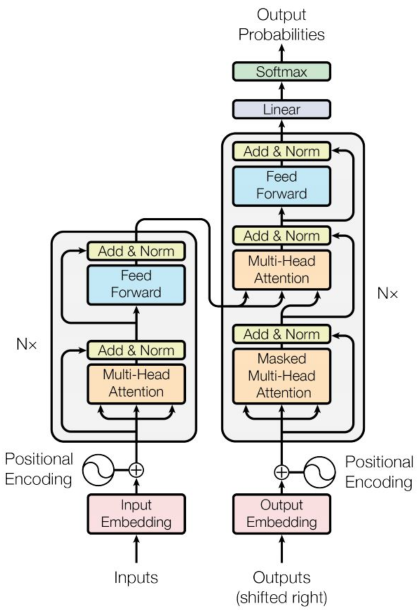
Syntax-based MT
[Yamada and Knight, 2001]

and many more ...

Deep Learning in NLP



Recurrent Networks

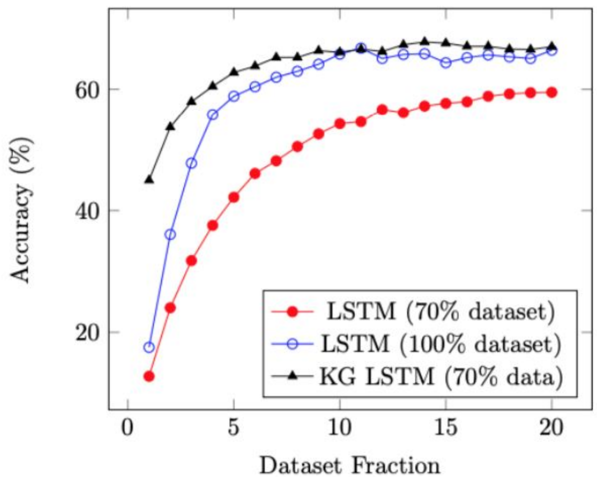


Transformers

It is not clear how to incorporate graph structures.

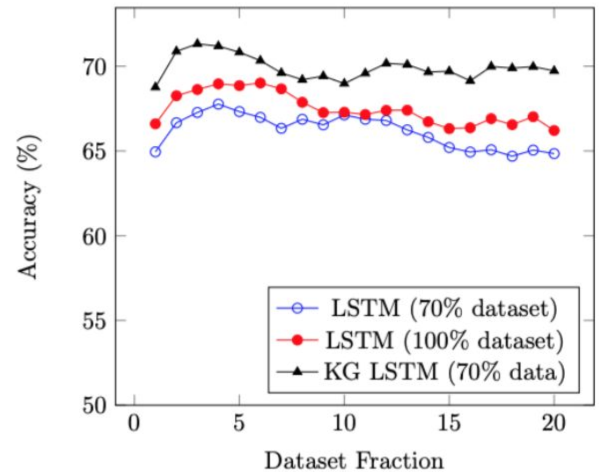
Image credits: colah.github.io, Google

Why Deep Learning over Graph for NLP?



(a)

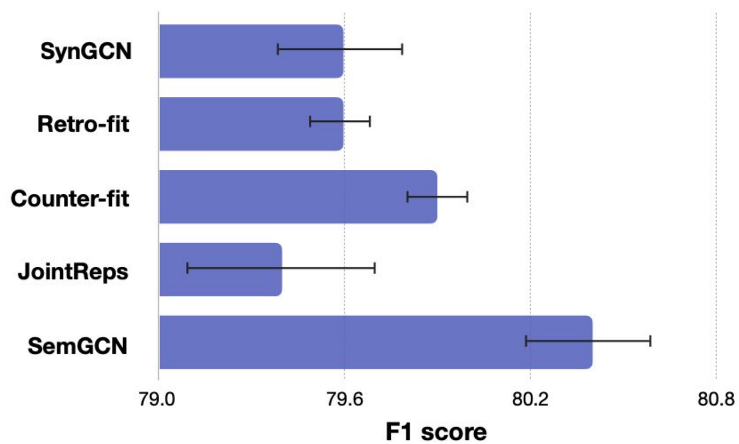
News20



(c)

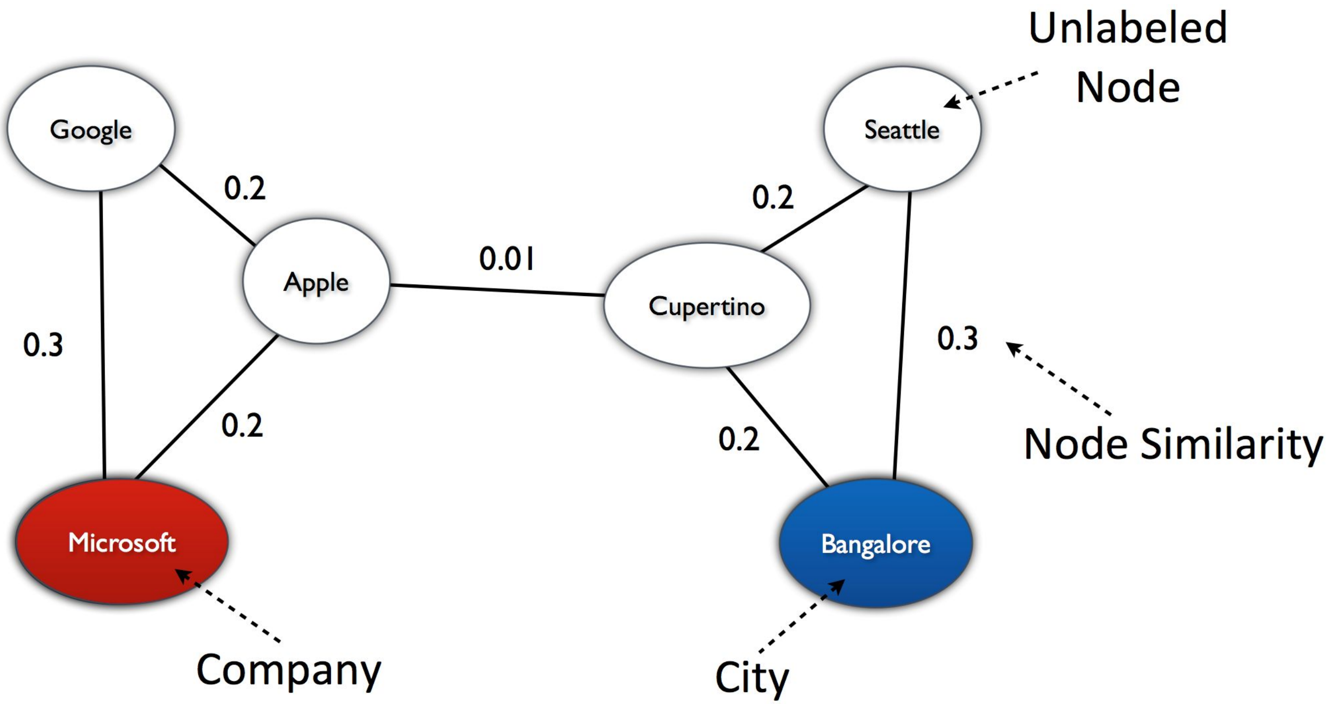
SNLI

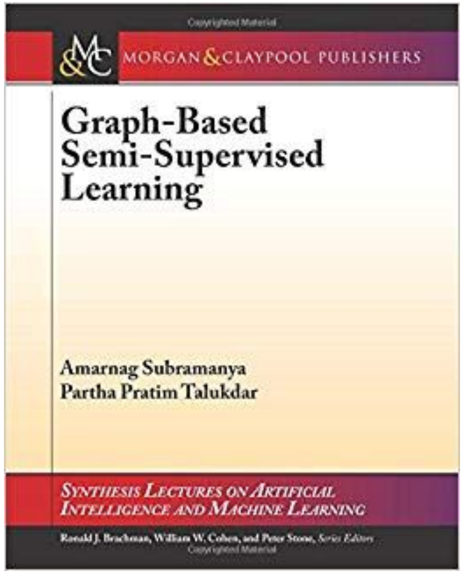
[Annervaz et al., NAACL 2018]



[Vashishth et al., ACL 2019]

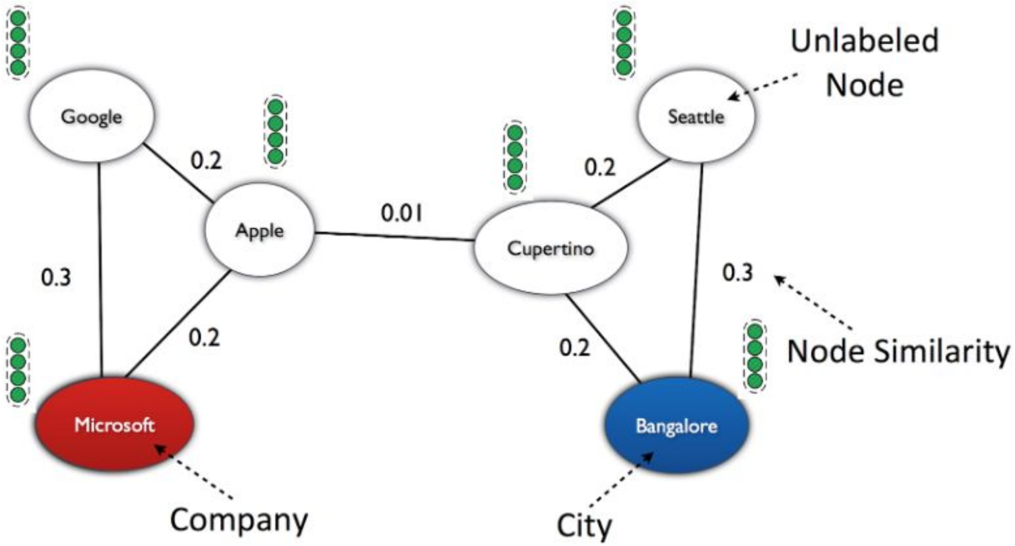
Graph-based Semi-supervised Learning (GraphSSL / LP)





ACL 2012 Tutorial: http://graph-ssl.wikidot.com/

GNNs vs GraphSSL



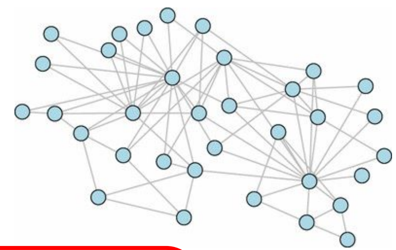
GNNs vs **GraphSSL/LP**

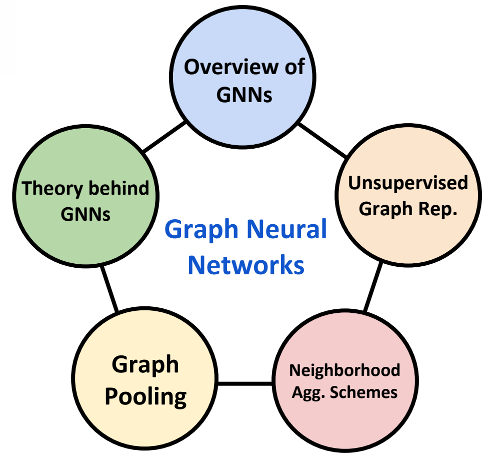
- GNNs learn embeddings (nodes, graphs etc)
- GNNs can handle relations beyond similarity
- Implicit regularization
- GNNs tend to be more effective

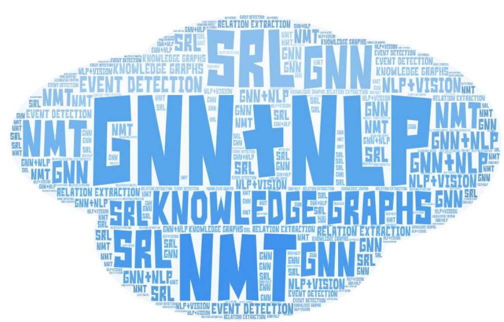
Method	Citeseer	Cora	Pubmed	NELL
LP	45.3	68.0	63.0	26.5
GCN	**70.3**	**81.5**	**79.0**	**66.0**

Tutorial Outline

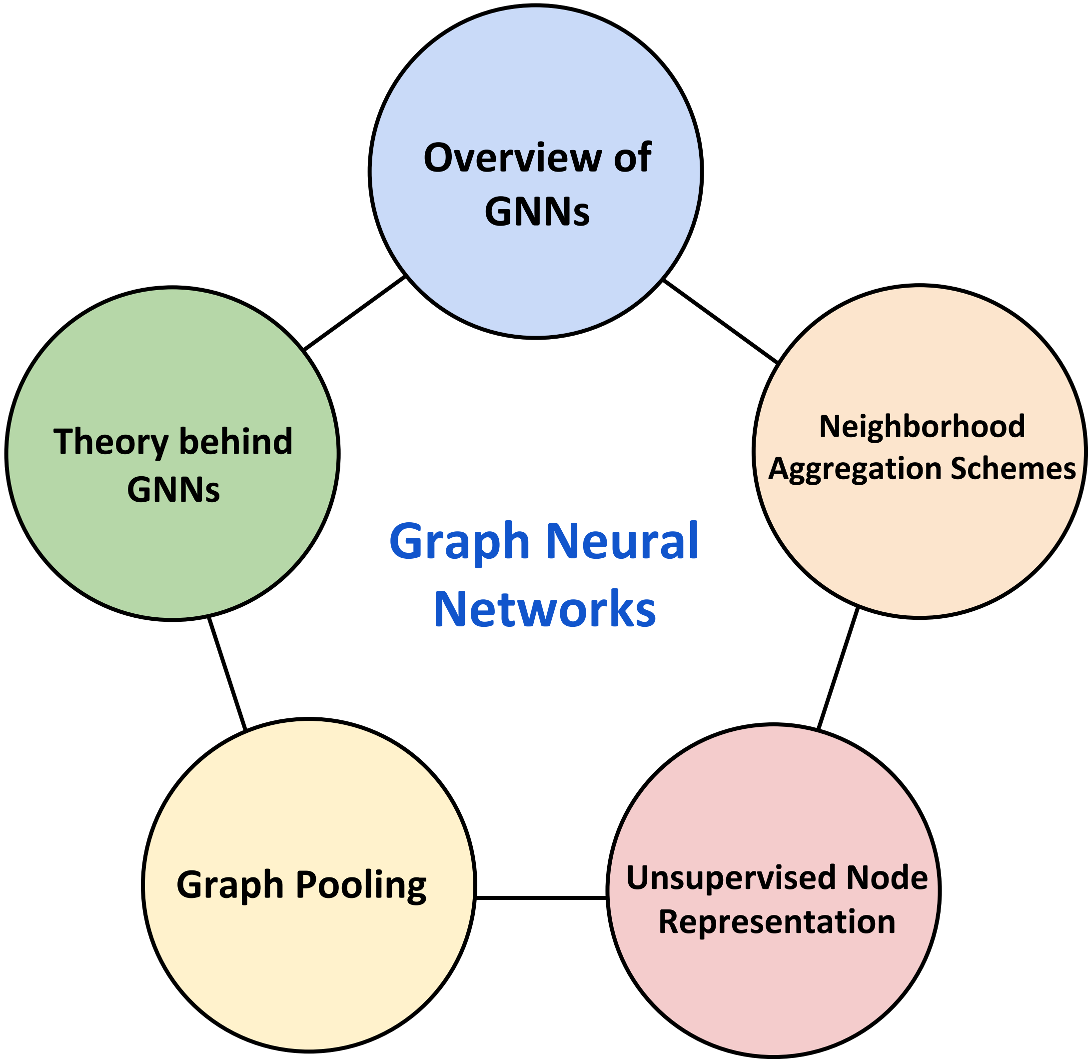
- **Introduction**
  - ✓ Motivation
  - ✓ GNN Foundation
- **Methods**
  - Overview of GNNs
  - Graph Pooling
  - Neighborhood Aggregation in GNNs
  - Unsupervised Learning using GNNs
  - Theoretical background
- **Implementing GCNs**
- **Applications**
  - Semantic Role Labeling, NMT
  - Text Classification, Extraction
  - Knowledge Graphs
  - Vision + NLP
- **Open Problems and Conclusion**



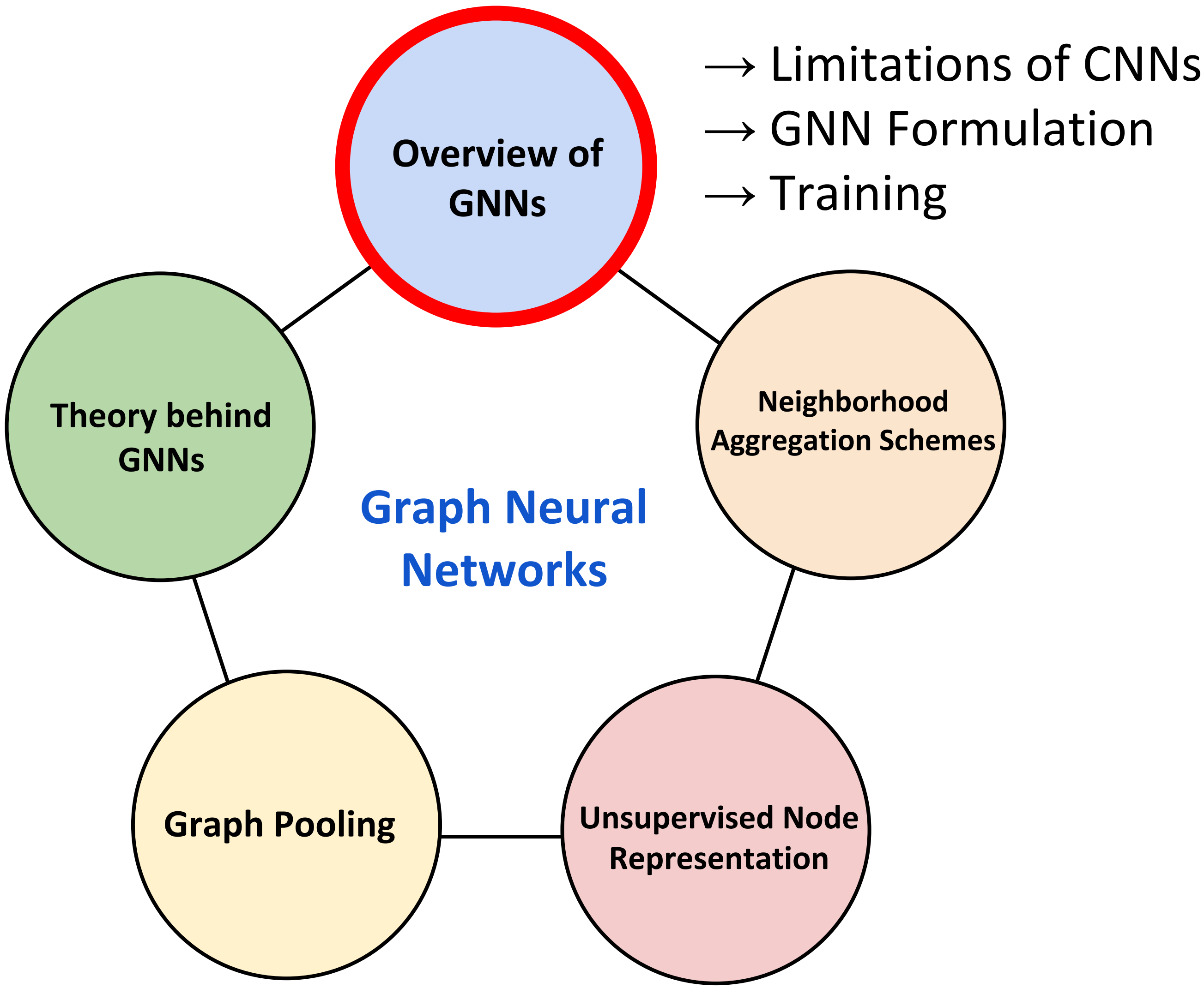




Graph Neural Networks

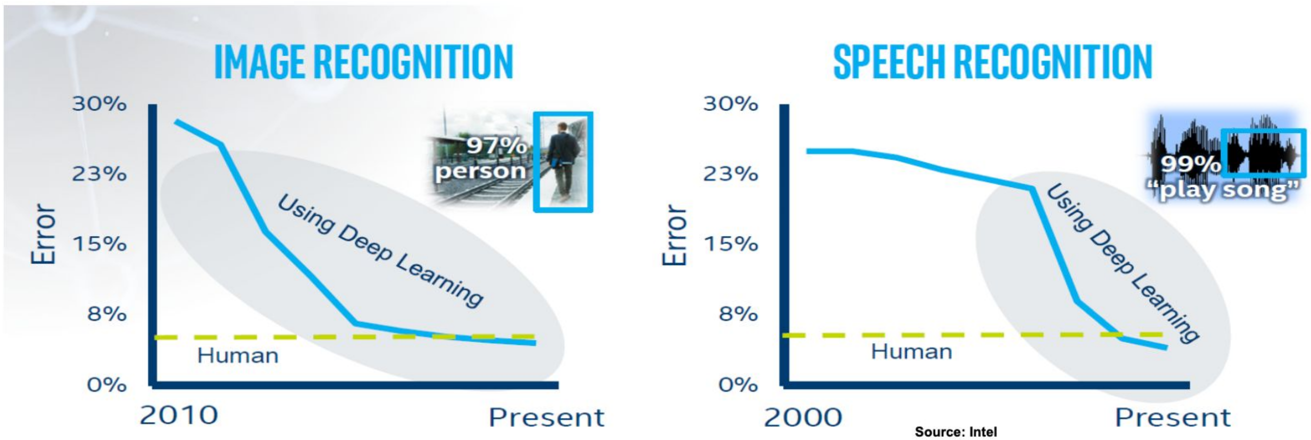


Graph Neural Networks

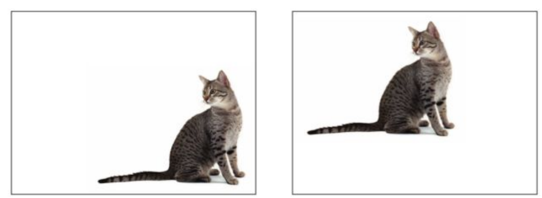


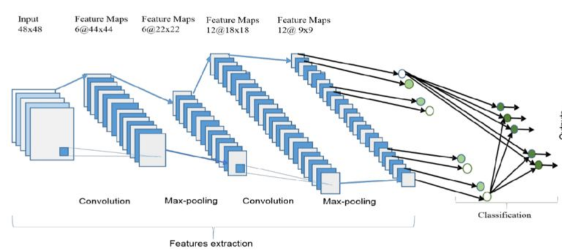
Convolutional Neural Networks (CNNs)

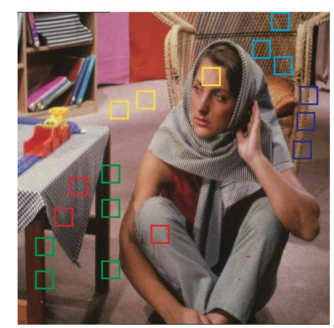
- CNNs’ BIG impact!



- Key properties of CNNs:
    - Translation Invariance
    - Localized filters in space
    - Multiple Layers

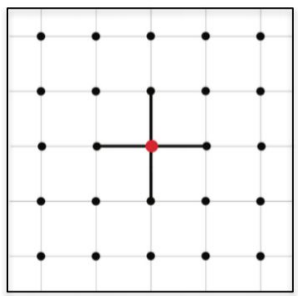




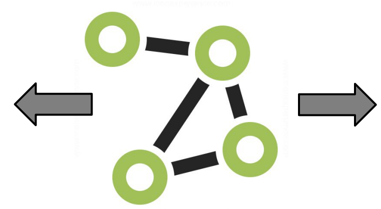


How CNNs for Graphs?

- Translation



$$f(x + a, y + b)$$



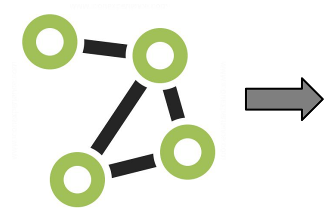
- Downsampling (Pooling)

2	1	3	1
1	0	1	**4**
0	6	**9**	5
7	1	4	1

$\rightarrow$

Max Pool

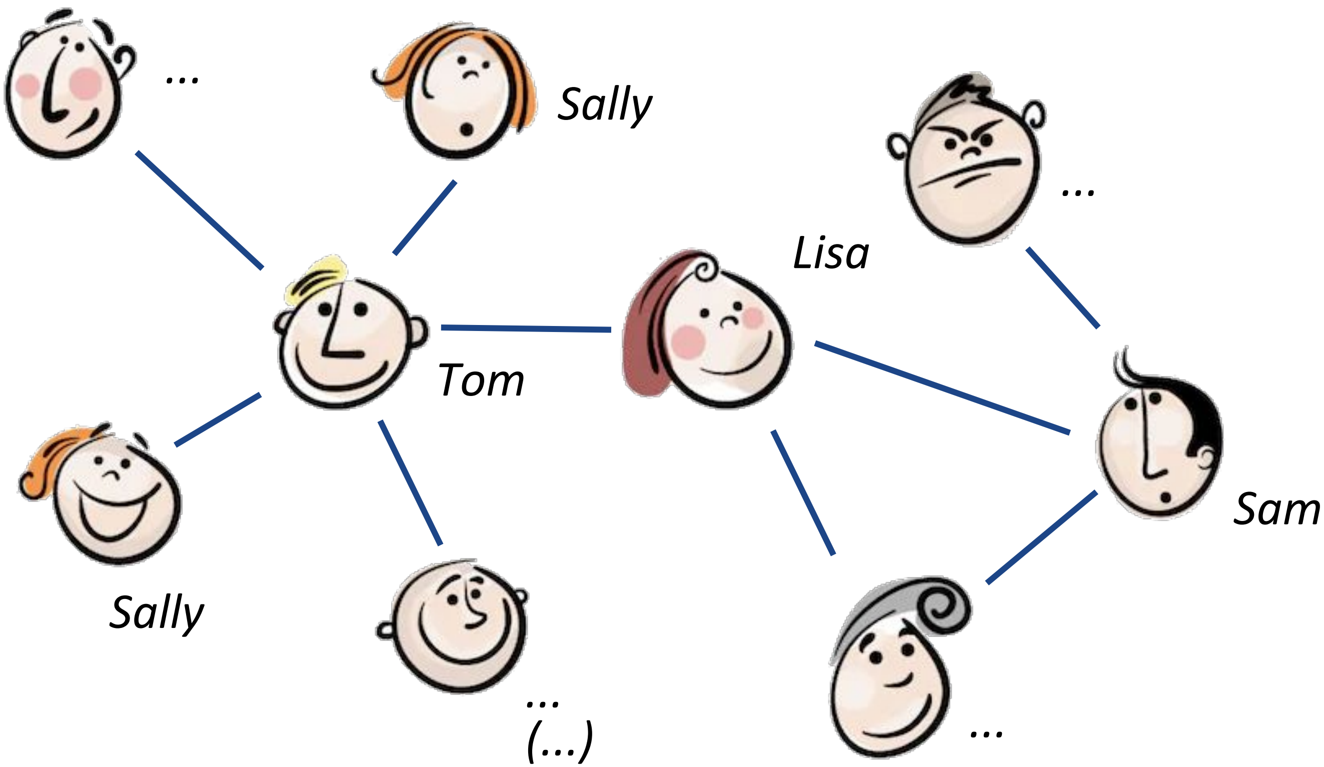
2	4
7	9



?

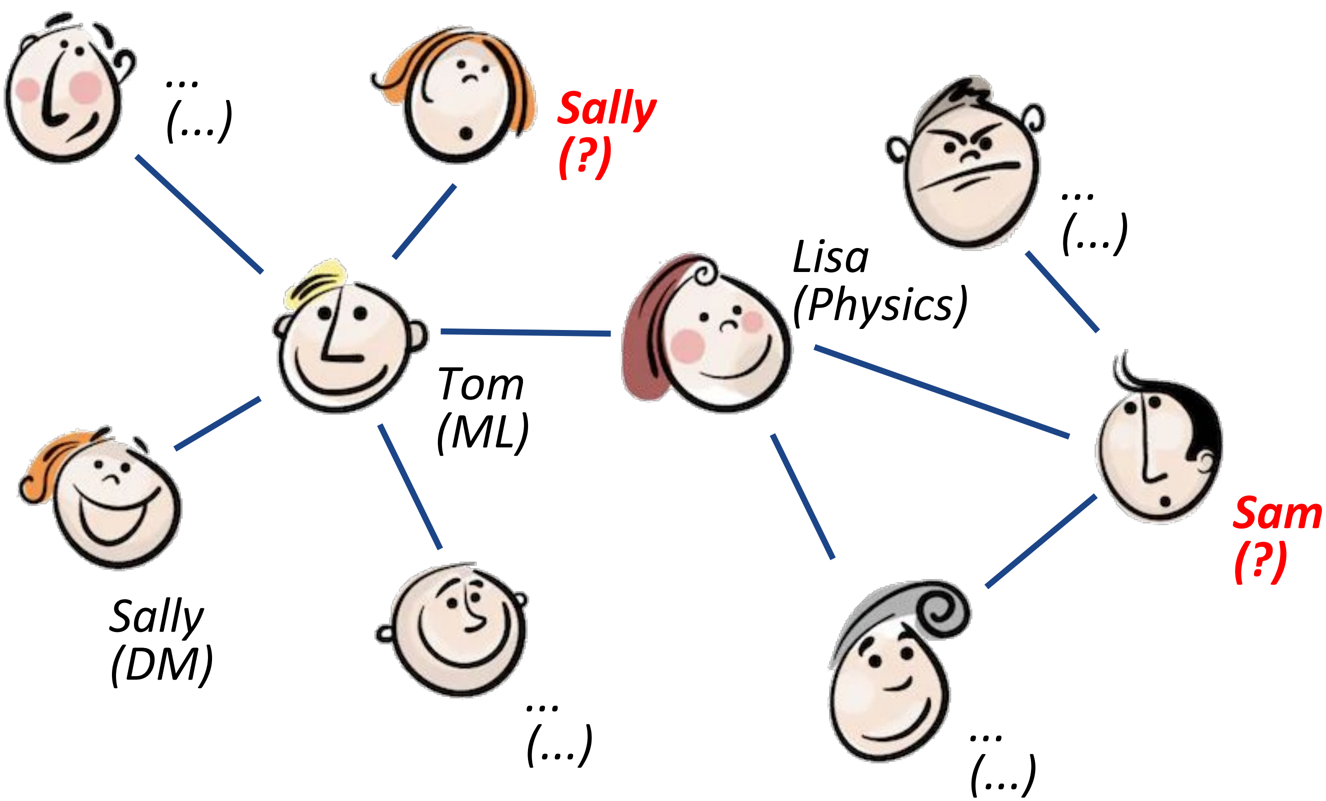
Motivating Example

- **Co-authorship Network**
  - **Nodes**: Authors, **Edges**: Co-authorship



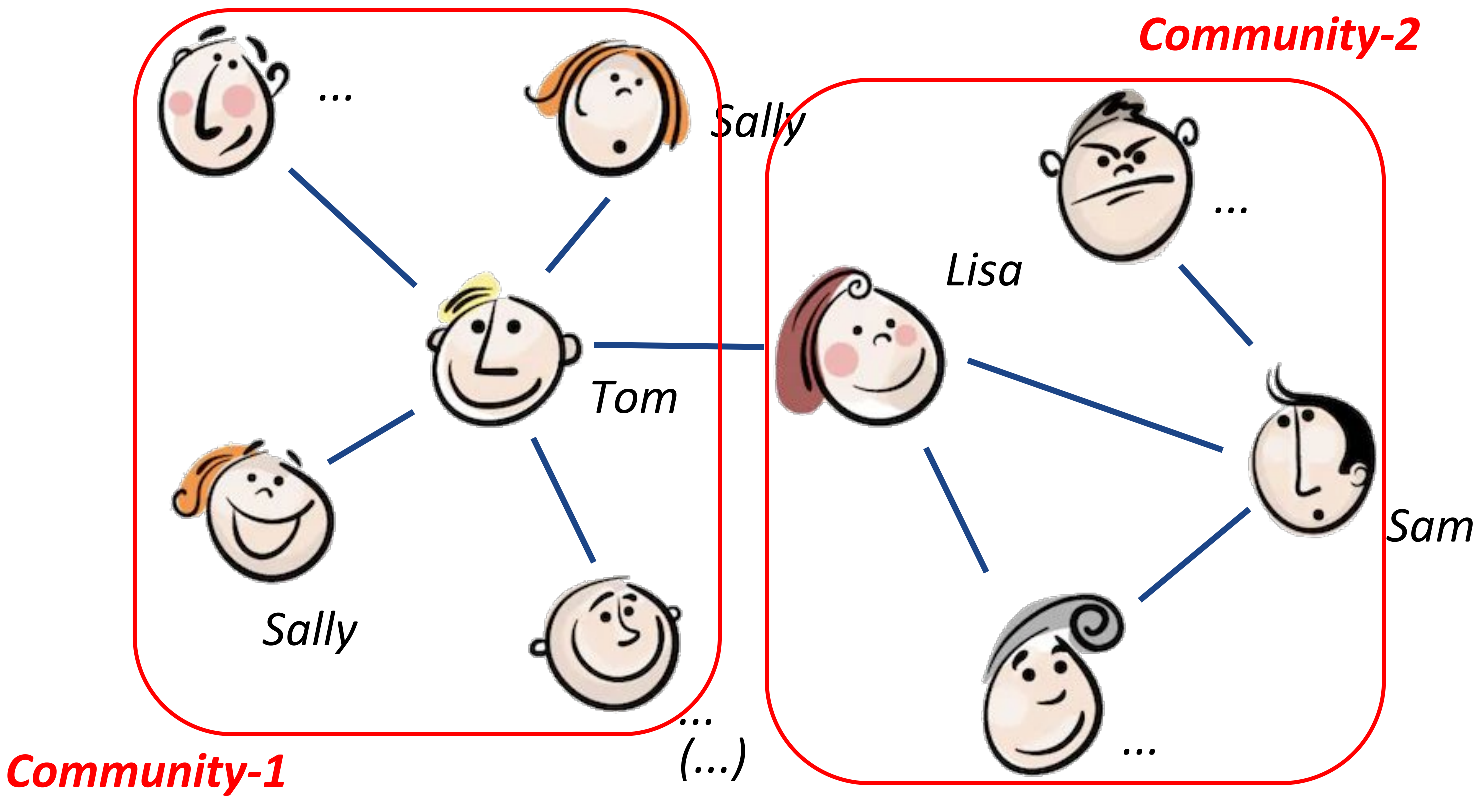
Motivating Example

- **Node Classification:** (Semi-supervised Learning)
  - Predict research area of unlabeled authors



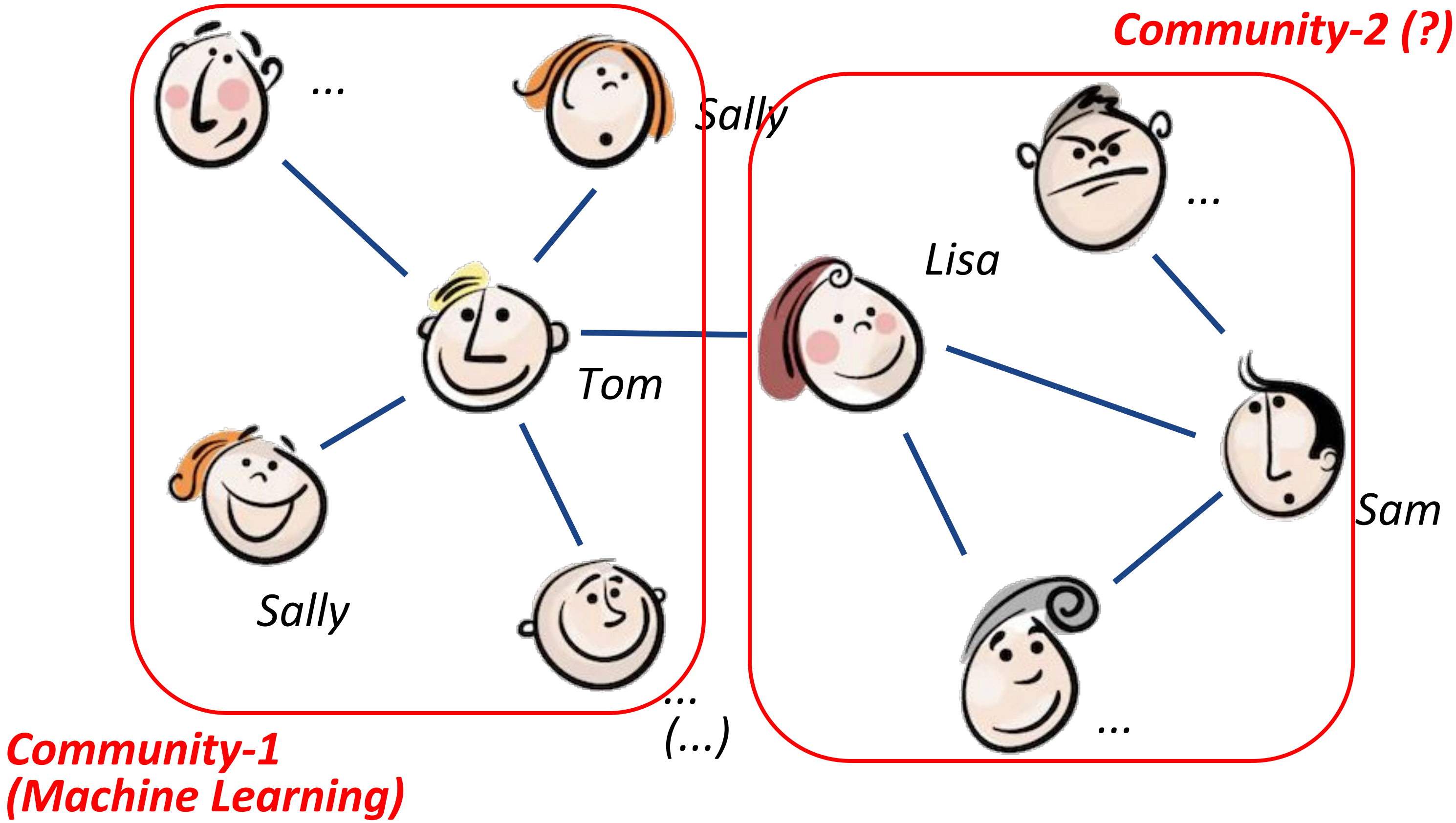
Motivating Example

- **Identify Communities:** (Unsupervised)
  - Grouping authors with similar research interests



Motivating Example

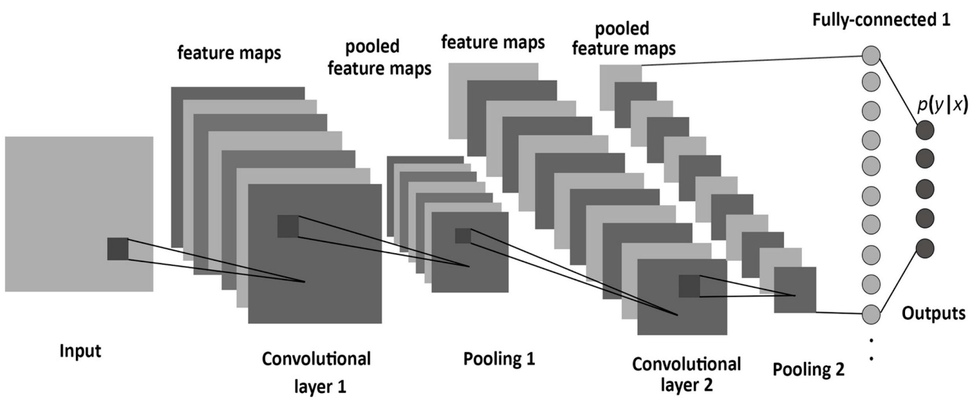
- **Graph Classification:** (Supervised)
  - Identifying class of each community.

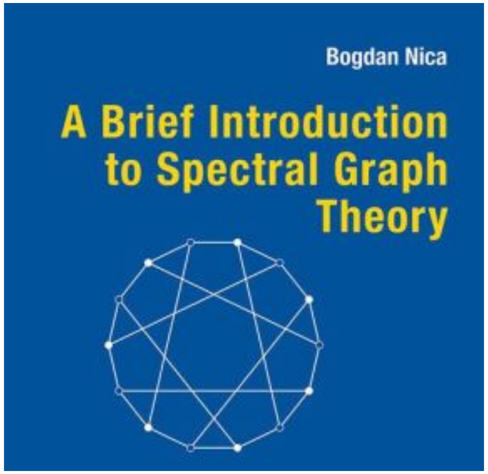


Graph Convolutional Networks (GCN)

- GCN formulation by [Kipf et al., ICLR 2016]

[Derivation in later part of the tutorial]





$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$

Graph Convolutional Networks

- GNN formulation by [Kipf et al., ICLR 2016]

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \boxed{W}\,\boxed{x_u} + b\right), \quad \forall v \in \mathcal{V}.$$

Filter matrix (Model-parameter)

Initial Node Features

Nodes == Words $\longrightarrow$ Word2vec Embeddings
Nodes == Authors $\longrightarrow$ 0/1 value indicating frequently used keywords
No features $\longrightarrow$ One-hot vector (length = #Nodes)

Graph Convolutional Networks

- GNN formulation by [Kipf et al., ICLR 2016]

$$h_v = \boxed{f}\left(\boxed{\frac{1}{|\mathcal{N}(v)|}} \boxed{\sum_{u\in\mathcal{N}(v)} \boxed{W}\boxed{x_u}} + \boxed{b}\right), \quad \forall v \in \mathcal{V}.$$

Normalization

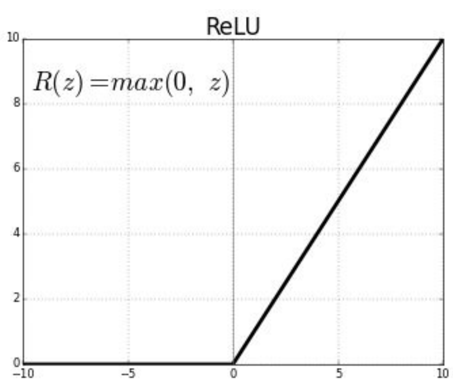
Neighborhood Aggregation

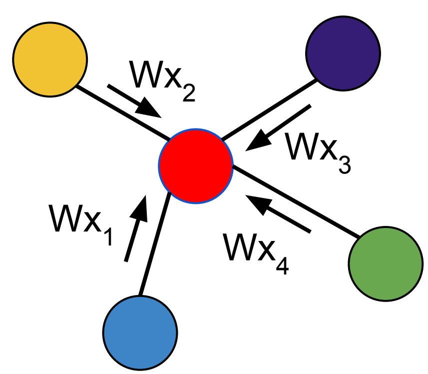
Bias (Model-parameter)

Non-Linearity

Filter matrix (Model-parameter)

Initial Node Features





Graph Convolutional Networks

- GNN formulation by [Kipf et al., ICLR 2016]

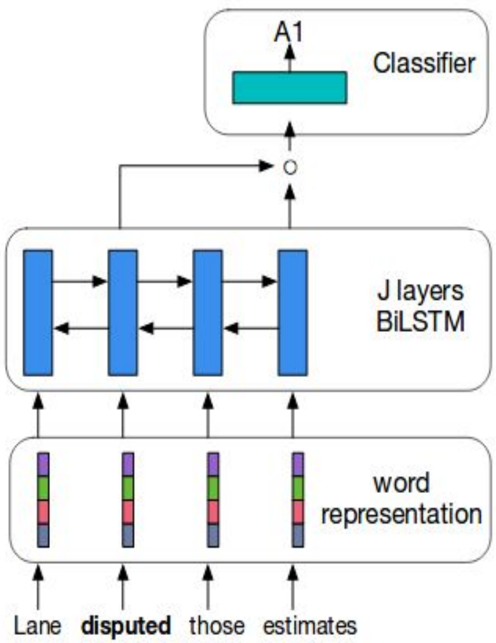
$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$

The above formulation is restricted to capturing just 1-hop of nodes

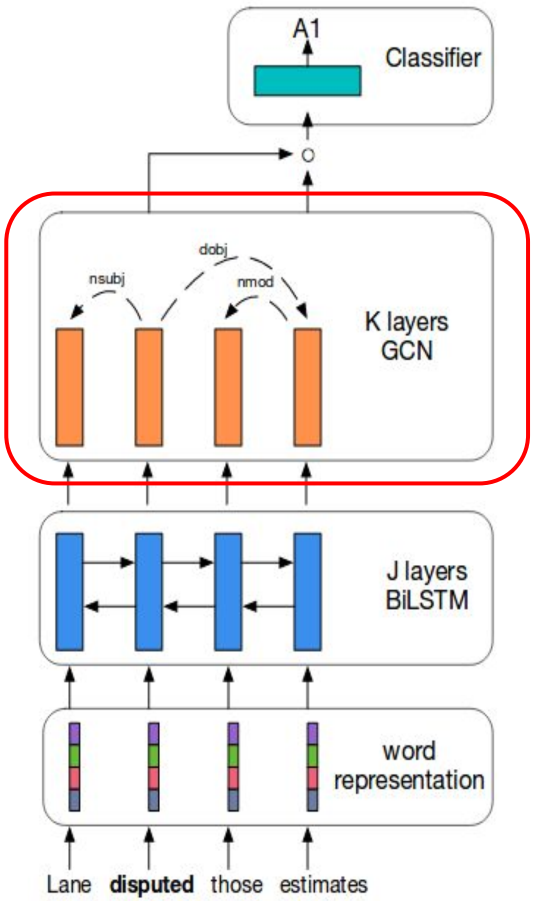
- Stacking K-GNN Layers for capturing K-hop nbd

$$h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W^k \boxed{h_u^k} + b^k\right), \quad \forall v \in \mathcal{V}.$$

Example: GNNs for Semantic Role Labeling



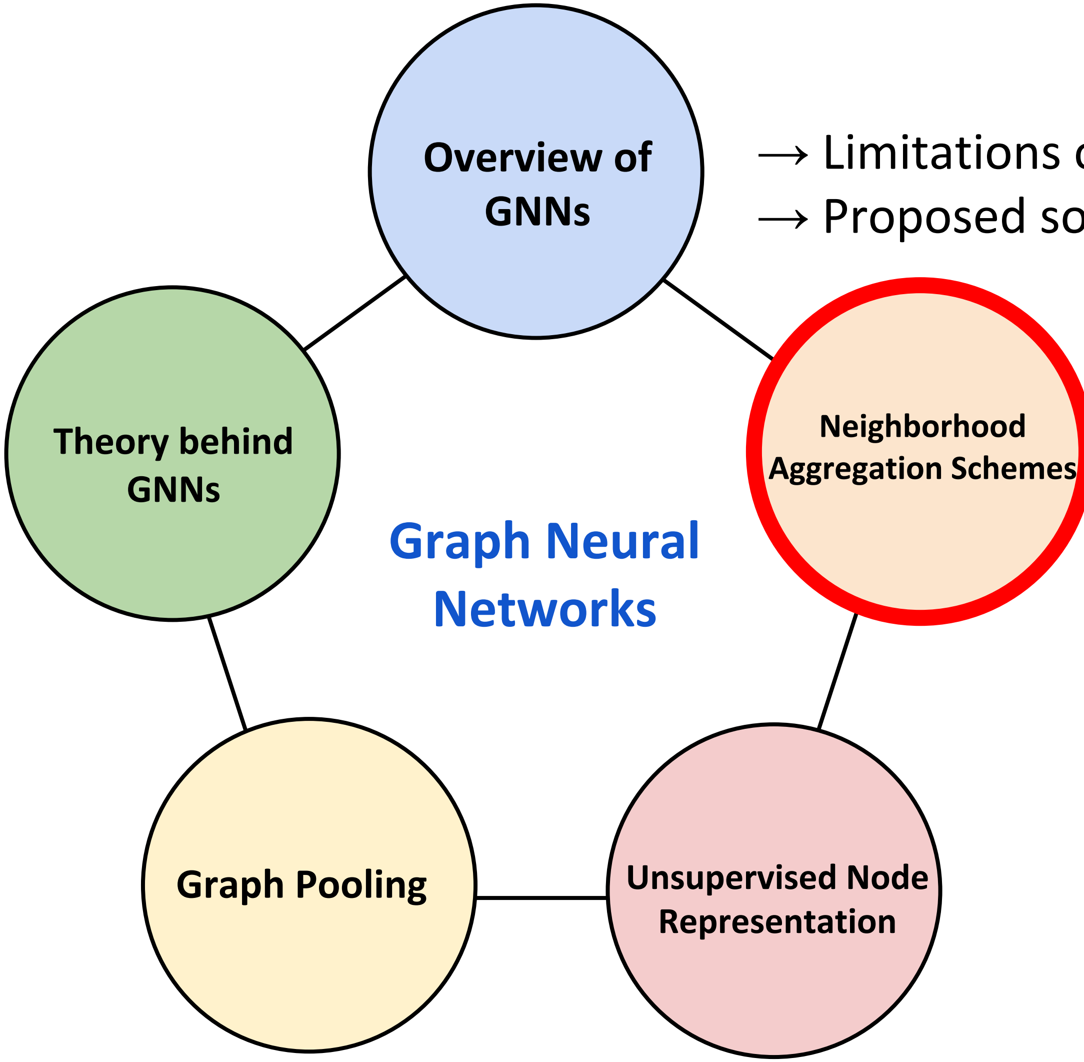
Standard Deep Learning Architecture for NLP problems
(above is for Semantic-Role Labeling (SRL))



Model with GCN as part of the network

GCN weights are trained based on the final objective

Graph Neural Networks

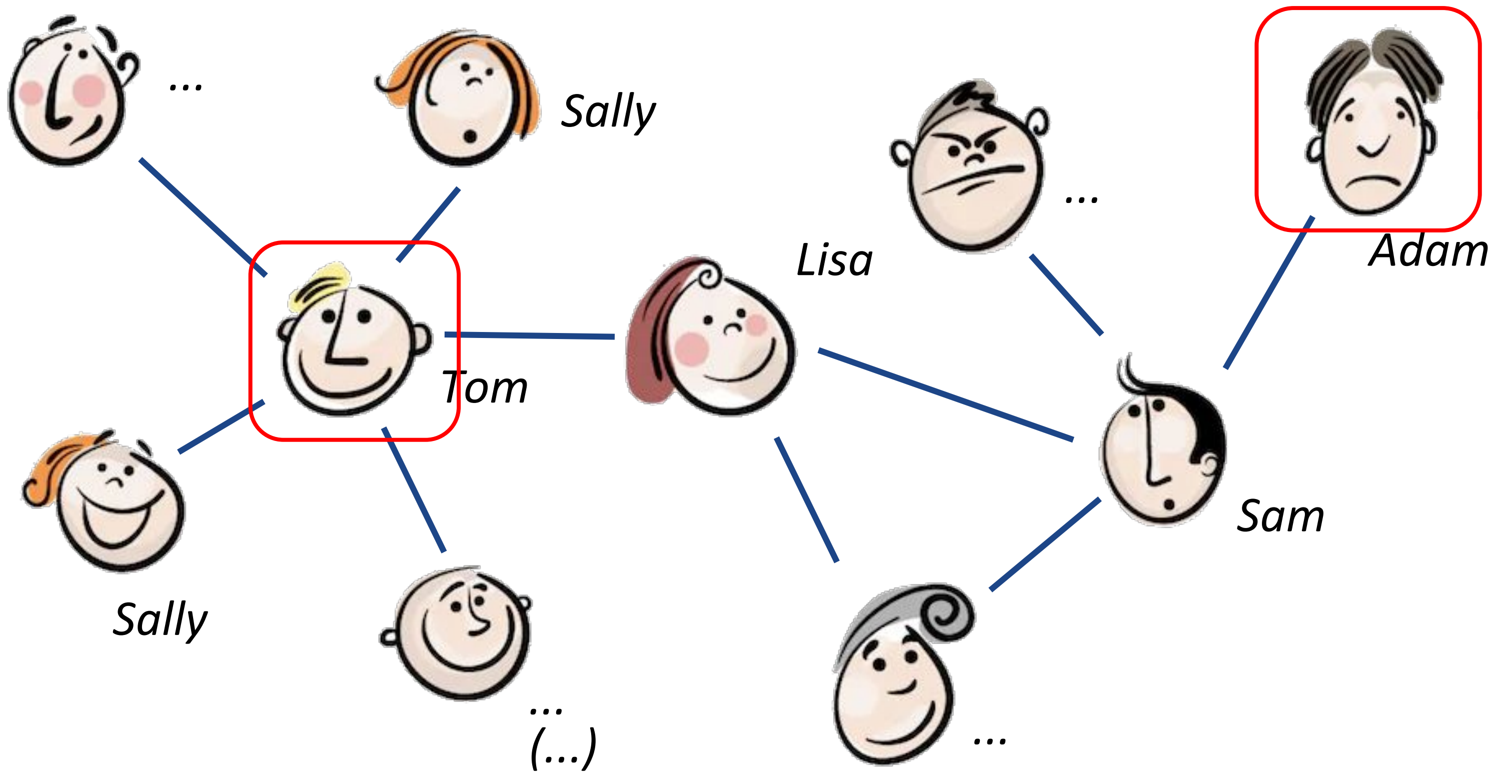


→ Limitations of existing schemes

→ Proposed solutions

Motivating Example: Co-authorship Network

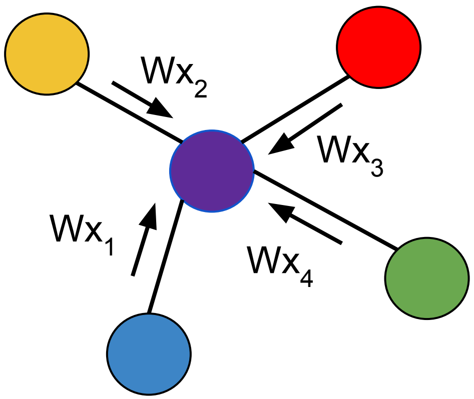
- Handling heterogeneous graphs



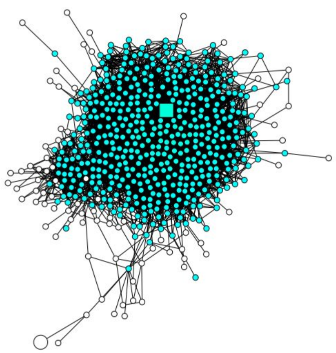
Neighborhood Aggregations in GCNs

- Standard GCN neighborhood aggregation

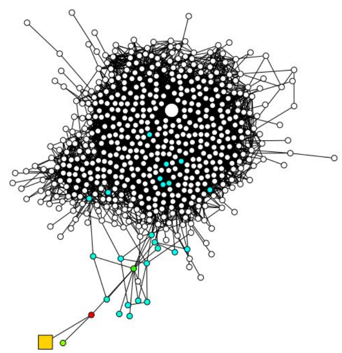
$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$



- No restriction on influence neighborhood



Hub Node



Leaf Node

- Methods:
  - Graph Attention Networks (GAT)
  - Confidence-based GCN (ConfGCN)

Graph Attention Networks (Velickovic´ et al. ICLR '18)

- Uses self-attention for GCNs

Input features: $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$

Importance of v_j to v_i : $e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j) \quad \alpha_{ij} = \mathrm{softmax}_j(e_{ij})$

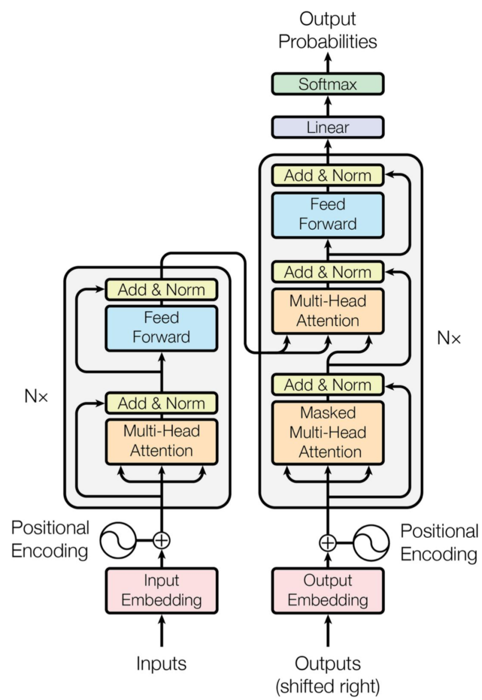


Figure 1: The Transformer - model architecture.

Utilizing Transformer's self-attention idea for improving GNNs

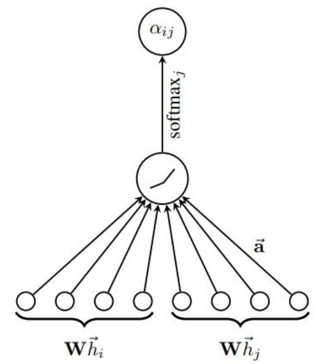
Graph Attention Networks (Velickovic´ et al. ICLR '18)

- Uses self-attention for GCNs

Input features: $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$

Importance of v_j to v_i : $e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j) \quad \alpha_{ij} = \mathrm{softmax}_j(e_{ij})$

$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$



x K

K = # Heads

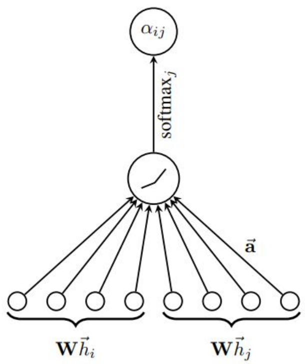
Graph Attention Networks (Velickovic´ et al. ICLR '18)

- Uses self-attention for GCNs

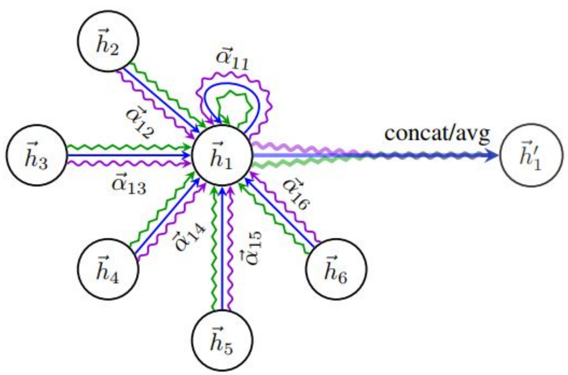
Input features: $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$

Importance of v_j to v_i : $e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j) \quad \alpha_{ij} = \mathrm{softmax}_j(e_{ij})$

$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$



x K
K = # Heads



$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$ ←GAT Update (K=#heads)

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$ ←Final Layer

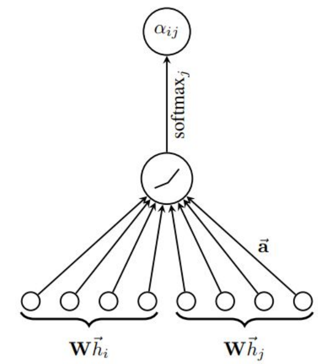
Graph Attention Networks (Velickovic´ et al. ICLR ‘18)

- Uses self-attention for GCNs

Input features: $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$

Importance of v_j to v_i : $e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j) \quad \alpha_{ij} = \mathrm{softmax}_j(e_{ij})$

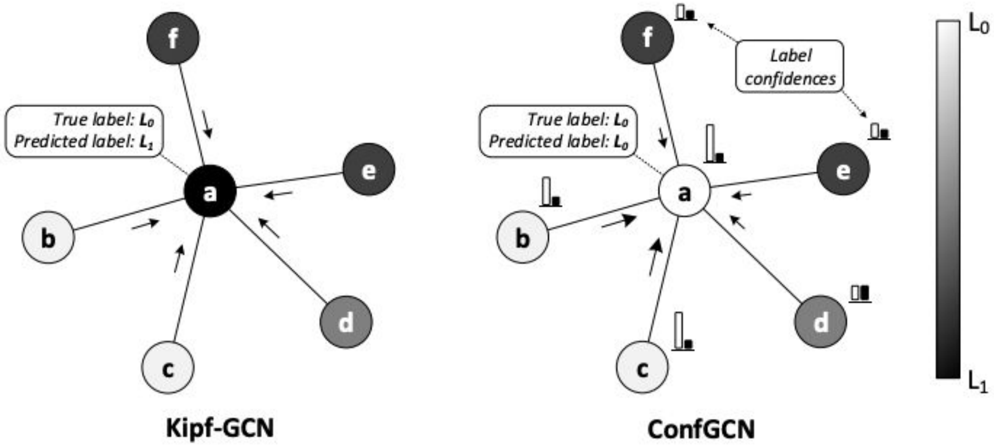
$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$



Method	Cora	Citeseer	Pubmed
GCN	81.5%	70.3%	**79.0%**
GAT	**83.0** ± 0.7%	**72.5** ± 0.7%	**79.0** ± 0.3%

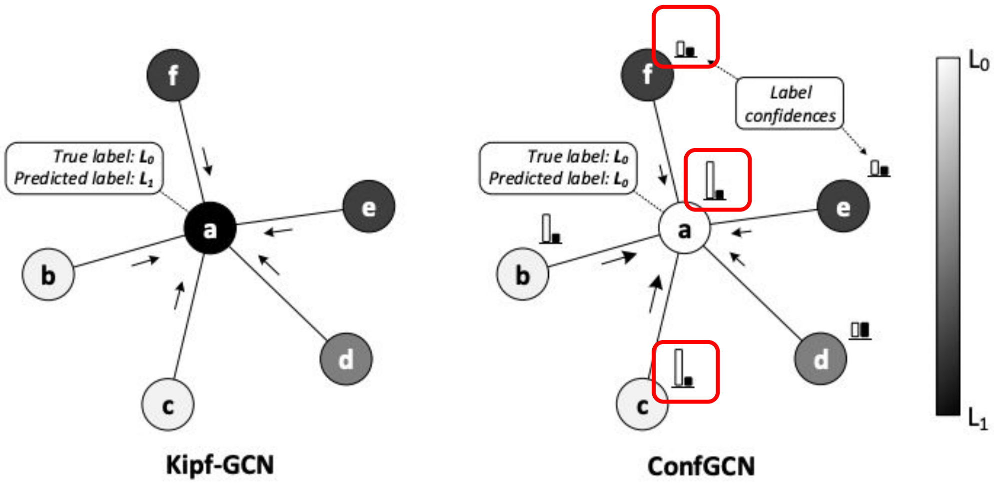
Confidence-based GCN [Vashishth et al., AISTATS '19]

- Comparison with standard GCN model



Confidence-based GCN [Vashishth et al., AISTATS '19]

- Comparison with standard GCN model



$\boldsymbol{\Sigma}_u^{-1} \downarrow \quad r_{uv} \downarrow$

$\boldsymbol{\Sigma}_u^{-1} \uparrow$

$(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)' \downarrow \quad r_{uv} \uparrow$

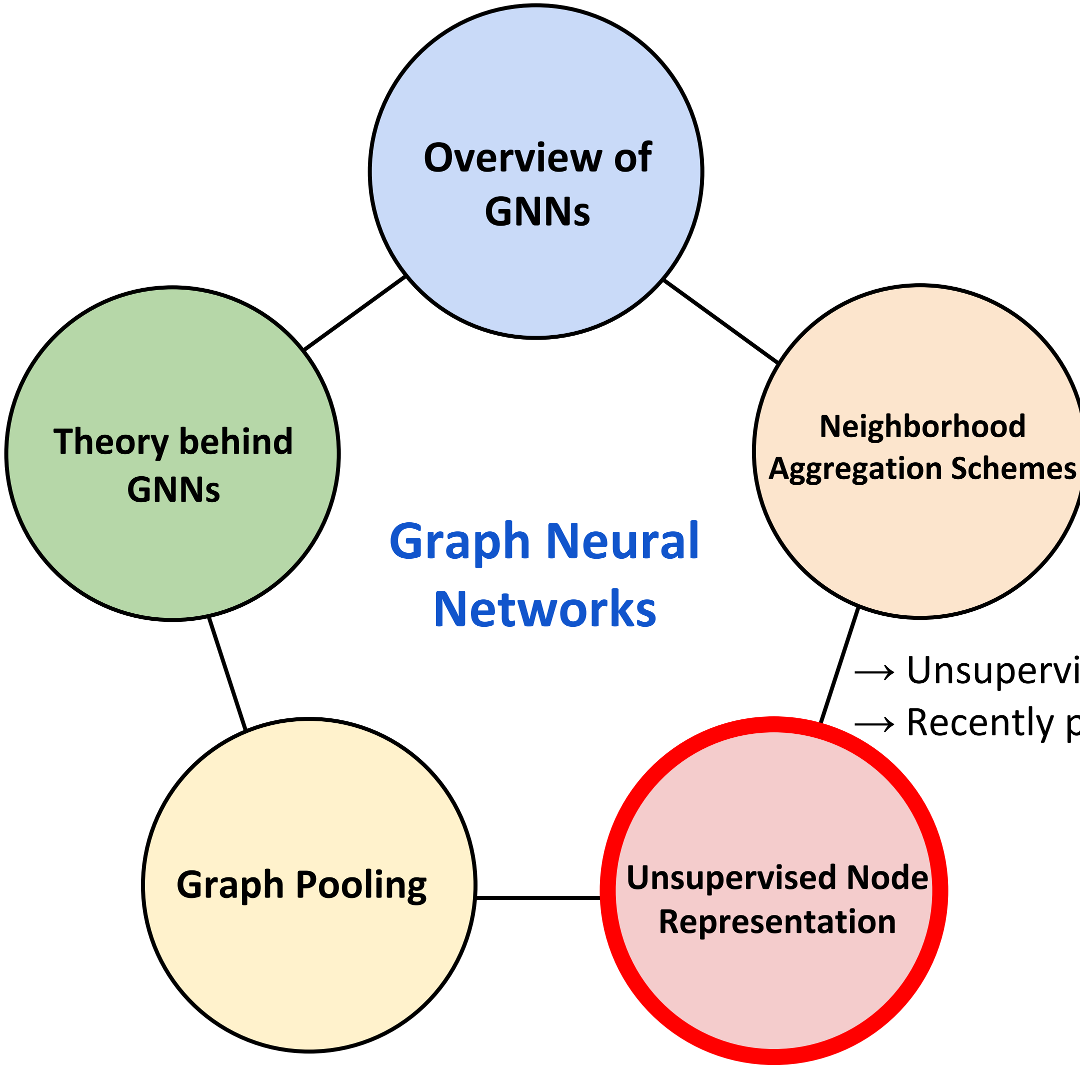
$(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)' \uparrow \quad r_{uv} \downarrow$

- Importance for a node is calculated as:

$$r_{uv} = \frac{1}{d_M(u,v)}. \qquad d_M(u,v) = (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v).$$

  - μ_u, μ_v are label distribution and Σ_u, Σ_v denote co-variance matrices.

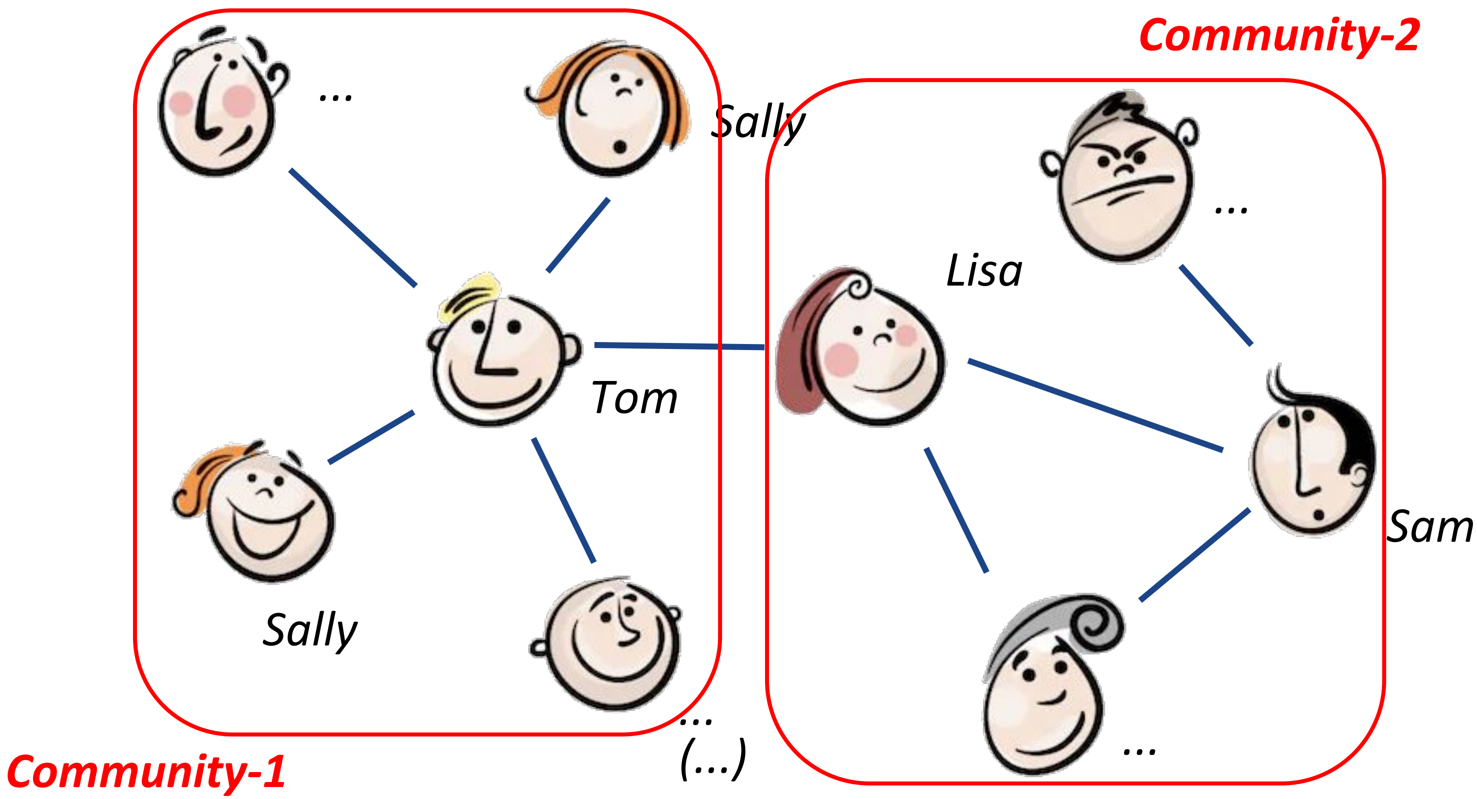
Graph Neural Networks



→ Unsupervised Learning with GNNs
→ Recently proposed methods

Motivating Example

- **Identify Communities:** (Unsupervised)
  - Grouping authors with similar research interests



Unsupervised Representation Learning

- Labeled data is expensive
- Allows to discover interesting structure from large-scale graphs

Unsupervised Representation Learning

- Labeled data is expensive
- Allows to discover interesting structure from large-scale graphs

- **Methods**
  - GraphSAGE
  - Graph Auto-Encoder (GAE)
  - Deep Graph Infomax (DGI)

GraphSAGE [Hamilton et al. NeurIPS '17]

- Propose three neighborhood aggregators

Mean Aggregator:

$$\mathbf{h}_v^k \leftarrow \sigma(\mathbf{W} \cdot \text{MEAN}(\{\mathbf{h}_v^{k-1}\} \cup \{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\}).$$

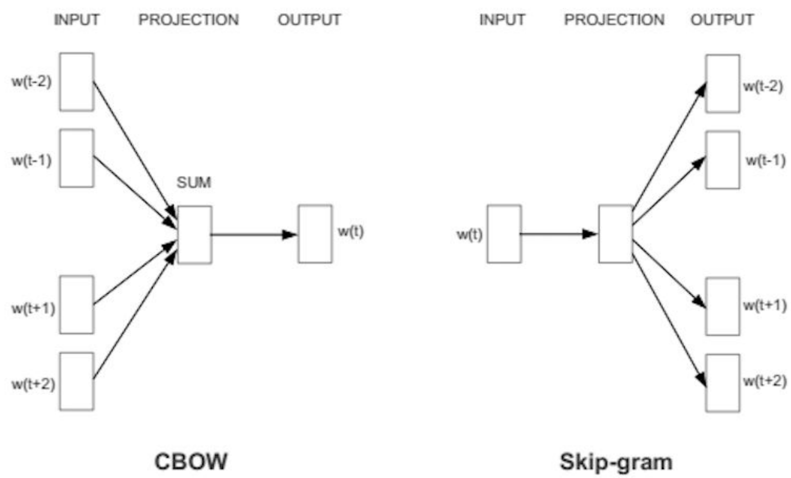
LSTM Aggregator:

Applies LSTM to a random permutation of neighbors.

Pooling Aggregator:

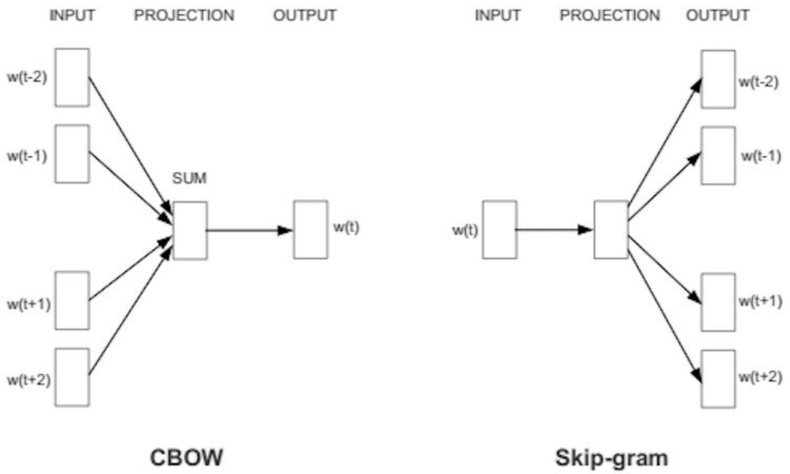
$$\text{AGGREGATE}_k^{\text{pool}} = \max(\{\sigma\left(\mathbf{W}_{\text{pool}}\mathbf{h}_{u_i}^k + \mathbf{b}\right), \forall u_i \in \mathcal{N}(v)\}),$$

GraphSAGE (Hamilton et al. NeurIPS ‘17)



$$J_{\mathcal{G}}(\mathbf{z}_u) = -\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right),$$

GraphSAGE (Hamilton et al. NeurIPS '17)



$$J_{\mathcal{G}}(\mathbf{z}_u) = \boxed{-\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right)} - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right),$$

Nearby nodes →

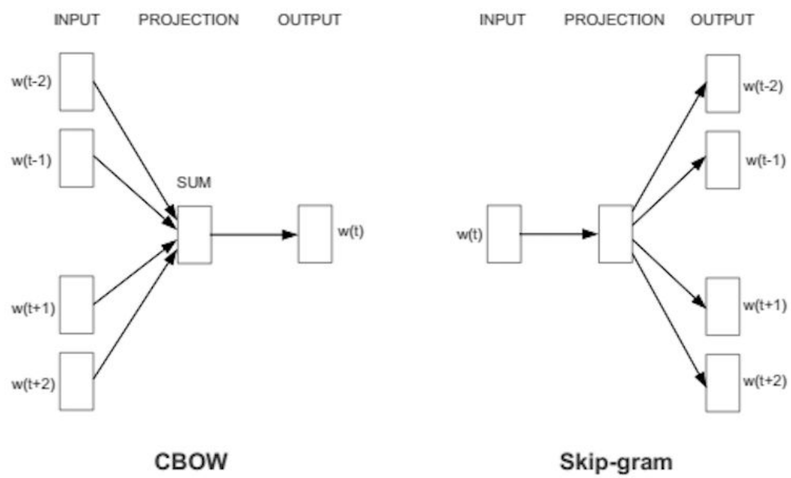
Similar representation

v: *is a node that co-occurs near* ***u***

P_n: Negative sampling distribution (from word2vec)

z_u: Embedding from GraphSAGE

GraphSAGE (Hamilton et al. NeurIPS '17)



$$J_{\mathcal{G}}(\mathbf{z}_u) = \boxed{-\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right)} - \boxed{Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right),}$$

Nearby nodes → Similar representation

Disparate nodes → Highly distinct representation

v: *is a node that co-occurs near* ***u***

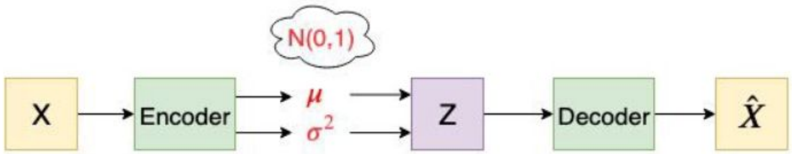
P_n: Negative sampling distribution (from word2vec)

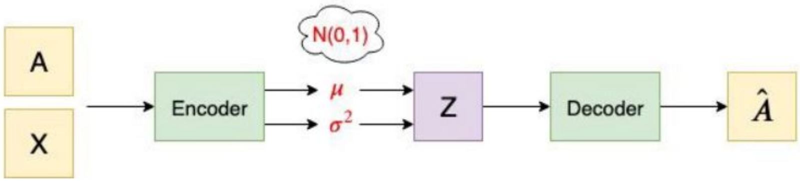
z_u: Embedding from GraphSAGE

Q: Number of negative samples

Graph Auto-Encoder (GAE) [Kipf et al., BDL-NeurIPS '16]

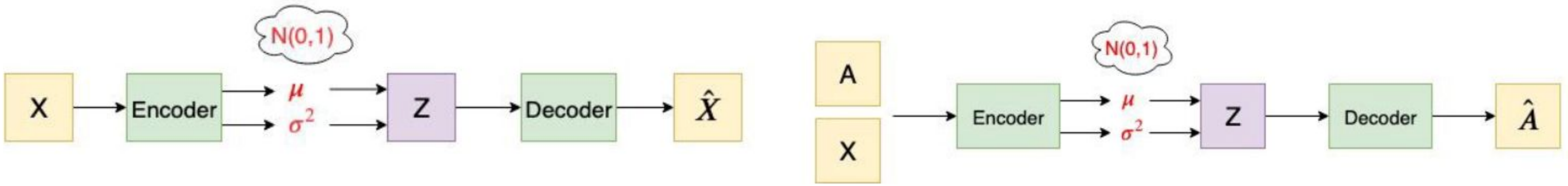
- Variational autoencoder (VAE) based model





Graph Auto-Encoder (GAE) [Kipf et al., BDL-NeurIPS '16]

- Variational autoencoder (VAE) based model

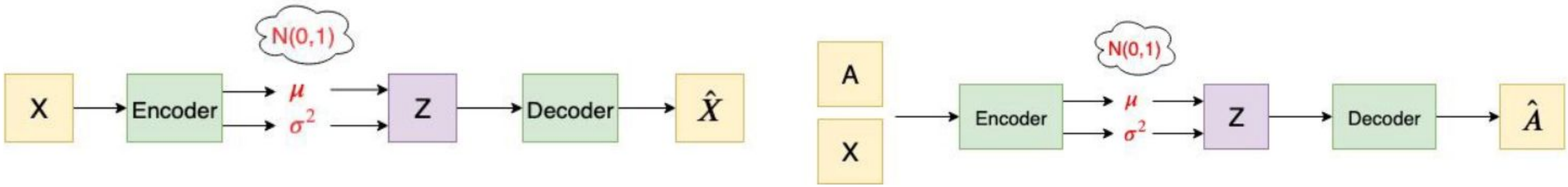


Encoder:

$$\mu = GCN_\mu(X, A)$$
$$\sigma = GCN_\sigma(X, A),$$
$$GCN(X, A) = \hat{A} ReLU(\hat{A} X W_0) W_1$$

Graph Auto-Encoder (GAE) [Kipf et al., BDL-NeurIPS '16]

- Variational autoencoder (VAE) based model



Encoder:

$$\mu = GCN_{\mu}(X, A)$$
$$\sigma = GCN_{\sigma}(X, A),$$
$$GCN(X, A) = \hat{A} ReLU(\hat{A} X W_0) W_1$$

Decoder:

$$p(\mathbf{A} \mid \mathbf{Z}) = \prod_{i=1}^{N} \prod_{j=1}^{N} p(A_{ij} \mid \mathbf{z}_i, \mathbf{z}_j) \,,$$
$$\text{with} \quad p(A_{ij} = 1 \mid \mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^\top \mathbf{z}_j) \,,$$

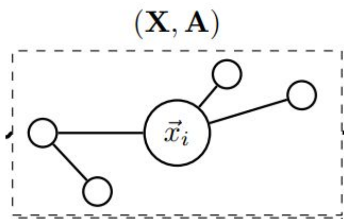
Loss:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}\big[\log p(\mathbf{A} \mid \mathbf{Z})\big] - \mathrm{KL}\big[q(\mathbf{Z} \mid \mathbf{X}, \mathbf{A}) \,||\, p(\mathbf{Z})\big] \,,$$

Reconstruction Loss

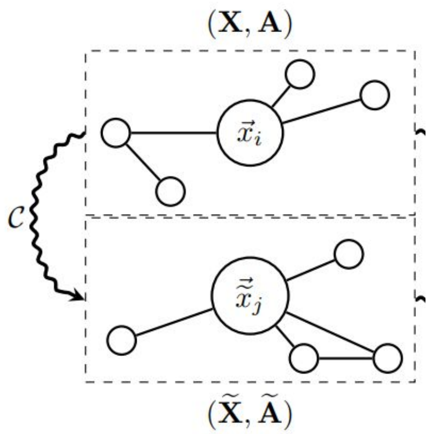
KL-Divergence

Deep Graph Infomax (Velickovic´ et al. ICLR '19)



Deep Graph Infomax (Velickovic´ et al. ICLR '19)

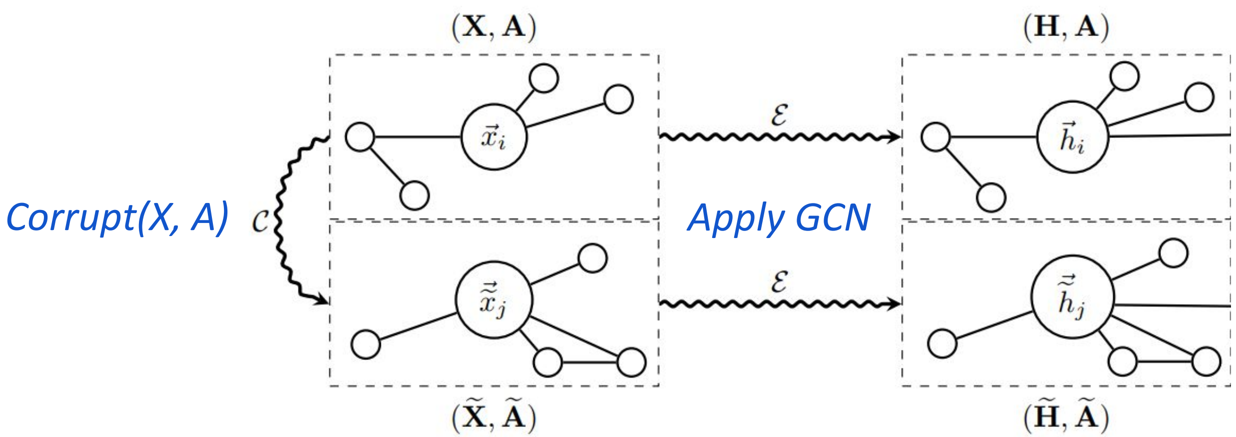
Corrupt(X, A)



Deep Graph Infomax (Velickovic´ et al. ICLR '19)

- Maximizes mutual information across graph's patch representations.

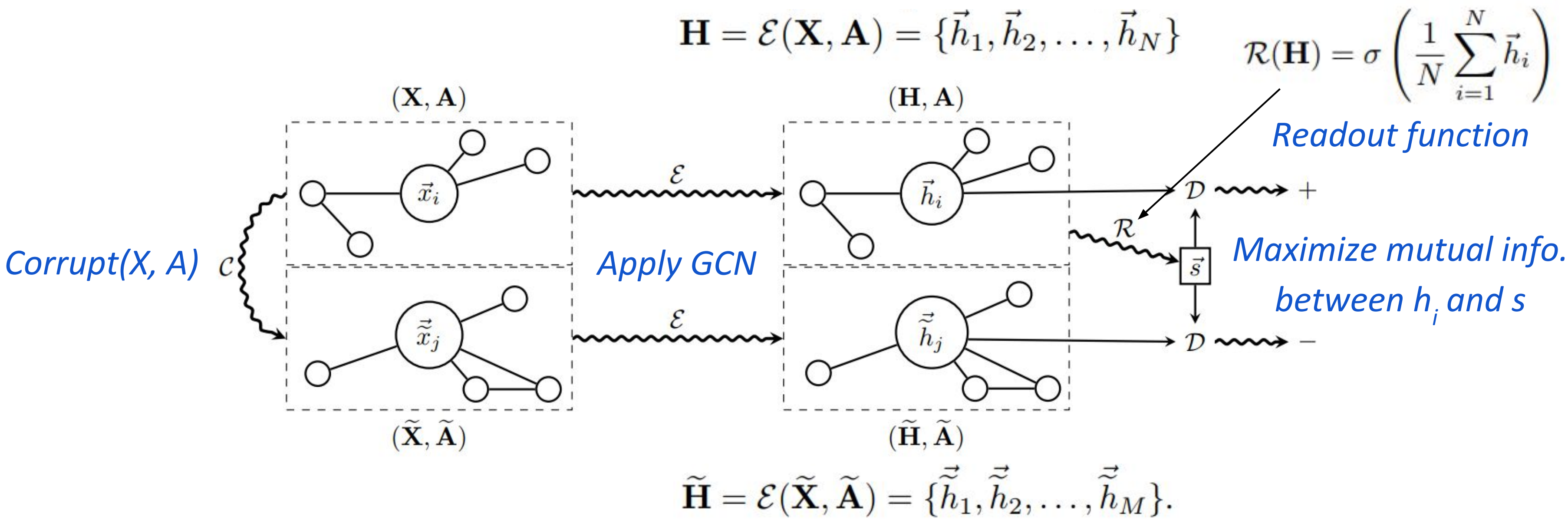
$$\mathbf{H} = \mathcal{E}(\mathbf{X}, \mathbf{A}) = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}$$



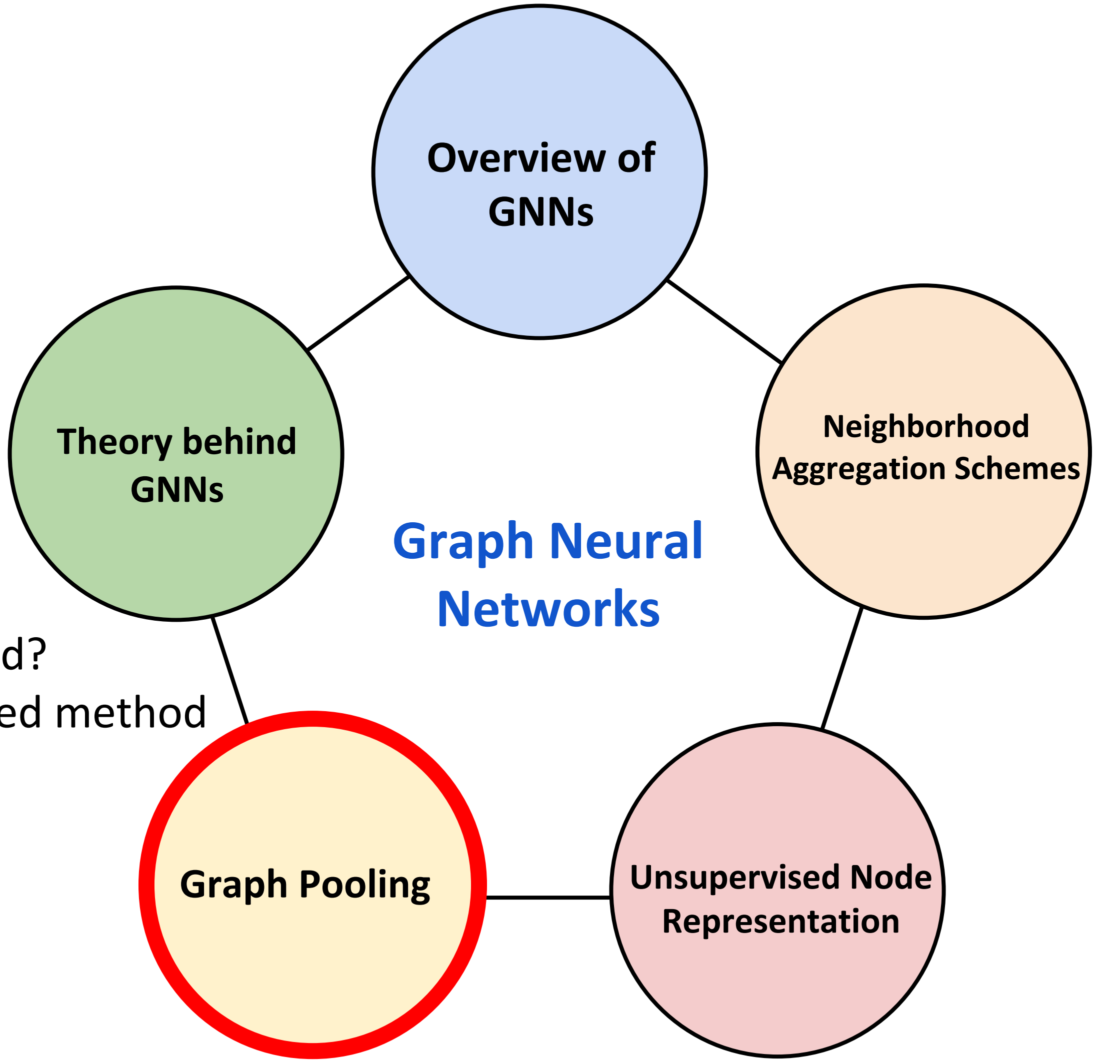
$$\widetilde{\mathbf{H}} = \mathcal{E}(\widetilde{\mathbf{X}}, \widetilde{\mathbf{A}}) = \{\vec{\widetilde{h}}_1, \vec{\widetilde{h}}_2, \ldots, \vec{\widetilde{h}}_M\}.$$

Deep Graph Infomax (Velickovic´ et al. ICLR ‘19)

- Maximizes mutual information across graph’s patch representations.



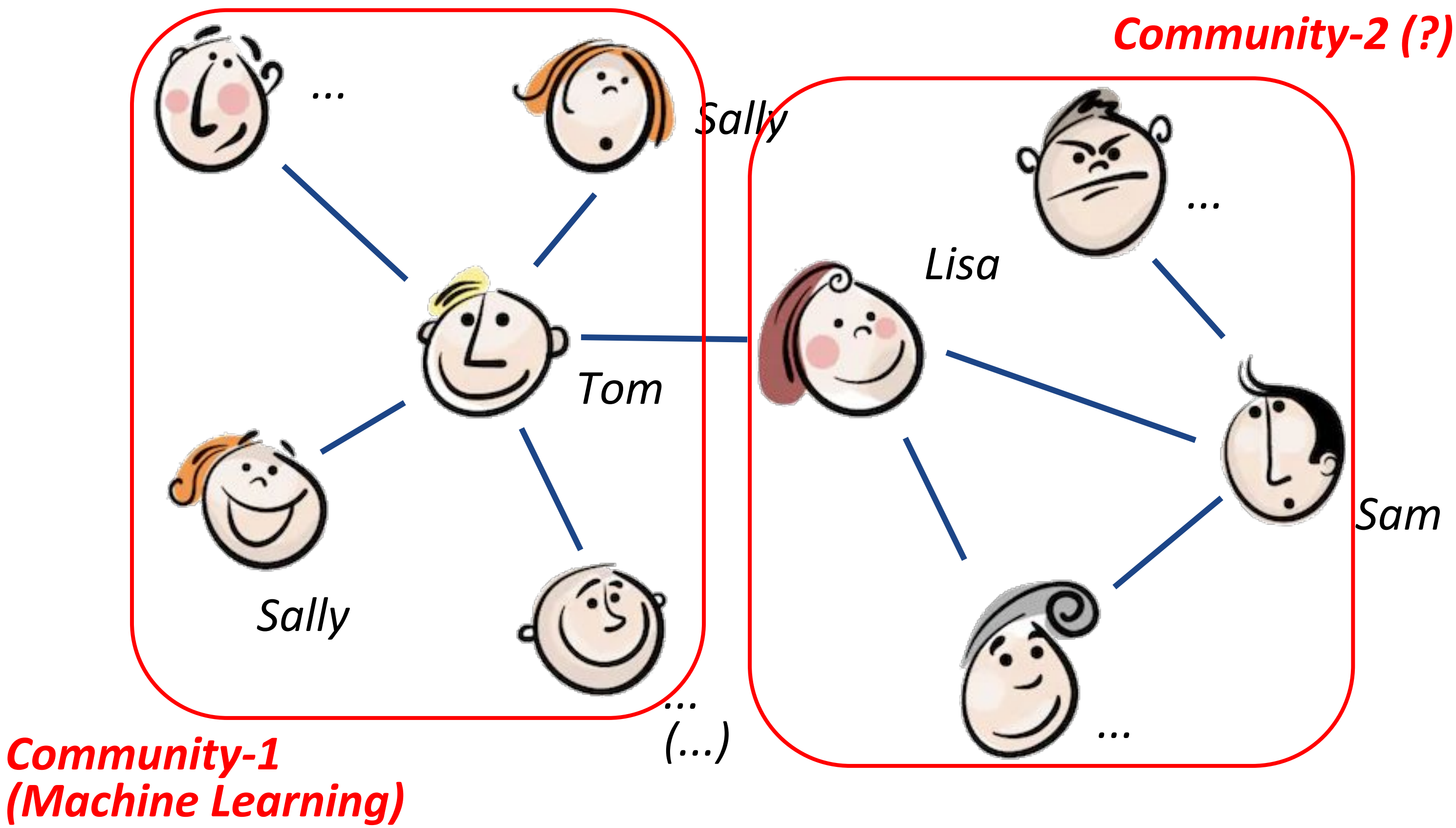
Graph Neural Networks



→ Why is it required?
→ Recently proposed method

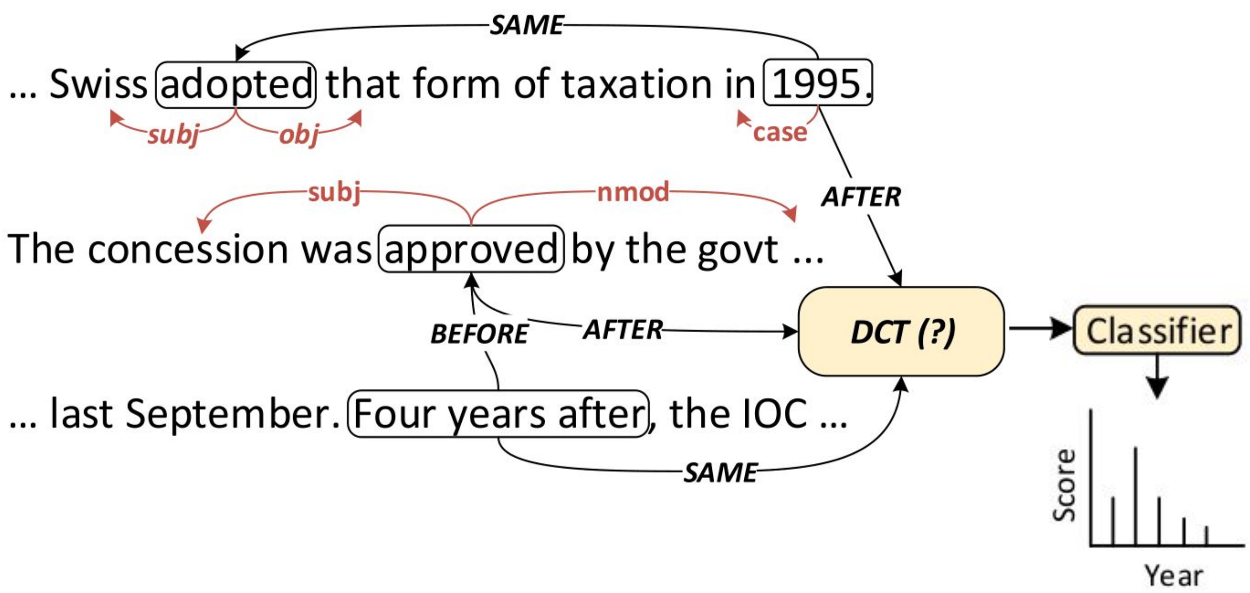
Motivating Example

- **Graph Classification:** (Supervised)
  - Identifying class of each community.



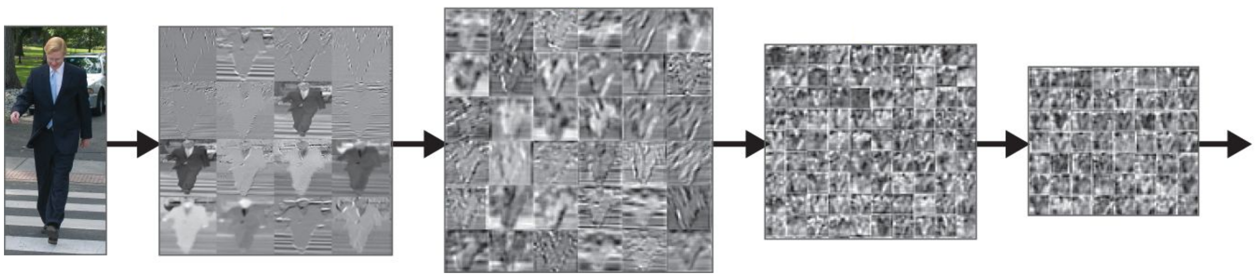
Motivating Example from NLP

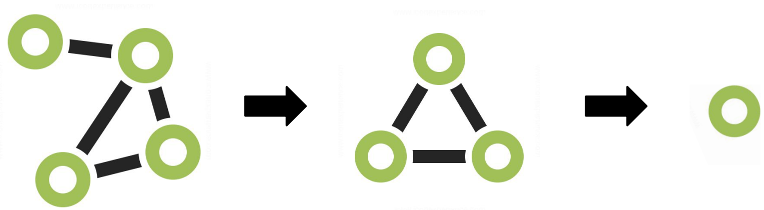
- **Document Classification**
  - Getting a representation for the entire document



Graph Pooling

- Essential for graph-level tasks (Graph Classification)
- Goal: Get an embedding for the entire graph



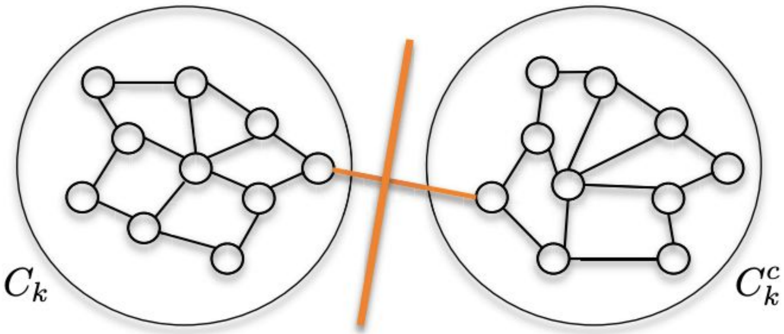


- Graph clustering is NP-Hard.

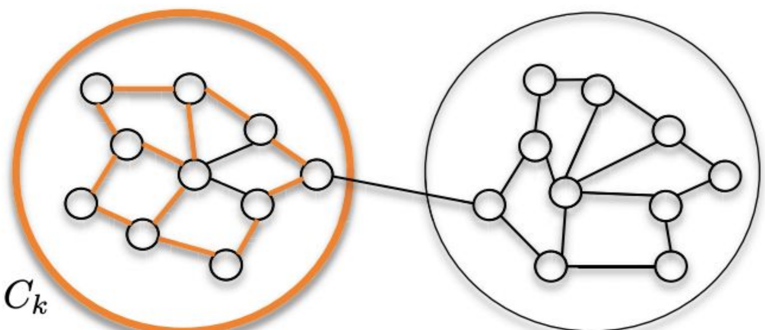
Graph Pooling

- Methods
  - Simple max/mean pooling
    - Inefficient and overlooks node ordering
  - Graclus clustering algorithm
  - Set2Set method
  - Differentiable Pooling (DiffPool)

Graph Clustering Measures



Partitioning by min edge cuts.



Partitioning by max vertex matching.

Normalized Cut:

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} \frac{\mathrm{Cut}(C_k, C_k^c)}{\mathrm{Vol}(C_k)}$$

$\Updownarrow$ Equivalence by complementarity

Normalized Association:

$$\max_{C_1,\ldots,C_K} \sum_{k=1}^{K} \frac{\mathrm{Assoc}(C_k)}{\mathrm{Vol}(C_k)}$$

where

$$Cut(A,B) = \textstyle\sum_{i\in A, j\in B} W_{ij}$$

$$Vol(A) = \textstyle\sum_{i\in A} d_i$$

$$Assoc(A) = \textstyle\sum_{i\in A, j\in A} W_{ij}$$

Image credits: Xavier Bresson

Graph Pooling: Graclus Clustering

- Greedy algorithm, minimizes Normalized Association.
  - Vertex Matching
  - Graph Coarsening

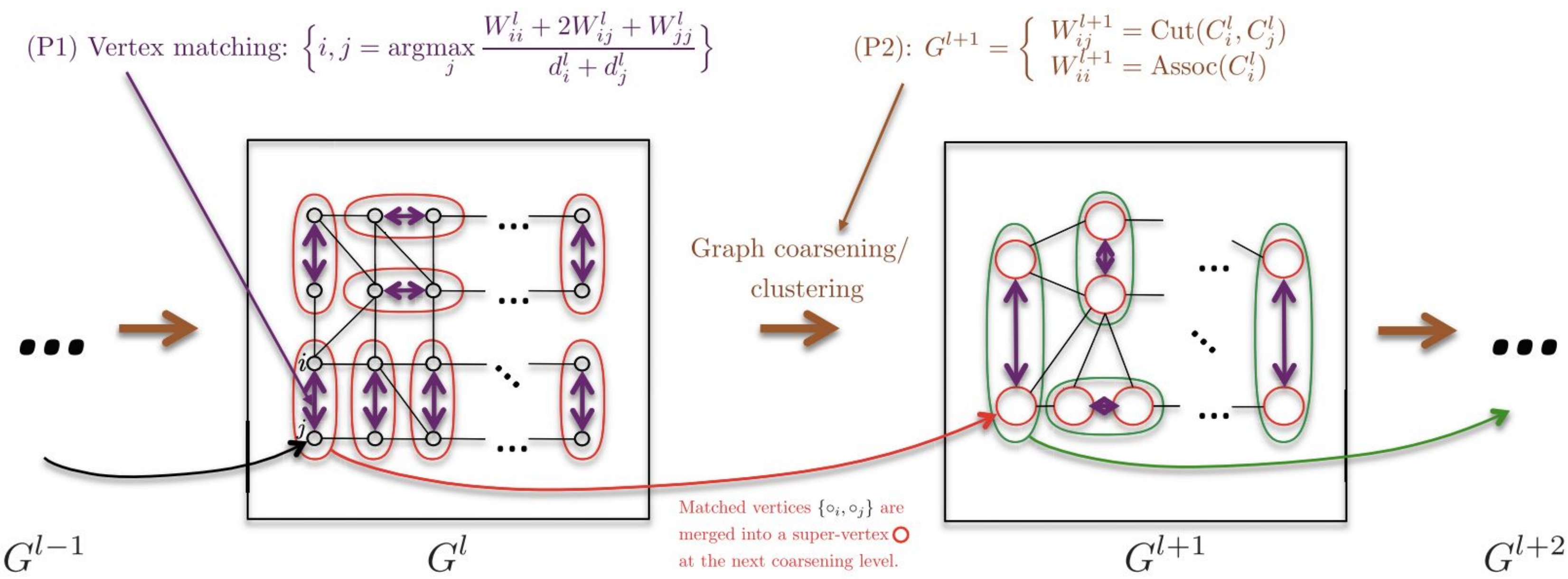
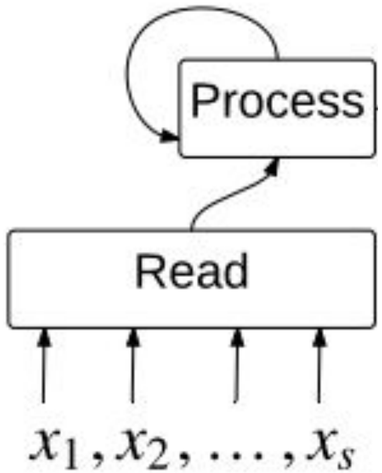


Image credits: Xavier Bresson

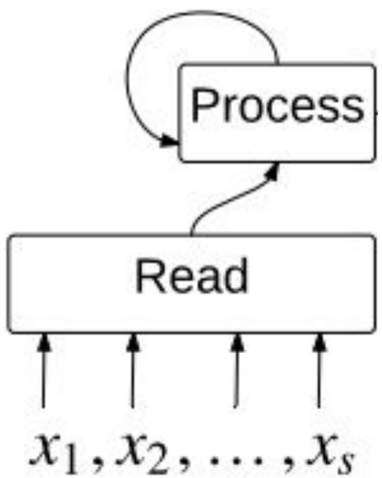
Graph Pooling: Set2Set (Vinyals et al. ICLR '15)

- Set2Set has three components:
    - Reading block: Embeds each element onto a memory unit.
    - Process block: Performs T steps of LSTM w/o input/output



Graph Pooling: Set2Set (Vinyals et al. ICLR '15)

- Set2Set has three components:
  - Reading block: Embeds each element onto a memory unit.
  - Process block: Performs T steps of LSTM w/o input/output



Assumes an ordering on all the vertices

Takes random initial state q_0

$$\boxed{q_t = LSTM(q^*_{t-1}) \quad (3)}$$

$$e_{i,t} = f(m_i, q_t) \quad (4)$$

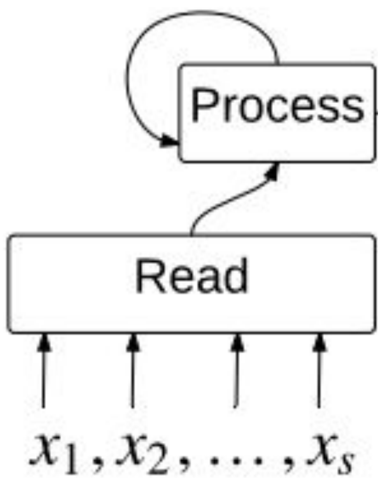
$$a_{i,t} = \frac{\exp(e_{i,t})}{\sum_j \exp(e_{j,t})} \quad (5)$$

$$r_t = \sum_i a_{i,t} m_i \quad (6)$$

$$q^*_t = [q_t \; r_t] \quad (7)$$

Graph Pooling: Set2Set (Vinyals et al. ICLR '15)

- Set2Set has three components:
  - Reading block: Embeds each element onto a memory unit.
  - Process block: Performs T steps of LSTM w/o input/output



Assumes an ordering on all the vertices

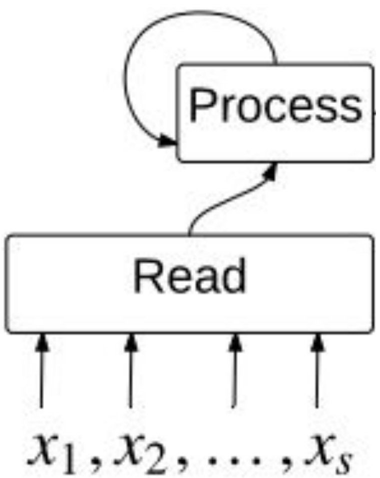
Takes random initial state q_0

$$
\begin{aligned}
q_t &= LSTM(q^*_{t-1}) && (3)\\
e_{i,t} &= f(m_i, q_t) && (4)\\
a_{i,t} &= \frac{\exp(e_{i,t})}{\sum_j \exp(e_{j,t})} && (5)\\
r_t &= \sum_i a_{i,t} m_i && (6)\\
q^*_t &= [q_t \; r_t] && (7)
\end{aligned}
$$

Computes attention scores over all nodes

Graph Pooling: Set2Set (Vinyals et al. ICLR '15)

- Set2Set has three components:
  - Reading block: Embeds each element onto a memory unit.
  - Process block: Performs T steps of LSTM w/o input/output



Assumes an ordering on all the vertices

Takes random initial state q_0

$$q_t = LSTM(q^*_{t-1}) \quad (3)$$

$$e_{i,t} = f(m_i, q_t) \quad (4)$$

$$a_{i,t} = \frac{\exp(e_{i,t})}{\sum_j \exp(e_{j,t})} \quad (5)$$

Computes attention scores over all nodes

$$\boxed{\begin{aligned} r_t &= \sum_i a_{i,t} m_i && (6) \\ q^*_t &= [q_t \; r_t] && (7) \end{aligned}}$$

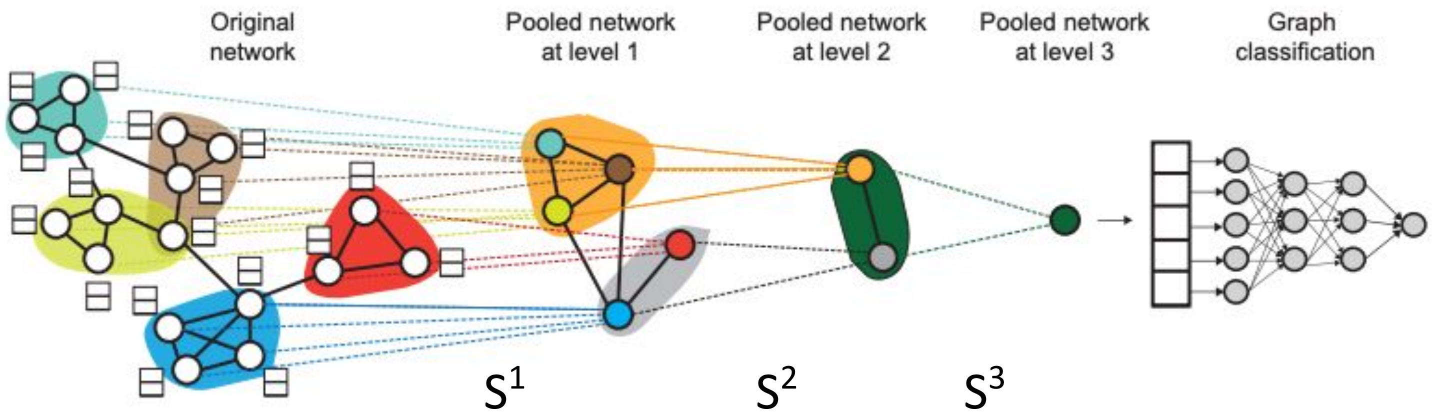
Computes weighted sum and pass it back to LSTM

q_T is the graph representation which is invariant of initial vertex ordering

Graph Pooling: DiffPool (Ying et al. NeurIPS '18)

- Learns differentiable soft cluster assignment (S^l) from nodes in layer l to nodes in layer (l+1)

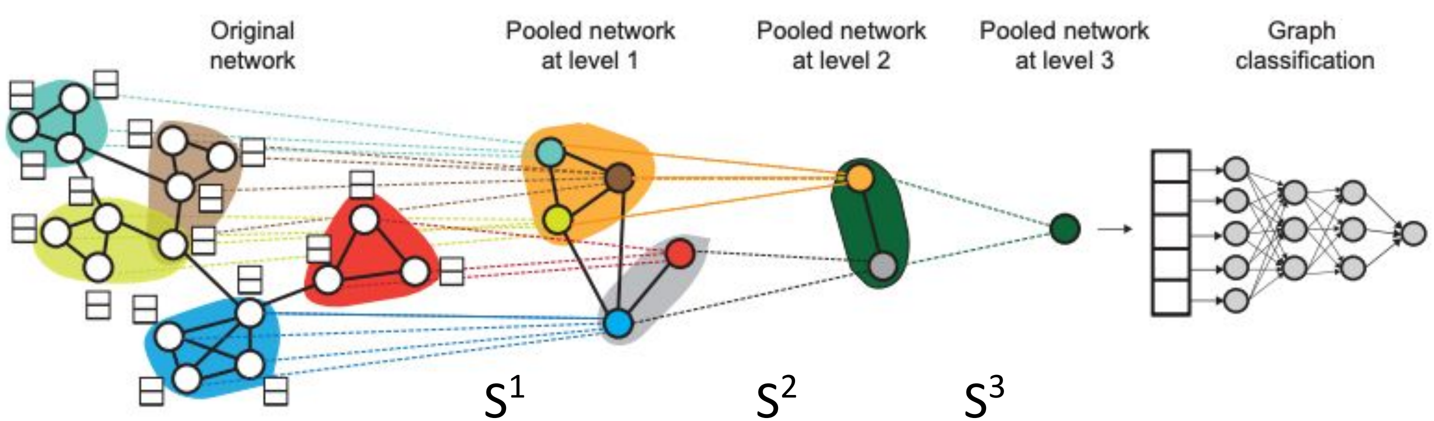
$$S^l \in \mathbb{R}^{n_l \times n_{l+1}}$$



Graph Pooling: DiffPool [(Ying et al. NeurIPS '18)]

- Learns differentiable soft cluster assignment (S^l) from nodes in layer l to nodes in layer (l+1)

$$S^l \in \mathbb{R}^{n_l \times n_{l+1}}$$



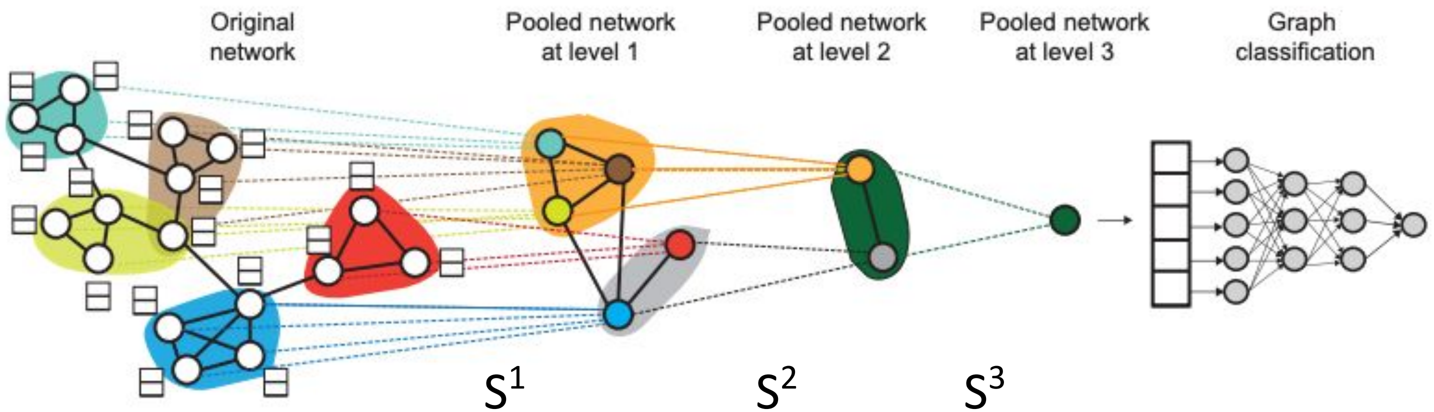
$$S^{(l)} = \mathrm{softmax}\left(\mathrm{GNN}_{l,\mathrm{pool}}(A^{(l)}, X^{(l)})\right) \quad X^{(l+1)} = {S^{(l)}}^T Z^{(l)} \in \mathbb{R}^{n_{l+1} \times d},$$

$$Z^{(l)} = \mathrm{GNN}_{l,\mathrm{embed}}(A^{(l)}, X^{(l)}), \quad A^{(l+1)} = {S^{(l)}}^T A^{(l)} S^{(l)} \in \mathbb{R}^{n_{l+1} \times n_{l+1}}.$$

Graph Pooling: DiffPool (Ying et al. NeurIPS '18)

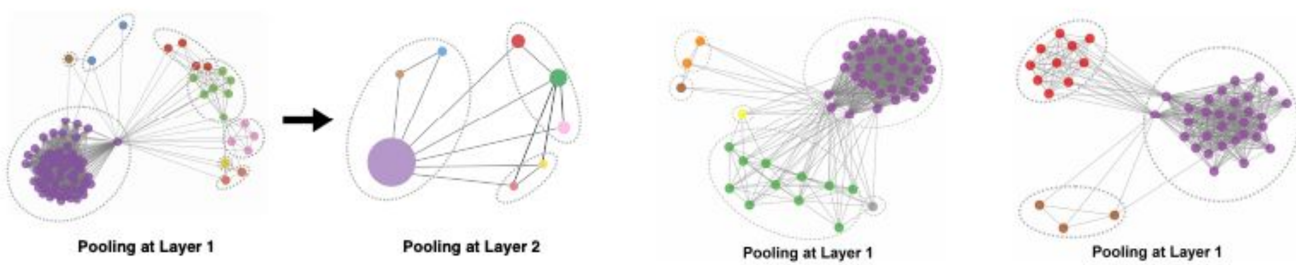
- Learns differentiable soft cluster assignment (S^l) from nodes in layer l to nodes in layer (l+1)

$$S^l \in \mathbb{R}^{n_l \times n_{l+1}}$$



$$(A^{(l+1)}, X^{(l+1)}) = \textsc{DiffPool}(A^{(l)}, Z^{(l)})$$

Graph Pooling: DiffPool (Ying et al. NeurIPS ‘18)



Learned clusters are interpretable

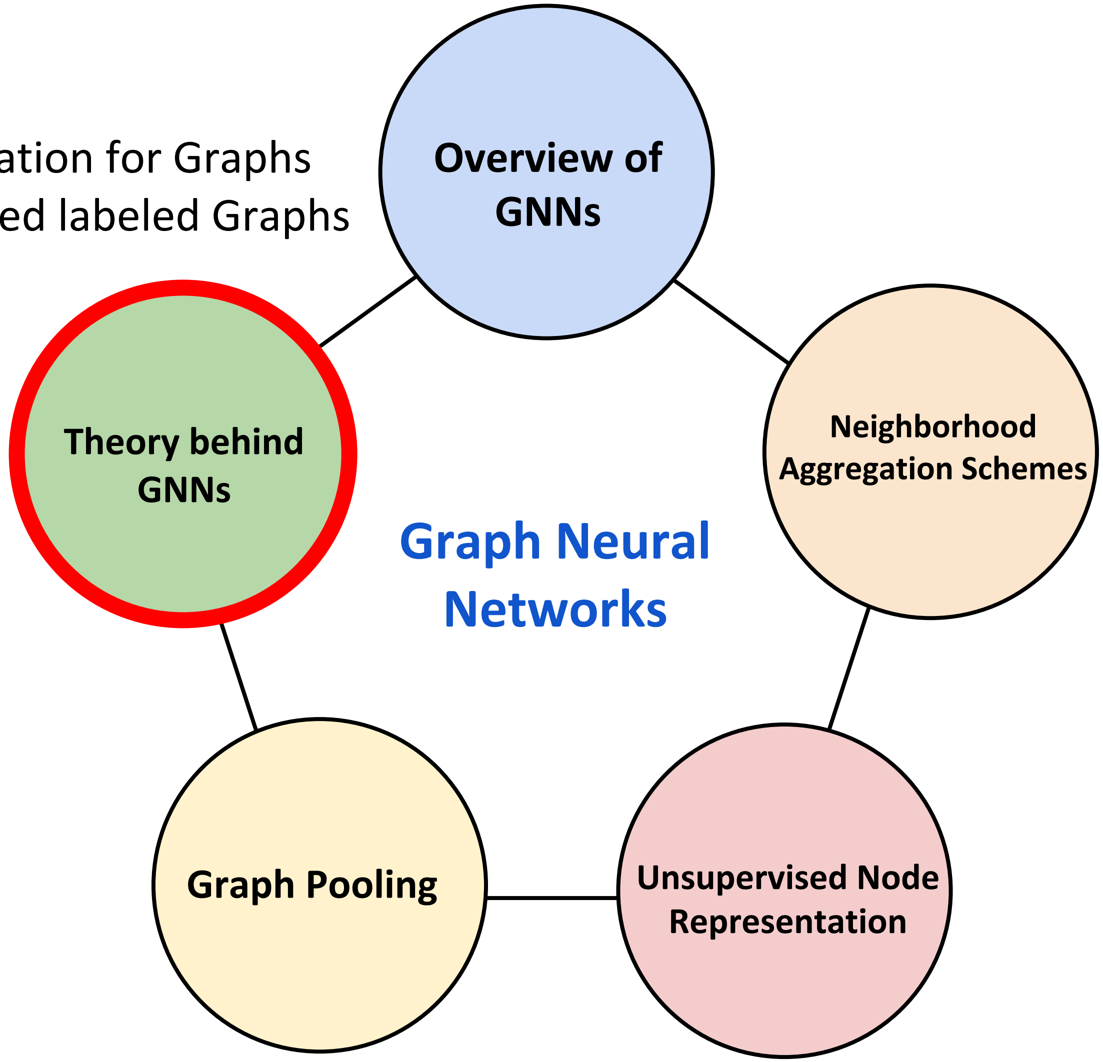
	Method	ENZYMES	D&D
GNN	GRAPHSAGE	54.25	75.42
	ECC	53.50	74.10
	SET2SET	60.15	78.12
	SORTPOOL	57.12	79.37
	DIFFPOOL-DET	58.33	75.47
	DIFFPOOL-NOLP	61.95	79.98
	DIFFPOOL	**62.53**	**80.64**

Outperforms existing methods

Graph Neural Networks

→ CNNs generalization for Graphs

→ GCNs for directed labeled Graphs



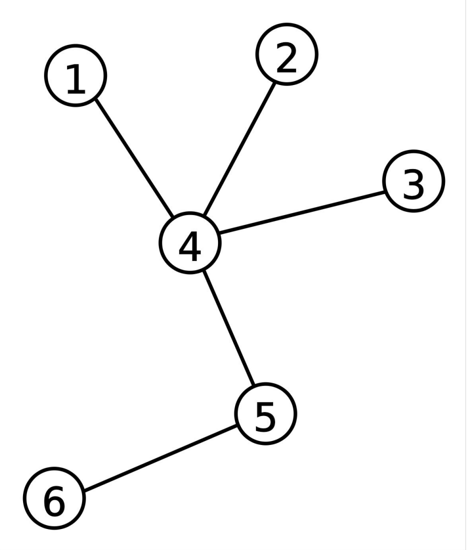

BRACE YOURSELVES
MATH IS COMING
memegenerator.net

Notations

- Given graph is undirected, connected and weighted

$$\mathcal{G} = \{V, E, \mathbf{W}\}$$

- where V denotes vertices with $|V| = \mathrm{N}$,
- E indicates set of edges
- $\boldsymbol{W}$ is a weighted adjacency matrix,
- W_{ij} denotes weight of the edge



Graph Signal

- Function (f) defined on the vertices of the graph

$$f : V \to \mathbb{R}$$

- Can be represented as a vector:

$$\mathbf{f} = [f(v_1), f(v_2), \ldots, f(v_N)]$$

$$\mathbf{f} \in \mathbb{R}^N$$

- where i^{th} entry corresponds to i^{th} vertex in the graph.

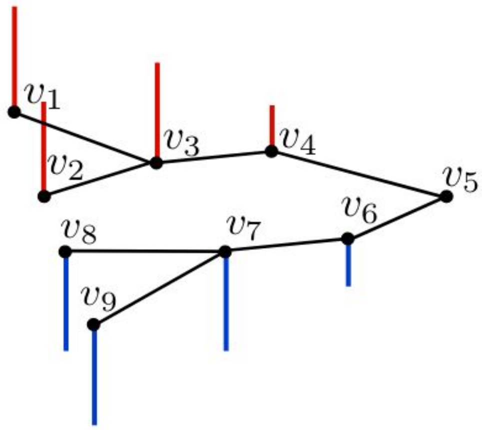


Image credits: Shuman, David I. et al. “The emerging field of signal processing on graphs”.

Convolution in Euclidean space

- Given two function f, g: [-π, π] →R, convolution is given as

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

- Properties
  - Convolution Theorem: $\widehat{(f \star g)} = \hat{f} \cdot \hat{g}$
  - Computationally efficient

Original Image

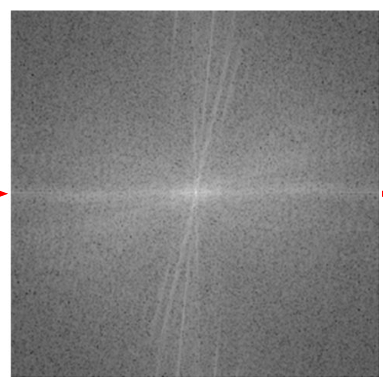
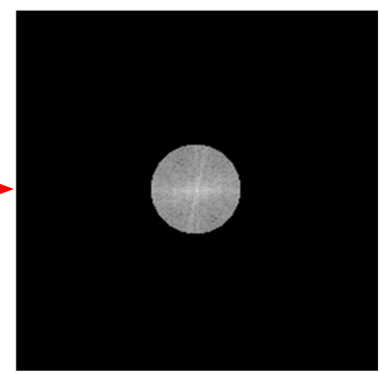


Image Fourier Transform



Product with filter

Inverse Fourier Transform

Convolution in Graph space

- **By analogy**, given two graph signals f, g.

$$(f \star g) = \widehat{\underbrace{\hat{f}}_{\text{Graph Fourier Transform}} \cdot \hat{g}}$$

Graph Inverse Fourier Transform

Multiplication with a Filter

Graph Fourier Transform

Convolution in Graph space

- **By analogy**, given two graph signals f, g.

$$(f \star g) = \widehat{\hat{f} \cdot \hat{g}}$$

Graph Inverse Fourier Transform

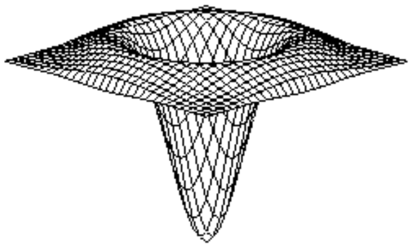
Multiplication with a Filter

Graph Fourier Transform

Spectral Graph Theory

Spectral Graph Theory

- Graph Laplacian
  - For any signal **f**, it is defined as



$$(\Delta f)(i) = \sum_{j \in N_i} W_{i,j}[f(i) - f(j)]$$

  - where N_i denotes neighbors of i^{th} vertex.
  - Difference between f and its local average
    - Small if function is smooth, large if it oscillates
  - Represented as Laplacian matrix

$$\Delta = D - \mathbf{W}$$

where $D = diag(\sum_{i \neq j} W_{i,j})$

Spectral Graph Theory

- Graph Laplacian
  - Graph Laplacian matrix Δ positive semidefinite [Proof]
    - Has complete set of orthonormal eigenvectors (Φ)
    - Eigenvalues (Λ) are non-negative

$$\Delta = \Phi^T \Lambda \Phi$$ (Spectral Decomposition)

Spectral Graph Theory

- Graph Laplacian
  - Graph Laplacian matrix Δ positive semidefinite [Proof]
    - Has complete set of orthonormal eigenvectors (Φ)
    - Eigenvalues (Λ) are non-negative

$$\mathbf{\Delta} = \mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}$$ (Spectral Decomposition)

- Graph Fourier Transform
  - In matrix notation, Fourier transform is given as

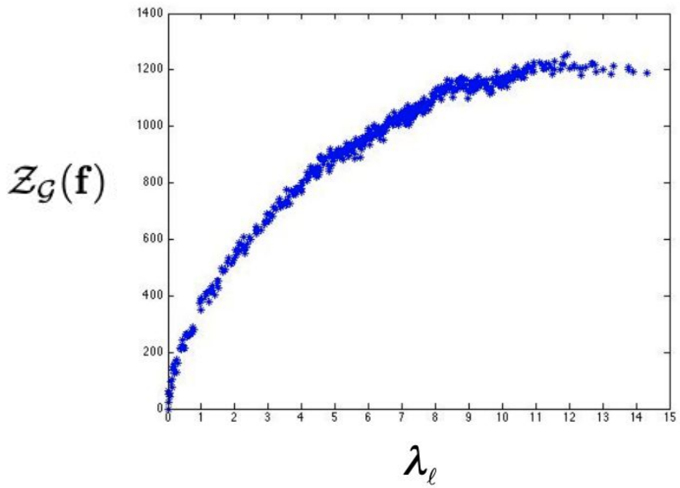
$$\hat{\mathbf{f}} = \mathbf{\Phi}^T \mathbf{f} \qquad \mathbf{f} = \mathbf{\Phi} \hat{\mathbf{f}}$$

(Fourier Transform) (Inverse Fourier Transform)

Spectral Graph Theory

- Graph Fourier Transform
  - Graph Laplacian eigenvalues and eigenvectors provide a similar notion of frequency
  - Eigenvectors corresponding to small eigenvalue are smooth and vice versa
  - Let *cross zero edges* be defined as:

$$\mathcal{Z}_{\mathcal{G}}(\mathbf{f}) := \{e = (i,j) \in \mathcal{E} : f(i)f(j) < 0\};$$



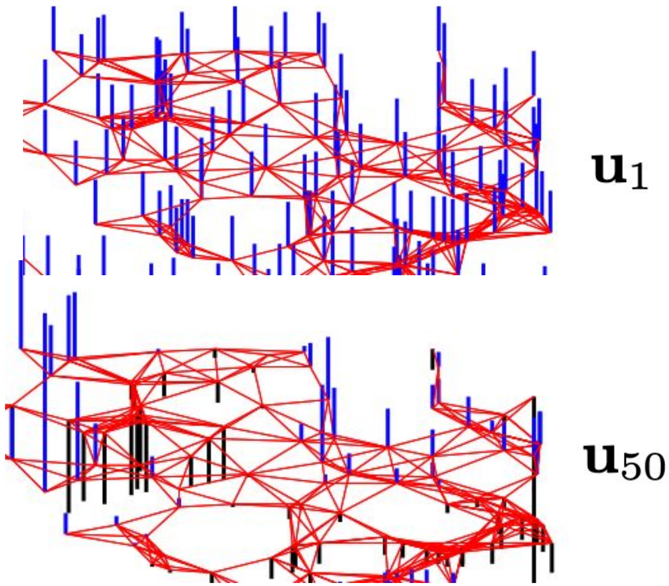


Image credits: Shuman, David I. et al. “The emerging field of signal processing on graphs”.

Spectral Graph Theory

- The below result reinforce the previous observation

Measure of roughness

$$\lambda_0 = \min_{\substack{\mathbf{f}\in\mathbb{R}^N \\ \|\mathbf{f}\|_2=1}} \boxed{\{\mathbf{f}^T \Delta \mathbf{f}\},}$$

Solution: First eigenvector

$$\text{and } \lambda_l = \min_{\substack{\mathbf{f}\in\mathbb{R}^N \\ \|\mathbf{f}\|_2=1 \\ \mathbf{f}\perp span\{\mathbf{u}_0,\ldots,\mathbf{u}_{l-1}\}}} \{\mathbf{f}^T \Delta \mathbf{f}\} \quad l = 1, 2, \ldots, N-1,$$

Solution: l^{th} eigenvector

Convolution over Graphs

- In Euclidean space

$$(\mathbf{f} \star \mathbf{g}) = \Phi(\Phi^T \mathbf{g} \circ \Phi^T \mathbf{f})$$

- By analogy in non-Euclidean space:

$$\begin{aligned}
\mathbf{f} \star \mathbf{g} &= \mathbf{\Phi}\,(\mathbf{\Phi}^\top \mathbf{g} \circ \mathbf{\Phi}^\top \mathbf{f}) \\
&= \underbrace{\mathbf{\Phi}\,\mathrm{diag}(\hat{g}_1, \ldots, \hat{g}_n)\mathbf{\Phi}^\top}_{\mathbf{G}} \mathbf{f} \\
&= \mathbf{\Phi}\hat{g}(\mathbf{\Lambda})\mathbf{\Phi}^\top \mathbf{f} = \hat{g}(\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top)\mathbf{f} = \hat{g}(\Delta)\mathbf{f}
\end{aligned}$$

  - Filter coefficients depend on Fourier basis
  - Expensive computation

Convolution in non-Euclidean space

- Solution [Defferrard et al. 2016]
  - Parameterize $\hat{g}(\Delta)$ as a polynomial function
  - Chebyshev polynomial, computed recursively from Δ
  - $O(n^2) << O(K|E|)$

$$\hat{g}(\Delta)\mathbf{f} = \sum_{k=0}^{K} \theta_k T_k(\tilde{\Delta})\mathbf{f}$$

  - $T_k(\tilde{\Delta})$ is the Chebyshev polynomial of order k
  - $\tilde{\Delta} = 2\Delta/\lambda_{max} - I_N$, scaled eigenvalues in [-1, 1]
  - $T_0(x) = 1, T_1(x) = x, T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$

Convolution in non-Euclidean space

- GCN first-order approximation [Kipf et al. 2016]
  - Taking K=1, in Chebyshev polynomial formulation

$$\hat{g}(\Delta)\mathbf{f} = (\theta_0 + \theta_1 \tilde{\Delta})\mathbf{f}$$

$$\tilde{\Delta} = 2\Delta/\lambda_{max} - I_N = \Delta - I_N \leftarrow \lambda_{max} = 2$$

(Assumption)

$$\hat{g}(\Delta)\mathbf{f} = (\theta_0 + \theta_1(\Delta - I_N))\mathbf{f}$$

$$\hat{g}(\Delta)\mathbf{f} = (\theta_0 - \theta_1 D^{-1/2} A D^{-1/2})\mathbf{f}$$

(Normalized Laplacian)

$$\Delta = I_N - D^{-1/2} A D^{-1/2}$$

$$\hat{g}(\Delta)\mathbf{f} = \theta(I_N + D^{-1/2} A D^{-1/2})\mathbf{f} \leftarrow \theta_0 = -\theta_1 = \theta$$

(Assumption)

$$\hat{g}(\Delta)\mathbf{f} = \theta(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2})\mathbf{f}$$

$$I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}},$$

(Renormalization)

where $\tilde{A} = A + I_N$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$

Convolution in non-Euclidean space

- GCN first-order approximation [Kipf et al. 2016]

$$\boldsymbol{H} = f((\tilde{\boldsymbol{D}}^{-\frac{1}{2}}(\boldsymbol{A}+\boldsymbol{I})\tilde{\boldsymbol{D}}^{-\frac{1}{2}})\mathcal{X}\boldsymbol{W})$$

$$\mathbf{H} = f(\tilde{A}\mathcal{X}\mathbf{W})$$

Can be rewritten as:

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$

For capturing k-hold neighborhood, multiple of them can be stacked

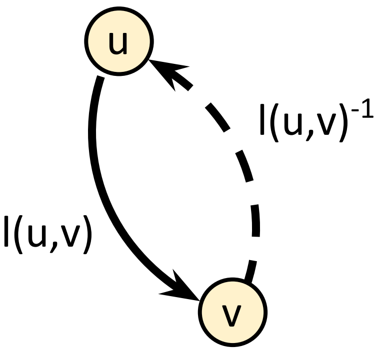
$$h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W^k h_u^k + b^k\right), \quad \forall v \in \mathcal{V}.$$

GCNs for Directed Labeled Graphs [Marcheggiani et al., EMNLP '17]

- Earlier formulation was limited to undirected graphs

$$h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W^k h_u^k + b^k\right), \quad \forall v \in \mathcal{V}.$$

- For each directed edge, add an inverse edge



- GCN update equation is given as:

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

- Edgewise gating:

$$g_{u,v}^k = \sigma\left(h_u^k \cdot \hat{w}_{l(u,v)}^k + \hat{b}_{l(u,v)}^k\right), \quad h_v^{k+1} = f\left(\sum_{u\in\mathcal{N}(v)} g_{u,v}^k \times \left(W_{l(u,v)}^k h_u^k + b_{l(u,v)}^k\right)\right).$$

Message Passing Neural Networks [Gilmer et al., ICML '17]

- Single Common Framework for all GCN models

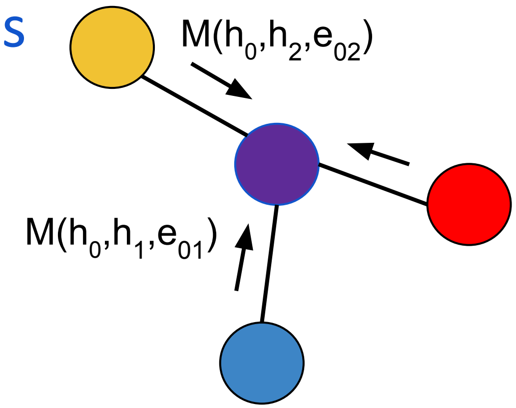
$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw})$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

M_t: Message function, U_t: Update function, e_{vw}: edge features

Message Passing Neural Networks [Gilmer et al., ICML '17]

- Single Common Framework for all GCN models



$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw})$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

M_t: Message function, U_t: Update function, e_{vw}: edge features

- Readout phase:

$$\hat{y} = R(\{h_v^T \mid v \in G\}).$$

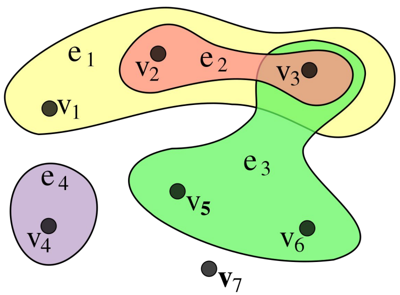
  - R must be invariant to the order of the nodes

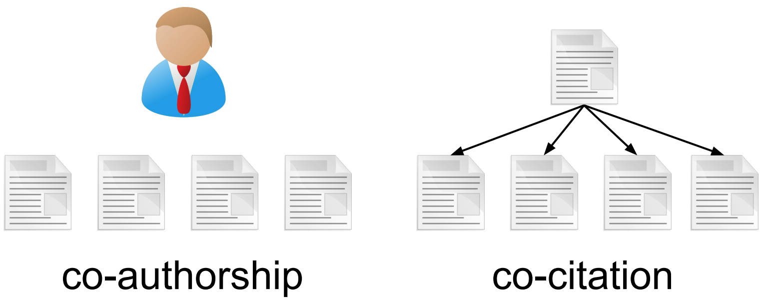
Hypergraph Convolutional Network (Yadati et al. NeurIPS‘19)

- Graphs have relationships **beyond pairwise**
- HyperGraphs:

$$\mathcal{H} = (V, E)$$

$$E \subseteq 2^V$$

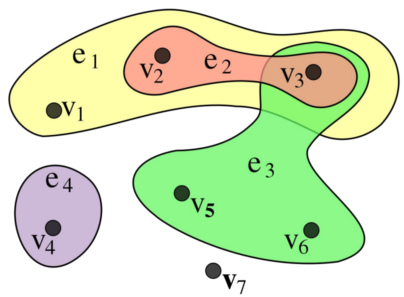


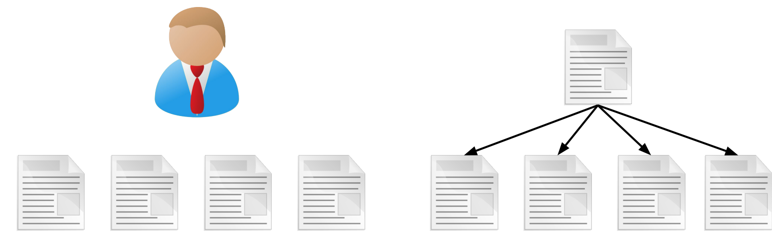


Hypergraph Convolutional Network (Yadati et al. NeurIPS'19)

- Graphs have relationships **beyond pairwise**
- HyperGraphs:

$$\mathcal{H} = (V, E)$$
$$E \subseteq 2^V$$

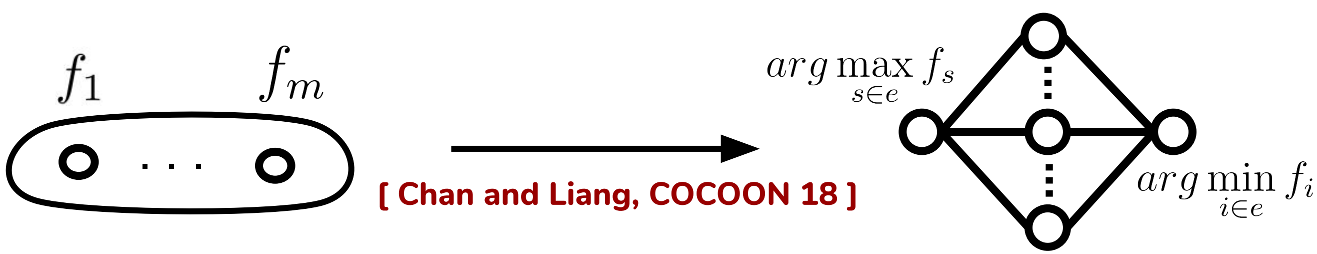




co-authorship

co-citation

- Extend GCNs to hypergraphs for Semi-Supervised Learning



$$f_{HyperGCN} = \text{softmax}\left(\mathbf{A}_1^{(e)} \; \text{ReLU}\left(\mathbf{A}_2^{(e)} \; \underline{XW^{(1)}}\right) W^{(2)}\right)$$

Other papers on GCNs

- Scaling GCNs to Large Graphs
  - FastGCN [Chen et al., ICLR 2018]
  - ClusterGCN [Chiang et al., KDD 2018]

Other papers on GCNs

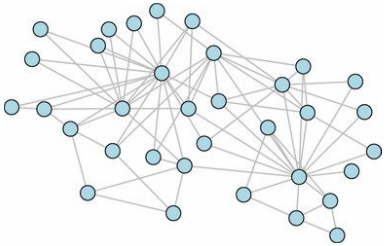
- Scaling GCNs to Large Graphs
  - FastGCN [Chen et al., ICLR 2018]
  - ClusterGCN [Chiang et al., KDD 2018]

- Theoretical Analysis:
  - How powerful are Graph Neural Networks? [Xu et al., ICLR 2019]
  - Open Problems: Approximation Power of Invariant GNNs [Maron et al., NIPS ’19]

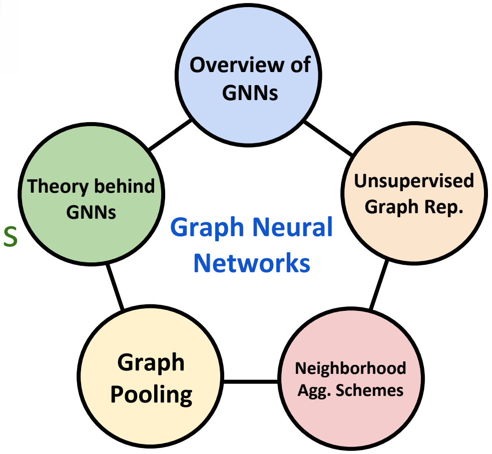
Other papers on GCNs

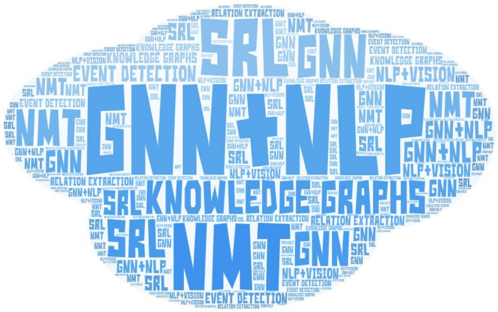
- Scaling GCNs to Large Graphs
  - FastGCN [Chen et al., ICLR 2018]
  - ClusterGCN [Chiang et al., KDD 2018]
- Theoretical Analysis:
  - How powerful are Graph Neural Networks? [Xu et al., ICLR 2019]
  - Open Problems: Approximation Power of Invariant GNNs [Maron et al., NIPS ’19]
- Survey Papers:
  - A Comprehensive Survey of GNNs [Wu et al., ‘19]
  - GNNs: A Review of Methods and Applications [Zhou et al., ‘19]
  - Deep Learning on Graphs: A Survey [Zhang et al., ‘18]

Tutorial Outline



- **Introduction**
  - ✓ Motivation
  - ✓ GNN Foundation
- **Methods**
  - ✓ Introduction to GNNs
  - ✓ Graph Pooling Unsupervised Learning using GNNs
  - ✓ Neighborhood Aggregation in GNNs
  - ✓ Other GCN Variants
- **Implementing GCNs**
- **Applications**
  - Semantic Role Labeling, NMT
  - Text Classification, Extraction
  - Knowledge Graphs
  - Vision + NLP
- **Open Problems and Conclusion**





?

Implementing GCNs

- Pytorch
  - Pytorch-geometric:
    - Implements Message Passing Neural Network
    - Contains implementation of several recent papers
    - Easy to understand entire codebase

Implementing GCNs

- Pytorch
  - Pytorch-geometric:
    - Implements Message Passing Neural Network
    - Contains implementation of several recent papers
    - Easy to understand entire codebase
  - Deep Graph Library:
    - Works with MXNet/Gluon and PyTorch
    - Allows batching of computations
    - Good scalability to large graphs (10 million nodes)

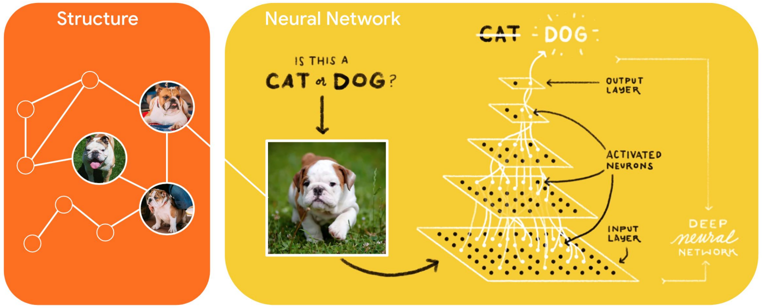
Implementing GCNs

- TensorFlow
  - Graph Nets Library
    - From DeepMind
    - Doesn't contain implementation of latest papers
      [as of 4/11/19]

Implementing GCNs

- TensorFlow
  - Graph Nets Library
    - From DeepMind
    - Doesn't contain implementation of latest papers [as of 4/11/19]

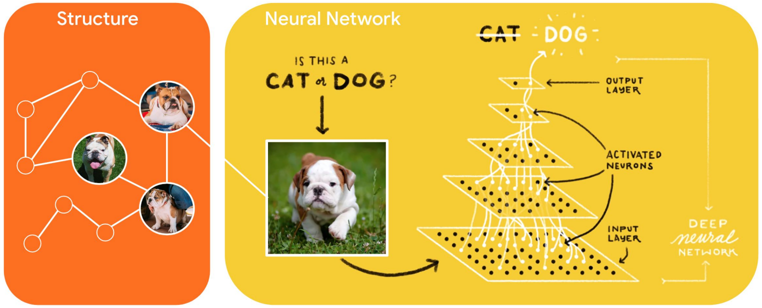
  - Neural Structured Learning
    - Keras API
    - Allows to construct graphs



Implementing GCNs

- TensorFlow
  - Graph Nets Library
    - From DeepMind
    - Doesn’t contain implementation of latest papers [as of 4/11/19]

  - Neural Structured Learning
    - Keras API
    - Allows to construct graphs



  - Other starting points:
    - https://github.com/tkipf/gcn
    - https://github.com/svjan5/GNNs-for-NLP

GCNs for Node Classification

- https://github.com/svjan5/GCN-Tutorial-EMNLP19
  - tf_gcn.py contains simplified implementation of first-order approximation of GCN model proposed by Kipf et. al. (2016)
  - pytorch_gcn.py is pytorch equivalent of tf_gcn.py implemented using pytorch-geometric.
  - Relevant references for getting started with GCNs and list of recent papers [link].

Sample GCN Code

```python
def GCNLayer(self, gcn_in, adj_mat, input_dim, output_dim, act, dropout, num_nonzero, input_sparse=False, name='GCN'):
        """
        GCN Layer Implementation

        Parameters
        ----------
        gcn_in:         Input to GCN Layer
        adj_mat:        Adjacency matrix
        input_dim:      Dimension of input to GCN Layer
        output_dim:     Dimension of output of GCN Layer
        act:            Activation function used
        dropout:        Dropout probability
        num_numzero:    Number of non-zero elements in input features (used when input_sparse=True)
        input_sparse:   Whether input features are sparse or not
        name;           Name of the Layer

        Returns
        -------
        Output of GCN Layer

        """
```

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$

Sample GCN Code

```
def GCNLayer(self, gcn_in, adj_mat, input_dim, output_dim, act, dropout, num_nonzero, input_sparse=False, name='GCN'):

    with tf.name_scope(name):
        with tf.variable_scope('{}_vars'.format(name)) as scope:
            wts  = tf.get_variable('weights', [input_dim, output_dim],
            bias = tf.get_variable('bias',    [output_dim],
            self.l2_vars.extend([wts, bias])

        if input_sparse:
            gcn_in  = self.sparse_dropout(gcn_in, 1 - dropout, num_nonzero)
            pre_sup = tf.sparse_tensor_dense_matmul(gcn_in,  wts)
        else:
            gcn_in = tf.nn.dropout(gcn_in, 1-dropout)
            pre_sup = tf.matmul(gcn_in,  wts)

        support = tf.sparse_tensor_dense_matmul(adj_mat, pre_sup)

    return act(support)
```

Define Parameters

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \boxed{W} x_u + \boxed{b}\right), \quad \forall v \in \mathcal{V}.$$

Sample GCN Code

```
def GCNLayer(self, gcn_in, adj_mat, input_dim, output_dim, act, dropout, num_nonzero, input_sparse=False, name='GCN'):
    with tf.name_scope(name):
        with tf.variable_scope('{}_vars'.format(name)) as scope:
            wts  = tf.get_variable('weights', [input_dim, output_dim],
            bias = tf.get_variable('bias',    [output_dim],
            self.l2_vars.extend([wts, bias])

        if input_sparse:
            gcn_in  = self.sparse_dropout(gcn_in, 1 - dropout, num_nonzero)
            pre_sup = tf.sparse_tensor_dense_matmul(gcn_in,  wts)
        else:
            gcn_in = tf.nn.dropout(gcn_in, 1-dropout)
            pre_sup = tf.matmul(gcn_in,  wts)

        support = tf.sparse_tensor_dense_matmul(adj_mat, pre_sup)

    return act(support)
```

Define Parameters

Multiplying node features and filters

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \boxed{W x_u} + b\right), \quad \forall v \in \mathcal{V}.$$

Sample GCN Code

```
def GCNLayer(self, gcn_in, adj_mat, input_dim, output_dim, act, dropout, num_nonzero, input_sparse=False, name='GCN'):

    with tf.name_scope(name):
        with tf.variable_scope('{}_vars'.format(name)) as scope:
            wts  = tf.get_variable('weights', [input_dim, output_dim],
            bias = tf.get_variable('bias',    [output_dim],
            self.l2_vars.extend([wts, bias])

        if input_sparse:
            gcn_in  = self.sparse_dropout(gcn_in, 1 - dropout, num_nonzero)
            pre_sup = tf.sparse_tensor_dense_matmul(gcn_in,  wts)
        else:
            gcn_in = tf.nn.dropout(gcn_in, 1-dropout)
            pre_sup = tf.matmul(gcn_in,  wts)

        support = tf.sparse_tensor_dense_matmul(adj_mat, pre_sup)

    return act(support)
```

Define Parameters

Multiplying node features and filters

$$h_v = f\left(\boxed{\frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} W x_u} + b\right), \quad \forall v \in \mathcal{V}.$$

GCNs for Node Classification

```
class KipfGCN(torch.nn.Module):
        def __init__(self, data, num_class, params):
                super(KipfGCN, self).__init__()
                self.p     = params
                self.data  = data
                self.conv1 = GCNConv(self.data.num_features, self.p.gcn_dim, cached=True)
                self.conv2 = GCNConv(self.p.gcn_dim, num_class,  cached=True)

        def forward(self, x, edge_index):
                x               = F.relu(self.conv1(x, edge_index))
                x               = F.dropout(x, p=self.p.dropout, training=self.training)
                x               = self.conv2(x, edge_index)
                return F.log_softmax(x, dim=1)
```

GCNs on Multiple Small Graphs

- **Case:** Each data point has a different graph
  - Applying Syntactic GCNs for text classification
  - How to do batching?

$$\begin{bmatrix} \boxed{B_1} & 0 & 0 \\ 0 & \boxed{B_2} & 0 \\ 0 & 0 & \boxed{B_3} \end{bmatrix}$$

One element in the batch

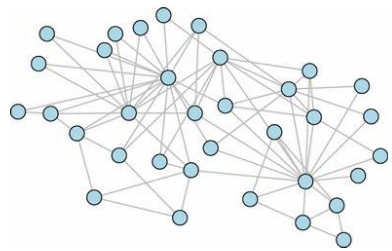
Represent Adjacency matrix as a block-diagonal matrix

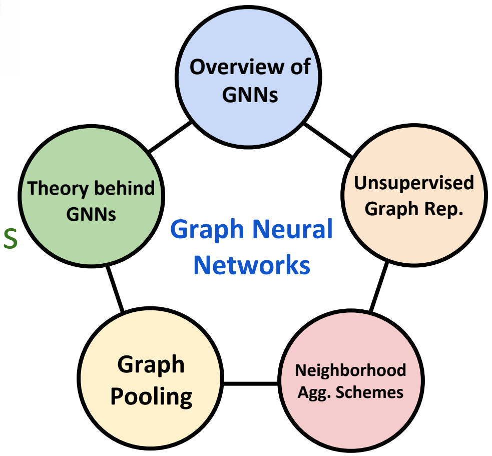
$$X = [X_1; X_2; X_3]$$

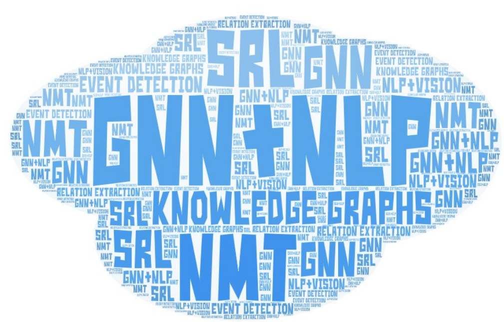
Concatenate Node Features

Tutorial Outline

- **Introduction**
  - ✓ Motivation
  - ✓ GNN Foundation
- **Methods**
  - ✓ Introduction to GNNs
  - ✓ Graph Pooling Unsupervised Learning using GNNs
  - ✓ Neighborhood Aggregation in GNNs
  - ✓ Other GNN Variants
- **Implementing GCNs**
- **Applications**
  - Semantic Role Labeling, NMT
  - Text Classification, Extraction
  - Knowledge Graphs
  - Vision + NLP
- **Open Problems and Conclusion**

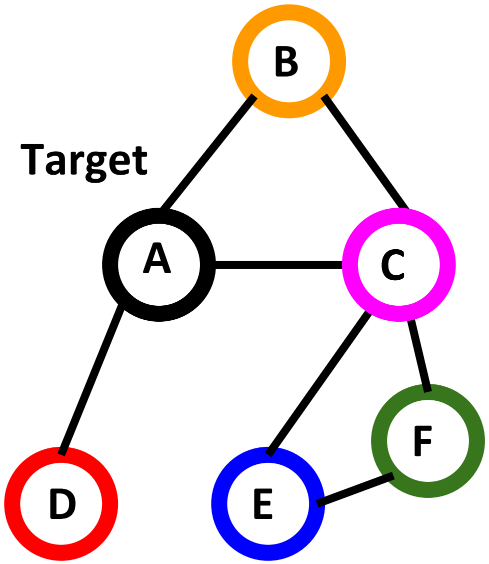






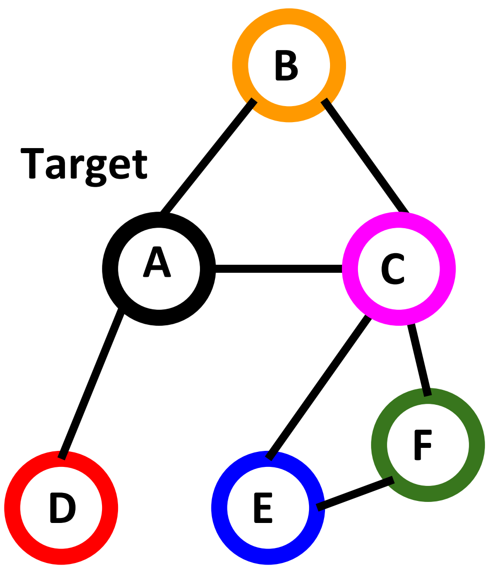
?

Two Popular GNNs for NLP



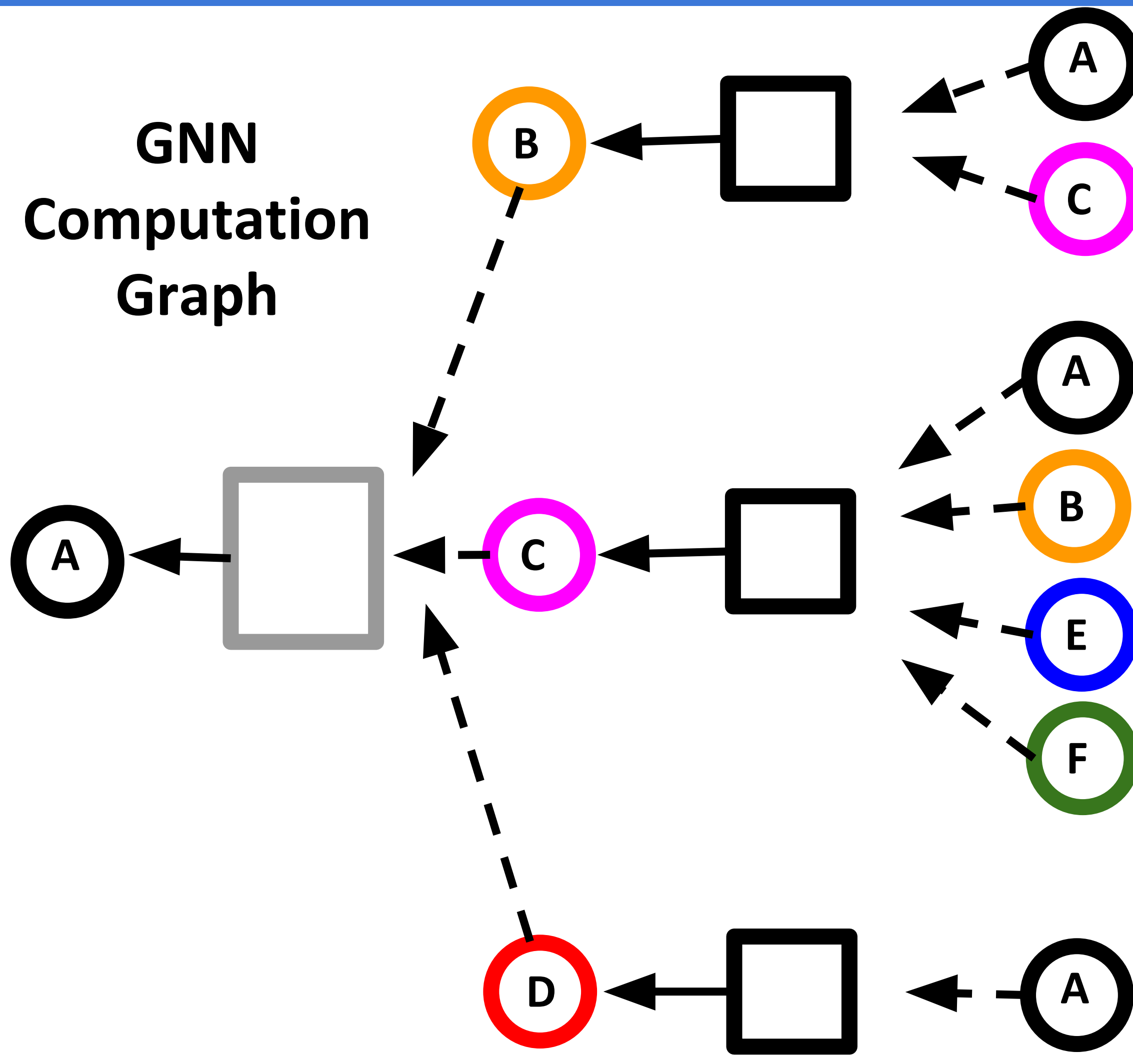
Input Graph

Two Popular GNNs for NLP

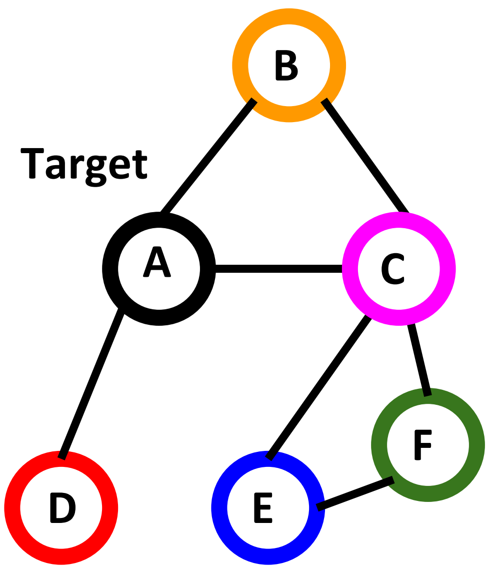


Input Graph

GNN Computation Graph

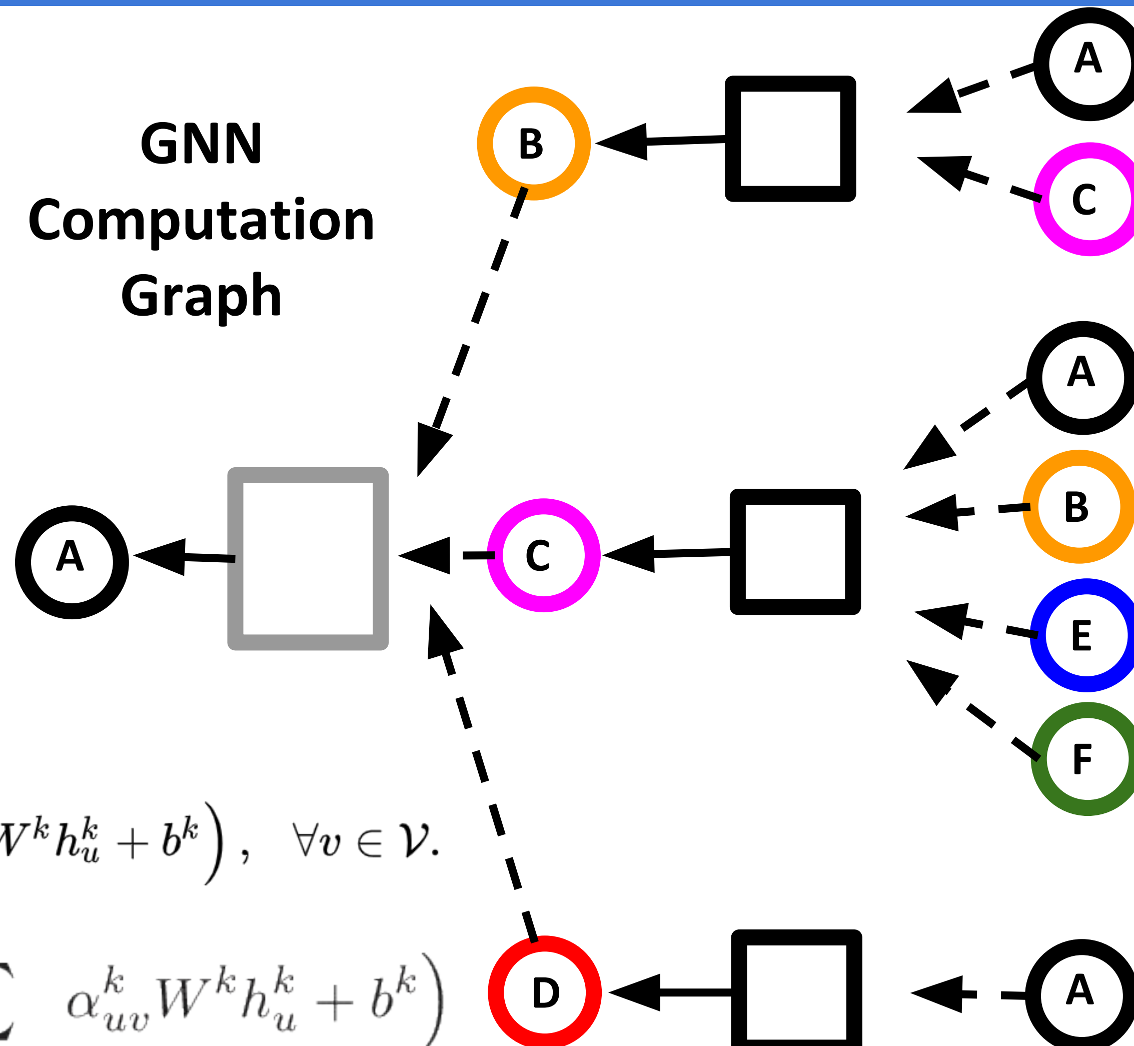


Two Popular GNNs for NLP



Input Graph

GNN Computation Graph



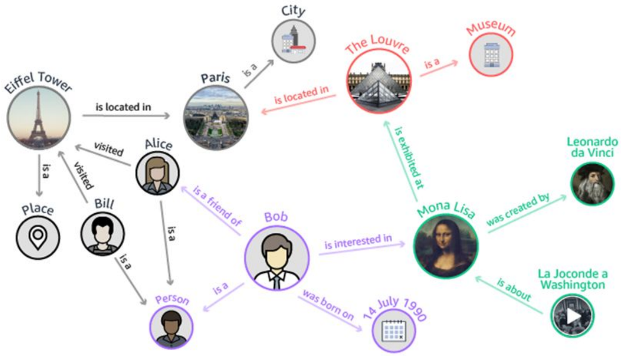
GCN $h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W^k h_u^k + b^k\right), \quad \forall v \in \mathcal{V}.$

GAT $$h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} \alpha_{uv}^k W^k h_u^k + b^k\right)$$

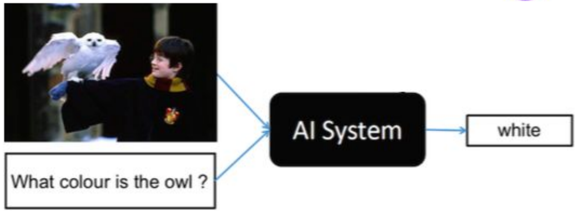
Applications of Graph Neural Nets

- Semantic Role Labelling, Machine Translation
- Text Classification, Extraction

- Knowledge Graphs



- Vision + NLP



Semantic Role Labelling (SRL) [Marcheggiani et al., EMNLP'17]

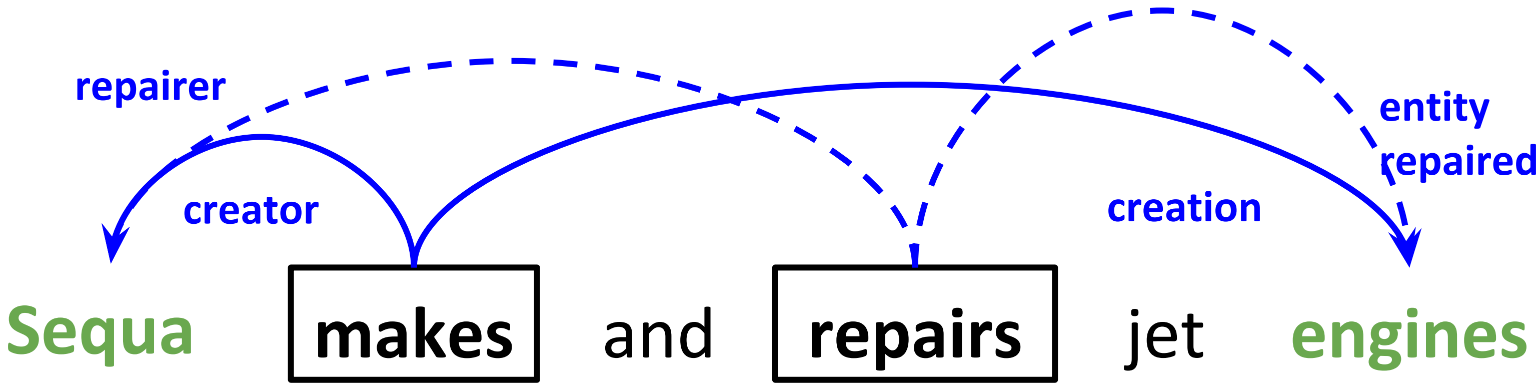
Sequa makes and repairs jet engines

Semantic Role Labelling (SRL) [Marcheggiani et al., EMNLP'17]

Sequa $\boxed{\textbf{makes}}$ and $\boxed{\textbf{repairs}}$ jet **engines**

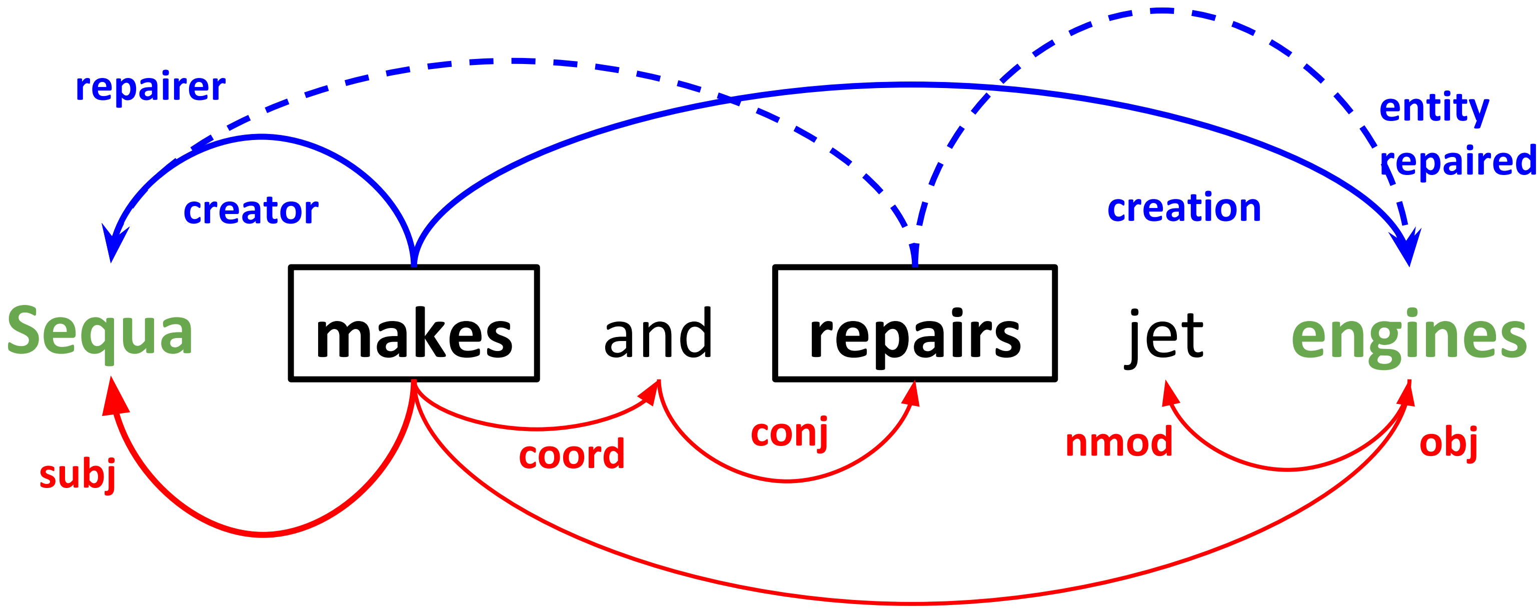
- Discover **predicates**
- Identify **arguments**

Semantic Role Labelling (SRL) [Marcheggiani et al., EMNLP'17]



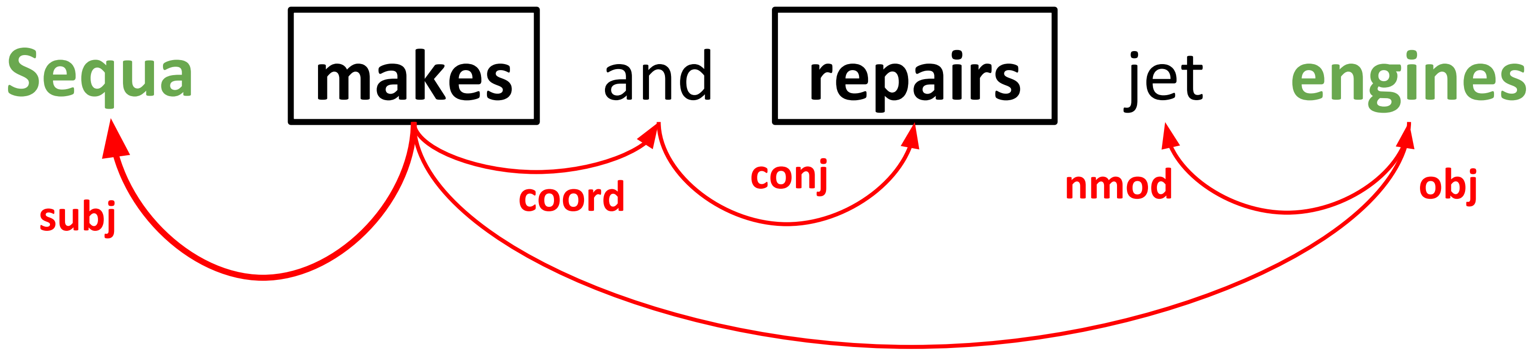
- Discover **predicates**
- Identify **arguments**, their **semantic roles**
- **Part of std. NLP pipeline for QA, IE, etc.**

SRL and Syntax [Marcheggiani and Titov, EMNLP'17]



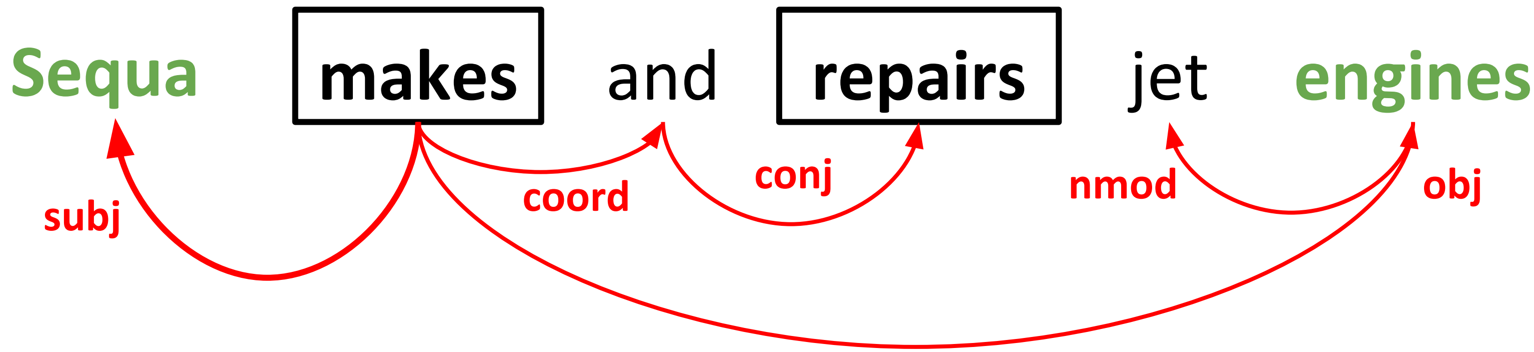
- **Syntax** mirrors **semantics**
- Exploit syntax using convolution

Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



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

Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



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

word emb of v

edge-wise gating
$g_{u,v} = \sigma\Big(\hat{w}_{d(u,v)} \; h_u + \hat{b}_{l(u,v)} \Big)$

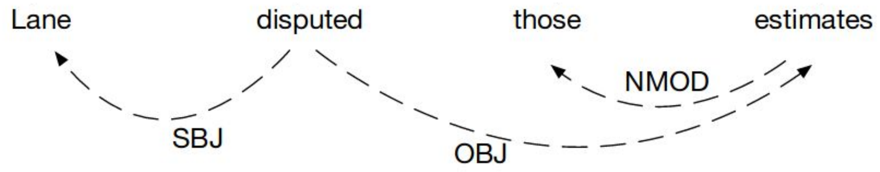
weight of direction

bias of label + direction

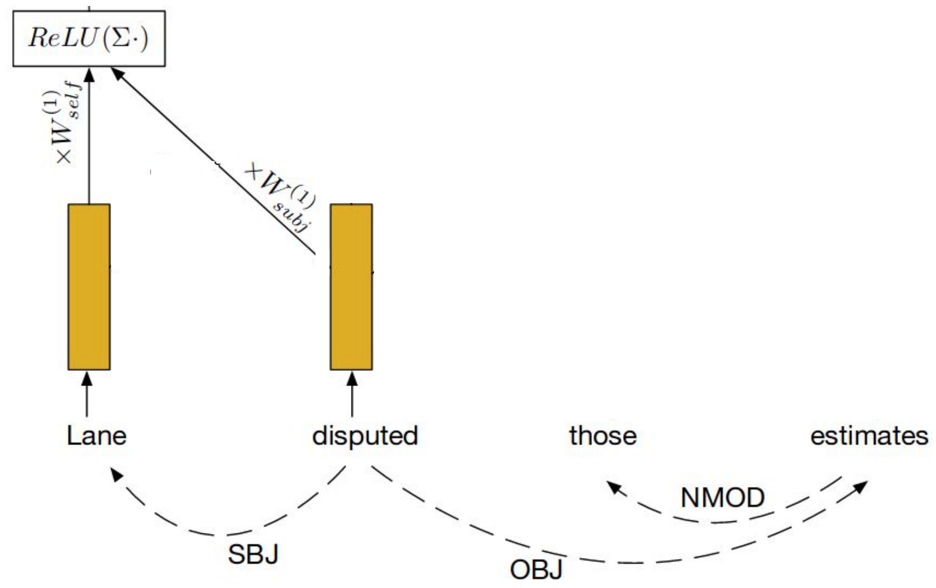
Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]

Lane disputed those estimates

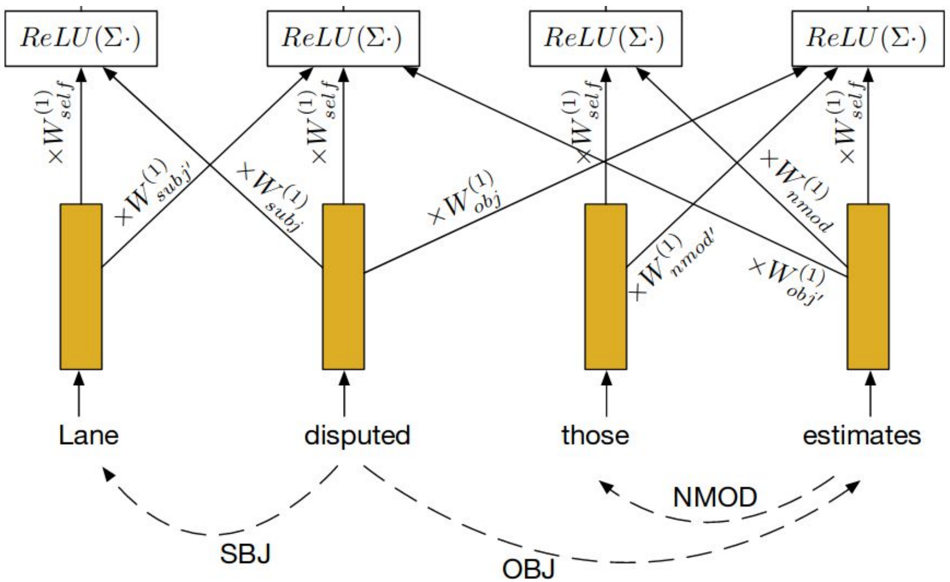
Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



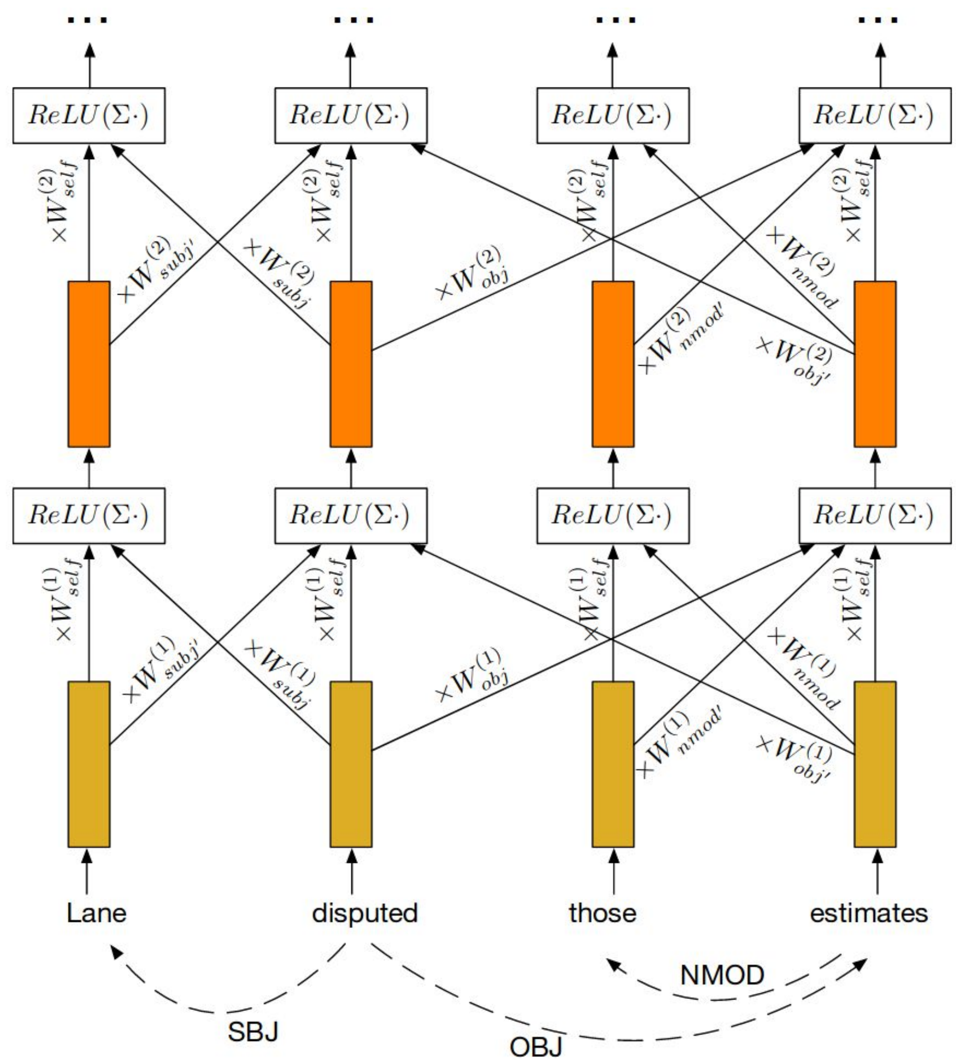
Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



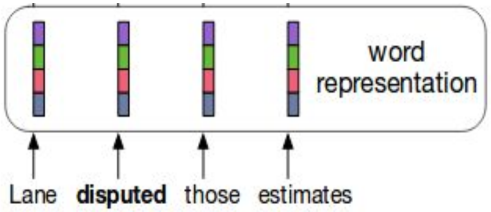
Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



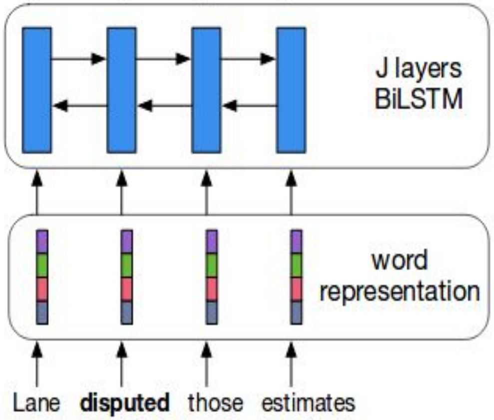
Syntactic graph convolution [Marcheggiani et. al., EMNLP'17]



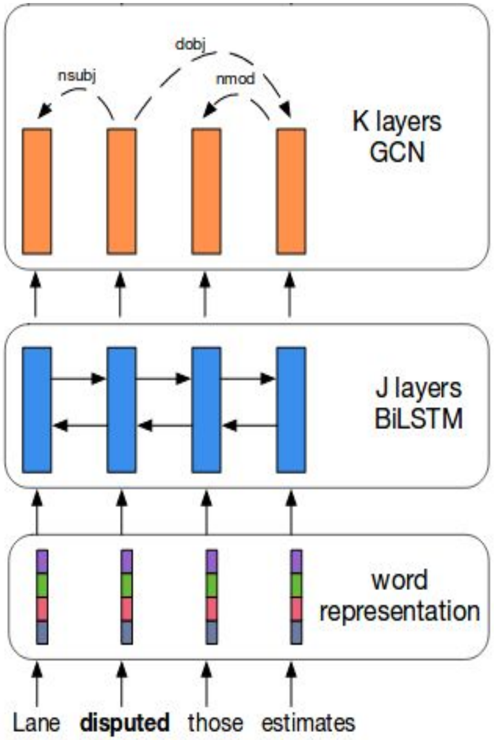
Syntactic GCN for SRL [Marcheggiani and Titov, EMNLP'17]



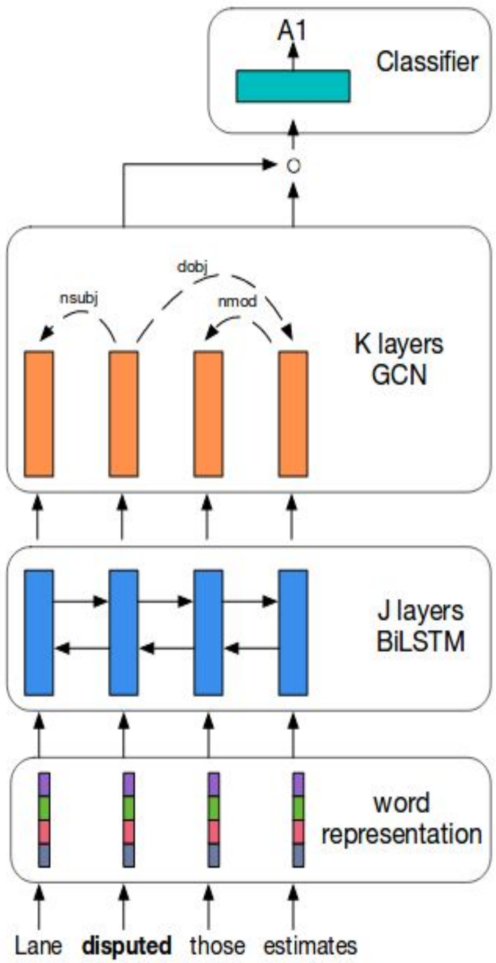
Syntactic GCN for SRL [Marcheggiani and Titov, EMNLP'17]



Syntactic GCN for SRL [Marcheggiani and Titov, EMNLP'17]



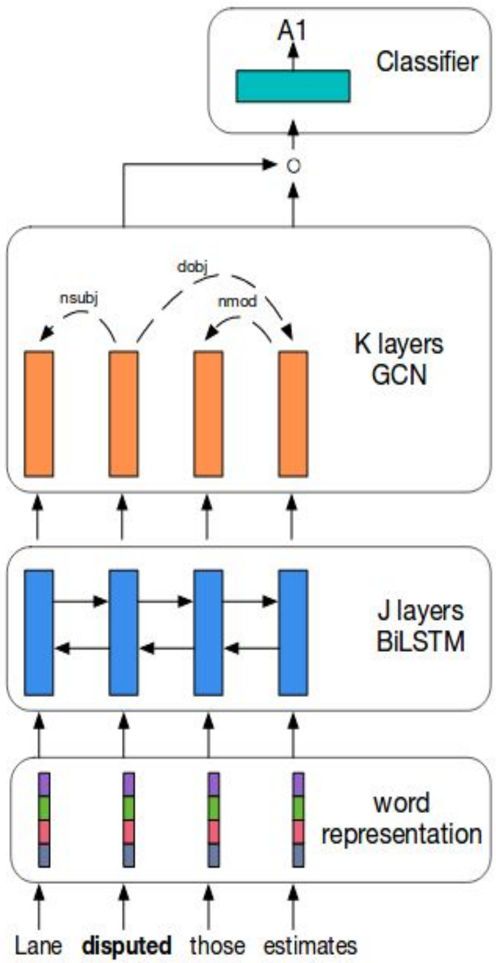
Syntactic GCN for SRL [Marcheggiani and Titov, EMNLP'17]



trained with cross-entropy

Arguments far away come closer because of syntactic arcs

Syntactic GCN for SRL [Marcheggiani and Titov, EMNLP'17]

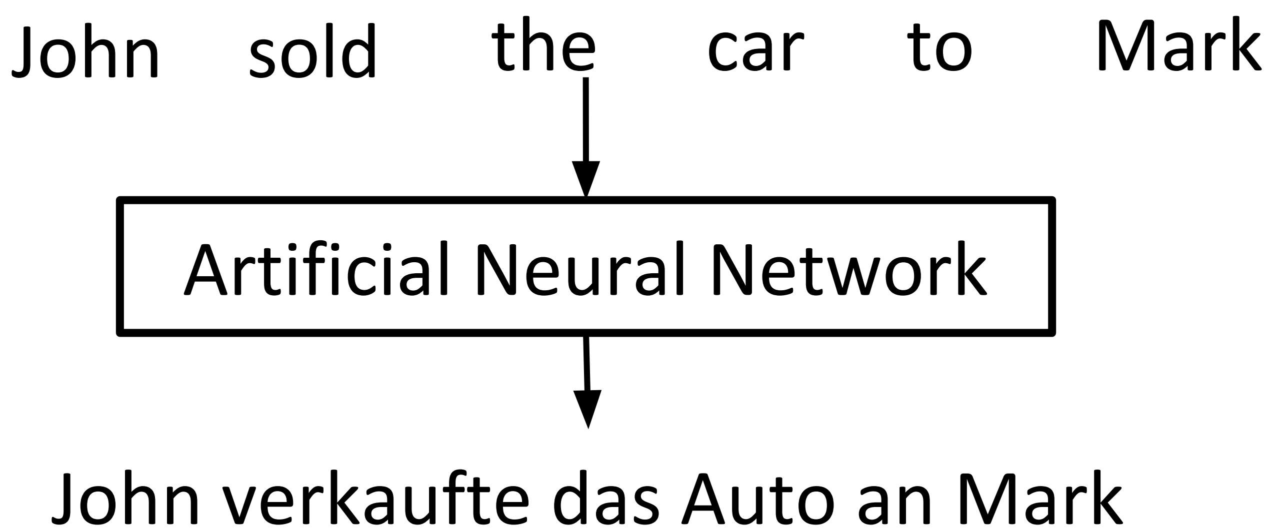


F1 on CoNLL-2009

BiLSTM	82.7
BiLSTM + GCN	**83.3**

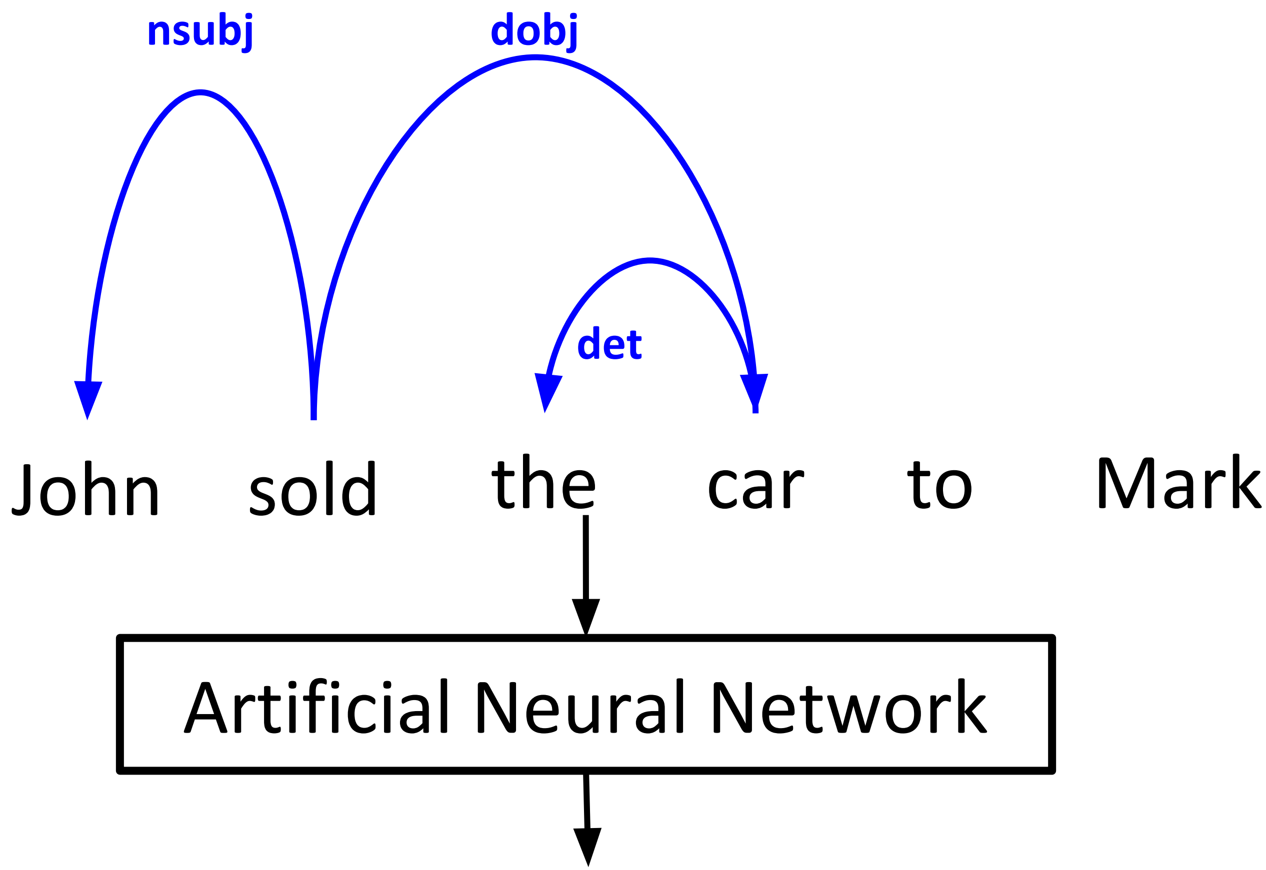
- GCN integrates syntax, context
- GCN, LSTM complement each other

Neural Machine Translation (NMT) [Bastings et al., EMNLP'17]



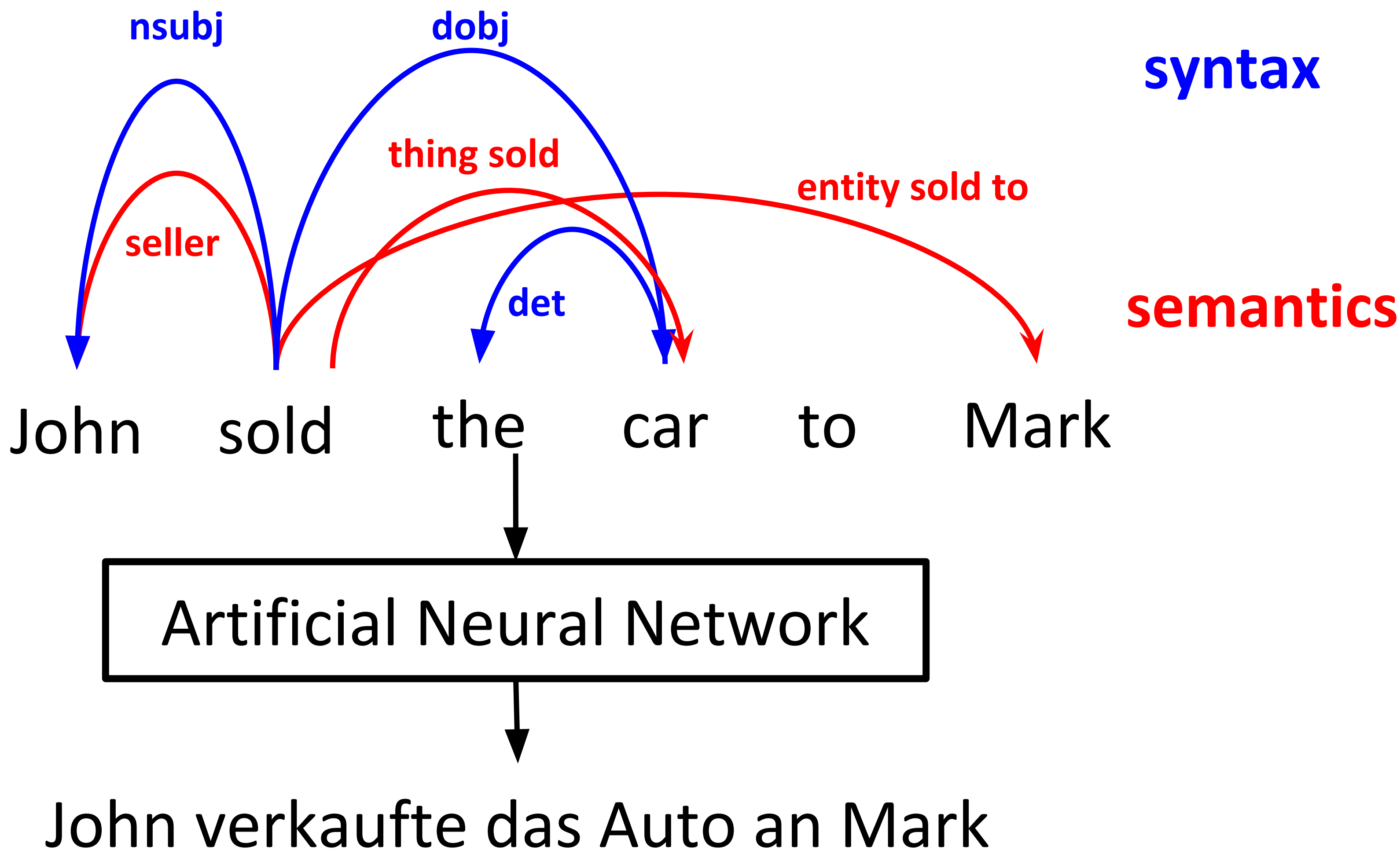
Neural Machine Translation (NMT) [Bastings et al., EMNLP'17]

syntax



John verkaufte das Auto an Mark

Neural Machine Translation (NMT) [Bastings et al., EMNLP'17]



GCN on NMT [Bastings et al., EMNLP'17, Marcheggiani et al., NAACL'18]

English - German NMT on News Commentary

Encoder	BLEU
Bag-of-words	9.5
Bag-of-words + Syntactic GCN	12.2
BiGRU	14.9
BiGRU + Syntactic GCN	16.1
BiGRU + Semantic GCN	15.6
BiGRU + (Semantic + Syntactic) GCN	15.8

- Attention-based decoder of [Bahdanau et al., ICLR 15]
- BiGRU, GCN complement each other

Addressing GCN Limitations [Beck et al., ACL'18]

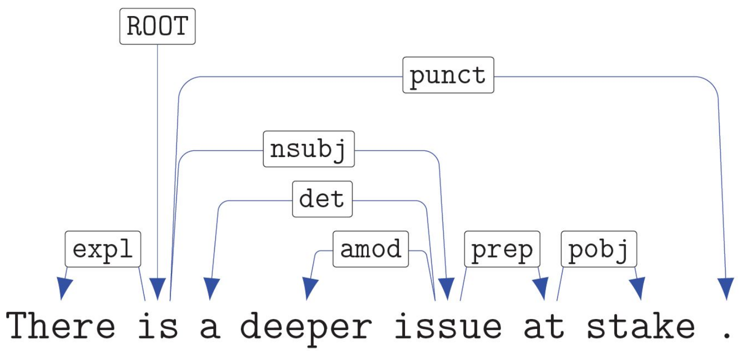
- **Limitations**
  - ✗ Parameters increase quadratically with # edge labels
  - ✗ No parameter sharing across layers
  - ✗ Edge labels are not encoded

Addressing GCN Limitations [[Beck et al., ACL'18]]

- **Limitations**
  - ✗ Parameters increase quadratically with # edge labels
  - ✗ No parameter sharing across layers
  - ✗ Edge labels are not encoded
- **GraphGRU (GGNN)**
  - ✓ Best of BiGRU + GCN worlds
  - ✓ Arbitrary # layers w/o increasing parameters

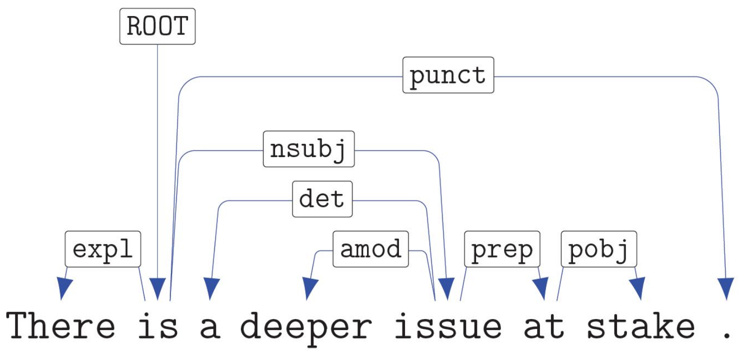
Levi graph [Beck et al., ACL'18]

E.g. syntactic dependency



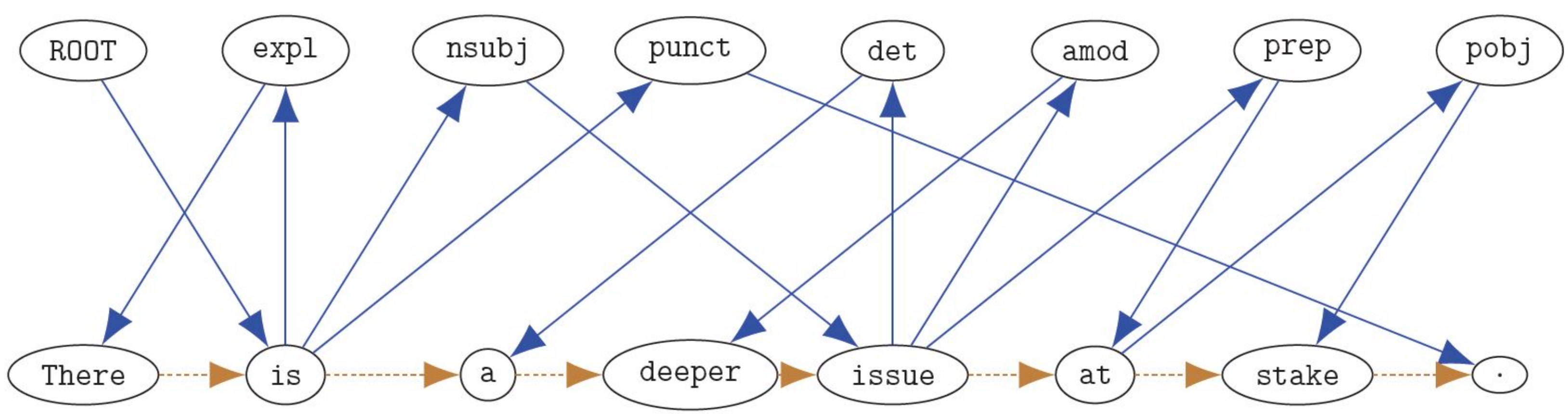
Levi graph [Beck et al., ACL'18]

E.g. syntactic dependency



- **An edge for every (node, edge)**

✓ **Edge labels have hidden emb**



Levi graph

GraphGRU on Levi graph [Beck et al., ACL'18]

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

$$\mathbf{r}_v^t = \sigma\left(c_v^r \quad \mathbf{W}_{\ell_e}^r \mathbf{h}_u^{(t-1)} + \mathbf{b}_{\ell_e}^r \right)$$ **reset gate**

$$\mathbf{z}_v^t = \sigma\left(c_v^z \quad \mathbf{W}_{\ell_e}^z \mathbf{h}_u^{(t-1)} + \mathbf{b}_{\ell_e}^z \right)$$ **update gate**

$$\widetilde{\mathbf{h}}_v^t = \rho\left(c_v \quad \mathbf{W}_{\ell_e} \left(\mathbf{r}_u^t \odot \mathbf{h}_u^{(t-1)} \right) + \mathbf{b}_{\ell_e} \right)$$

$$\mathbf{h}_v^t = (1 - \mathbf{z}_v^t) \odot \mathbf{h}_v^{(i-1)} + \mathbf{z}_v^t \odot \widetilde{\mathbf{h}}_v^t$$

GraphGRU on Levi graph [[Beck et al., ACL'18]]

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

$$\mathbf{r}_v^t = \sigma\left(c_v^r \sum_{u \in \mathcal{N}_v} \mathbf{W}_{\ell_e}^r \mathbf{h}_u^{(t-1)} + \mathbf{b}_{\ell_e}^r\right)$$ **reset gate**

$$\mathbf{z}_v^t = \sigma\left(c_v^z \sum_{u \in \mathcal{N}_v} \mathbf{W}_{\ell_e}^z \mathbf{h}_u^{(t-1)} + \mathbf{b}_{\ell_e}^z\right)$$ **update gate**

$$\widetilde{\mathbf{h}}_v^t = \rho\left(c_v \sum_{u \in \mathcal{N}_v} \mathbf{W}_{\ell_e}\left(\mathbf{r}_u^t \odot \mathbf{h}_u^{(t-1)}\right) + \mathbf{b}_{\ell_e}\right)$$

$$\mathbf{h}_v^t = (1 - \mathbf{z}_v^t) \odot \mathbf{h}_v^{(i-1)} + \mathbf{z}_v^t \odot \widetilde{\mathbf{h}}_v^t$$

LeviGraphGRU on NMT [Beck et al., ACL'18]

English - German NMT on News Commentary

Encoder	BLEU
Bag-of-words	9.5
Bag-of-words + Syntactic GCN	12.2
BiGRU	14.9
BiGRU + Syntactic GCN	16.1
BiGRU + Semantic GCN	15.6
BiGRU + (Semantic + Syntactic) GCN	15.8
Bag-of-words + Levi GraphGRU	**19.6**

Embedding edge labels is effective

Summary of GNNs for SRL, NMT

- **Takeaways**
  - **Syntax, semantics** helpful for NLP esp. NMT
  - **Levi graph** enables edge label representations

Summary of GNNs for SRL, NMT

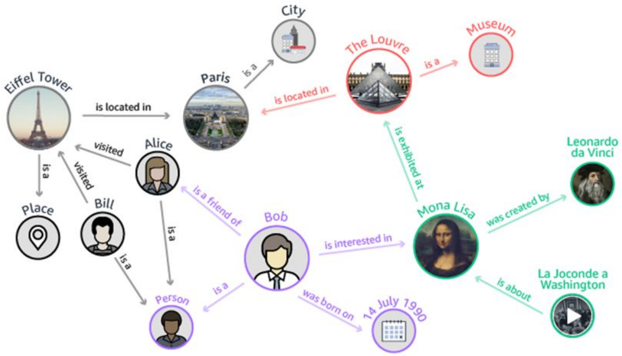
- **Takeaways**
  - **Syntax, semantics** helpful for NLP esp. NMT
  - **Levi graph** enables edge label representations
- **Future directions**
  - Exploit **semantics** for other tasks
  - Edge labels, nodes **share** same space in Levi graph
    - Not ideal, use decoupling [Kearnes et al., JCAMD'16]

Applications of Graph Neural Nets

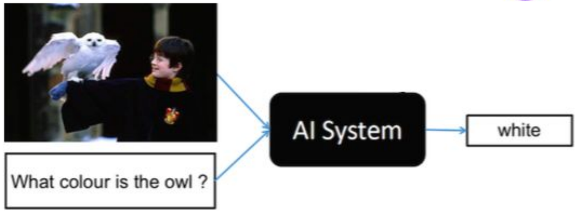
✔ Semantic Role Labelling, Machine Translation

- Text Classification, Extraction

- Knowledge Graphs



- Vision + NLP



?

GNNs for Text Classification, Extraction

Event Detection / Timestamping

[RE-NET]	ICLR'19 WS
[JMEE]	EMNLP'18
[AD3]	EMNLP'18
[NeuralDater]	ACL'18
[AAP]	AAAI'18

GNNs for Text Classification, Extraction

GNNs for Text Classification, Extraction

Event Detection / Timestamping

RE-NET	ICLR'19 WS
JMEE	EMNLP'18
AD3	EMNLP'18
NeuralDater	ACL'18
AAP	AAAI'18

GNNs for Text Classification, Extraction

Relation Extraction

EOG	EMNLP'19
INTERE	ACL'19
GraphRel	ACL'19
AG-GCN	ACL'19
ENTREL	ACL'19
GP-GNN	ACL'19
VRD	NAACL'19
GraphIE	NAACL'19
KATT	NAACL'19
CGCN	EMNLP'18
RESIDE	EMNLP'18

GNNs for Text Classification, Extraction

Event Detection / Timestamping

RE-NET	ICLR'19 WS
JMEE	EMNLP'18
AD3	EMNLP'18
NeuralDater	ACL'18
AAP	AAAI'18

GNNs for Text Classification, Extraction

Relation Extraction

EOG	EMNLP'19
INTERE	ACL'19
GraphRel	ACL'19
AG-GCN	ACL'19
ENTREL	ACL'19
GP-GNN	ACL'19
VRD	NAACL'19
GraphIE	NAACL'19
KATT	NAACL'19
CGCN	EMNLP'18
RESIDE	EMNLP'18

Sentiment Analysis

TDGAT	EMNLP'19
ASGCN	EMNLP'19
DialogueGCN	EMNLP'19

GNNs for Text Classification, Extraction

GNNs for Text Classification, Extraction

Event Detection / Timestamping

RE-NET	ICLR'19 WS
JMEE	EMNLP'18
AD3	EMNLP'18
NeuralDater	ACL'18
AAP	AAAI'18

Word Embedding / Text Classification

HGAT	EMNLP'19
SynGCN	ACL'19
TextGCN	AAAI'19
HR-GCN	WWW'18

Sentiment Analysis

TDGAT	EMNLP'19
ASGCN	EMNLP'19
DialogueGCN	EMNLP'19

Relation Extraction

EOG	EMNLP'19
INTERE	ACL'19
GraphRel	ACL'19
AG-GCN	ACL'19
ENTREL	ACL'19
GP-GNN	ACL'19
VRD	NAACL'19
GraphIE	NAACL'19
KATT	NAACL'19
CGCN	EMNLP'18
RESIDE	EMNLP'18

Event Detection [Nguyen and Grishman, AAAI 18]

The police officer, who fired into a car full of teenagers, was fired yesterday

Event Detection [Nguyen and Grishman, AAAI'18]

ATTACK

The police officer, who **fired** into a car full of teenagers, was **fired** yesterday

END-POSITION

- Identify **event triggers**
- Identify **event type** for each trigger

Event Detection [Nguyen and Grishman, AAAI'18]

ATTACK

The police officer, who **fired** into a car full of teenagers, was **fired** yesterday

END-POSITION

- Identify **event triggers**
- Identify **event type** for each trigger

BiLSTM	70.5
BiLSTM + Syntactic GCN	**71.4**

F1 the ACE 2005 dataset

GCN, LSTM complement each other

Multiple Events Extraction (MEE) [Liu et al., EMNLP'18]

He **left** the company

END-POSITION

Multiple Events Extraction (MEE) [Liu et al., EMNLP'18]

He **left** the company, and planned to **go** home directly

END-POSITION ✗ **TRANSPORT**

TRANSPORT ✓

Multiple Events Extraction (MEE) [Liu et al., EMNLP'18]

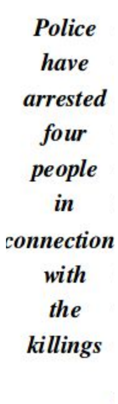
He **left** the company, and planned to **go** home directly

END-POSITION ✗ **TRANSPORT**

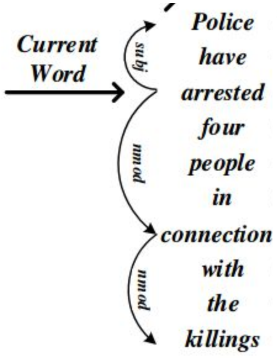
TRANSPORT ✓

- **co-occurring triggers** reduce ambiguity
- common in real-world (e.g. injure, die co-occur often)
- 26% in **ACE 2005 data**

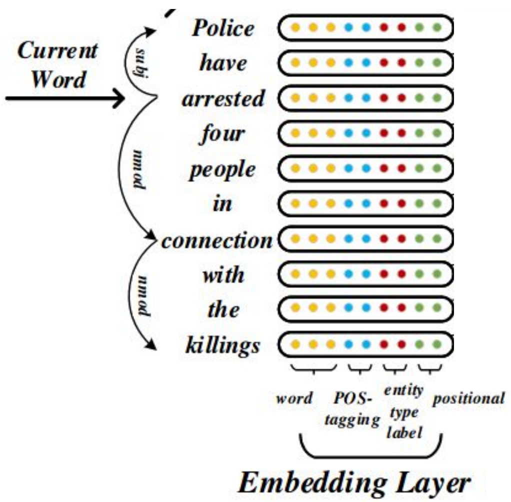
Joint MEE (JMEE) [Liu et al., EMNLP'18]



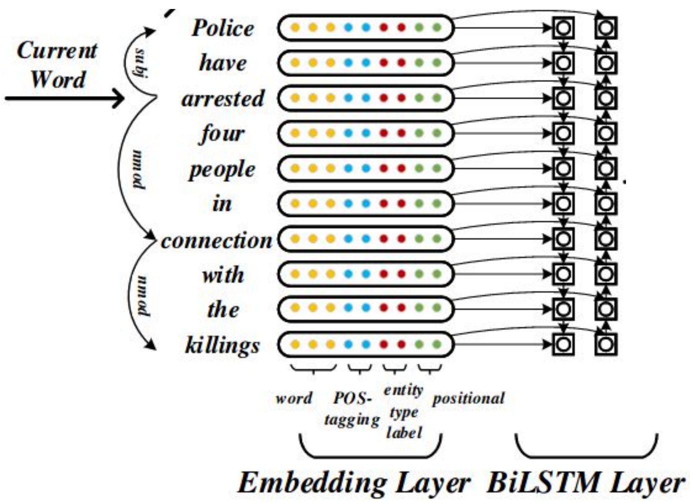
Joint MEE (JMEE) [Liu et al., EMNLP'18]



Joint MEE (JMEE) [Liu et al., EMNLP'18]

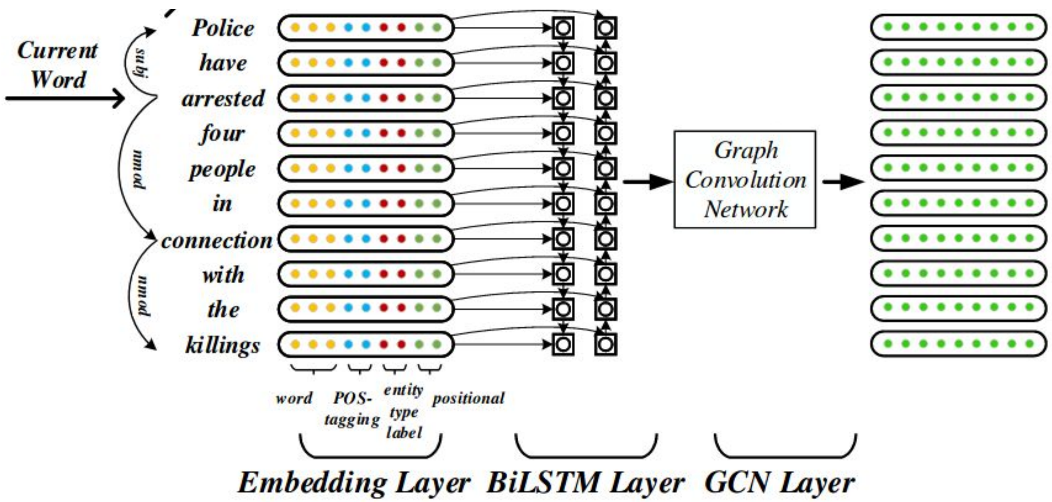


Joint MEE (JMEE) [Liu et al., EMNLP'18]



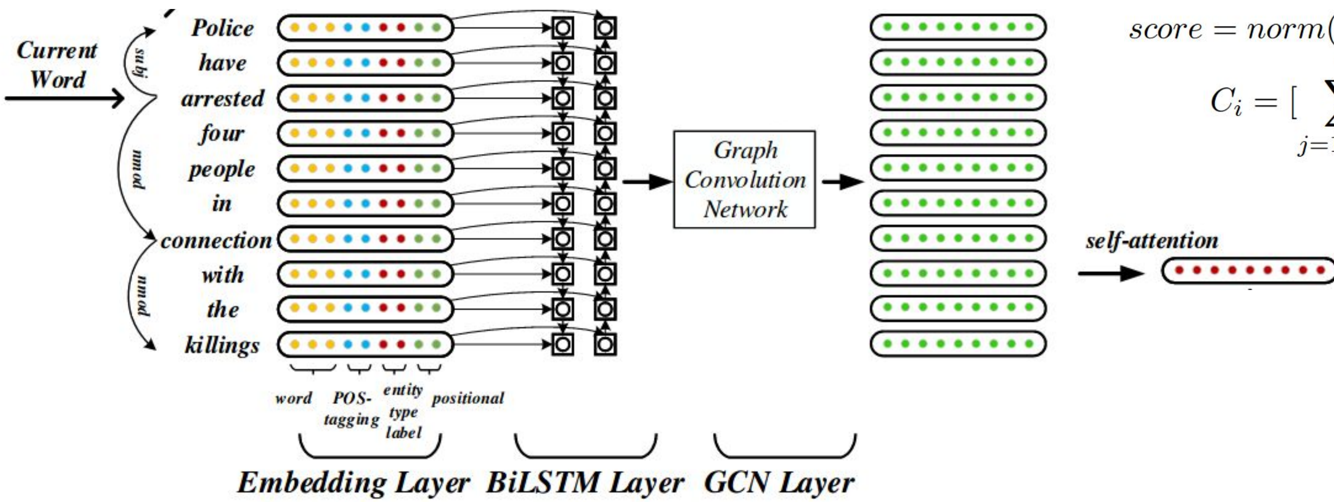
$$\overrightarrow{p}_t = \overrightarrow{LSTM}(\overrightarrow{p}_{t-1}, x_t)$$
$$\overleftarrow{p}_t = \overleftarrow{LSTM}(\overleftarrow{p}_{t-1}, x_t)$$

Joint MEE (JMEE) [Liu et al., EMNLP'18]



$$h_v^{(k+1)} = f\Big(\sum_{u \in \mathcal{N}(v)} g_{u,v}^{(k)} \big(W_{K(u,v)}^{(k)} h_u^{(k)} + b_{K(u,v)}^{(k)}\big)\Big)$$

Joint MEE (JMEE) [Liu et al., EMNLP'18]

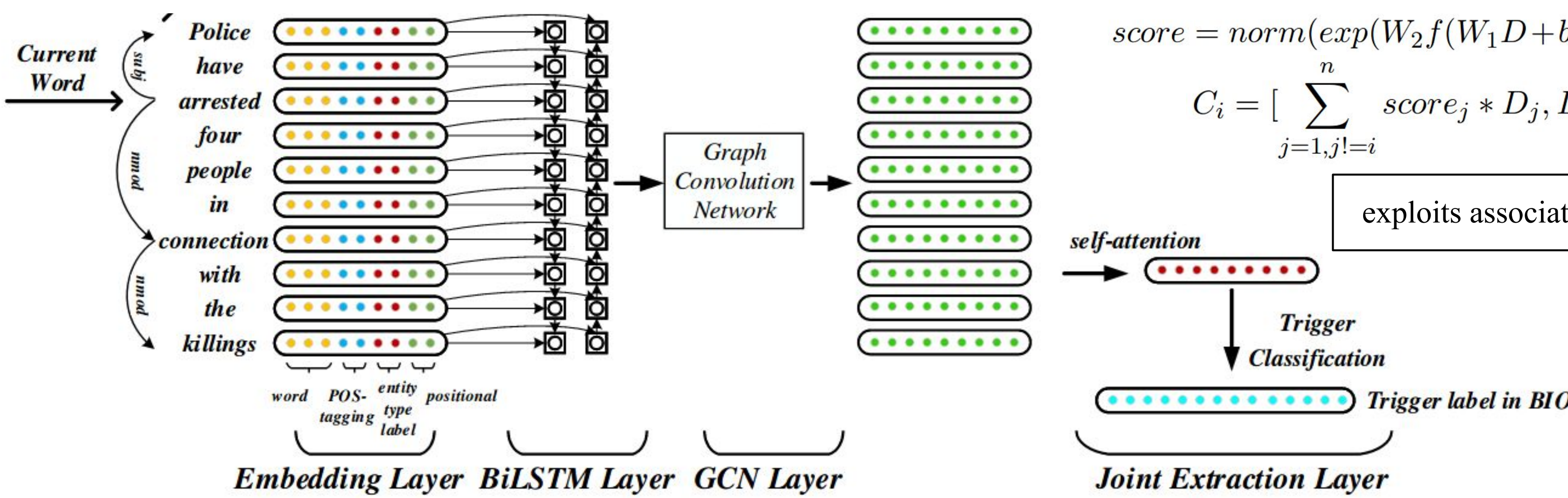


$$score = norm(exp(W_2 f(W_1 D + b_1) + b_2))$$

$$C_i = [\sum_{j=1, j!=i}^{n} score_j * D_j, D_i]$$

exploits associations b/w triggers

Joint MEE (JMEE) [[Liu et al., EMNLP'18]]



$$score = norm(exp(W_2 f(W_1 D + b_1) + b_2))$$

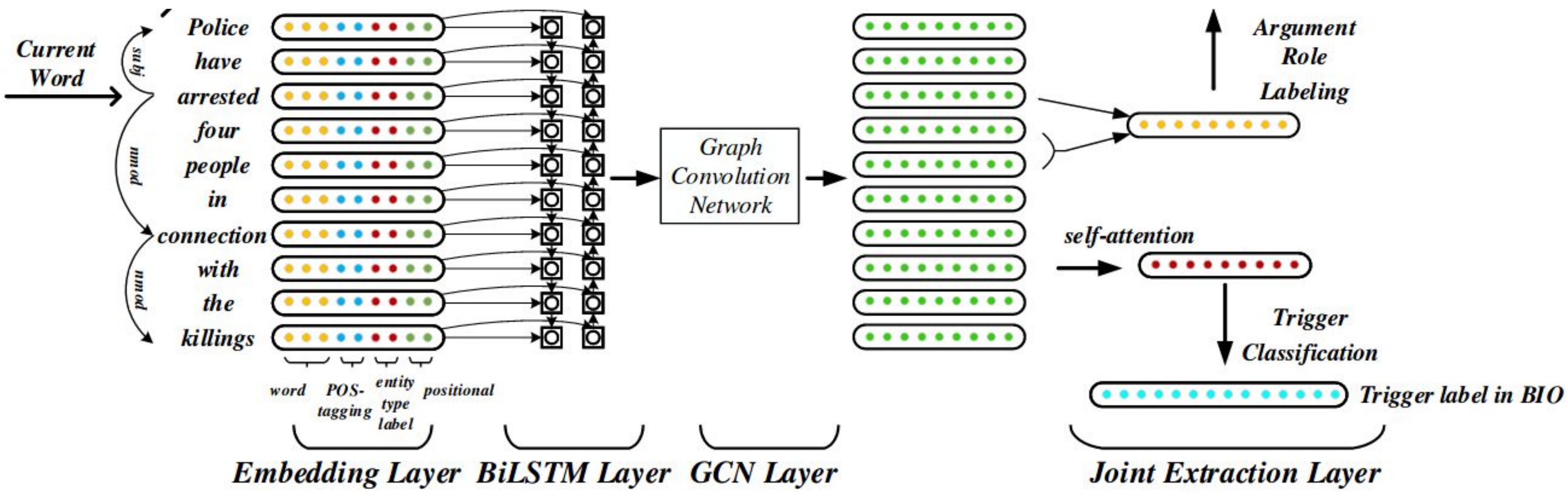
$$C_i = [\sum_{j=1, j!=i}^{n} score_j * D_j, D_i]$$

exploits associations b/w triggers

$$\overline{C}_i = f(W_c C_i + b_c)$$

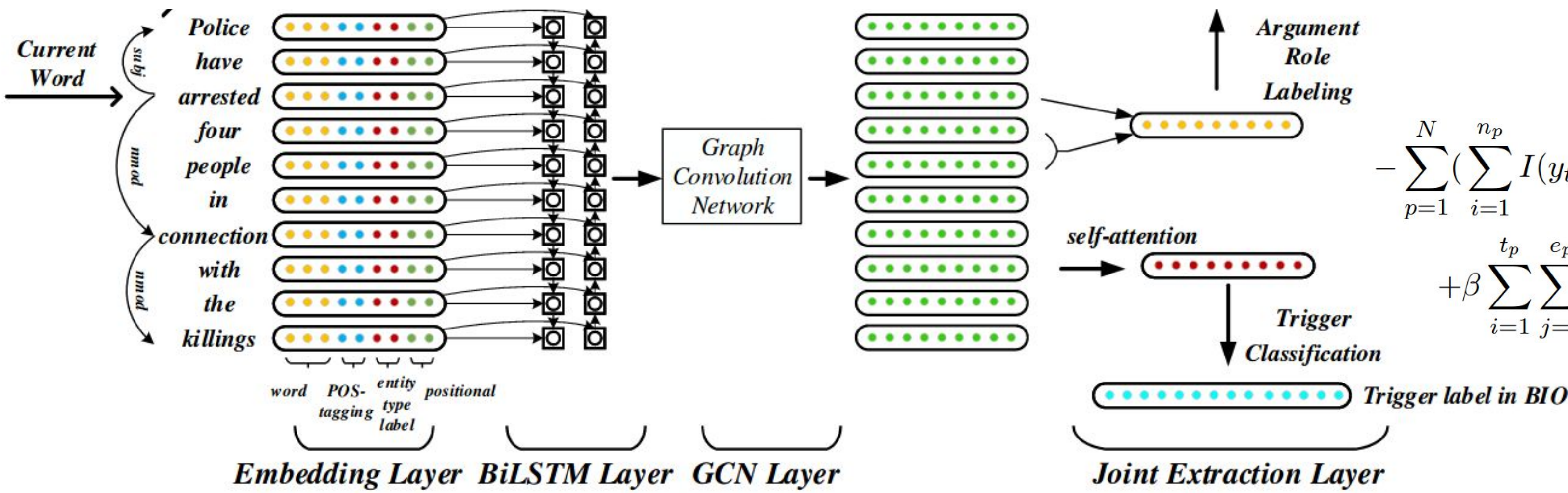
Joint MEE (JMEE) [Liu et al., EMNLP'18]

$$y_{a_{ij}} = softmax(W_a[T_i, E_j] + b_a)$$



$$\overline{C}_i = f(W_c C_i + b_c)$$

Joint MEE (JMEE) [Liu et al., EMNLP'18]



$$-\sum_{p=1}^{N}\left(\sum_{i=1}^{n_p} I(y_{t_i}) log(p(y_{t_i}|\theta))\right.$$
$$\left.+\beta\sum_{i=1}^{t_p}\sum_{j=1}^{e_p} log(p(y_{a_{ij}}|\theta))\right)$$

Joint MEE (JMEE) [Liu et al., EMNLP'18]

F1 the ACE 2005 dataset

Method	Trigger Classification	Argument Role Labelling
dBRNN [Sha et al., AAAI'18]	71.9	58.7
JMEE	**73.7**	**60.3**

GCN, LSTM complement each other

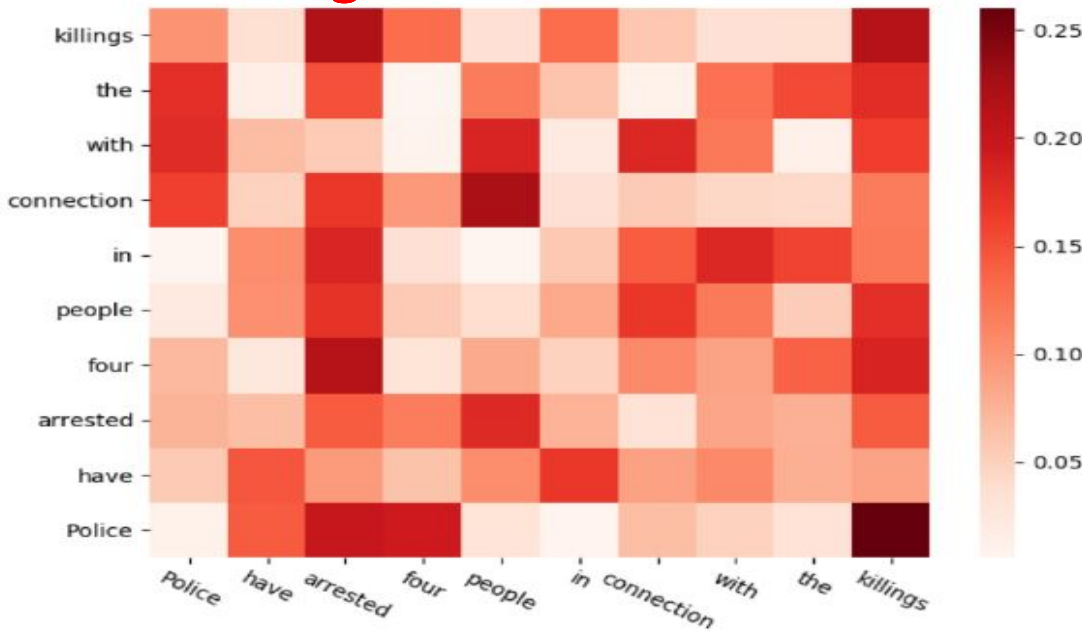
Joint MEE (JMEE) [Liu et al., EMNLP'18]

F1 the ACE 2005 dataset

Method	Trigger Classification	Argument Role Labelling
dBRNN [Sha et al., AAAI'18]	71.9	58.7
JMEE	**73.7**	**60.3**

GCN, LSTM complement each other

high



police have **arrested** four people in connection with the **killing**

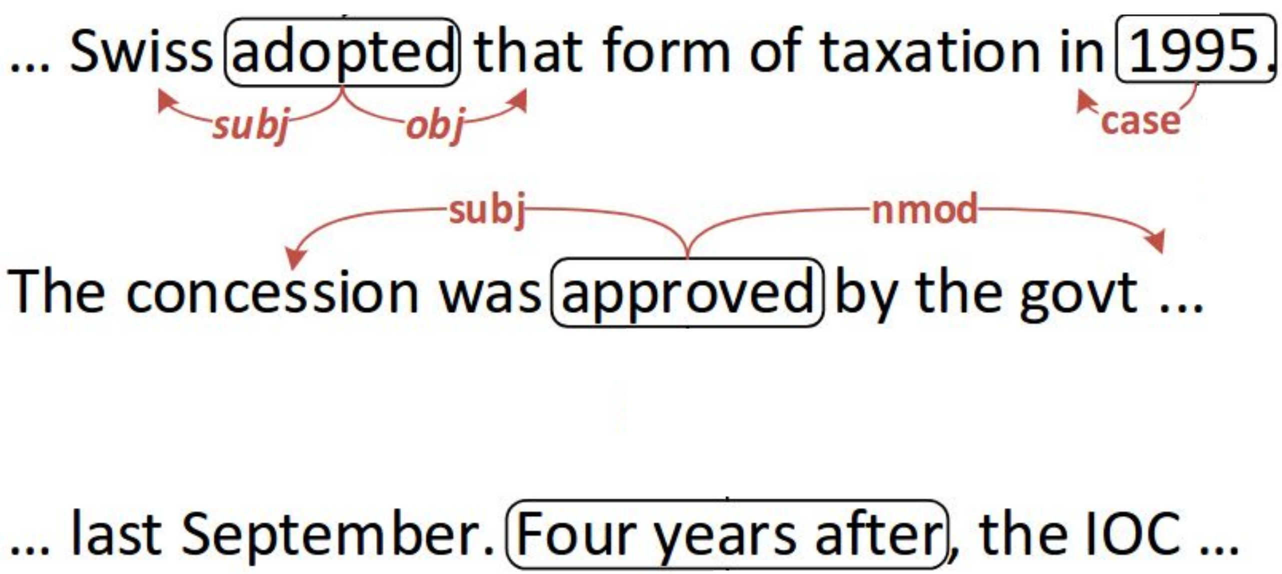
Document Timestamping [Vashishth et al., ACL’18]

... Swiss adopted that form of taxation in 1995.

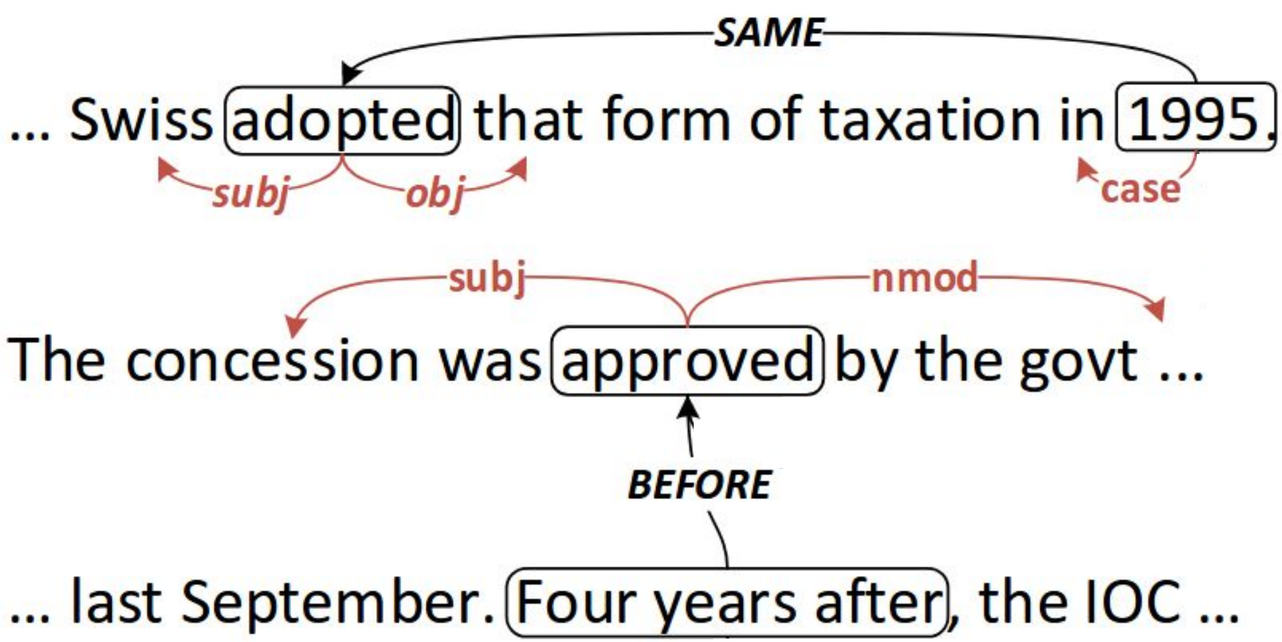
The concession was approved by the govt ...

... last September. Four years after, the IOC ...

Document Timestamping [Vashishth et al., ACL'18]

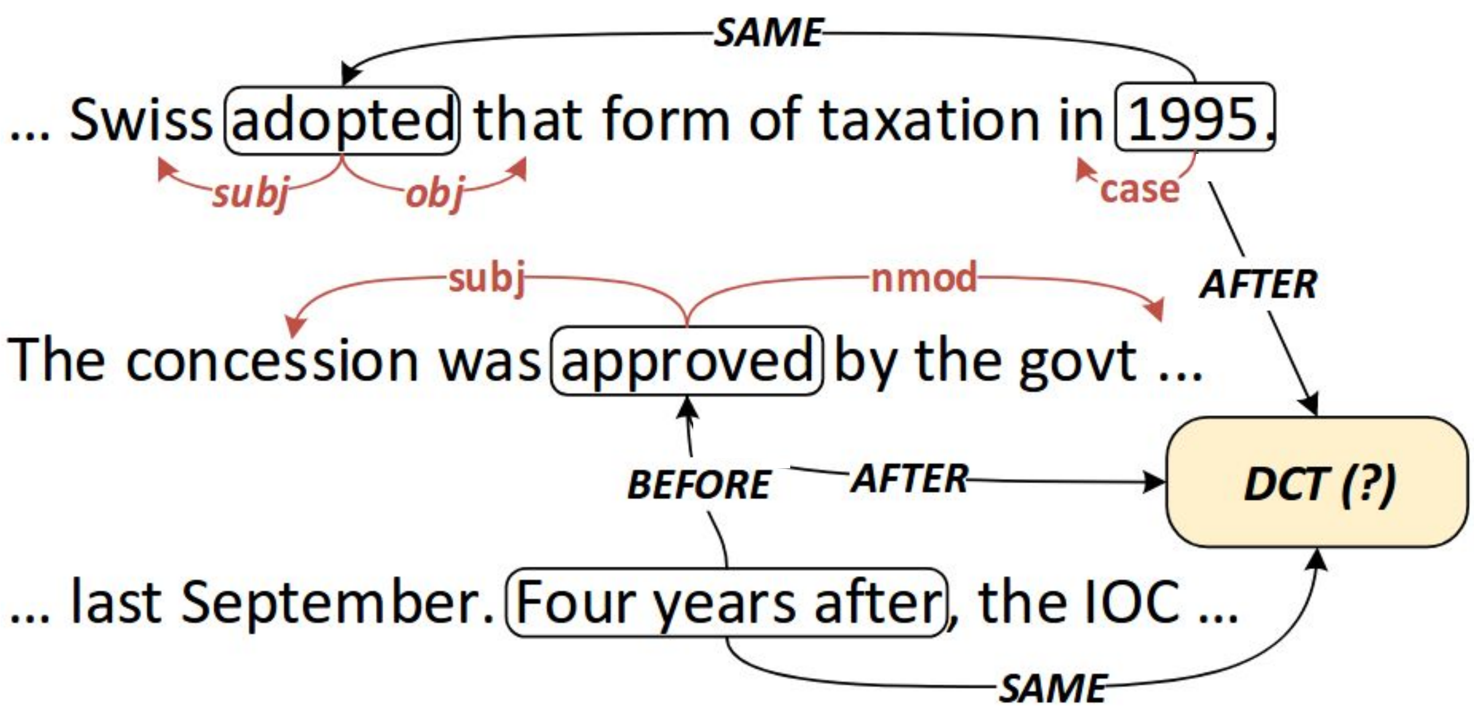


Document Timestamping [Vashishth et al., ACL'18]



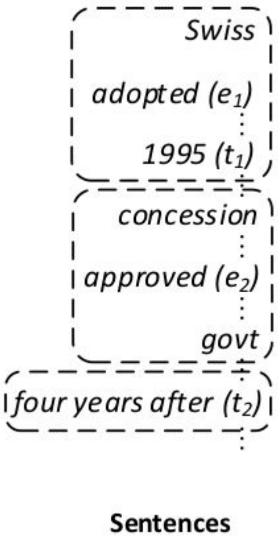
- Use CATENA for temporal graph [Mirza et al., COLING'16]

Document Timestamping [Vashishth et al., ACL'18]

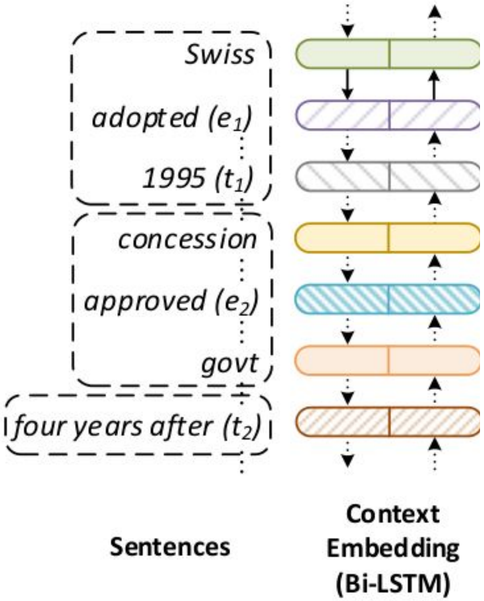


- Use CATENA for temporal graph [Mirza et al., COLING'16]
- Predict Document Creation Time

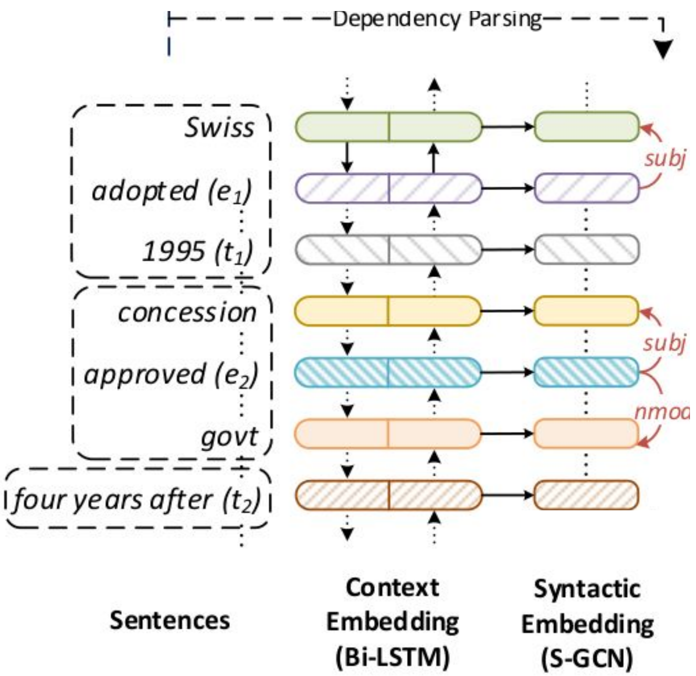
GCN for Timestamping [Vashishth et al., ACL'18]



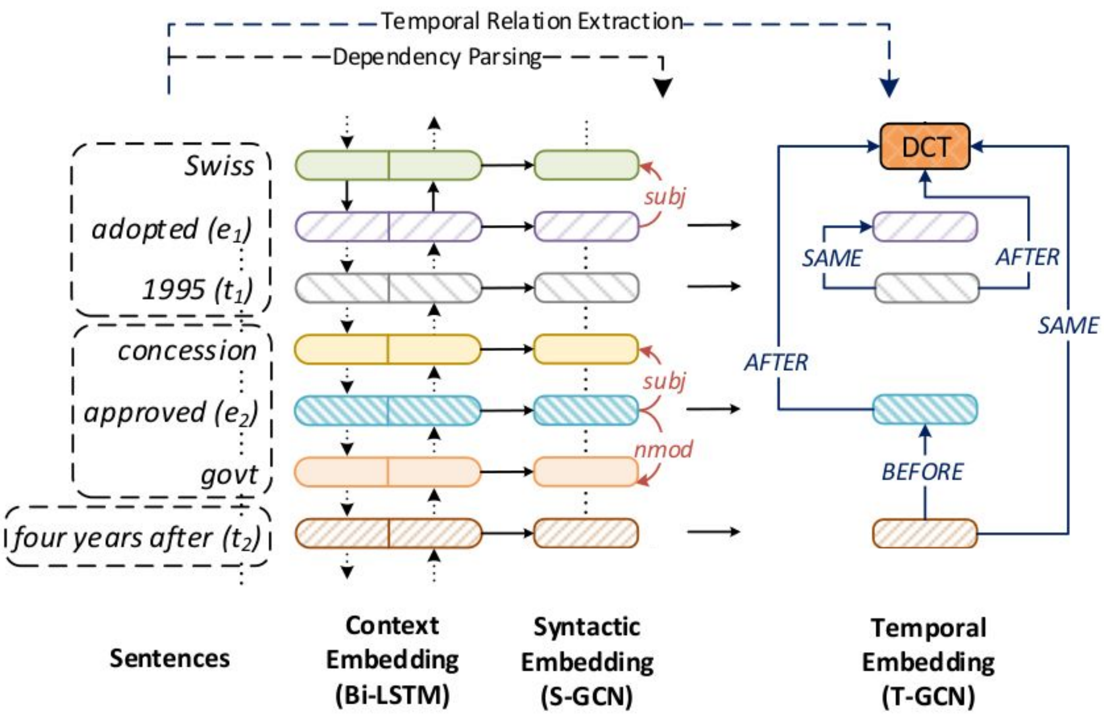
GCN for Timestamping [Vashishth et al., ACL'18]



GCN for Timestamping [Vashishth et al., ACL'18]

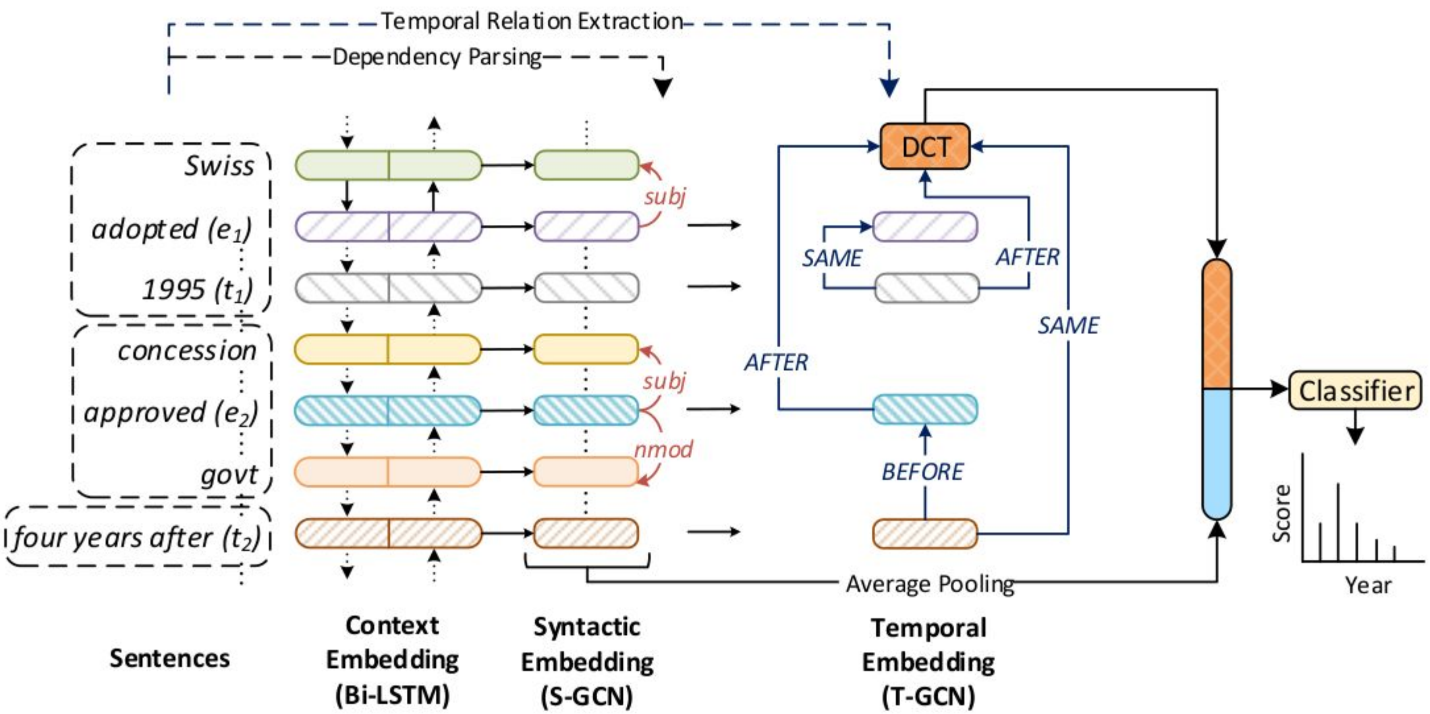


GCN for Timestamping [Vashishth et al., ACL'18]

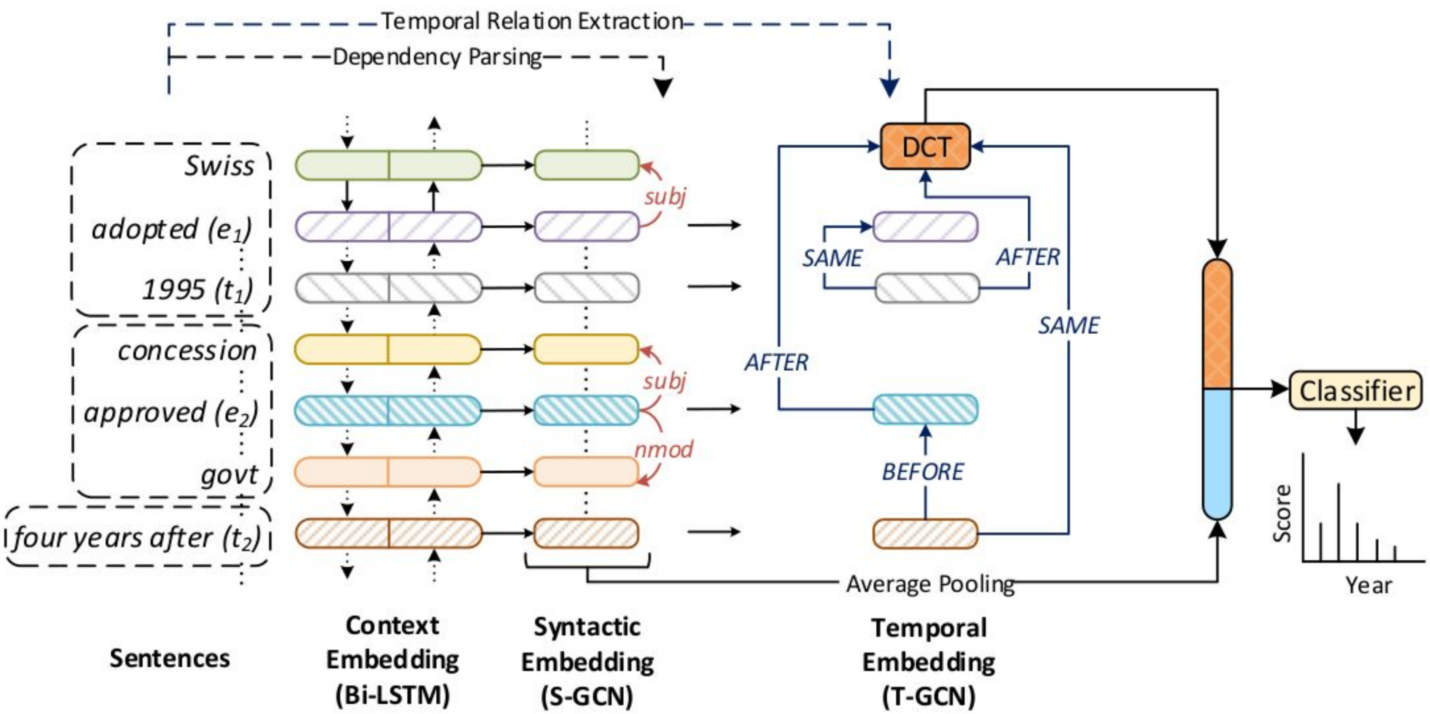


CATENA [Mirza et al., COLING'16]

GCN for Timestamping [Vashishth et al., ACL'18]



GCN for Timestamping [Vashishth et al., ACL'18]

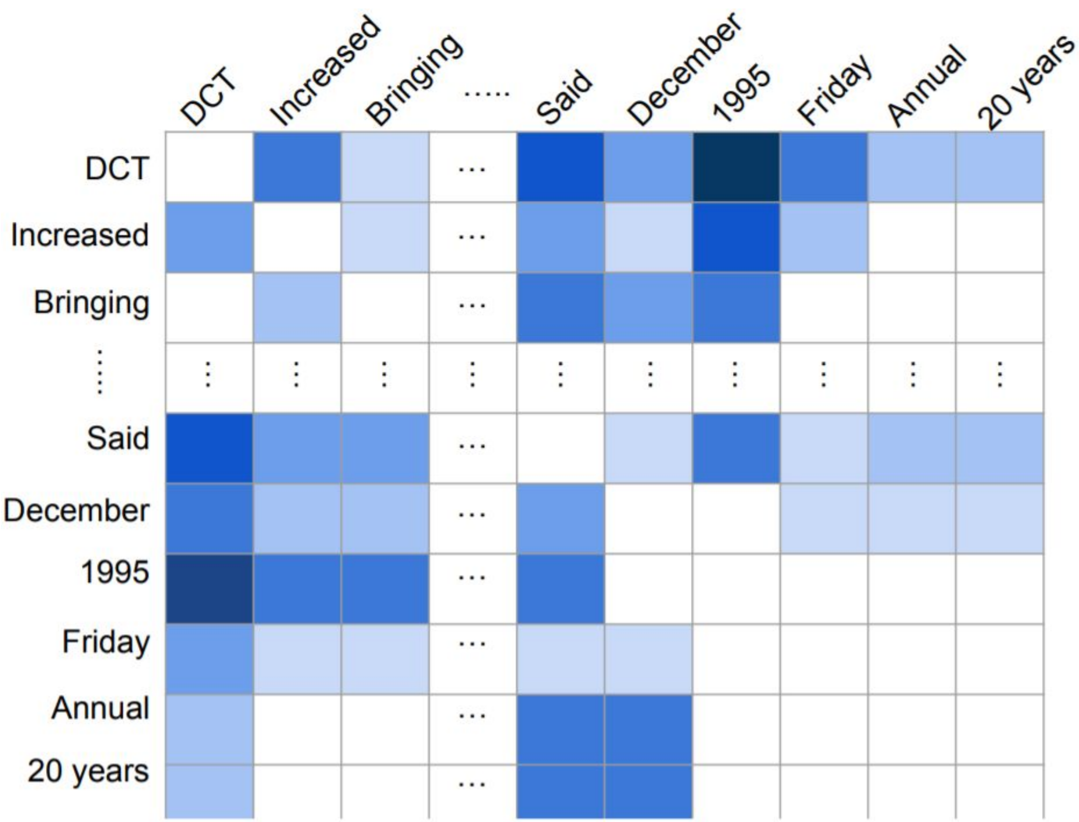


Method	Accuracy
Bi-LSTM	58.6
Bi-LSTM + Temporal GCN	60.5
Bi-LSTM + Syntactic GCN + Temporal GCN	64.1

Associated Press Worldstream

Time, Syntax improve contextual features

Graph Attention for Timestamping [Ray et al., EMNLP'18]



Method	Accuracy
T-GCN of NeuralDater	61.8
OE-GCN	**63.9**
S-GCN of NeuralDater	63.2
AC-GCN	**65.6**

Associated Press Worldstream

Relation Extraction

- Identify **relation** between entities.

- Google was **founded** in California in 1998.
  - **Founding-year** (Google, 1998)
  - **Founding-location** (Google, California)

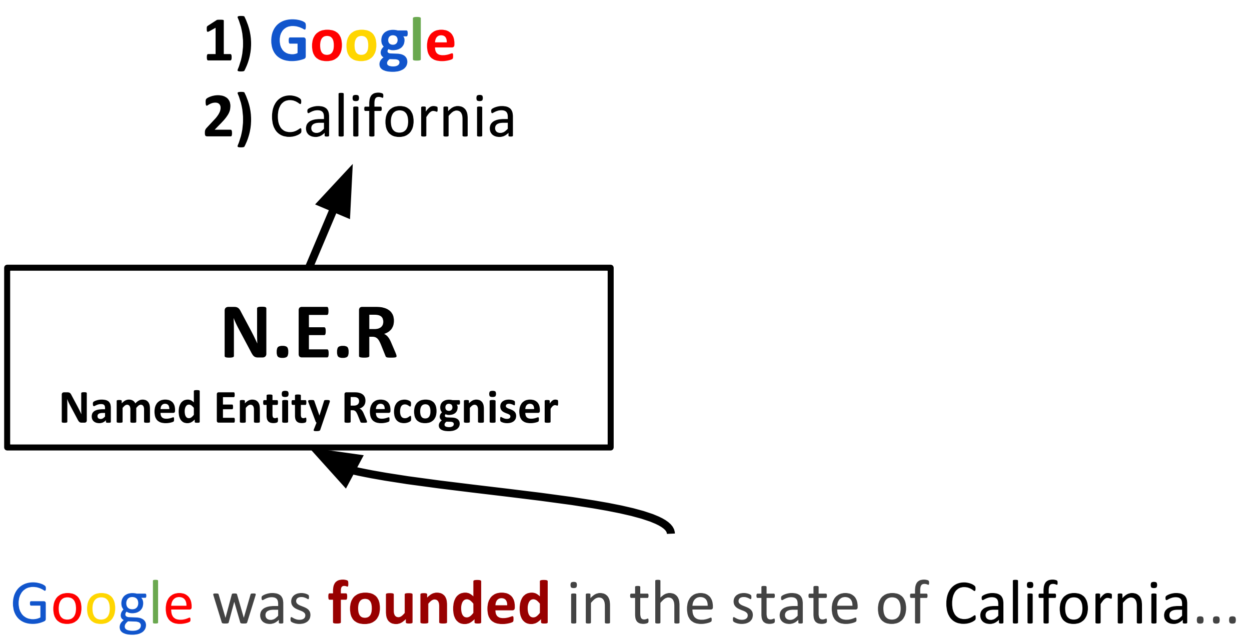
Relation Extraction

- Identify **relation** between entities.

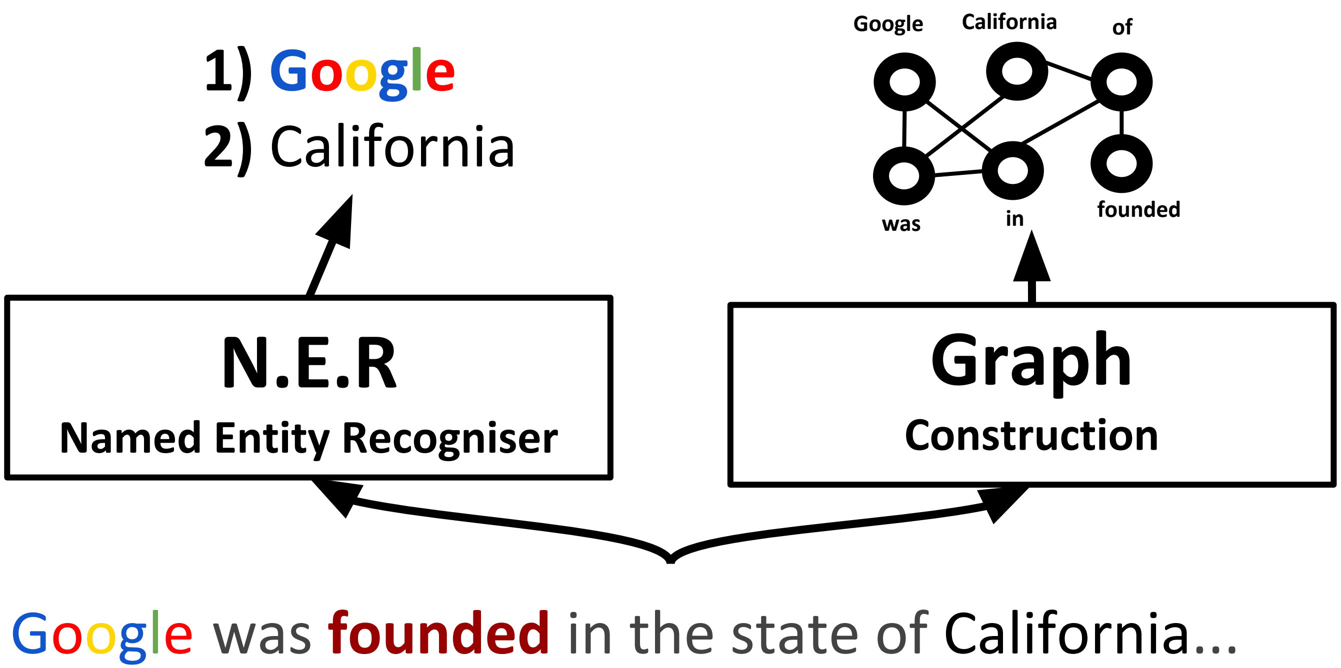
- Google was **founded** in California in 1998.
  - **Founding-year** (Google, 1998)
  - **Founding-location** (Google, California)

- Used for
  - Knowledge base population
  - Biomedical knowledge discovery
  - Question answering

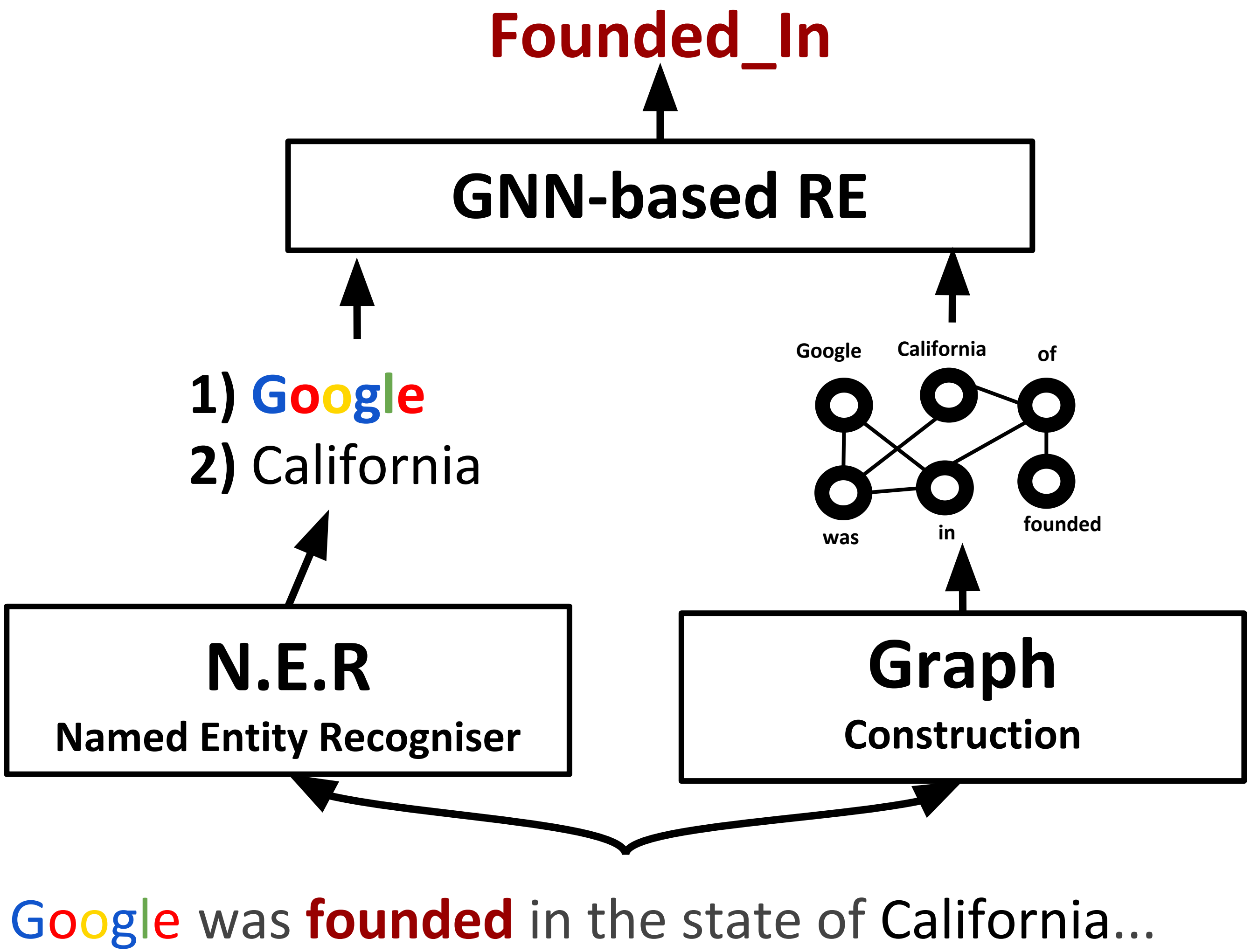
GNNs for Relation Extraction



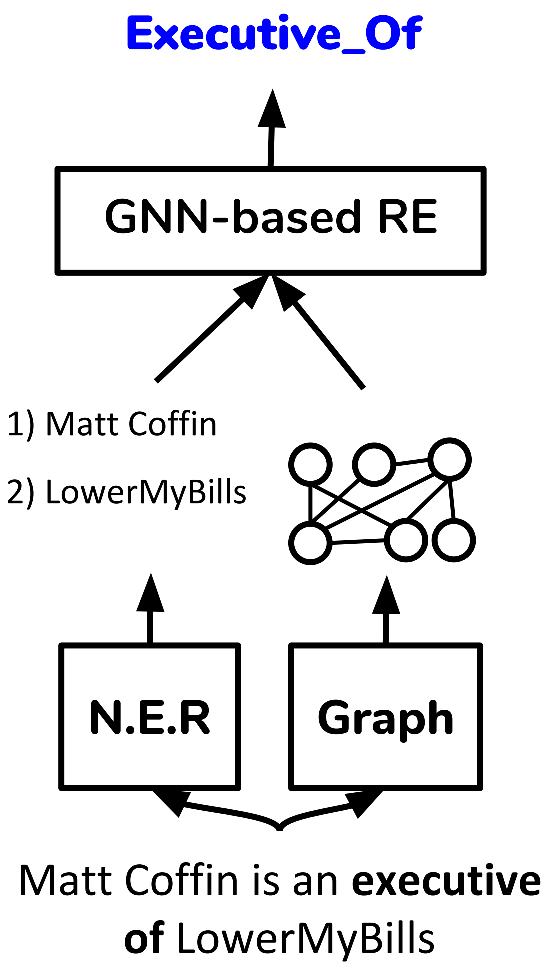
GNNs for Relation Extraction



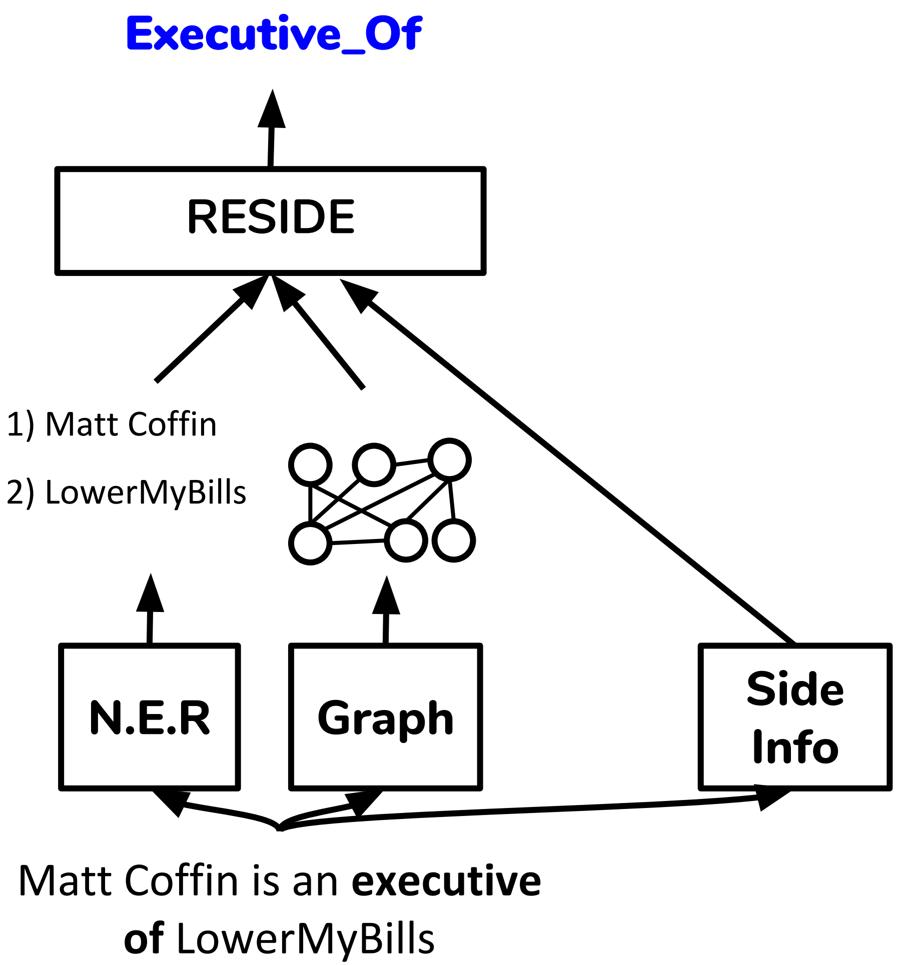
GNNs for Relation Extraction



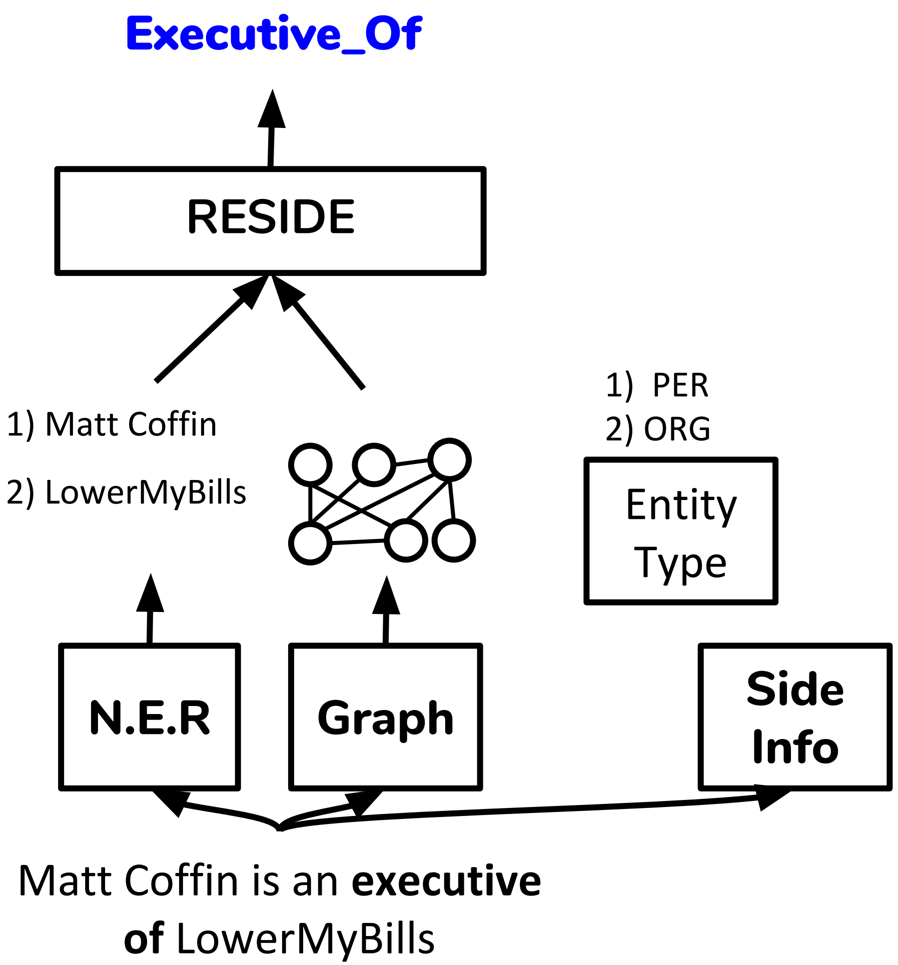
RE-SIDE [Vashishth et al., EMNLP'18]



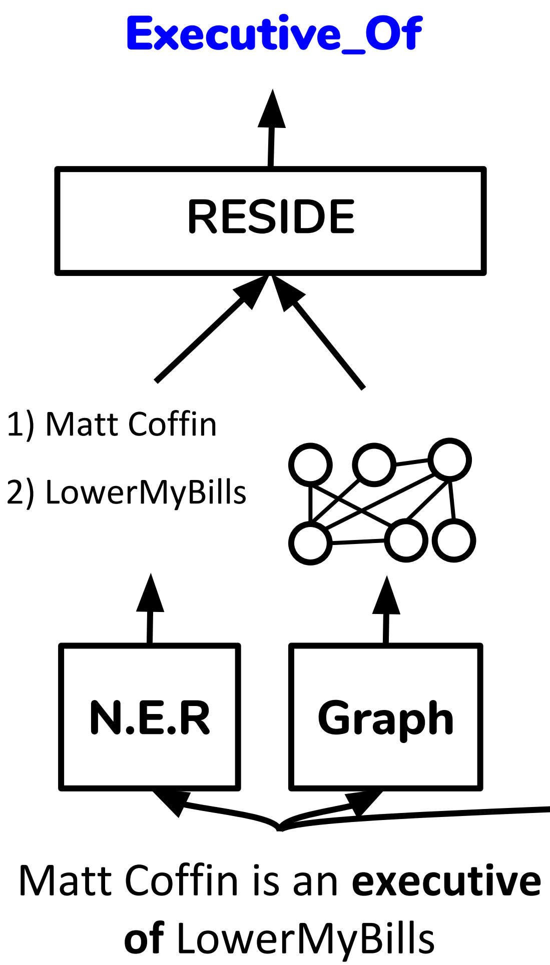
RE-SIDE [Vashishth et al., EMNLP'18]



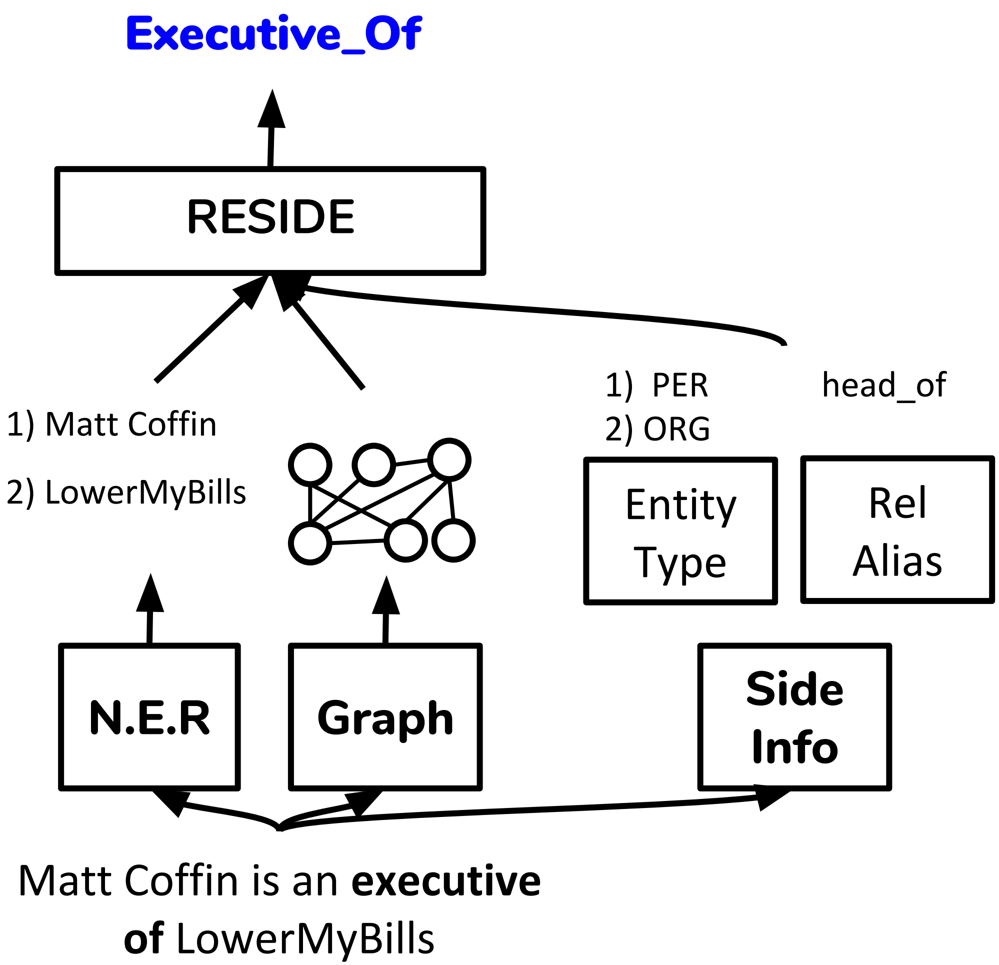
RE-SIDE [Vashishth et al., EMNLP'18]



RE-SIDE [Vashishth et al., EMNLP'18]



RE-SIDE [Vashishth et al., EMNLP'18]

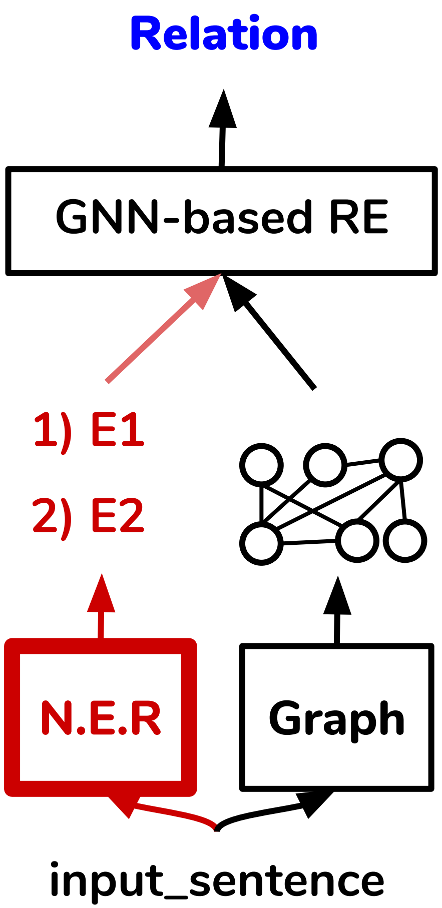


P@300 on Riedel dataset

PCNN + ATT	67
BGWA	72
RESIDE	**75**

Even limited side info improves performance

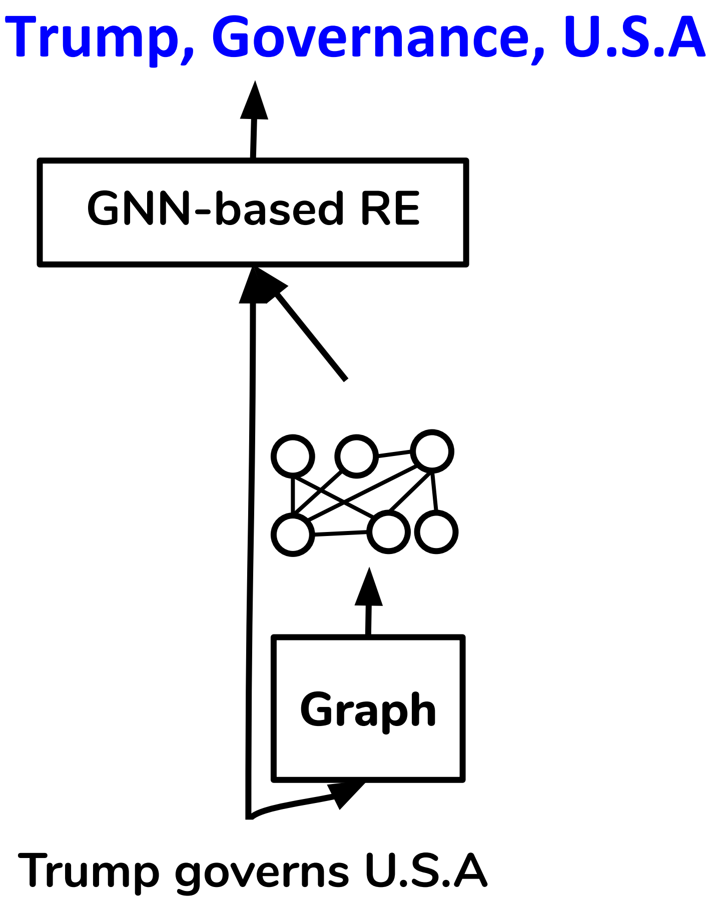
Joint Entity and RE [Fu et al., ACL'19]



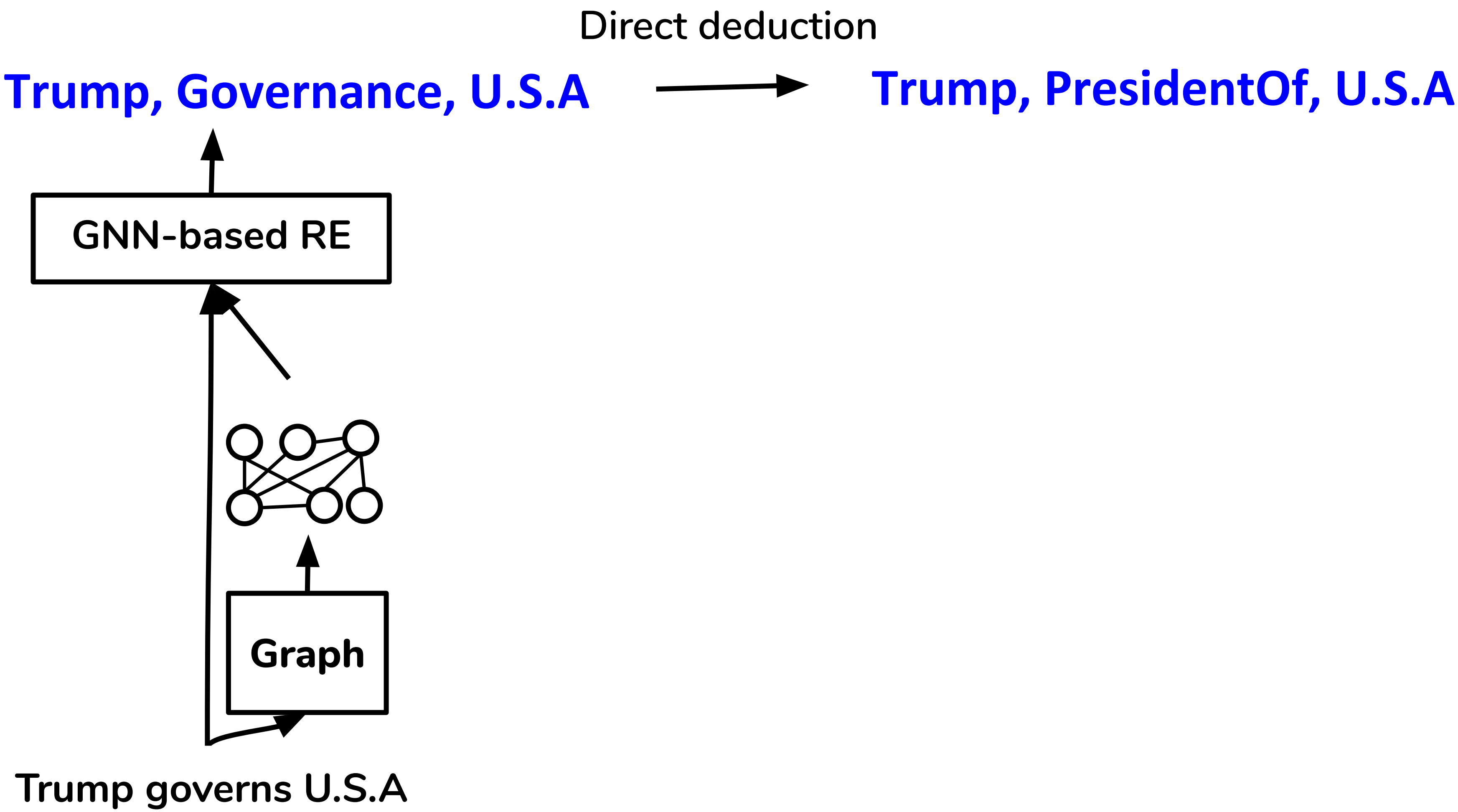
× **Assumes an N.E.R**

Errors are propagated without any feedback

Joint Entity and RE [Fu et al., ACL'19]



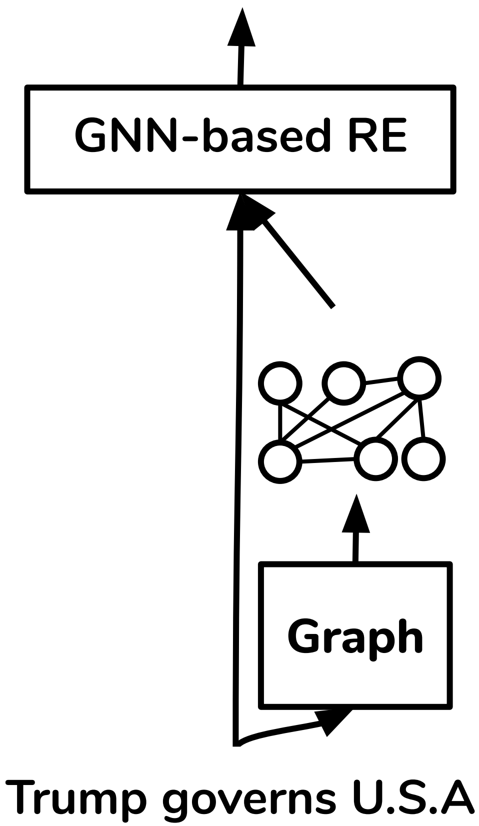
Joint Entity and RE [Fu et al., ACL'19]



Joint Entity and RE [Fu et al., ACL'19]

Direct deduction

Trump, Governance, U.S.A → **Trump, PresidentOf, U.S.A**



Indirect inference ↑

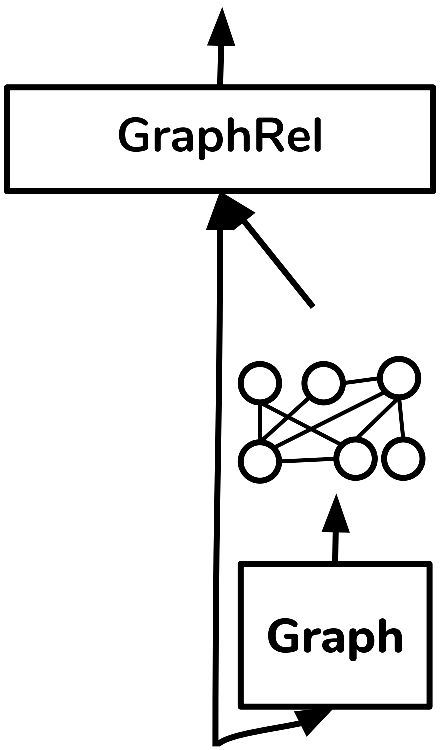
WhiteHouse, PresidentialPalace, USA
Trump, LiveIn, WhiteHouse

- Facilitates KBC

Joint Entity and RE [Fu et al., ACL'19]

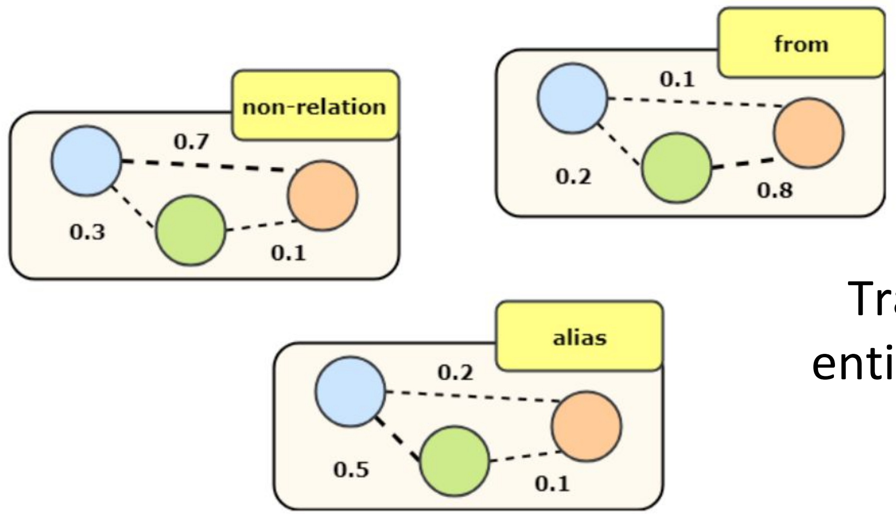
Direct deduction

Trump, Governance, U.S.A → **Trump, PresidentOf, U.S.A**



Trump governs U.S.A

Idea: GCN on relational graphs



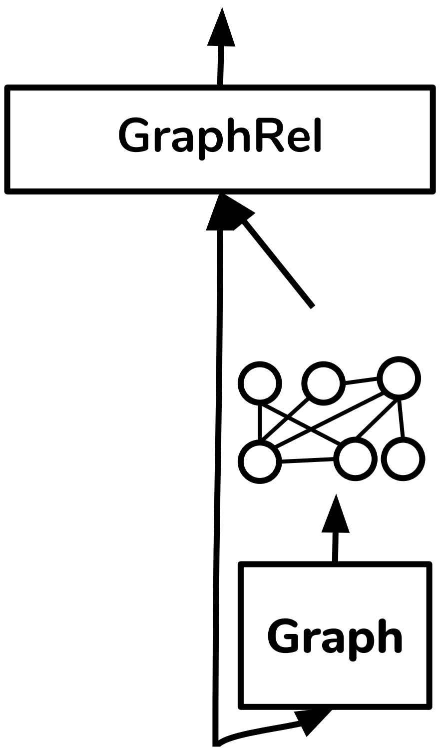
Trained with entity + relation losses

MultiDecoder	59
GraphRel	**62**

F1 on NYT dataset

Joint Entity and RE [Fu et al., ACL'19]

Trump, Governance, U.S.A



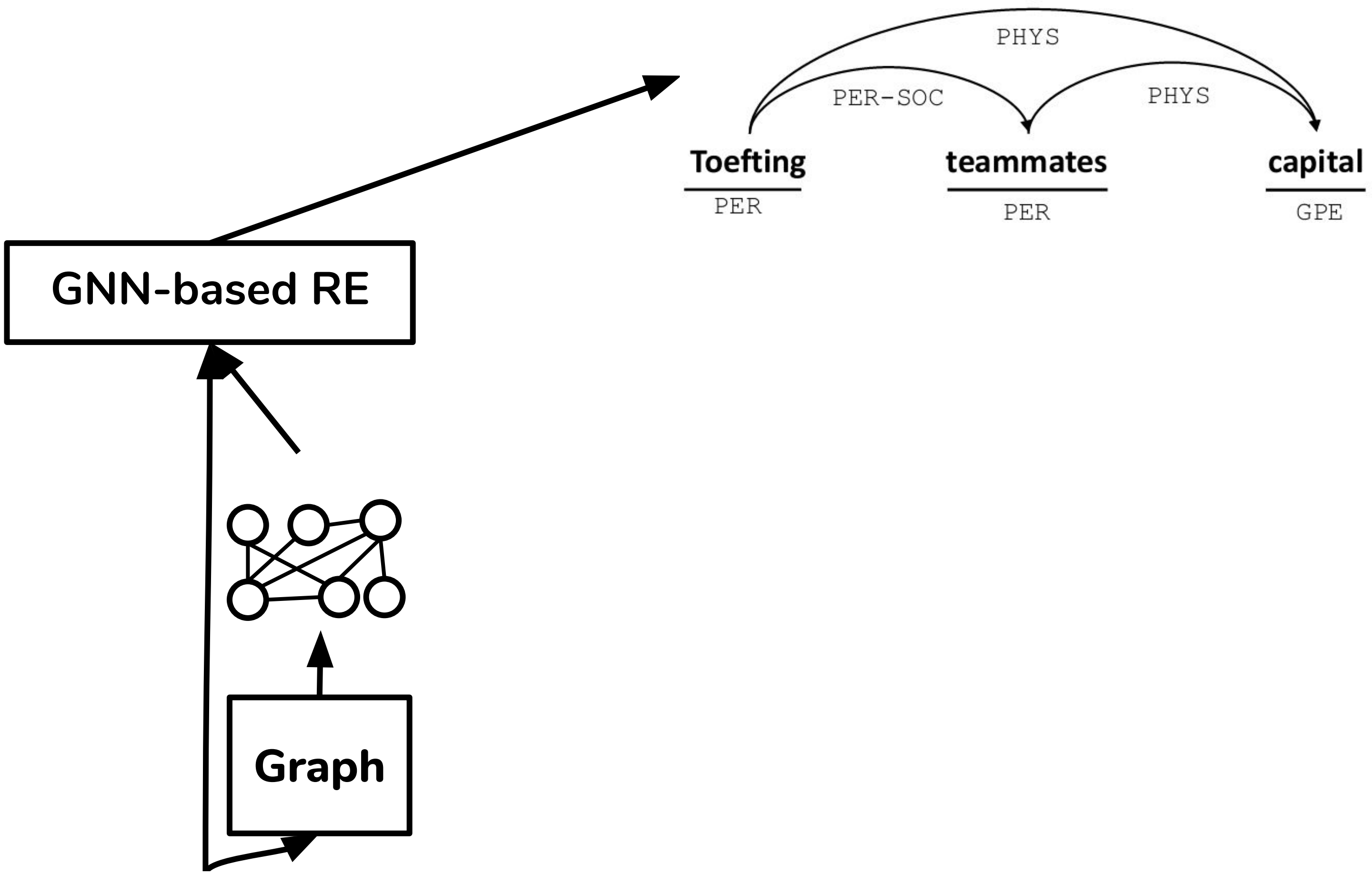
Trump governs U.S.A

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 padeh, region, Malay Peninsula)
(Asam pedas, country, Malaysia)
(Asam padeh, country, Malaysia)

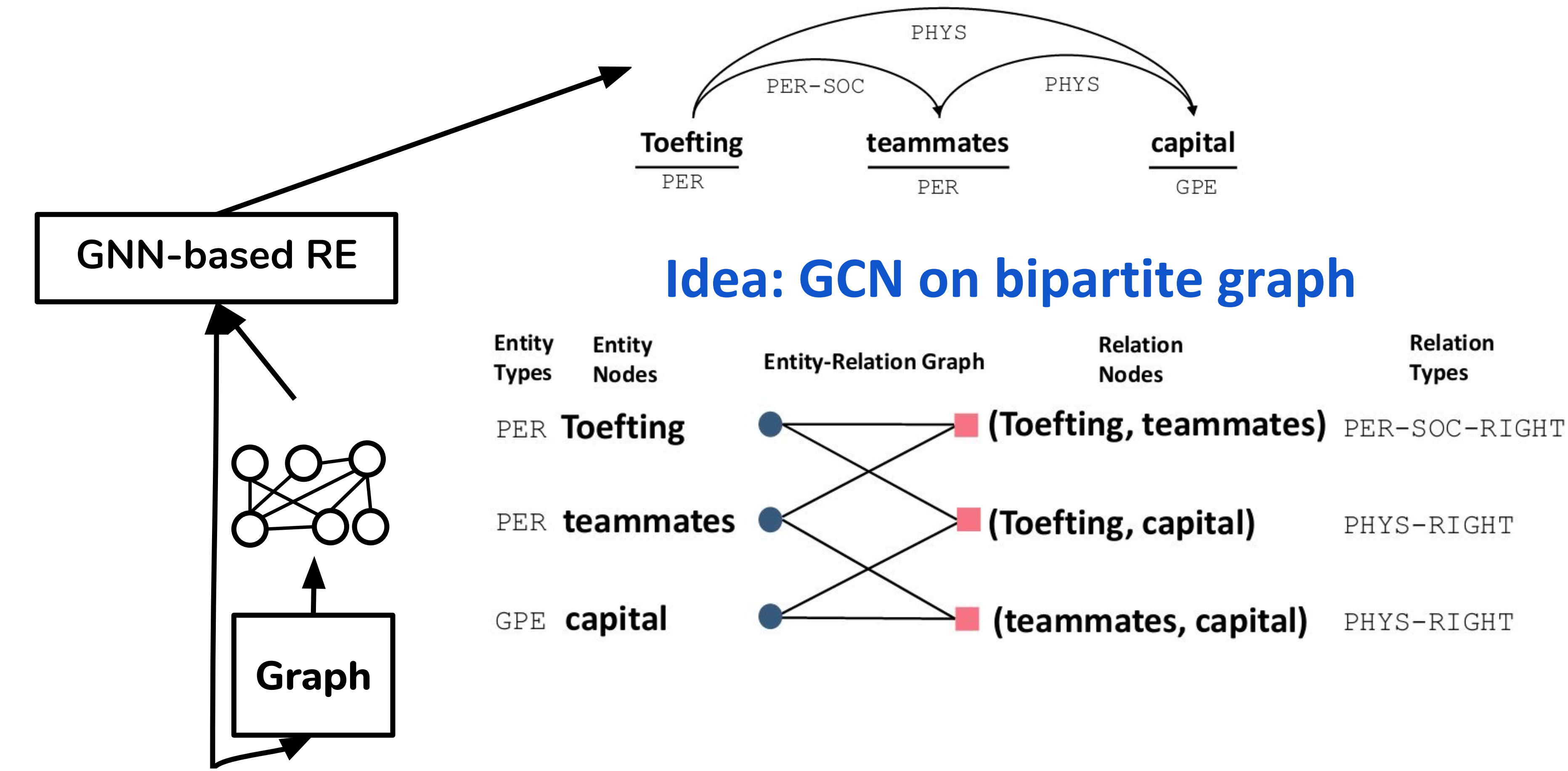
GNNs infer related triples

Joint Type Inference [Sun et al., ACL'19]



Toefting was convicted with **teammates** in the **capital** ...

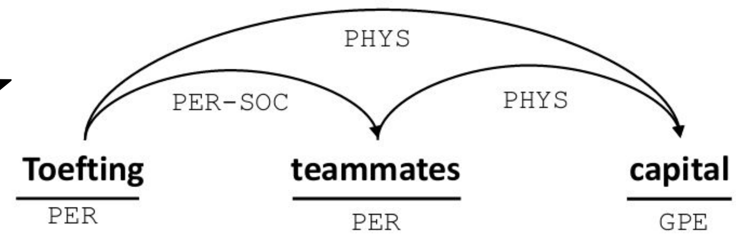
Joint Type Inference [Sun et al., ACL'19]

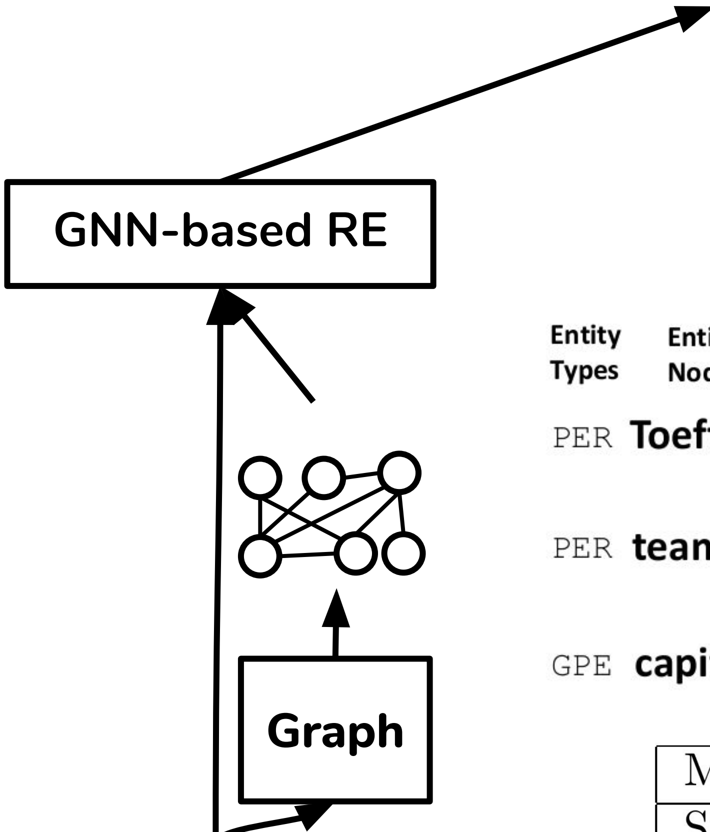


Idea: GCN on bipartite graph

Toefting was convicted with **teammates** in the **capital** ...

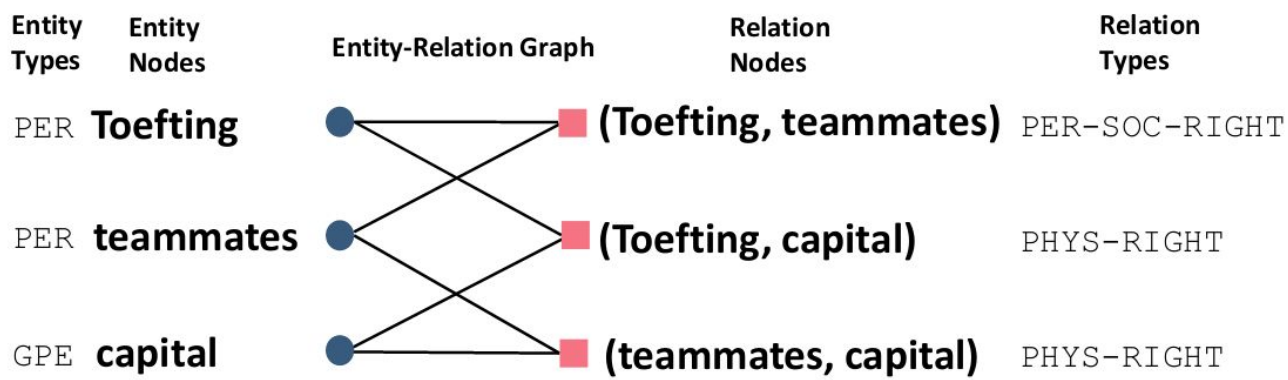
Joint Type Inference [Sun et al., ACL'19]





Toefting was convicted with **teammates** in the **capital** ...

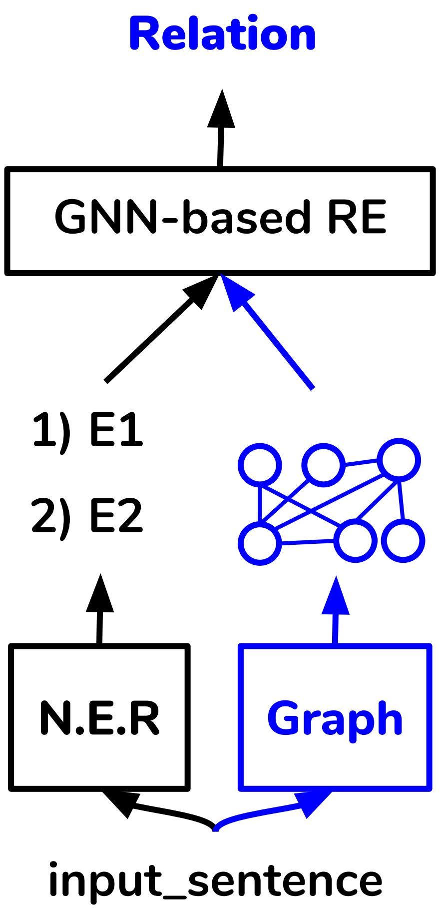
Idea: GCN on bipartite graph



Method	Entity	Relation
Sun et al., EMNLP'18	83.4	57.8
GCN	**84.2**	**59.1**

F1 on ACE05 dataset

Inter-Sentence RE [Sahu et al., ACL'19]

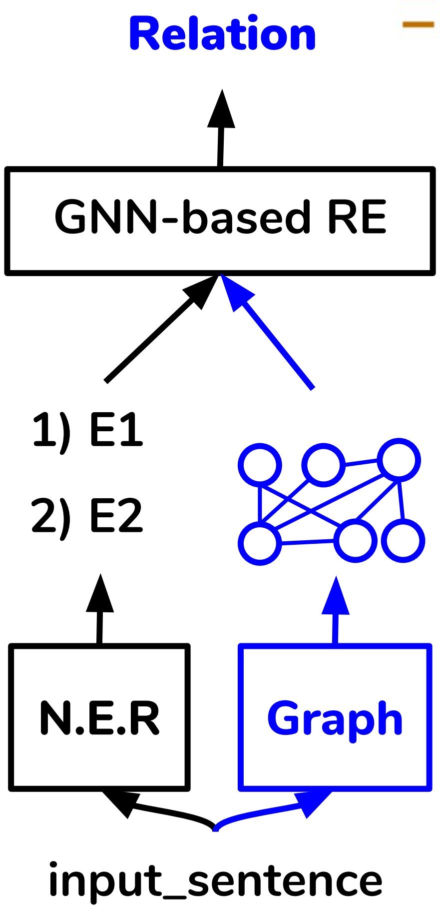


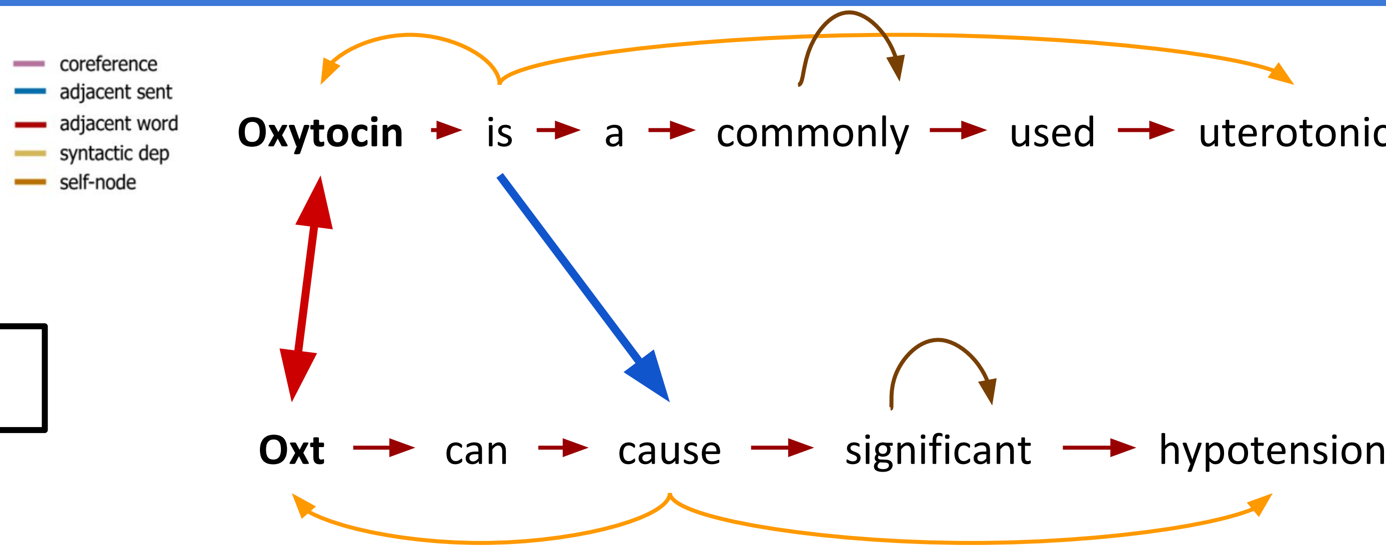
Oxytocin is a commonly used uterotonic

Oxt can cause significant hypotension

Oxytocin , ? , hypotension

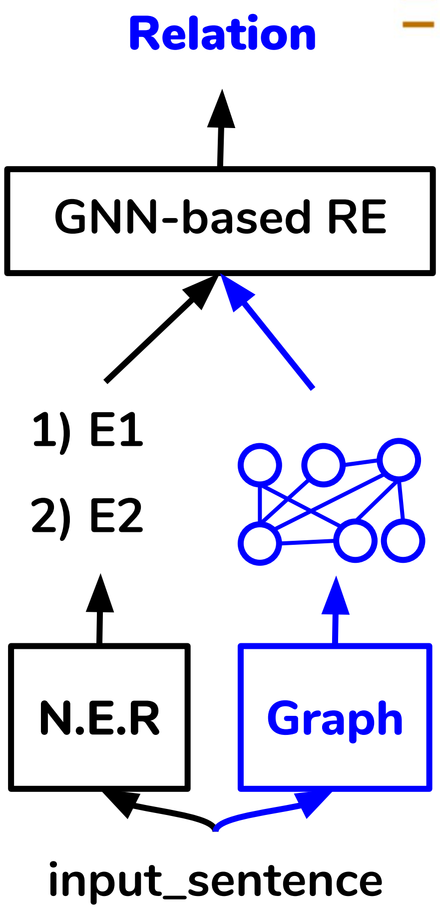
Inter-Sentence RE [Sahu et al., ACL'19]

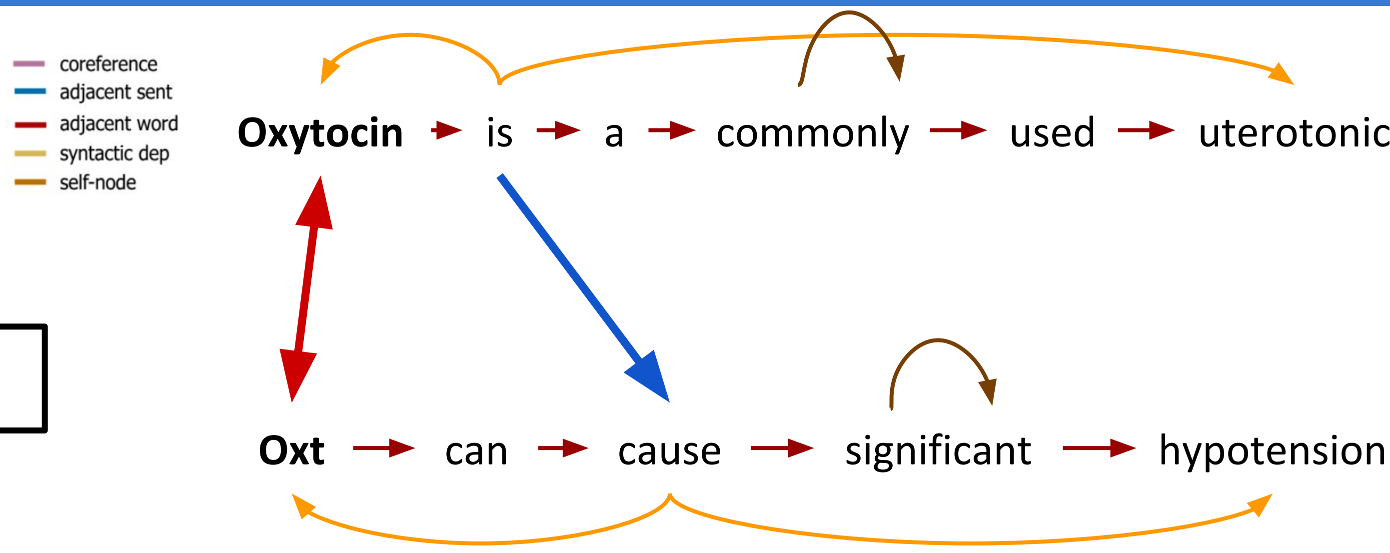




Oxytocin , ? , hypotension

Inter-Sentence RE [Sahu et al., ACL'19]



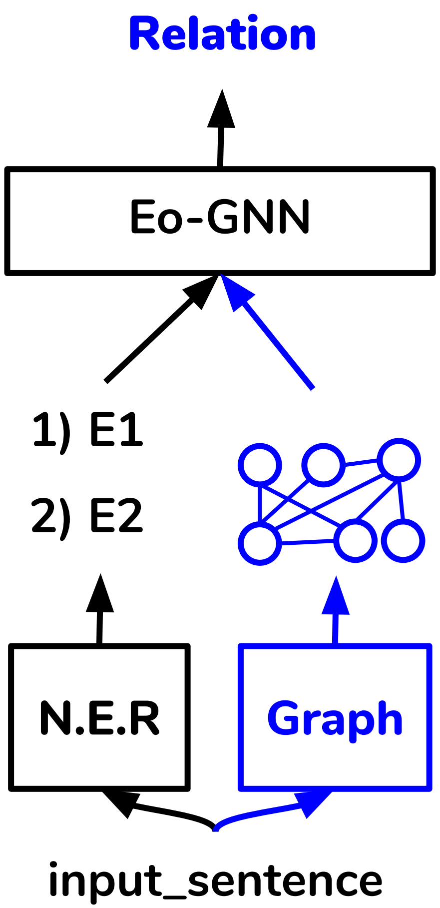


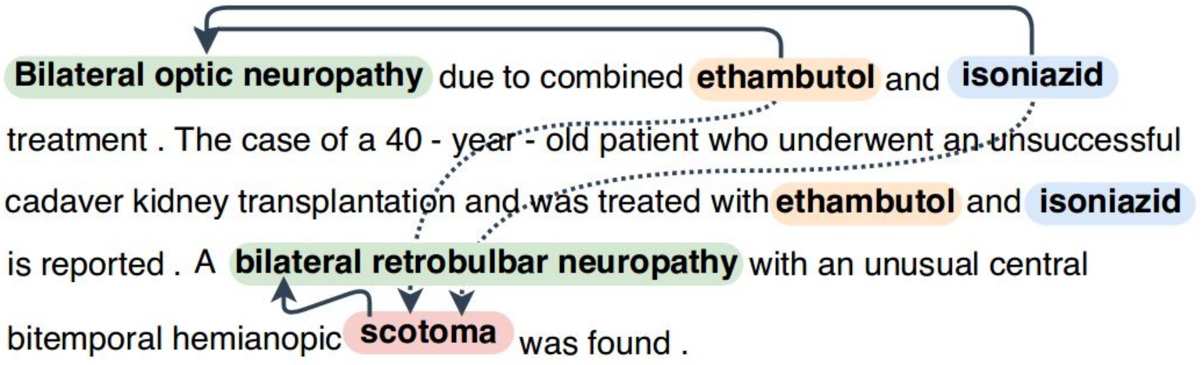
Oxytocin , ? , hypotension

Bi-LSTM	84.1
Bi-LSTM + GCN	**87.5**

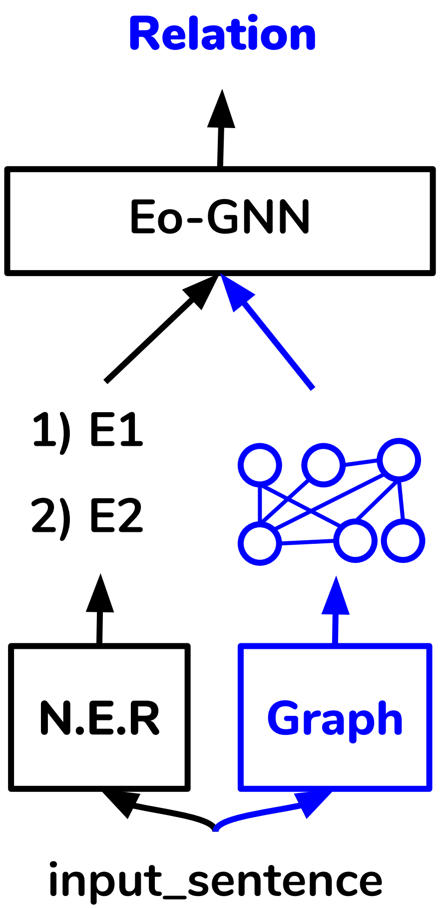
Coreference edges improve inter RE

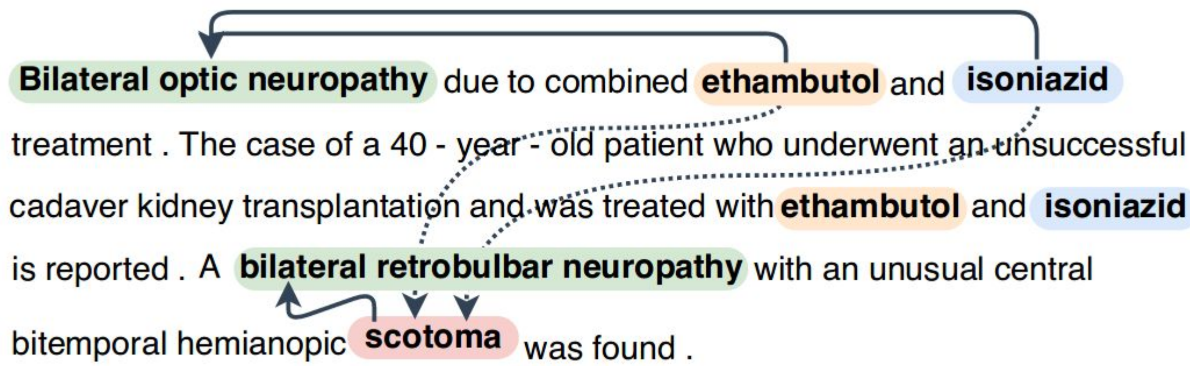
Edge-oriented Graph [Christopoulou et al., ACL'18, EMNLP'19]





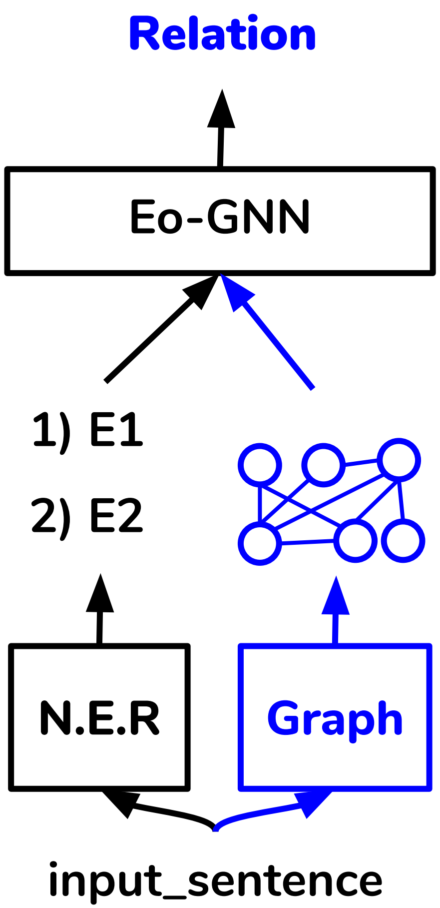
Edge-oriented Graph [Christopoulou et al., ACL’18, EMNLP’19]

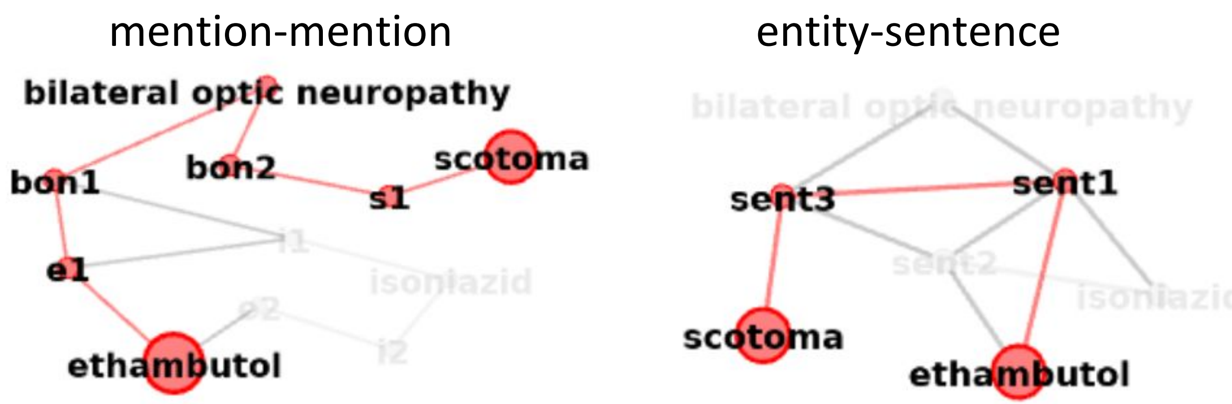




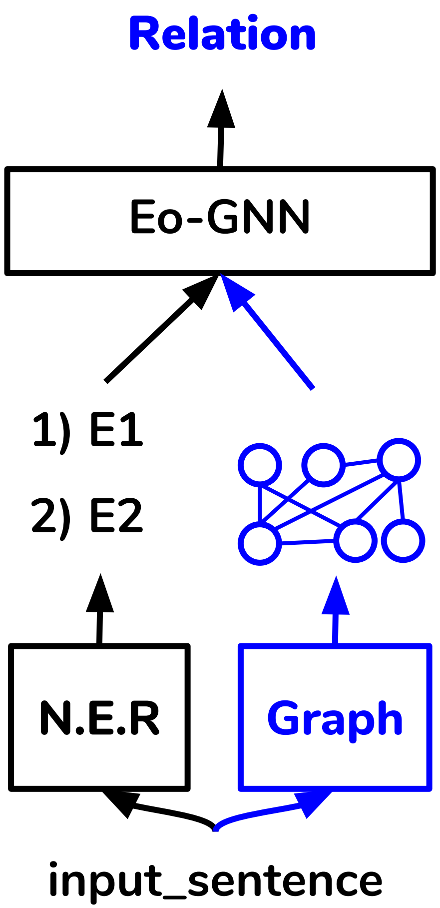
- ethambutol, scotoma have an inter-sentence relation
- can only be inferred from a chain of intra-sentence relations
- unique representation for a pair has better expressiveness

Edge-oriented Graph [Christopoulou et al., ACL'18, EMNLP'19]

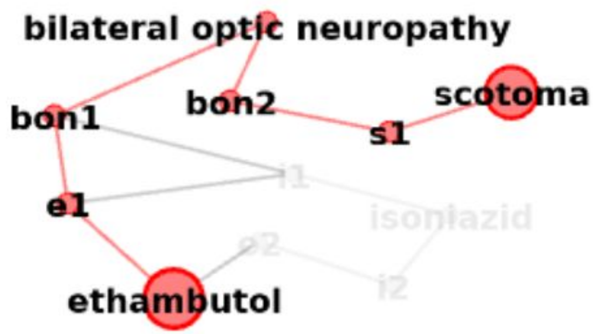




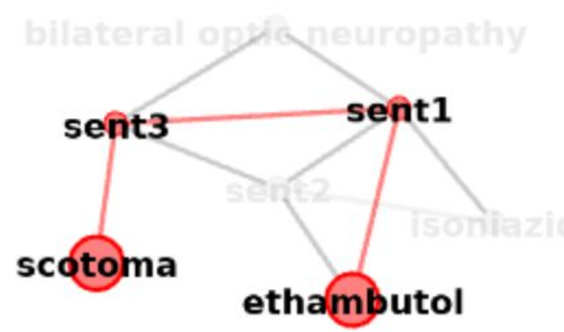
Edge-oriented Graph [Christopoulou et al., ACL'18, EMNLP'19]



mention-mention



entity-sentence

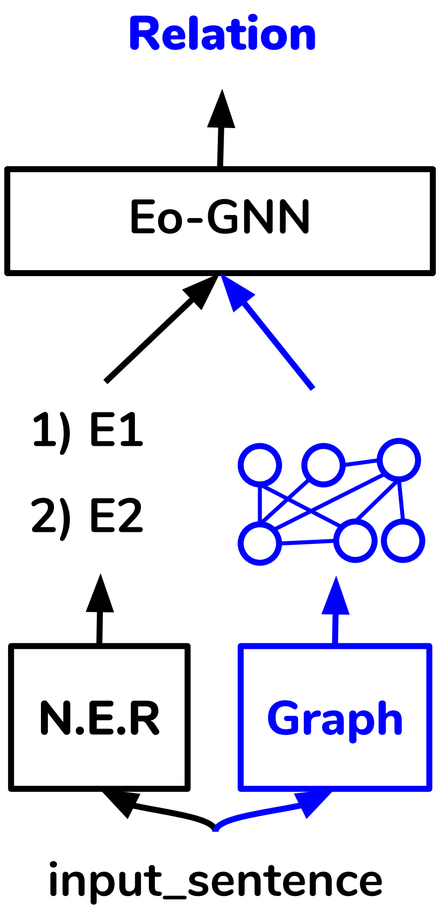


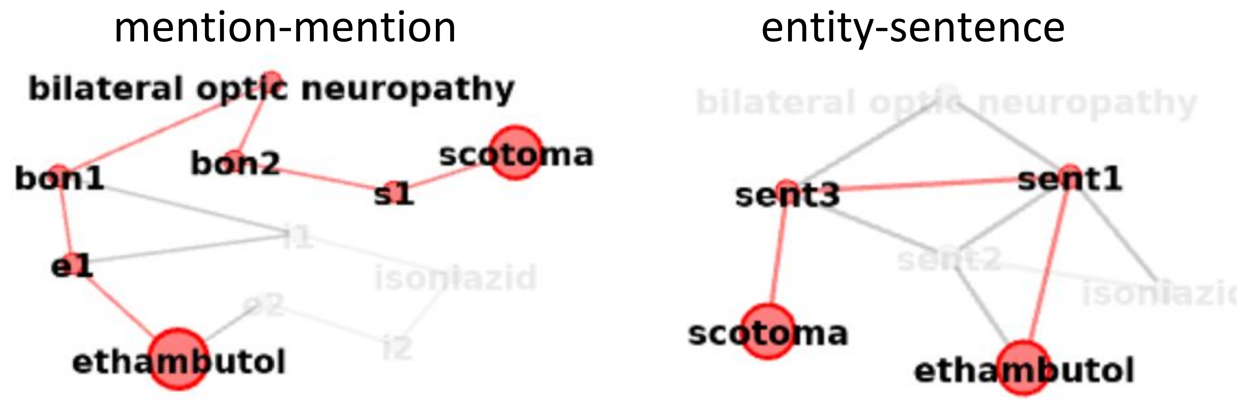
$$\mathbf{n}_m = [\mathrm{avg}_{w_i \in m}(\mathbf{w}_i); \mathbf{t}_m]$$

$$\mathbf{n}_e = [\mathrm{avg}_{m_i \in e}(\mathbf{m}_i); \mathbf{t}_e]$$

$$\mathbf{n}_s = [\mathrm{avg}_{w_i \in s}(\mathbf{w}_i); \mathbf{t}_s]$$

Edge-oriented Graph [Christopoulou et al., ACL'18, EMNLP'19]

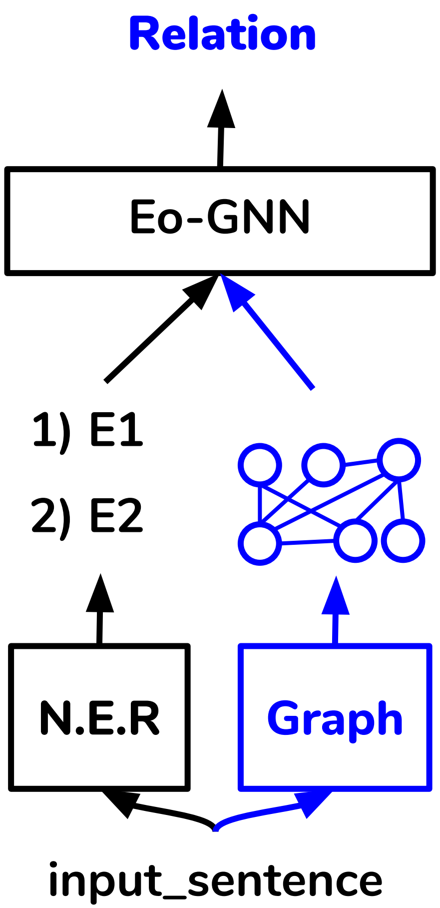


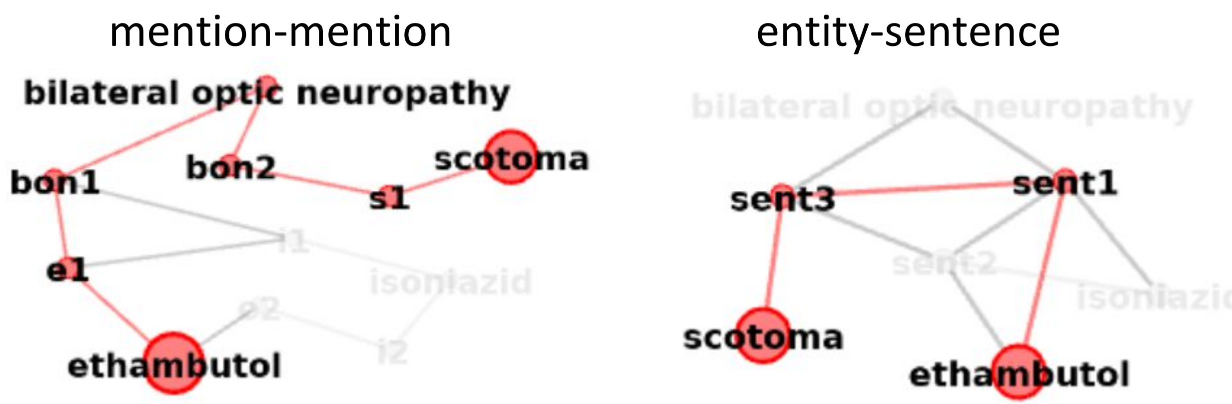


$$x_{MM} = [\mathbf{n}_{m_i}; \mathbf{n}_{m_j}; \mathbf{c}_{m_i,m_j}; \mathbf{d}_{m_i,m_j}]$$

$$x_{MS} = [\mathbf{n}_m; \mathbf{n}_s]$$

Edge-oriented Graph [Christopoulou et al., ACL'18, EMNLP'19]





$$\mathbf{x}_{\mathrm{MM}} = [\mathbf{n}_{m_i}; \mathbf{n}_{m_j}; \mathbf{c}_{m_i,m_j}; \mathbf{d}_{m_i,m_j}]$$

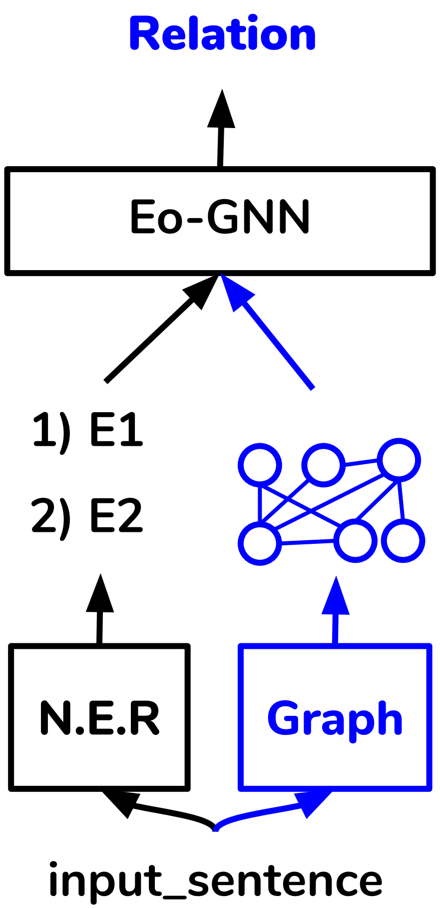
$$\mathbf{x}_{\mathrm{MS}} = [\mathbf{n}_m; \mathbf{n}_s]$$

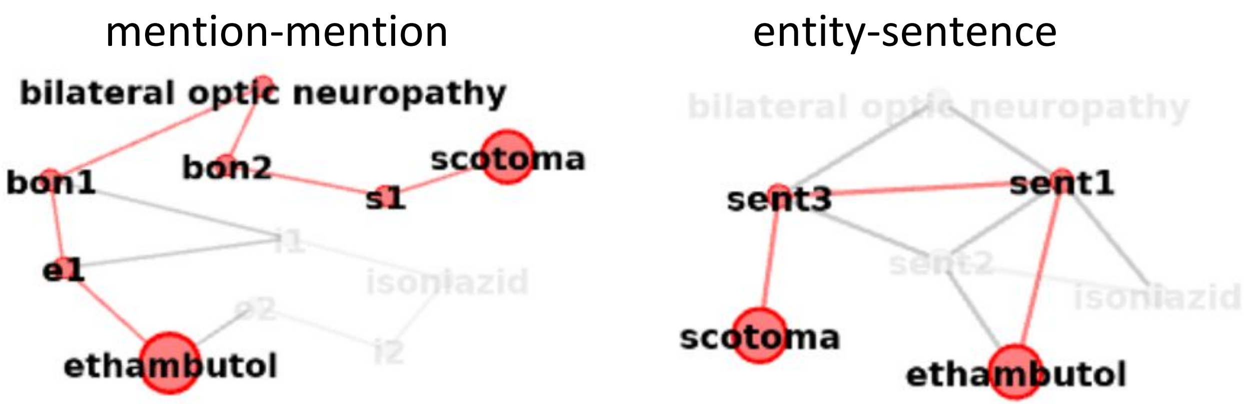
$$\mathbf{x}_{\mathrm{ME}} = [\mathbf{n}_m; \mathbf{n}_e]$$

$$\mathbf{x}_{\mathrm{SS}} = [\mathbf{n}_{s_i}; \mathbf{n}_{s_j}; \mathbf{d}_{s_i,s_j}]$$

$$\mathbf{x}_{\mathrm{ES}} = [\mathbf{n}_e; \mathbf{n}_s]$$

Edge-oriented GNN [Christopoulou et al., ACL'18, EMNLP'19]

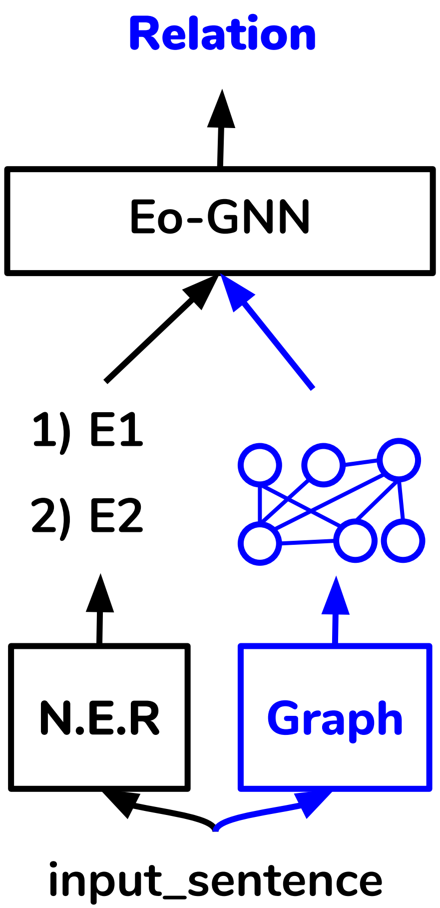


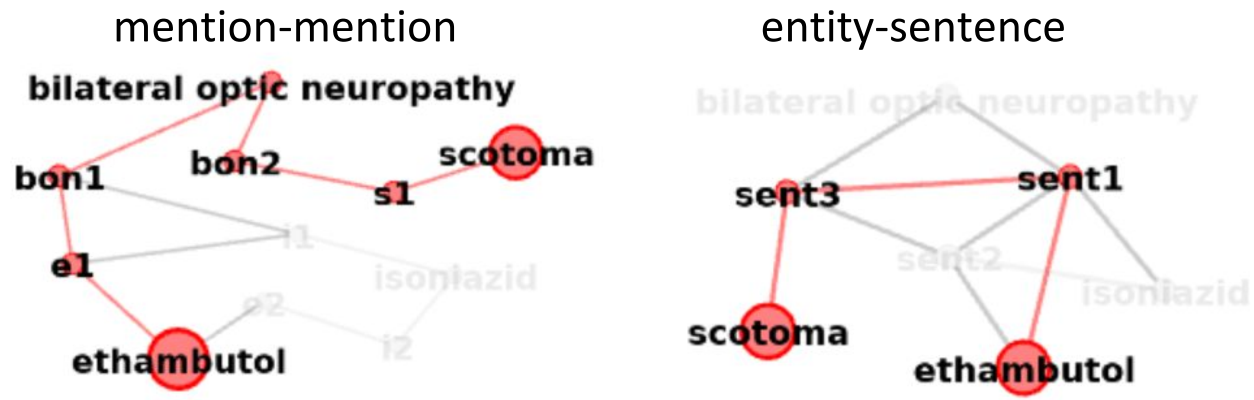


$$\mathbf{e}_z = \mathbf{W}_z\, \mathbf{x}_z \qquad z \in [\mathrm{MM}, \mathrm{MS}, \mathrm{ME}, \mathrm{SS}, \mathrm{ES}]$$

$$f\left(\mathbf{e}_{ik}^{(l)}, \mathbf{e}_{kj}^{(l)}\right) = \sigma\left(\mathbf{e}_{ik}^{(l)} \odot \left(\mathbf{W}\ \mathbf{e}_{kj}^{(l)}\right)\right)$$

Edge-oriented GNN [Christopoulou et al., ACL'18, EMNLP'19]





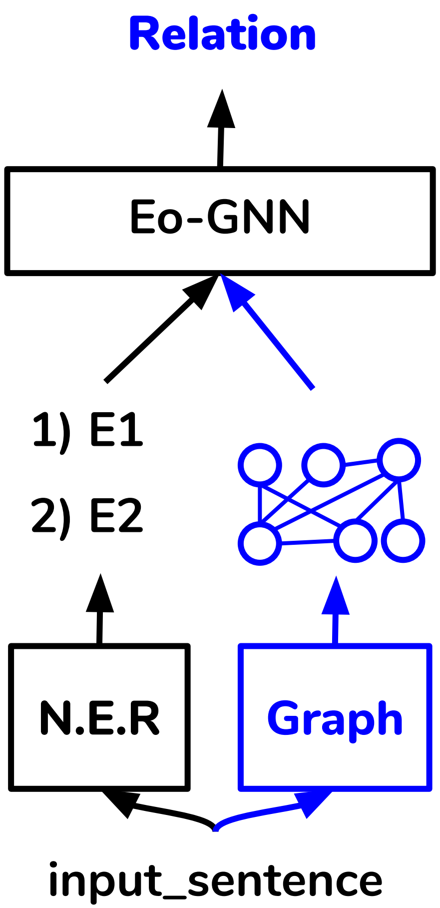
$$\mathbf{e}_z = \mathbf{W}_z\, \mathbf{x}_z \qquad z \in [\mathrm{MM}, \mathrm{MS}, \mathrm{ME}, \mathrm{SS}, \mathrm{ES}]$$

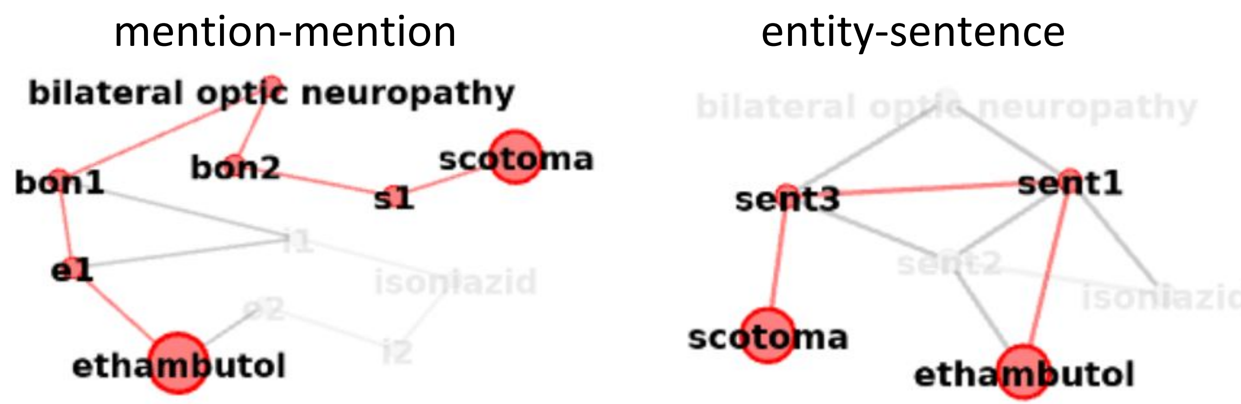
$$f\left(\mathbf{e}_{ik}^{(l)}, \mathbf{e}_{kj}^{(l)}\right) = \sigma\left(\mathbf{e}_{ik}^{(l)} \odot \left(\mathbf{W}\, \mathbf{e}_{kj}^{(l)}\right)\right)$$

$$\mathbf{e}_{ij}^{(2l)} = \beta\, \mathbf{e}_{ij}^{(l)} + (1-\beta) \sum_{k \neq i,j} f\left(\mathbf{e}_{ik}^{(l)}, \mathbf{e}_{kj}^{(l)}\right)$$

$$\mathbf{y} = \mathrm{softmax}\left(\mathbf{W}_c\, \mathbf{e}_{\mathrm{EE}} + \mathbf{b}_c\right)$$

Edge-oriented GNN [[Christopoulou et al., ACL'18, EMNLP'19]]





CDR dataset

Method	F1
CNN-Char	62.3
Eo-GNN	**63.6**

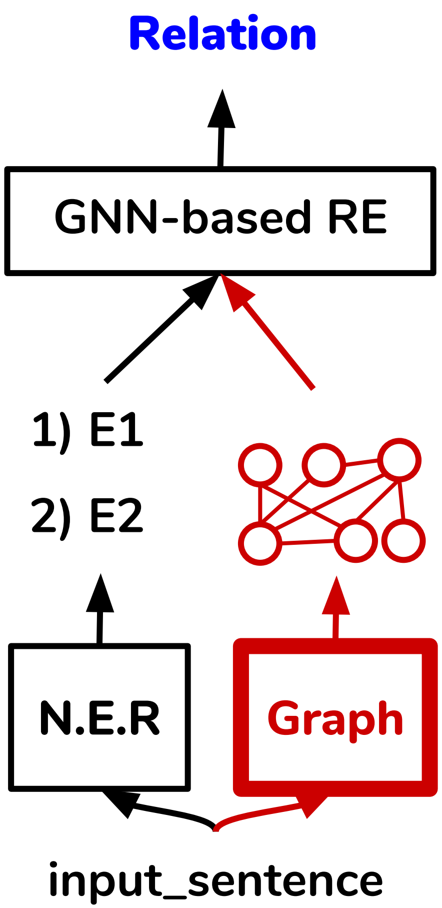
Method	F1 (Intra)	F1(Inter)
Graph Kernels	65.1	45.7
Eo-GNN	**68.2**	**50.9**

Method	F1
Eo-GNN(sent)	73.8
Eo-GNN (NoInf)	74.6
Eo-GNN (full)	80.8
Eo-GNN	**81.5**

Ablation on GDA dataset

Heterogeneity models relationships b/w intra-, inter- relations

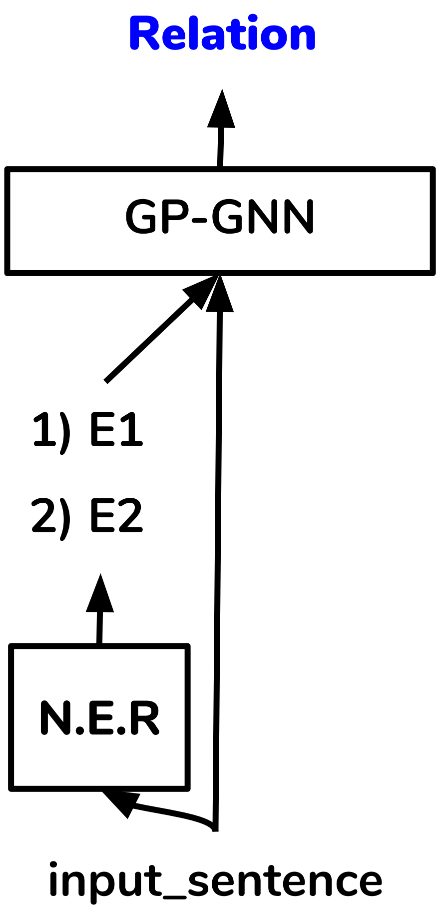
GNN on text [Zhu et al., ACL'19]



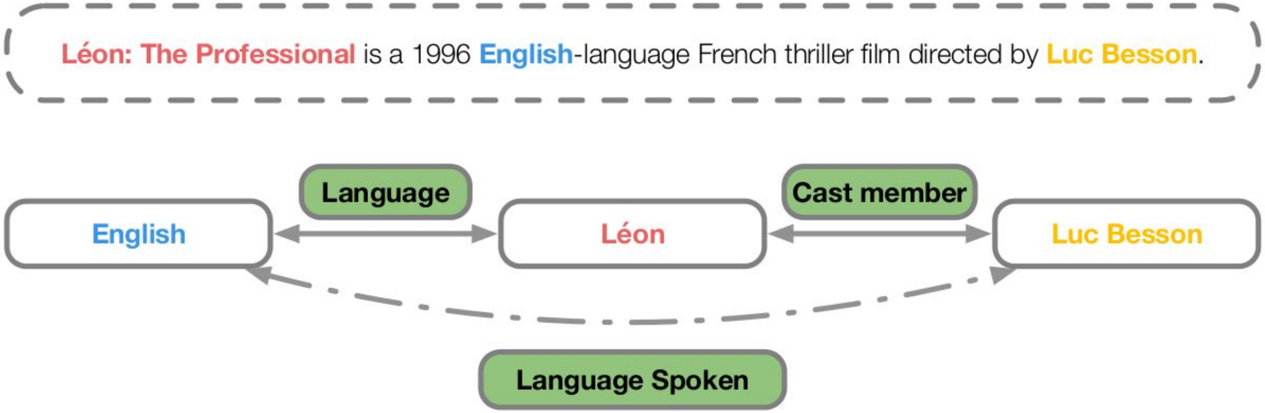
× **GNN needs pre-defined graphs**

Cannot be directly applied on text

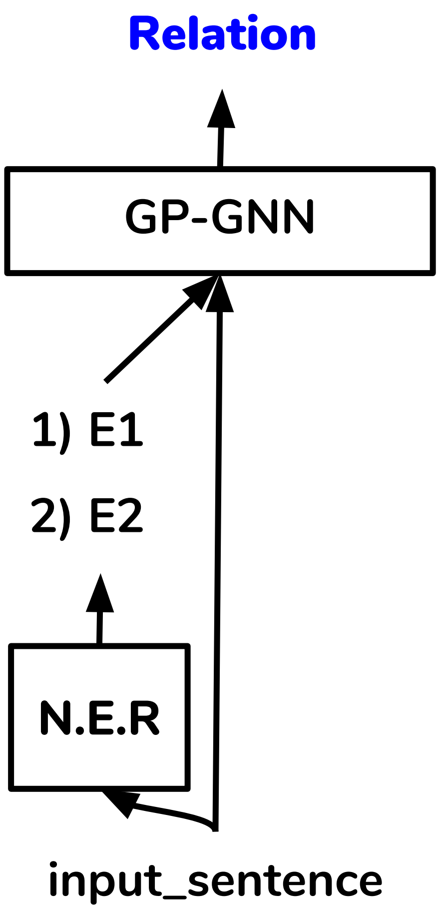
GNN on text [Zhu et al., ACL'19]



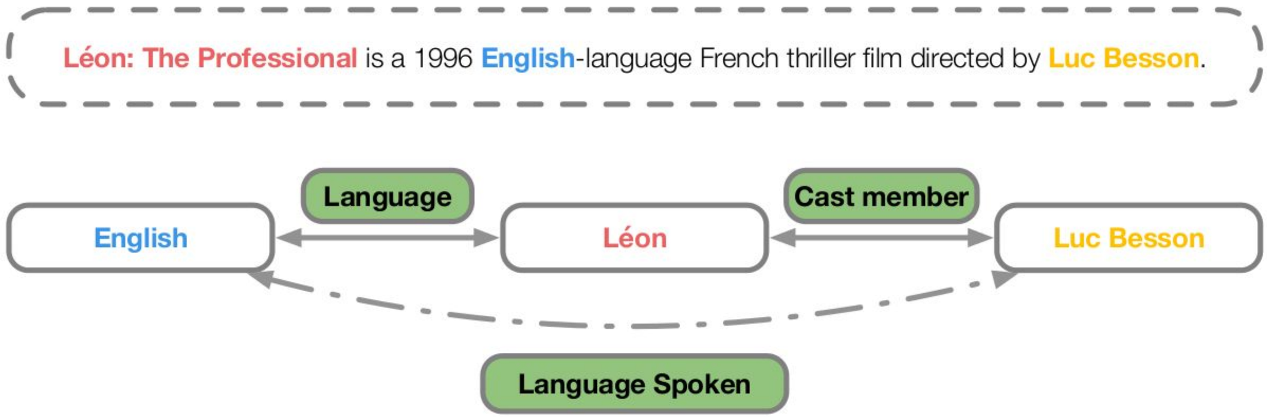
Relational Reasoning



GNN on text [Zhu et al., ACL'19]



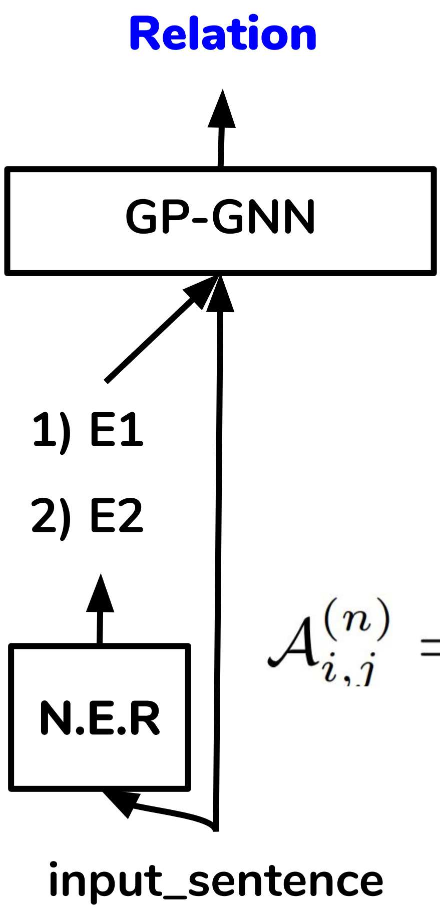
Relational Reasoning

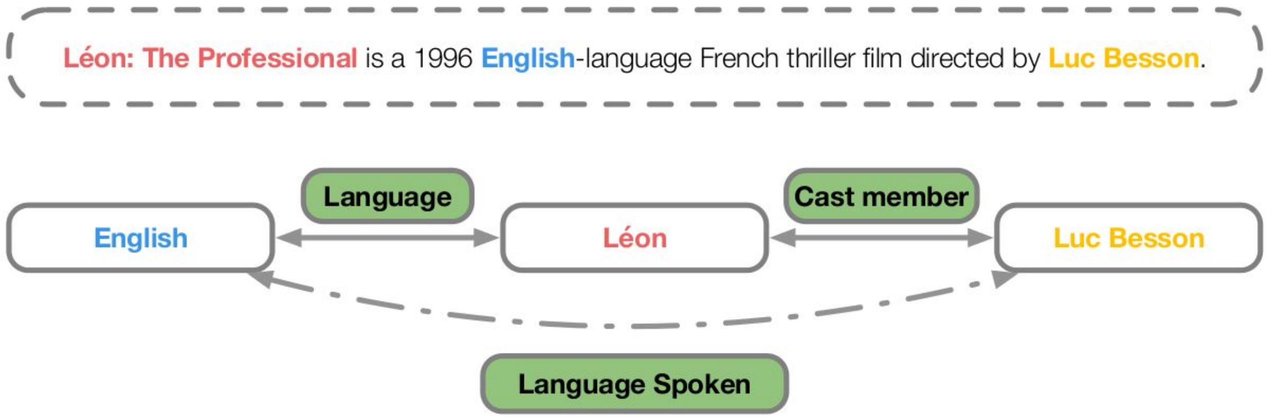


Encode each word
w.r.t given entity pair $E(x_t^{i,j}) = [\boldsymbol{x}_t; \boldsymbol{p}_t^{i,j}]$

GNN on text [Zhu et al., ACL'19]

Relational Reasoning



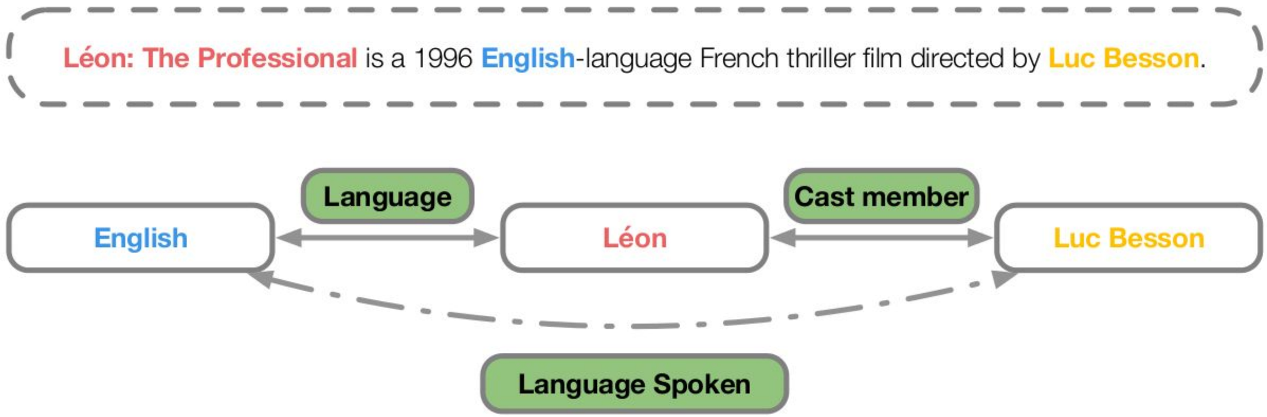


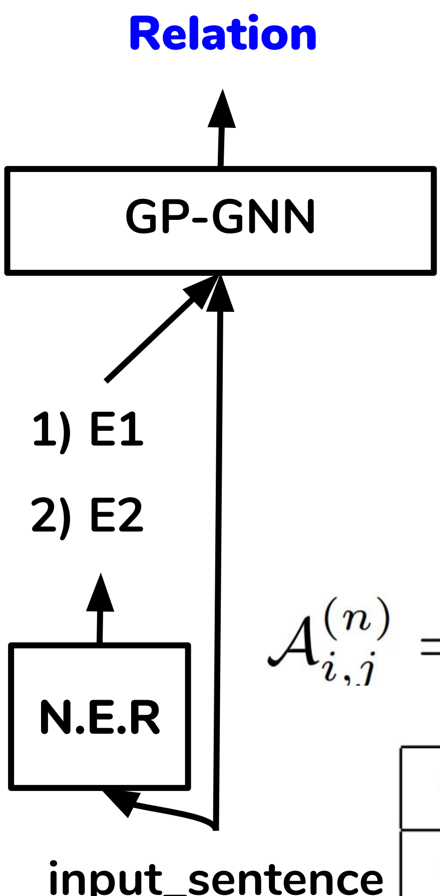
Encode each word w.r.t given entity pair $E(x_t^{i,j}) = [\boldsymbol{x}_t; \boldsymbol{p}_t^{i,j}]$

$$\mathcal{A}_{i,j}^{(n)} = [\mathtt{MLP}_n(\mathtt{BiLSTM}_n((E(x_0^{i,j}), E(x_1^{i,j}), \cdots, E(x_{l-1}^{i,j}))]$$

GNN on text [Zhu et al., ACL'19]

Relational Reasoning





Encode each word w.r.t given entity pair $E(x_t^{i,j}) = [\boldsymbol{x}_t; \boldsymbol{p}_t^{i,j}]$

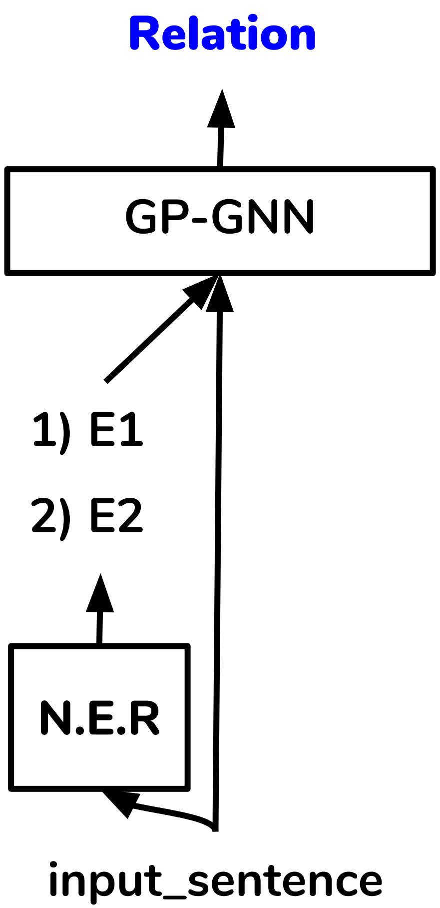
$$\mathcal{A}_{i,j}^{(n)} = [\mathtt{MLP}_n(\mathtt{BiLSTM}_n((E(x_0^{i,j}), E(x_1^{i,j}), \cdots, E(x_{l-1}^{i,j}))]$$

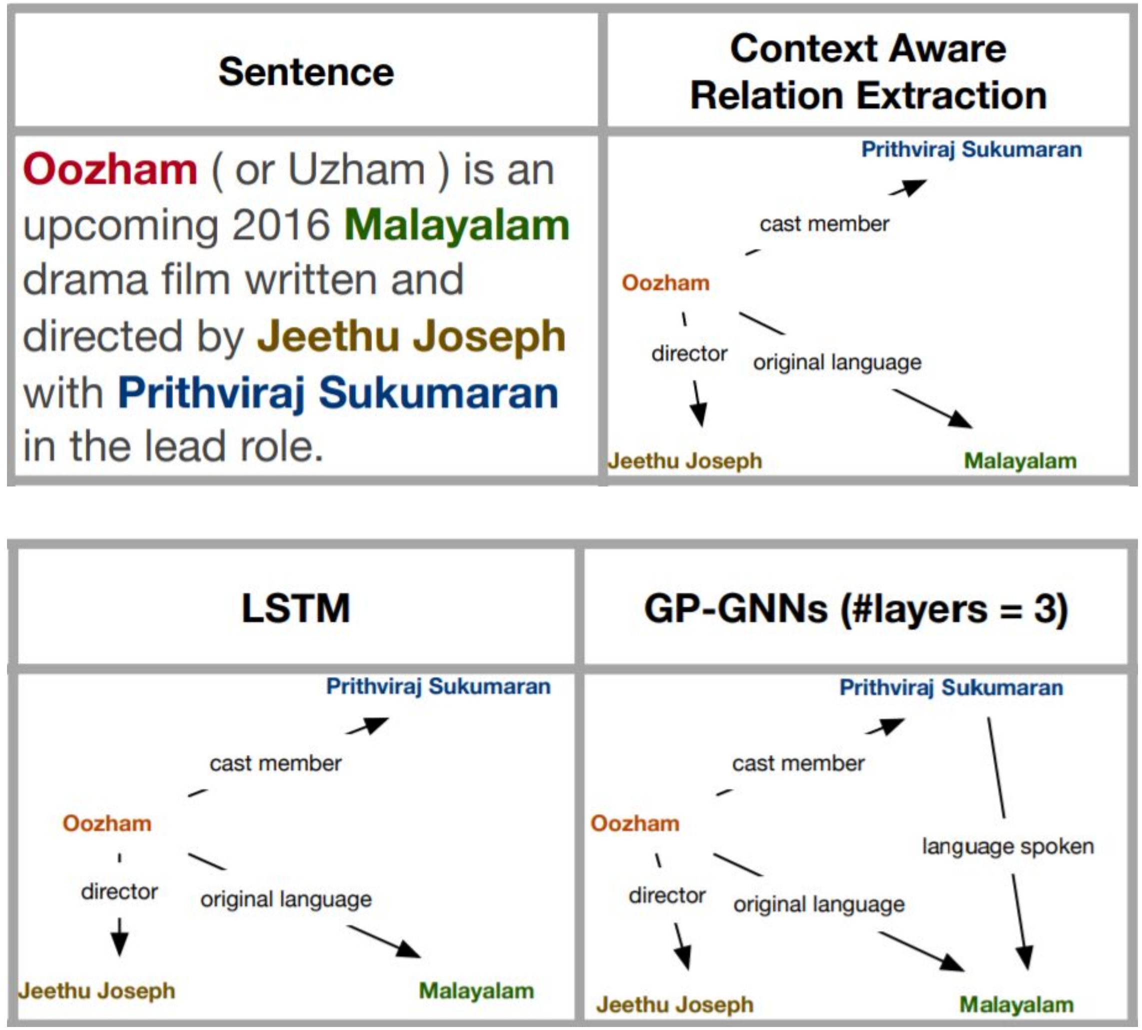
Context-aware RE	87.2
GP-GNN	**88.3**

P@20% on Wikipedia

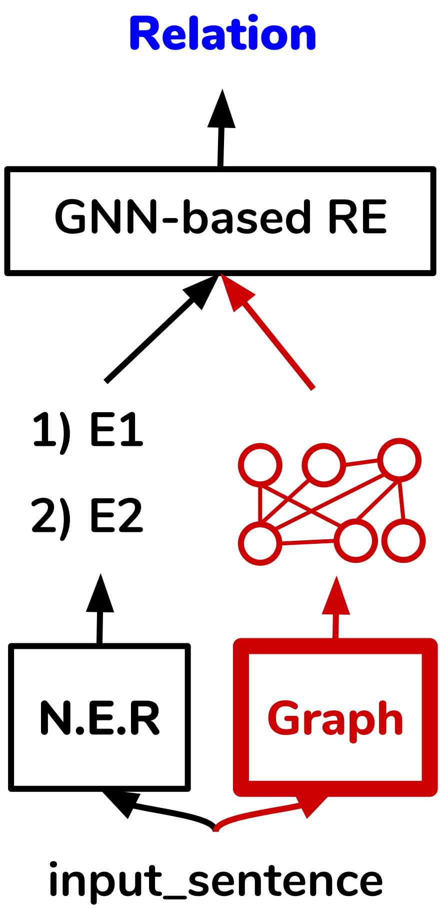
Graphs for RE are induced through GNNs

GNN on text [Zhu et al., ACL'19]





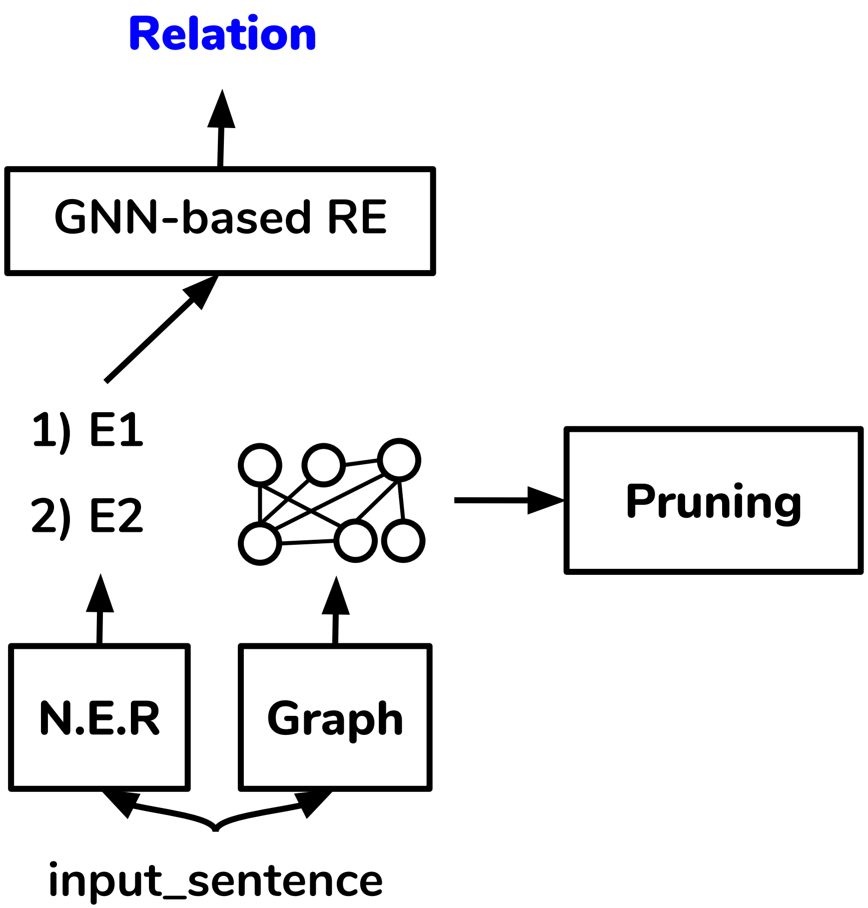
GNN on text [Zhu et al., ACL'19]



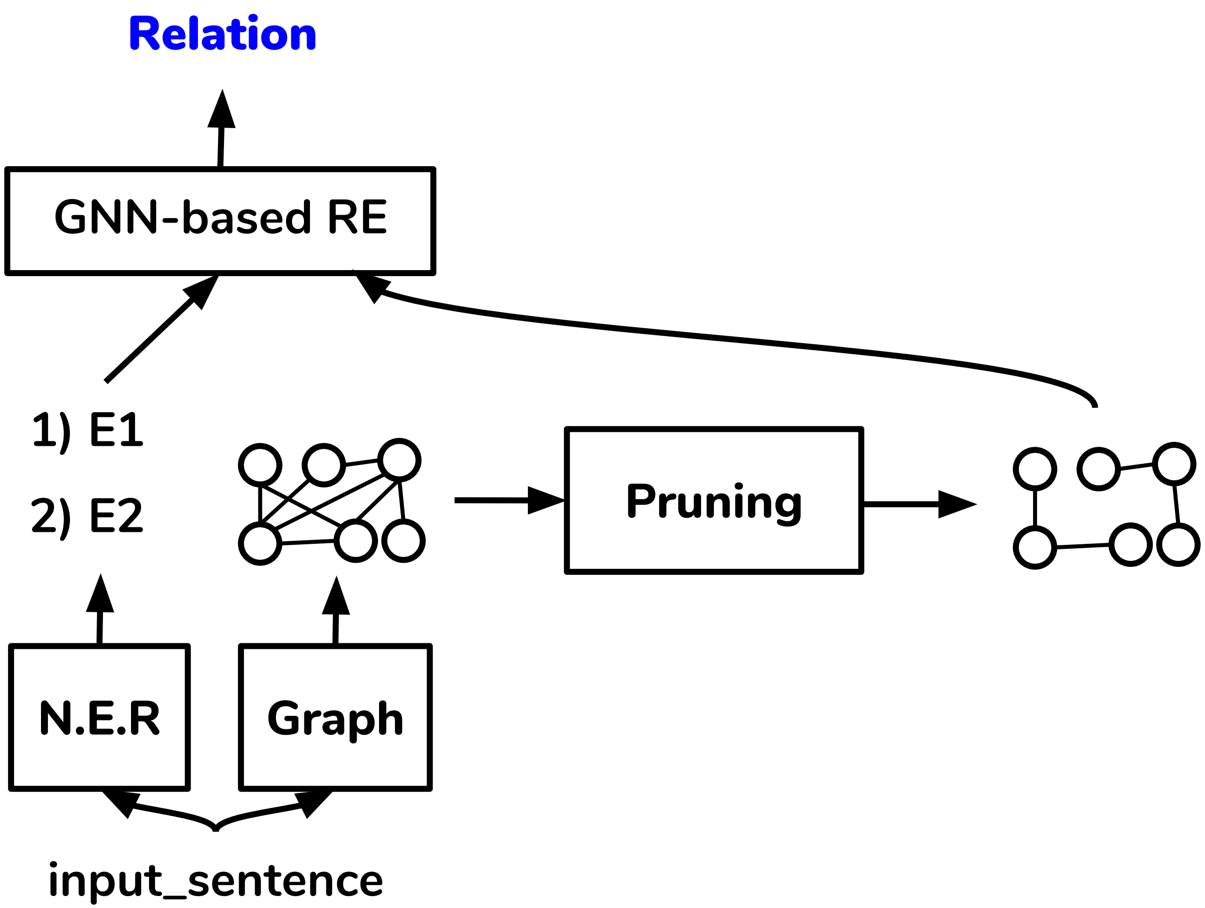
× **Graph may be noisy**

may include unwanted edges

Graph pruning [Zhang et al., EMNLP'18, Guo et al., ACL'19]

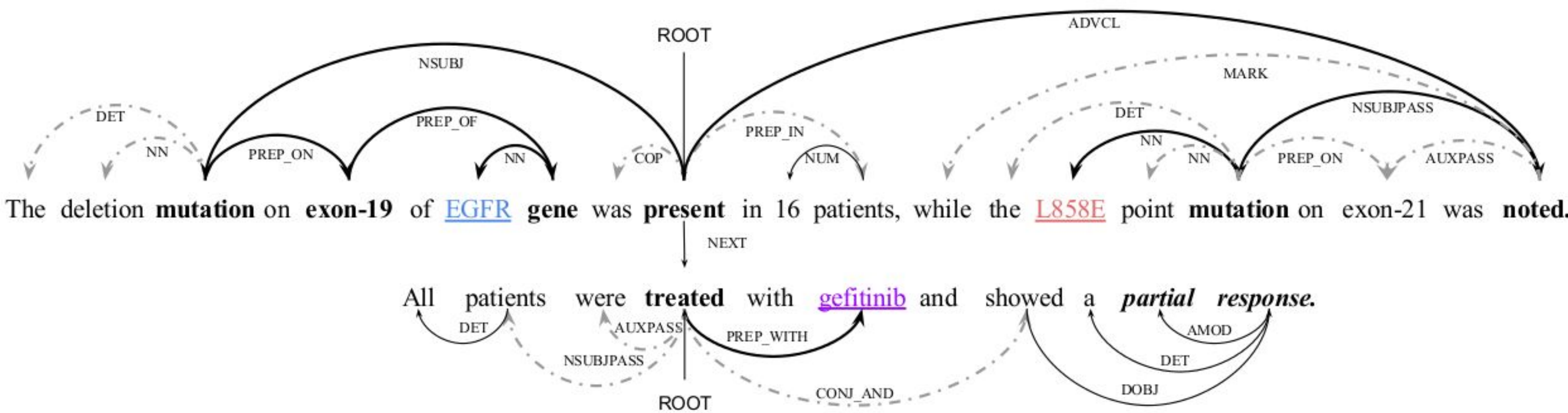


Graph pruning [Zhang et al., EMNLP'18, Guo et al., ACL'19]



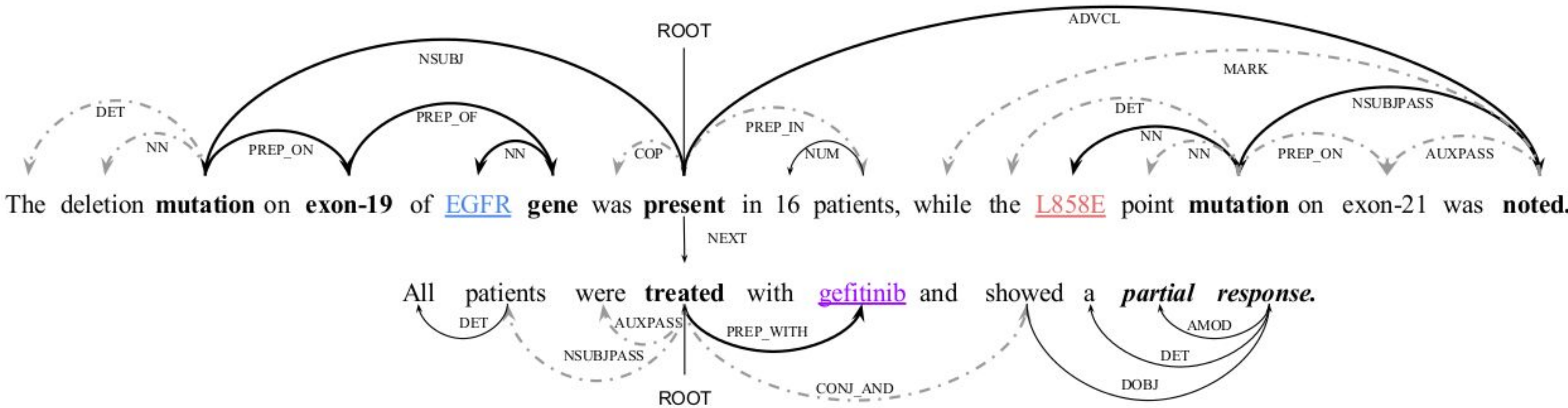
Dependency-based n-ary RE [Guo et al., ACL'19]

Relation (EGFR, L858E, gefitinib) = ?



Dependency-based n-ary RE [Guo et al., ACL'19]

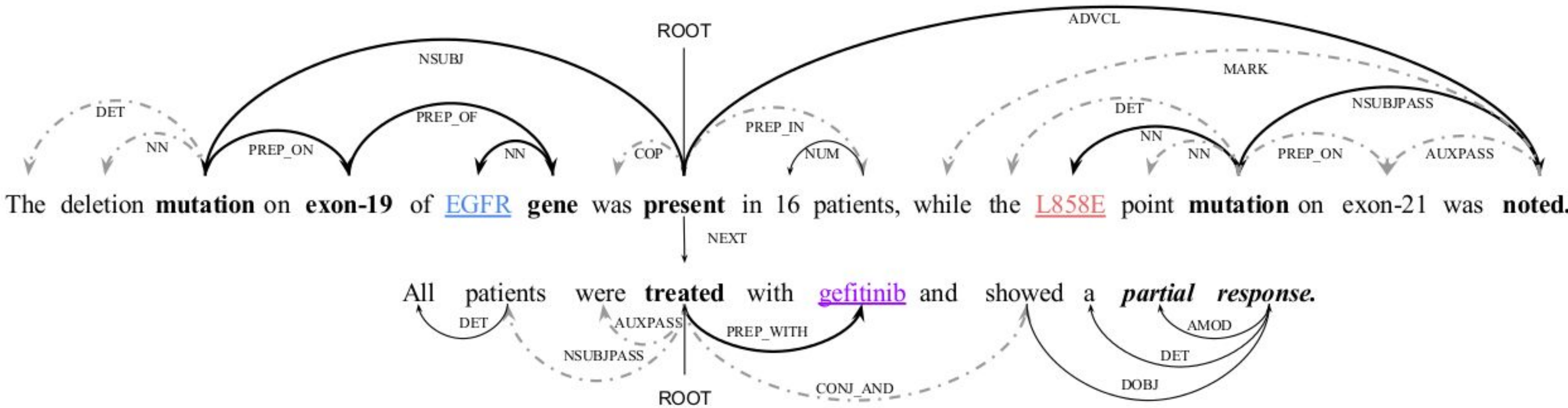
Relation (EGFR, L858E, gefitinib) = ?



- **Pruning improves performance**
  - Shortest path [EMNLP'15] + LCA subtree [ACL'16]
  - LCA subtree + K-hop [EMNLP'18]

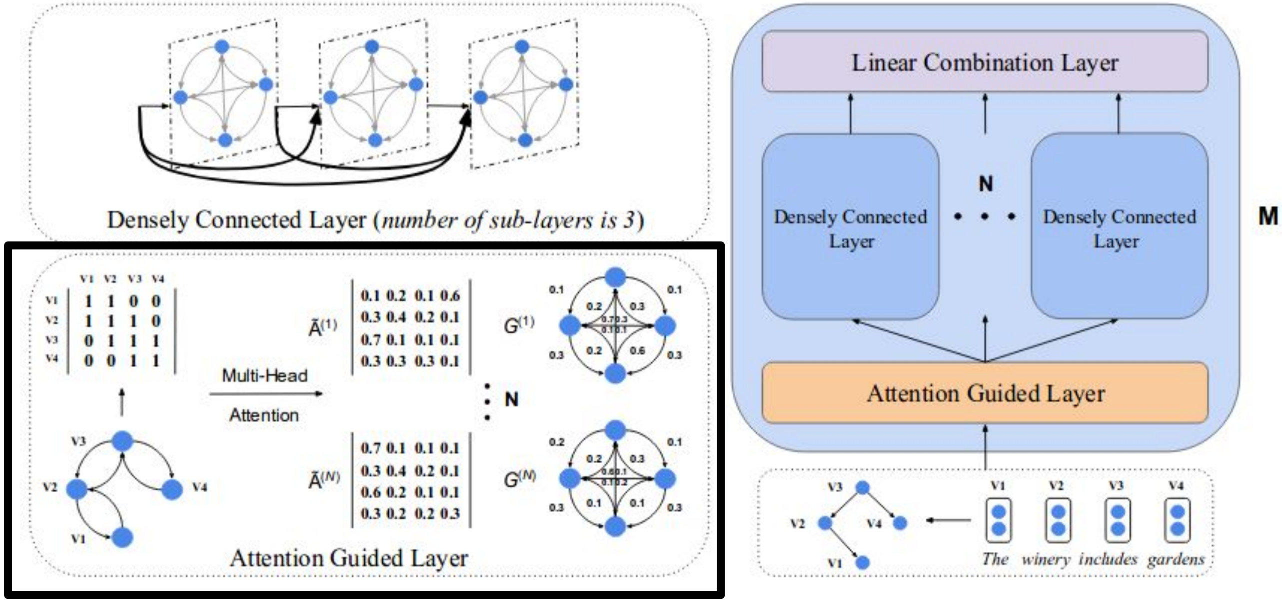
Dependency-based n-ary RE [[Guo et al., ACL'19]]

Relation (EGFR, L858E, gefitinib) = ?



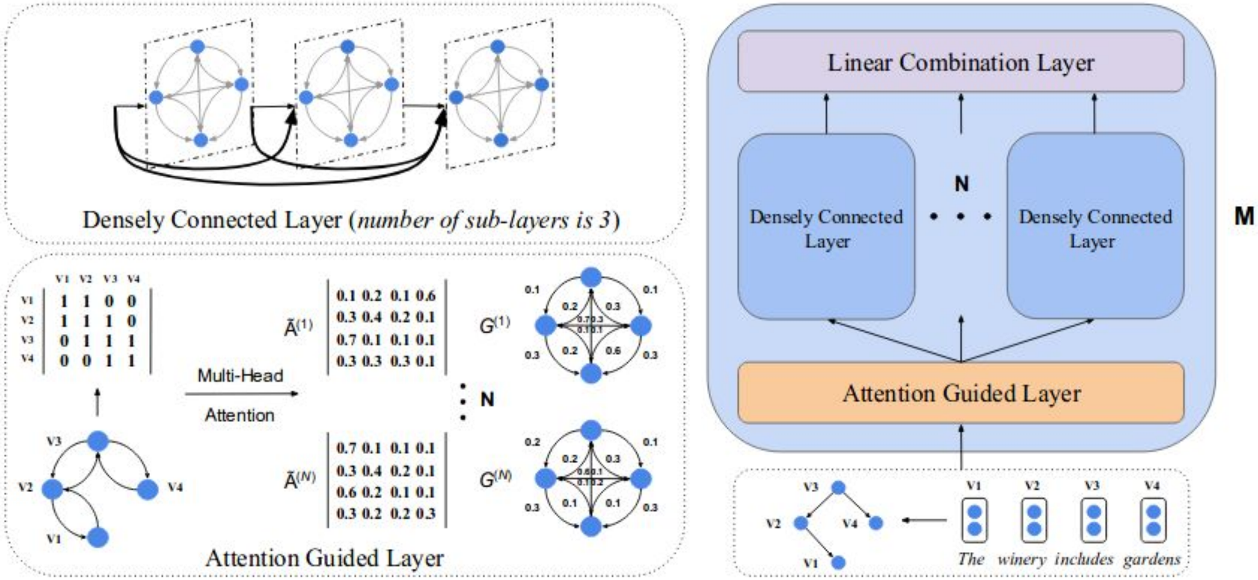
- **Pruning improves performance**
  - Shortest path [EMNLP'15] + LCA subtree [ACL'16]
  - LCA subtree + K-hop [EMNLP'18]
- **Rule-based, may exclude key tokens**

Attention-Guided GCN [Guo et al., ACL'19]



Key step

Attention-Guided GCN [Guo et al., ACL'19]



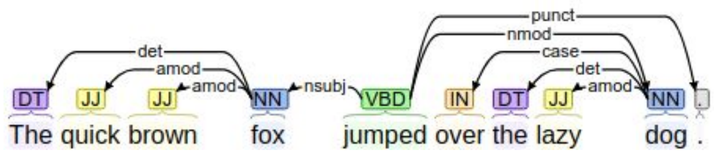
Test Accuracies on PubMed

C-GCN	78.1
AGGCN	**79.7**

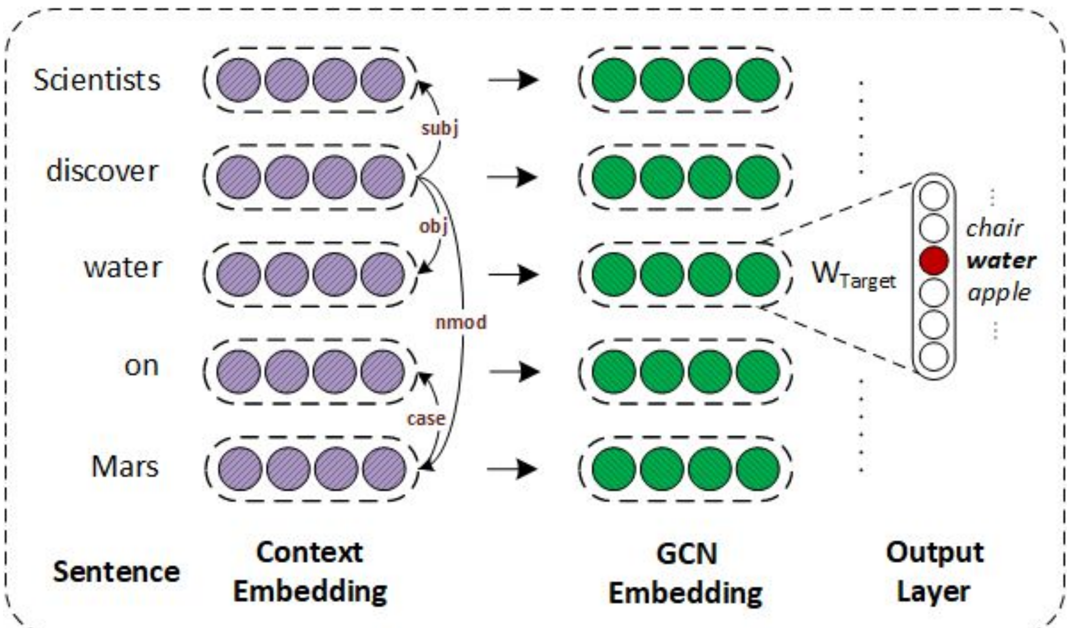
Learning to prune is superior to rule-based pruning

Word Embeddings [[Vashishth et al., ACL'19]]

- Given a sentence, obtain its syntax.



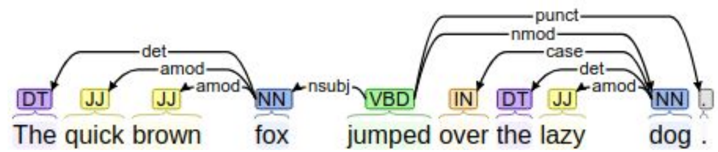
- Exploit syntax for predicting a word.



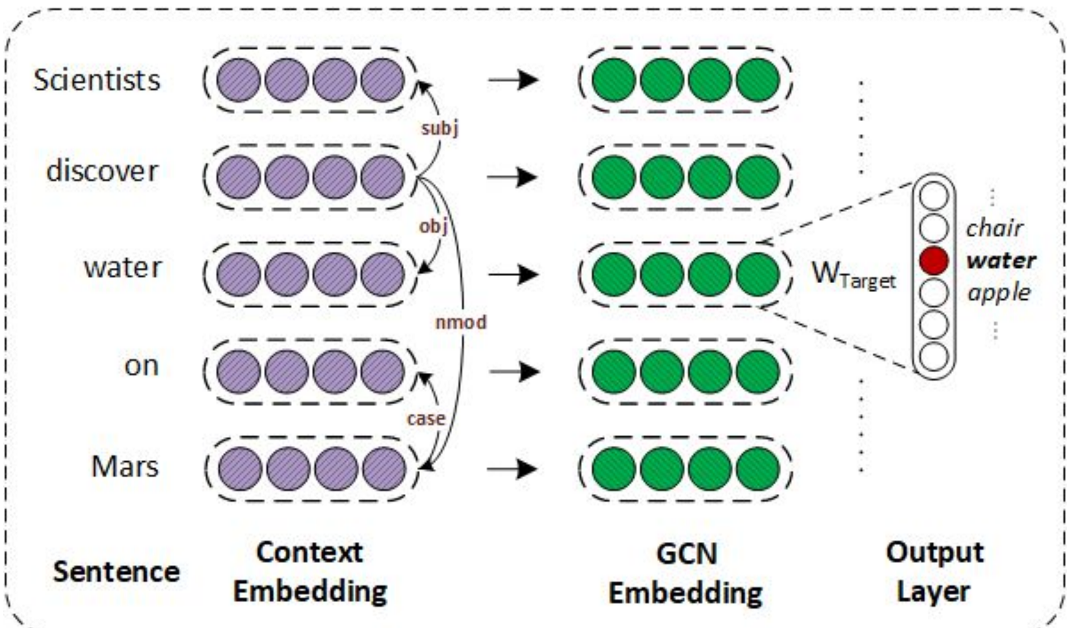
SynGCN (Sentence-level)

Word Embeddings [Vashishth et al., ACL'19]

- Given a sentence, obtain its syntax.



- Exploit syntax for predicting a word.



SynGCN (Sentence-level)

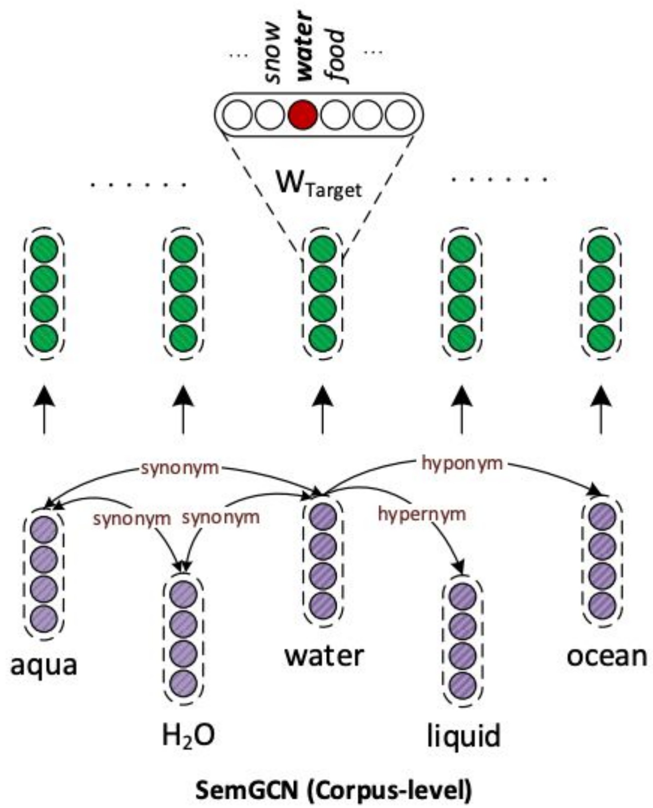
SynGCN

Method	WS353S
Word2vec	71.4
GloVe	69.2
Deps	65.7
EXT	69.6
SynGCN	**73.2**

F1 Score

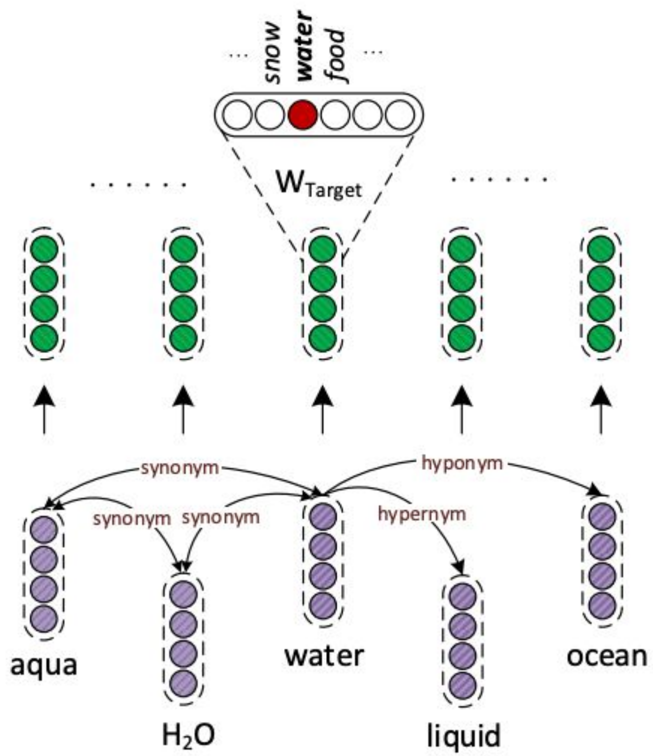
SemGCN [Vashishth et al., ACL'19]

- Exploits semantics in pre-trained word embeddings
- Unlike prior work, SemGCN jointly exploits synonym, hypernym,



SemGCN [Vashishth et al., ACL'19]

- Exploits semantics in pre-trained word embeddings
- Unlike prior work, SemGCN jointly exploits synonym, hypernym,



SemGCN (Corpus-level)

SemGCN

Datasets	WS353
Performance of X	63.0
Retro-fit (X,1)	63.4
Counter-fit (X,2)	60.3
JointReps (X,4)	60.9
SemGCN (X,4)	**64.8**

F1 Score

Syntax, Semantics help word embeddings

Short text classification (STC) [Hu et al., EMNLP'19]

sports

Shawn hit home two runs, as LA defeated Atlanta at Dodger

- online news, queries, reviews, tweets are widespread
- news tagging
- sentiment analysis, query intent classification

Challenges for STC [Hu et al., EMNLP'19]

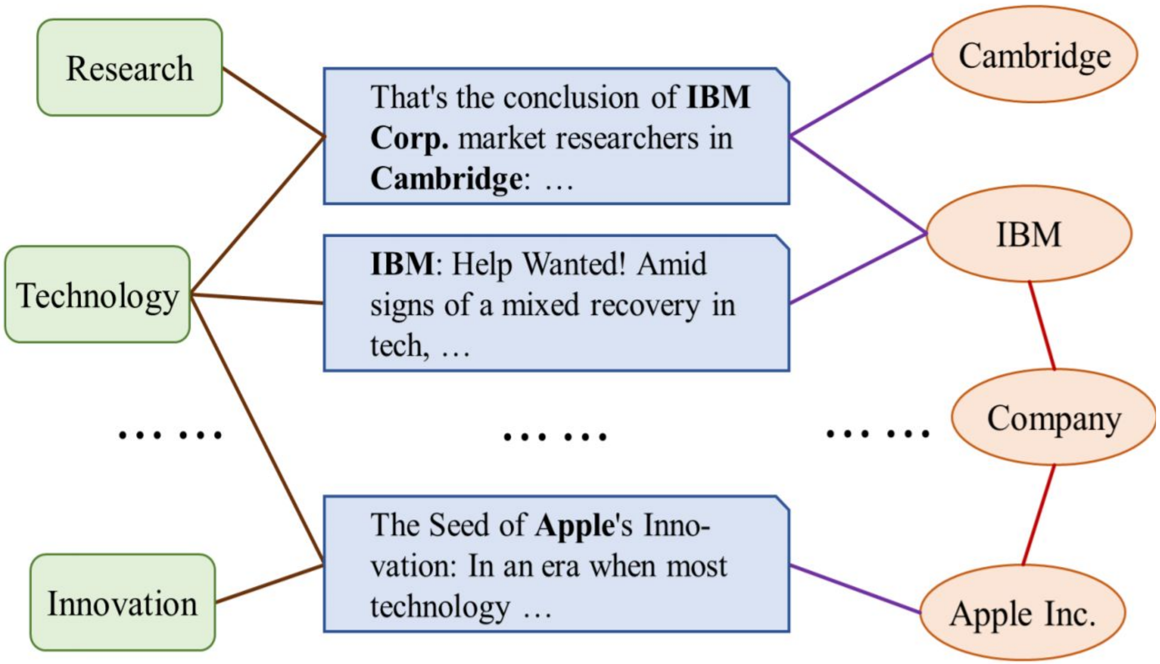
sports

Shawn hit home two runs, as LA defeated Atlanta at Dodger

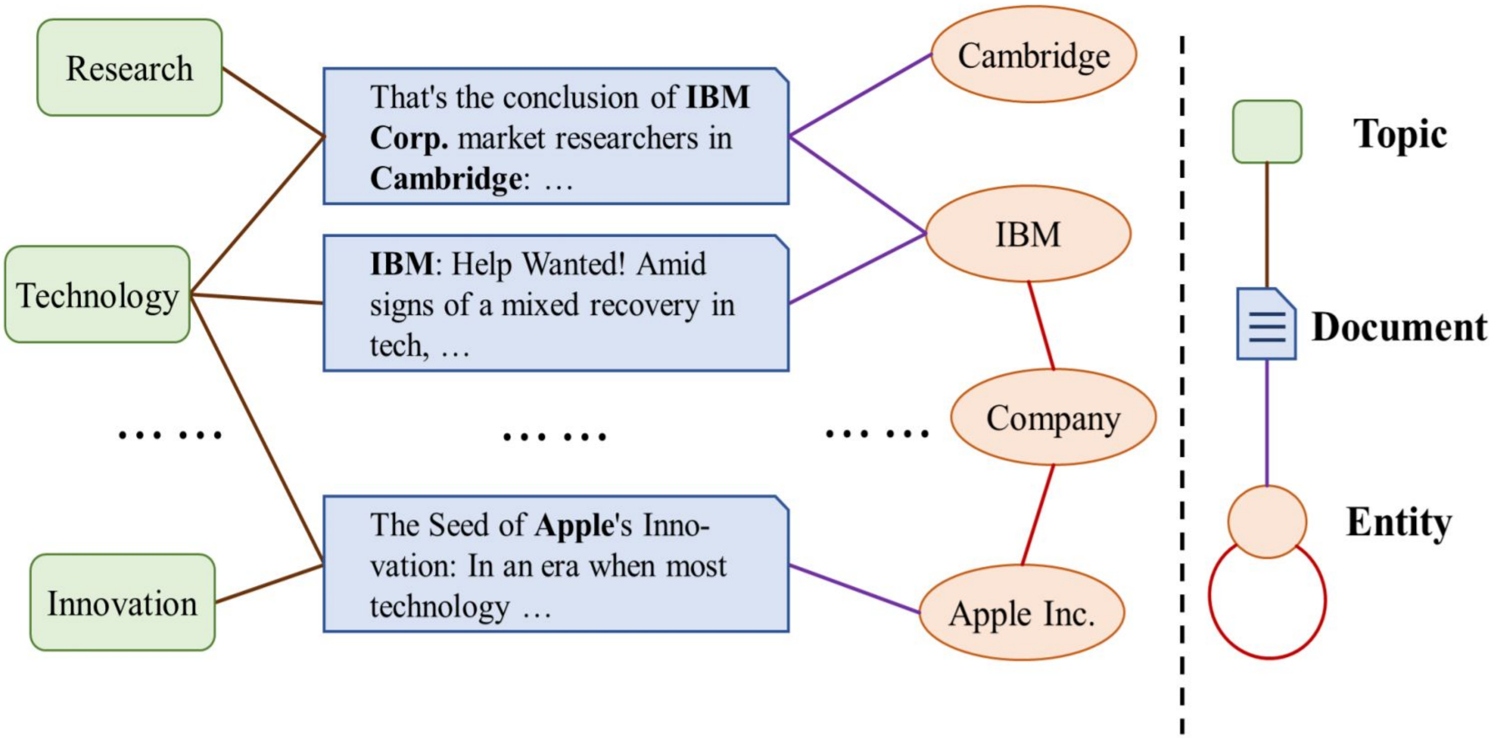
- **scarce labelled data, expensive human labelling**
- **semantically sparse, lack context**

- **Semi-supervised STC on heterogeneous graph**

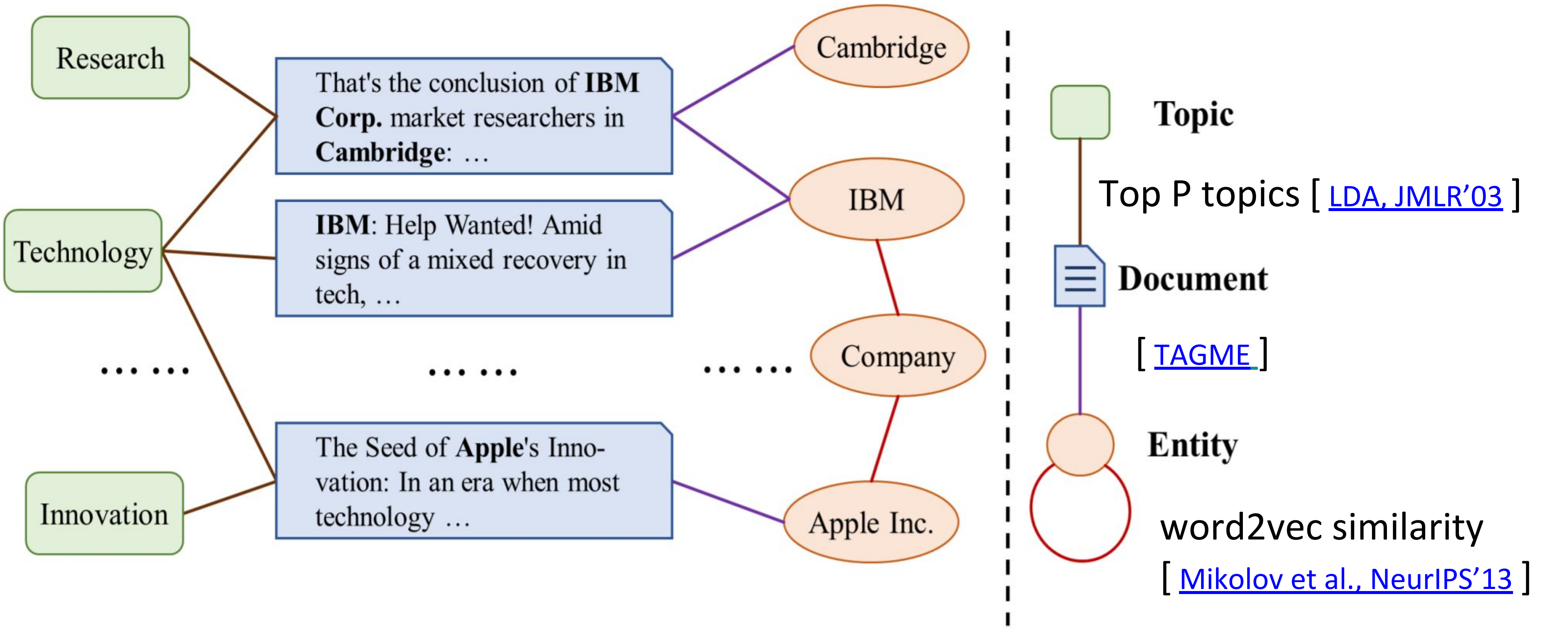
Heterogeneous graph attention [Hu et al., EMNLP'19]



Heterogeneous graph attention [Hu et al., EMNLP'19]



Heterogeneous graph attention [Hu et al., EMNLP'19]

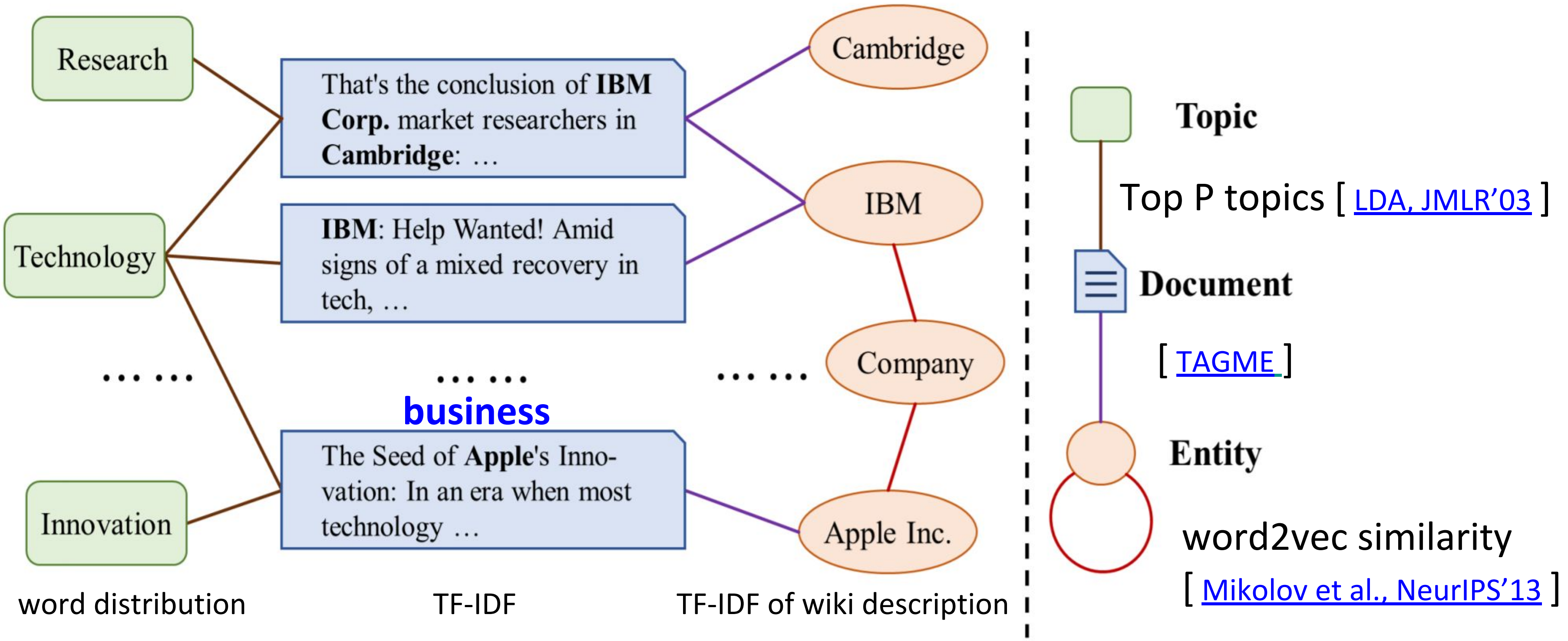


Top P topics [LDA, JMLR'03]

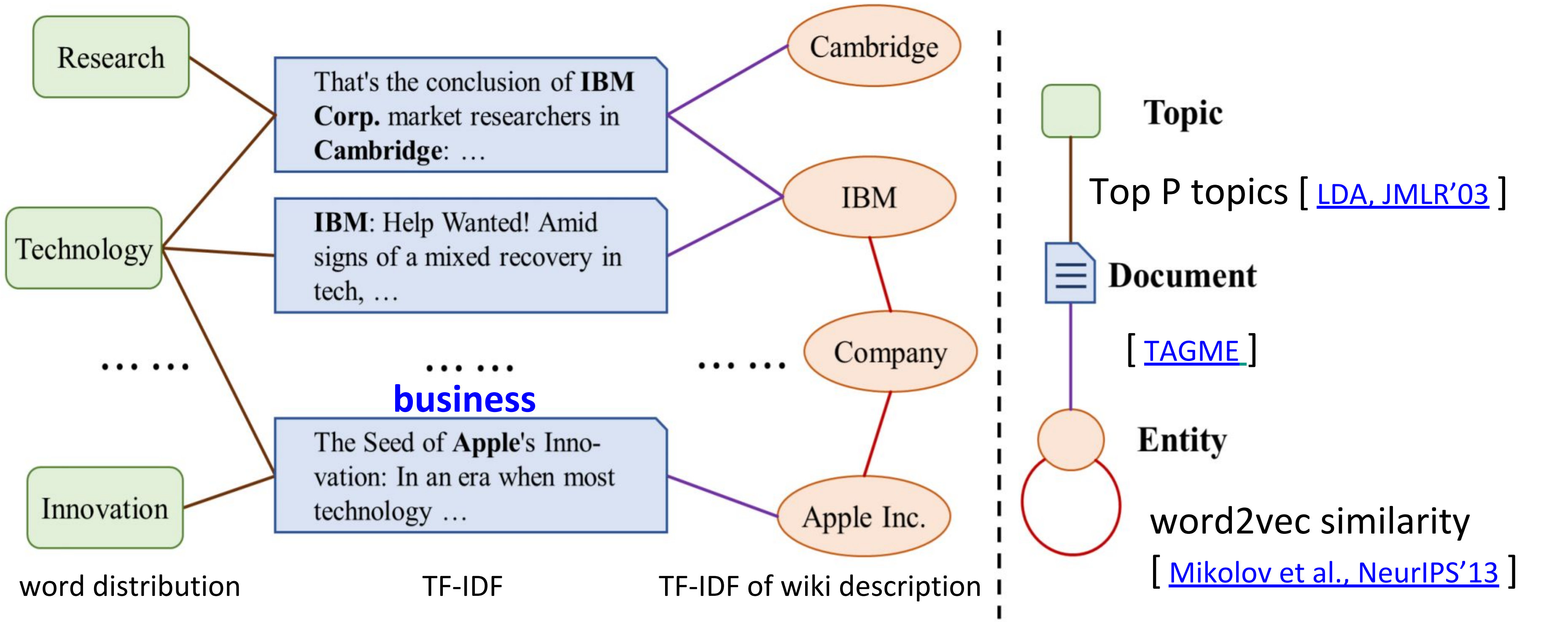
[TAGME]

word2vec similarity
[Mikolov et al., NeurIPS'13]

Heterogeneous graph attention [Hu et al., EMNLP'19]



Heterogeneous graph attention [Hu et al., EMNLP'19]

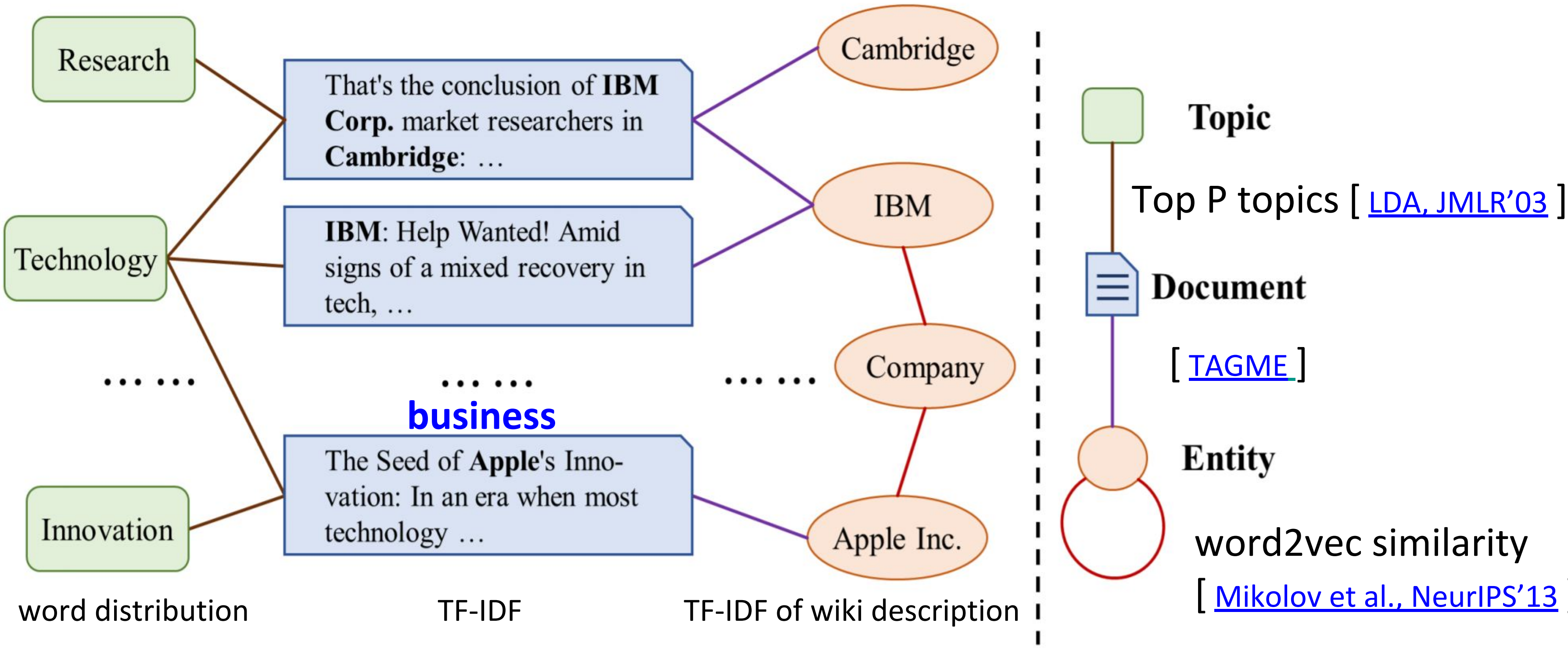


HGC

$$H^{(l+1)} = \sigma\left(\sum_{\tau \in \mathcal{T}} \tilde{A}_\tau \cdot H_\tau^{(l)} \cdot W_\tau^{(l)}\right)$$

$$\tilde{A}_\tau \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}_\tau|}$$

Heterogeneous graph attention [Hu et al., EMNLP'19]



HGC

$$H^{(l+1)} = \sigma\Big(\sum_{\tau \in \mathcal{T}} \tilde{A}_\tau \cdot H_\tau^{(l)} \cdot W_\tau^{(l)}\Big)$$

$$\tilde{A}_\tau \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}_\tau|}$$

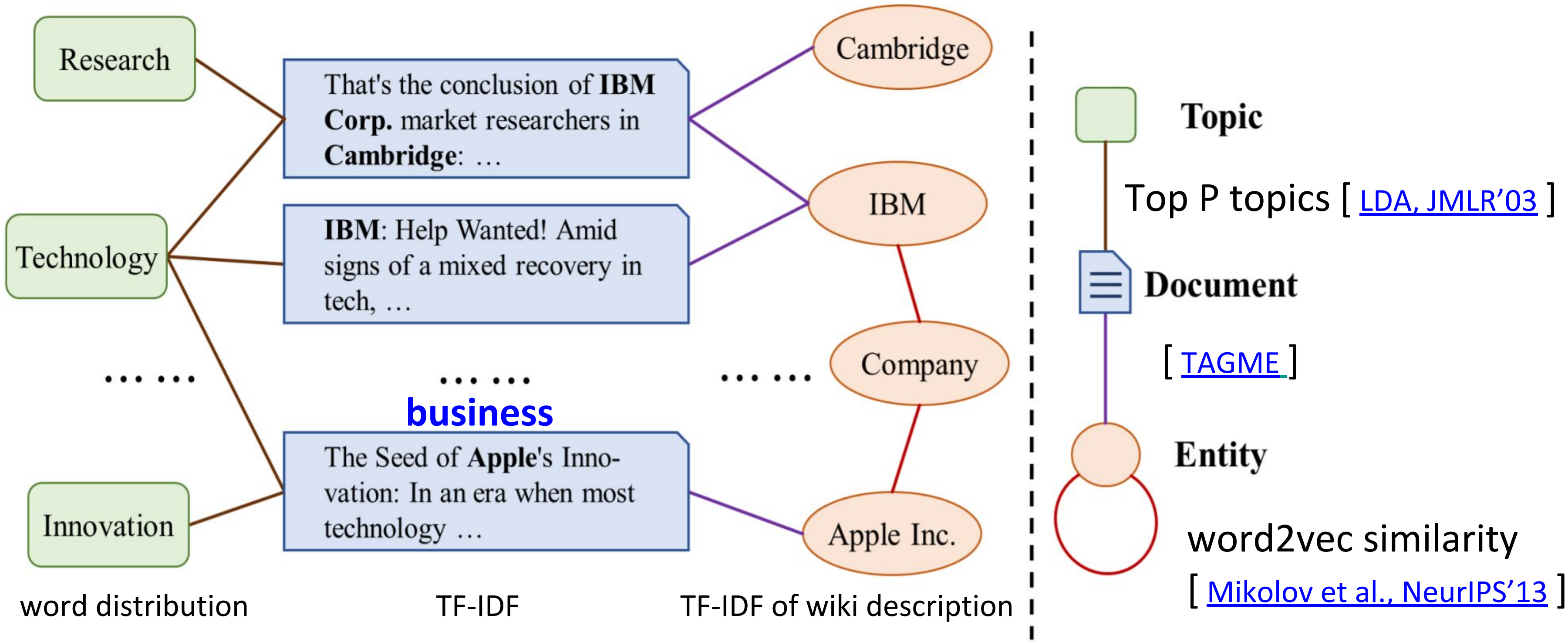
type-level attention

$$h_\tau = \sum_{v'} \tilde{A}_{vv'} h_{v'}$$

$$a_\tau = \sigma(\mu_\tau^T \cdot [h_v || h_\tau])$$

$$\alpha_\tau = \frac{\exp(a_\tau)}{\sum_{\tau' \in \mathcal{T}} \exp(a_{\tau'})}$$

Heterogeneous graph attention [Hu et al., EMNLP'19]



HGC

$$H^{(l+1)} = \sigma\Big(\sum_{\tau \in \mathcal{T}} \tilde{A}_\tau \cdot H_\tau^{(l)} \cdot W_\tau^{(l)}\Big)$$

$$\tilde{A}_\tau \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}_\tau|}$$

type-level attention

$$h_\tau = \textstyle\sum_{v'} \tilde{A}_{vv'} h_{v'}$$

$$a_\tau = \sigma(\mu_\tau^T \cdot [h_v || h_\tau])$$

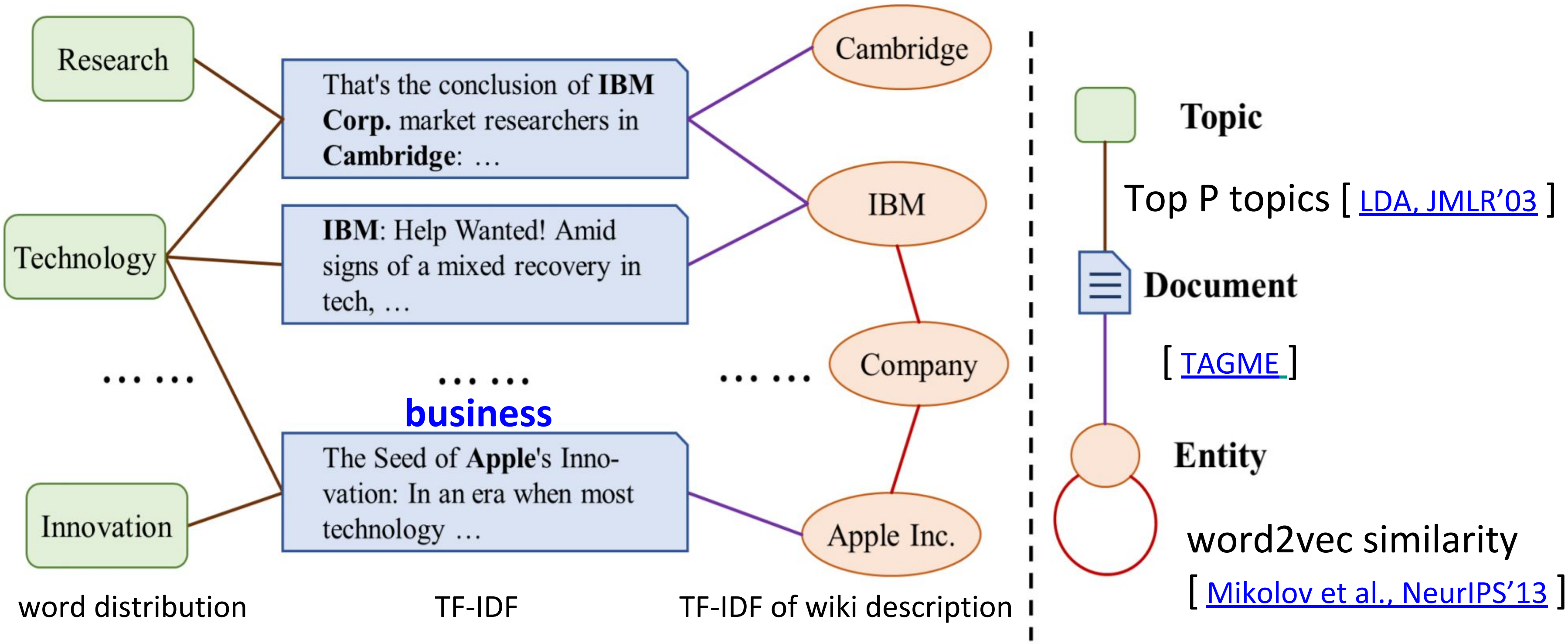
$$\alpha_\tau = \frac{\exp(a_\tau)}{\sum_{\tau' \in \mathcal{T}} \exp(a_{\tau'})}$$

node-level attention

$$b_{vv'} = \sigma(\nu^T \cdot \alpha_{\tau'} [h_v || h_{v'}])$$

$$\beta_{vv'} = \frac{\exp(b_{vv'})}{\sum_{i \in \mathcal{N}_v} \exp(b_{vi})}$$

Heterogeneous graph attention [Hu et al., EMNLP'19]



HGC

$$H^{(l+1)} = \sigma\Big(\sum_{\tau \in \mathcal{T}} \tilde{A}_\tau \cdot H_\tau^{(l)} \cdot W_\tau^{(l)}\Big)$$

$$\tilde{A}_\tau \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}_\tau|}$$

type-level attention

$$h_\tau = \textstyle\sum_{v'} \tilde{A}_{vv'} h_{v'}$$

$$a_\tau = \sigma(\mu_\tau^T \cdot [h_v || h_\tau])$$

$$\alpha_\tau = \frac{\exp(a_\tau)}{\sum_{\tau' \in \mathcal{T}} \exp(a_{\tau'})}$$

node-level attention

$$b_{vv'} = \sigma(\nu^T \cdot \alpha_{\tau'} [h_v || h_{v'}])$$

$$\beta_{vv'} = \frac{\exp(b_{vv'})}{\sum_{i \in \mathcal{N}_v} \exp(b_{vi})}$$

$$H^{(l+1)} = \sigma\Big(\sum_{\tau \in \mathcal{T}} \mathcal{B}_\tau \cdot H_\tau^{(l)} \cdot W_\tau^{(l)}\Big)$$

$$\mathcal{L} = -\sum_{i \in D_{\text{train}}} \sum_{j=1}^{C} Y_{ij} \cdot \log Z_{ij} + \eta \|\Theta\|_2$$

Experiments [Hu et al., EMNLP'19]

AGNews dataset

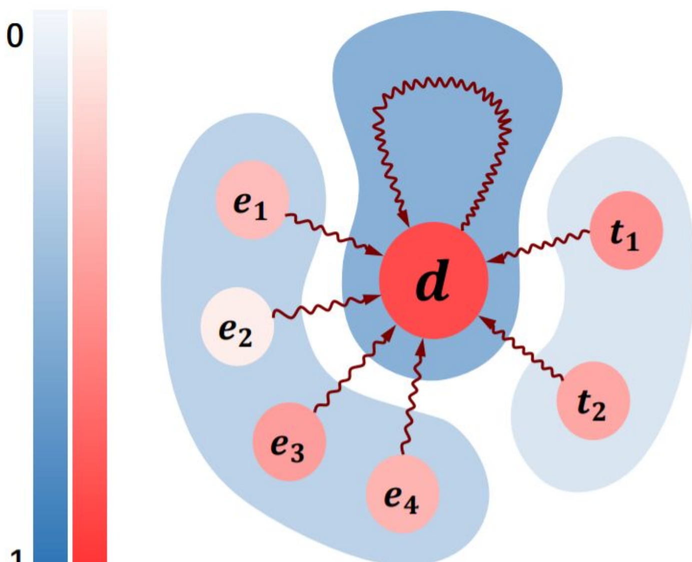
Method	F1
TextGCN	67.61
GCN-HIN	70.87
HGAT w/o Att	70.97
HGAT - type	71.54
HGAT - node	71.76
HGAT	**72.10**

Experiments [Hu et al., EMNLP'19]

AGNews dataset

Method	F1
TextGCN	67.61
GCN-HIN	70.87
HGAT w/o Att	70.97
HGAT - type	71.54
HGAT - node	71.76
HGAT	**72.10**

Heterogeneous graphs are effective for STC



Short Text d

Shawn Green (Entity e_1) hit two home runs, as **Los Angeles (Entity e_2)** defeated the **Atlanta Braves (Entity e_3)** 7-4 in a battle of National League division leaders at **Dodger Stadium (Entity e_4)**.

Topic t_1:

game	sox	red	beat	team
clubs	season	win	astros	run

Topic t_2:

wins	awards	prix	star	prize
greek	china	grand	british	olympics

Summary of GNNs for Text

- **Takeaways**
  - **Joint models:** multiple events, joint event + relation
  - Graph **Construction, induction, pruning** for RE
  - **Corpus-level** graphs (temporal, semantic) help

Summary of GNNs for Text

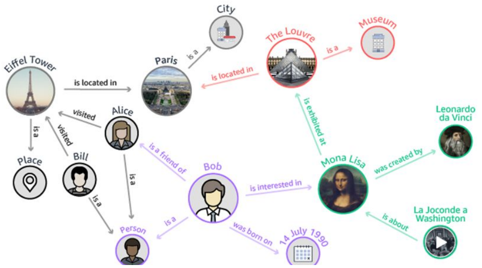
- **Takeaways**
  - **Joint models:** multiple events, joint event + relation
  - Graph **Construction, induction, pruning** for RE
  - **Corpus-level** graphs (temporal, semantic) help

- **Future directions**
  - Edge Pruning for **graph-level tasks**
  - GNN for zero-shot RE
    - GNN for Long-tail RE [Zhang et al., NAACL'19]

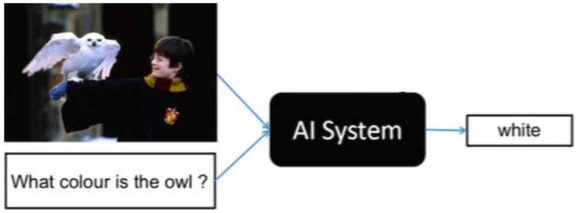
Applications of Graph Neural Nets

- ✓ Semantic Role Labelling, Machine Translation
- ✓ Text Classification, Extraction

- Knowledge Graphs



- Vision + NLP



GNN + Knowledge Graph (KG)

GNN + KG

KG Embedding

CaRe	EMNLP'19
AttGCN	ACL'19
LAN	AAAI'19
SACN	AAAI'19
GMatching	EMNLP'18
R-GCN	ESWC'18
OOKB	IJCAI'17

GNN + Knowledge Graph (KG)

GNN + KG

KG Embedding

CaRe	EMNLP'19
AttGCN	ACL'19
LAN	AAAI'19
SACN	AAAI'19
GMatching	EMNLP'18
R-GCN	ESWC'18
OOKB	IJCAI'17

Label Correlation

DGP	CVPR'19
KATT	NAACL'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

GNN + Knowledge Graph (KG)

GNN + KG

KG Embedding

CaRe	EMNLP'19
AttGCN	ACL'19
LAN	AAAI'19
SACN	AAAI'19
GMatching	EMNLP'18
R-GCN	ESWC'18
OOKB	IJCAI'17

Label Correlation

DGP	CVPR'19
KATT	NAACL'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

Text Generation

PaperRobot	ACL'19
GraphWriter	NAACL'19
CCM	IJCAI'18

GNN + Knowledge Graph (KG)

GNN + KG

KG Embedding

CaRe	EMNLP'19
AttGCN	ACL'19
LAN	AAAI'19
SACN	AAAI'19
GMatching	EMNLP'18
R-GCN	ESWC'18
OOKB	IJCAI'17

Recommender System

HA-GNN	EMNLP'19
KBRD	EMNLP'19
GENI	KDD'19
KGSRU	IJCAI'19
GATMCO	KDD'19
KGAT	KDD'19
KGCN	KDD'19
KGCN	WWW'19

Text Generation

PaperRobot	ACL'19
GraphWriter	NAACL'19
CCM	IJCAI'18

Label Correlation

DGP	CVPR'19
KATT	NAACL'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

GNN + Knowledge Graph (KG)

GNN + KG

KG / Entity Alignment

KECG	EMNLP'19
GMN	ACL'19
MuGNN	ACL'19
RD-GCN	IJCAI'19
VR-GCN	IJCAI'19
CLKGCN	EMNLP'18

KG Embedding

CaRe	EMNLP'19
AttGCN	ACL'19
LAN	AAAI'19
SACN	AAAI'19
GMatching	EMNLP'18
R-GCN	ESWC'18
OOKB	IJCAI'17

Recommender System

HA-GNN	EMNLP'19
KBRD	EMNLP'19
GENI	KDD'19
KGSRU	IJCAI'19
GATMCO	KDD'19
KGAT	KDD'19
KGCN	KDD'19
KGCN	WWW'19

Text Generation

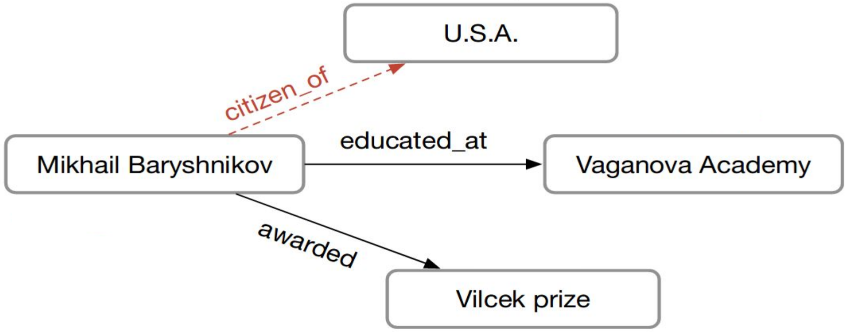
PaperRobot	ACL'19
GraphWriter	NAACL'19
CCM	IJCAI'18

Label Correlation

DGP	CVPR'19
KATT	NAACL'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

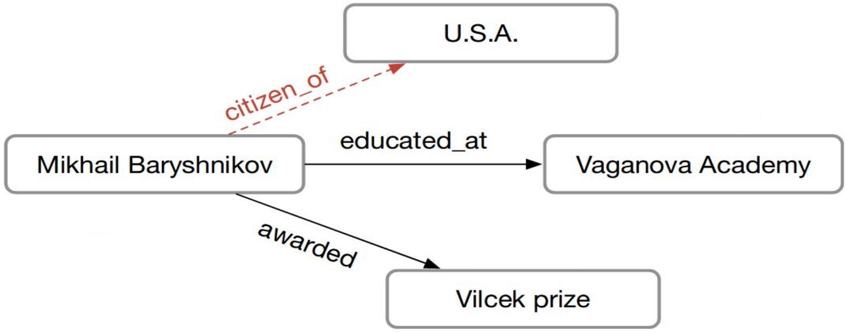
GNNs for KG Embedding

$$h_s^{\{l+1\}} = \sigma\left(\sum_{r\in\mathcal{R}} \sum_{o\in\mathcal{N}_r(s)} f\big(s, r, o, l\big)\right)$$



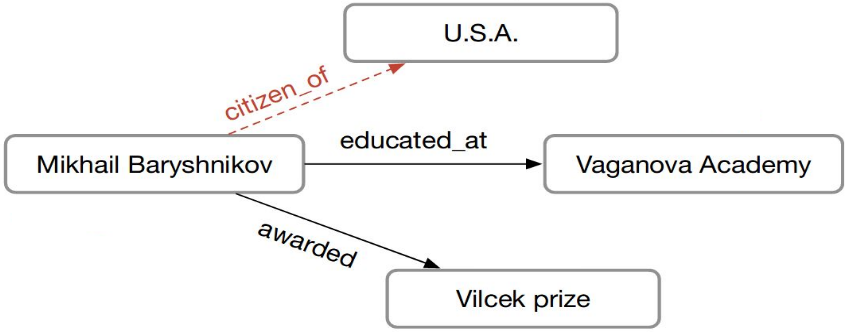
GNNs for KG Embedding

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



GNNs for KG Embedding

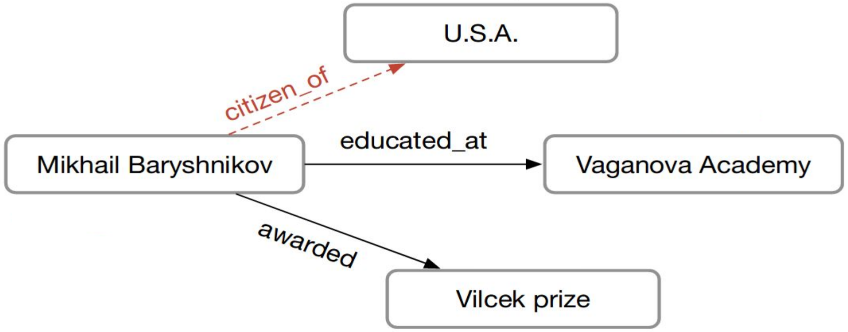
$$h_s = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{o \in \mathcal{N}_r(s)} f\left(s, r, o\right)\right)$$



$f\left(s, r, o\right)$	Method
$W_r \; h_o$	**R-GCN**, ESWC'18

GNNs for KG Embedding

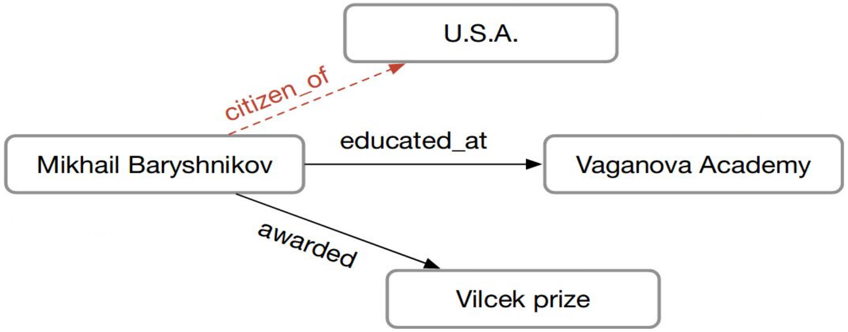
$$h_s = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{o \in \mathcal{N}_r(s)} f\Big(s, r, o\Big)\right)$$



$f\Big(s, r, o\Big)$	Method
$W_r \; h_o$	**R-GCN**, ESWC'18
$\alpha_r \; W \; h_o$	**SACN**, AAAI'19

GNNs for KG Embedding

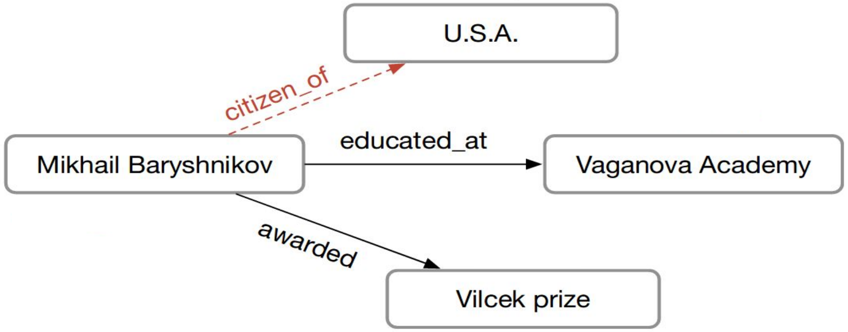
$$h_s = \sigma\left(\sum_{r\in\mathcal{R}} \sum_{o\in\mathcal{N}_r(s)} f\left(s, r, o\right)\right)$$



$f\left(s, r, o\right)$	Method
$W_r \; h_o$	**R-GCN**, ESWC'18
$\alpha_r \; W \; h_o$	**SACN**, AAAI'19
$W \; \Phi\left(h_r, h_o\right)$	**VR-GCN**, IJCAI'19

GNNs for KG Embedding

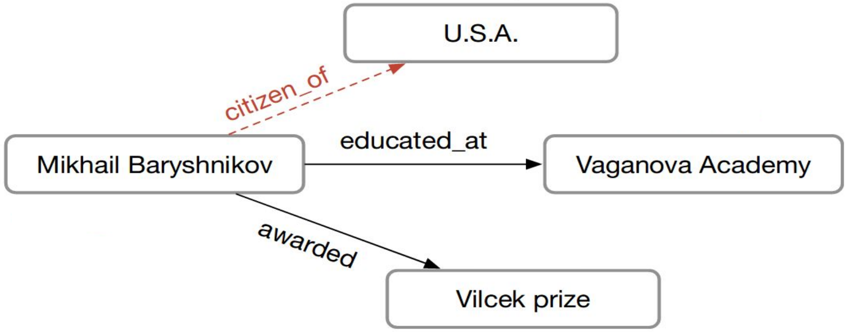
$$h_s = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{o \in \mathcal{N}_r(s)} f\left(s, r, o\right)\right)$$



$f\left(s, r, o\right)$	Method
$W_r \; h_o$	**R-GCN**, ESWC'18
$\alpha_r \; W \; h_o$	**SACN**, AAAI'19
$W \; \Phi\left(h_r, h_o\right)$	**VR-GCN**, IJCAI'19
$\alpha_{sro} \; W \; h_{sro}$	**KBGAT**, ACL'19

GNNs for KG Embedding

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



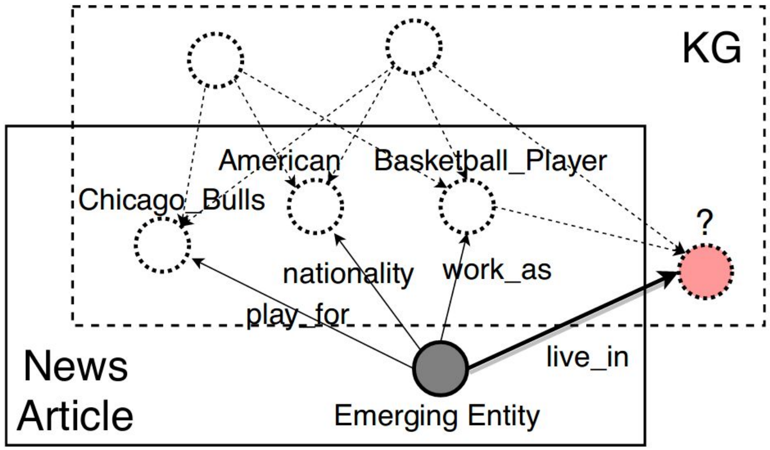
$f\left(s,r,o\right)$	Method
$W_r\ h_o$	**R-GCN**, ESWC'18
$\alpha_r\ W\ h_o$	**SACN**, AAAI'19
$W\ \Phi\left(h_r, h_o\right)$	**VR-GCN**, IJCAI'19
$\alpha_{sro}\ W\ h_{sro}$	**KBGAT**, ACL'19

MRR on WN18RR

R-GCN	0.42
SACN	0.47
VR-GCN	0.48
KBGAT	0.44
RotationH	**0.49**

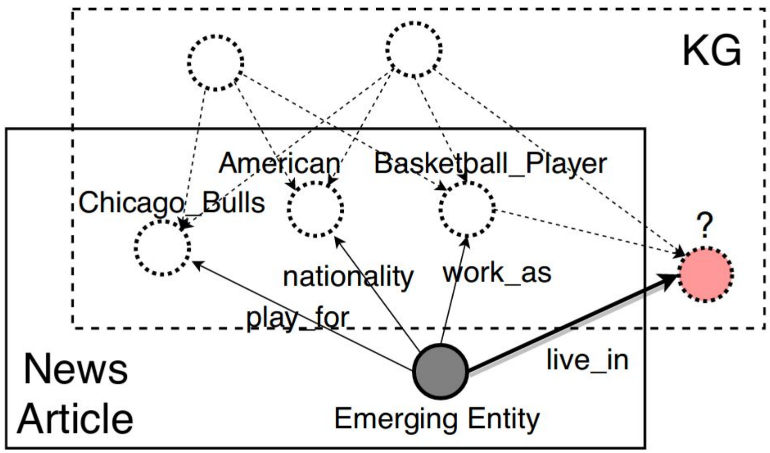
Close to current SOTA

Inductive KG-embedding [Wang et al., AAAI'19]



✕ **Retraining is infeasible**

Inductive KG-embedding [Wang et al., AAAI'19]



× **Retraining is infeasible**

Properties

- **Redundancy-aware**
  play_for Chicago_Bulls => works_as Basketball_player
- **Query relation-aware**
  exploit "live_in" at test time
- **Permutation invariant**
  Chicago_Bulls, American are unordered neighbours

LAN (Logic Attention Network) [Wang et al., AAAI'19]

$$e_i = \sum_{(r,j)\in\mathcal{N}(i)} (\alpha_{j|i,q})\, T_r(e_j)$$

$$\downarrow$$

$$\alpha^{\text{Logic}}_{j|i,q} + \alpha^{\text{NN}}_{j|i,q}$$

play_for => live_in   Chicago_bulls => Chicago

LAN (Logic Attention Network) [Wang et al., AAAI'19]

$$e_i = \sum_{(r,j)\in\mathcal{N}(i)} (\alpha_{j|i,q})\; T_r(e_j)$$

$$\alpha^{\text{Logic}}_{j|i,q} + \alpha^{\text{NN}}_{j|i,q}$$

play_for => live_in

Chicago_bulls => Chicago

Logic Rules

$$P(r_1 \implies r_2) = \frac{\#\text{co-occurrences of } \; r_1, r_2}{\#\text{occurrences of } \; r_1}$$

$$\alpha^{\text{Logic}}_{j|i,q} = \frac{P(r \implies q)}{\max\left\{P(s \implies r) : \; s \in \mathcal{N}(i)\right\}}$$

LAN (Logic Attention Network) [Wang et al., AAAI'19]

$$e_i = \sum_{(r,j)\in\mathcal{N}(i)} (\alpha_{j|i,q})\, T_r(e_j)$$

$$\alpha^{\text{Logic}}_{j|i,q} + \alpha^{\text{NN}}_{j|i,q}$$

play_for => live_in

Chicago_bulls => Chicago

Logic Rules

$$P(r_1 \implies r_2) = \frac{\#\text{co-occurrences of } r_1, r_2}{\#\text{occurrences of } r_1}$$

$$\alpha^{\text{Logic}}_{j|i,q} = \frac{P(r \implies q)}{\max\left\{P(s \implies r) : s \in \mathcal{N}(i)\right\}}$$

Softmax Attention

$$\alpha^{\text{NN}}_{j|i,q} = \text{softmax}\Big(u^T \cdot \tanh\big(W \cdot [z_q; T_r(e_j)]\big)\Big)$$

Experiments [Wang et al., AAAI'19]

Accuracy of Triplet classification on WN11

MEAN	87.3
LSTM	87.0
LAN	**88.8**

MRR of Link Prediction on FB15k

MEAN	0.31
LSTM	0.25
LAN	**0.39**

Experiments [Wang et al., AAAI'19]

Accuracy of Triplet classification on WN11

MEAN	87.3
LSTM	87.0
LAN	**88.8**

Ablation (MRR)

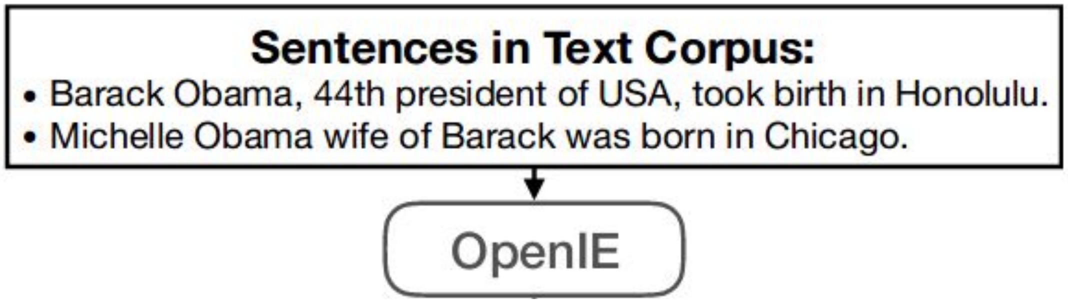
Global-Attention	0.33
Query-Attention	0.35
Logic Rules Only	0.37
LAN	**0.39**

MRR of Link Prediction on FB15k

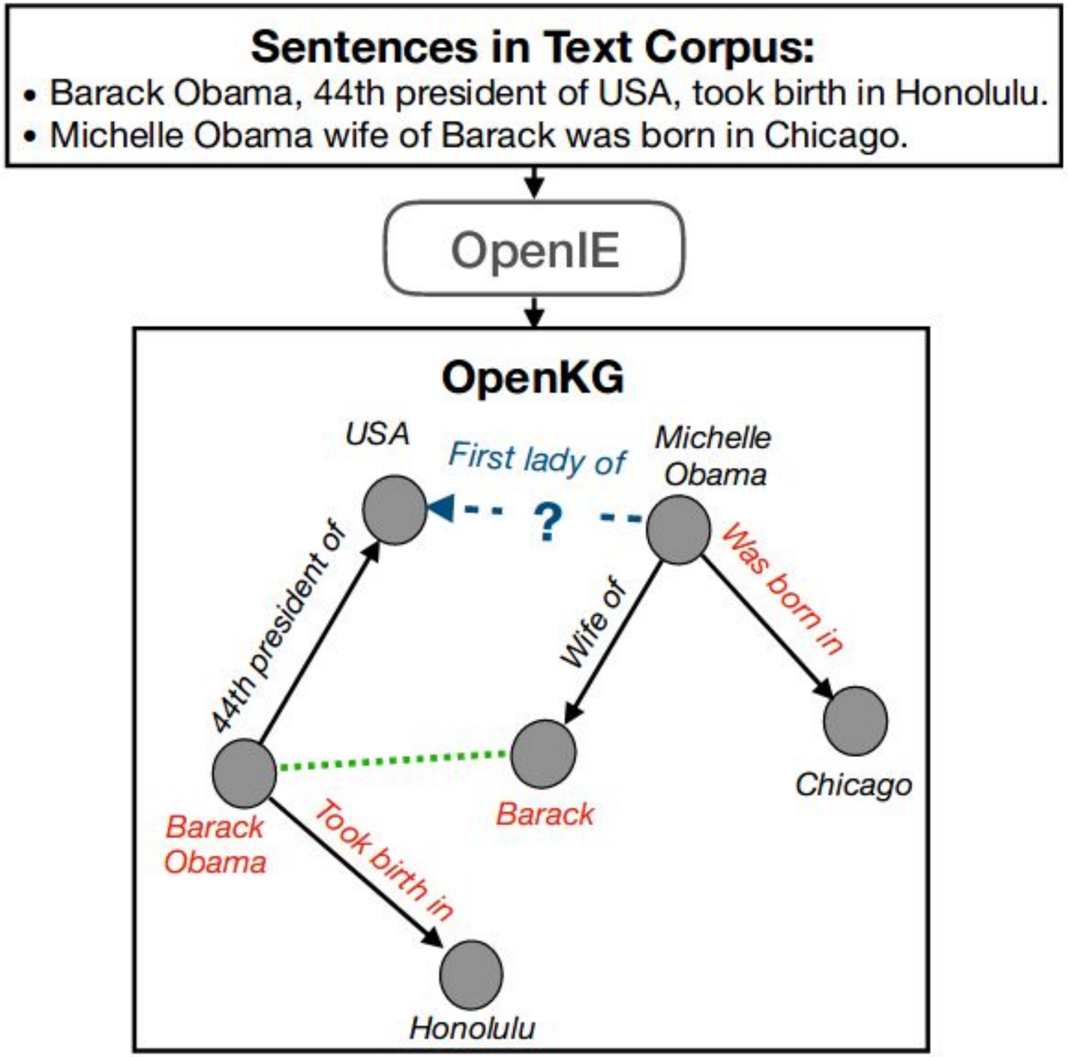
MEAN	0.31
LSTM	0.25
LAN	**0.39**

Logic rules, attention improve inductive learning

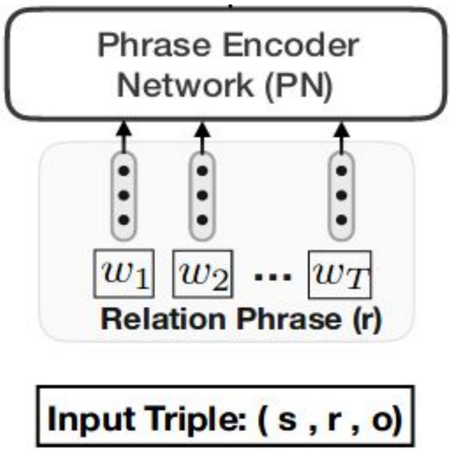
Open KG Embedding [Gupta et al., EMNLP'19]



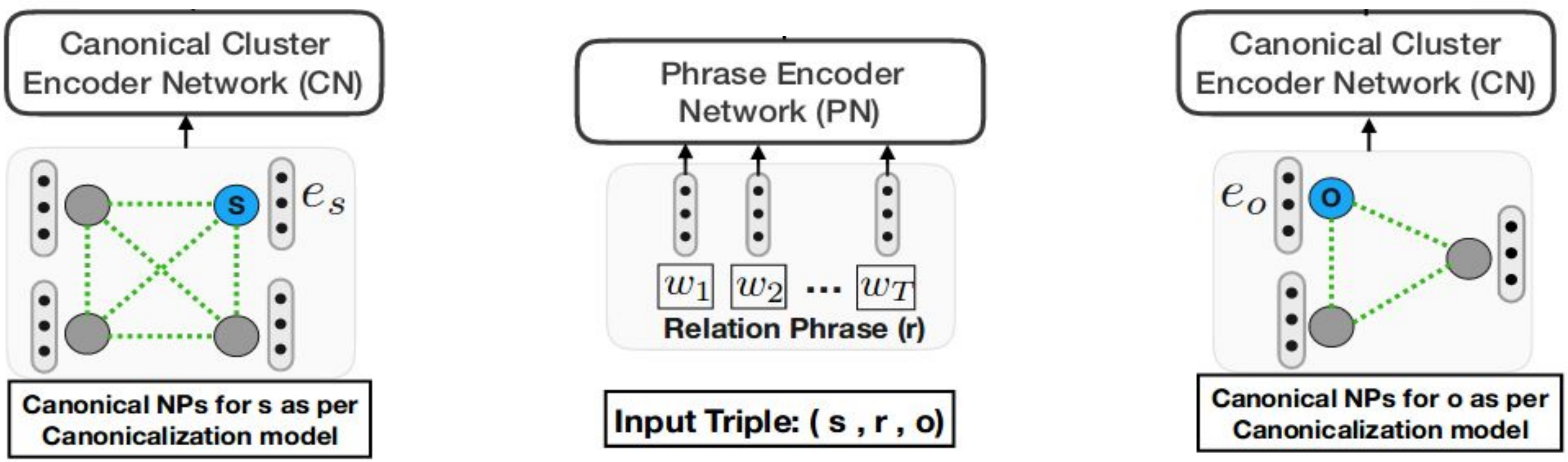
Open KG Embedding [Gupta et al., EMNLP'19]



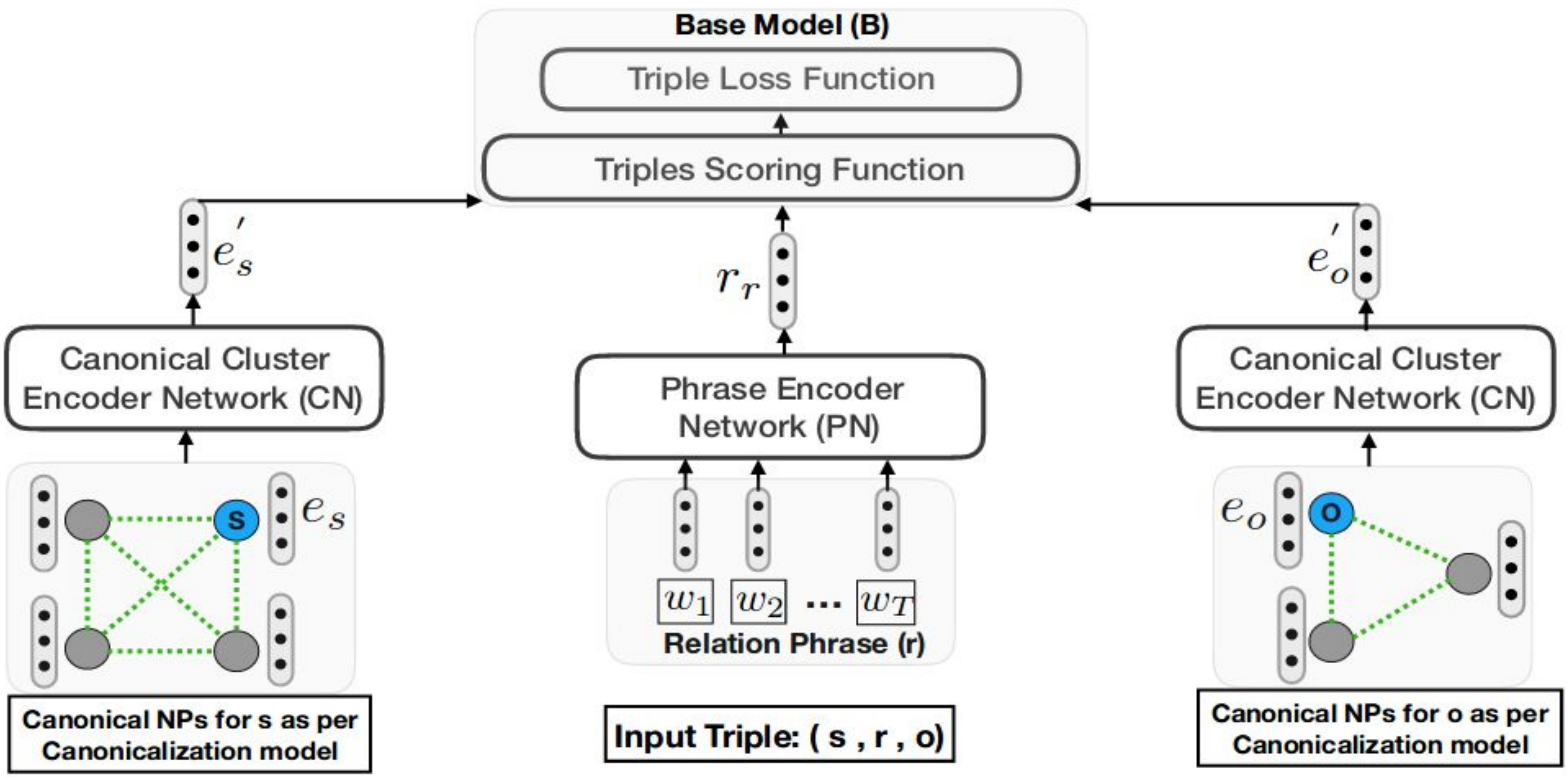
Canonicalised Infused-Representation [Gupta et al., EMNLP'19]



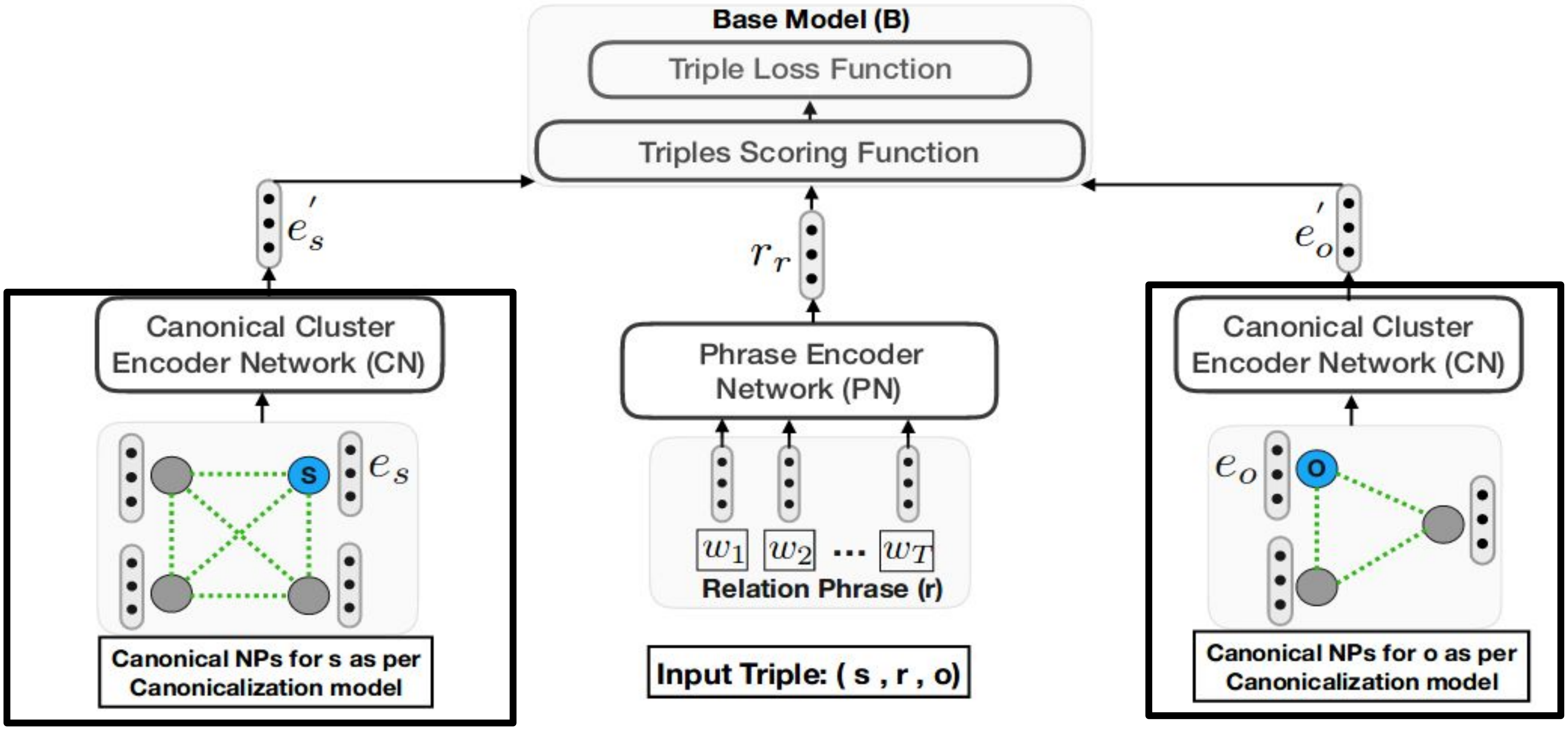
Canonicalised Infused-Representation [Gupta et al., EMNLP'19]



Canonicalised Infused-Representation [Gupta et al., EMNLP'19]

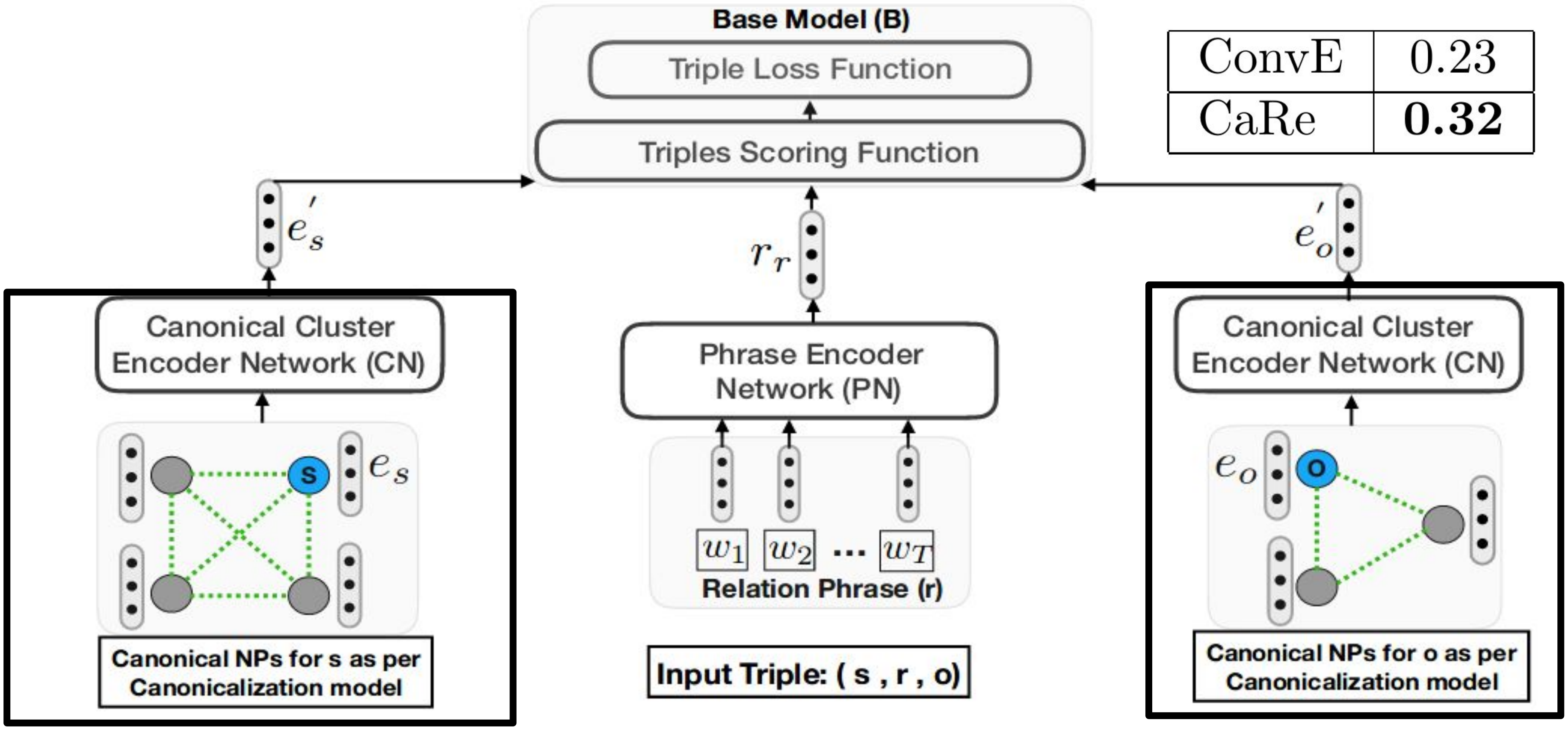


Canonicalised Infused-Representation [Gupta et al., EMNLP'19]



Canonicalisation + graph structure lead to new SOTA

Canonicalised Infused-Representation [Gupta et al., EMNLP'19]

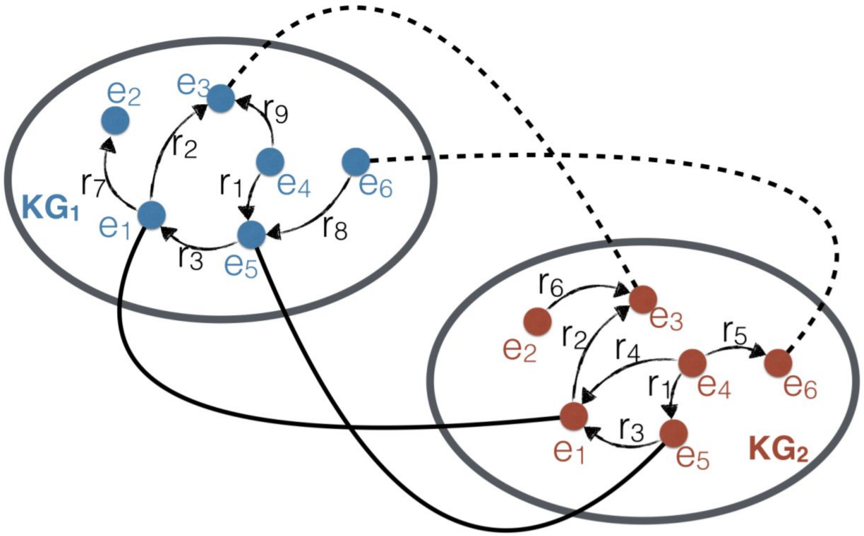


MRR on Reverb45k

ConvE	0.23
CaRe	**0.32**

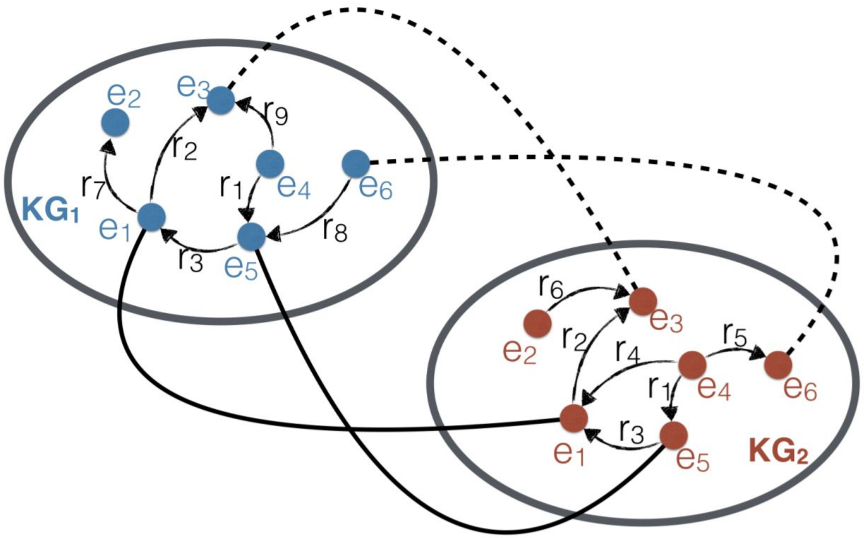
Canonicalisation + graph structure lead to new SOTA

KG Alignment [Cao et al., ACL'19]



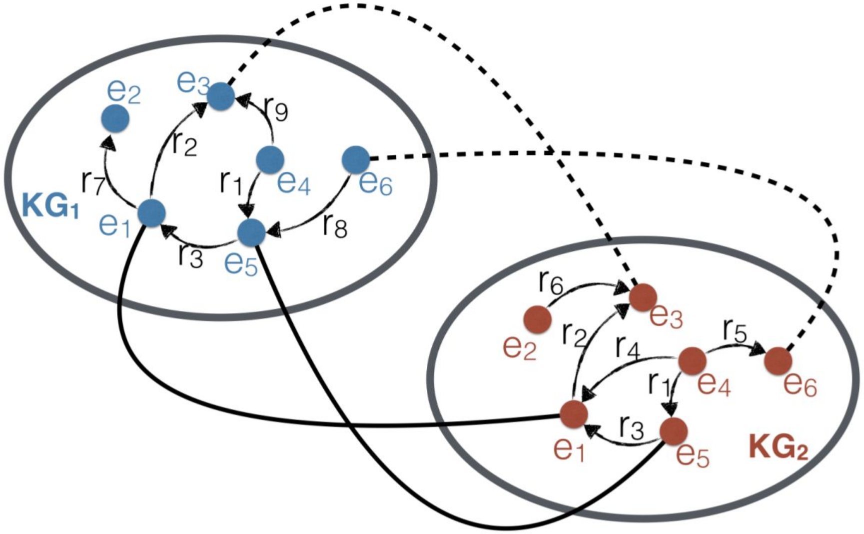
- Freebase, NELL, YAGO - priors for NLP

KG Alignment [Cao et al., ACL'19]



- Freebase, NELL, YAGO - priors for NLP
- constructed separately
  - distinct surface forms
  - supplementary in contents

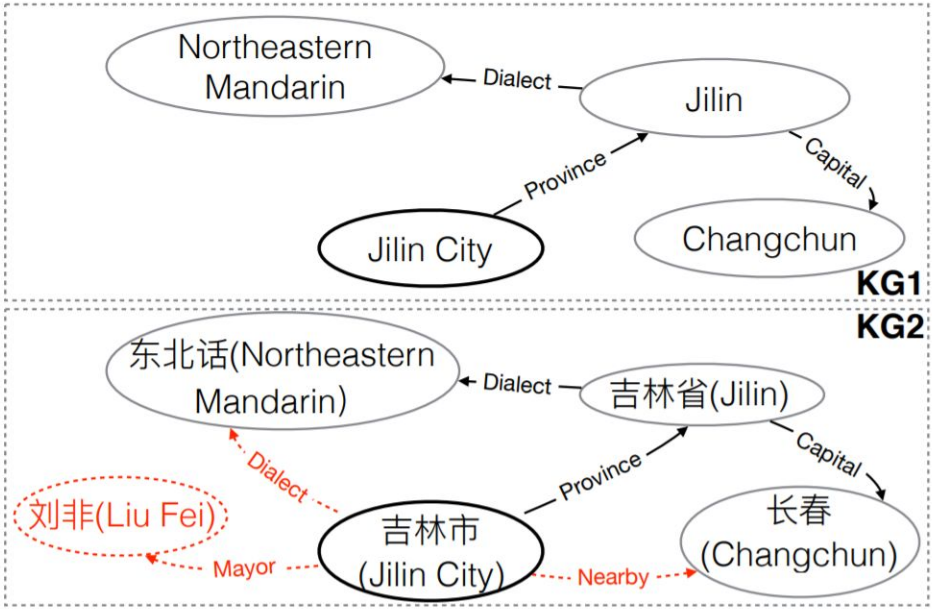
KG Alignment [Cao et al., ACL'19]



- Freebase, NELL, YAGO - priors for NLP
- constructed separately
  - distinct surface forms
  - supplementary in contents
- Wikipedia is multilingual

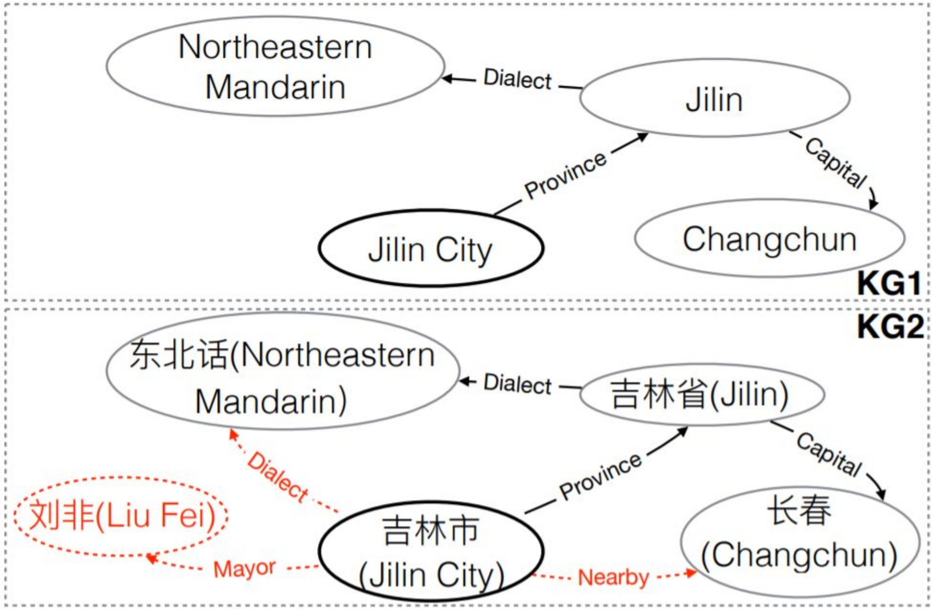
Essential to align entities in KGs to provide a unified KG

Prior Work [Cao et al., ACL'19]



- Symbolic method [Suchanek et al., VLDB'12]
- Embedding methods [Wang et al., EMNLP'18], [Zhu et al., IJCAI'17]
  Assumption: counterparts have similar structures

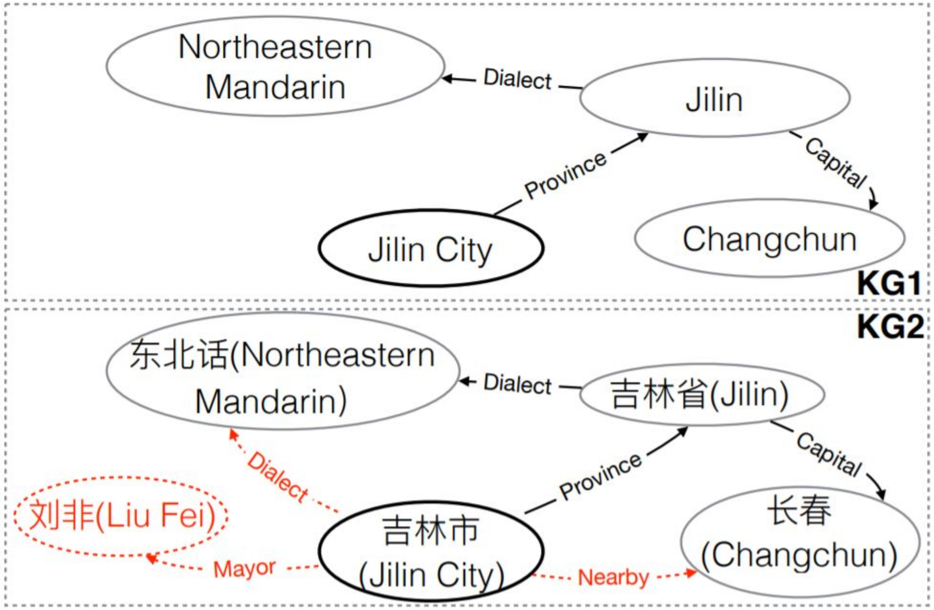
Prior Work [Cao et al., ACL'19]



- Symbolic method [Suchanek et al., VLDB'12]
  X Ineffective for distinct surface forms

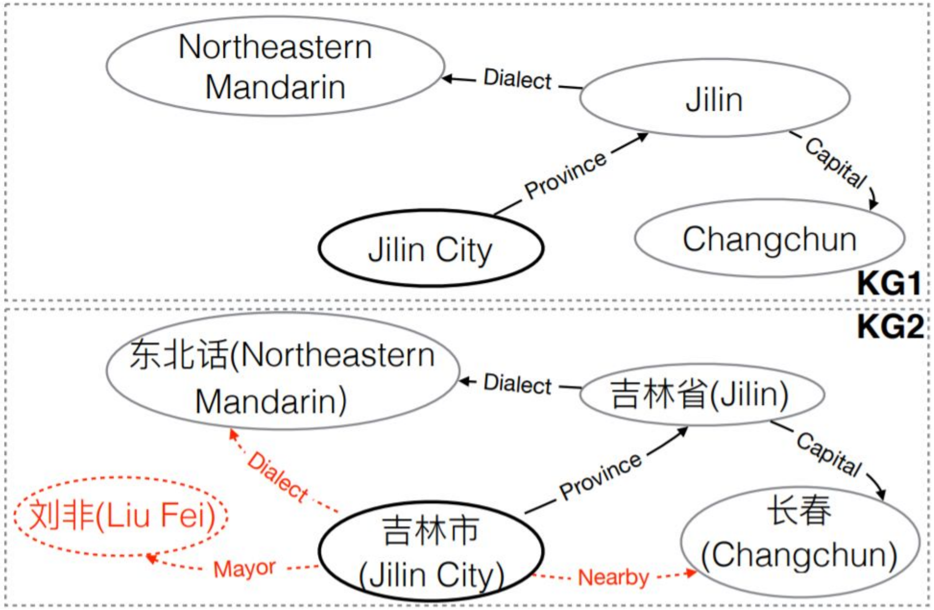
- Embedding methods [Wang et al., EMNLP'18], [Zhu et al., IJCAI'17]
  Assumption: counterparts have similar structures
  X Ineffective for structural heterogeneity

Contributions [Cao et al., ACL'19]



- Robust MuGNN against structural heterogeneity
  - missing relation prediction
  - exclusive entity pruning

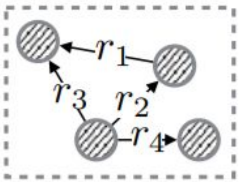
Contributions [Cao et al., ACL'19]

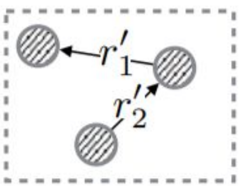


- Robust MuGNN against structural heterogeneity
  - missing relation prediction
  - exclusive entity pruning
- Joint KG inference and alignment
- Experiments

MuGNN [Cao et al., ACL'19]

$G = (E, R, T)$

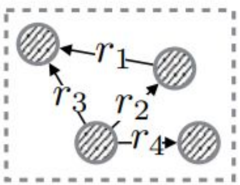


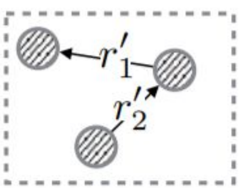


$G' = (E', R', T')$

MuGNN [Cao et al., ACL'19]

$G = (E, R, T)$



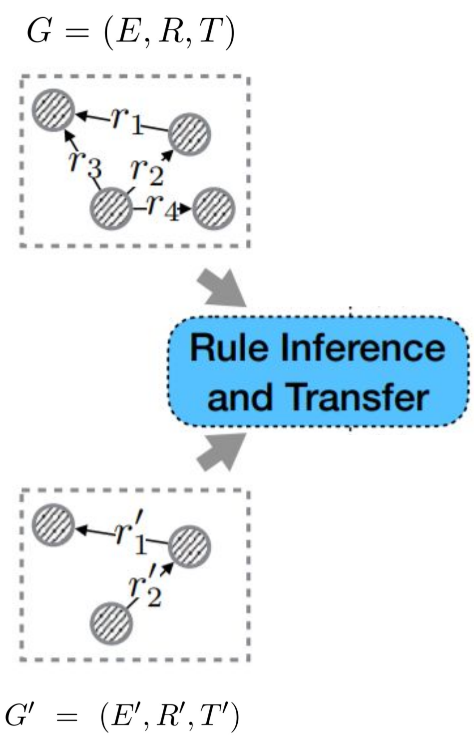


$G' = (E', R', T')$

maximise $|\mathcal{A}_e|$ where $\mathcal{A}_e = \left\{(e, e') \in E \times E' | e \leftrightarrow e'\right\}$

given $\mathcal{A}_e^s$ and $\mathcal{A}_r^s = \left\{(r, r') \in R \times R' | r \leftrightarrow r'\right\}$

MuGNN [Cao et al., ACL'19]

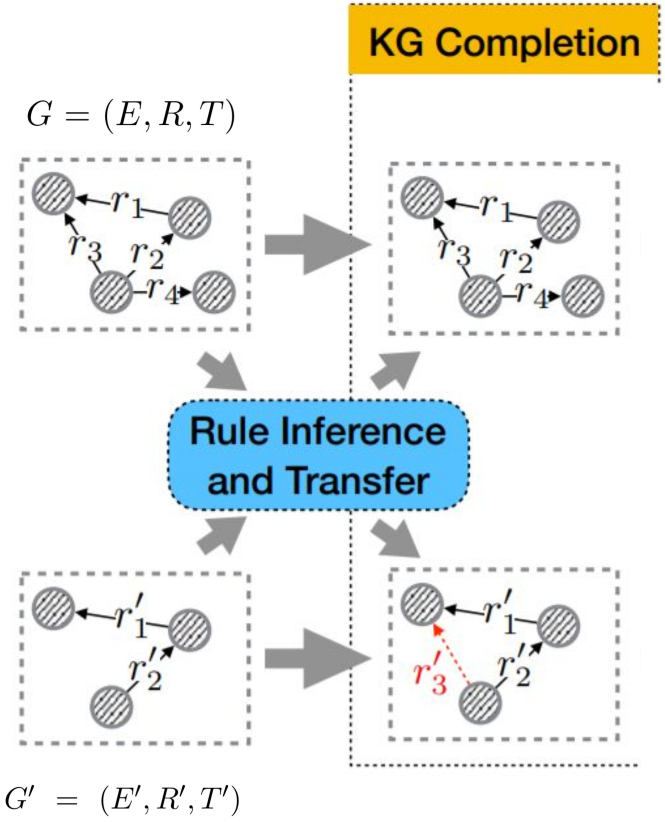


AMIE+ [Gallaraga et al., VLDB'15]

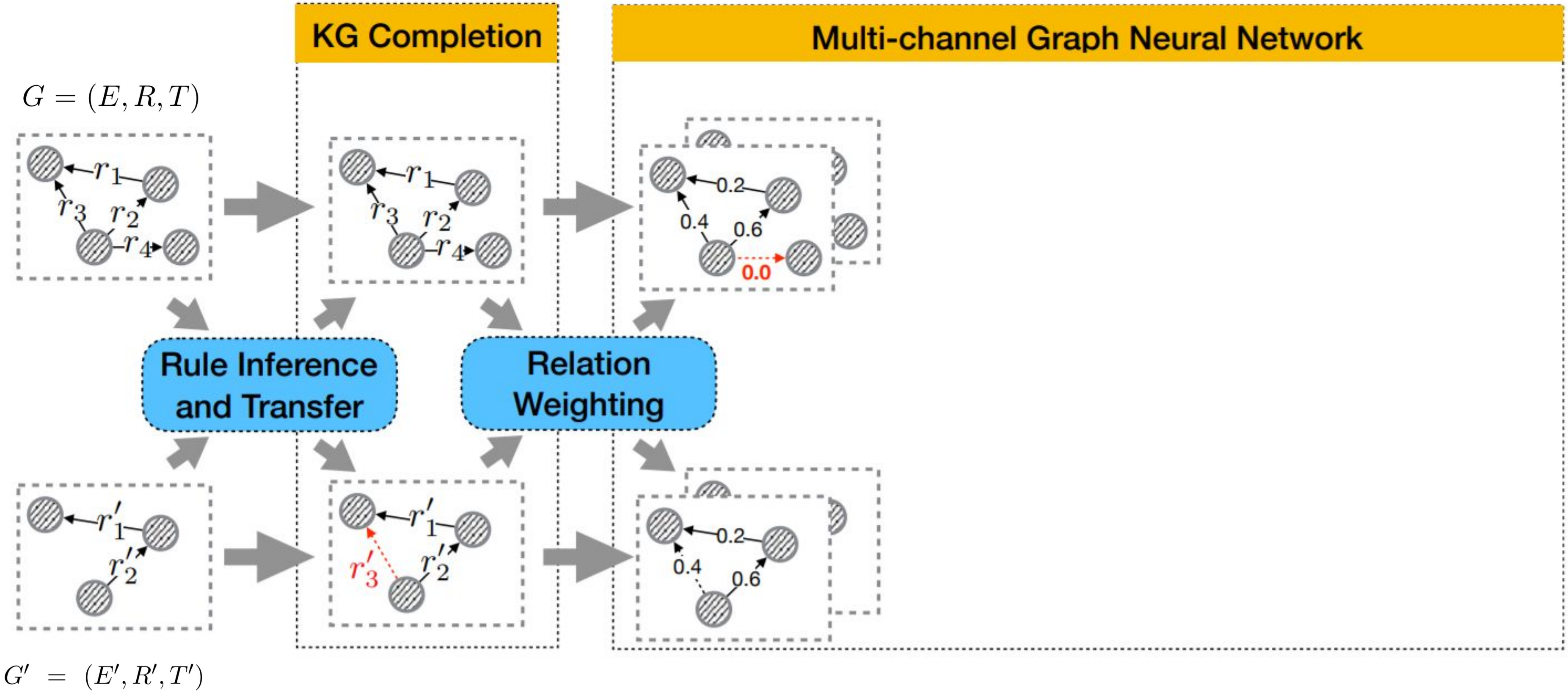
$$bornIn(x, y) \wedge cityOf(y, z) \Rightarrow nationality(x, z)$$

Transfer through $\mathcal{A}_e^s$ and $\mathcal{A}_r^s$

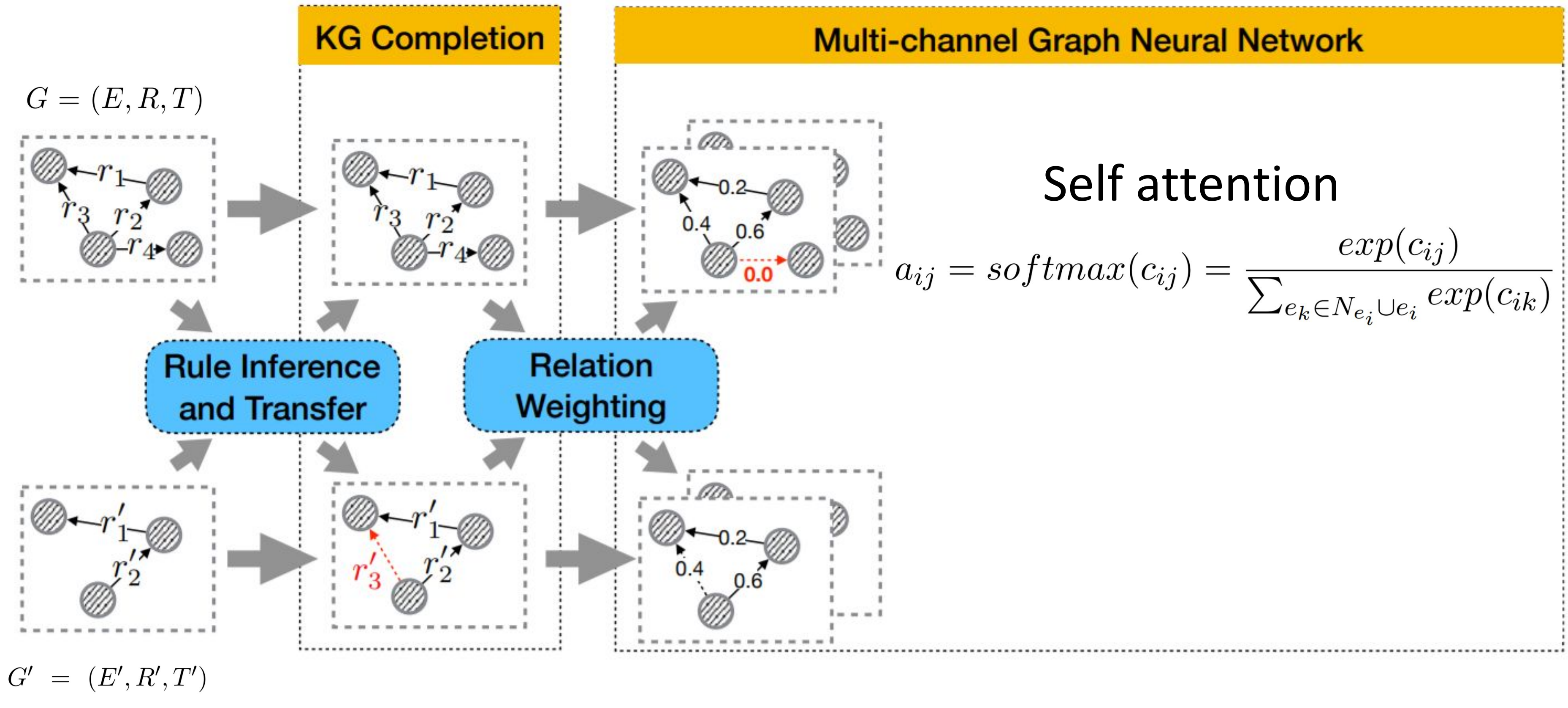
MuGNN [Cao et al., ACL'19]



MuGNN [Cao et al., ACL'19]



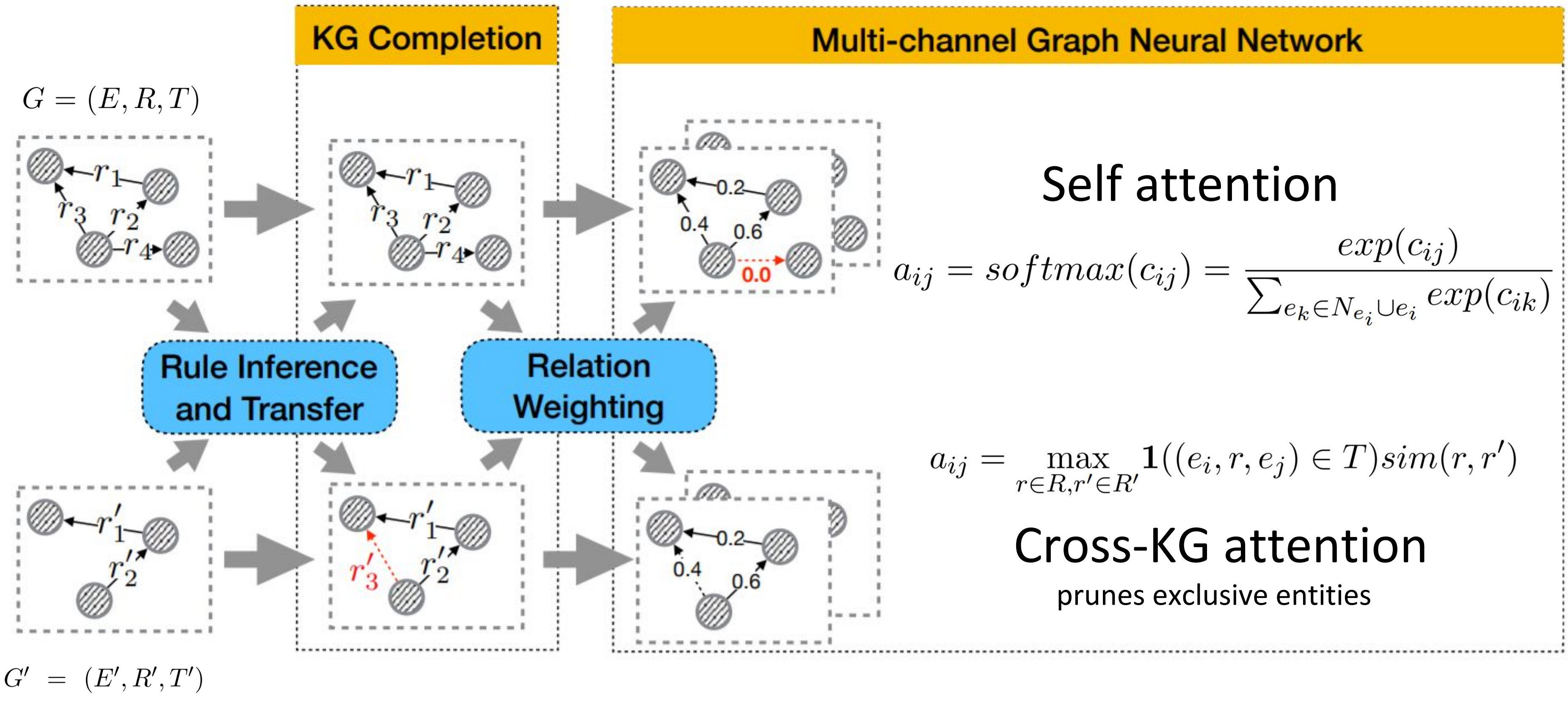
MuGNN [Cao et al., ACL'19]



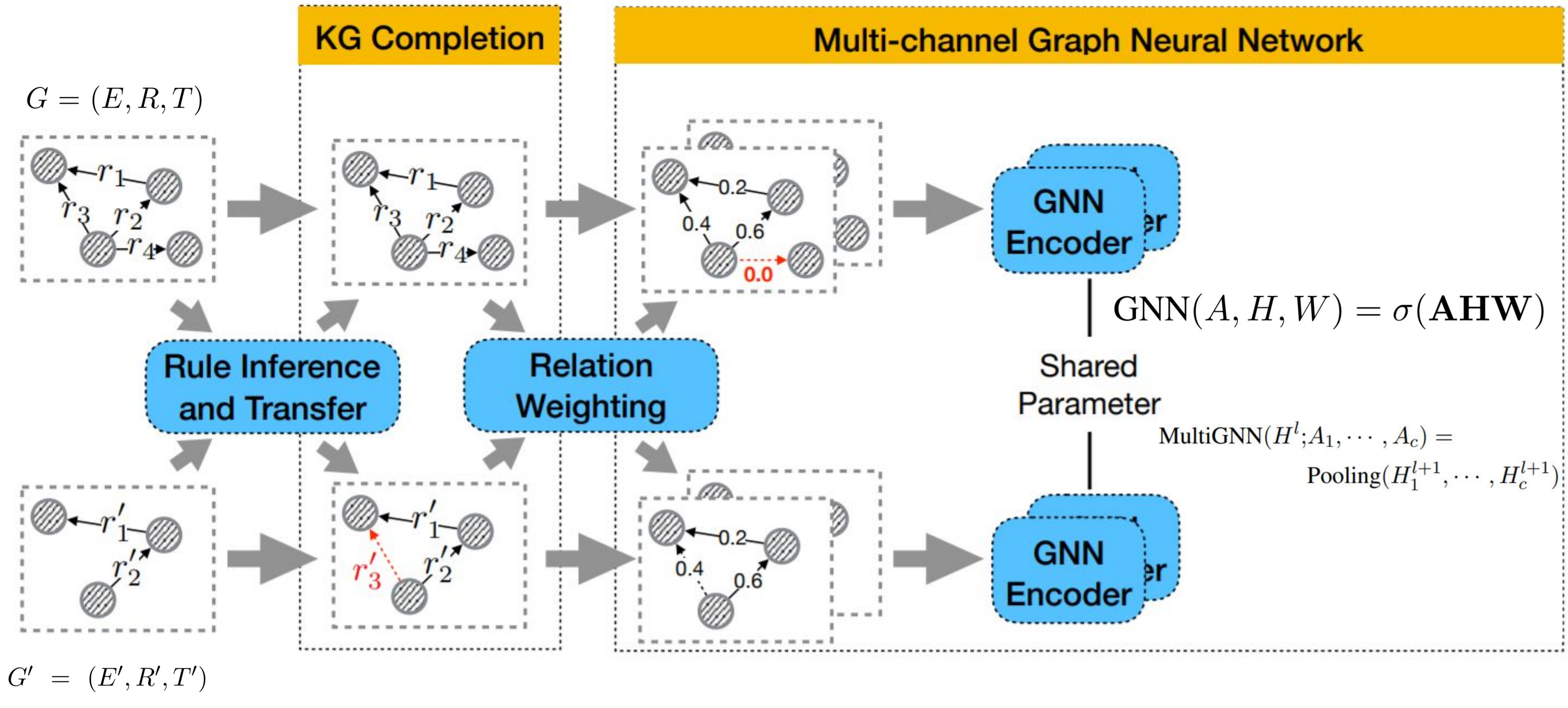
Self attention

$$a_{ij} = softmax(c_{ij}) = \frac{exp(c_{ij})}{\sum_{e_k \in N_{e_i} \cup e_i} exp(c_{ik})}$$

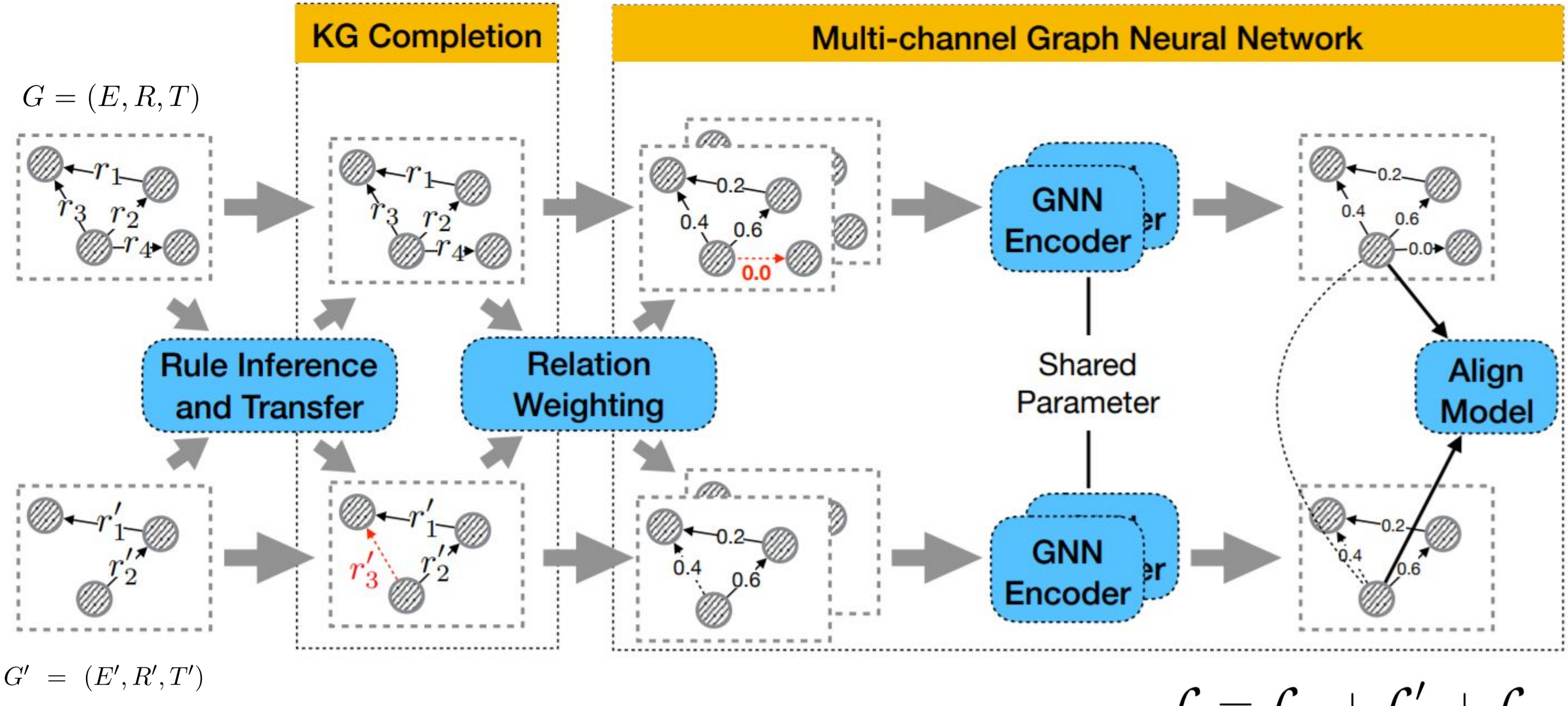
MuGNN [Cao et al., ACL’19]



MuGNN [Cao et al., ACL'19]

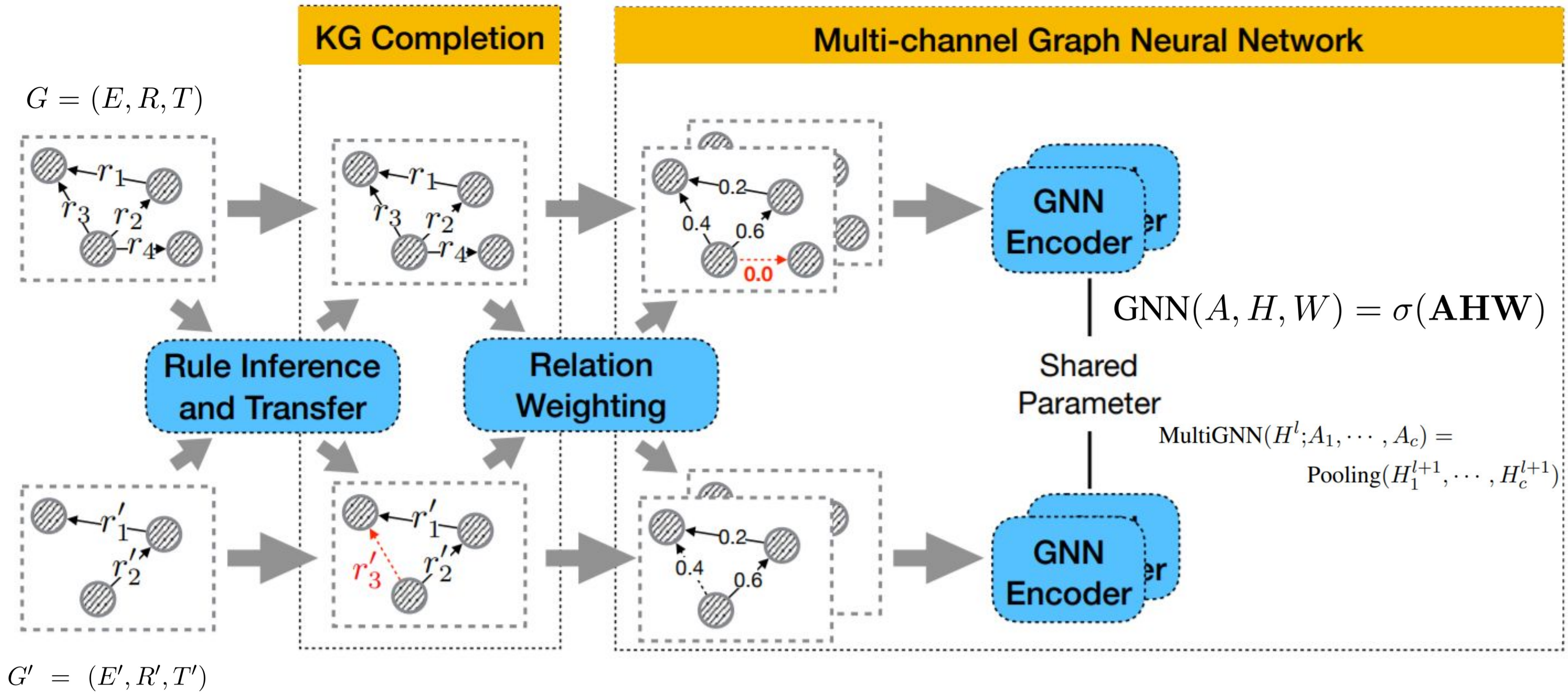


MuGNN [Cao et al., ACL'19]



$$\mathcal{L} = \mathcal{L}_a + \mathcal{L}'_r + \mathcal{L}_r$$

MuGNN [Cao et al., ACL'19]



MRR on DBP-YG

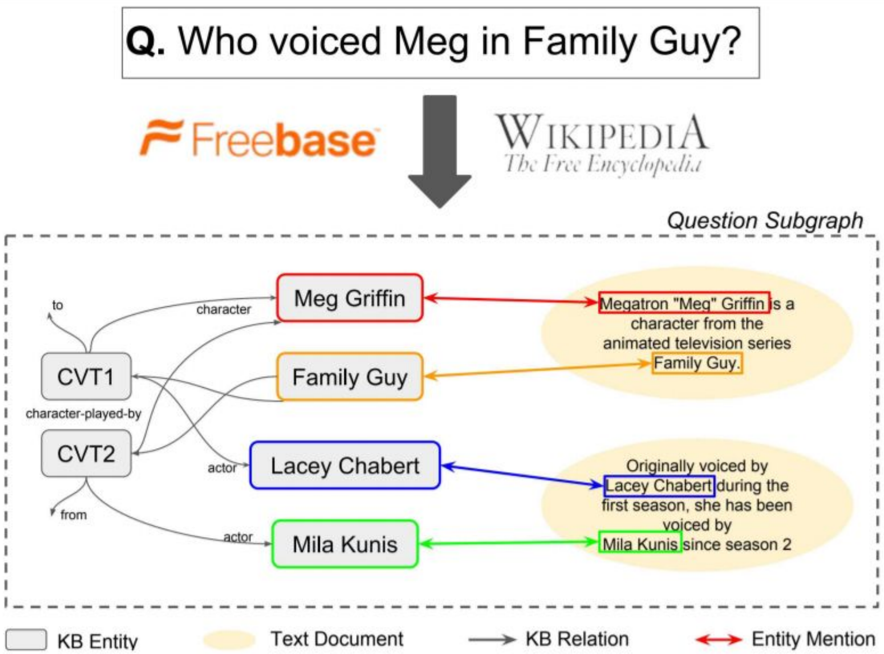
AlignEA	0.71
MuGNN	**0.81**

Joint KG inference + alignment helps

QA from KG+Text [Sun et al., EMNLP'18]

Q. Who voiced Meg in Family Guy?

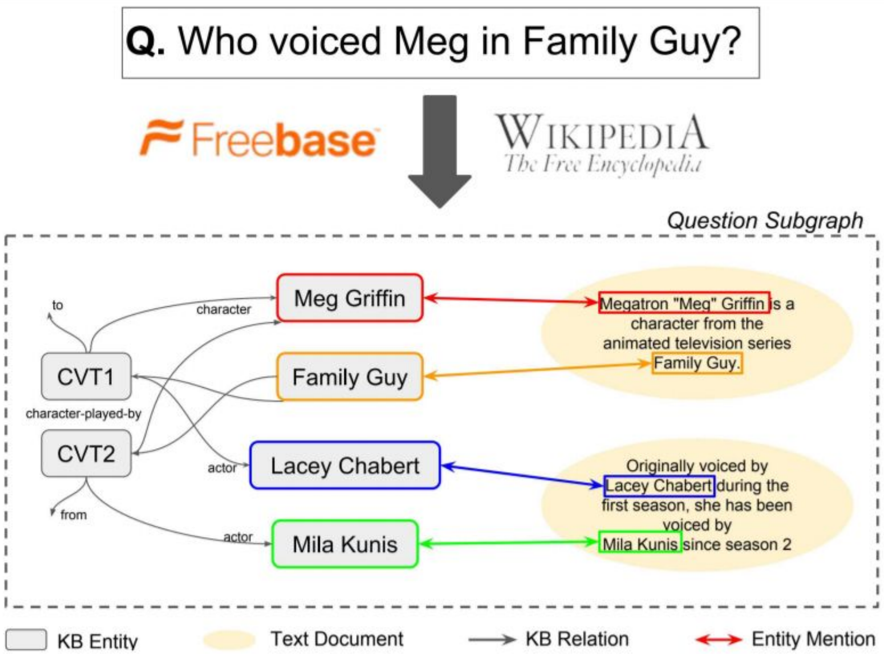
QA from KG+Text [Sun et al., EMNLP'18]

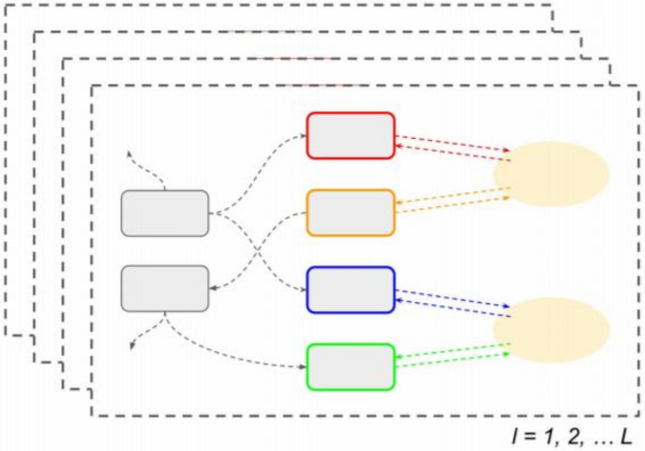


QA from KG+Text [Sun et al., EMNLP'18]

$$\tilde{H}_{d,p}^{(l)} = \mathrm{FFN}\left(H_{d,p}^{(l-1)}, \sum_{v \in L(d,p)} h_v^{(l-1)}\right)$$

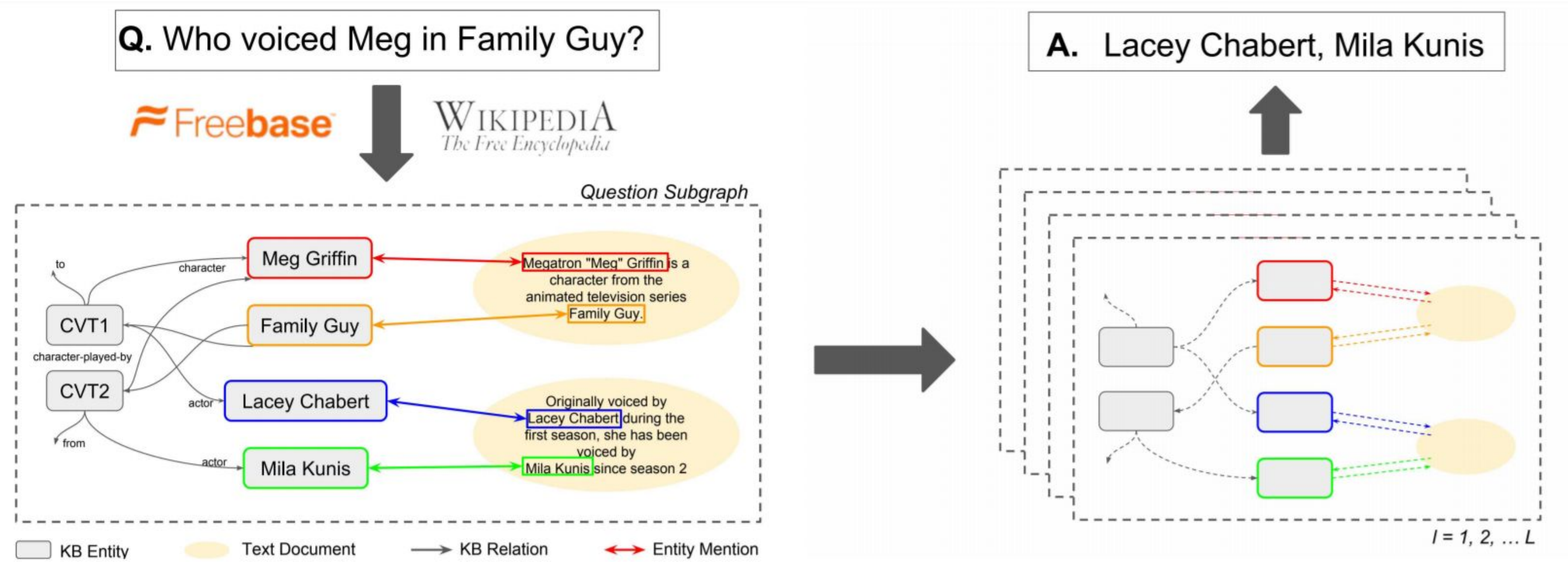
$$H_d^{(l)} = \mathrm{LSTM}(\tilde{H}_d^{(l)})$$





$$h_v^{(l)} = \mathrm{FFN}\left(\begin{bmatrix} h_v^{(l-1)} \\ h_q^{(l-1)} \\ \sum_r \sum_{v' \in N_r(v)} \alpha_r^{v'} \psi_r(h_{v'}^{(l-1)}) \\ \sum_{(d,p) \in M(v)} H_{d,p}^{(l-1)} \end{bmatrix}\right)$$

QA from KG+Text [Sun et al., EMNLP'18]

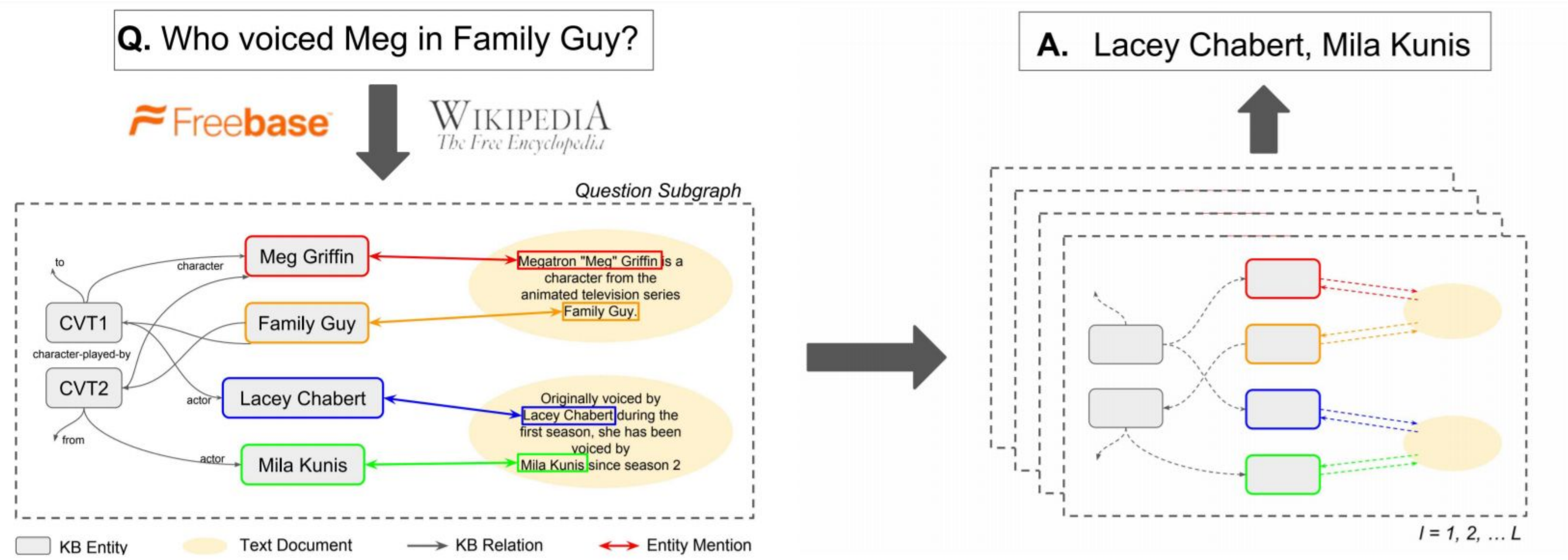


Hits@1 on WikiMovies

KG + Text

KV-MemNN	75.3
GRAFT-NET	**88.4**

QA from KG+Text [Sun et al., EMNLP'18]



Hits@1 on WikiMovies

KG + Text

KV-MemNN	75.3
GRAFT-NET	**88.4**

KG Only

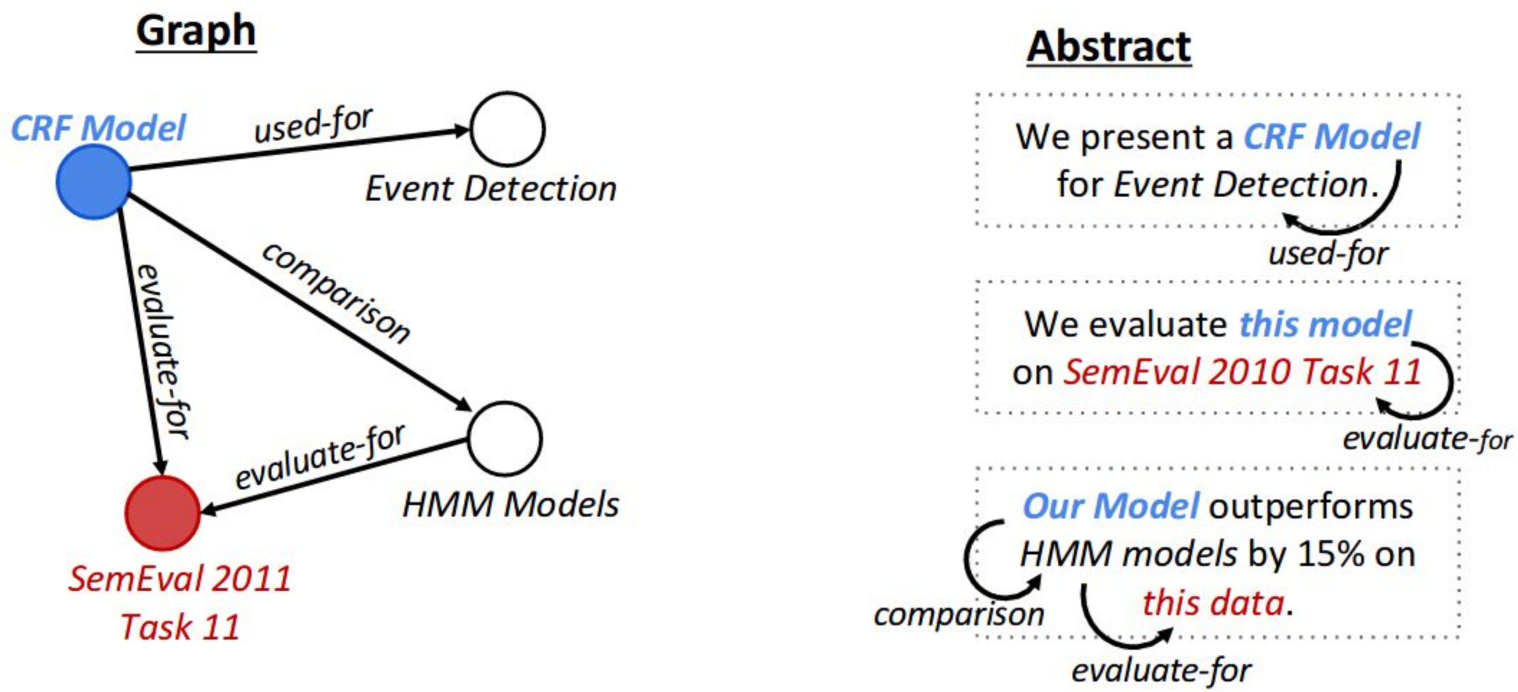
Minerva	97.0
GRAFT-NET	96.8

Text Only

R2-AsV	85.8
GRAFT-NET	86.6

Heterogeneous updates help

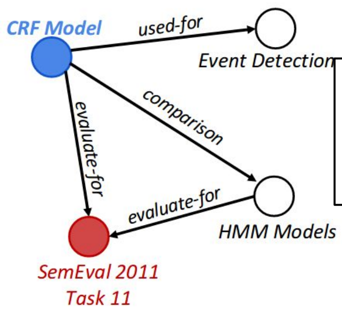
Text Generation from KG [Koncel-Kedziorskiupta et al., NAACL'19]



Input **Output**

GraphWriter [Koncel-Kedziorskiupta et al., NAACL’19]

Graph



Title

Event detection with conditional random fields

Abstract

We present a CRF Model for Event Detection.

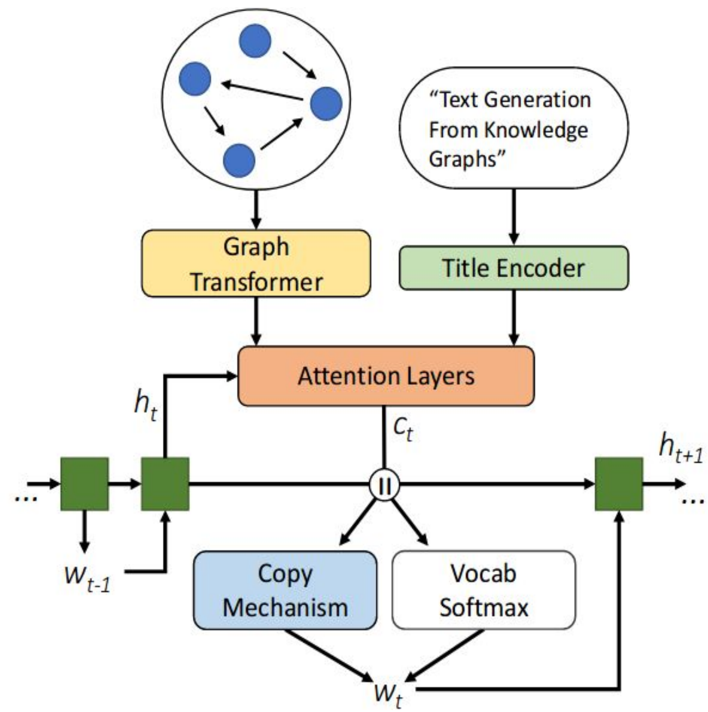
...

We evaluate this model on SemEval 2010 Task 11.

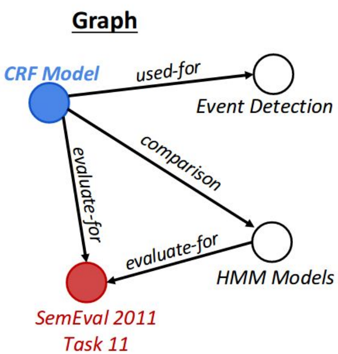
...

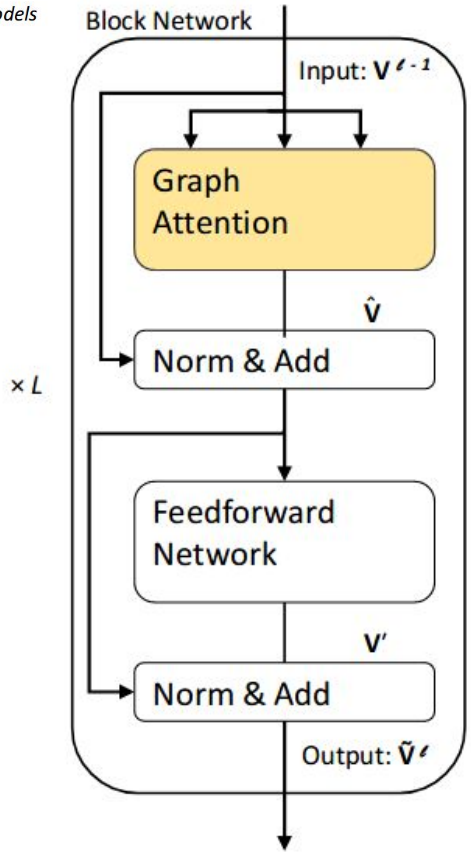
Our model outperforms HMM by 15% on this data.

...



GraphTransformer [Koncel-Kedziorskiupta et al., NAACL'19]





Velickovic et al., ICLR'18

$$\hat{\mathbf{v}}_i = \mathbf{v}_i + \Big\Vert_{n=1}^{N} \sum_{j \in \mathcal{N}_i} \alpha_{ij}^n \mathbf{W}_V^n \mathbf{v}_j$$

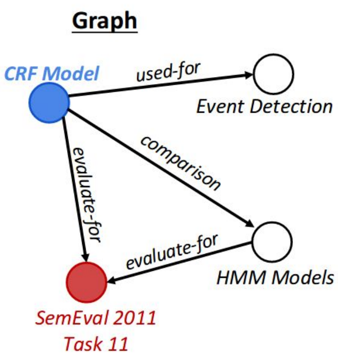
$$\alpha_{ij}^n = a^n(\mathbf{v}_i, \mathbf{v}_j) \qquad a(\mathbf{q}_i, \mathbf{k}_j) = \frac{\exp((\mathbf{W}_K \mathbf{k}_j)^\top \mathbf{W}_Q \mathbf{q}_i)}{\sum_{z \in \mathcal{N}_i} \exp((\mathbf{W}_K \mathbf{k}_z)^\top \mathbf{W}_Q \mathbf{q}_i)}$$

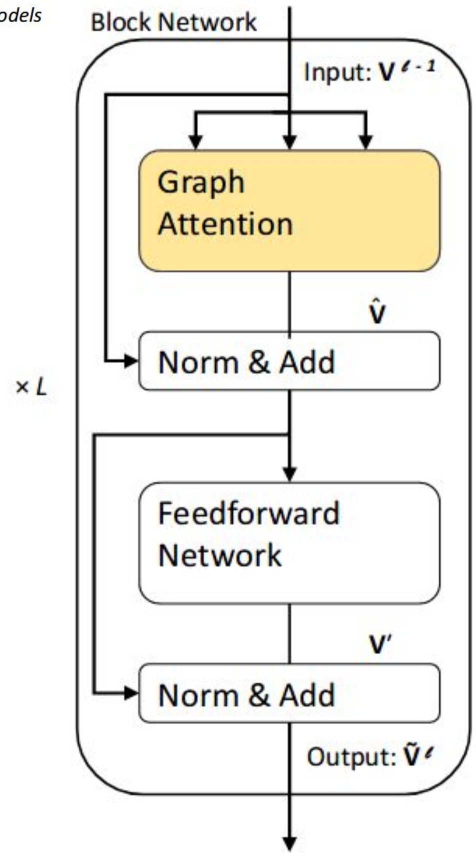
$$\tilde{\mathbf{v}}_i = \mathrm{LayerNorm}(\mathbf{v}_i' + \mathrm{LayerNorm}(\hat{\mathbf{v}}_i))$$

$$\mathbf{v}_i' = \mathrm{FFN}(\mathrm{LayerNorm}(\hat{\mathbf{v}}_i))$$

Vaswani et al., NeurIPS'17

GraphTransformer [Koncel-Kedziorskiupta et al., NAACL'19]





Velickovic et al., ICLR'18

$$\hat{\mathbf{v}}_i = \mathbf{v}_i + \Big\Vert_{n=1}^{N} \sum_{j\in\mathcal{N}_i} \alpha_{ij}^n \mathbf{W}_V^n \mathbf{v}_j$$

$$\alpha_{ij}^n = a^n(\mathbf{v}_i, \mathbf{v}_j) \qquad a(\mathbf{q}_i, \mathbf{k}_j) = \frac{\exp((\mathbf{W}_K\mathbf{k}_j)^\top \mathbf{W}_Q\mathbf{q}_i)}{\sum_{z\in\mathcal{N}_i}\exp((\mathbf{W}_K\mathbf{k}_z)^\top \mathbf{W}_Q\mathbf{q}_i)}$$

$$\tilde{\mathbf{v}}_i = \mathrm{LayerNorm}(\mathbf{v}'_i + \mathrm{LayerNorm}(\hat{\mathbf{v}}_i))$$

$$\mathbf{v}'_i = \mathrm{FFN}(\mathrm{LayerNorm}(\hat{\mathbf{v}}_i))$$

Vaswani et al., NeurIPS'17

BLEU on AGENDA dataset

GAT	12.2 ± 0.44
GraphWriter	$\mathbf{14.3 \pm 1.01}$

GNNs on KG effective for Generation

Summary of GNNs for KGs

- **Takeaways**
  - GNNs for KGs **on top of existing models** help
  - **Open KG, Inductive** approaches are underexplored

Summary of GNNs for KGs

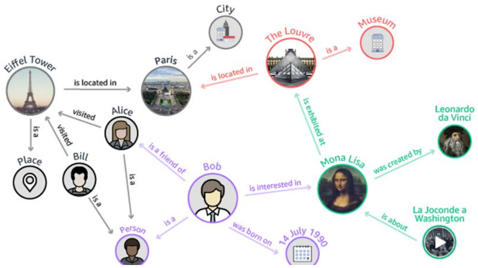
- **Takeaways**
  - GNNs for KGs **on top of existing models** help
  - **Open KG, Inductive** approaches are underexplored

- **Future directions**
  - GNNs for **Web-scale KGs** (importance sampling [Ying et al., KDD'18])
  - **Hyperbolic** GNN + Rotation
  - **Theoretical** Analysis (e.g. Stability bounds [Verma et al., KDD'19])

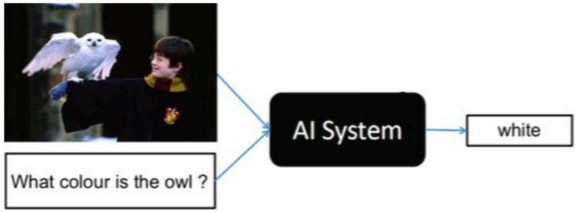
Applications of Graph Neural Nets

- ✓ Semantic Role Labelling, Machine Translation
- ✓ Text Classification, Extraction

- ✓ Knowledge Graphs



- Vision + NLP



?

GNN + Vision + NLP

Visual Question Answering

ReGAT	ICCV'19
VisDialogue	CVPR'19
OotB	NeurIPS 18
GGNN	CVPR'17

GNN +
Vision + NLP

GNN + Vision + NLP

Visual Question Answering

ReGAT	ICCV'19
VisDialogue	CVPR'19
OotB	NeurIPS 18
GGNN	CVPR'17

Label Correlation

DGP	CVPR'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

GNN + Vision + NLP

GNN + Vision + NLP

Visual Question Answering

ReGAT	ICCV'19
VisDialogue	CVPR'19
OotB	NeurIPS 18
GGNN	CVPR'17

GNN + Vision + NLP

Label Correlation

DGP	CVPR'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

Language Grounding

G3	ICCV'19
LCGN	ICCV'19

GNN + Vision + NLP

GNN + Vision + NLP

Visual Question Answering

ReGAT	ICCV’19
VisDialogue	CVPR’19
OotB	NeurIPS 18
GGNN	CVPR’17

Label Correlation

DGP	CVPR’19
KGSRU	IJCAI’19
KERL	IJCAI’19
ML-ZSL	CVPR’18
Zero-Shot	CVPR’18
GSNN	CVPR’17

Expression Comprehension

DGA	ICCV’19
LGRAN	CVPR’19

Language Grounding

G3	ICCV’19
LCGN	ICCV’19

GNN + Vision + NLP

GNN + Vision + NLP

Visual Question Answering

ReGAT	ICCV'19
VisDialogue	CVPR'19
OotB	NeurIPS 18
GGNN	CVPR'17

Label Correlation

DGP	CVPR'19
KGSRU	IJCAI'19
KERL	IJCAI'19
ML-ZSL	CVPR'18
Zero-Shot	CVPR'18
GSNN	CVPR'17

Image Captioning

HIP	ICCV'19
SGAE	CVPR'19
GCN-LSTM	ECCV'18

Expression Comprehension

DGA	ICCV'19
LGRAN	CVPR'19

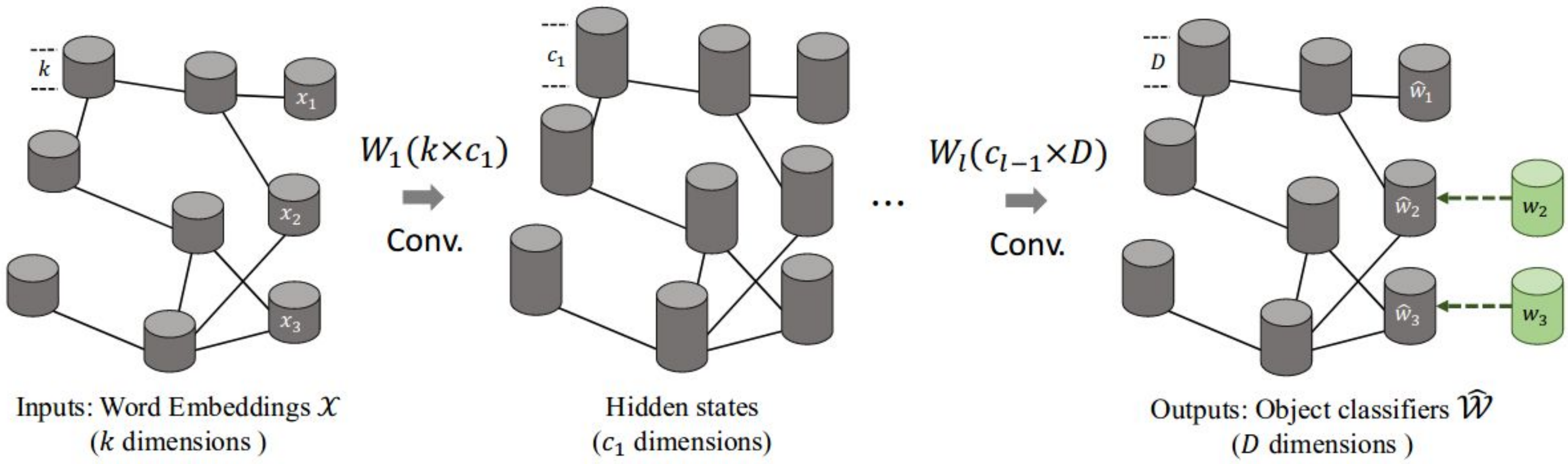
Language Grounding

G3	ICCV'19
LCGN	ICCV'19

Zero-shot Learning [Wang et al., CVPR'18]

Okapi is a "zebra-stripped four-legged animal with a brown torso and a deer-like face"

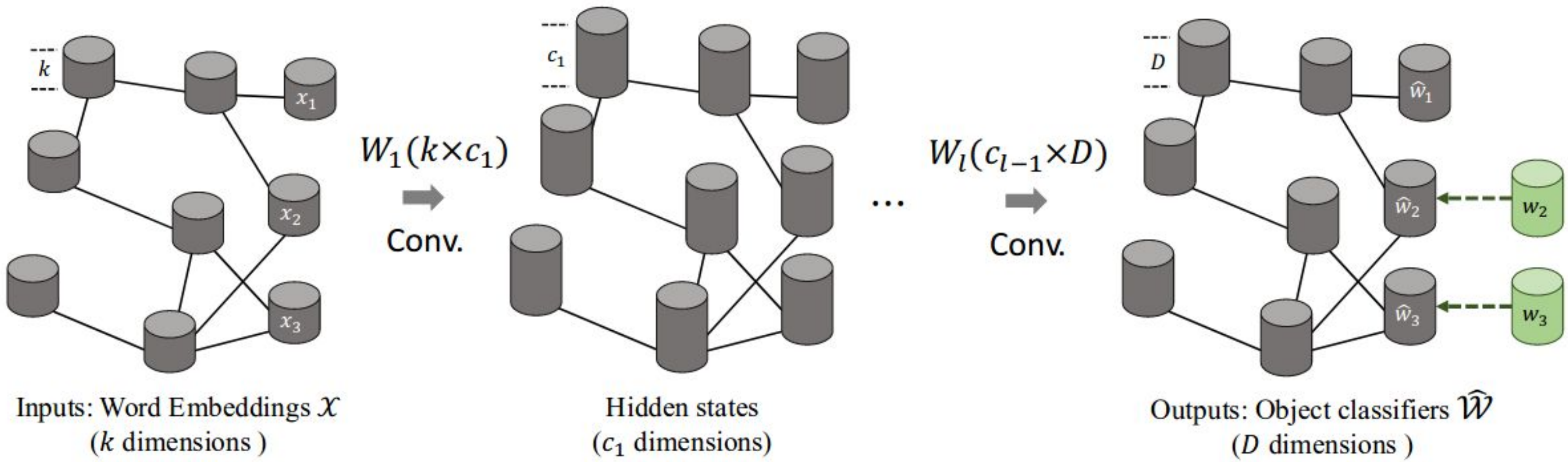
GCN predicts visual classifier [Wang et al., CVPR'18]



$$Z = \hat{A}X'W$$

Okapi is a "zebra-stripped four-legged animal with a brown torso and a deer-like face"

GCN predicts visual classifier [Wang et al., CVPR'18]



$$Z = \hat{A}X'W$$

Okapi is a "zebra-stripped four-legged animal with a brown torso and a deer-like face"

Hits@20 on ImageNet

EXEM	55.2
GCN	**72.0**

GCN can integrate vertex features (text) and the graph structure (KG)

Visual Question Answering [Narasimhan et al., NeurIPS'18]

What is the area used for?

***Relation*: UsedFor**
Field, UsedFor, Grazing
***Answer*: Grazing**

Visual Question Answering [Narasimhan et al., NeurIPS'18]

What is the area used for?

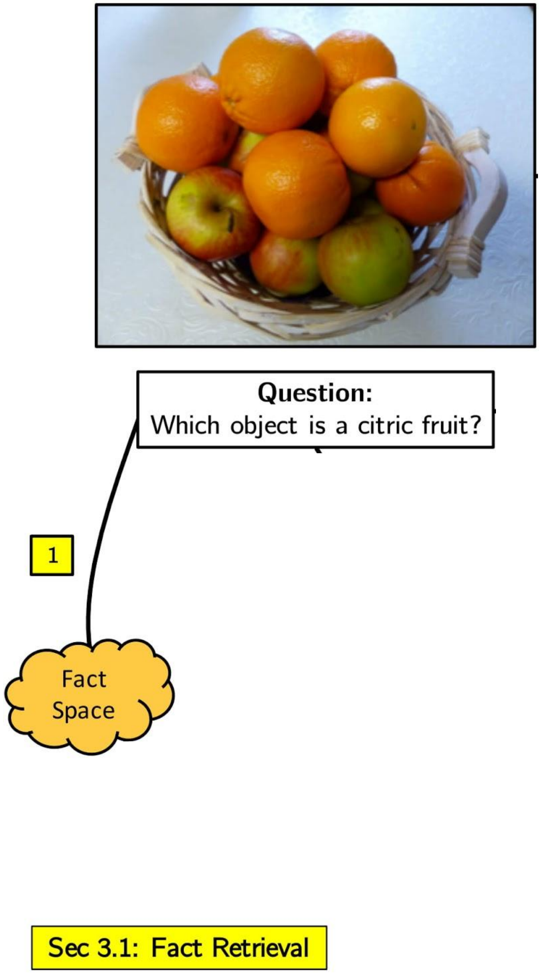
Relation: UsedFor
Field, UsedFor, Grazing
Answer: Grazing

Which looks more like tiger?

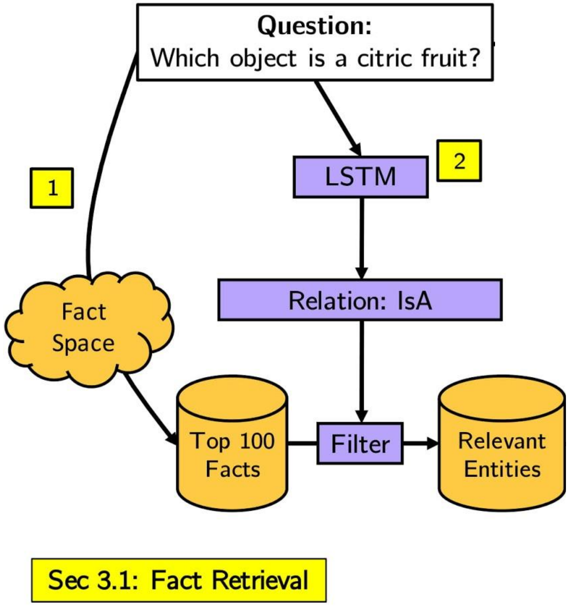
Relation: RelatedTo
Cat, RelatedTo, Tiger
Answer: Cat

GCN for F-VQA [Narasimhan et al., NeurIPS'18]

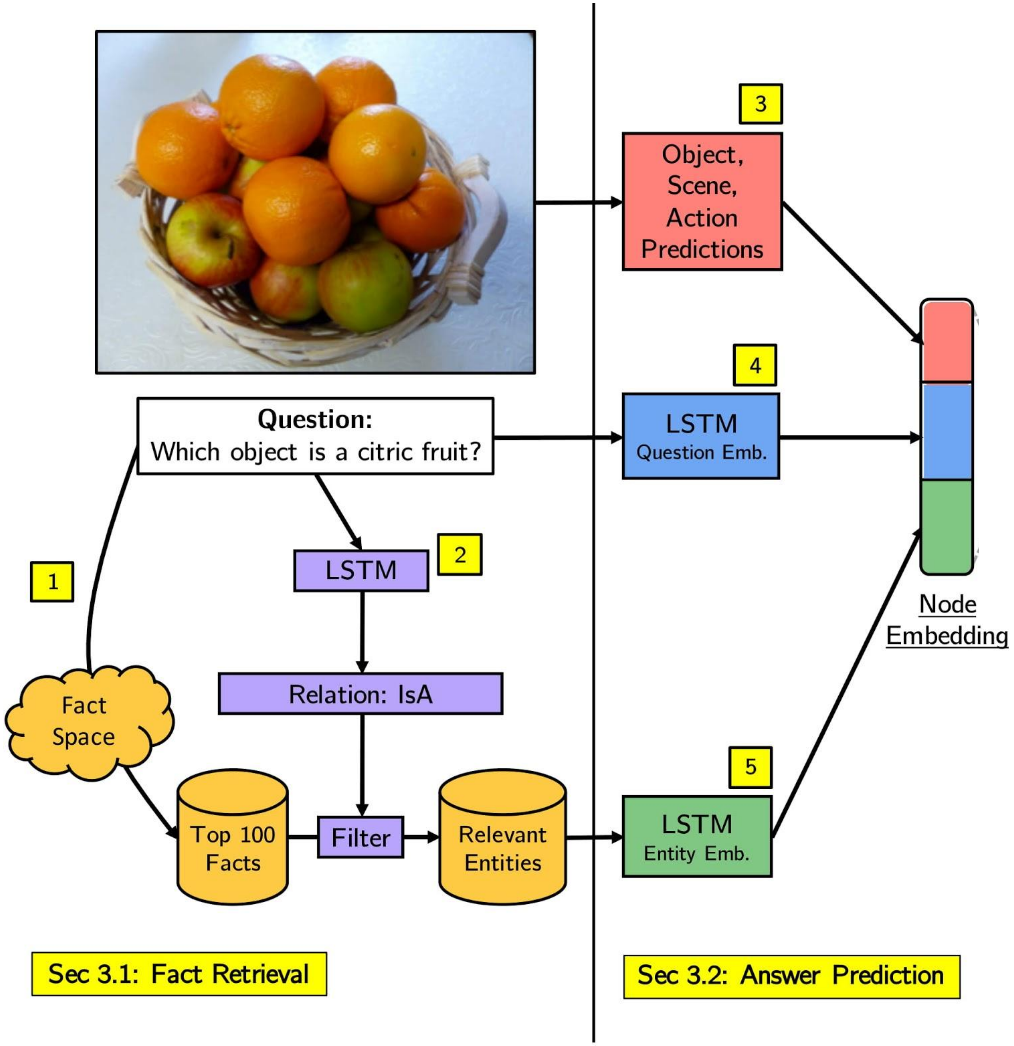
GCN for F-VQA [Narasimhan et al., NeurIPS'18]



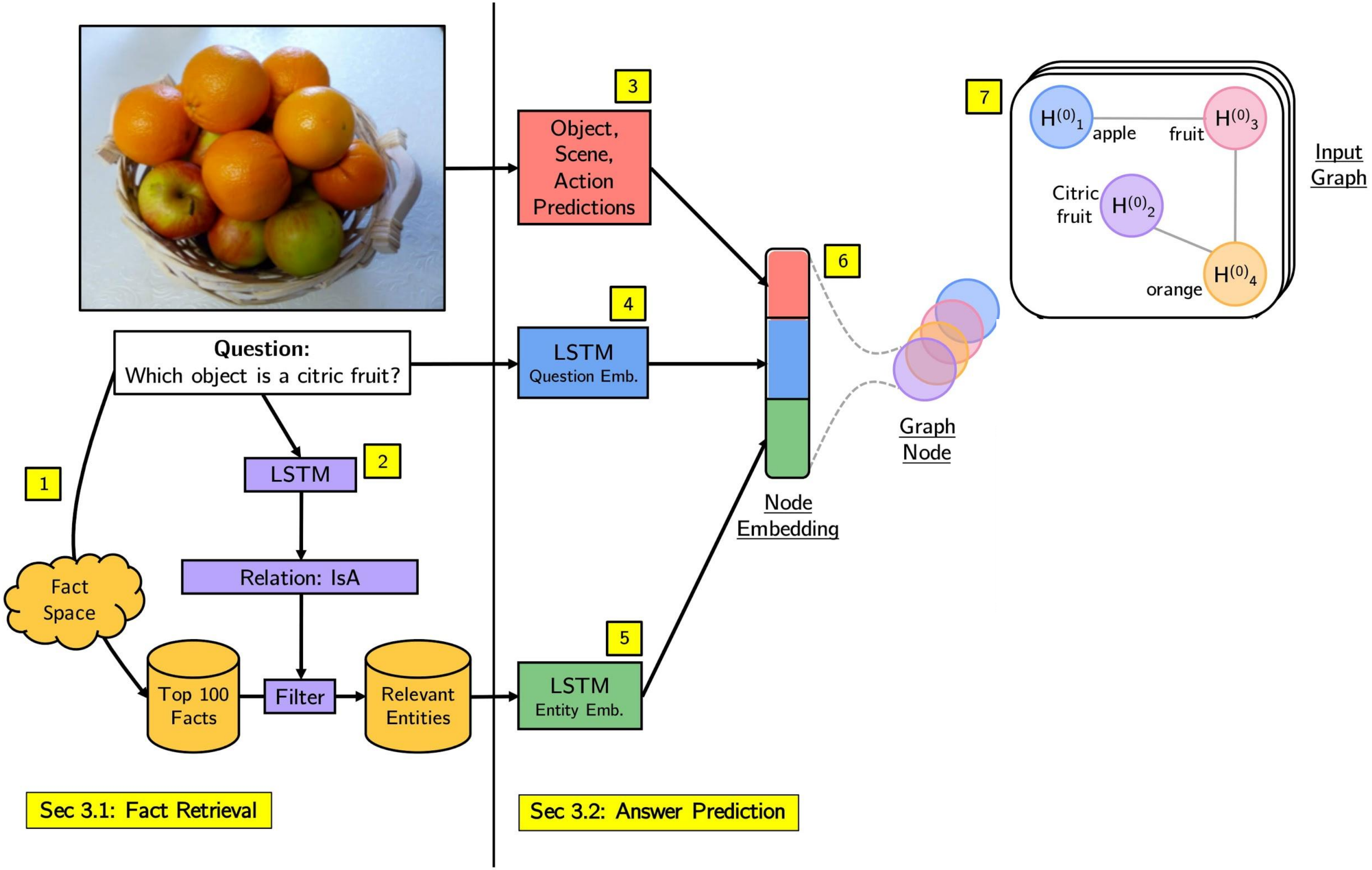
GCN for F-VQA [Narasimhan et al., NeurIPS'18]



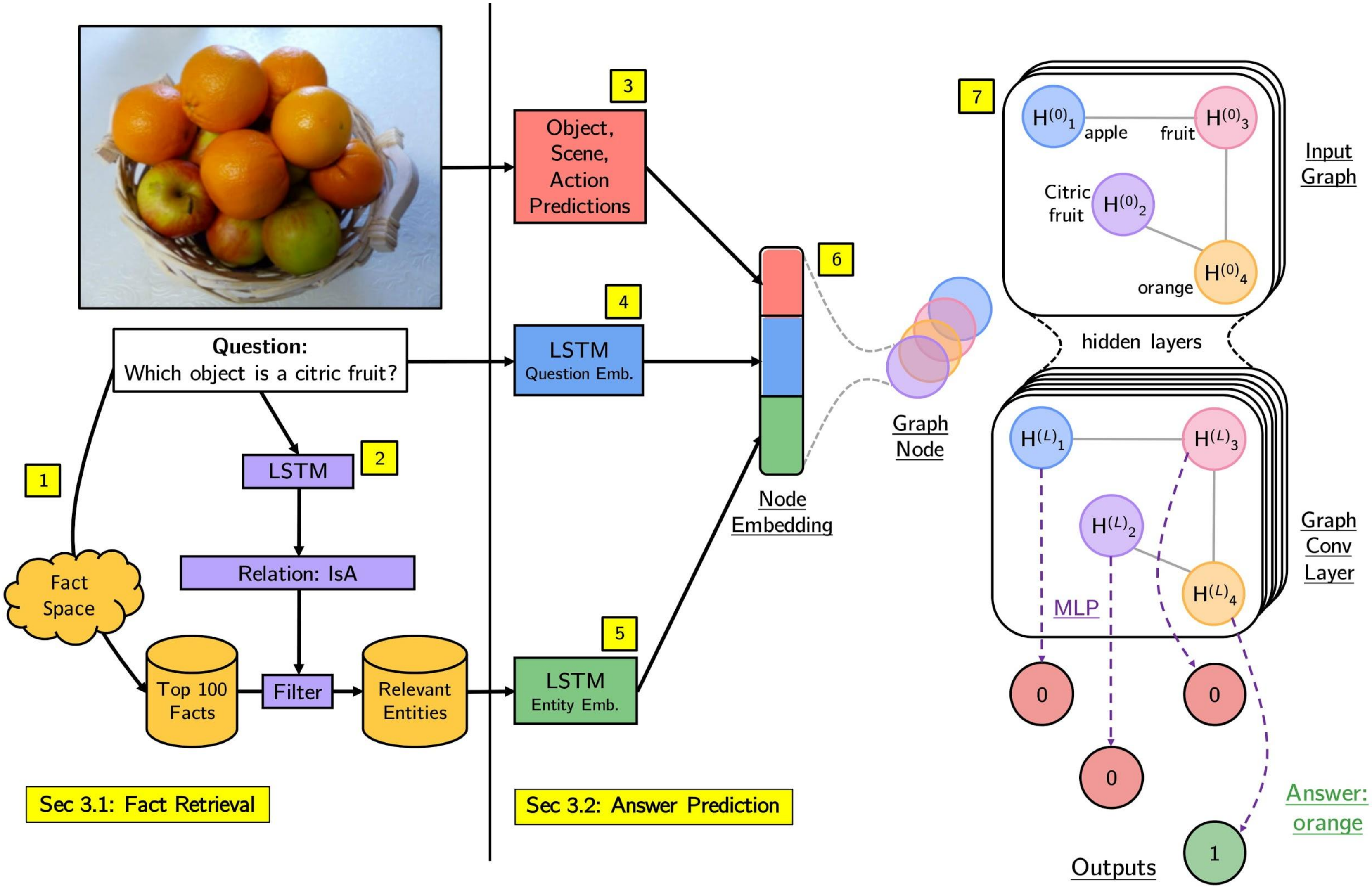
GCN for F-VQA [Narasimhan et al., NeurIPS'18]



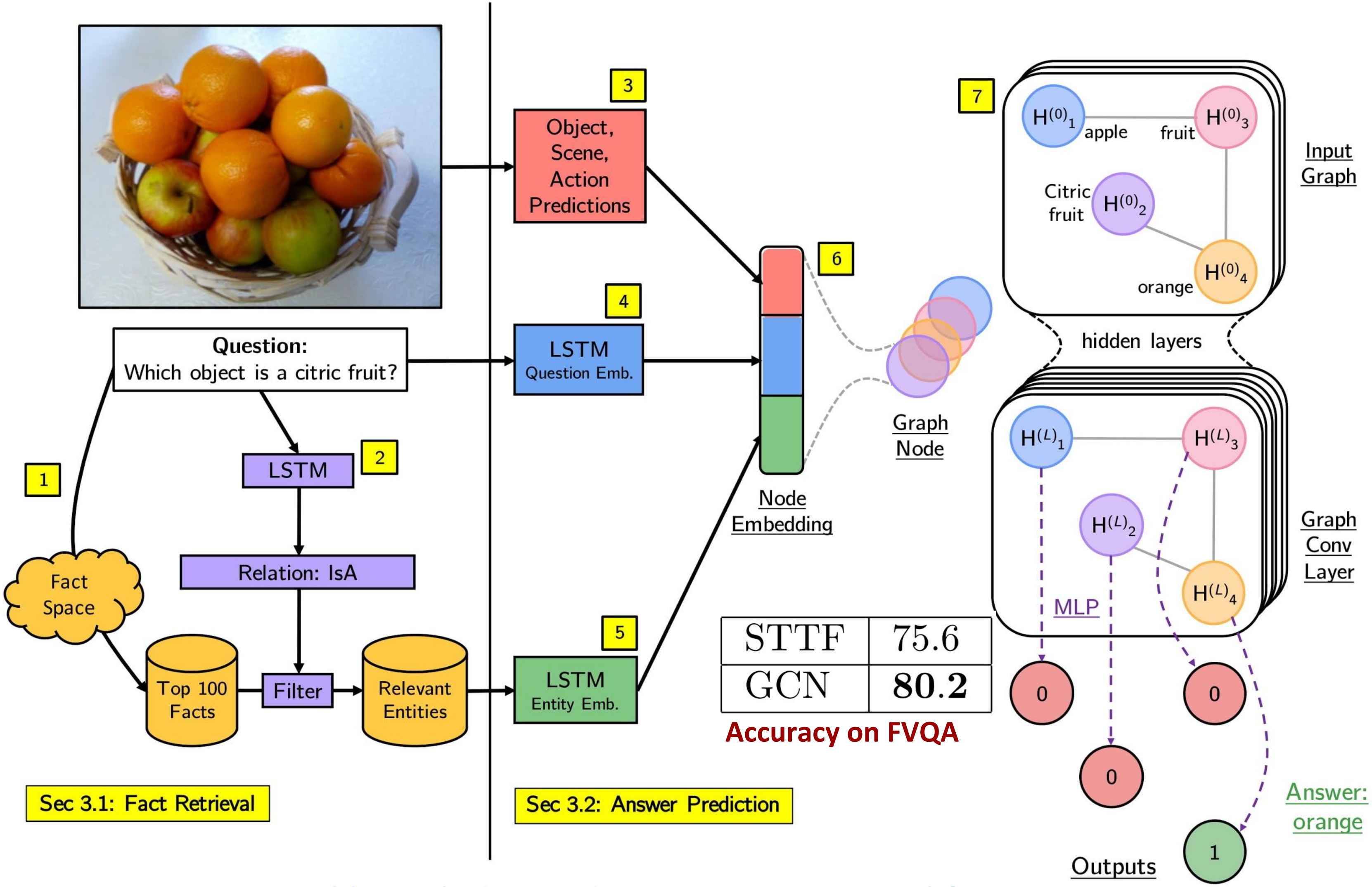
GCN for F-VQA [Narasimhan et al., NeurIPS’18]



GCN for F-VQA [Narasimhan et al., NeurIPS'18]



GCN for F-VQA [Narasimhan et al., NeurIPS'18]



STTF	75.6
GCN	**80.2**

Accuracy on FVQA

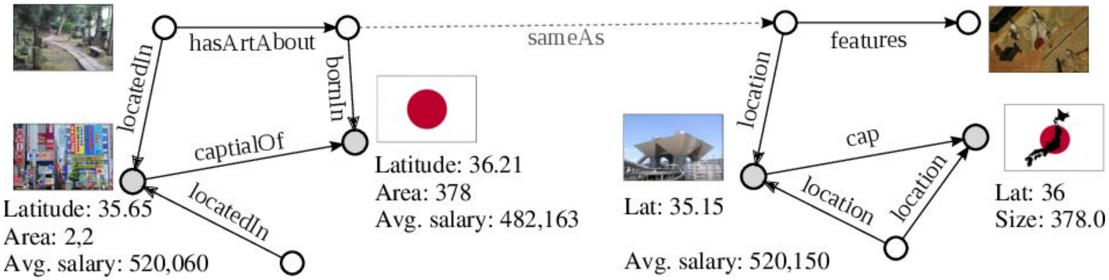
GCN integrates external knowledge and question sequential features

Summary of GNNs for Vision + NLP

- **Takeaways**
  - GNNs exploit **KG + text** for zero-shot visual learning
  - GNNs can **reason about answers** in VQA

Summary of GNNs for Vision + NLP

- **Takeaways**
  - GNNs exploit **KG + text** for zero-shot visual learning
  - GNNs can **reason about answers** in VQA
- **Future direction**
  - Multi-modal KGs
    - Image - text, Image - KG entity mapping
    - KG + Image + Text



Other papers on GNNs for NLP

- **QA and Reading Comprehension**

  [Lin et al., EMNLP'19, Tu et al., ACL'19, Qiu et al., ACL'19, Ding et al., ACL'19,Cao et al., NAACL'19, Cao et al., NAACL'19]

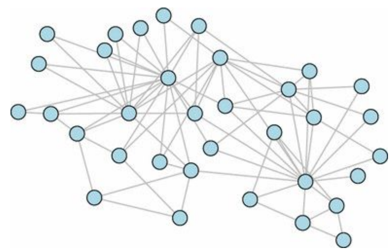
- **AMR - to - text Generation**

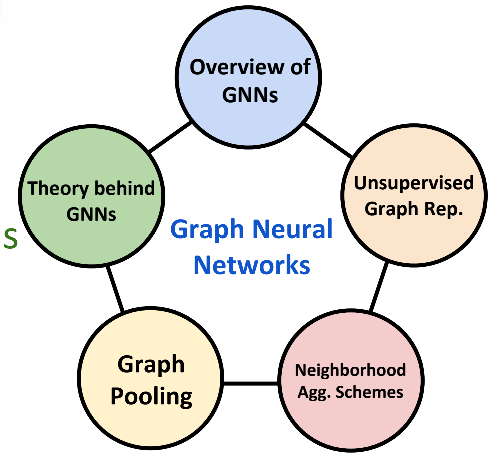
  [Xu et al., EMNLP'18, Songu et al., ACL'19, Damonteiu et al., NAACL'19, Ribiero et al., EMNLP'19, Zhu et al., EMNLP'19,]

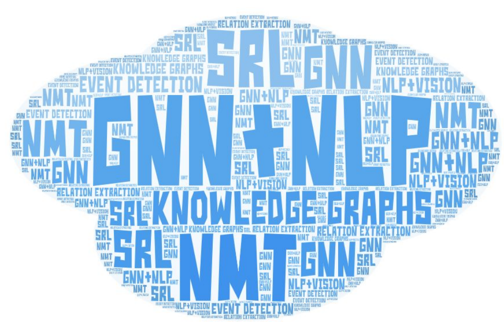
- **Sentiment analysis and many more...**

Tutorial Outline

- **Introduction**
  - ✓ Motivation
  - ✓ GNN Foundation
- **Methods**
  - ✓ Introduction to GNNs
  - ✓ Graph Pooling Unsupervised Learning using GNNs
  - ✓ Neighborhood Aggregation in GNNs
  - ✓ Other GNN Variants
- **Implementing GCNs**
- **Applications**
  - ✓ Semantic Role Labeling, NMT
  - ✓ Text Classification, Extraction
  - ✓ Knowledge Graphs
  - ✓ Vision + NLP
- **Open Problems and Conclusion**





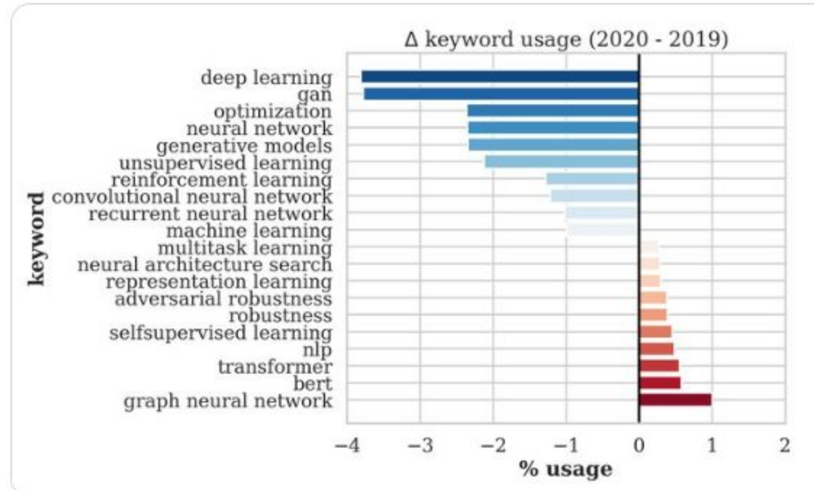


?

Open Problems

- GNN+NLP is on the rise!

#ICLR2020 submissions on graph neural networks, NLP and robustness have the greatest growth. @iclr_conf @openreviewnet



- There exist several graph structures in NLP which can be exploited using GCNs
  - Abusive language detection, Social Navigation ...
- Spectral GCNs rather than first-order approximation

Open Problems

- Most GNN architectures are shallow
  - Performance drops after 2-3 layers
- GNNs for handling Dynamic Graphs are underexplored
- Scaling GNNs to Large Graphs
- Several GNN models are applicable to NLP, only a small subset explored
- Finding best graph for the downstream task

Conclusion

- Graphs are everywhere and effective tool for exploiting such graph structure in end-to-end learning.
- GNNs are versatile, can be applied over
  - Learning settings: Semi-supervised
  - Graph granularity: node level, link, subgraph, whole graph
  - Graph types: undirected, directed, multi-relational
- GNNs have achieved considerable success on several tasks. Many more possibilities ahead!
- Slides and Code of the tutorial will be made available
  - https://github.com/svjan5/GNNs-for-NLP

Other Related Tutorials

- NIPS 2017 "Geometric Deep Learning on Graphs and Manifolds" by M. Bronstein et. al. [link]
- CVPR 2017 "Geometric deep learning on graphs" by Bronstein et al. [link]
- SGP 2017 "Geometric deep learning on graphs" by M. Bronstein [link]
- IPAM Workshop 2018 "Convolutional Neural Networks on Graphs" by Xavier Bresson [link]
- TheWebConf (WWW) 2018 "Representation Learning on Networks" By Hamilton et. al. [link]

Questions?

References

- References

  - Shuman, David I., Sunil K. Narang, Pascal Frossard, Antonio Ortega and Pierre Vandergheynst. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." IEEE Signal Processing Magazine 30 (2013): 83-98. https://arxiv.org/abs/1211.0053
  - Defferrard, Michaël, Xavier Bresson and Pierre Vandergheynst. "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering." NIPS (2016). https://arxiv.org/abs/1606.09375
  - Kipf, Thomas N. and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks." ArXiv abs/1609.02907 (2016): n. Pag. https://arxiv.org/abs/1609.02907
  - Velickovic, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò and Yoshua Bengio. "Graph Attention Networks." ArXiv abs/1710.10903 (2017): n. Pag.
  - Gilmer, Justin, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals and George E. Dahl. "Neural Message Passing for Quantum Chemistry." ArXiv abs/1704.01212 (2017): n. Pag.
  - Hamilton, William L., Zhitao Ying and Jure Leskovec. "Inductive Representation Learning on Large Graphs." NIPS (2017).
  - Vinyals, Oriol, Samy Bengio and Manjunath Kudlur. "Order Matters: Sequence to sequence for sets." CoRR abs/1511.06391 (2015): n. Pag.
  - Marcheggiani, Diego and Ivan Titov. "Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling." EMNLP (2017).

- References

  - Bojchevski, Aleksandar and Stephan Günnemann. “Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking.” ICLR (2017).
  - Kipf, Thomas N. and Max Welling. “Variational Graph Auto-Encoders.” ArXiv abs/1611.07308 (2016): n. Pag.
  - Ying, Zhitao, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton and Jure Leskovec. “Hierarchical Graph Representation Learning with Differentiable Pooling.” NeurIPS (2018).
  - Zhang, Muhan, Zhicheng Cui, Marion Neumann and Yixin Chen. “An End-to-End Deep Learning Architecture for Graph Classification.” AAAI (2018).
  - Yadav, Prateek, Madhav Nimishakavi, Naganand Yadati, Shikhar Vashishth, Arun Rajkumar and Partha Pratim Talukdar. “Lovasz Convolutional Networks.” ArXiv abs/1805.11365 (2018): n. Pag. https://arxiv.org/abs/1805.11365
  - Vashishth, Shikhar, Prateek Yadav, Manik Bhandari and Partha Pratim Talukdar. “Confidence-based Graph Convolutional Networks for Semi-Supervised Learning.” AISTATS (2019). https://arxiv.org/abs/1901.08255
  - Velickovic, Petar, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio and R. Devon Hjelm. “Deep Graph Infomax.” ArXiv abs/1809.10341 (2018): n. pag.
  - Vashishth, Shikhar, Shib Sankar Dasgupta, Swayambhu Nath Ray and Partha Pratim Talukdar. “Dating Documents using Graph Convolution Networks.” ACL (2018). https://arxiv.org/abs/1902.00175

- References

    - Chen, Jian Jhen, Tengfei Ma and Cao Xiao. “FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling.” ArXiv abs/1801.10247 (2018): n. Pag.
    - Zhang, Ziwei, Peng Cui and Wenwu Zhu. “Deep Learning on Graphs: A Survey.” ArXiv abs/1812.04202 (2018): n. Pag.
    - Zhou, Jie, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu and Maosong Sun. “Graph Neural Networks: A Review of Methods and Applications.” ArXiv abs/1812.08434 (2018): n. Pag.
    - Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang and Philip S. Yu. “A Comprehensive Survey on Graph Neural Networks.” ArXiv abs/1901.00596 (2019): n. Pag.
    - Xu, Keyulu, Weihua Hu, Jure Leskovec and Stefanie Jegelka. “How Powerful are Graph Neural Networks?” ArXiv abs/1810.00826 (2018): n. Pag.
    - Chiang, Wei-Lin, Xuanqing Liu, Si Si, Yang Li, Samy Bengio and Cho-Jui Hsieh. “Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks.” ArXiv abs/1905.07953 (2019): n. Pag.
    - Yadati, Naganand, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis and Partha Pratim Talukdar. “HyperGCN: A New Method of Training Graph Convolutional Networks on Hypergraphs.” NeurIPS 2019. https://arxiv.org/abs/1809.02589
    - Vashishth, Shikhar, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya and Partha Pratim Talukdar. “Incorporating Syntactic and Semantic Information in Word Embeddings using Graph Convolutional Networks.” ACL (2018). https://arxiv.org/abs/1809.04283

- References

    - Vashishth, Shikhar, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya and Partha Pratim Talukdar. “RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information.” EMNLP (2018). https://arxiv.org/abs/1812.04361
    - Narasimhan, Medhini, Svetlana Lazebnik and Alexander G. Schwing. “Out of the Box: Reasoning with Graph Convolution Nets for Factual Visual Question Answering.” *NeurIPS 2018*.
    - Bastings, Joost and Titov, Ivan and Aziz, Wilker and Marcheggiani, Diego and Sima{'}an, Khalil. “Graph Convolutional Encoders for Syntax-aware Neural Machine Translation.” EMNLP 2017
    - Marcheggiani, Diego and Bastings, Joost and Titov, Ivan. “Exploiting Semantics in Neural Machine Translation with Graph Convolutional Networks.” NAACL 2018
    - Beck, Daniel and Haffari, Gholamreza and Cohn, Trevor. “Graph-to-Sequence Learning using Gated Graph Neural Networks.” ACL 2018
    - Nguyen, T., & Grishman, R. “Graph Convolutional Networks With Argument-Aware Pooling for Event Detection.” AAAI 2018
    - Liu, Xiao and Luo, Zhunchen and Huang, Heyan. “Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation.” EMNLP 2018
    - Ray, Swayambhu Nath and Dasgupta, Shib Sankar and Talukdar, Partha. “{AD}3: Attentive Deep Document Dater.” EMNLP 2018
    - Fu, Tsu-Jui and Li, Peng-Hsuan and Ma, Wei-Yun. “{G}raph{R}el: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction.” ACL 2019

- References

    - Vashishth, Shikhar, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya and Partha Pratim Talukdar. “RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information.” EMNLP (2018). https://arxiv.org/abs/1812.04361
    - Narasimhan, Medhini, Svetlana Lazebnik and Alexander G. Schwing. “Out of the Box: Reasoning with Graph Convolution Nets for Factual Visual Question Answering.” *NeurIPS 2018*.
    - Bastings, Joost and Titov, Ivan and Aziz, Wilker and Marcheggiani, Diego and Sima{'}an, Khalil. “Graph Convolutional Encoders for Syntax-aware Neural Machine Translation.” EMNLP 2017
    - Marcheggiani, Diego and Bastings, Joost and Titov, Ivan. “Exploiting Semantics in Neural Machine Translation with Graph Convolutional Networks.” NAACL 2018
    - Beck, Daniel and Haffari, Gholamreza and Cohn, Trevor. “Graph-to-Sequence Learning using Gated Graph Neural Networks.” ACL 2018
    - Nguyen, T., & Grishman, R. “Graph Convolutional Networks With Argument-Aware Pooling for Event Detection.” AAAI 2018
    - Liu, Xiao and Luo, Zhunchen and Huang, Heyan. “Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation.” EMNLP 2018
    - Ray, Swayambhu Nath and Dasgupta, Shib Sankar and Talukdar, Partha. “{AD}3: Attentive Deep Document Dater.” EMNLP 2018
    - Fu, Tsu-Jui and Li, Peng-Hsuan and Ma, Wei-Yun. “{G}raph{R}el: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction.” ACL 2019