Deep Learning on Graphs for Natural Language Processing

Lingfei Wu, Yu Chen, Heng Ji, and Bang Liu

SIGIR-2021 Tutorial

July 11th, 2021

Outline

DLG4NLP Introduction

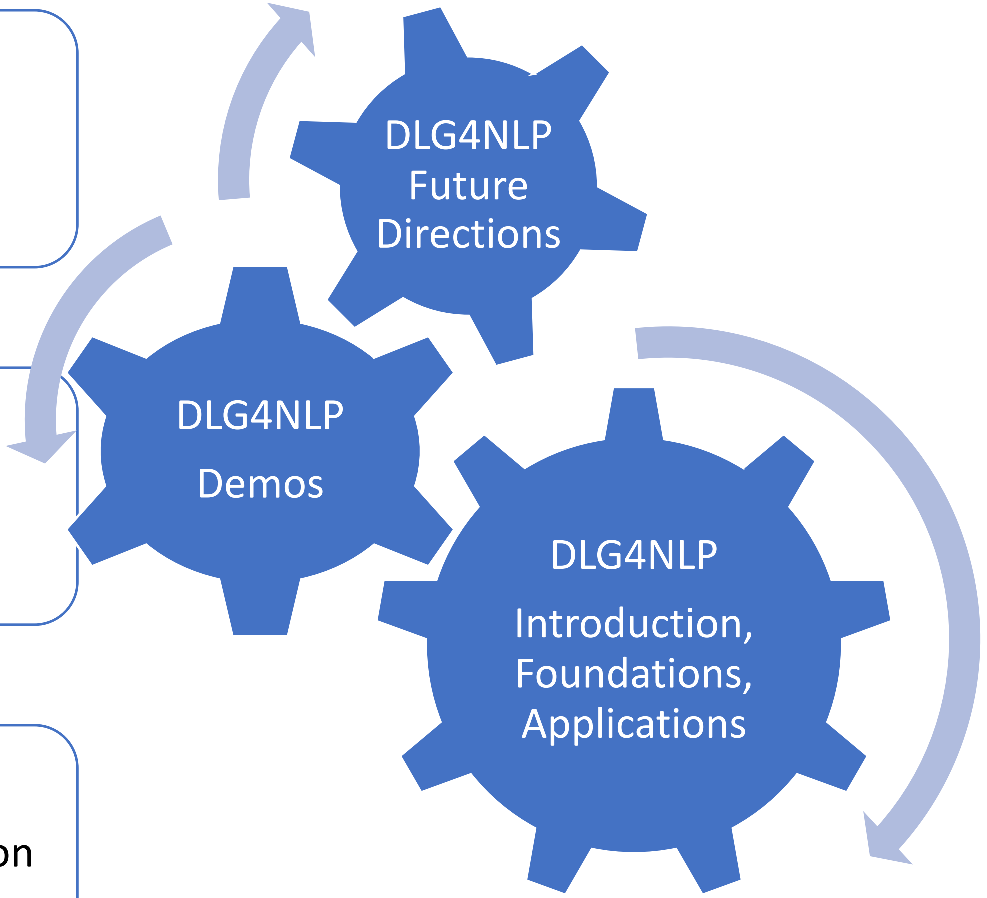
- Why Graphs for NLP?
- Conventional ML for NLP
- Deep Learning on Graphs: Foundations and Models

DLG4NLP Foundations

- Graph Construction for NLP
- Graph Representation Learning for NLP
- Graph Encoder-Decoder Models for NLP

DLG4NLP Applications

- Information Extraction
- Semantic Parsing and Machine Reading Comprehension
- Natural Language Generation and Machine Translation

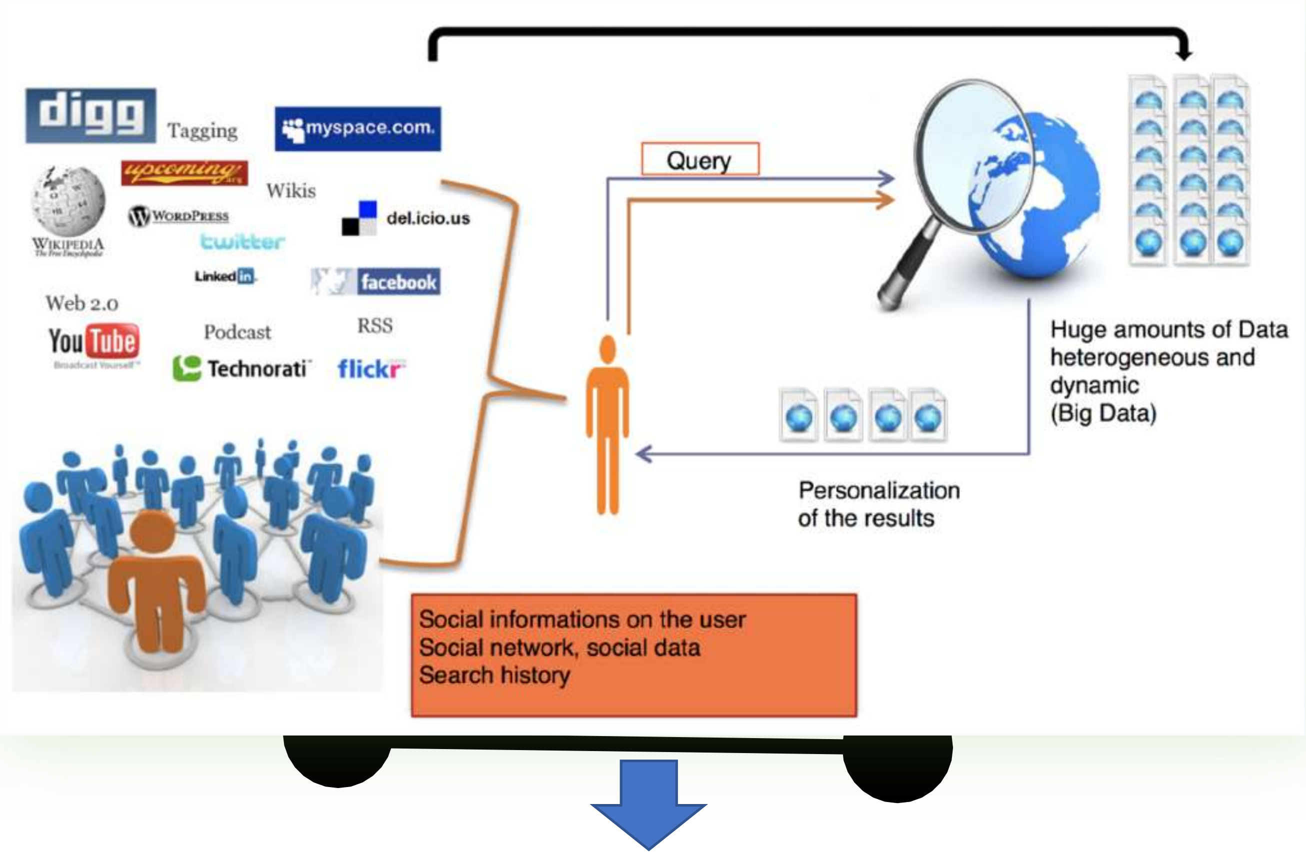


DLG4NLP
Introduction

Why graphs?

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

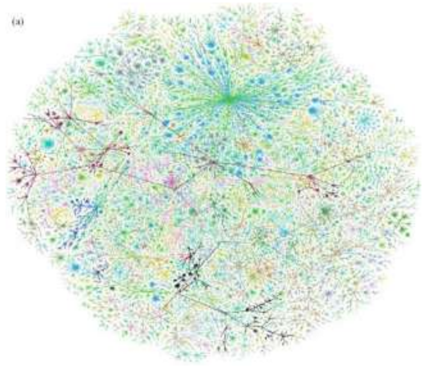
Source from Laboratoire Hubert Curien - Université Jean Monnet



Graph Topology + Node Attributes + Node Types

Graph!

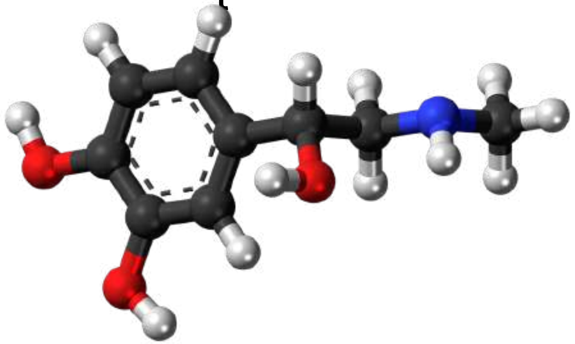
Graph-structured data are ubiquitous



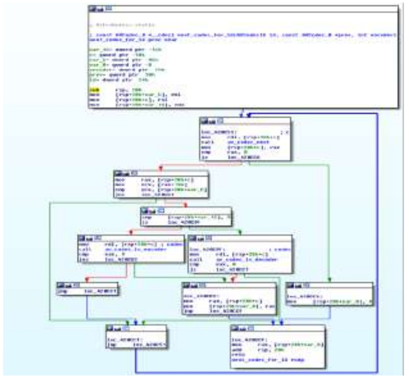
Internet

Social networks

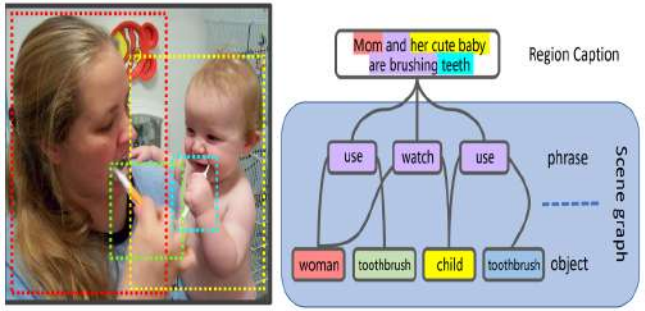
Information retrieval



Biomedical graphs

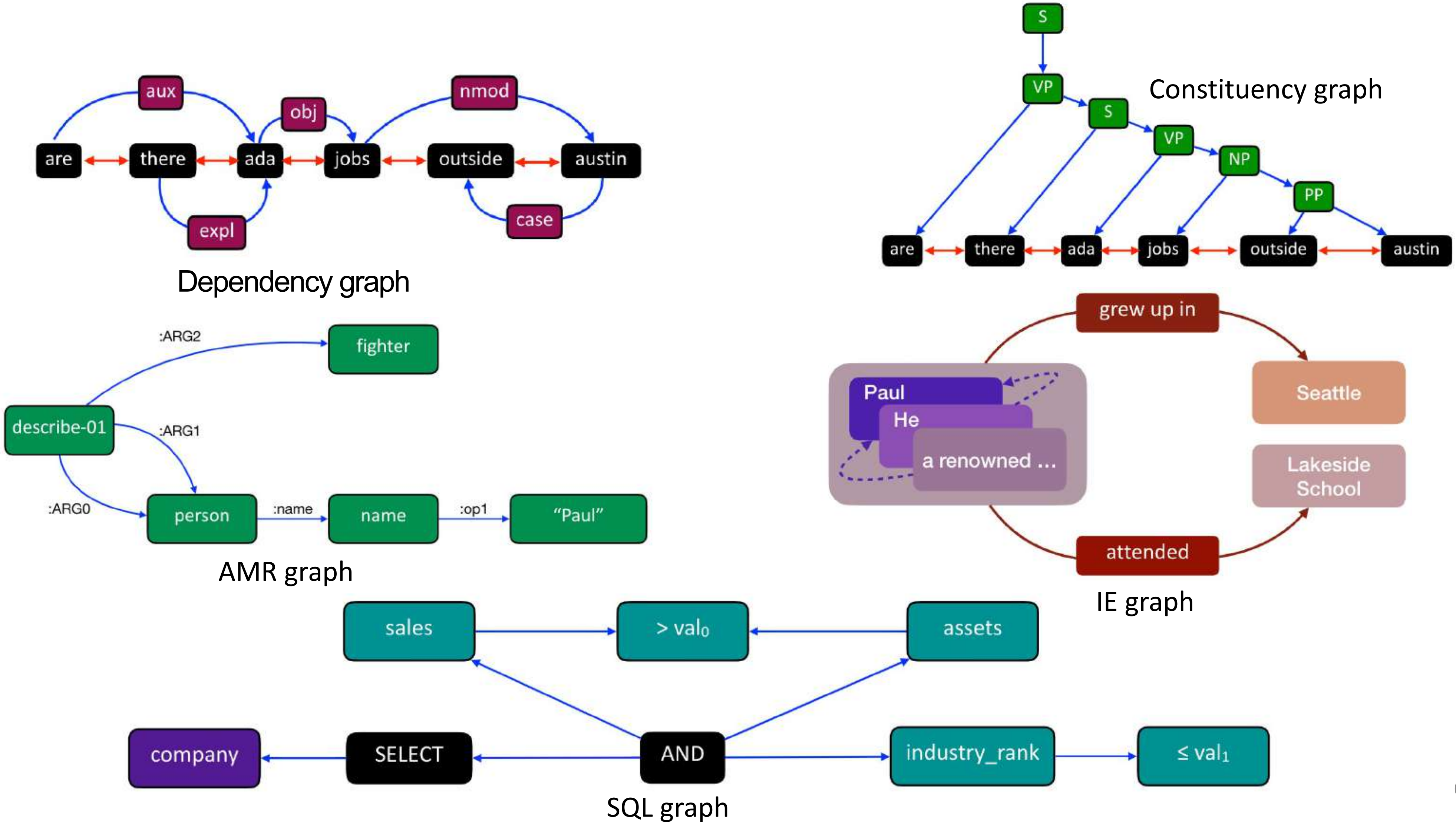


Program graphs



Scene graphs

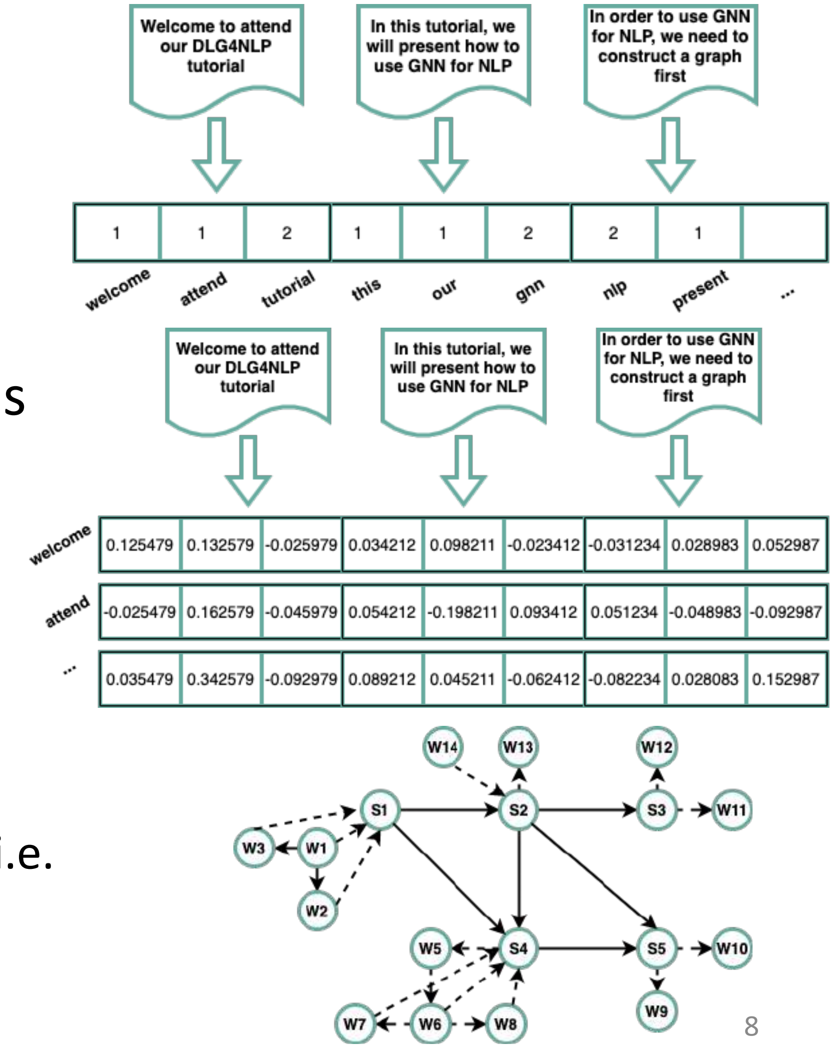
Graphs are ubiquitous in NLP As Well



Machine Learning on Graphs for NLP

Natural Language Processing: A Graph Perspective

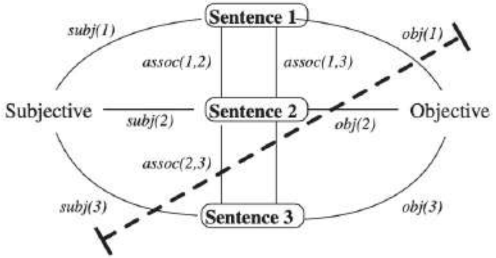
- Represent natural language as a bag of tokens
  - BOW, TF-IDF
  - Topic Modeling: text as a mixture of topics
- Represent natural language as a sequence of tokens
  - Linear-chain CRF
  - Word2vec, Glove
- Represent natural language as a graph
  - Dependency graphs, constituency graphs, AMR graphs, IE graphs, and knowledge graphs
  - Text graph containing multiple hierarchies of elements, i.e. document, sentence and word

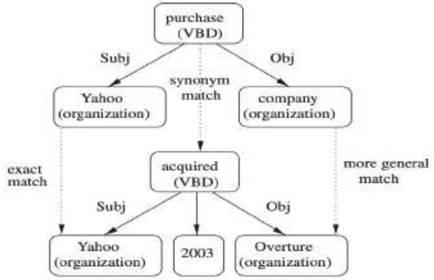


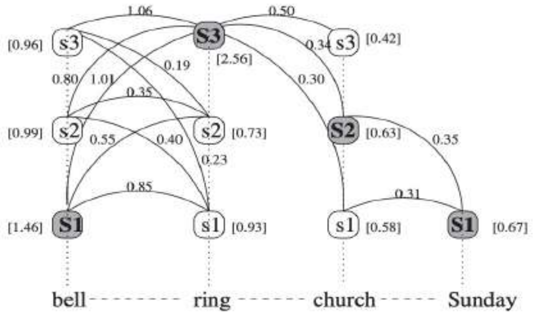
Graph Based Methods for NLP

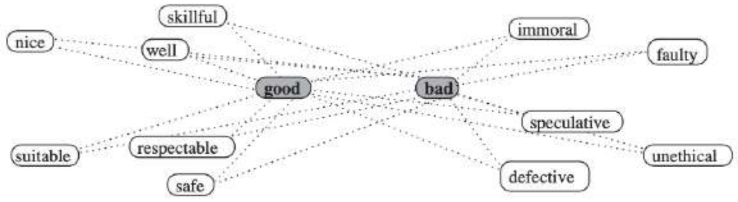
[Mihalcea and Radev, 2011]

- Random Walk Algorithms
  - Generate random paths, one can obtain a stationary distribution over all the nodes in a graph
  - Applications: semantic similarity of texts, name disambiguation
- Graph Clustering Algorithms
  - Spectral clustering, random walk clustering and min-cut clustering for text clustering
- Graph Matching Algorithms
  - Compute the similarity between two graphs for textual entailment task
- Label Propagation Algorithms
  - Propagate labels from labeled data points to previously unlabeled data points
  - Applications: word-sense disambiguation, sentiment analysis





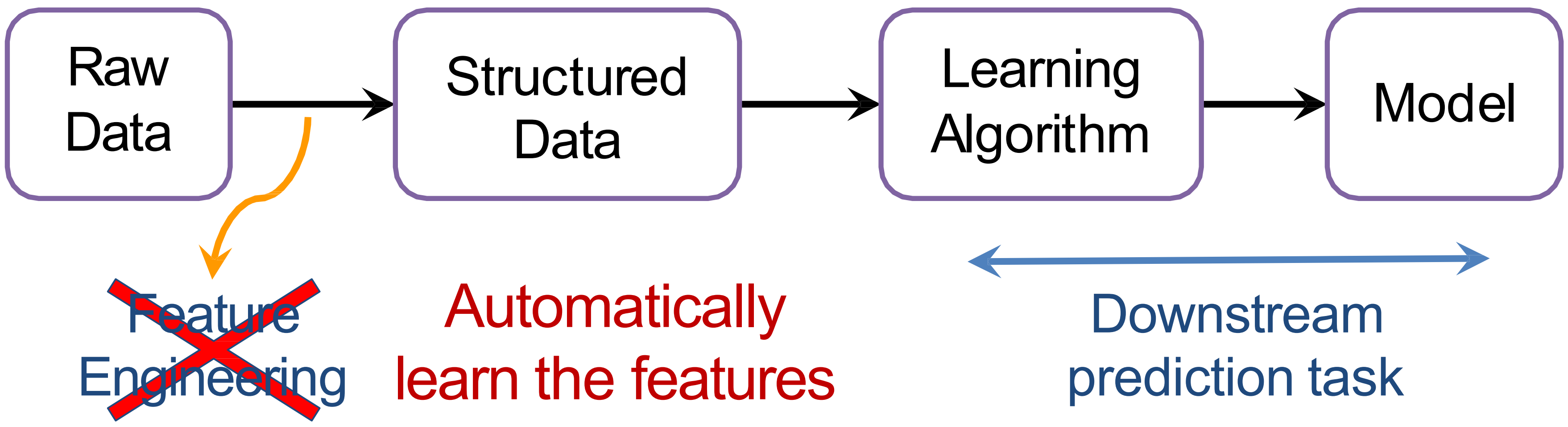




Deep Learning on Graphs: Foundations and Models

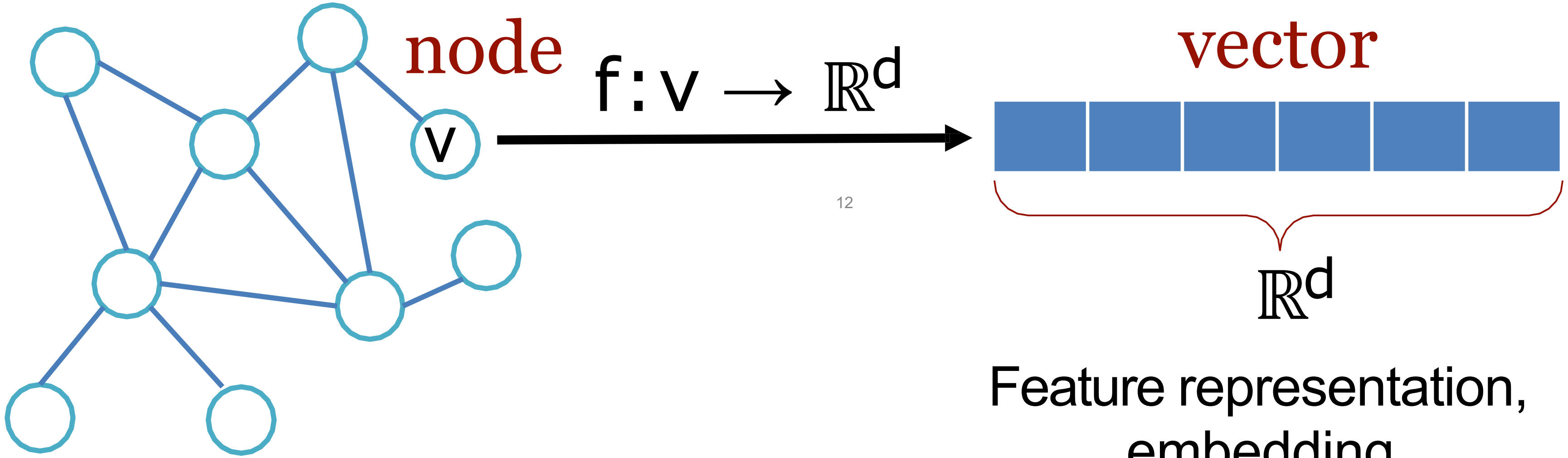
Machine Learning Lifecycle

- (Supervised) Machine Learning Lifecycle: feature learning is the key



Feature Learning in Graphs

- Our Goal: Design efficient task-independent/ task-dependent feature learning for machine learning in graphs!



Feature representation, embedding

Graph Neural Networks: Foundations

- Learning node embeddings:

A graph filter

adjacency matrix

$$\mathbf{h}_i^{(l)} = f_{\text{filter}}(A, \mathbf{H}^{(l-1)})$$

Updated node embeddings

Input node embeddings

$f_{\text{filter}}(\cdot,\cdot)$
- Spectral-based
- Spatial-based
- Attention-based
- Recurrent-based

- Learning graph-level embeddings:

$$A', \mathbf{H}' = f_{\text{pool}}(A, \mathbf{H})$$

A small graph w/ fewer nodes

New node embeddings

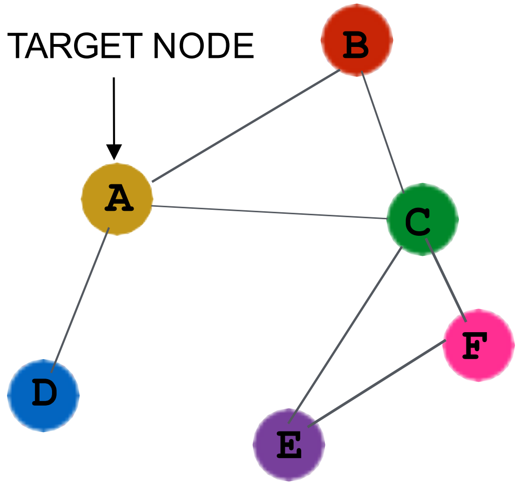
Input graph

Input node embeddings

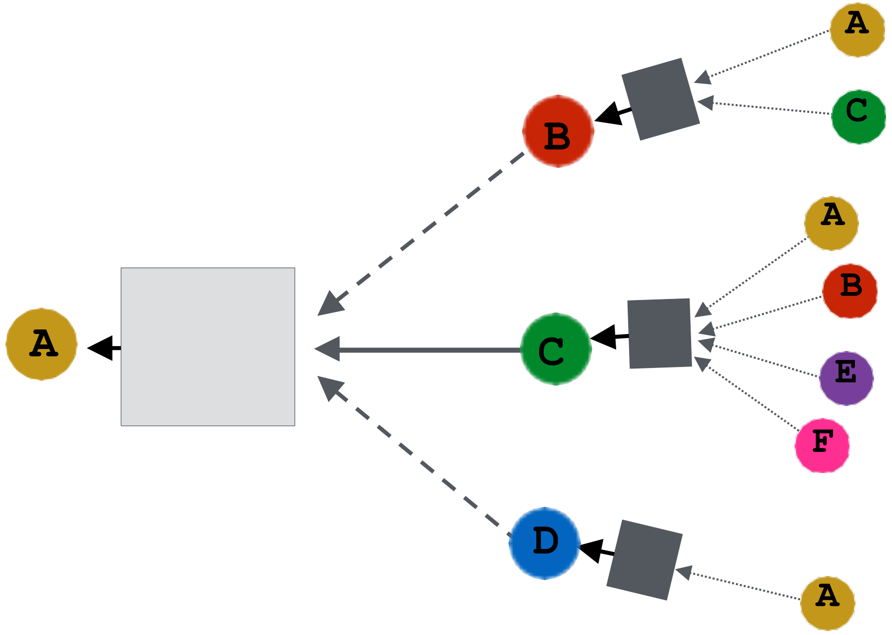
$f_{\text{pool}}(\cdot,\cdot)$
- Flat Graph Pooling (i.e. Max, Ave, Min)
- Hierarchical Graph Pooling (i.e. Diffpool)

Graph Neural Networks: Basic Model

- Key idea: Generate node embeddings based on local neighborhoods.



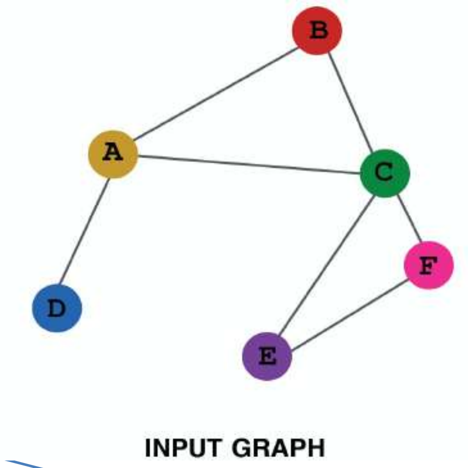
INPUT GRAPH

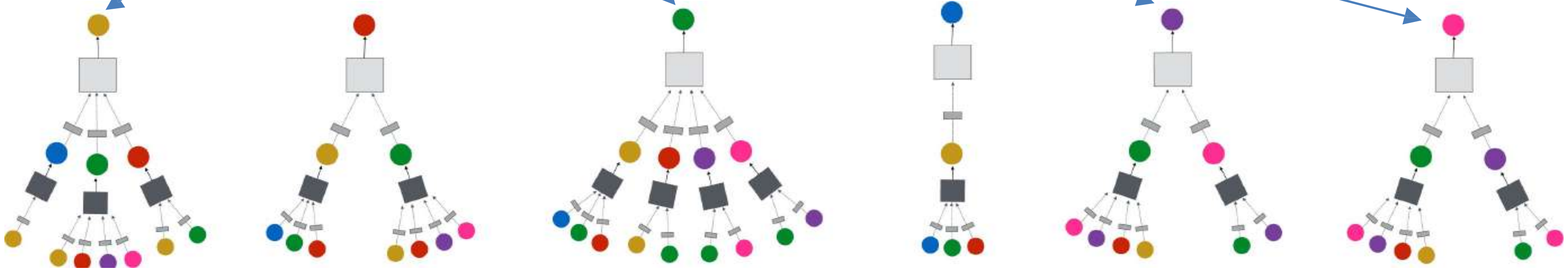


Neighborhood Aggregation

- Intuition: Network neighborhood defines a computation graph

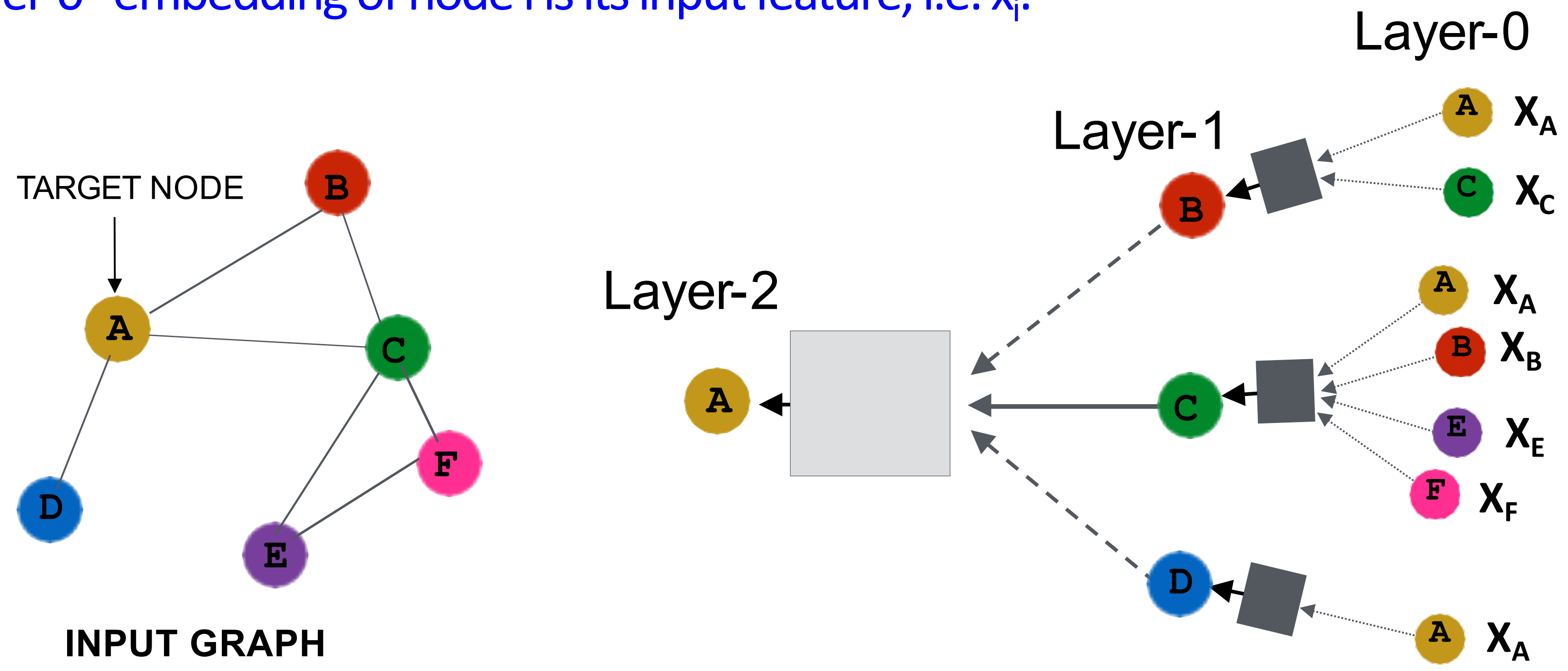
Every node defines a unique computation graph!





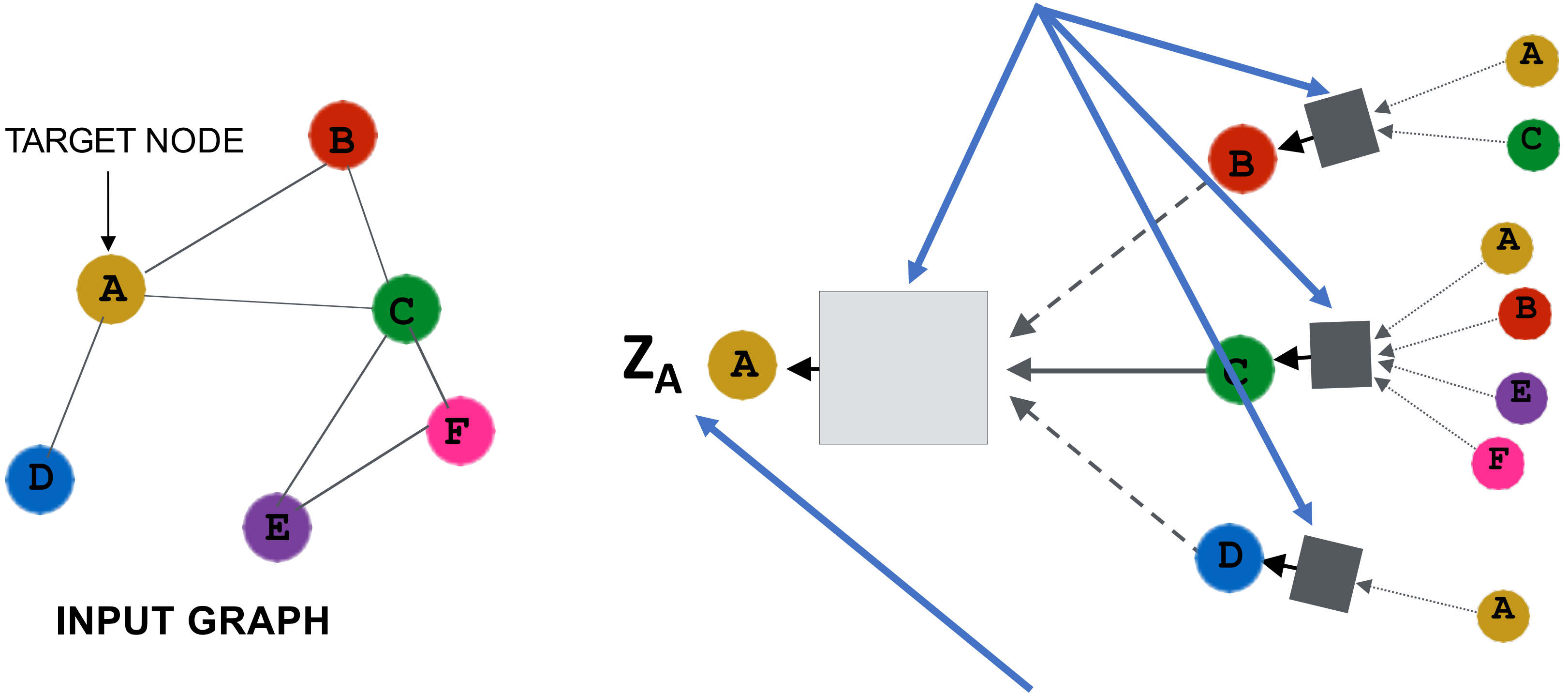
Neighborhood Aggregation

- Nodes have embeddings at each layer.
- Model can be arbitrary depth.
- “layer-0” embedding of node i is its input feature, i.e. x_i.



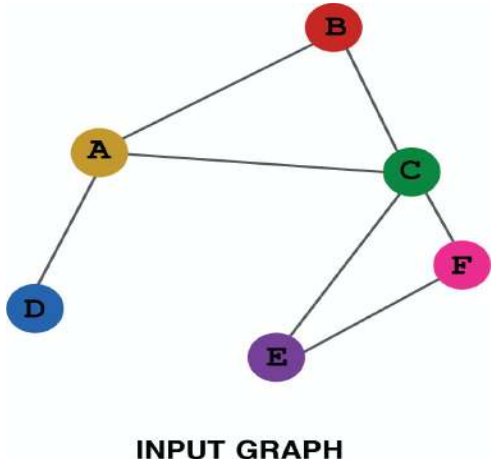
Overview of GNN Model

1) Define a neighborhood aggregation function

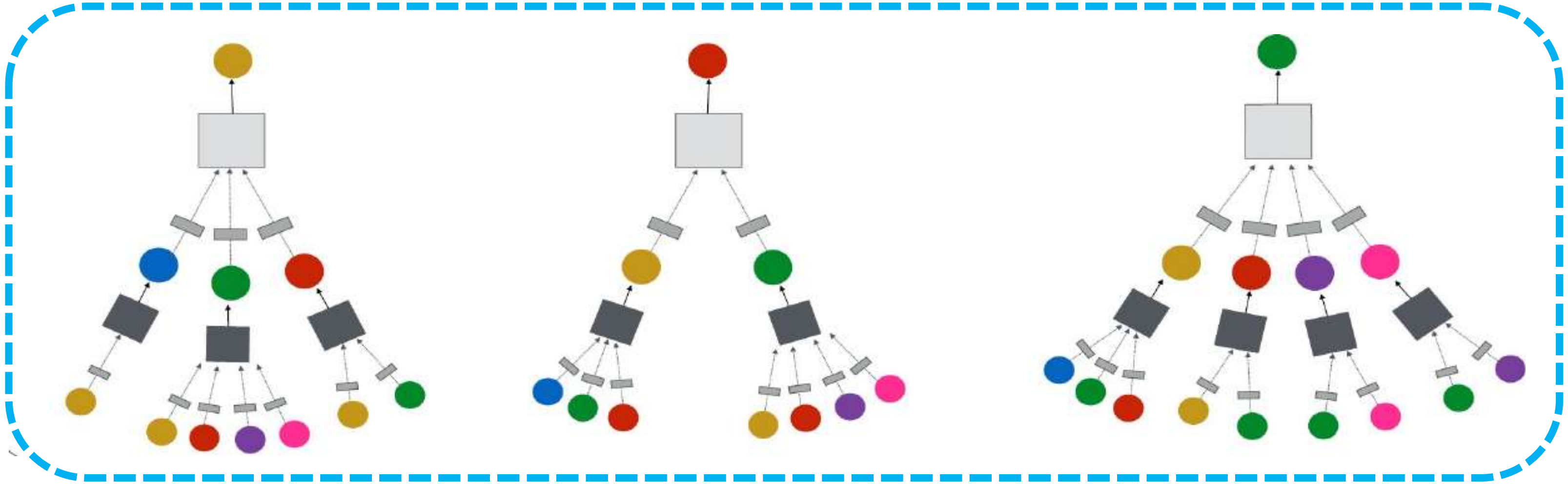


2) Define a loss function on the embeddings, **$L(z_v)$**

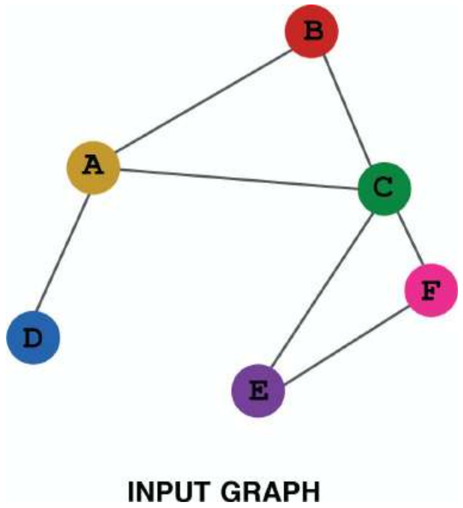
Overview of GNN Model



3) Train on a set of nodes, i.e., a batch of computation graphs

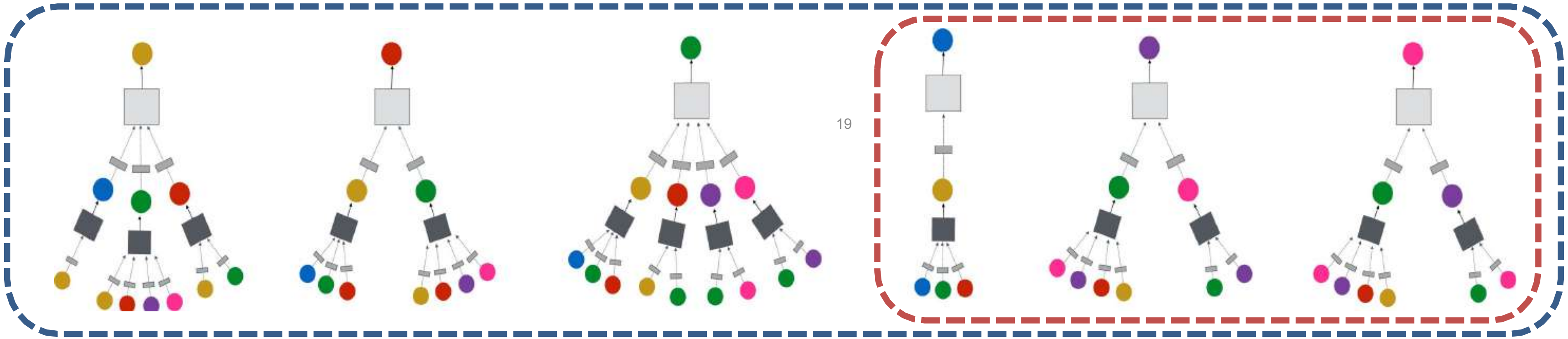


Overview of GNN Model



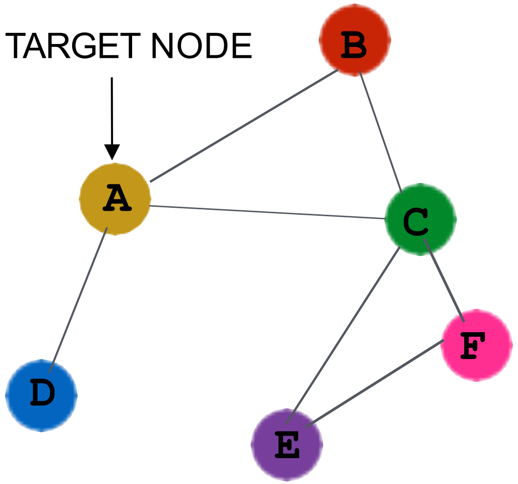
4) Generate embeddings for nodes as needed

Even for nodes we never trained on!



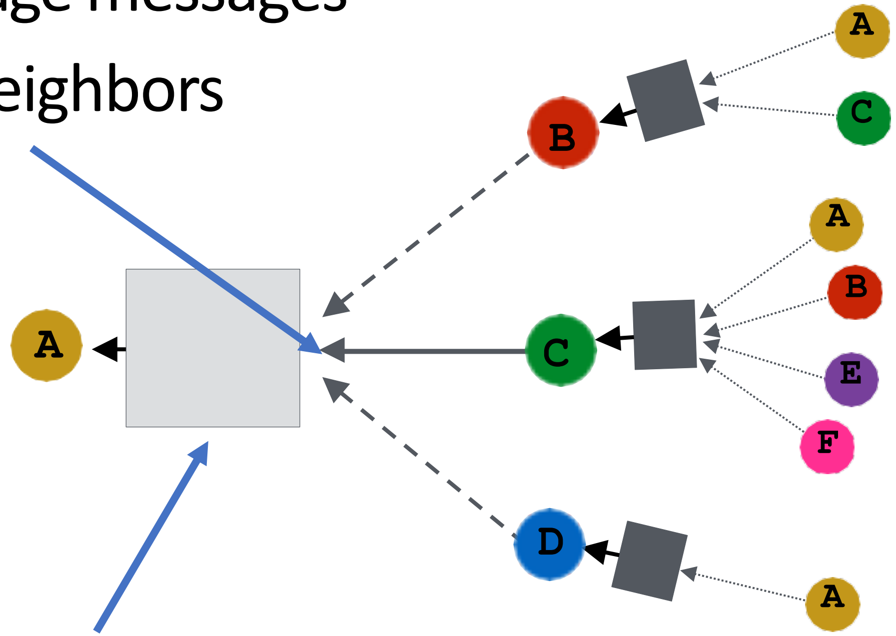
GNN Model: A Case Study

- Basic approach: Average neighbor information and apply a neural network



INPUT GRAPH

1) average messages from neighbors



2) apply neural network

GNN Model: A Case Study

- Basic approach: Average neighbor information and apply a neural network.

Initial "layer 0" embeddings are equal to node features

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

previous layer embedding of v

$$\boxed{\mathbf{h}_v^k} = \boxed{\sigma}\left(\mathbf{W}_k \boxed{\sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}} + \mathbf{B}_k \boxed{\mathbf{h}_v^{k-1}}\right), \quad \forall k > 0$$

kth layer embedding of v

non-linearity (e.g., ReLU or tanh)

average of neighbor's previous layer embeddings

GNN Model: Quick Summary

- Key idea: generate node embeddings by aggregating neighborhood information.
  - Allows for parameter sharing in the encoder
  - Allows for inductive learning

Graph Neural Networks: Popular Models

- Spectral-based Graph Filters
  - GCN (Kipf & Welling, ICLR 2017), Chebyshev-GNN (Defferrard et al. NIPS 2016)
- Spatial-based Graph Filters
  - MPNN (Gilmer et al. ICML 2017), GraphSage (Hamilton et al. NIPS 2017)
  - GIN (Xu et al. ICLR 2019)
- Attention-based Graph Filters
  - GAT (Velickovic et al. ICLR 2018)
- Recurrent-based Graph Filters
  - GGNN (Li et al. ICLR 2016)

Graph Convolution Networks (GCN)

Key idea: spectral convolution on graphs

Eigen-decomposition is expensive

$$f_{\text{filter}} * \mathbf{x}_i = \mathbf{U} f(\Lambda) \mathbf{U}^T \mathbf{x}_i$$

Chebyshev polynomials accelerates but still not powerful

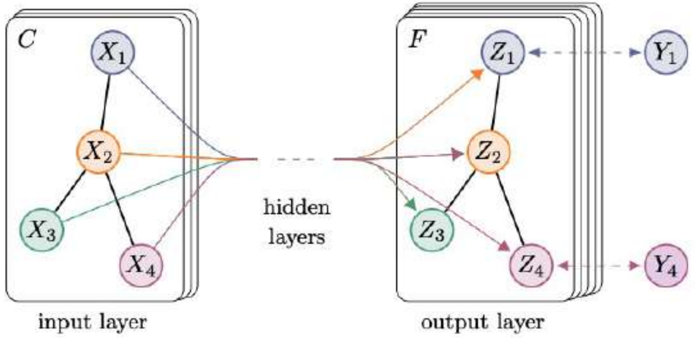
$$f'_{\text{filter}} * \mathbf{x}_i \approx \sum_{p=0}^{P} \theta'_p \mathbf{T}_p(\tilde{\mathbf{L}}) \mathbf{x}_i$$

First-order approxima-tion fast and powerful

$$f_{\text{filter}} * \mathbf{h}_i^{(l)} \approx \theta(I_n + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) \mathbf{h}_i^{(l)}$$

Renormalization trick stabilizes the numerical computation

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)})$$



GCN in NLP Tasks:

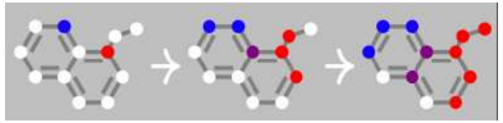
- Text classification
- Question Answering
- Text Matching
- Topic Modeling
- Information Extration

Message Passing Neural Network (MPNN)

Key idea: graph convolutions as a message passing process

MPNN:

$$\mathbf{h}_i^{(l)} = f_{\text{filter}}(A, \mathbf{H}^{(l-1)}) = f_U(\mathbf{h}_i^{(l-1)}, \sum_{v_j \in N(v_i)} f_M(\mathbf{h}_i^{(l-1)}, \mathbf{h}_j^{(l-1)}, \mathbf{e}_{i,j}))$$



expensive if the number of nodes are large

Update and aggregation functions

Node and edge embeddings

GraphSage:

$$f_{\text{filter}}(A, \mathbf{H}^{(l-1)}) = \sigma(\mathbf{W}^{(l)} \cdot f_M(\mathbf{h}_i^{(l-1)}, \{\mathbf{h}_j^{(l-1)}, \forall v_j \in N(v_i)\}))$$

sampling to obtain a fixed number of neighbors

Aggregation functions

Node embeddings

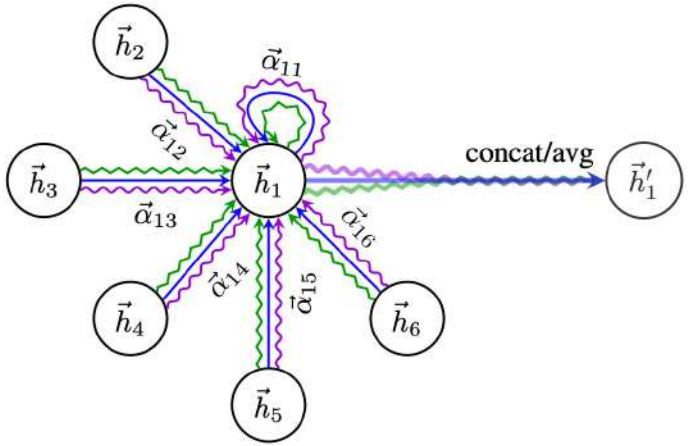
MPNN and GraphSage in NLP Tasks:

- Knowledge graph
- Information extraction
- Semantic parsing

Graph4NLP

Graph Attention Network (GAT)

Key idea: dynamically learn the weights (attention scores) on the edges when performing message passing



Weighted sum of node embeddings

$$\mathbf{h}_i^{(l)} = f_{\text{filter}}(A, \mathbf{H}^{(l-1)}) = \sigma\Big(\sum_{v_j \in N(v_i)} \alpha_{ij} \mathbf{W}^{(l)} \mathbf{h}_j^{(l-1)}\Big)$$

Learned local weights with self-attention

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(\mathbf{u}^{(l)^T}[\mathbf{W}^{(l)}\mathbf{h}_i^{(l-1)} || \mathbf{W}^{(l)}\mathbf{h}_j^{(l-1)}]))}{\sum_{v_k \in N(v_i)} \exp(\text{LeakyReLU}(\mathbf{u}^{(l)^T}[\mathbf{W}^{(l)}\mathbf{h}_i^{(l-1)} || \mathbf{W}^{(l)}\mathbf{h}_k^{(l-1)}]))}$$

Intermediate node embeddings

$$f_{\text{filter}}(A, \mathbf{H}^{(l-1)}) = ||_{k=1}^{K} \sigma\Big(\sum_{v_j \in N(v_i)} \alpha_{ij}^k \mathbf{W}_k^{(l)} \mathbf{h}_j^{(l-1)}\Big)$$

Final node embeddings

$$f_{\text{filter}}(A, \mathbf{H}^{(L-1)}) = \sigma\Big(\frac{1}{K}\sum_{k=1}^{K} \sum_{v_j \in N(v_i)} \alpha_{ij}^k \mathbf{W}_k^{(L)} \mathbf{h}_j^{(L-1)}\Big)$$

GAT in NLP Tasks:

- Text classification
- Question Answering
- Knowledge graph
- Information extraction
- Semantic parsing

Gated Graph Neural Networks (GGNN)

Key idea: the use of Gated Recurrent Units while taking into account edge type and directions

Zero-padding input node embeddings

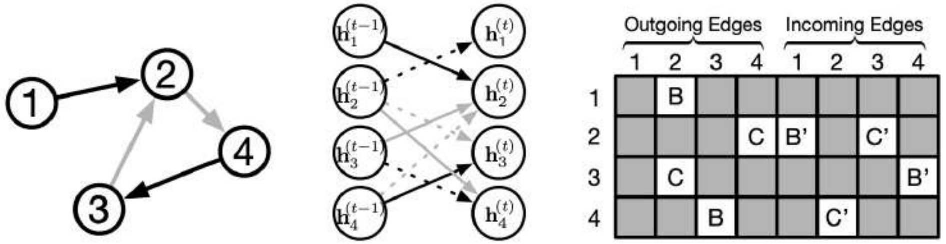
Incoming & outgoing edges for node v_i

$$\mathbf{h}_i^{(0)} = [\mathbf{x}_i^T, \mathbf{0}]^T$$

$$\mathbf{a}_i^{(l)} = A_{i:}^T [\mathbf{h}_1^{(l-1)} \ldots \mathbf{h}_n^{(l-1)}]^T$$

$$\mathbf{h}_i^{(l)} = \mathrm{GRU}(\mathbf{a}_i^{(l)}, \mathbf{h}_i^{(l-1)})$$

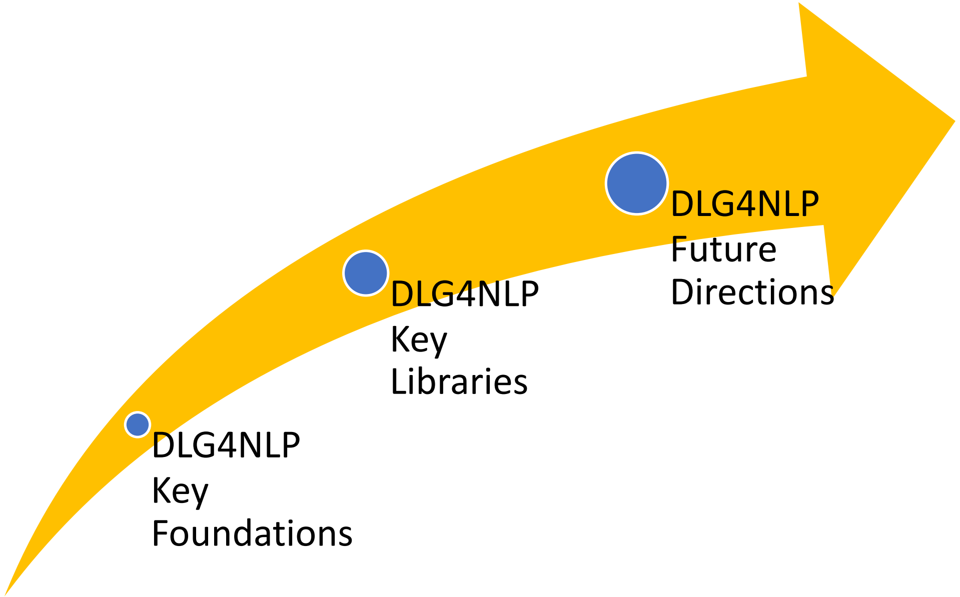
GRU for fusing node embeddings

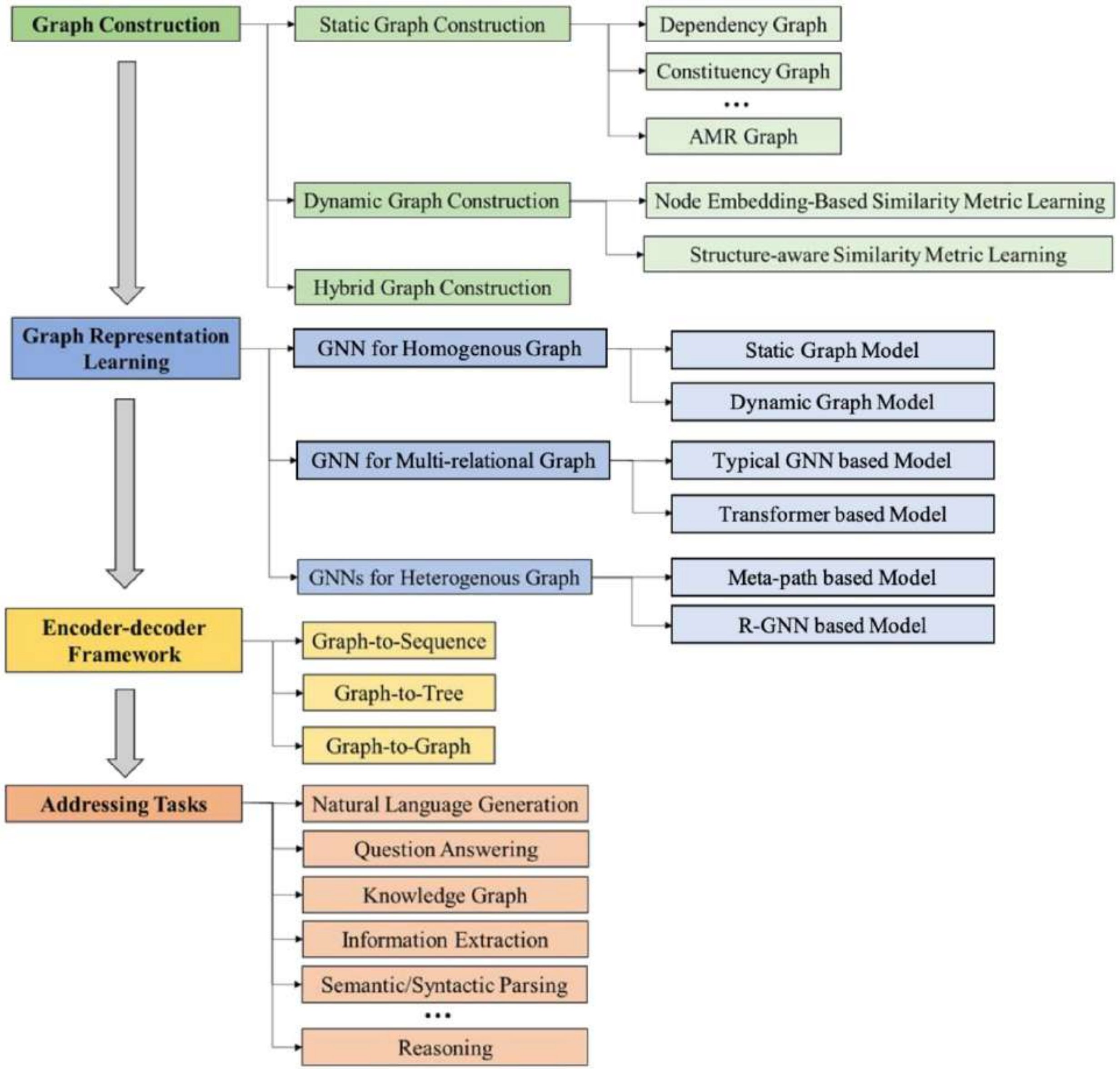


GGNN in NLP Tasks:

- Semantic parsing
- Machine translation

DLG4NLP: A Roadmap



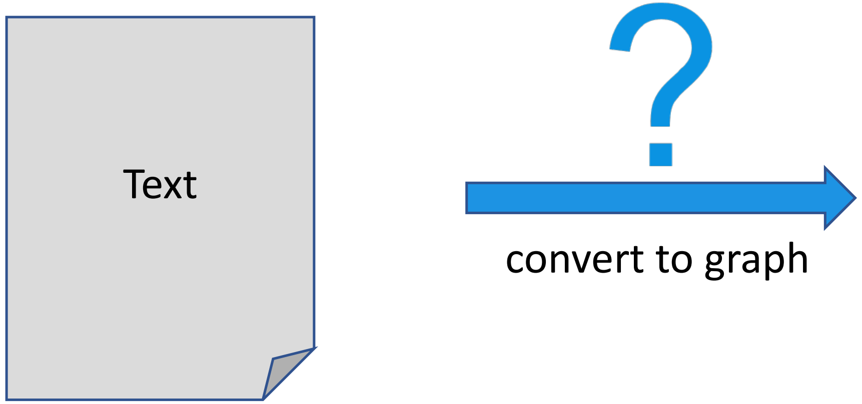


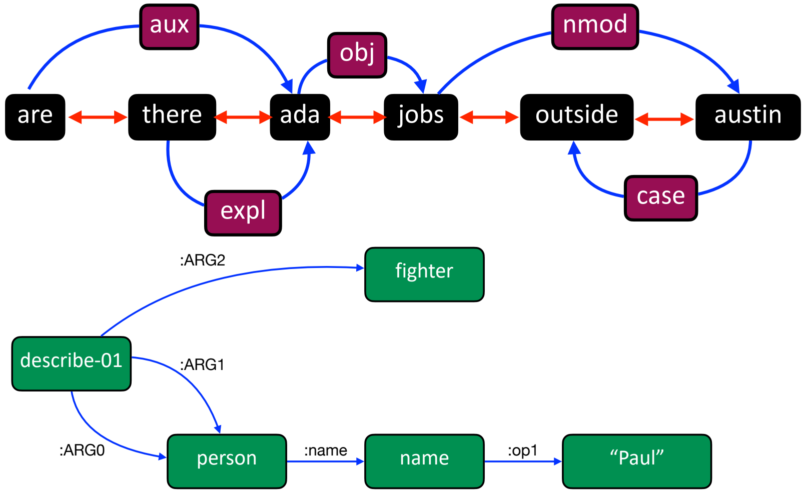
DLG4NLP Foundations

Graph Construction for NLP

Why Graph Construction for NLP?

- Representation power: graph > sequence > bag
- Different NLP tasks require different aspects of text , e.g., syntax, semantics.
- Different graphs capture different aspects of the text
- Two categories: static vs dynamic graph construction
- Goal: good downstream task performance

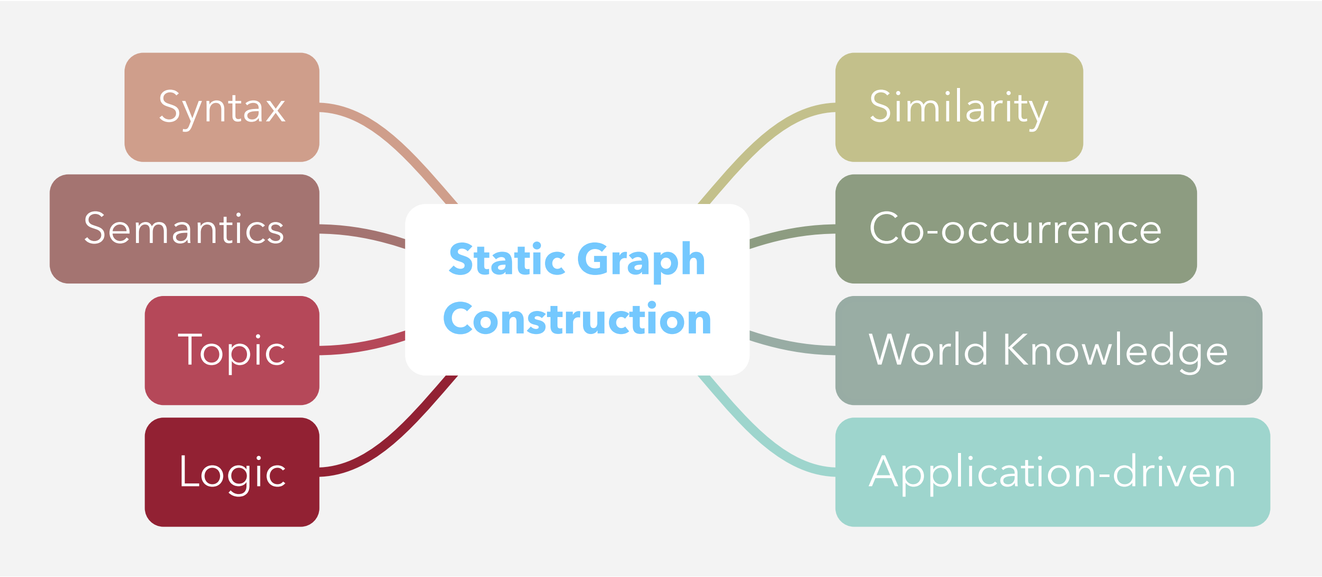




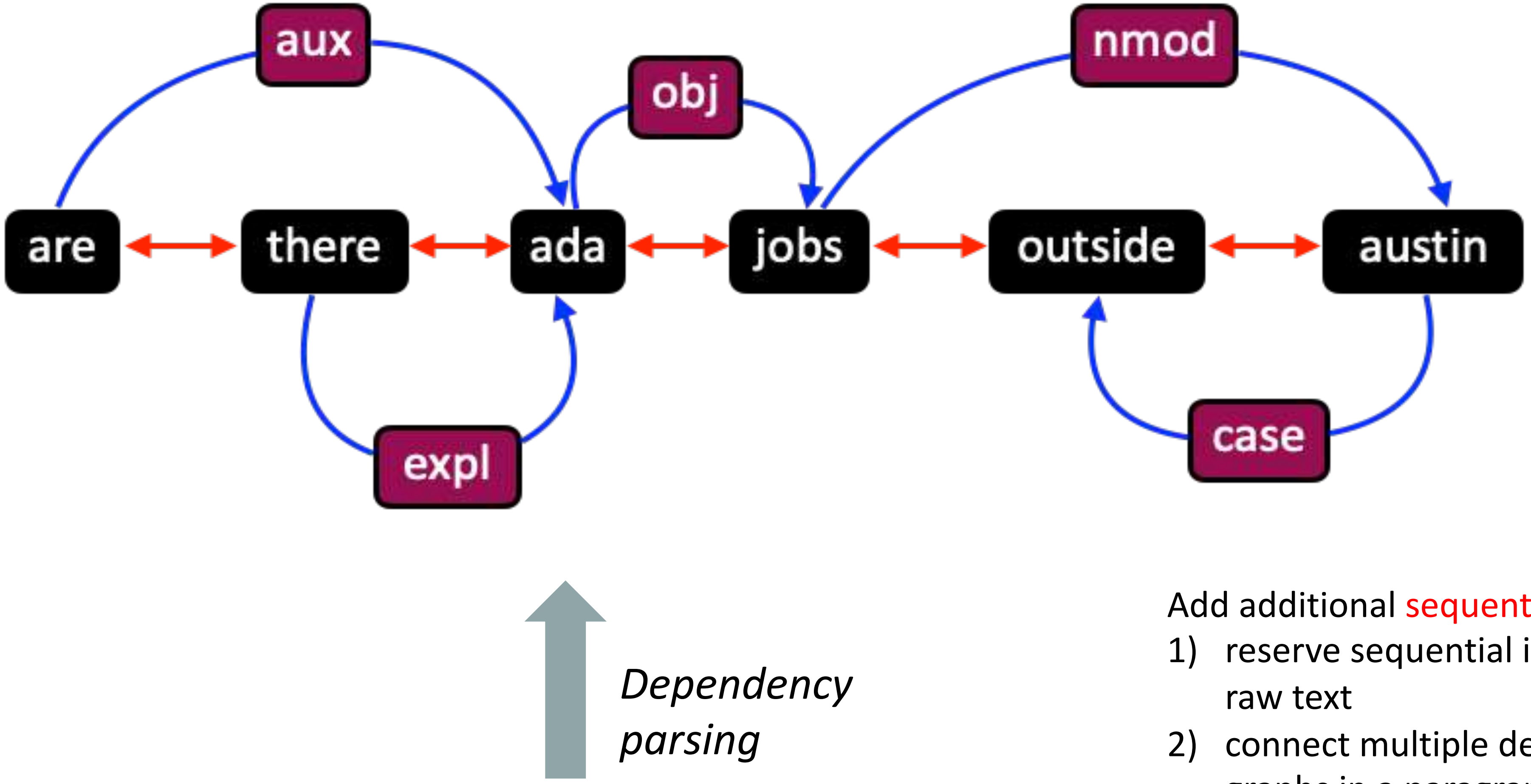
many more graph options…

Static Graph Construction

- Problem setting:
  - Input: raw text (e.g., sentence, paragraph, document, corpus)
  - Output: graph
- Conducted during preprocessing by augmenting text with domain knowledge



Static Graph Construction: Dependency Graph

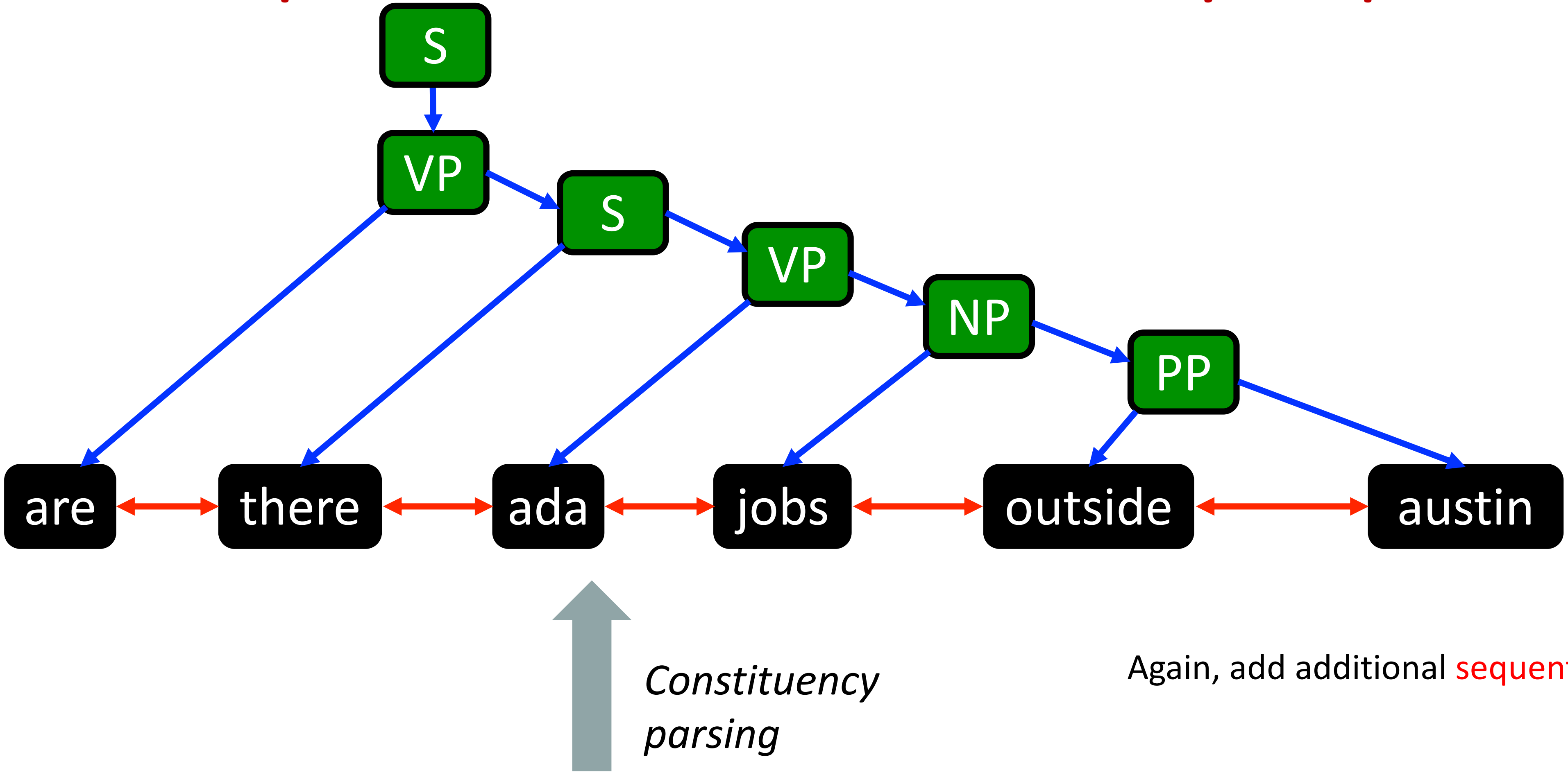


Add additional sequential edges to

1) reserve sequential information in raw text
2) connect multiple dependency graphs in a paragraph

Text input: are there ada jobs outside austin

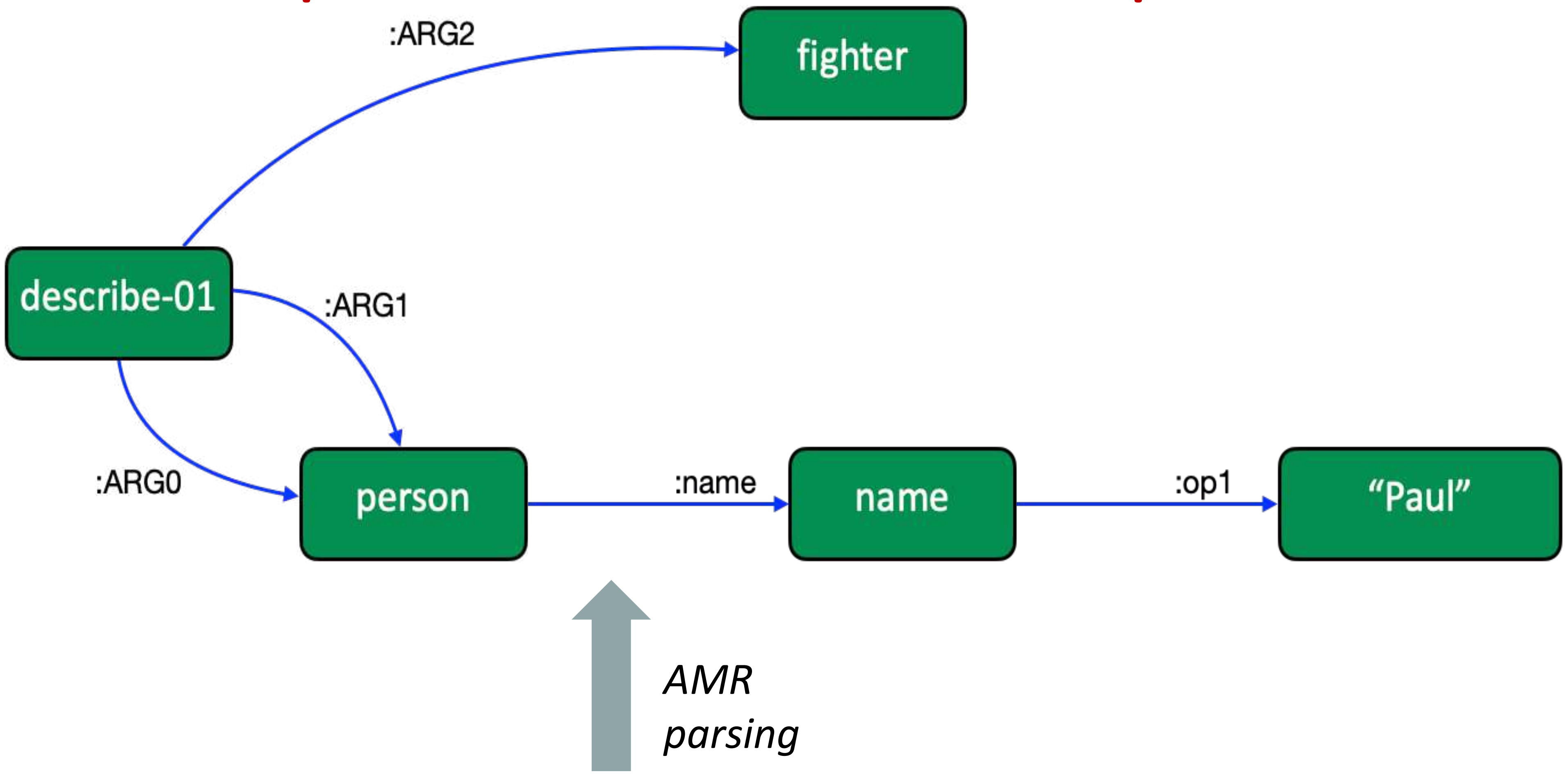
Static Graph Construction: Constituency Graph



Again, add additional sequential edges

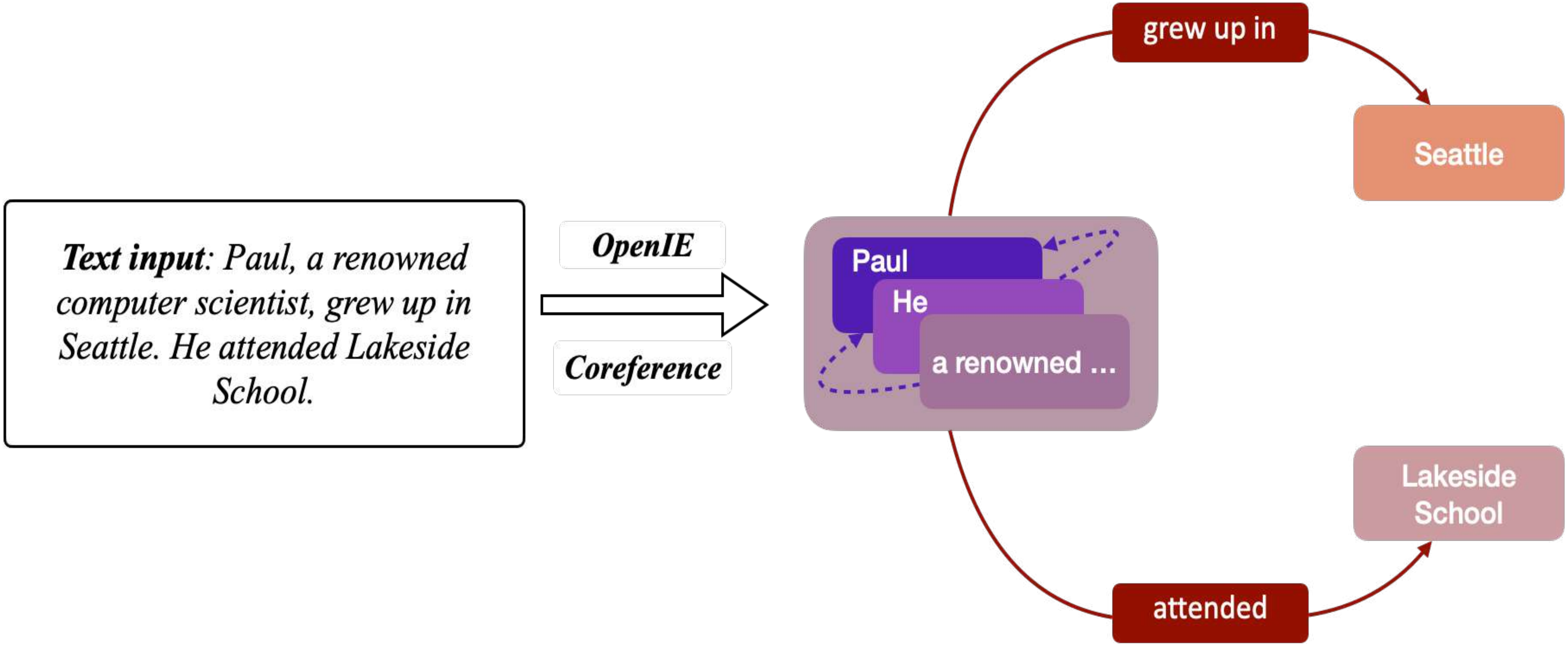
Text input: are there ada jobs outside austin

Static Graph Construction: AMR Graph

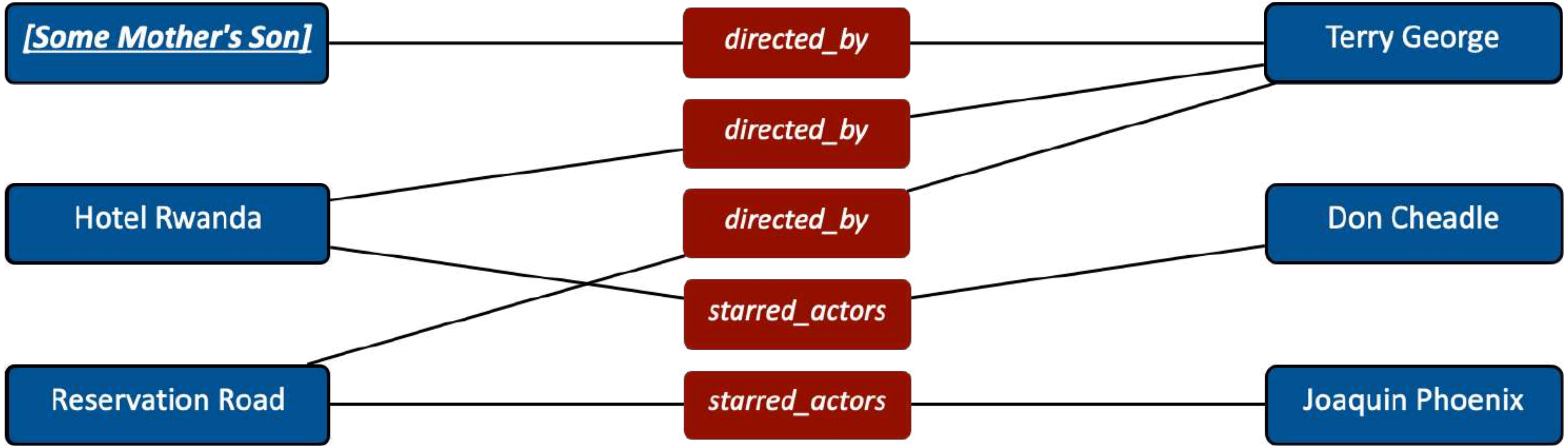


Text input: Paul's description of himself: a fighter

Static Graph Construction: IE Graph



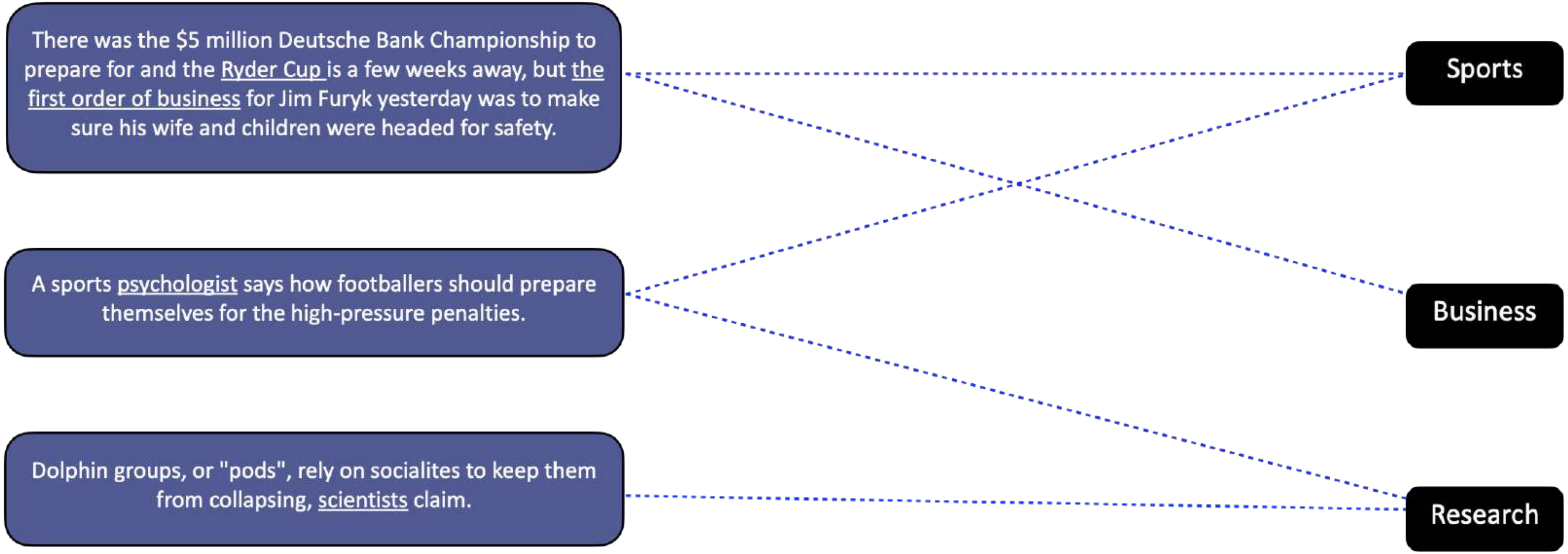
Static Graph Construction: Knowledge Graph



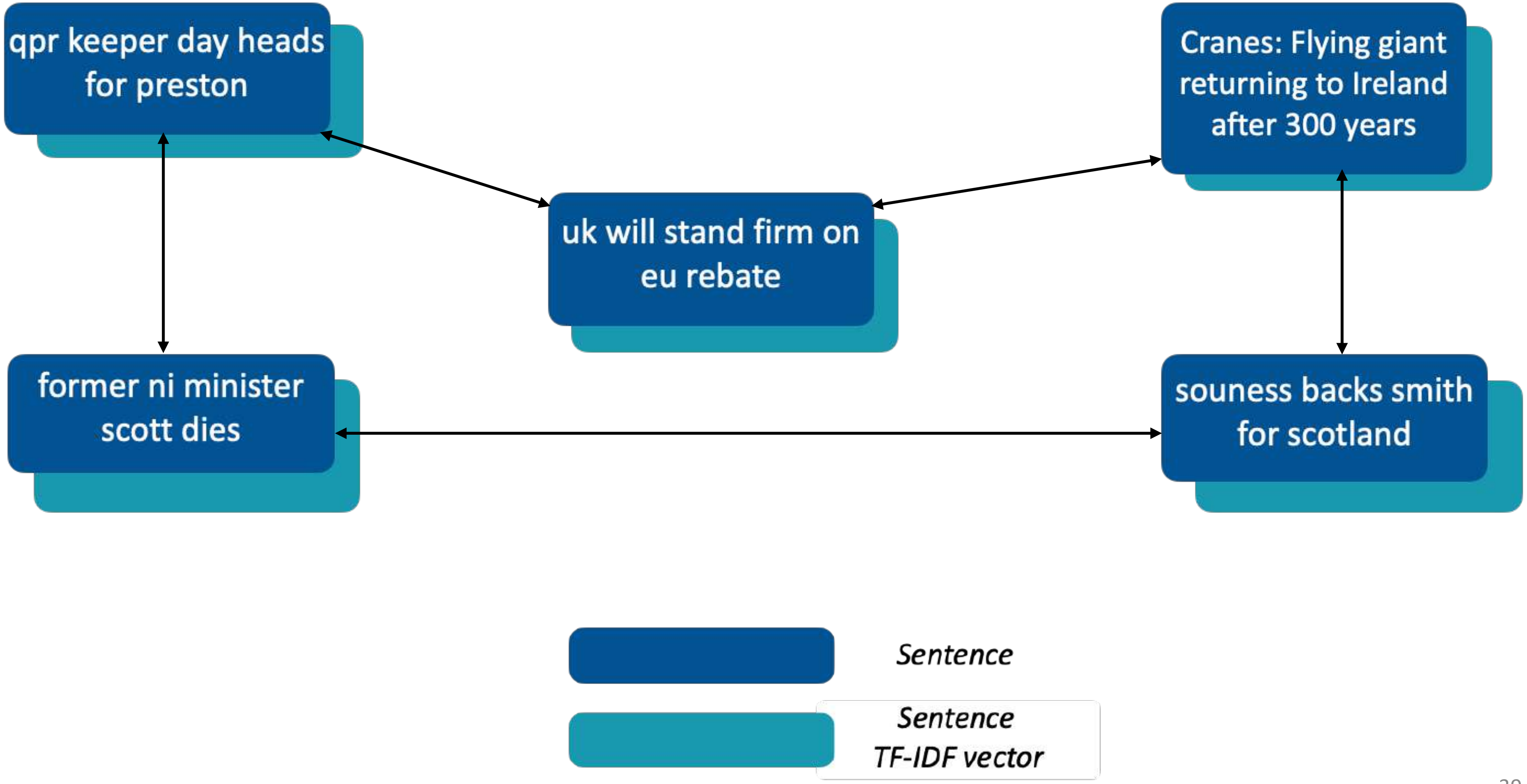
Get the concept sub-graph from KB

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

Static Graph Construction: Topic Graph



Static Graph Construction: Similarity Graph



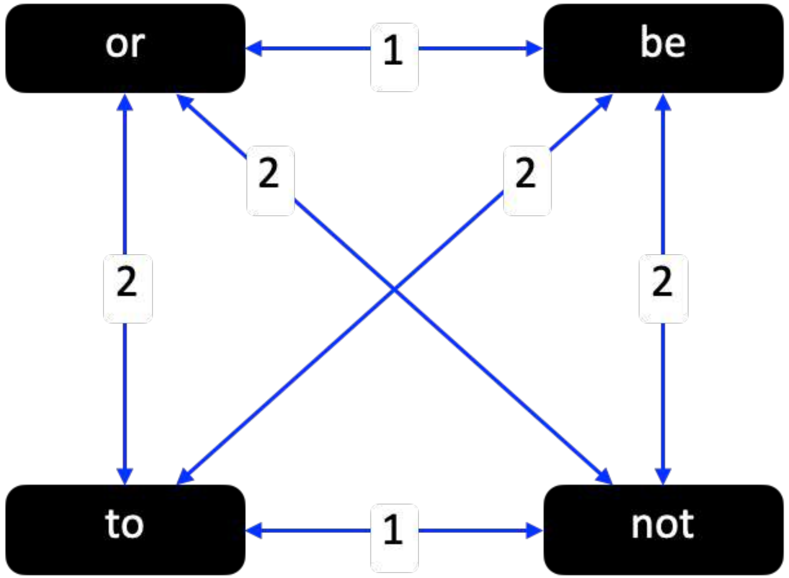
Static Graph Construction: Co-occurrence Graph

Text input: *To be, or not to be: ...*

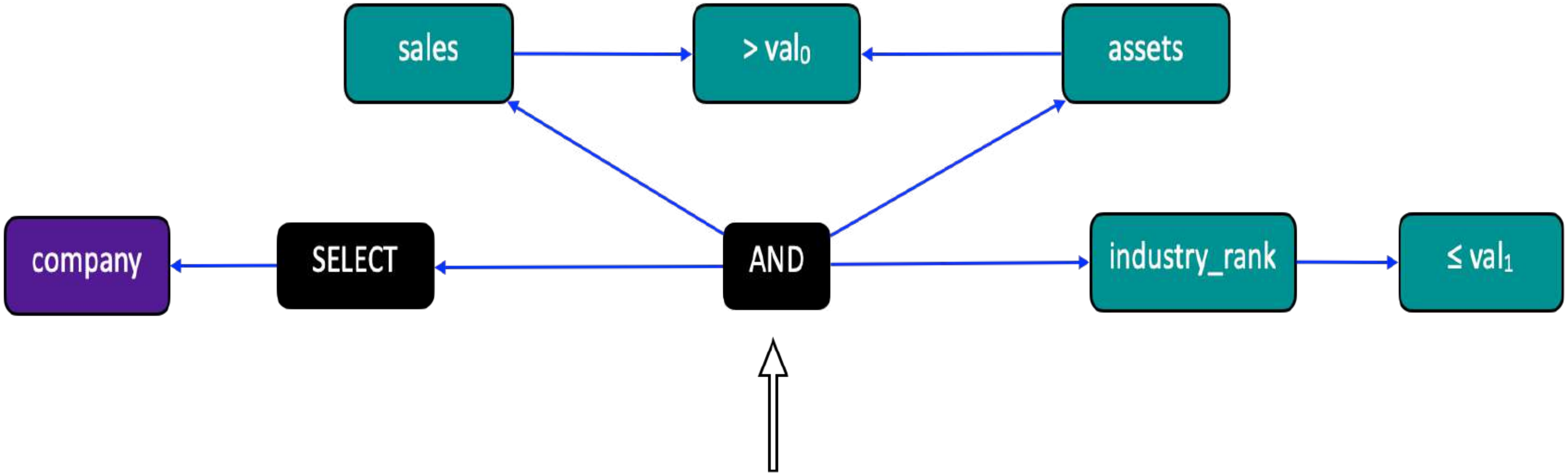
Co-occurrence matrix

	to	be	or	not
to		2	2	1
be	2		1	2
or	2	1		1
not	1	2	1	

Co-occurrence graph

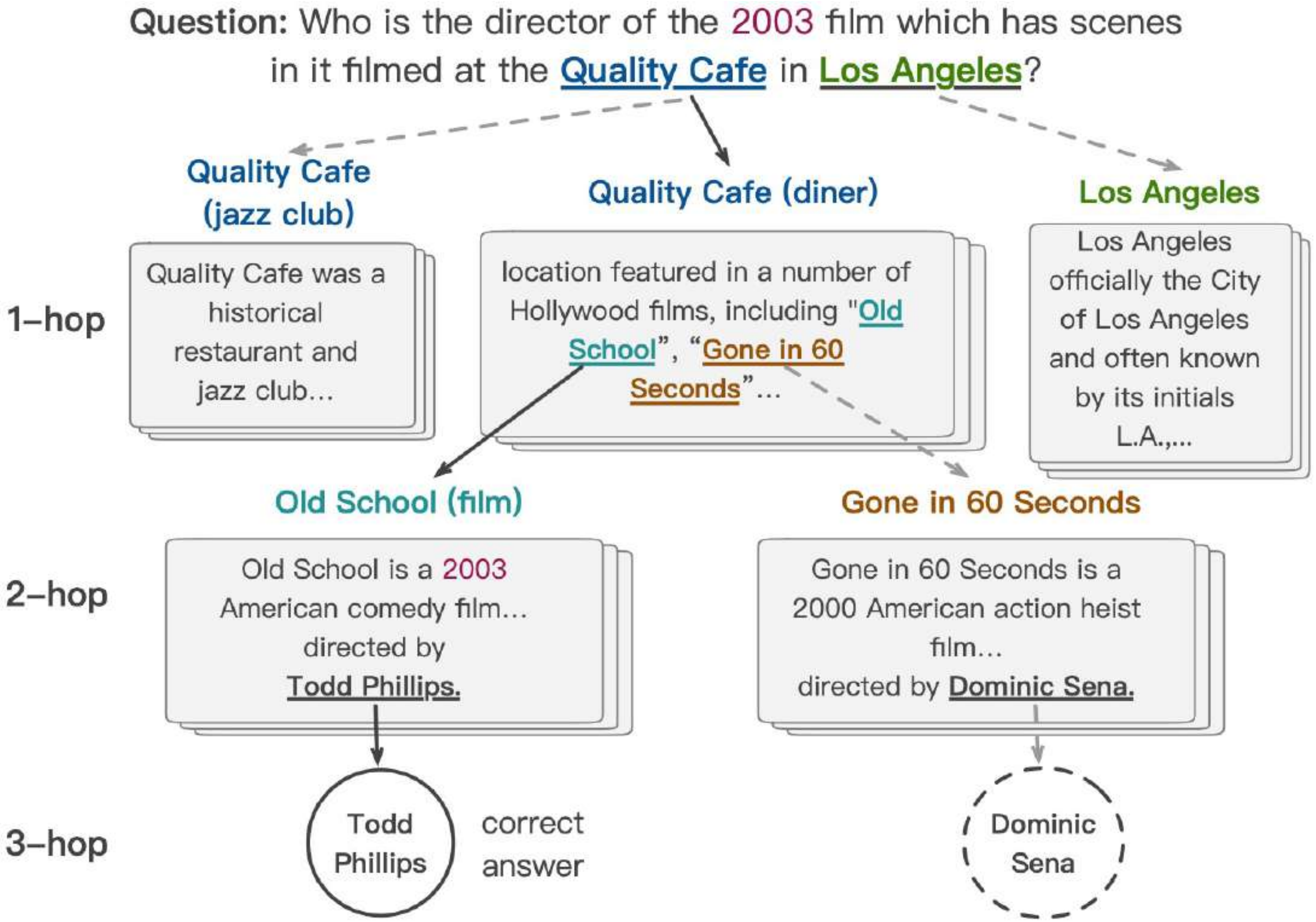


Static Graph Construction: SQL Graph



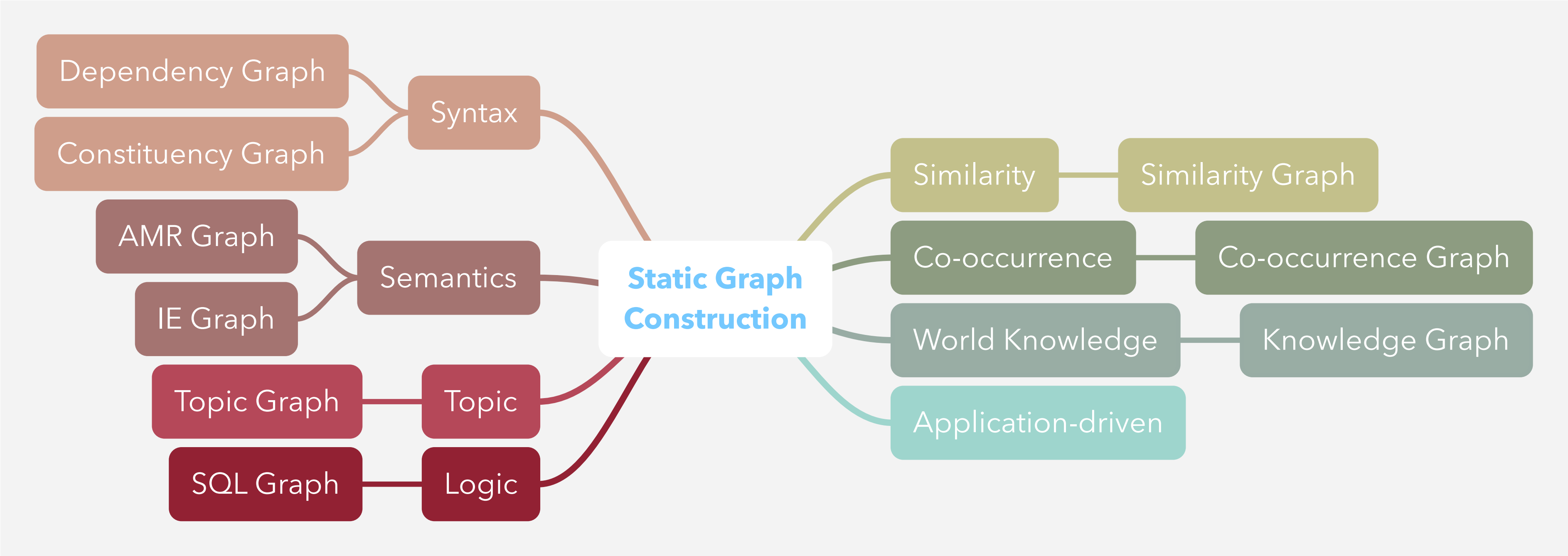
SQL query input: ***SELECT*** *company* ***WHERE*** *assets* $>$ *val*$_0$ ***AND*** *sales* $>$ *val*$_0$ ***AND*** *industry_rank* $\leq$ *val*$_1$

Static Graph Construction: Application-driven Graph



Ming Ding et al. "Cognitive Graph for Multi-Hop Reading Comprehension at Scale". ACL 2019.

Static Graph Construction: Summary

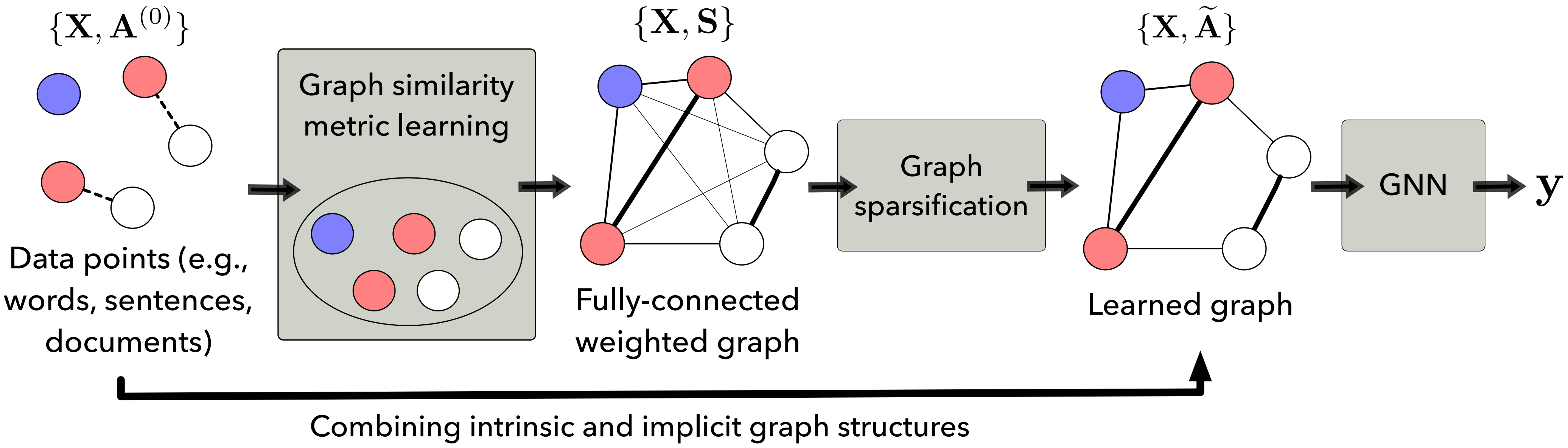


Widely used in various NLP applications such as NLG, MRC, semantic parsing, etc.

Dynamic Graph Construction

- Problem setting:
  - Input: raw text (e.g., sentence, paragraph, document, corpus)
  - Output: graph
- Graph structure (adjacency matrix) learning on the fly, joint with graph representation learning

Dynamic Graph Construction: Overview

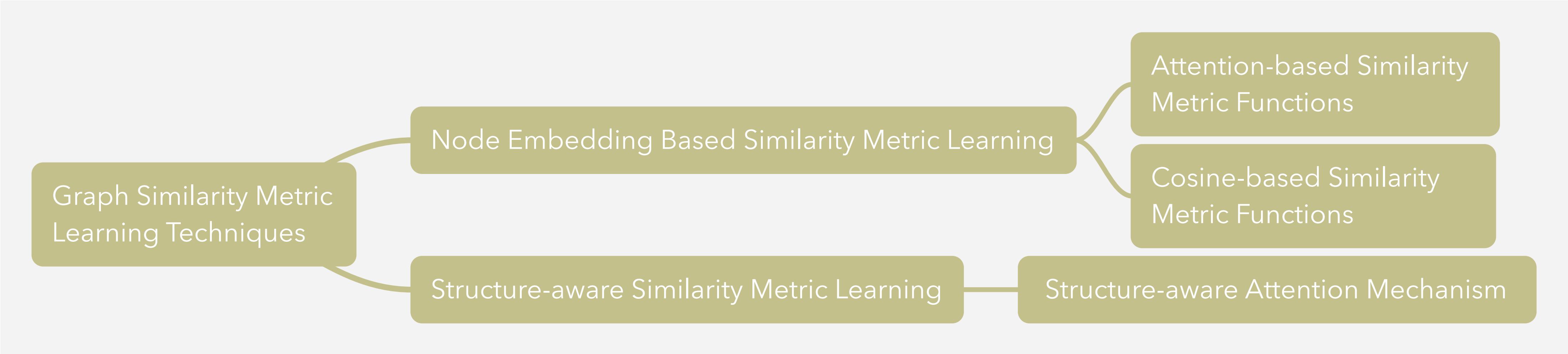


Dynamic Graph Construction Outline

- **Dynamic Graph Construction**
  - Graph Similarity Metric Learning Techniques
  - Graph Sparsification Techniques
  - Combining Intrinsic Graph Structures and Implicit Graph Structures
  - Learning Paradigms

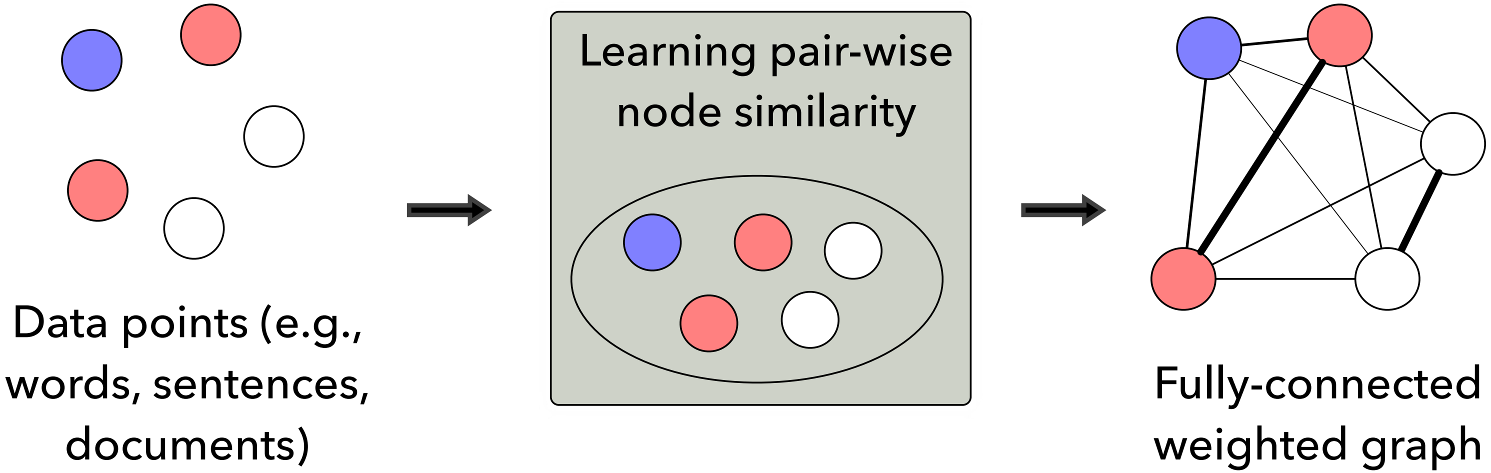
Graph Similarity Metric Learning Techniques

- Graph structure learning as similarity metric learning (in the node embedding space)
- Enabling inductive learning
- Various metric functions



Node Embedding Based Similarity Metric Learning

- Learning a weighted adjacency matrix by computing the pair-wise node similarity in the embedding space
- Common metrics functions
  - Attention-based similarity metric functions
  - Cosine-based similarity metric functions



Attention-based Similarity Metric Functions

Variant 1)

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

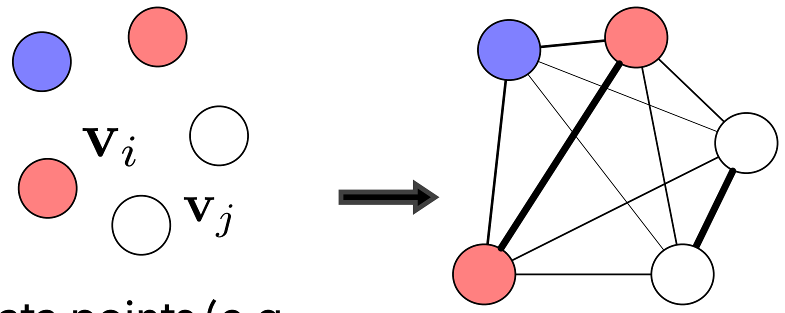
Node feature vector

Non-negative learnable weight vector

Variant 2)

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

Learnable weight matrix



Data points (e.g., words, sentences, documents)

Fully-connected weighted graph

Chen at al. "GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension". IJCAI 2020.

Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.

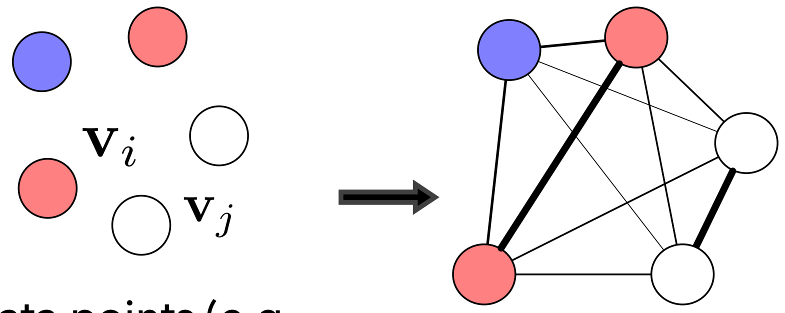
Cosine-based Similarity Metric Functions

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

Learnable weight vector

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

Multi-head similarity scores



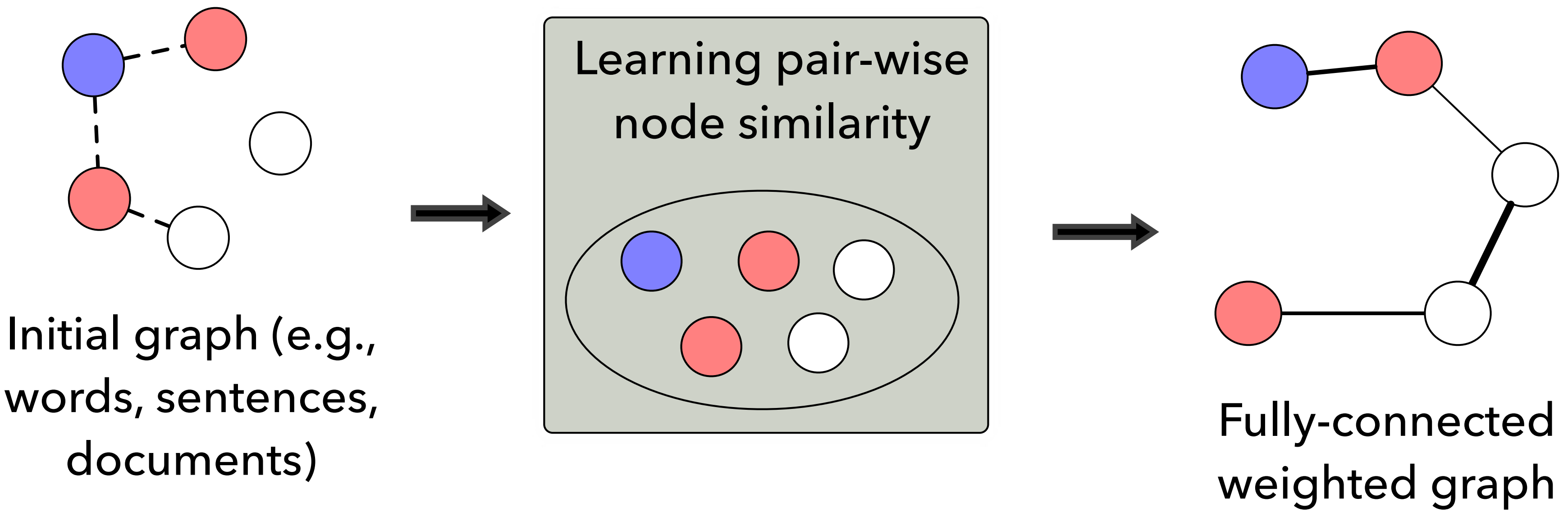
Data points (e.g., words, sentences, documents)

Fully-connected weighted graph

Chen et al. "Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings". NeurIPS 2021.

Structure-aware Similarity Metric Learning

- Learning a weighted adjacency matrix by computing the pair-wise node similarity in the embedding space
- Considering existing edge information of the intrinsic graph in addition to the node information



Attention-based Similarity Metric Functions

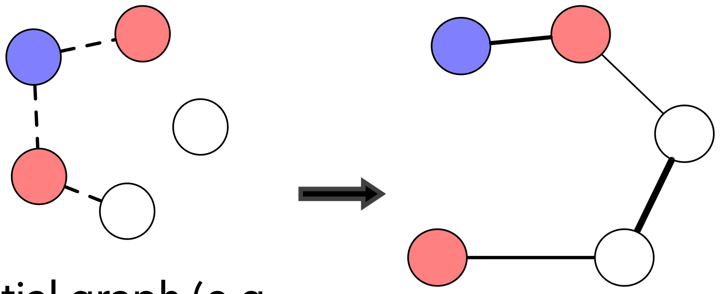
Variant 1)

$$S_{i,j}^{l} = \text{softmax}(\mathbf{u}^{T}\tanh(\mathbf{W}[\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{v}_i, \mathbf{v}_j, \mathbf{e}_{i,j}]))$$

Edge embeddings

Variant 2)

$$S_{i,j} = \frac{\text{ReLU}(\mathbf{W}^{Q}\mathbf{v}_i)^{T}(\text{ReLU}(\mathbf{W}^{K}\mathbf{v}_i) + \text{ReLU}(\mathbf{W}^{R}\mathbf{e}_{i,j}))}{\sqrt{d}}$$



Initial graph (e.g., words, sentences, documents)

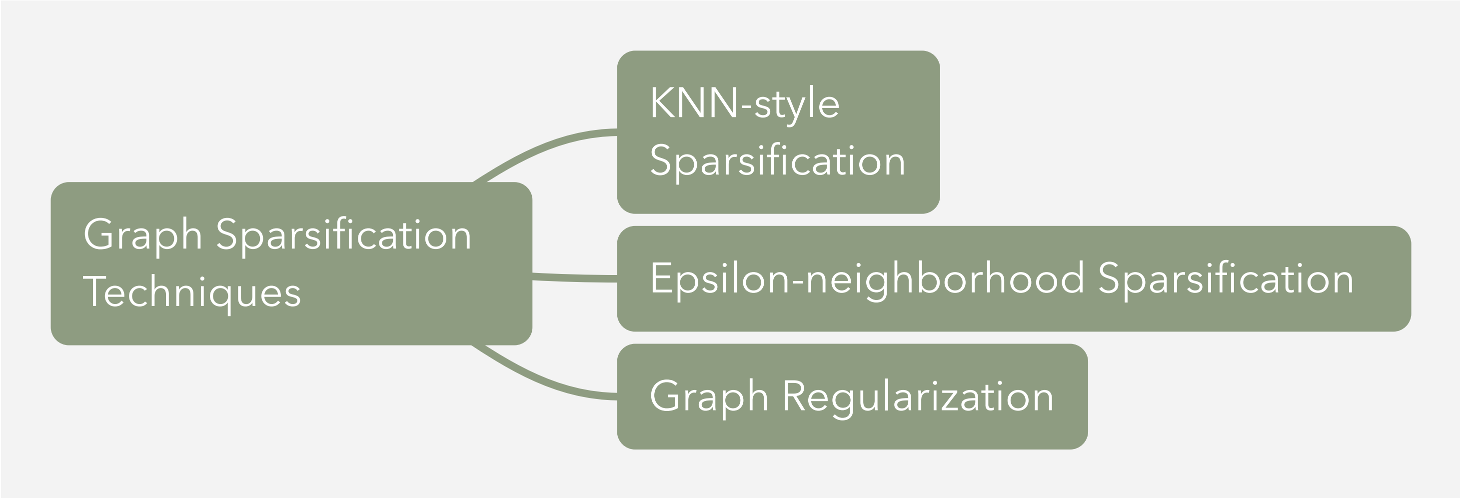
Fully-connected weighted graph

Liu et al. "Contextualized Non-local Neural Networks for Sequence Learning". AAAI 2019.

Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

Graph Sparsification Techniques

- Similarity metric functions learn a fully-connected graph
- Fully-connected graph is computationally expensive and might introduce noise
- Enforcing sparsity to the learned graph structure
- Various techniques



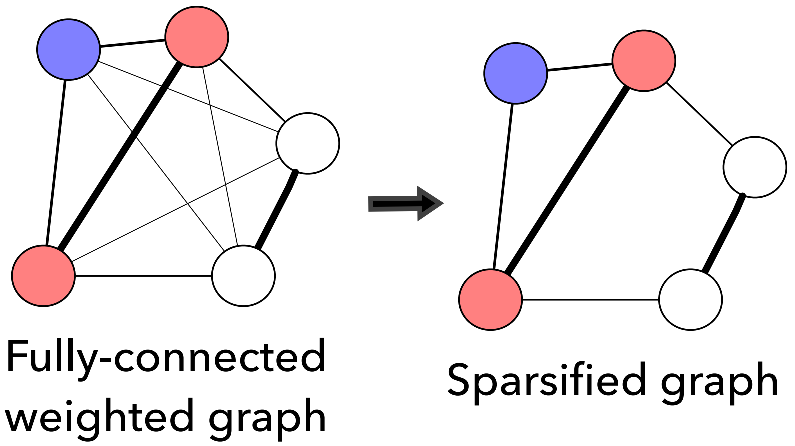
Common Graph Sparsification Options

Option 1) KNN-style Sparsification

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

Option 2) epsilon-neighborhood Sparsification

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



Option 3) graph Regularization

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

Combining Intrinsic and Implicit Graph Structures

- Intrinsic graph typically still carries rich and useful information
- Learned implicit graph is potentially a “shift” (e.g., substructures) from the intrinsic graph structure

$$\widetilde{A} = \lambda L^{(0)} + (1 - \lambda)\mathrm{f}(A)$$

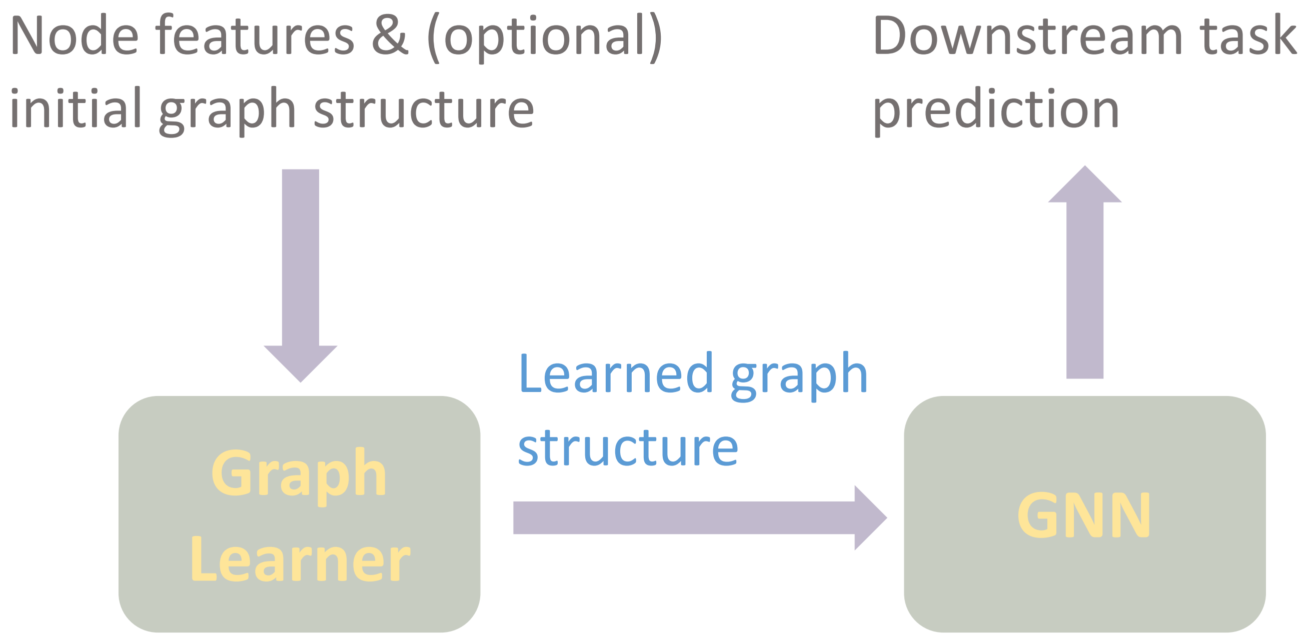
Normalized graph Laplacian

f(A) can be arbitrary operation, e.g., graph Laplacian, row-normalization

Li et al. “Adaptive Graph Convolutional Neural Networks”. AAAI 2018.

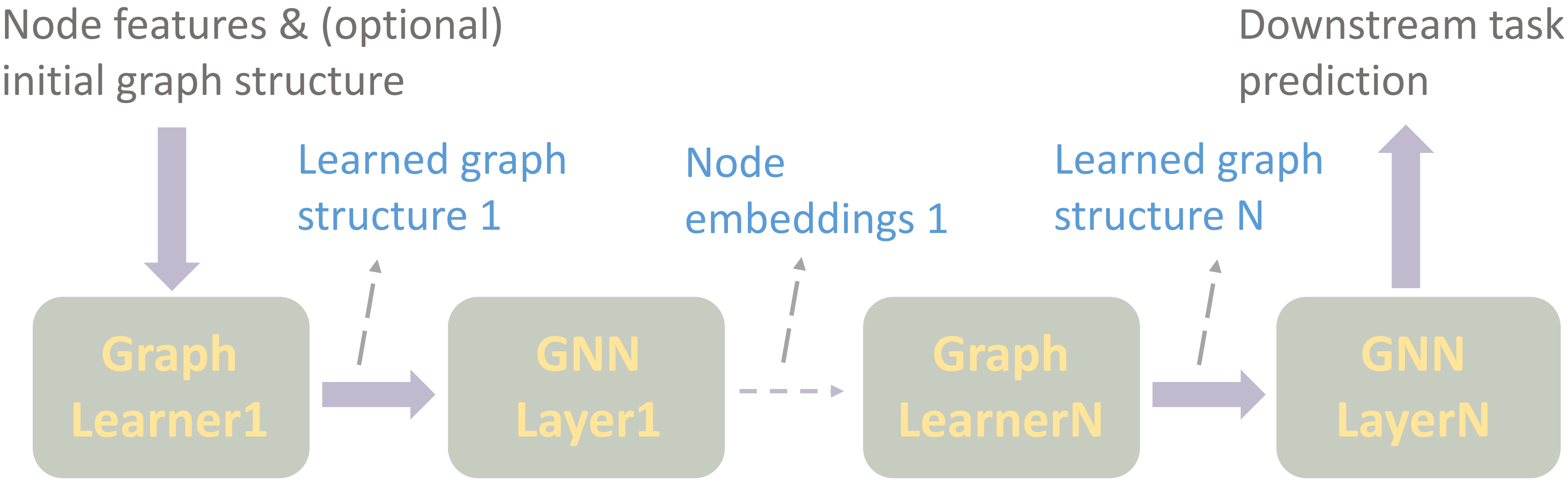
Chen et al. “Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings”. NeurIPS 2021.

Learning Paradigms: Joint Learning



Chen at al. "GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension". IJCAI 2020.
Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.
Liu et al. "Contextualized Non-local Neural Networks for Sequence Learning". AAAI 2019.
Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

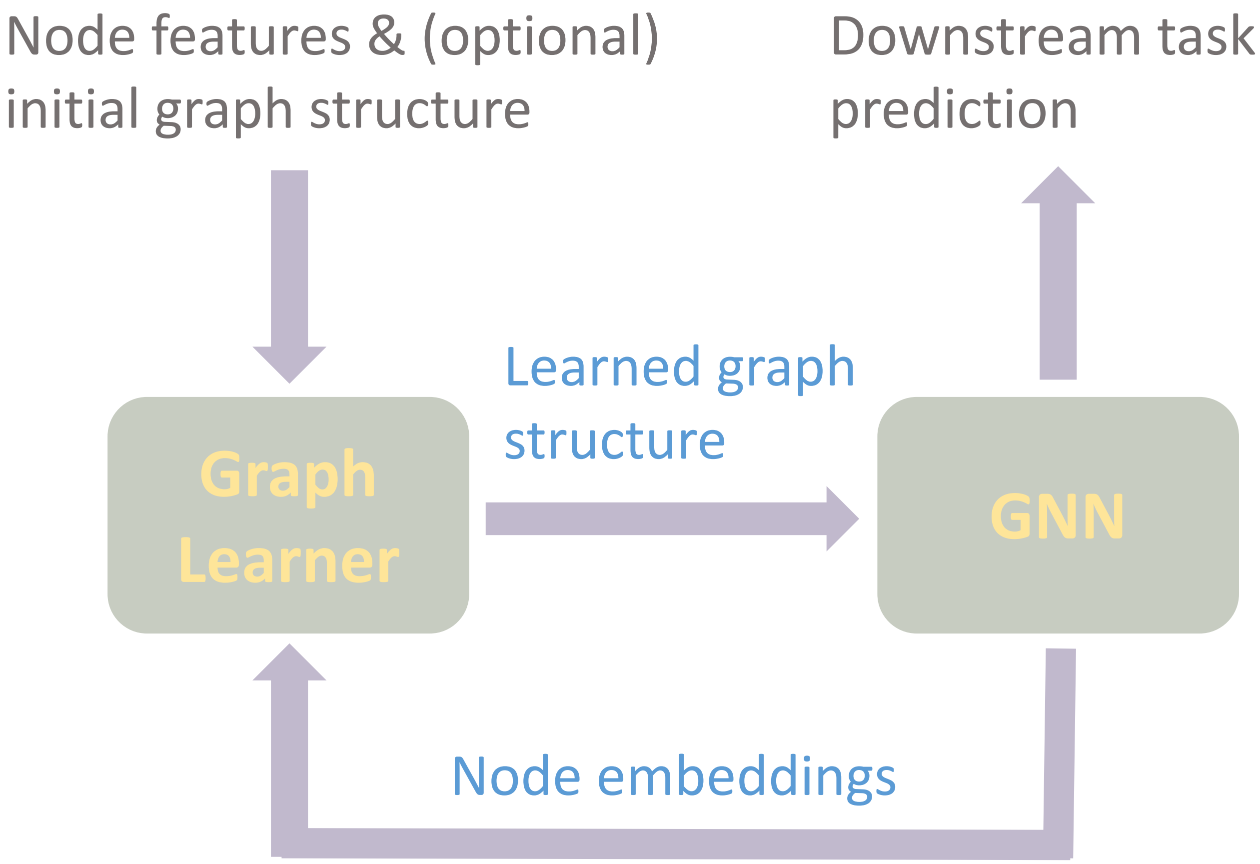
Learning Paradigms: Adaptive Learning



Repeat for fixed num. of stacked GNN layers

Li et al. "Adaptive Graph Convolutional Neural Networks". AAAI 2018.

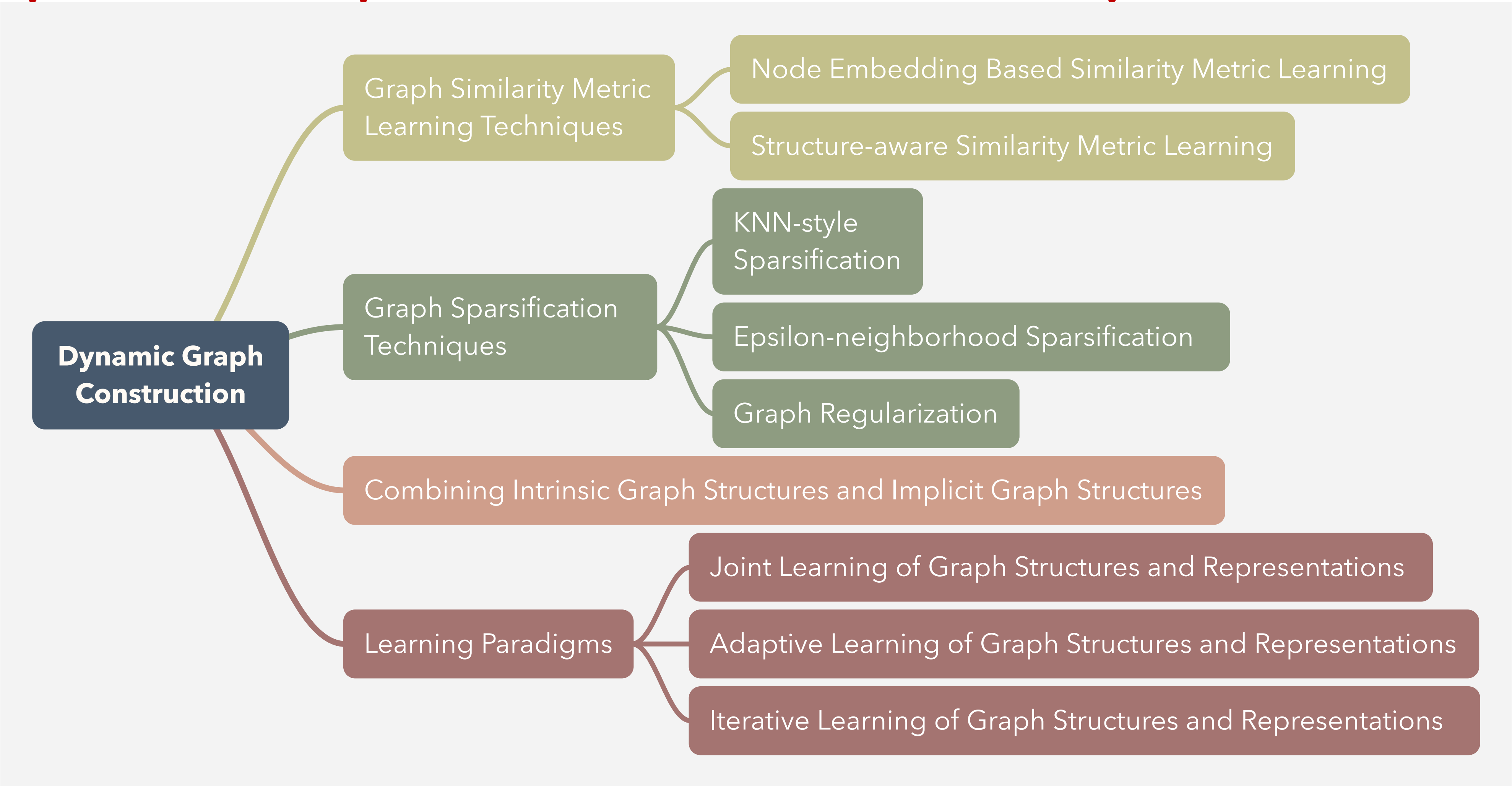
Learning Paradigms: Iterative Learning



Repeat until condition satisfied

Chen et al. "Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings". NeurIPS 2021.

Dynamic Graph Construction Summary



Graph4NLP

Static vs. Dynamic Graph Construction

New topic in DLG4NLP!

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

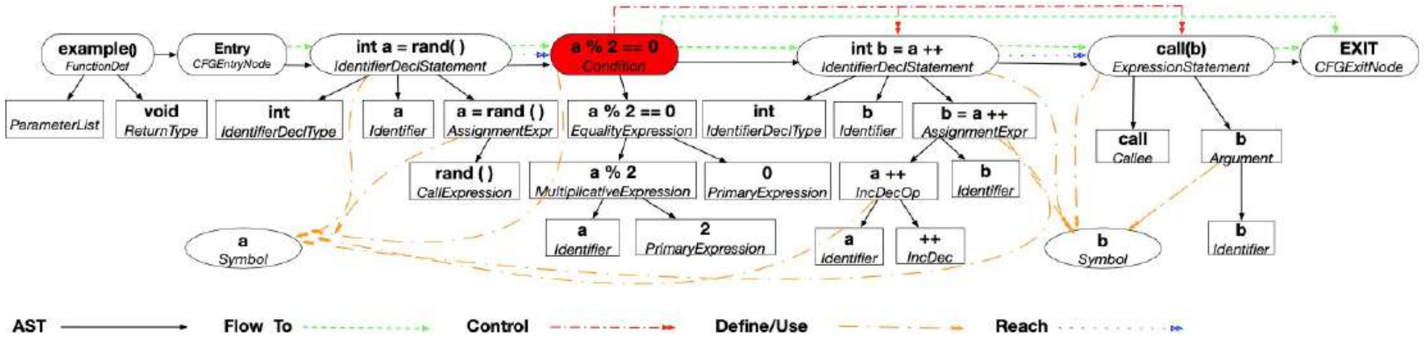
Static vs. Dynamic Graph Construction (cont)

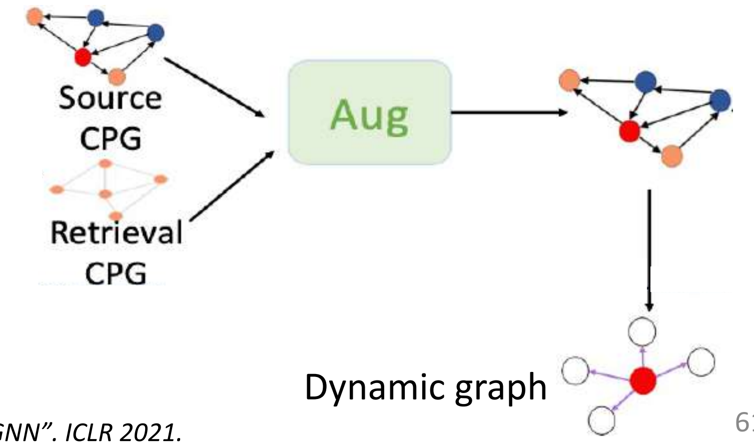
When to use static graph construction

- Domain knowledge which fits the task and can be presented as a graph

When to use dynamic graph construction

- Lack of domain knowledge which fits the task or can be presented as a graph
- Domain knowledge is incomplete or might contain noise
- To learn implicit graph which augments the static graph

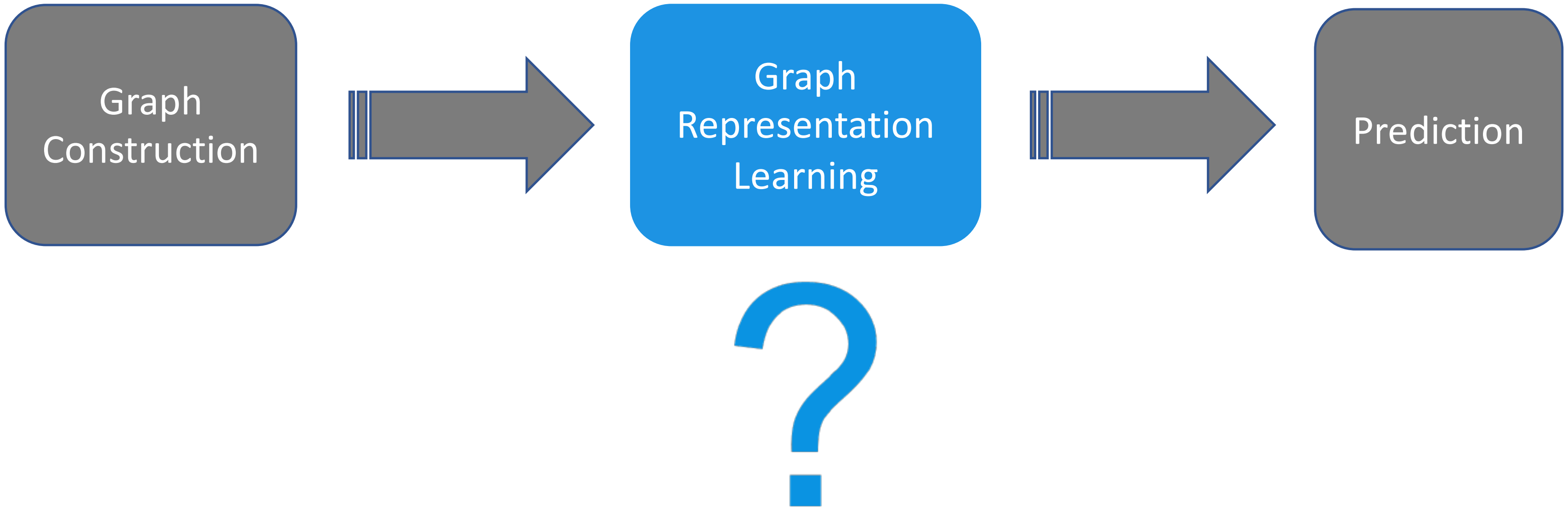




Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

Graph Representation Learning for NLP

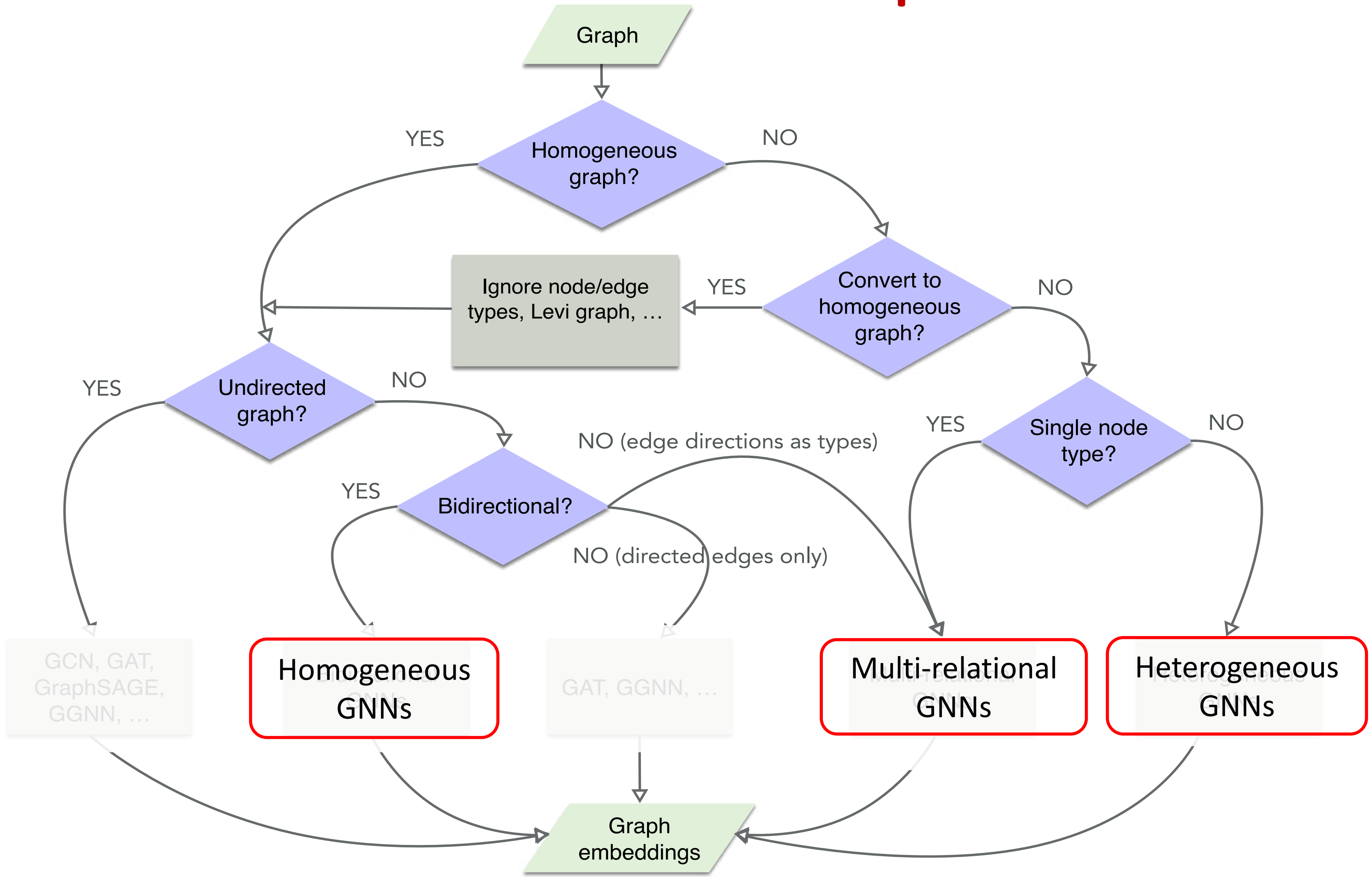
GNNs for Graph Representation Learning



Homogeneous vs Multi-relational vs Heterogeneous Graphs

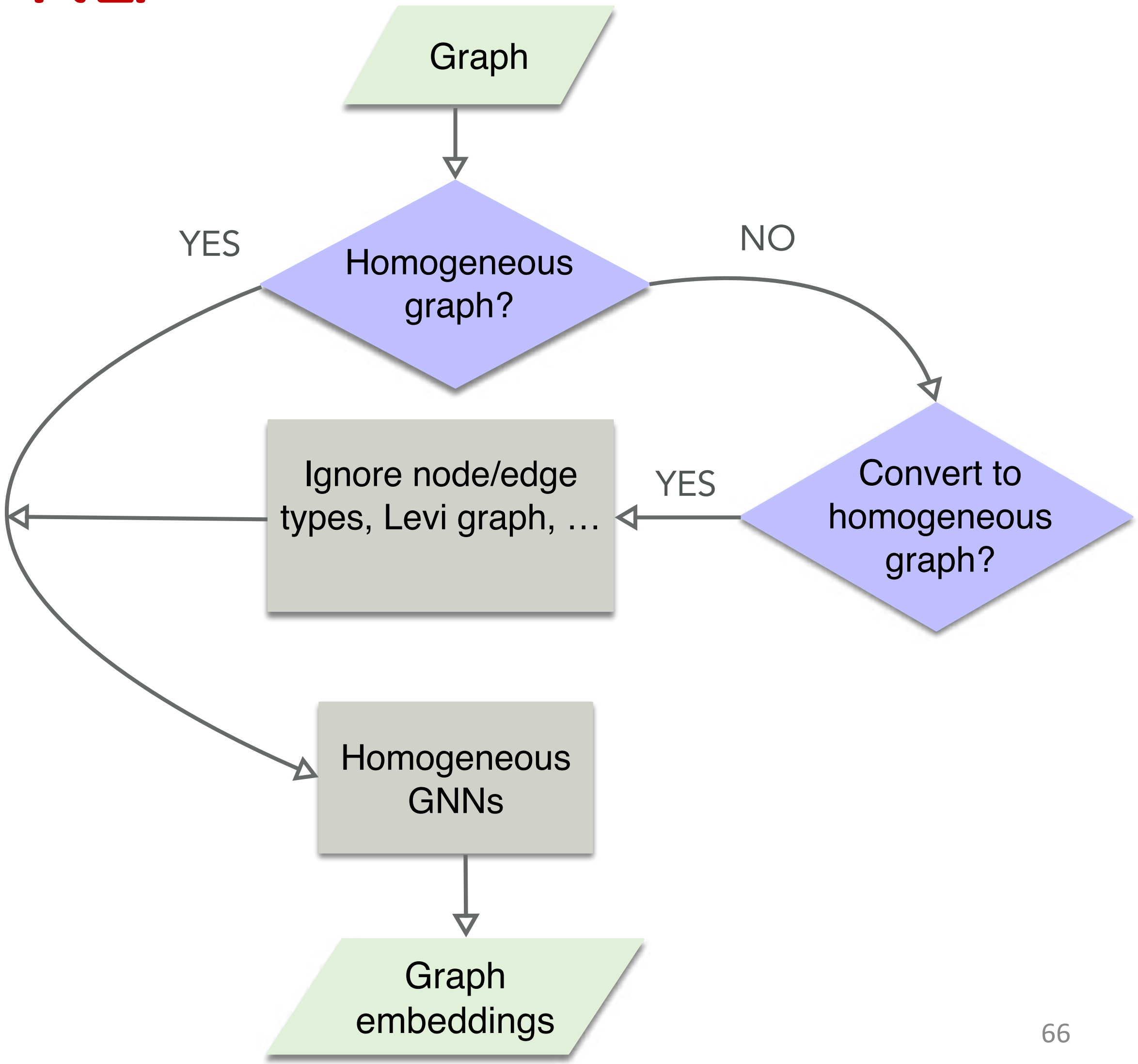
Graph types	Homogeneous	Multi-relational	Heterogeneous
# of node types	1	1	> 1
# of edge types	1	> 1	>= 1

Which GNNs to Use Given a Graph?

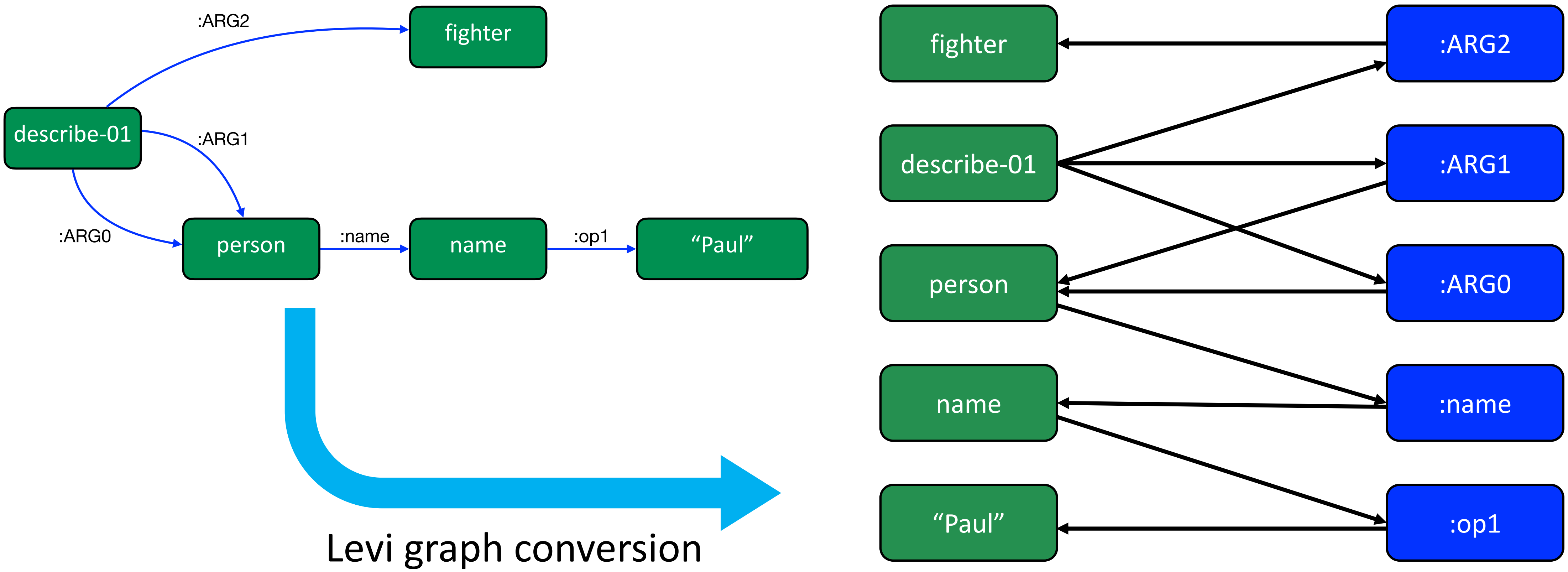


Homogeneous GNNs for NLP

- When to use homogeneous GNNs?
- Homogeneous GNNs
  - GCN
  - GAT
  - GraphSAGE
  - GGNN
  - …



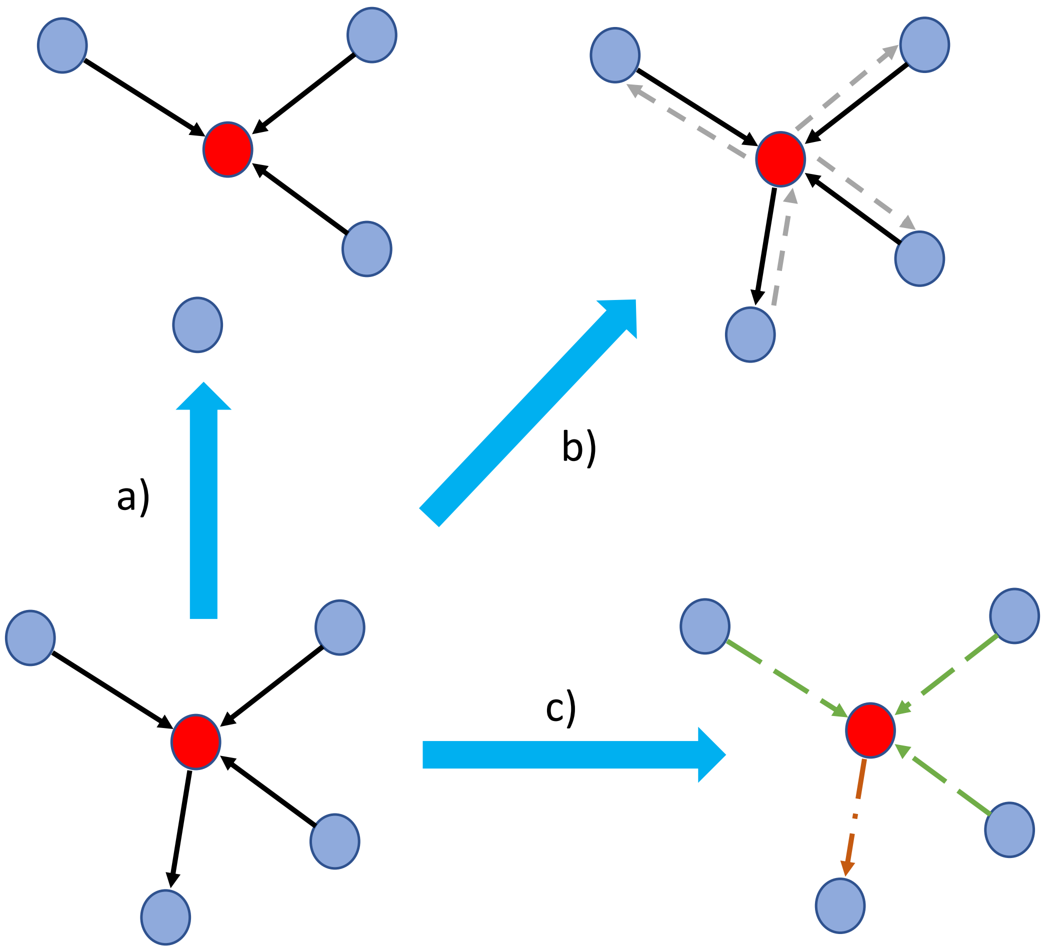
Non-homogeneous to Homogeneous Conversion via Levi Graph



Levi graph: edges as new nodes

How to Handle Edge Direction Information?

- Edge direction is important (think about BiLSTM, BERT)
- Common strategies for handling directed graphs
  a) Message passing only along directed edges (e.g., GAT, GGNN)
  b) Regarding edge directions as edge types (i.e., adding “reverse” edges)
  c) Bidirectional GNNs



Edge Directions as Edge Types

- Regarding edge directions as edge types, resulting in a multi-relational graph

$$dir_{i,j} = \begin{cases} default, & e_{i,j} \text{ is originally existing in the graph} \\ inverse, & e_{i,j} \text{ is the inverse edge} \\ self, & i = j \end{cases}$$

Then we can apply multi-relational GNNs

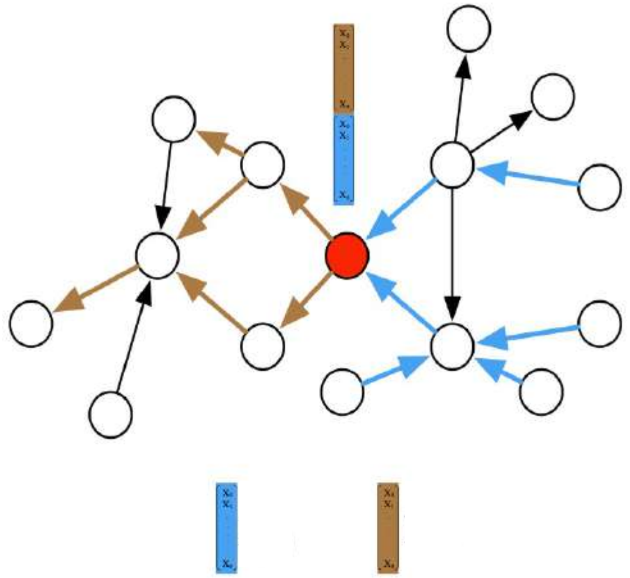
Bidirectional GNNs for Directed Graphs

Bi-Sep GNNs formulation:

Run multi-hop backward/forward GNN on the graph

$$\mathbf{h}_{i,\dashv}^{k} = GNN(\mathbf{h}_{i,\dashv}^{k-1}, \{\mathbf{h}_{j,\dashv}^{k-1} : \forall v_j \in \mathcal{N}_{\dashv}(v_i)\})$$

$$\mathbf{h}_{i,\vdash}^{k} = GNN(\mathbf{h}_{i,\vdash}^{k-1}, \{\mathbf{h}_{j,\vdash}^{k-1} : \forall v_j \in \mathcal{N}_{\vdash}(v_i)\})$$



Concatenate backward/forward node embeddings at last hop

$$\mathbf{h}_{i}^{K} = \mathbf{h}_{i,\dashv}^{K} || \mathbf{h}_{i,\vdash}^{K}$$

Xu et al. "Graph2Seq: Graph to Sequence Learning with Attention-based Neural Networks". 2018.

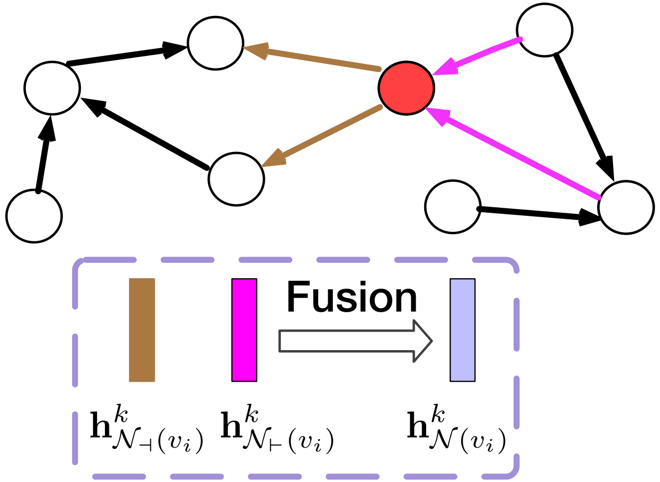
Bidirectional GNNs for Directed Graphs (cont)

Bi-Fuse GNNs formulation:

Run one-hop backward/forward node aggregation

$$\mathbf{h}^k_{\mathcal{N}_{\dashv}(v_i)} = AGG(\mathbf{h}^{k-1}_i, \{\mathbf{h}^{k-1}_j : \forall v_j \in \mathcal{N}_{\dashv}(v_i)\})$$

$$\mathbf{h}^k_{\mathcal{N}_{\vdash}(v_i)} = AGG(\mathbf{h}^{k-1}_i, \{\mathbf{h}^{k-1}_j : \forall v_j \in \mathcal{N}_{\vdash}(v_i)\})$$



Fuse backward/forward aggregation vectors at each hop

$$\mathbf{h}^k_{\mathcal{N}(v_i)} = Fuse(\mathbf{h}^k_{\mathcal{N}_{\dashv}(v_i)}, \mathbf{h}^k_{\mathcal{N}_{\vdash}(v_i)})$$

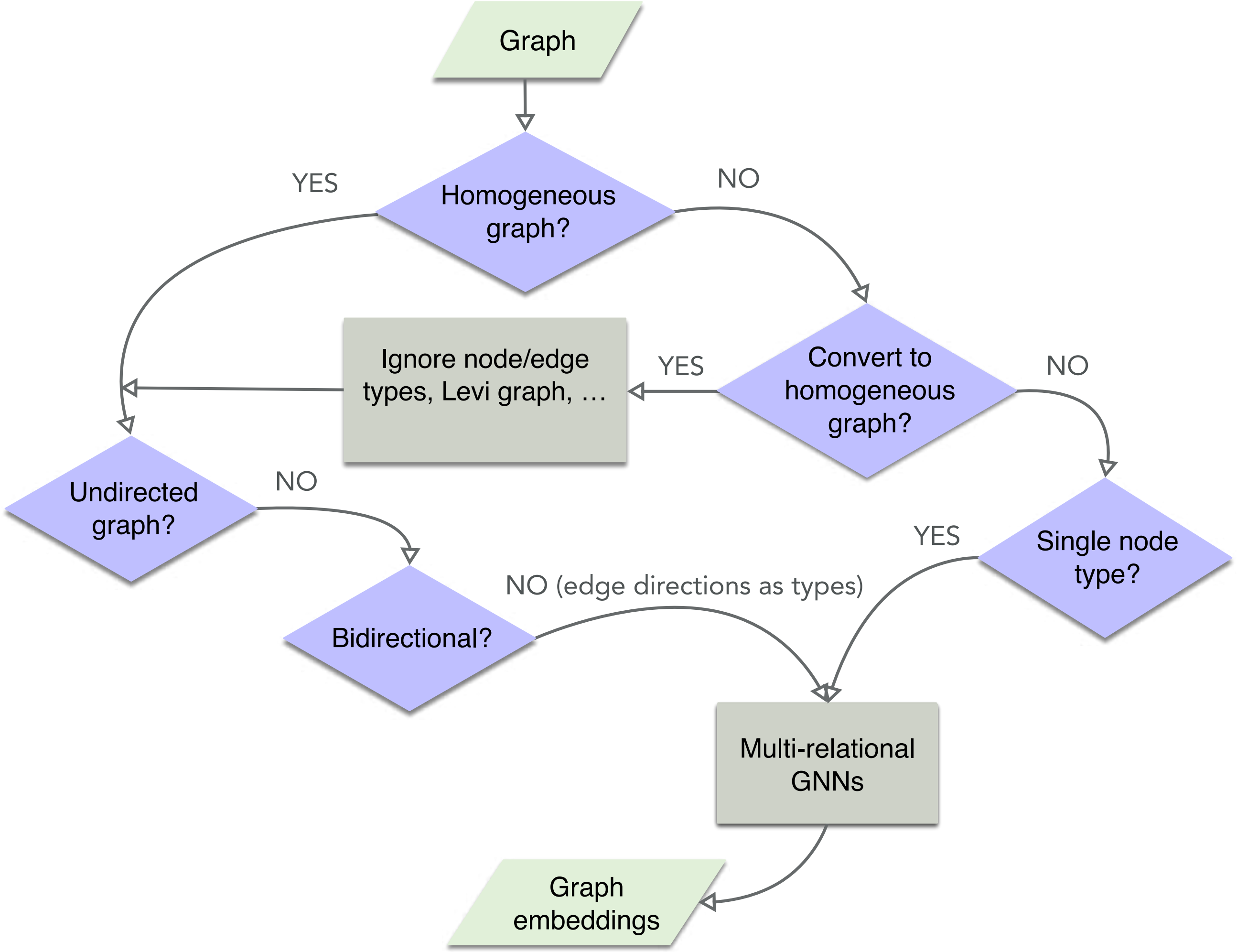
Update node embeddings with fused aggregation vectors at each hop

$$\mathbf{h}^k_i = \sigma(\mathbf{h}^{k-1}_i, \mathbf{h}^k_{\mathcal{N}(v_i)})$$

Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.

Multi-relational GNNs for NLP

- When to use multi-relational GNNs?
- Multi-relational GNNs
  a) Including relation-specific transformation parameters in GNN
  b) Including edge embeddings in GNN
  c) Multi-relational Graph Transformers



R-GNN: Overview

$$\mathbf{h}_i^k = \sigma(\mathbf{h}_i^{k-1}, \sum_{v_j \in \mathcal{N}(v_i)} AGG(\mathbf{h}_j^{k-1}, \theta^k))$$

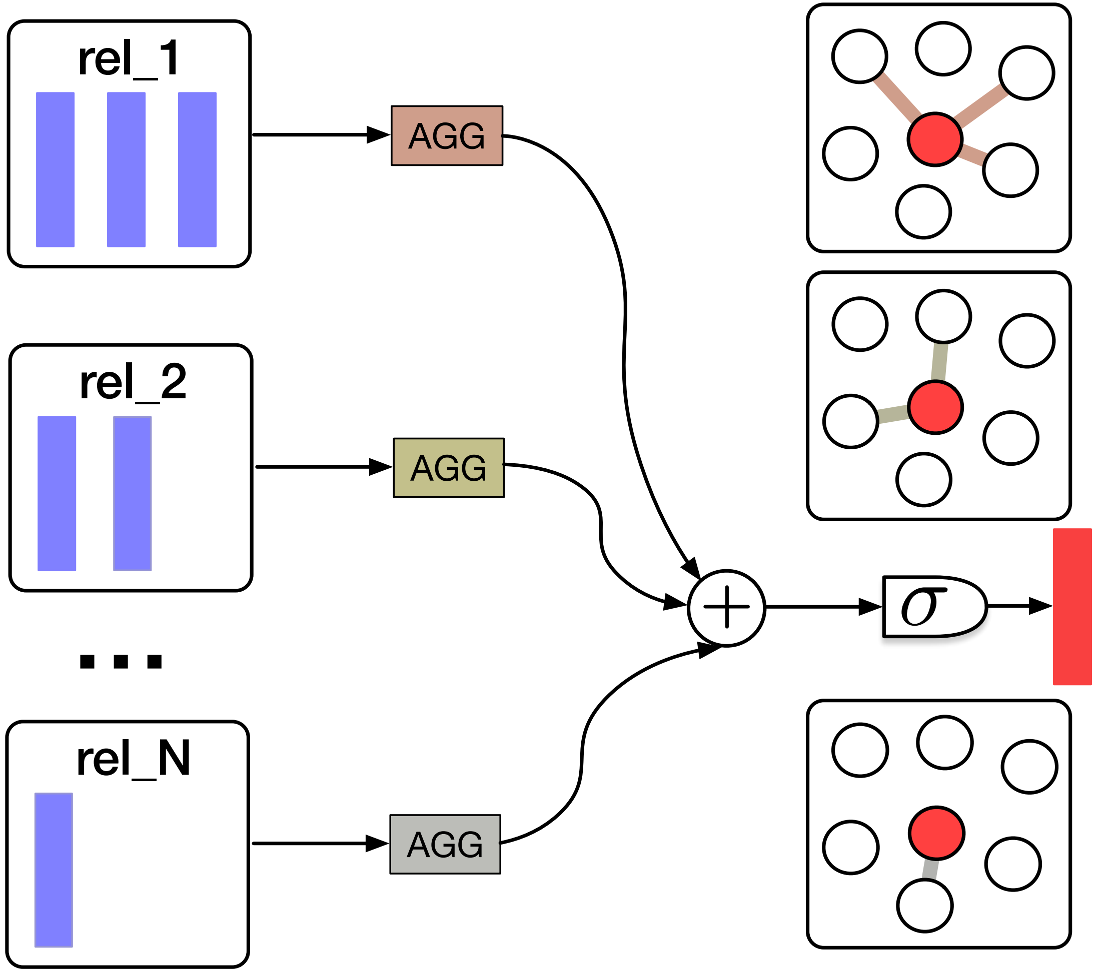
GNN

1) relation-specific transformation, e.g., node feature transformation, attention weight …

R-GNN

$$\mathbf{h}_i^k = \sigma(\mathbf{h}_i^{k-1}, \sum_{r \in \mathcal{E}} \sum_{v_j \in \mathcal{N}_r(v_i)} AGG(\mathbf{h}_j^{k-1}, \theta_r^k))$$

2) aggregation per relation-specific subgraph



R-GNN Variant: R-GCN

- Relation-specific node feature transformation during neighborhood aggregation

$$\mathbf{h}_i^k = \sigma\Big(\sum_{r\in\mathcal{E}} \sum_{v_j\in\mathcal{N}_r(v_i)} \frac{1}{c_{i,r}} \mathbf{W}_r^k \mathbf{h}_j^{k-1} + \mathbf{W}_0^k \mathbf{h}_i^{k-1}\Big), \quad c_{i,r} = |\mathcal{N}_r(v_i)|$$

Relation-specific d x d learnable weight matrix

Schlichtkrull et al. "Modeling Relational Data with Graph Convolutional Networks". 2017.

R-GNN: Avoiding Over-parameterization

Learning d x d transformation weight matrix for each relation is expensive!

O(Rd^2) parameters every GNN layer where R is the num of relation types

How to avoid over-parameterization?

Option 1) basis decomposition - linear hypothesis

$$\theta_r^k = \sum_{b=1}^{B} a_{rb}^k \mathbf{V}_b^k, \quad \mathbf{V}_b^{(k)} \in \mathbb{R}^{d \times d}$$

O(RB + Bd^2) parameters

Basis matrices

Option 2) block-diagonal decomposition - sparsity hypothesis

$$\theta_r^k = \bigoplus_{b=1}^{B} \mathbf{Q}_{br}^k = diag(\mathbf{Q}_{1r}^k, \mathbf{Q}_{2r}^k, ..., \mathbf{Q}_{Br}^k), \quad \mathbf{Q}_{br}^{(k)} \in \mathbb{R}^{d/B \times d/B}$$

O(Rd^2/B) parameters

Submatrices

Including Edge Embeddings in GNNs

Variant 1) Include edge embeddings in message passing

$$\mathbf{h}_i^k = \sigma(\mathbf{h}_i^{k-1}, \sum_{v_j \in \mathcal{N}(v_i)} AGG(\mathbf{h}_j^{k-1}, \mathbf{e}_{i,j}, \theta^k))$$

Edge embeddings

Variant 2) Update edge embedding in message passing

$$\mathbf{h}_i^k = \sigma(\mathbf{h}_i^{k-1}, \sum_{v_j \in \mathcal{N}(v_i)} AGG(\mathbf{h}_j^{k-1}, \mathbf{e}_{i,j}^{k-1}, \theta^k)), \quad \mathbf{e}_{i,j}^k = f(\mathbf{e}_{i,j}^{k-1}, \theta_{rel}^k)$$

Update edge embeddings

Chen et al. "Toward Subgraph Guided Knowledge Graph Question Generation with Graph Neural Networks". 2020.

Vashishth et al. "Composition-based Multi-Relational Graph Convolutional Networks". ICLR 2020.

Multi-relational Graph Transformers

- Transformers as a special class of GNNs which
  - jointly learn and encode a fully-connected graph via self-attention
  - share many similarities with GAT
  - fail to effectively handle arbitrary graph-structured data
    - e.g., position embeddings for sequential data, removing position embeddings for set
- Multi-relational graph transformers
  - employed with structure-aware self-attention
  - respect various relation types

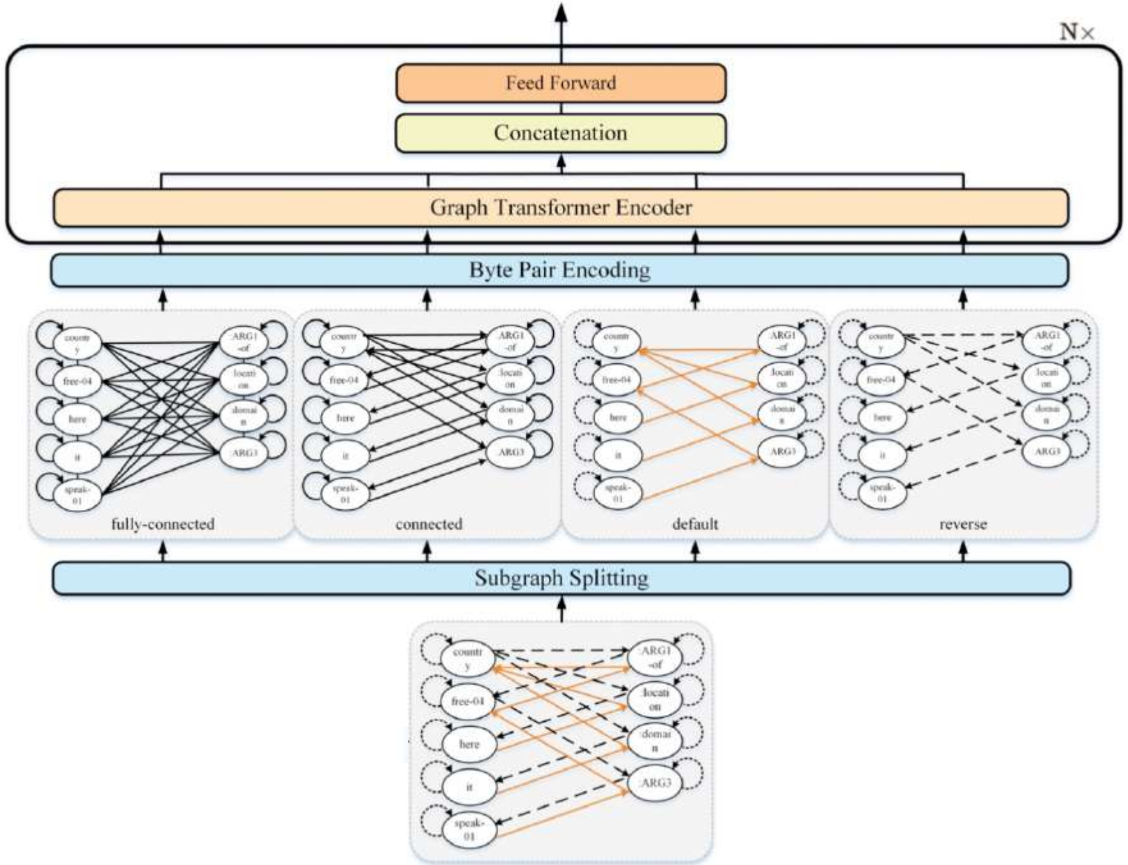
R-GAT based Graph Transformers

GAT-like masked attention

$$\mathbf{z}_i^{r,k} = \sum_{v_j \in \mathcal{N}_r(v_i)} \alpha_{i,j}^k \mathbf{W}_V^k \mathbf{h}_j^{k-1}, r \in \mathcal{E}$$

$$\mathbf{h}_i^k = \mathrm{FFN}^k(\mathbf{W}_O^k[\mathbf{z}_i^{R_1,k}, \ldots, \mathbf{z}_i^{R_m,k}])$$

Relation-specific learnable weight matrix



Yao et al. “Heterogeneous Graph Transformer for Graph-to-Sequence Learning”. ACL 2020.

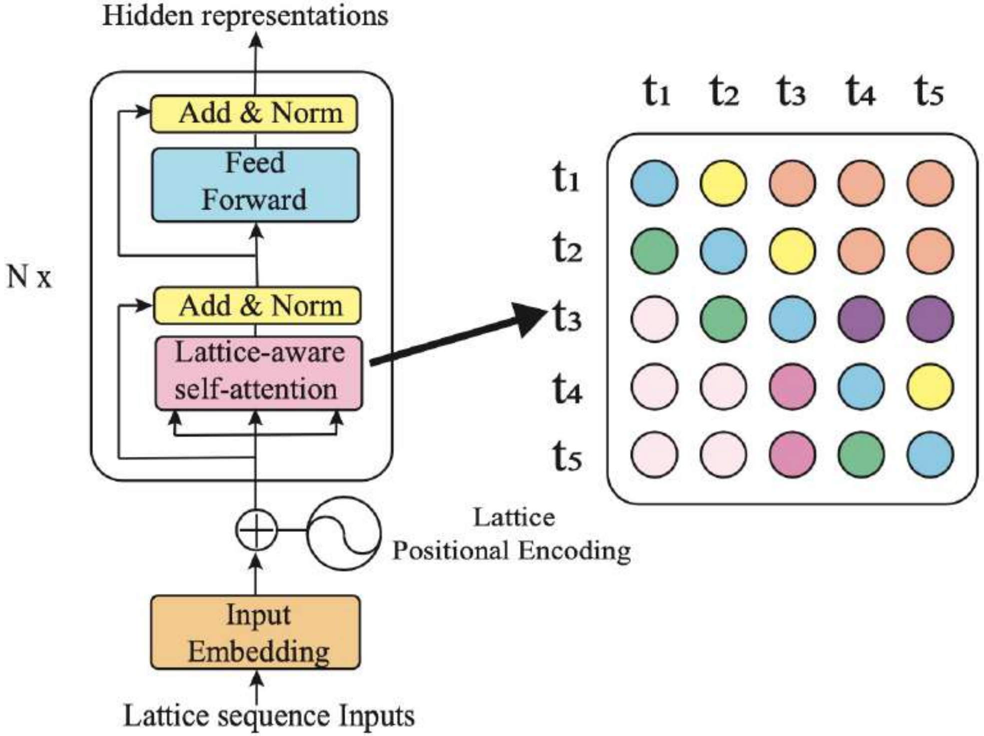
Structure-aware Self-attention based Graph Transformers

$$\mathbf{h}_i^k = \sum_j \alpha_{i,j}^k (\mathbf{W}_V^k \mathbf{h}_j^{k-1} + \mathbf{W}_F^k \mathbf{e}_{i,j})$$

Edge embeddings

$$\alpha_{i,j}^k = softmax(u_{i,j}^k)$$

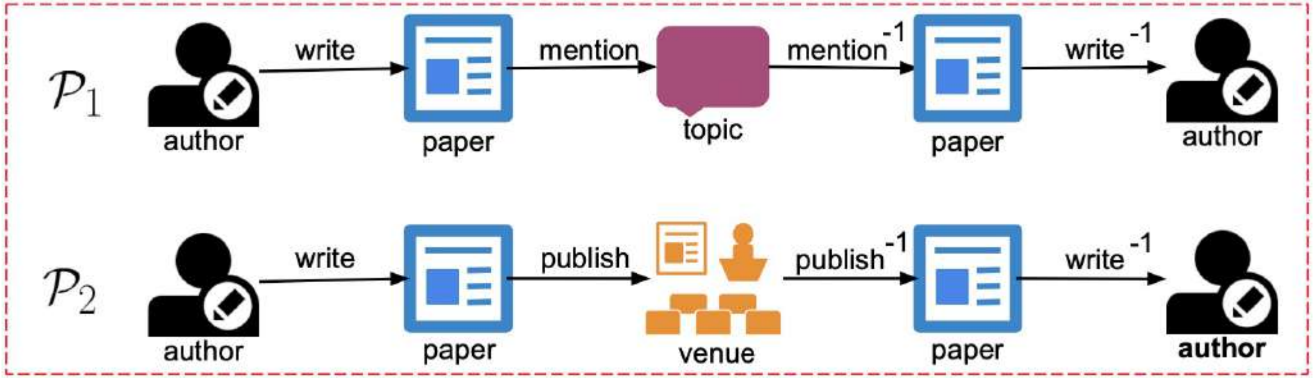
$$u_{i,j}^k = \frac{(\mathbf{W}_Q^k \mathbf{h}_i^{k-1})^T (\mathbf{W}_K^k \mathbf{h}_j^{k-1} + \mathbf{W}_R^k \mathbf{e}_{i,j})}{\sqrt{d}}$$



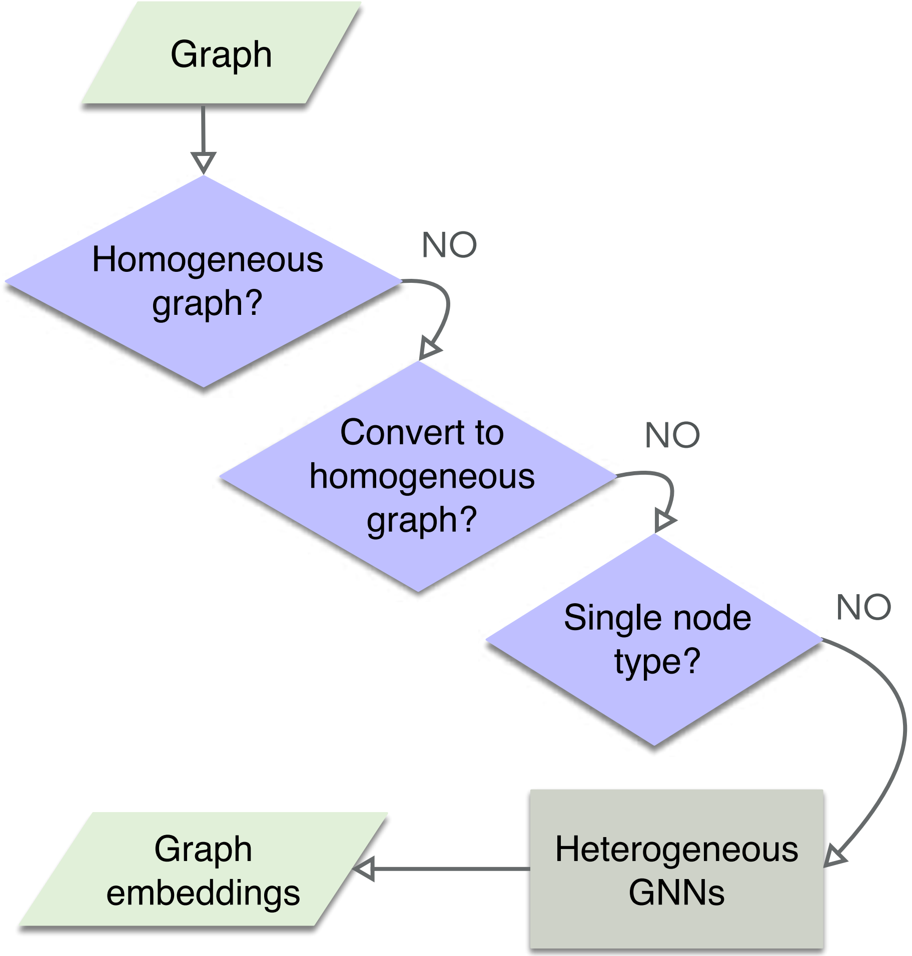
Xiao et al. "Lattice-Based Transformer Encoder for Neural Machine Translation". ACL 2019.
Zhu et al. "Modeling Graph Structure in Transformer for Better AMR-to-Text Generation". EMNLP 2019.

Heterogeneous GNNs

- When to use Heterogeneous GNNs?
- Heterogeneous GNNs
  a) Meta-path based Heterogeneous GNNs



Meta paths among author nodes



Meta-path based Heterogeneous GNN example: HAN

Step 1) type-specific node feature transformation

$$\mathbf{h}_i = \mathbf{W}_{\tau(v_i)} \mathbf{v}_i$$

Node-type specific learnable weight matrix

Step 2) node-level aggregation along each meta path

$$\mathbf{z}_{i,\Phi_k} = \sigma\Big(\sum_{v_j \in \mathcal{N}_{\Phi_k}(v_i)} \alpha_{i,j}^{\Phi_k} \mathbf{h}_j \Big)$$

Aggregate over neighboring nodes in k-length meta path

Step 3) meta-path level aggregation

$$\mathbf{z}_i = \sum_{k=1}^{p} \beta_{\Phi_k} \mathbf{z}_{i,\Phi_k}$$

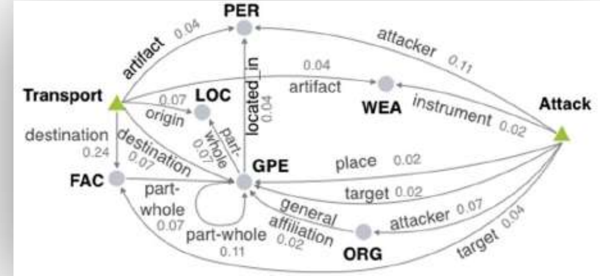
Attention weights over meta paths

Wang et al. "Heterogeneous Graph Neural Networks for Extractive Document Summarization". ACL 2020.

Graph Encoder-Decoder Models for NLP

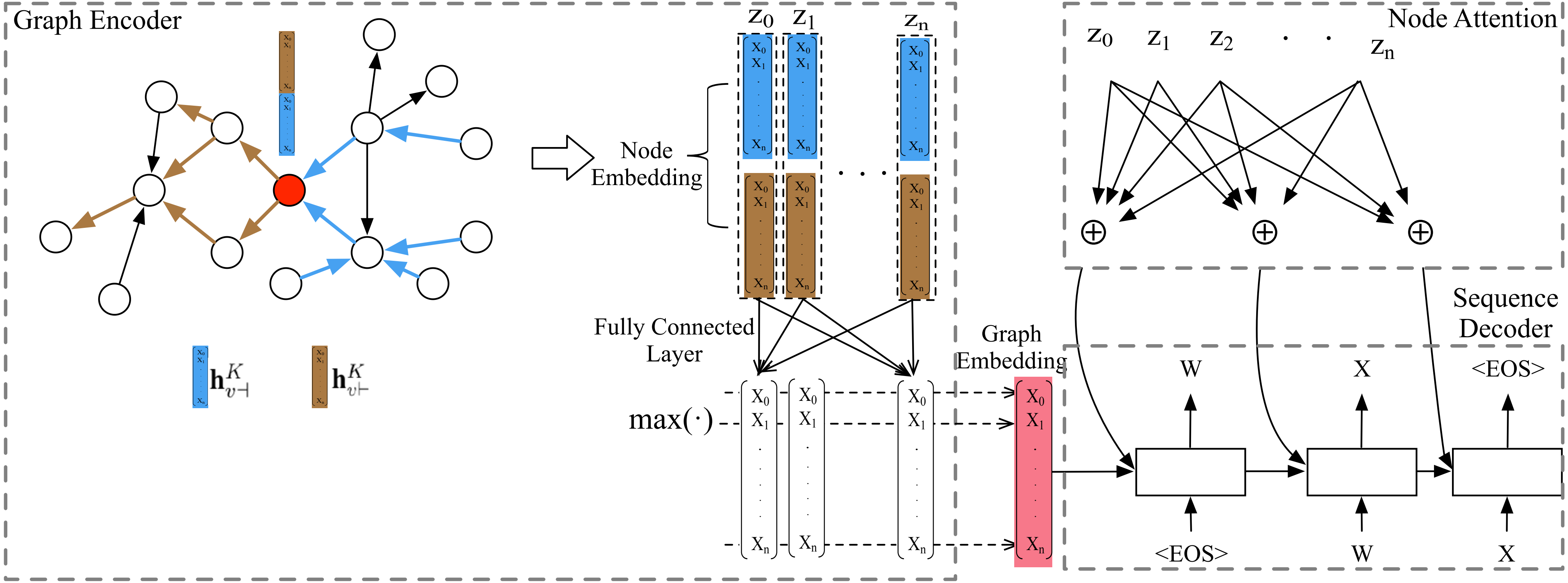
Seq2Seq: Applications and Challenges

- Applications
  - Machine translation
  - Natural language generation
  - Logic form translation
  - Information extraction



- Challenges
  - Only applied to problems whose inputs are represented as sequences
  - Cannot handle more complex structure such as graphs
  - Converting graph inputs into sequences inputs lose information
  - Augmenting original sequence inputs with additional structural information enhances word sequence feature

Graph-to-Sequence Model



[1] Kun Xu*, Lingfei Wu*, Zhiguo Wang, Yansong Feng, Michael Witbrock, and Vadim Sheinin (Equally Contributed), "Graph2Seq: Graph to Sequence Learning with Attention-based Neural Networks", arXiv 2018.

[2] Yu Chen, Lingfei Wu** and Mohammed J. Zaki (**Corresponding Author), "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation", ICLR'20.

Bidirectional GNNs for Directed Graphs

Bi-Sep GNNs formulation:

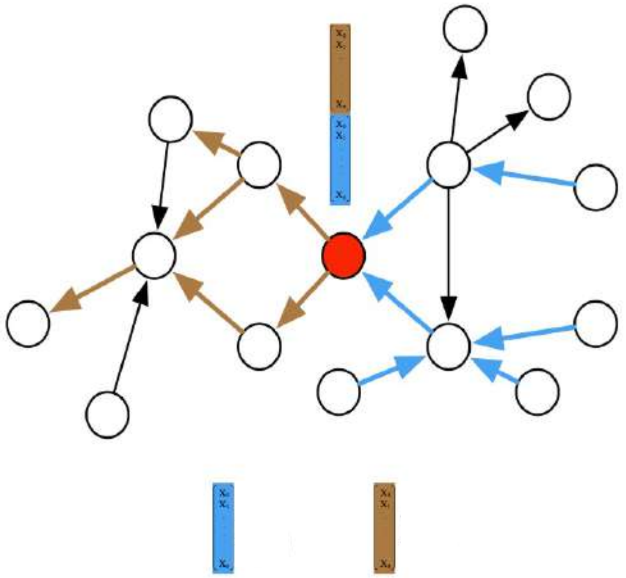
Run multi-hop backward/forward GNN on the graph

$$\mathbf{h}_{i,\dashv}^{k} = GNN(\mathbf{h}_{i,\dashv}^{k-1}, \{\mathbf{h}_{j,\dashv}^{k-1} : \forall v_j \in \mathcal{N}_{\dashv}(v_i)\})$$

$$\mathbf{h}_{i,\vdash}^{k} = GNN(\mathbf{h}_{i,\vdash}^{k-1}, \{\mathbf{h}_{j,\vdash}^{k-1} : \forall v_j \in \mathcal{N}_{\vdash}(v_i)\})$$

Concatenate backward/forward node embeddings at last hop

$$\mathbf{h}_{i}^{K} = \mathbf{h}_{i,\dashv}^{K} || \mathbf{h}_{i,\vdash}^{K}$$



$\mathbf{h}_{i,\dashv}^{K}$ $\mathbf{h}_{i,\vdash}^{K}$

Xu et al. "Graph2Seq: Graph to Sequence Learning with Attention-based Neural Networks". 2018.

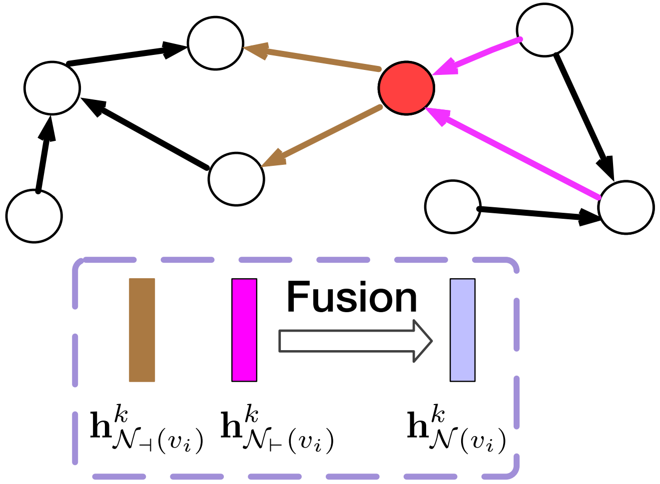
Bidirectional GNNs for Directed Graphs (cont)

Bi-Fuse GNNs formulation:

Run one-hop backward/forward node aggregation

$$\mathbf{h}^k_{\mathcal{N}_{\dashv}(v_i)} = AGG(\mathbf{h}^{k-1}_i, \{\mathbf{h}^{k-1}_j : \forall v_j \in \mathcal{N}_{\dashv}(v_i)\})$$

$$\mathbf{h}^k_{\mathcal{N}_{\vdash}(v_i)} = AGG(\mathbf{h}^{k-1}_i, \{\mathbf{h}^{k-1}_j : \forall v_j \in \mathcal{N}_{\vdash}(v_i)\})$$



Fuse backward/forward aggregation vectors at each hop

$$\mathbf{h}^k_{\mathcal{N}(v_i)} = Fuse(\mathbf{h}^k_{\mathcal{N}_{\dashv}(v_i)}, \mathbf{h}^k_{\mathcal{N}_{\vdash}(v_i)})$$

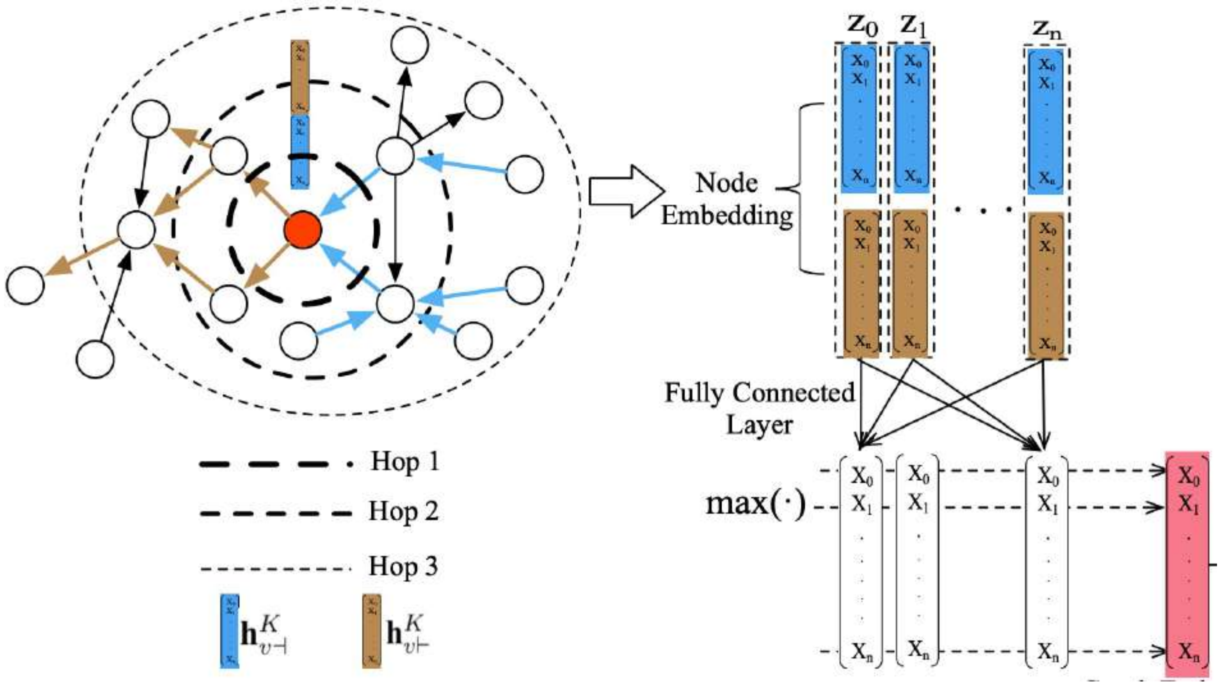
Update node embeddings with fused aggregation vectors at each hop

$$\mathbf{h}^k_i = \sigma(\mathbf{h}^{k-1}_i, \mathbf{h}^k_{\mathcal{N}(v_i)})$$

Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.

Graph Encoding

- Graph embedding
  - Pooling based graph embedding (*max, min and average pooling*)
  - Node based graph embedding
    - ◻ Add one super node which is connected to all other nodes in the graph
    - ◻ The embedding of this super node is treated as graph embedding



Attention Based Sequence Decoding

$$c_i = \sum_{j=1}^{\mathcal{V}} \alpha_{ij} h_j, \; where \; \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{\mathcal{V}} \exp(e_{ik})}, \; e_{ij} = a(s_{i-1}, h_j)$$

context vector

node representation

Attention Based Sequence Decoding

$$c_i = \sum_{j=1}^{\mathcal{V}} \alpha_{ij} h_j, \; where \; \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{\mathcal{V}} \exp(e_{ik})}, \; e_{ij} = a(s_{i-1}, h_j)$$

context vector

node representation

attention weights

alignment model

Attention Based Sequence Decoding

$$c_i = \sum_{j=1}^{\mathcal{V}} \alpha_{ij} h_j,\ where\ \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{\mathcal{V}} \exp(e_{ik})},\ e_{ij} = a(s_{i-1}, h_j)$$

context vector

node representation

attention weights

alignment model

- Objective Function

$$\theta^* = \arg\max_{\theta} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log p(y_t^n | y_{<t}^n, x^n)$$

Text Reasoning and Shortest Path

garden (A) bathroom (B) bedroom (C)
hallway (D) office (E) kitchen (F)

1 The **garden** is west of the **bathroom**.
2 The **bedroom** is north of the **hallway**.
3 The **office** is south of the **hallway**.
4 The **bathroom** is north of the **bedroom**.
5 The **kitchen** is east of the **bedroom**.

Transform →

A west B
B north D
E south D
B north C
F east C

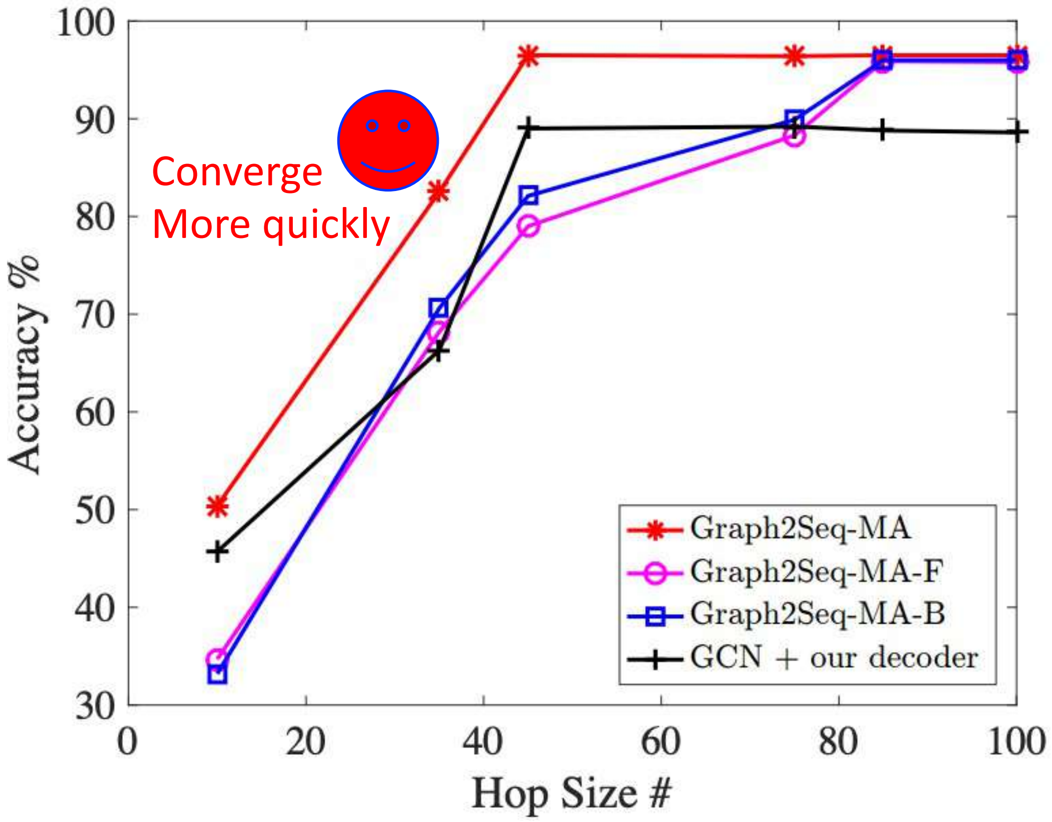
Q: How do you go from the **bathroom** to the **hallway**

Transform →

Q:path(B, D)

	bAbI T19	SP-S	SP-L
LSTM	25.2%	8.1%	2.2%
GGS-NN	98.1%	100.0%	95.2%
GCN	97.4%	100.0%	96.5%
Graph2Seq	**99.9%**	100.0%	**99.3%**

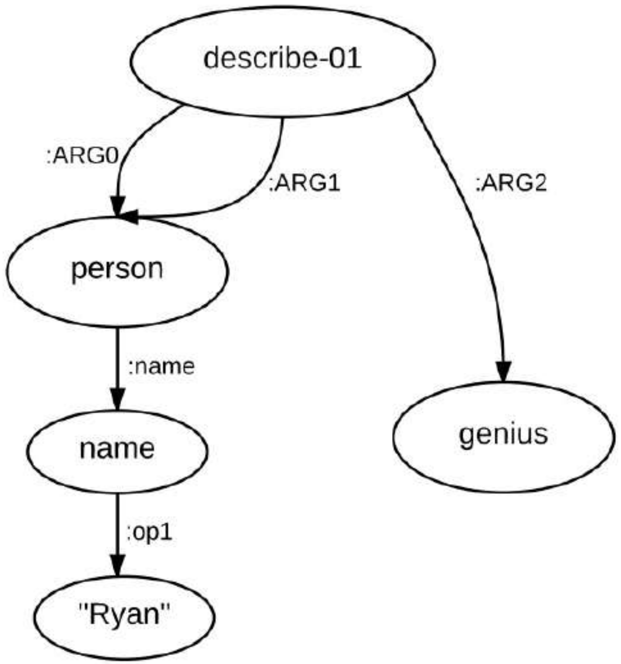
Effect of Bidirectional Node Embedding



Bidirectional Node Embedding
VS Unidirectional Node Embedding

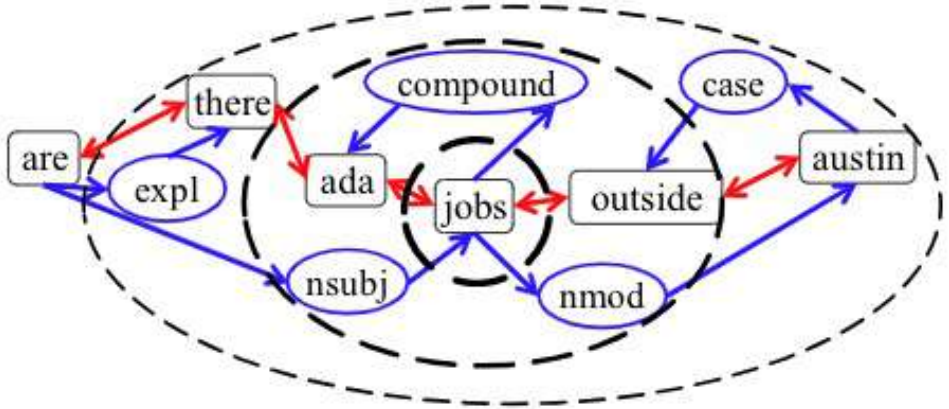
When Shall We Use Graph2Seq?

- Case I: the inputs are naturally or best represented in graph



"Ryan's description of himself: a genius."

- Case II: Hybrid Graph with sequence and its hidden structural information



Augmenting "are there ada jobs outside Austin" with its dependency parsing tree results

Learning Structured Input-Output Translation

- To bridge the semantic gap between the human-readable words and machine-understandable logics.
- Semantic parsing is important for question answering, text understanding
- Automatically solving of MWP is a growing interest.

SP	**Text Input:** what jobs are there for web developer who know 'c++' ?
	Structured output: answer(A , (job (A) , title (A , W) , const (W , 'Web Developer') , language (A , C) , const (C , 'c++')))
MWP	**Text input:** 0.5 of the cows are grazing grass . 0.25 of the cows are sleeping and 9 cows are drinking water from the pond . find the total number of cows .
	Structured output: ((0.5 * x) + (0.25 * x)) + 9.0 = x

Graph and Tree Constructions

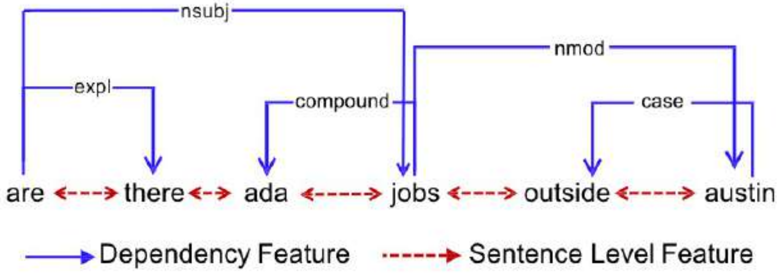


Figure 1: Dependency tree augmented text graph

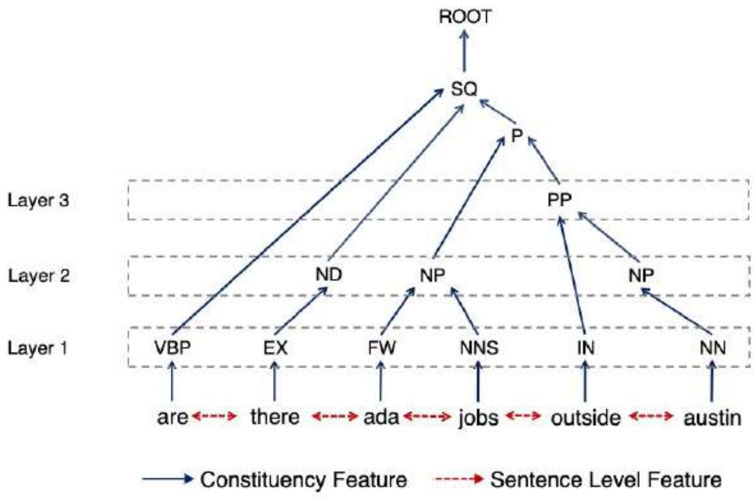


Figure 2: Constituency tree augmented text graph

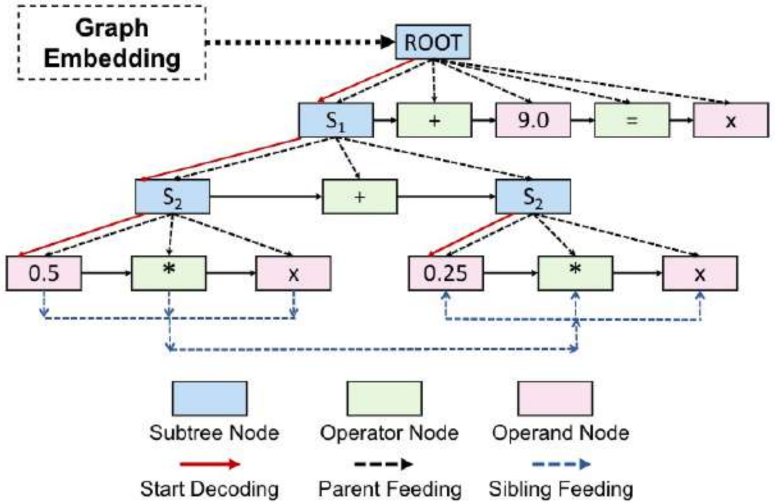
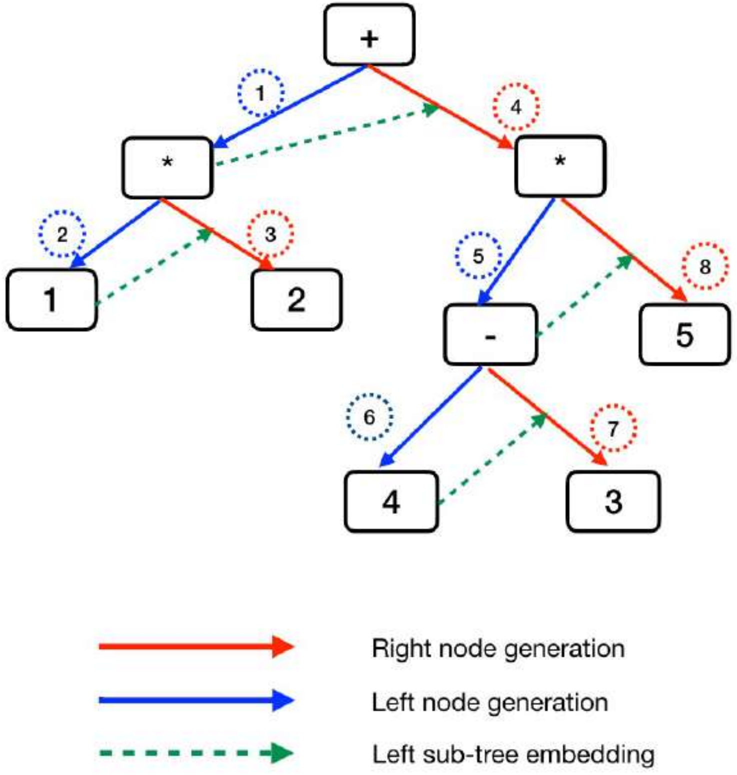
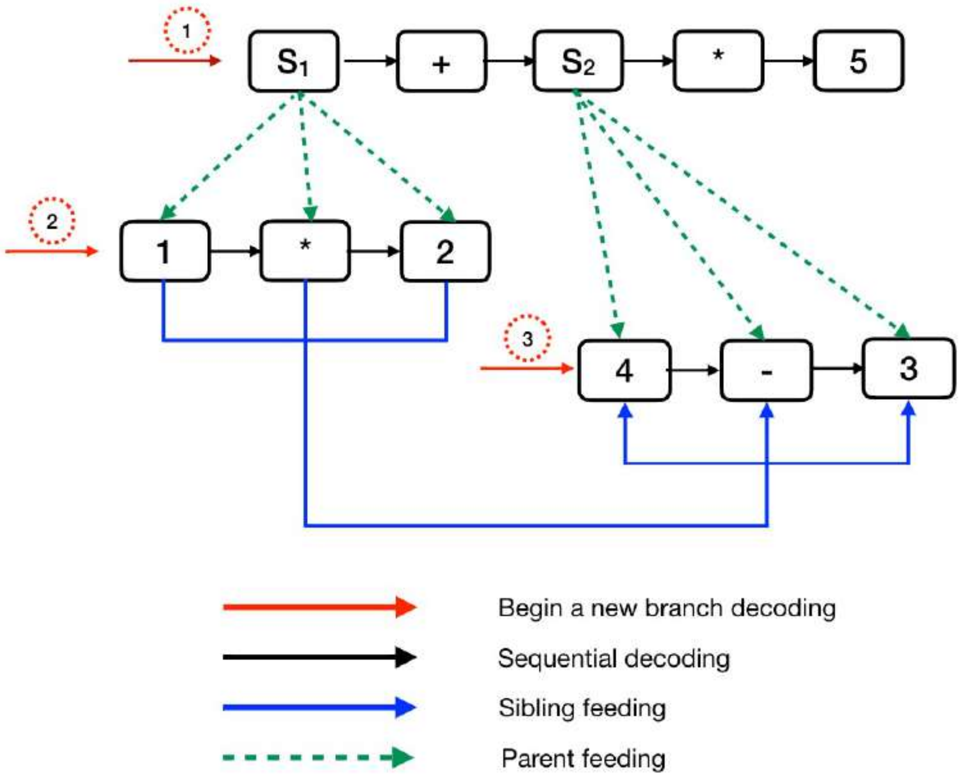


Figure 3: A sample tree output in our decoding process from expression "((**0.5** * **x**) + (**0.25** * **x**)) + **9.0** = **x**"

Tree Decoding

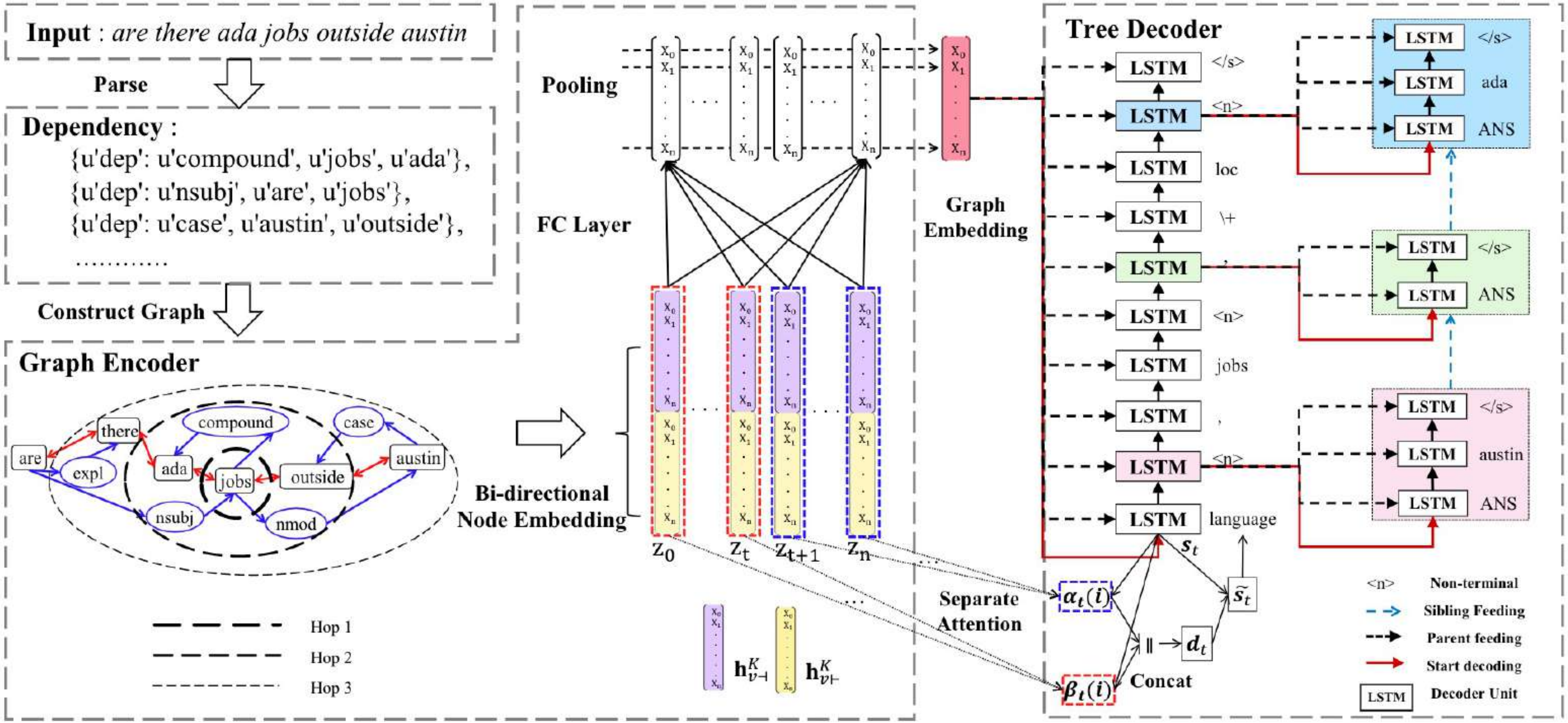


DFS-based tree decoder



BFS-based tree decoder

Graph-to-Tree Model



[1] Shucheng Li*, Lingfei Wu*, et al. "Graph-to-Tree Neural Networks for Learning Structured Input-Output Translation with Applications to Semantic Parsing and Math Word Problem", EMNLP 2020.

Separated Attention Based Tree Decoding

Context vector embeddings

$$\mathbf{c}_{v_1} = \sum \alpha_{t(v)} \mathbf{z}_v, \forall v \in \mathcal{V}_1$$
$$\mathbf{c}_{v_2} = \sum \beta_{t(v)} \mathbf{z}_v, \forall v \in \mathcal{V}_2$$

Separated attention weights

$$\alpha_{t(v)} = \frac{\exp(score(\mathbf{z}_v, \mathbf{s}_t))}{\exp(\sum_{k=1}^{V_1} score(\mathbf{z}_k, \mathbf{s}_t))}, \forall v \in \mathcal{V}_1$$

$$\beta_{t(v)} = \frac{\exp(score(\mathbf{z}_v, \mathbf{s}_t))}{\exp(\sum_{k=1}^{V_2} score(\mathbf{z}_k, \mathbf{s}_t))}, \forall v \in \mathcal{V}_2$$

Final attention hidden state

$$\tilde{\mathbf{s}_t} = \tanh(W_c \cdot [\mathbf{c}_{v_1}; \mathbf{c}_{v_2}; \mathbf{s}_t] + b_c).$$

Graph4NLP

Math Word Problem

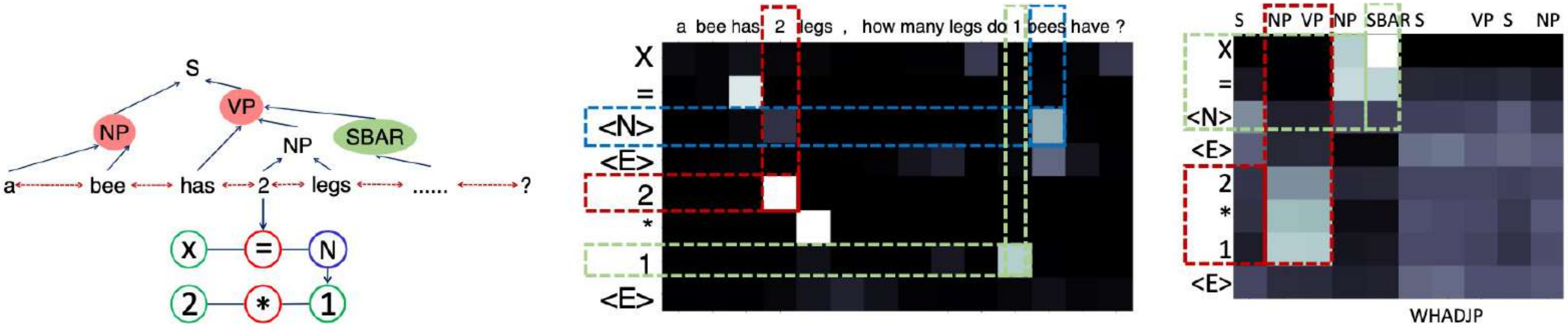
Methods		MAWPS
Oracle		84.8
Retrieval	Jaccard	45.6
	Cosine	38.8
Classification	BiLSTM	62.8
	Self-attention	60.4
Seq2seq	LSTM	25.6
	CNN	44.0
Seq2Tree		65.2
Graph2Seq		70.4
MathDQN		60.25
T-RNN	Full model	66.8
	W/o equantion normalization	63.9
	W/o self-attention	66.3
Group-Att		76.1
Graph2Tree	with constituency graph	**78.8**
	with dependency graph	76.8

Table 5: Solution accuracy comparison on *MAWPS*

Methods		MATHQA
Seq2Prog		51.9
Seq2Prog+Cat		54.2
TP-N2F		55.95
Seq2seq		58.36
Seq2Tree		64.15
Graph2Seq		65.36
Graph2Tree	with constituency graph	**69.65**
	with dependency graph	65.66

Table 6: Solution accuracy comparison on *MATHQA*

Visualization of Separated Attentions



(a) A graph-to-tree translation example (b) Attention for word nodes (c) Attention for structure nodes

Figure 5: Effect visualization of our separated attentions on both word and structure nodes in a graph.

Half-hour Break

Want to prepare for our demo session?

1) git clone https://github.com/graph4ai/graph4nlp_demo

2) follow Get Started instructions in README

References:

- Graph4NLP demo link: https://github.com/graph4ai/graph4nlp_demo
- Graph4NLP library link: https://github.com/graph4ai/graph4nlp
- DLG4NLP literature link: https://github.com/graph4ai/graph4nlp_literature

DLG4NLP Applications

Graph4NLP

Information Extraction

Outline

- Semantic Graph Parsing for Event Extraction
- Cross-lingual Structure Transfer for Relation Extraction and Event Extraction
- Cross-media Structured Common Space for Multimedia Event Extraction
- Graph Schema-guided Event Extraction and Prediction
- Cross-media Knowledge Graph based Misinformation Detection

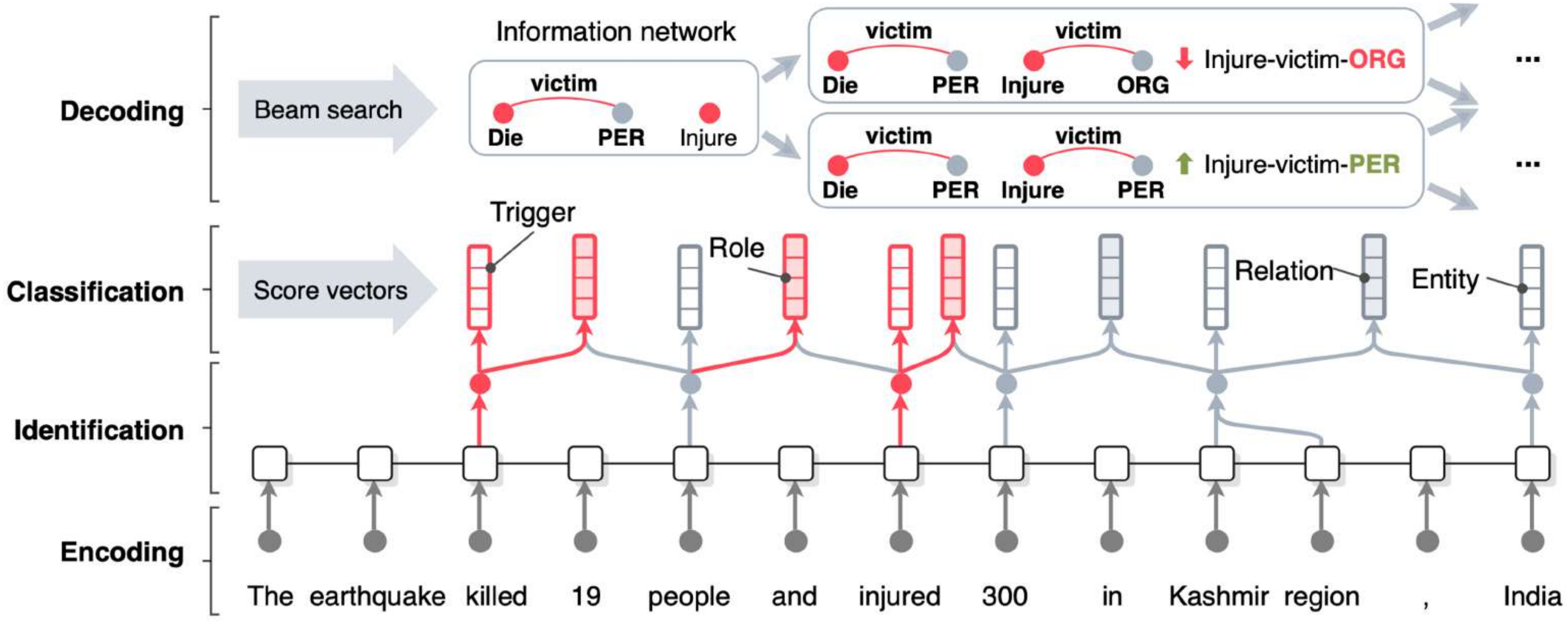
Improve Quality

Improve Portability

Extend Scope

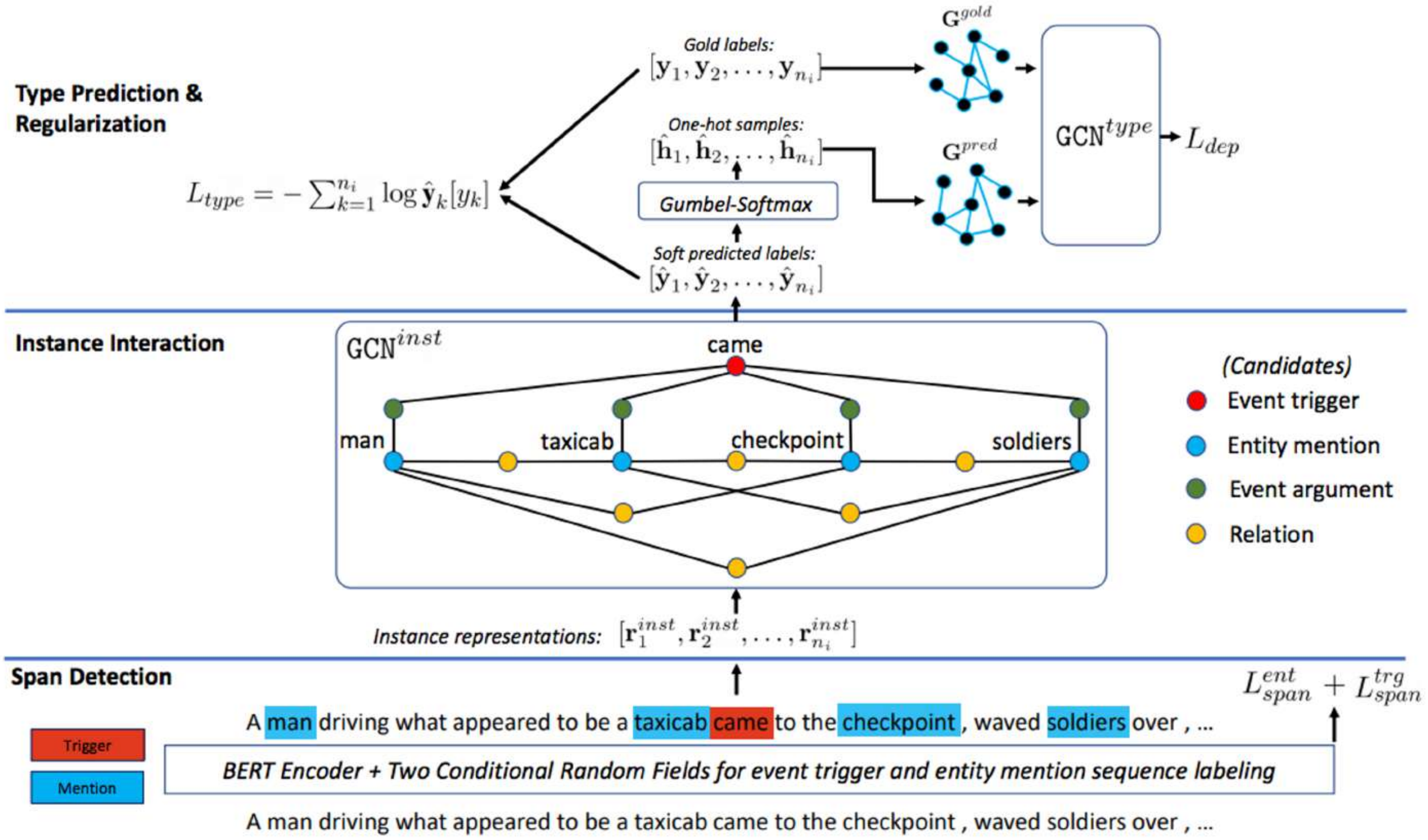
New Application

Information Extraction: a Sequence-to-Graph Task



- OneIE [Lin et al., ACL2020] framework extracts the information graph from a given sentence in four steps: encoding, identification, classification, and decoding

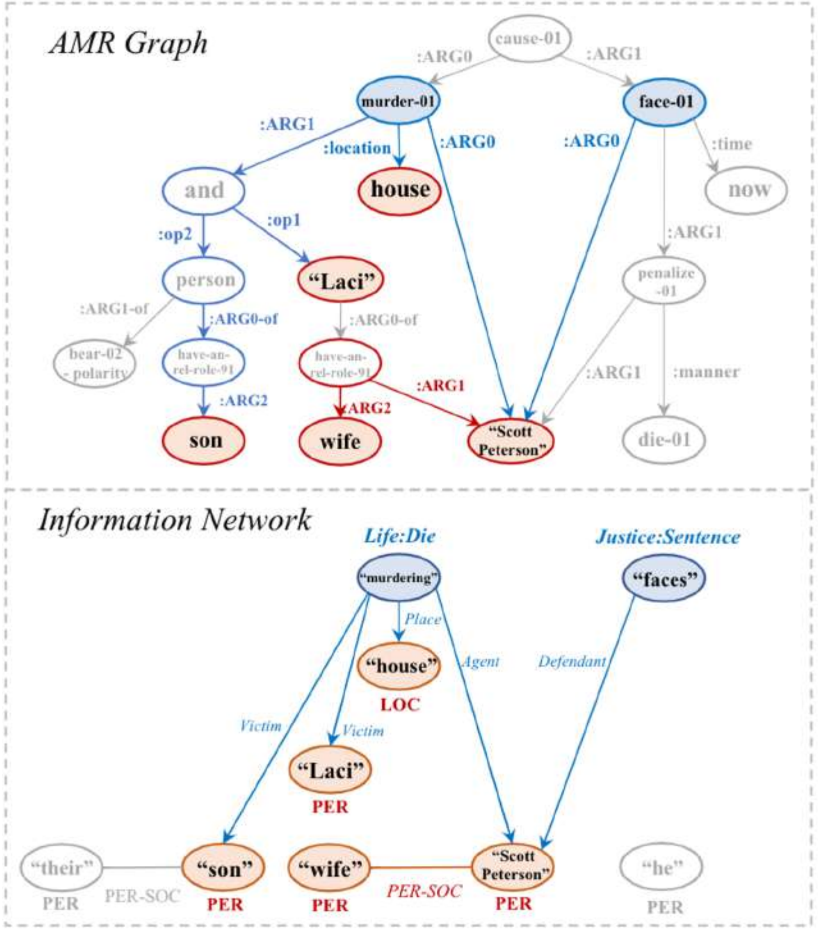
Extending to Graph-to-Graph Task



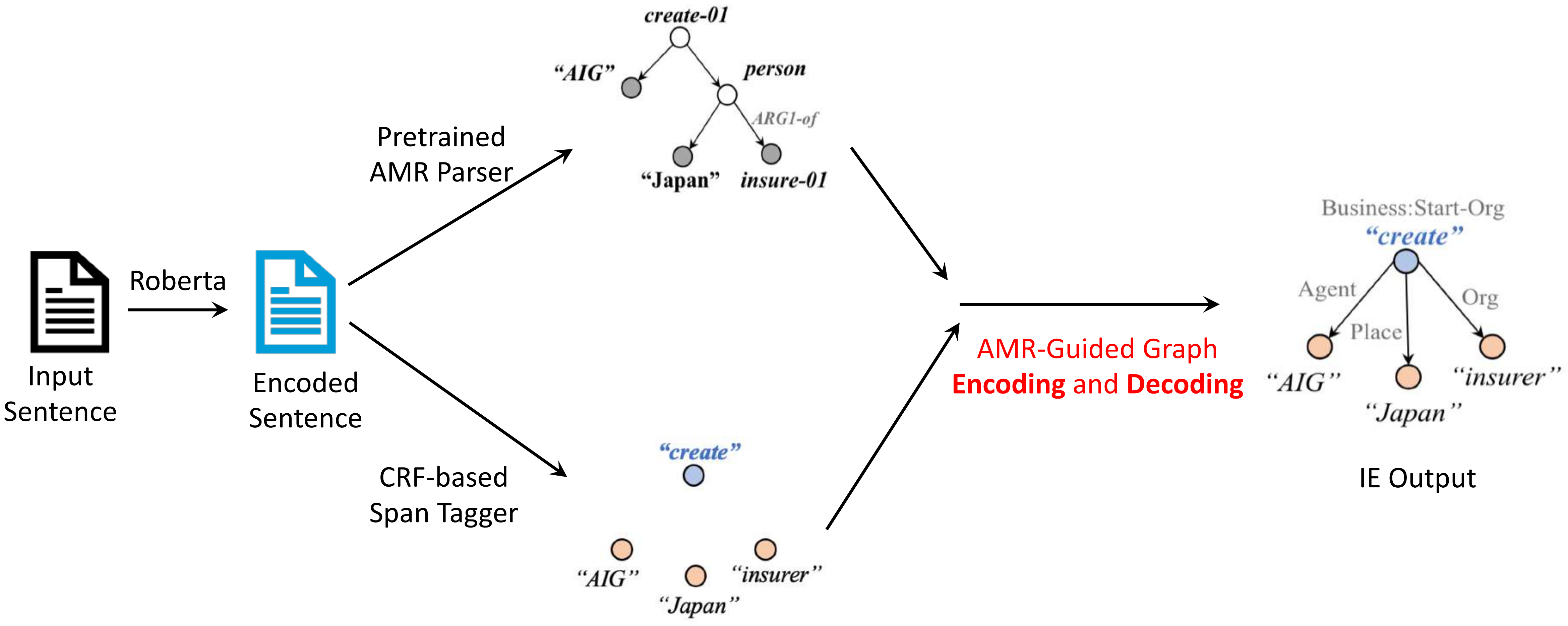
- [Nyuyen et al., NAACL2021]

Moving from Seq-to-Graph to Graph-to-Graph

- [Zhang and Ji, NAACL2021]
- Abstract Meaning Representation (AMR):
  - A kind of rich semantic parsing
  - Converts input sentence into a directed and acyclic graph structure with fine-grained node and edge type labels
- AMR parsing shares inherent similarities with information network (IE output)
  - Similar node and edge semantics
  - Similar graph topology
- Semantic graphs can better capture non-local context in a sentence
- Exploit the similarity between AMR and IE to help on joint information extraction



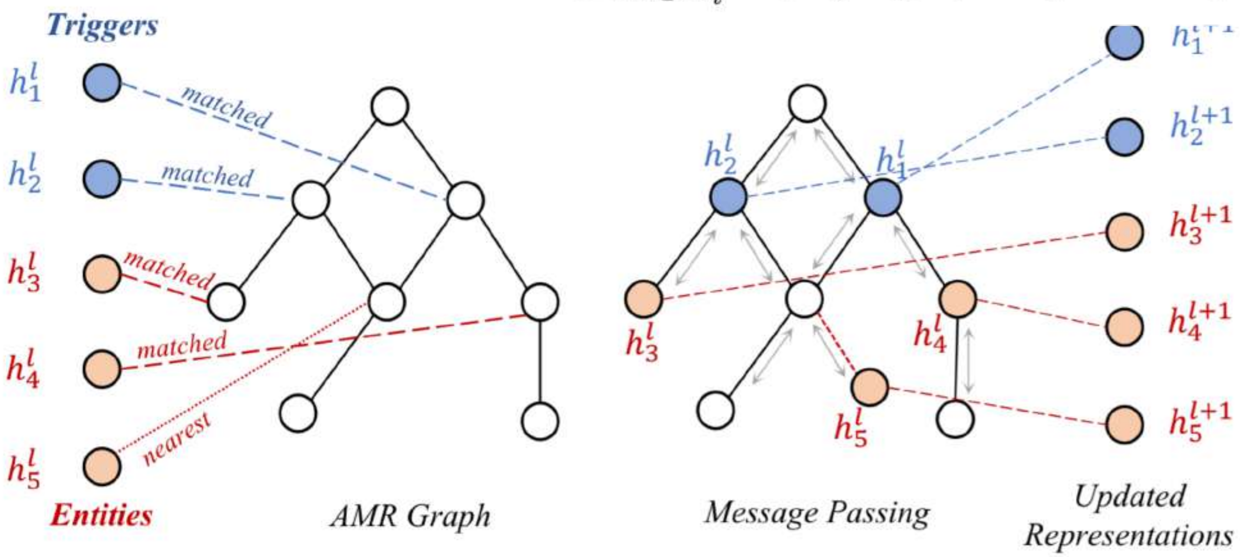
AMR-IE: An AMR-guided encoding and decoding framework for IE



AMR Guided Graph Encoding: Using an Edge-Conditioned GAT

- Map each candidate entity and event to AMR nodes.
- Update entity and event representations using an edge-conditioned GAT to incorporate information from AMR neighbors.

$$\alpha_{i,j}^{l} = \frac{\exp\left(\sigma\left(f^{l}[\mathbf{W}\boldsymbol{h}_i^l : \mathbf{W}_e\boldsymbol{e}_{i,j} : \mathbf{W}\boldsymbol{h}_j^l]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\sigma\left(f^{l}[\mathbf{W}\boldsymbol{h}_i^l : \mathbf{W}_e\boldsymbol{e}_{i,k} : \mathbf{W}\boldsymbol{h}_k^l]\right)\right)}$$



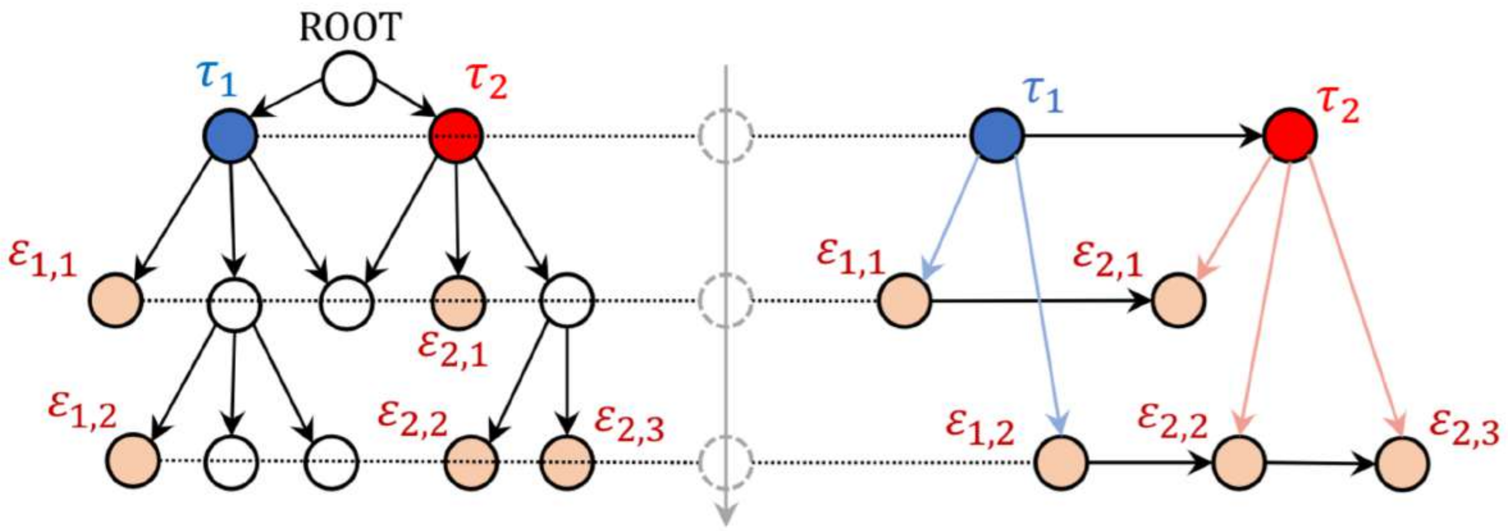
$$\boldsymbol{h}^* = \sum_{j\in\mathcal{N}_i}\alpha_{i,j}^{l}\boldsymbol{h}_i^l$$

$$\boldsymbol{h}^{l+1} = \boldsymbol{h}^{l} + \gamma\cdot\mathbf{W}^*\boldsymbol{h}^*$$

AMR Guided Graph Decoding: Ordered decoding guided by AMR

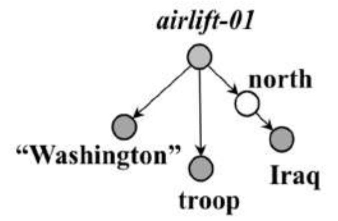
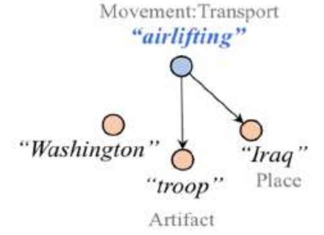
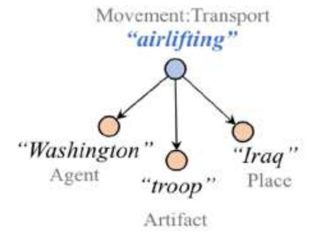
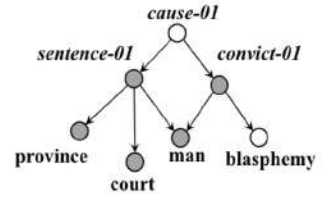
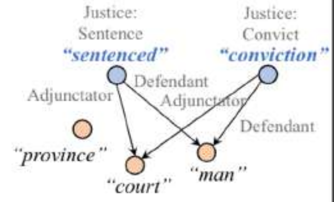
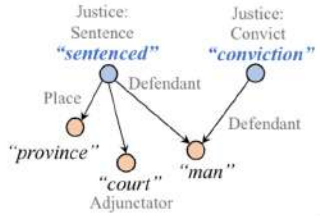
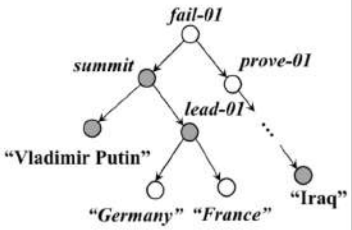
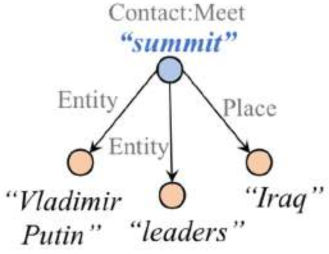
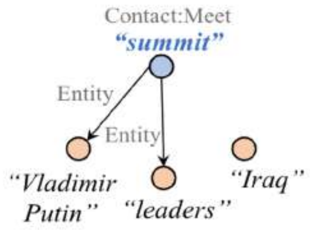
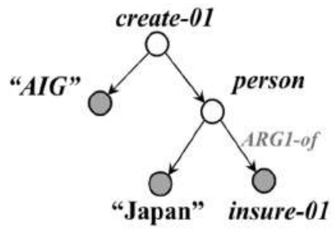
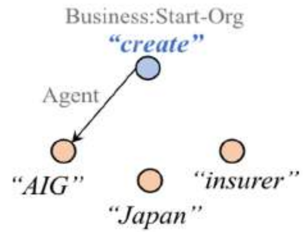
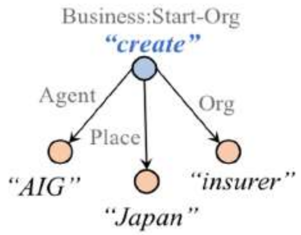
- Beam search based decoding as in *OneIE* (Lin et al. 2020).
- The decoding order of candidate nodes are determined by the hierarchy in AMR in a **top-to-down manner**.
- For example, the correct ordered decoding in the following graph is:

$$\tau_1, \tau_2, \varepsilon_{1,1}, \varepsilon_{2,1}, \varepsilon_{1,2}, \varepsilon_{2,2}, \varepsilon_{2,3}$$



Examples on how AMR graphs help

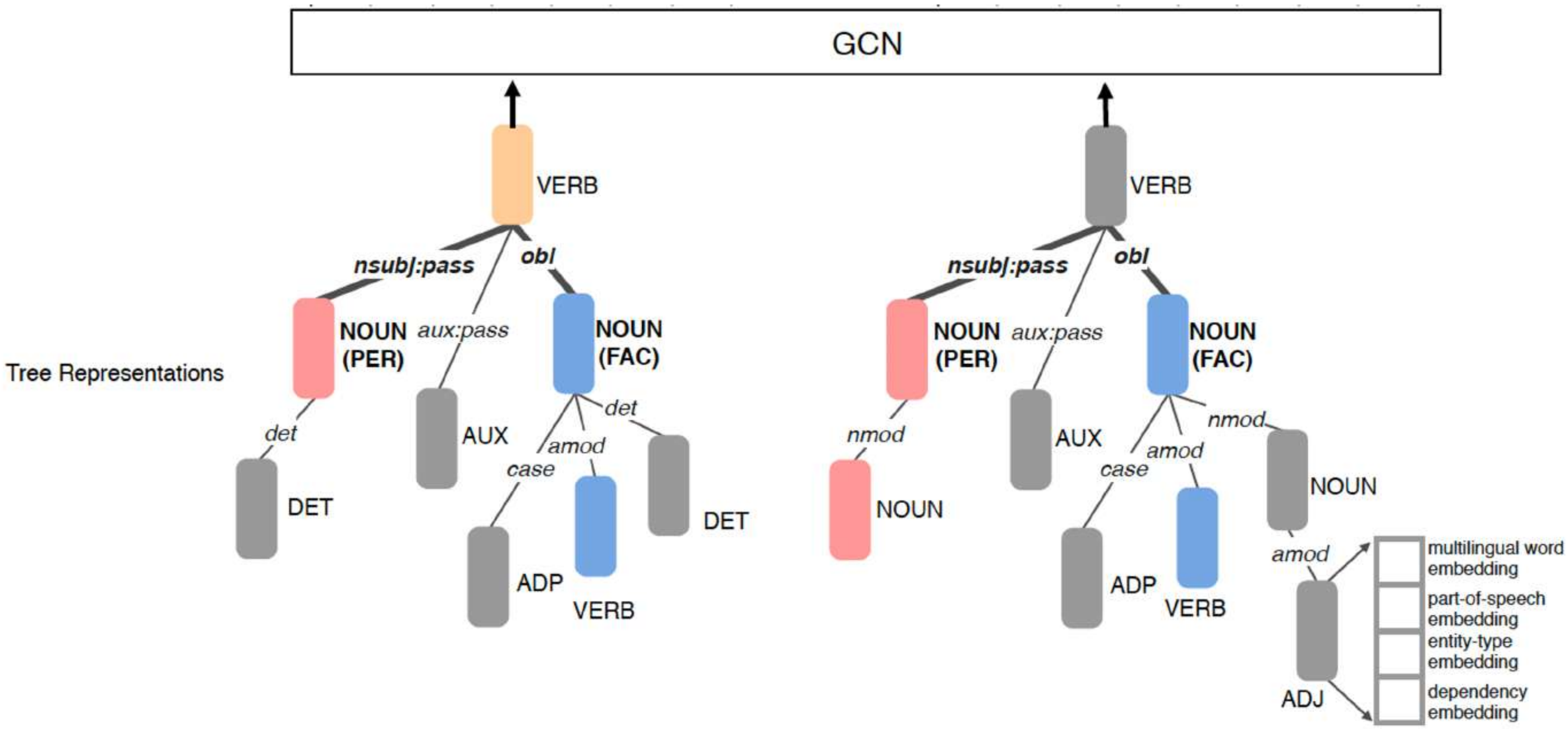
Graph4NLP

Sentence	AMR Parsing	*OneIE outputs*	*AMR-IE outputs*
If the resolution is not passed, **Washington** would likely want to use the airspace for strikes against Iraq and for **airlifting** **troops** to northern **Iraq**.			
A Pakistani **court** in central Punjab **province** has **sentenced** a Christian **man** to life imprisonment for a blasphemy **conviction**, police said Sunday.			
Russian President **Vladimir Putin**'s **summit** with the **leaders** of Germany and France may have been a failure that proves there can be no long-term "peace camp" alliance following the end of war in **Iraq**.			
Major US insurance group **AIG** is in the final stage of talks to take over General Electric's Japanese life insurance arm in a deal to **create** **Japan**'s sixth largest life **insurer**, reports said Wednesday.			

Outline

- Semantic Graph Parsing for Event Extraction
- Cross-lingual structure transfer for Relation Extraction and Event Extraction
- Cross-media Structured Common Space for Multimedia Event Extraction
- Graph Schema-guided Event Extraction and Prediction
- Cross-media Knowledge Graph based Misinformation Detection

Cross-lingual Structure Transfer



The detainees were taken to a processing center

Команды врачей были замечены в упакованных отделениях скорой помощи
(teams of doctors were seen in packed emergency rooms)

Graph Convolutional Networks (GCN) Encoder

- Extend the monolingual design (Zhang et al., 2018) to cross-lingual
- Convert a sentence with N tokens into N*N adjacency matrix *A*
- Node: token, each edge is a directed dependency edge
- Initialization of each node's representation

$$\boldsymbol{h}_i^{(0)} = \underbrace{\boldsymbol{x}_i^w}_{\text{Word embedding}} \oplus \underbrace{\boldsymbol{x}_i^p}_{\text{POS tag}} \oplus \underbrace{\boldsymbol{x}_i^d}_{\text{Dependency relation}} \oplus \underbrace{\boldsymbol{x}_i^e}_{\text{Entity type}}$$

- At the k^th layer, derive the hidden representation of each node from the representations of its neighbors at previous layer

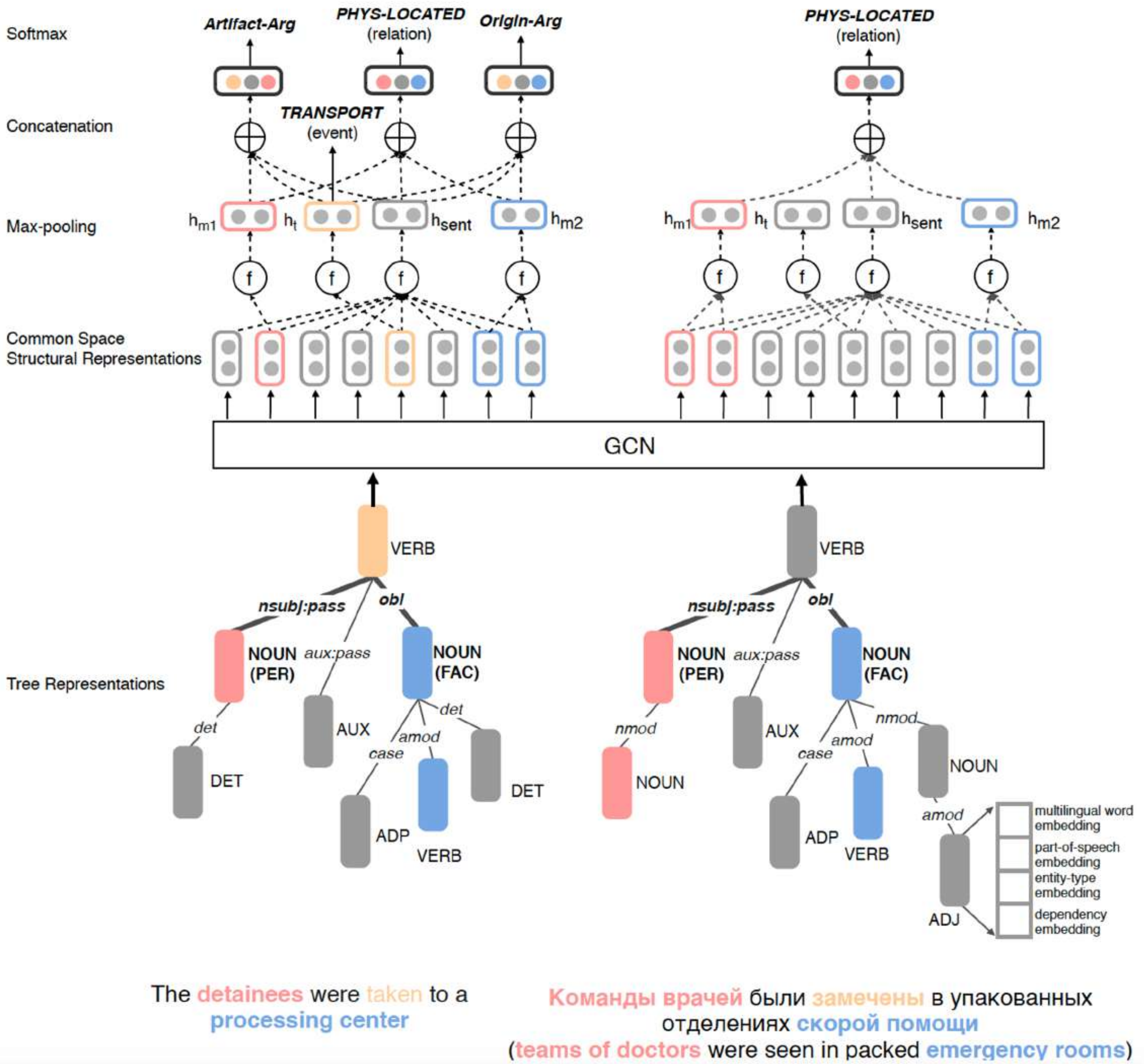
$$\boldsymbol{h}_i^{(k)} = \mathrm{ReLU}\left(\sum_{j=0}^{N} \frac{\boldsymbol{A}_{ij}\boldsymbol{W}^{(k)}\boldsymbol{h}_j^{(k-1)}}{d_i + b^{(k)}}\right)$$

Application on Event Argument Extraction

Graph4NLP

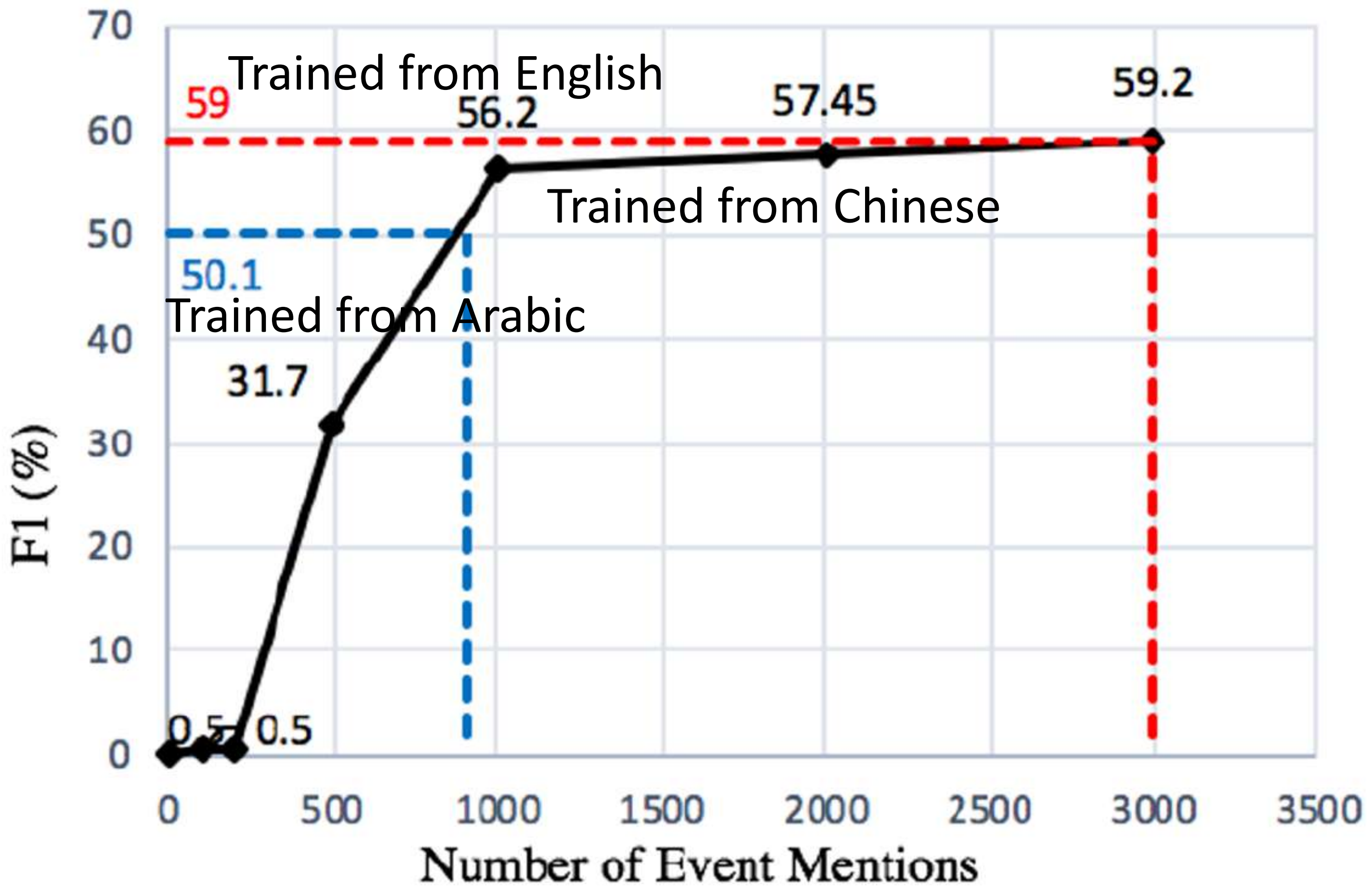
- Task: Classify each pair of event trigger and entity mentions into one of pre-defined event argument roles or NONE
- Max-pooling over the final node representations to obtain representations for sentence, trigger and argument candidate, and concatenate them
- A softmax output layer for argument role labeling

$$L^a = \sum_{i=1}^{N} \sum_{j=1}^{L_i} y_{ij} \log(\sigma(\boldsymbol{U}^a \cdot [\boldsymbol{h}_i^t; \boldsymbol{h}_{ij}^s; \boldsymbol{h}_j^a]))$$


Softmax
Concatenation
Max-pooling
Common Space
Structural Representations
Tree Representations
Artifact-Arg
PHYS-LOCATED
(relation)
Origin-Arg
TRANSPORT
(event)
GCN
VERB
nsubj:pass
obl
NOUN
(PER)
aux:pass
NOUN
(FAC)
det
AUX
amod
case
DET
ADP
VERB
nmod
NOUN
ADJ
multilingual word embedding
part-of-speech embedding
entity-type embedding
dependency embedding
The detainees were taken to a processing center
Команды врачей были замечены в упакованных отделениях скорой помощи
(teams of doctors were seen in packed emergency rooms)

Cross-lingual Edge Transfer Performance

- Chinese Event Argument Extraction



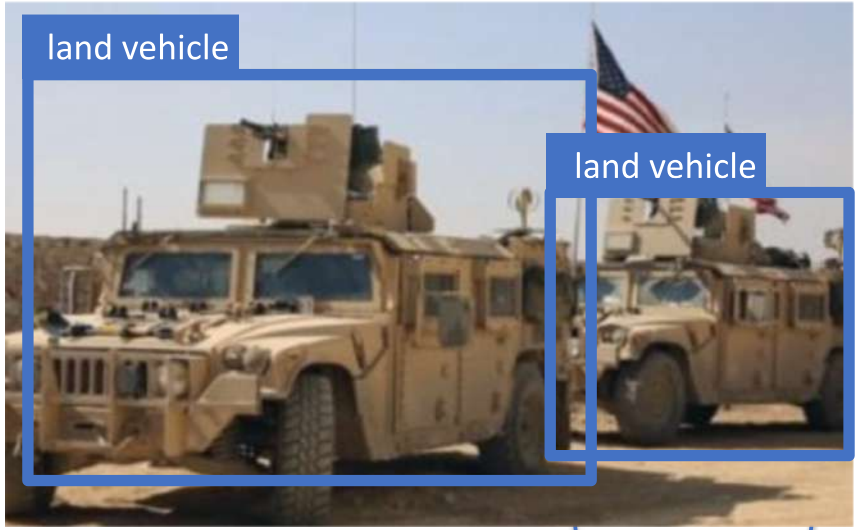
Outline

- Semantic Graph Parsing for Event Extraction
- Cross-lingual structure transfer for Relation Extraction and Event Extraction
- Cross-media Structured Common Space for Multimedia Event Extraction
- Graph Schema-guided Event Extraction and Prediction
- Cross-media Knowledge Graph based Misinformation Detection

Multimedia Event Extraction (M^2E^2)

[Li et al., ACL2020]

Last week , U.S . Secretary of State Rex Tillerson visited Ankara, the first senior administration official to visit Turkey, to try to seal a deal about the battle for Raqqa and to overcome President Recep Tayyip Erdogan's strong objections to Washington's backing of the Kurdish Democratic Union Party (PYD) militias. Turkish forces have attacked SDF forces in the past around Manbij, west of Raqqa, forcing the **United States** to **deploy** dozens of **soldiers** on the **outskirts** of the town in a mission to prevent a repeat of clashes, which risk derailing an assault on Raqqa.



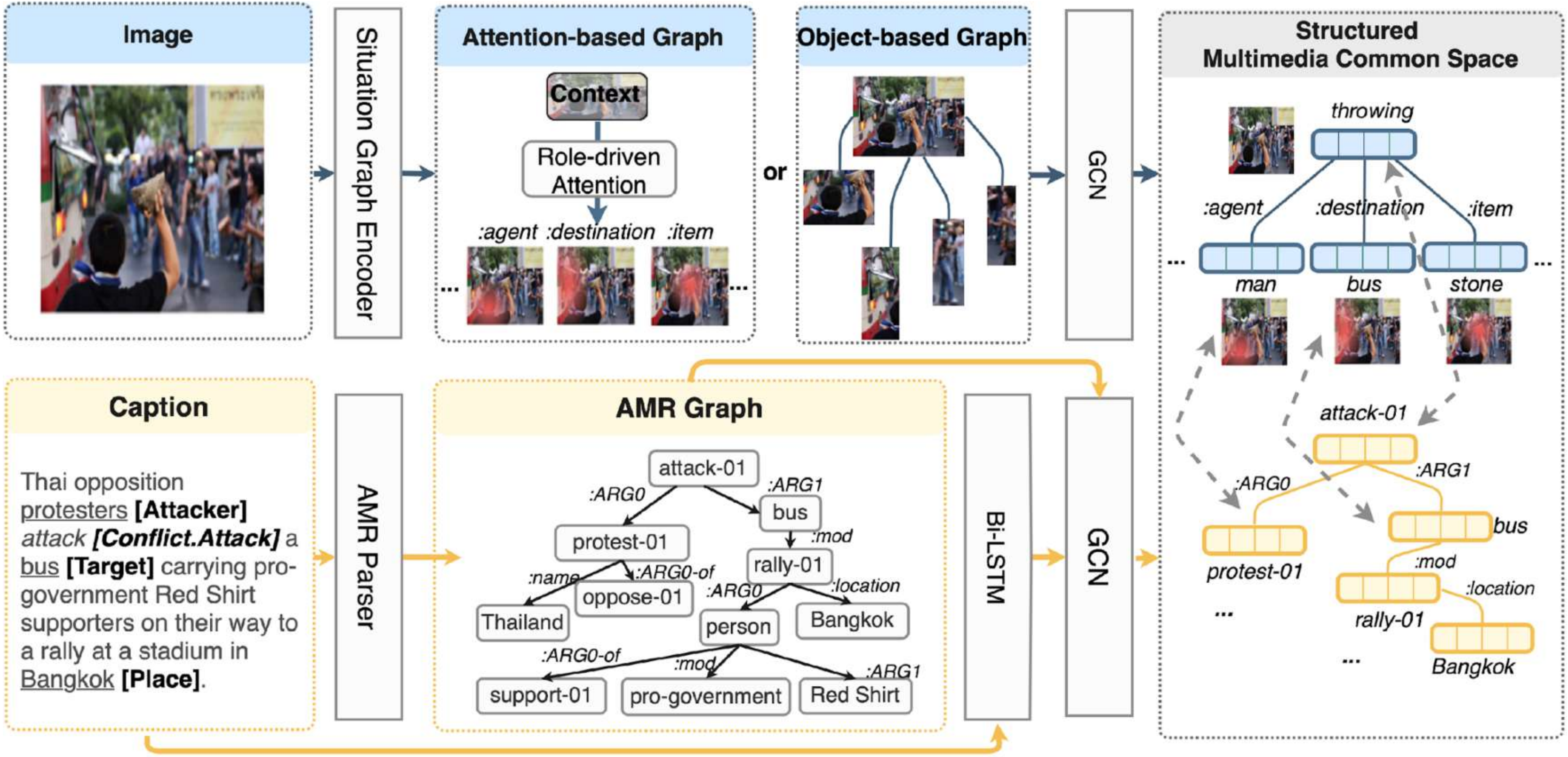
Output: Multimedia Events & Argument Roles

Event Type	Movement.Transport	
Event	**Text Trigger**	deploy
	Image	

Arguments	**Agent**	United States
	Destination	outskirts
	Artifact	soldiers
	Vehicle	
	Vehicle	

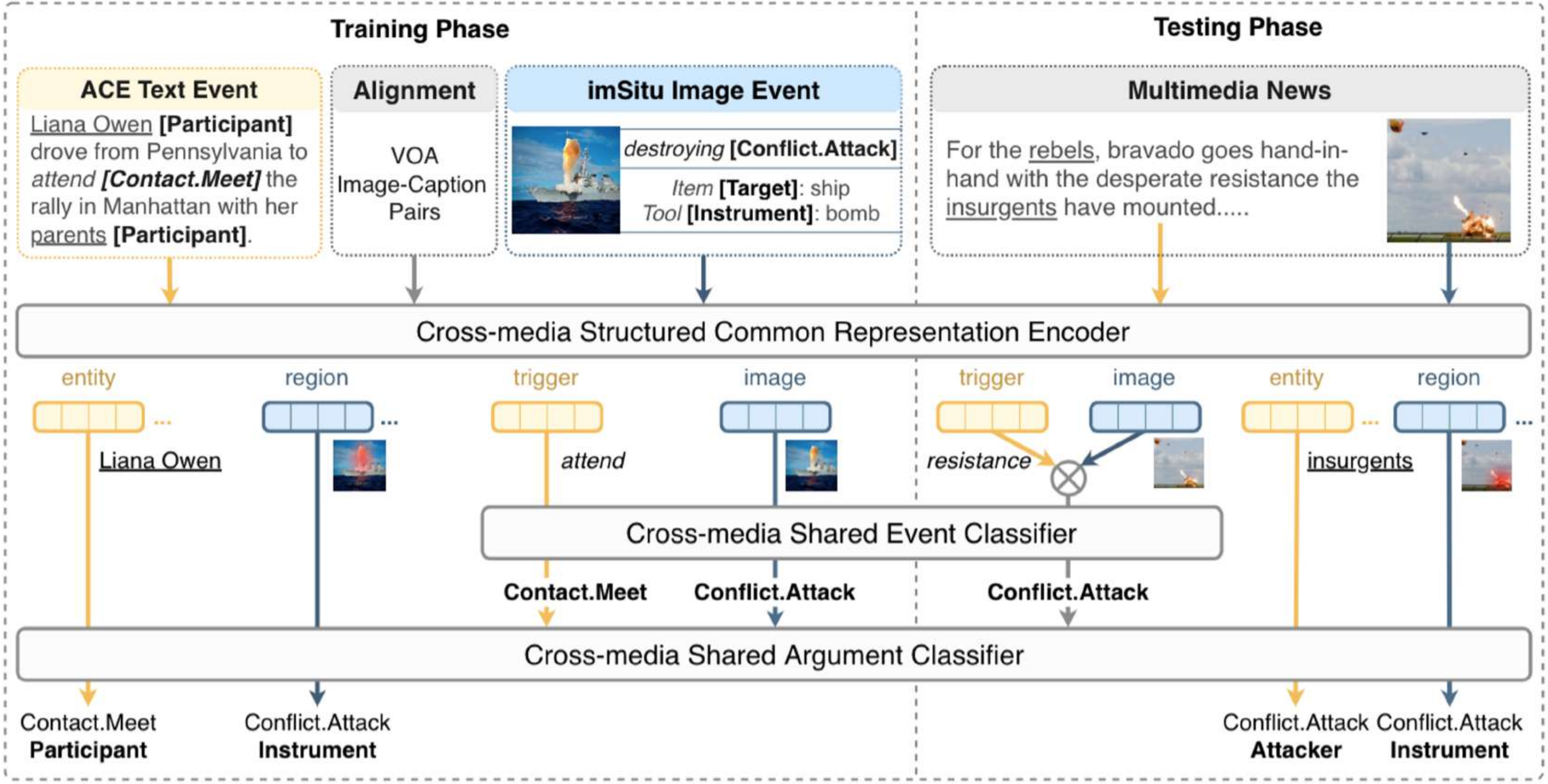
Weakly Aligned Structured Embedding

-- Training Phase (Common Space Construction)



Weakly Aligned Structured Embedding

-- Training and Test Phase (Cross-media shared classifiers)



Compare to Single Data Modality Extraction

- Surrounding sentence helps visual event extraction.
- Image helps textual event extraction.

People celebrate Supreme Court ruling on Same Sex Marriage in front of the Supreme Court in Washington.

Iraqi security forces *search* ***[Justice.Arrest]*** a civilian in the city of Mosul.

Compare to Cross-media Flat Representation

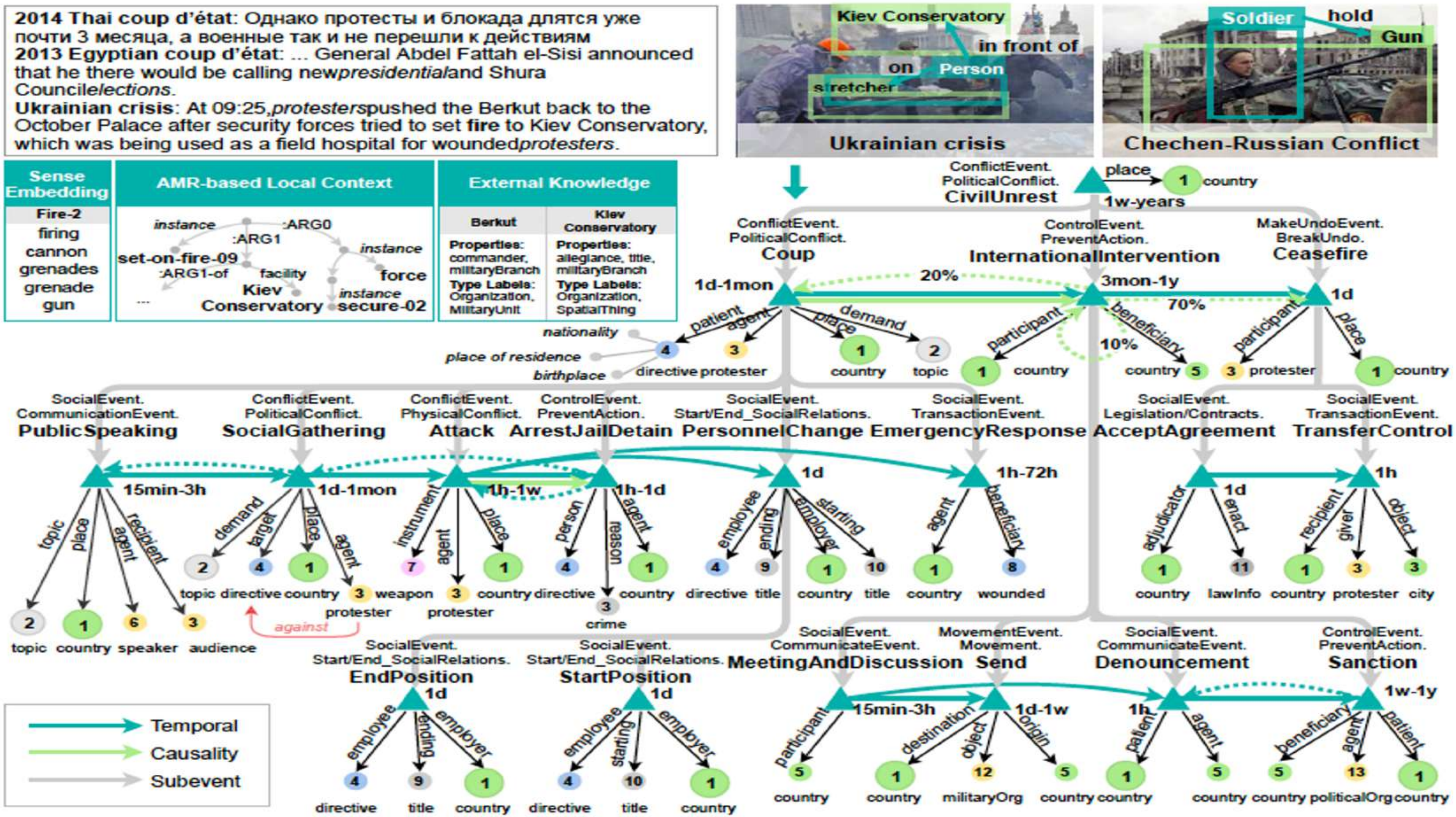
Model	Event Type	Argument Role	
Flat	Justice.ArrestJail	Agent =	man
Ours	Justice.ArrestJail	Entity =	man

Model	Event Type	Argument Role	
Flat	Movement.Transport	Artifact =	none
Ours	Movement.Transport	Artifact =	man

Outline

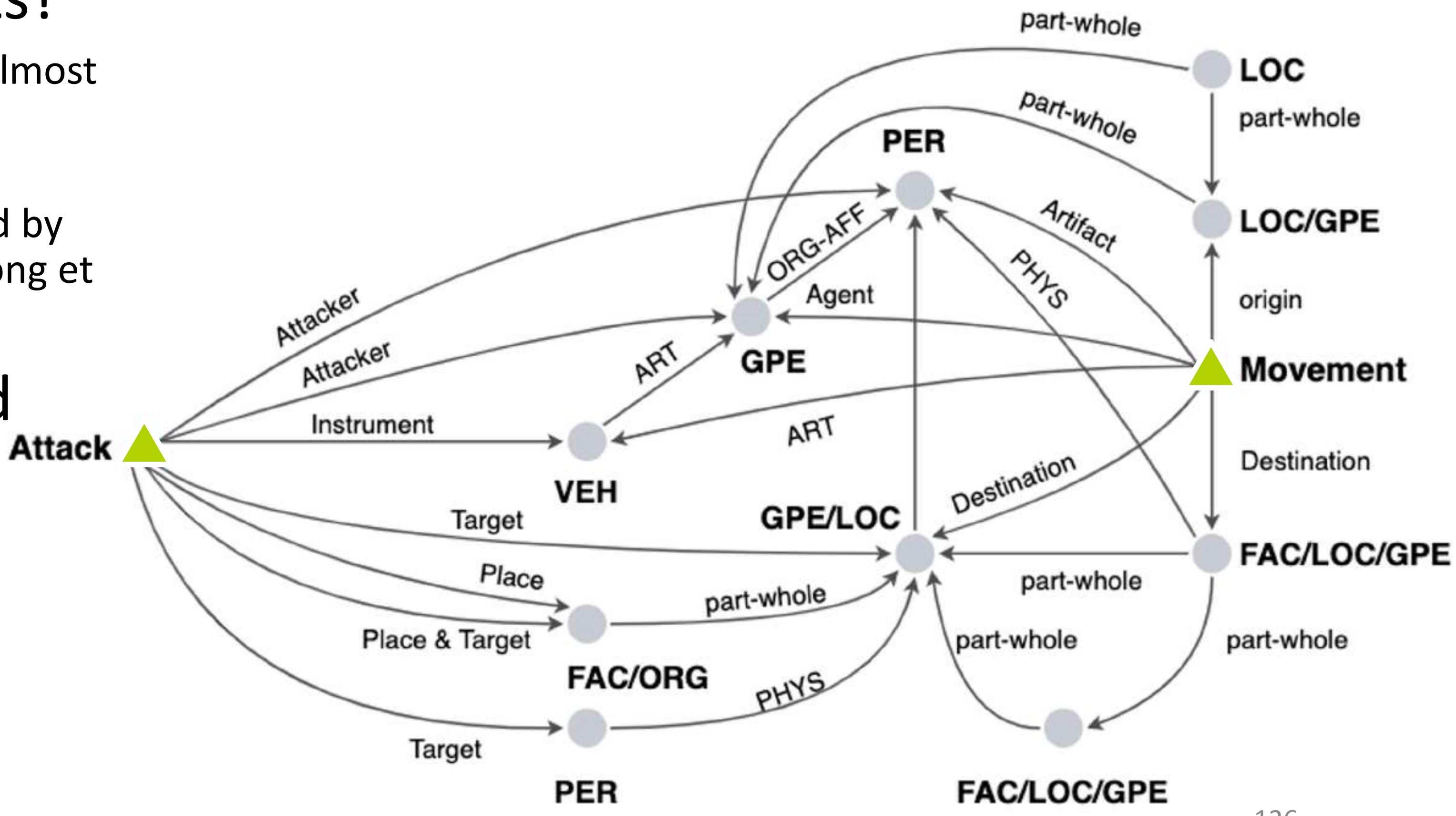
- Semantic Graph Parsing for Event Extraction
- Cross-lingual structure transfer for Relation Extraction and Event Extraction
- Cross-media Structured Common Space for Multimedia Event Extraction
- Graph Schema-guided Event Extraction and Prediction
- Cross-media Knowledge Graph based Misinformation Detection

Move from Entity-Centric to Event-Centric NLU



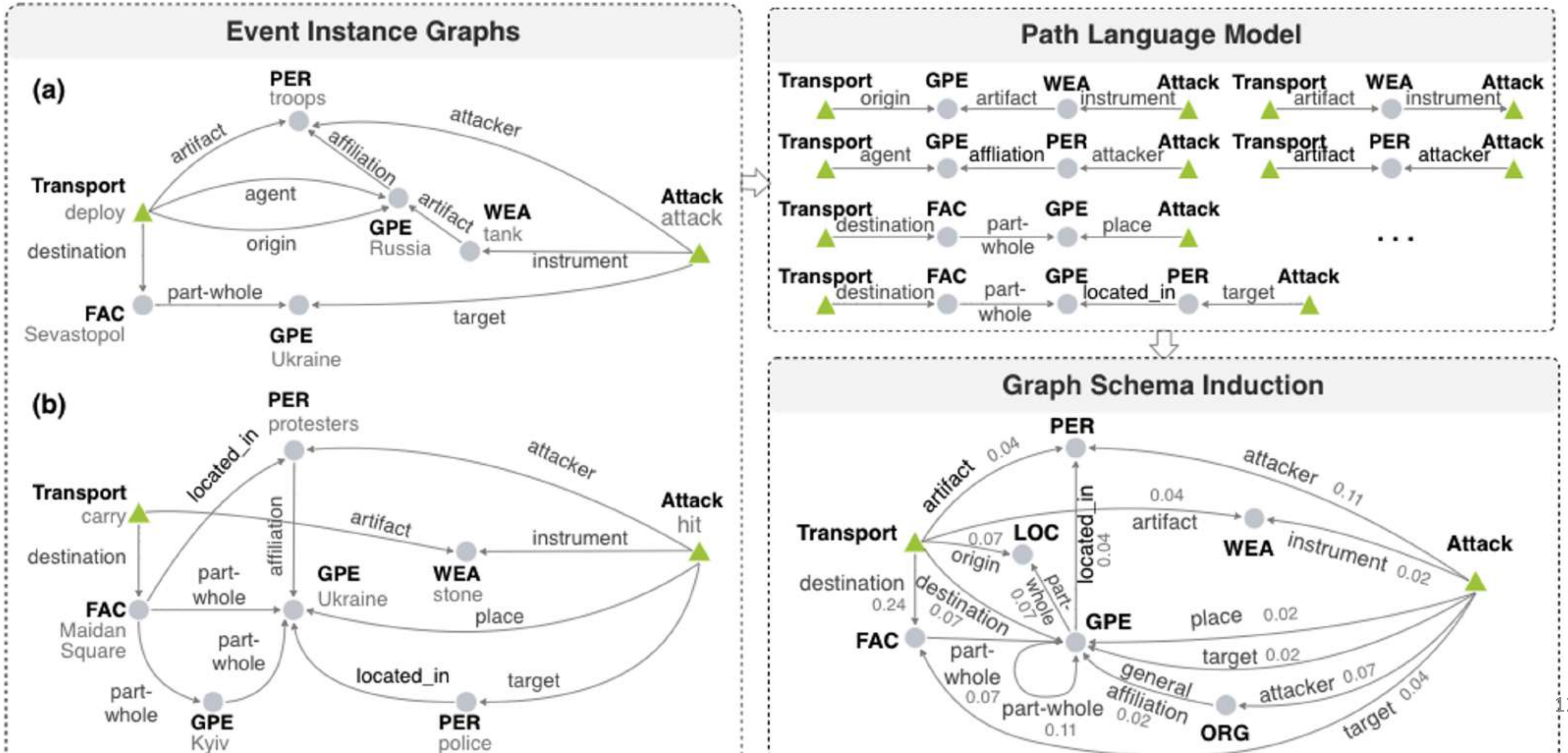
Event Graph Schema Induction

- [Li et al., EMNLP2020]
- How to capture complex connections among events?
  - Temporal relations exist between almost all events, even those that are not semantically related
  - Causal relations have been hobbled by low inter-annotator agreement (Hong et al., 2016)
- Two events are connected through entities and their relations



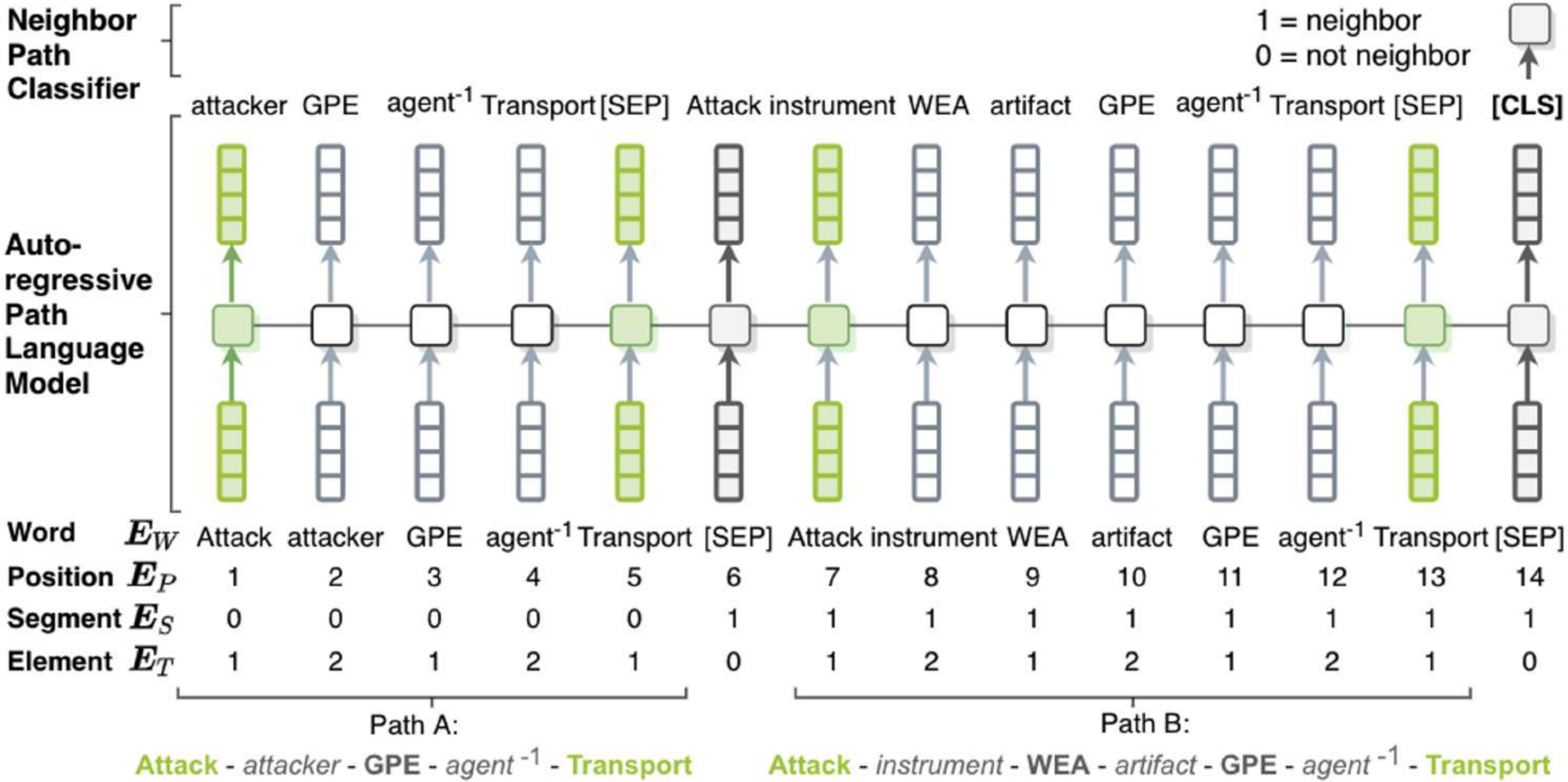
Event Graph Schema Induction

- History repeats itslef: Instance graphs (a) and (b) refer to very different event instances, but they both illustrate a same scenario
- We select salient and coherent paths based on Path Language Model, and merge them into graph schemas



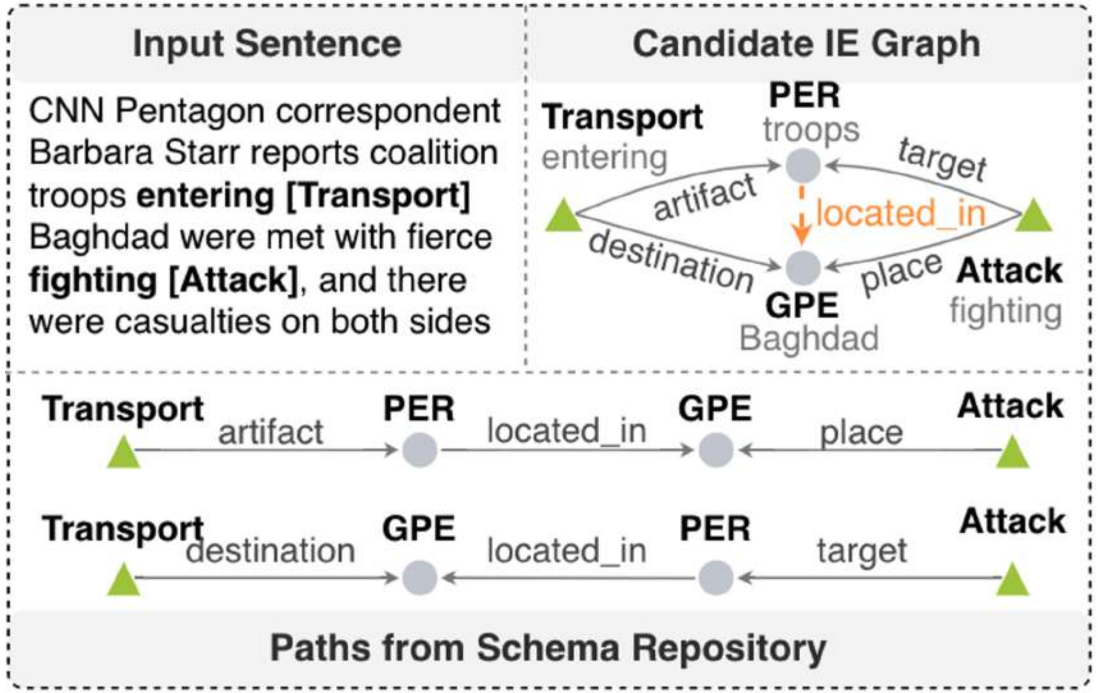
Path Language Model

- Path Language Model is trained on two tasks
  - Autoregressive Language Model Loss: capturing the frequency and coherence of a single path
  - Neighbor Path Classification Loss: capturing co-occurrence of two paths



Schema-Guided Information Extraction

- Use the state-of-the-art IE system OneIE (Lin et al, 2020) to decode converts each input document into an IE graph
- Each path in the graph schema is encoded as a single global feature for scoring candidate IE graphs
- OneIE promotes candidate IE graphs containing paths matching schema graphs
- http://blender.cs.illinois.edu/software/oneie
- F-scores (%) on ACE2005 data [Lin et al., ACL2020]:

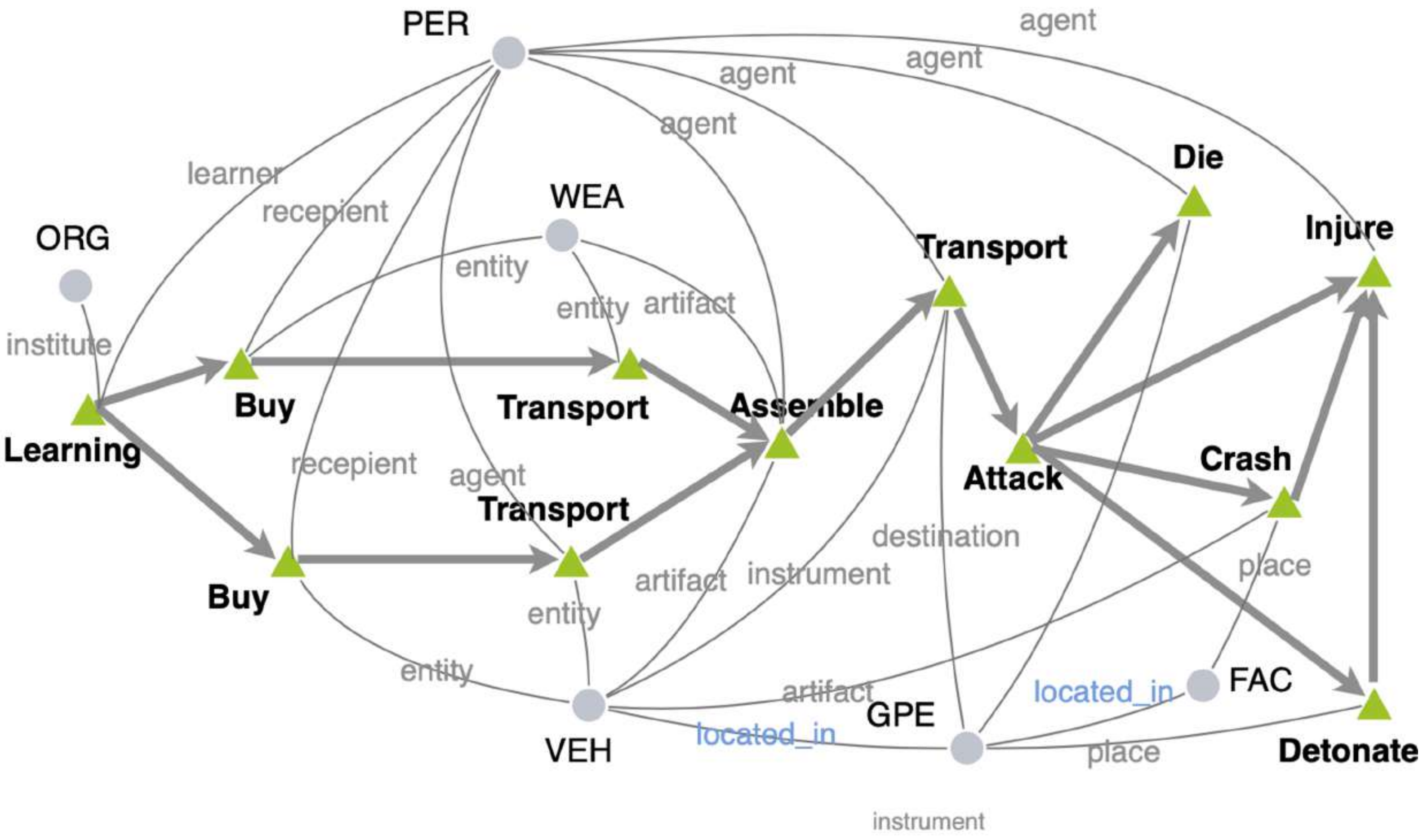


Dataset	Entity	Event Trigger Identification	Event Trigger Classification	Event Argument Identification	**Event Argument** Classification	Relation
Baseline	90.3	75.8	72.7	57.8	55.5	44.7
+PathLM	90.2	**76.0**	**73.4**	**59.0**	**56.6**	**60.9**

Temporal Complex Event Schema Composition

Graph4NLP

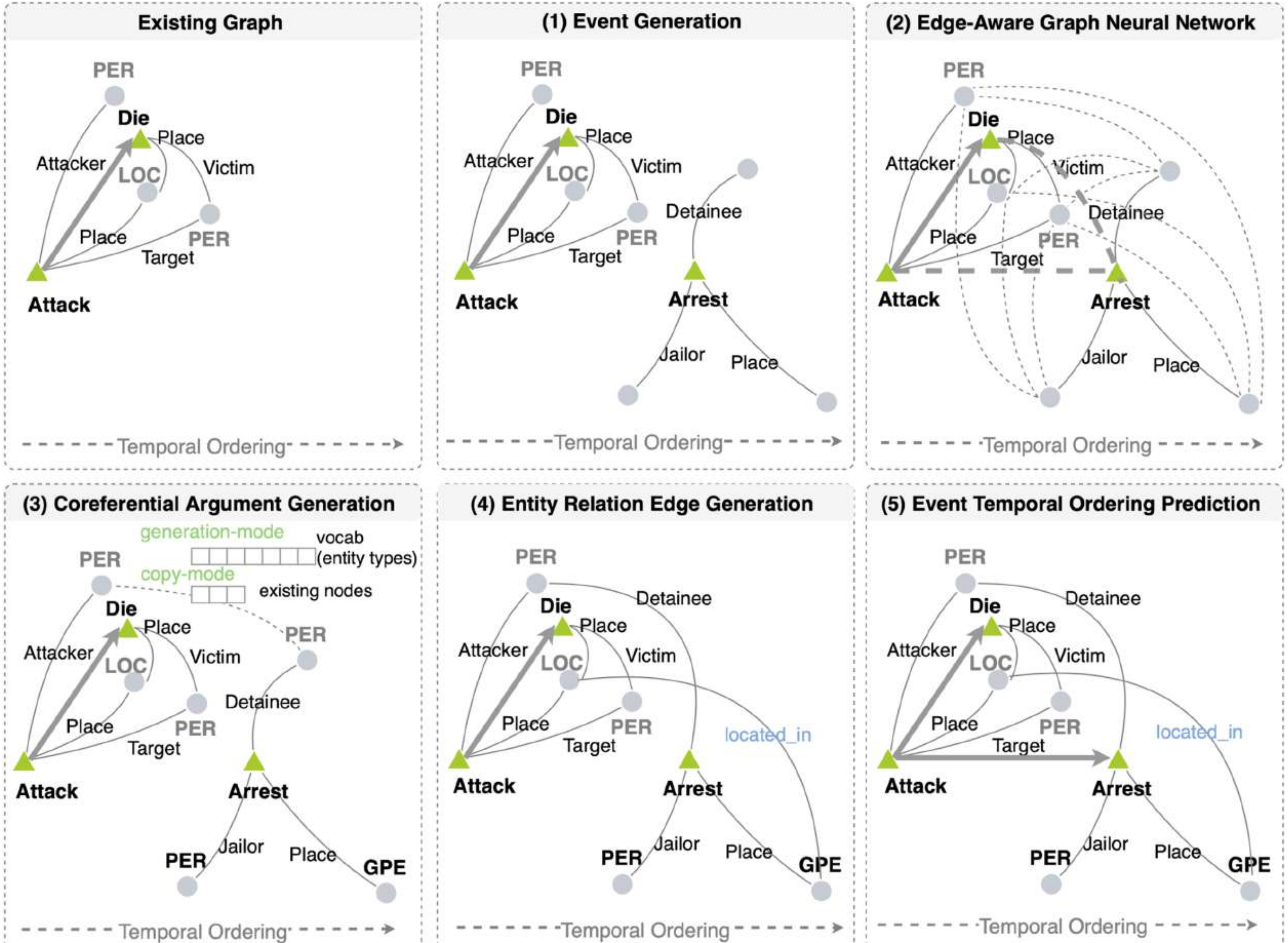
- Graph Structure Aware:
  - Encode entity coreference and entity relation
  - Capture the interdependency of events and entities (sequences can not)
- Scenario guided:
  - Train one model based on instance graphs of the same scenario
- Probabilistic:
  - Support downstream tasks, such as event prediction



Generative Event Graph Model

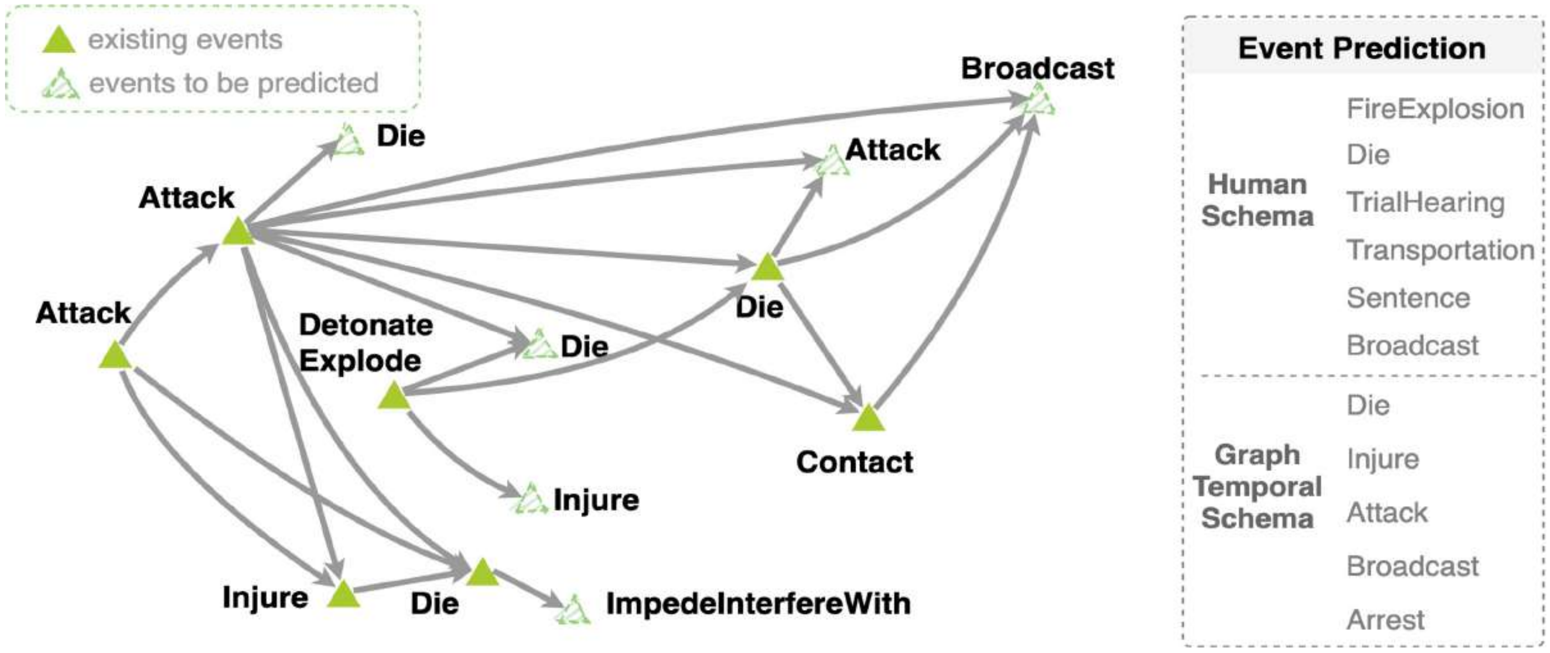
- Schemas are the hidden knowledge to control instance graph generation

- Step 1.
  Event Node Generation
- Step 2.
  Message Passing
- Step 3.
  Argument Node Generation
- Step 4.
  Relation Edge Generation
- Step 5.
  Temporal Edge Generation



Schema-guided Event Prediction

- **Schema-guided Event Prediction:** The task aims to predict ending events of each graph.
  - Considering that there can be multiple ending events in one instance graph, we rank event type prediction scores and adopt MRR and HITS@1 as evaluation metrics.



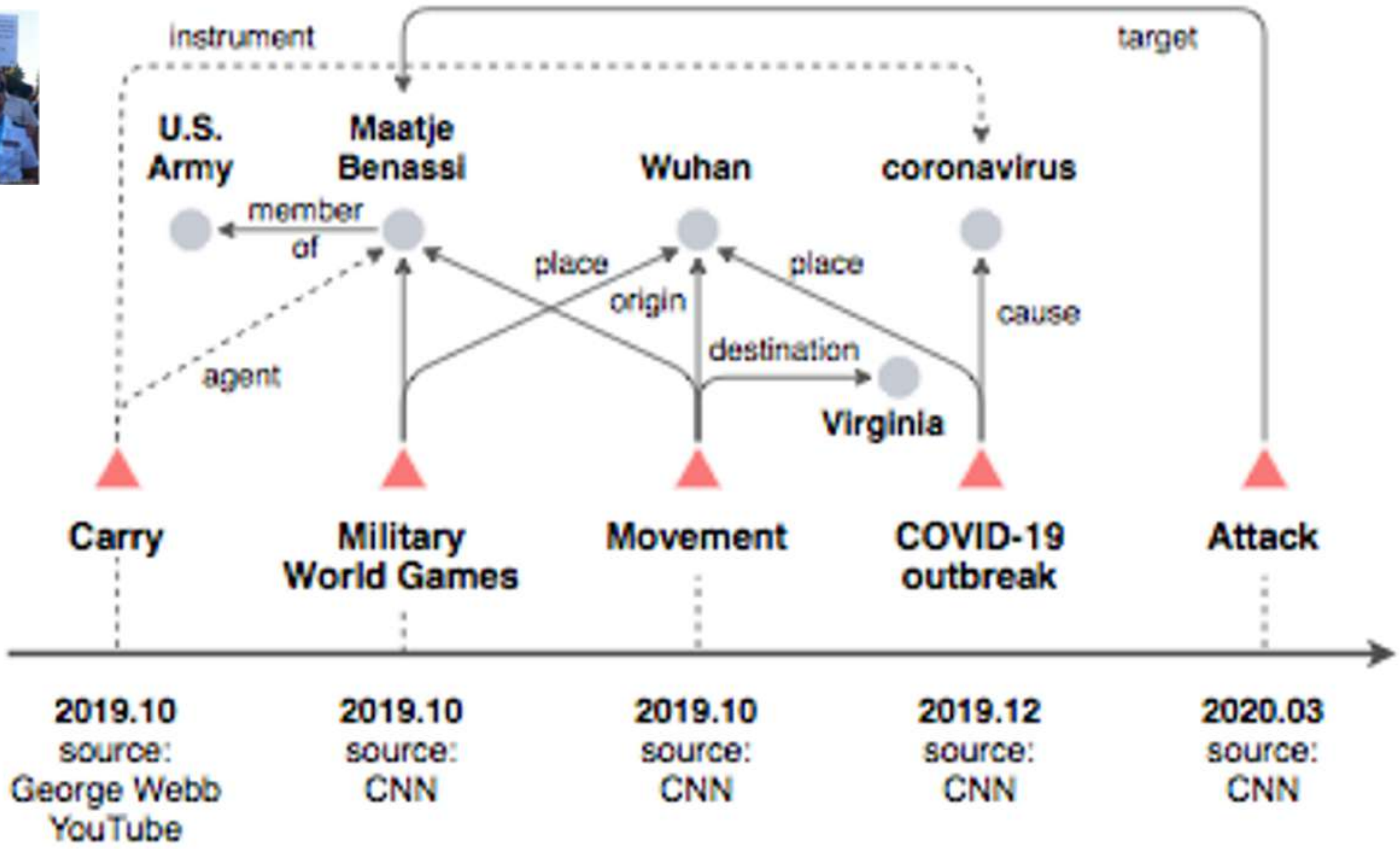
Dataset	Models	MRR	HITS@1
General	Human Schema	0.173	0.205
	Event Graph Model	**0.457**	**0.591**

Dataset	Models	MRR	HITS@1
IED	Human Schema	0.072	0.222
	Event Graph Model	**0.203**	**0.426**

Outline

- Semantic Graph Parsing for Event Extraction
- Cross-lingual structure transfer for Relation Extraction and Event Extraction
- Cross-media Structured Common Space for Multimedia Event Extraction
- Graph Schema-guided Event Extraction and Prediction
- Cross-media Knowledge Graph based Misinformation Detection

Information Pollution



- Why would anyone ever believe these rumors?
- Because humans are very good at connecting dots
- And perhaps too good →

Quiz Time! Which one is Fake News?

Burma's once-outlawed National League for Democracy is holding its first party congress since the opposition group was founded 25 years ago. Delegates in Rangoon will draw up a policy framework and elect a central committee during the three-day meeting that began Friday. Democracy icon Aung San Suu Kyi is also expected to be reappointed as head of the party. The Nobel laureate helped the NLD to a strong showing in historic April by-elections, which saw the party win 43 of the 45 contested seats. But the NLD is setting its sights on 2015, when it hopes to take power during national elections. But the party faces several challenges as it attempts to fashion itself into a viable political alternative to the military, which still dominates parliament and other government institutions. One of the most pressing issues is electing younger leaders to replace the party's elderly founding members, many of whom are in their 80s or 90s and in poor health.

Congress delegates prepare to pose for photographs as they arrive to attend the National League for Democracy party's (NLD) congress in Rangoon, March 8, 2013.

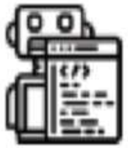
Delegates from the NLD gather in Rangoon for the party's annual congress. The NLD is headed by Nobel Peace Prize winner Aung San Suu Kyi. The party is expected to win a majority of seats in the parliament.

This year's NLD Congress is the first time the party has been able to elect its own leadership. Nyan Win, a member of NLD's executive committee, told VOA that the party is looking forward to the new generation of leaders.

The party has come a long way since the military seized power in 1962. The NLD was founded by a Briton. Since then, Burma has been ruled by a quasi-civilian government. However, the military has still maintained tight control over the country's political institutions. Phil Robertson, Asia director for Human Rights Watch, said he hopes the party will push forward with reforms that will allow the army to step down and allow the civilian government to take over.

Quiz Time! Which Caption is Fake?

On 24 May 2017 the Philippines militants left their barrack in the outskirts of southern Marawi city to reinforce fellow troops who had been under siege by Islamic troops.



Philippine troops arrive at their barracks to reinforce fellow troops following the siege by Muslim militants, on the outskirts of Marawi city in the southern Philippines, May 24, 2017.

Quiz Time! Which Caption is Fake?

Anis Amri (L), the Tunisian suspect of the Berlin Christmas market attack, is seen in this photo taken from security cameras at the Milan Central Train Station in downtown Milan, Italy December 23, 2016.

Anis Amri, a Tunisian suspected of defending the Christmas market in Milan, was seen in this photo given from a security camera at the Central Train Station of downtown Berlin on 23 December 2016 .

Knowledge Element-Level Misinformation Detection [Fung et al., ACL2021]

Graph4NLP

Motivation: misinformative parts of a fake news article lie along the fine-grained details

Current Issues:

- Fake news detection approaches tend to focus on checking facts, semantic inconsistencies, style or bias, lacking a *unified framework*.
- The document-level detection task lacks *precision* and *explainability*

Ex of Grover-Generated Fake News - News Spoofing

Hong Kong declared Independence from China Yesterday

- February 19, 2021

In a historic decision made yesterday, Hong Kong declared its independence from mainland China. The Senate of Hong Kong, the local government's legislative body, passed the inaugural Resolution of Independence after members of all races, sects and ages gathered in the senate chambers... "As the Chief Executive Council today endorsed the proposal of the Chief Executive Council to confirm the first proposed Resolution of Independence, Hong Kong is determined to complete the path of self-rule," said London-based broadcaster CNN yesterday... "We look forward to the motion being made by the Legislative Council and to firmly reaffirming our commitment to a prosperous and stable life of our people, while working together with China," Hong Kong's Chief Executive, Carrie Lam, said in a statement, according to AFP.

factually incorrect

awkward linguistic style

semantic drift

Compare with Previous Work

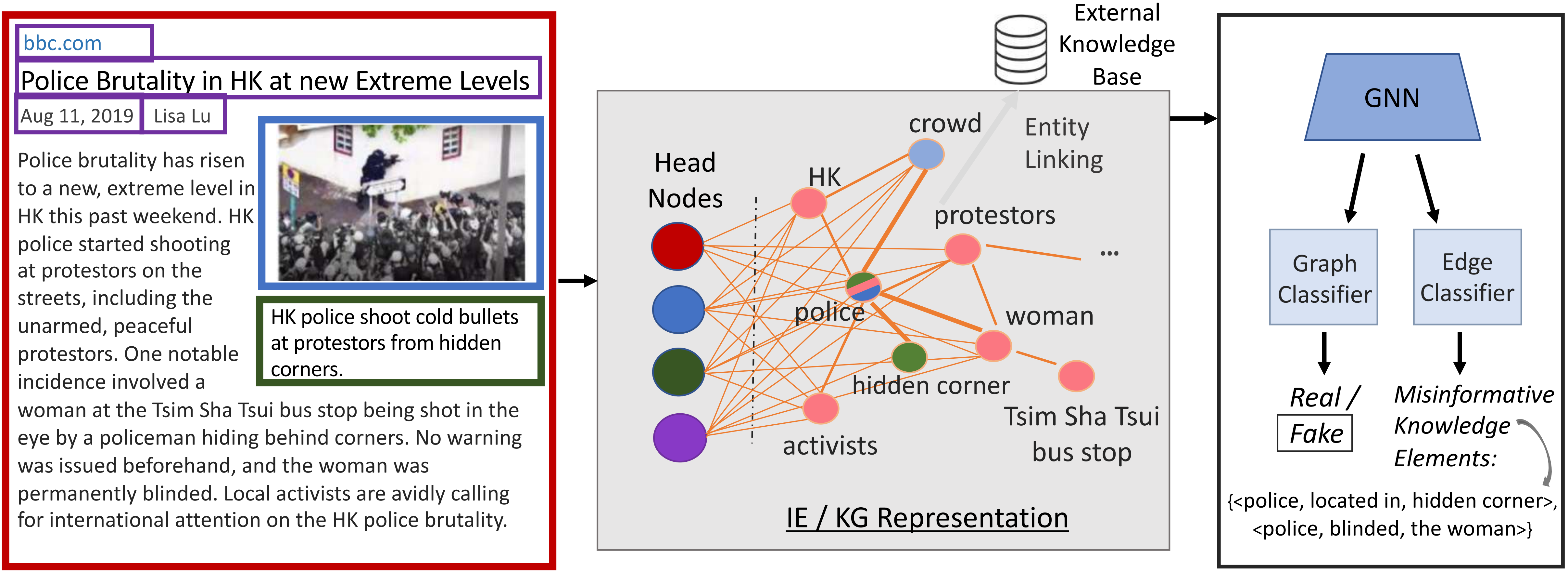
- <u>Motivation</u>: misinformative parts of a fake news article lie along the fine-grained details
  - Existing approaches lack a *unified framework* in checking facts, semantic inconsistencies, text features and bias

	Text Features	Structured Knowledge	Source Bias	Multimedia	Knowledge Element Level Detection
Perez-Rosas et al. (2018)	✓				
Pan et al. (2017)		✓			
Baly et al. (2018)	✓		✓		
Zellers et al. (2019)	✓		✓		
Tan et al. (2020)	✓			✓	
InfoSurgeon (Ours)	✓	✓	✓	✓	✓

Comparison with related work on fake news detection

Knowledge Element-Level Misinformation Detection

- Combine *local* and *global* features
- Leverage external knowledge to help pinpoint misinformation

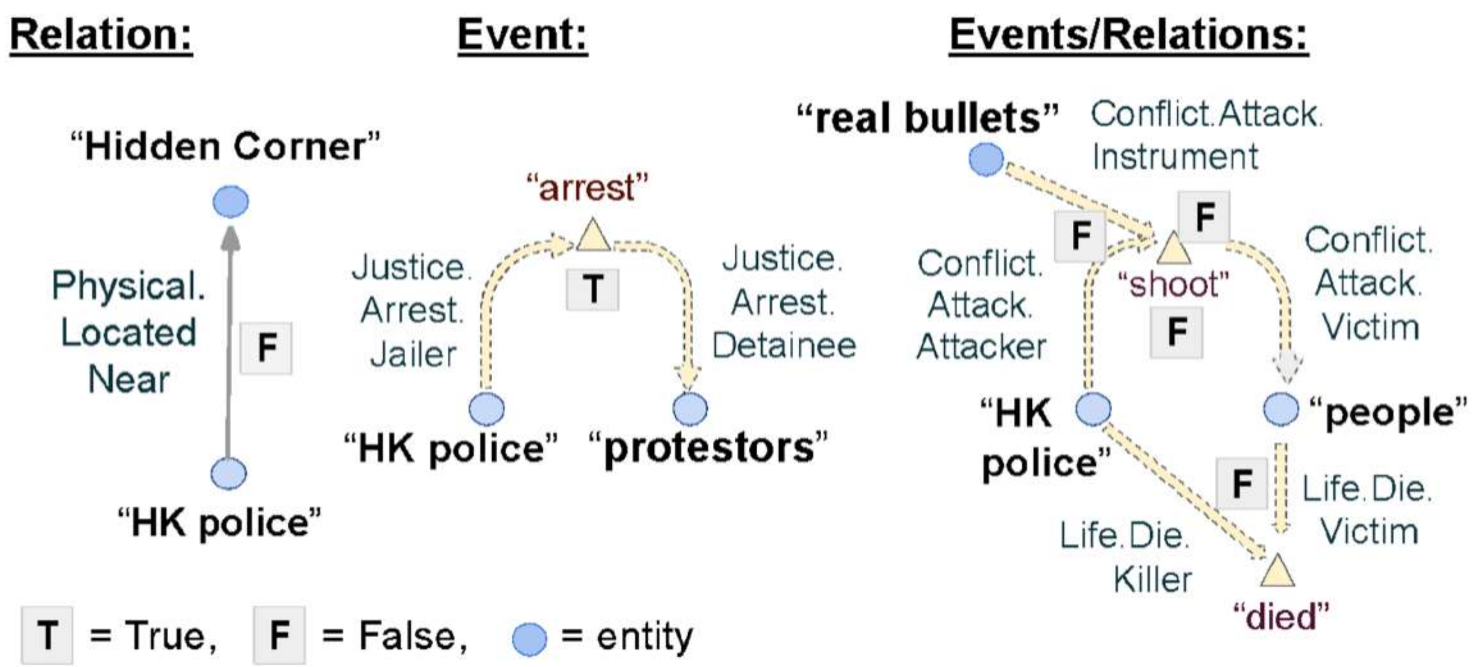


InfoSurgeon

Knowledge Element-Level Misinformation Detection

We also propose a new task in addition to document-level fake news detection that is more challenging but interesting.

Label each triplet connecting two entities as True/False



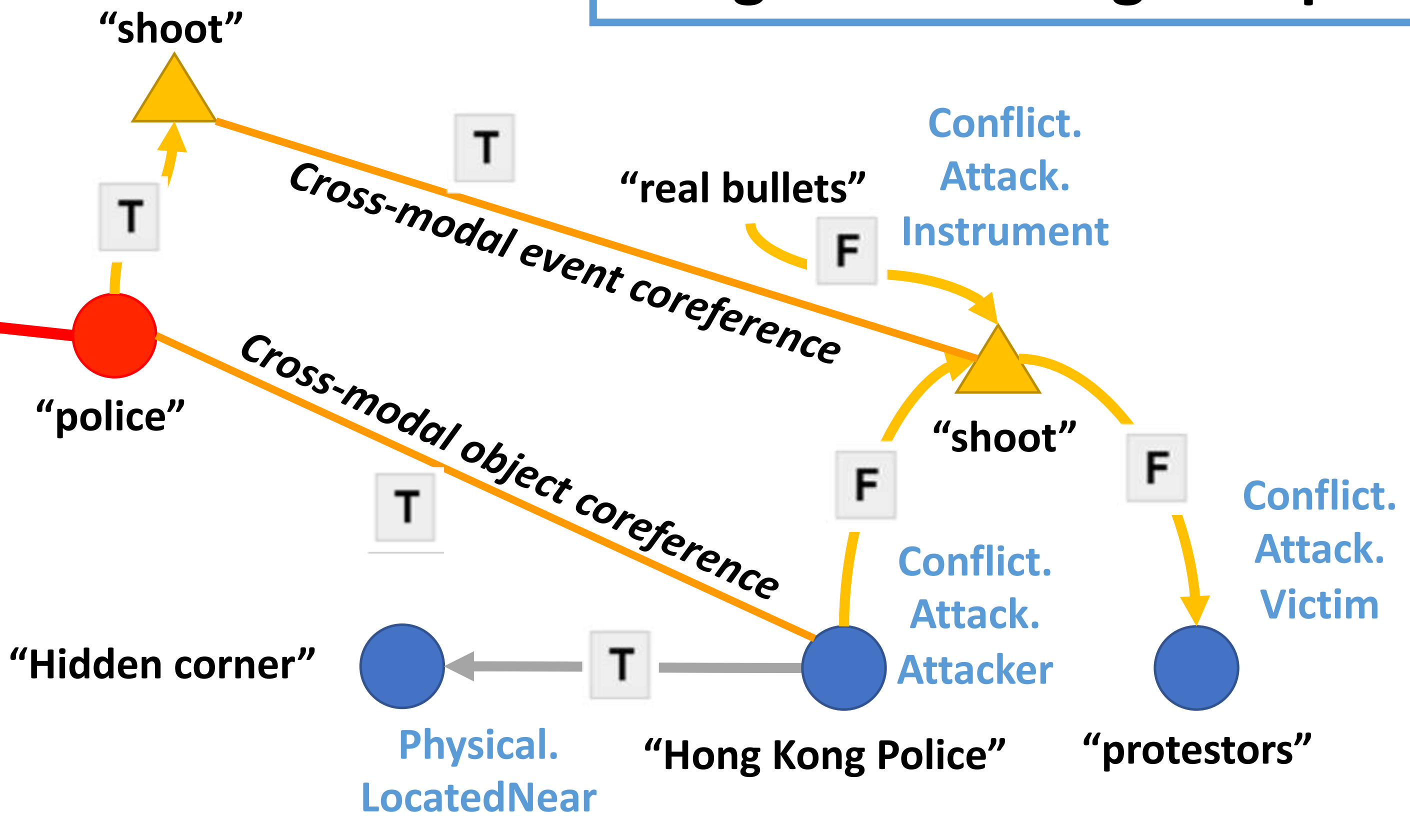
Multimedia Knowledge Graph Construction

Graph4NLP

Merged Knowledge Graph

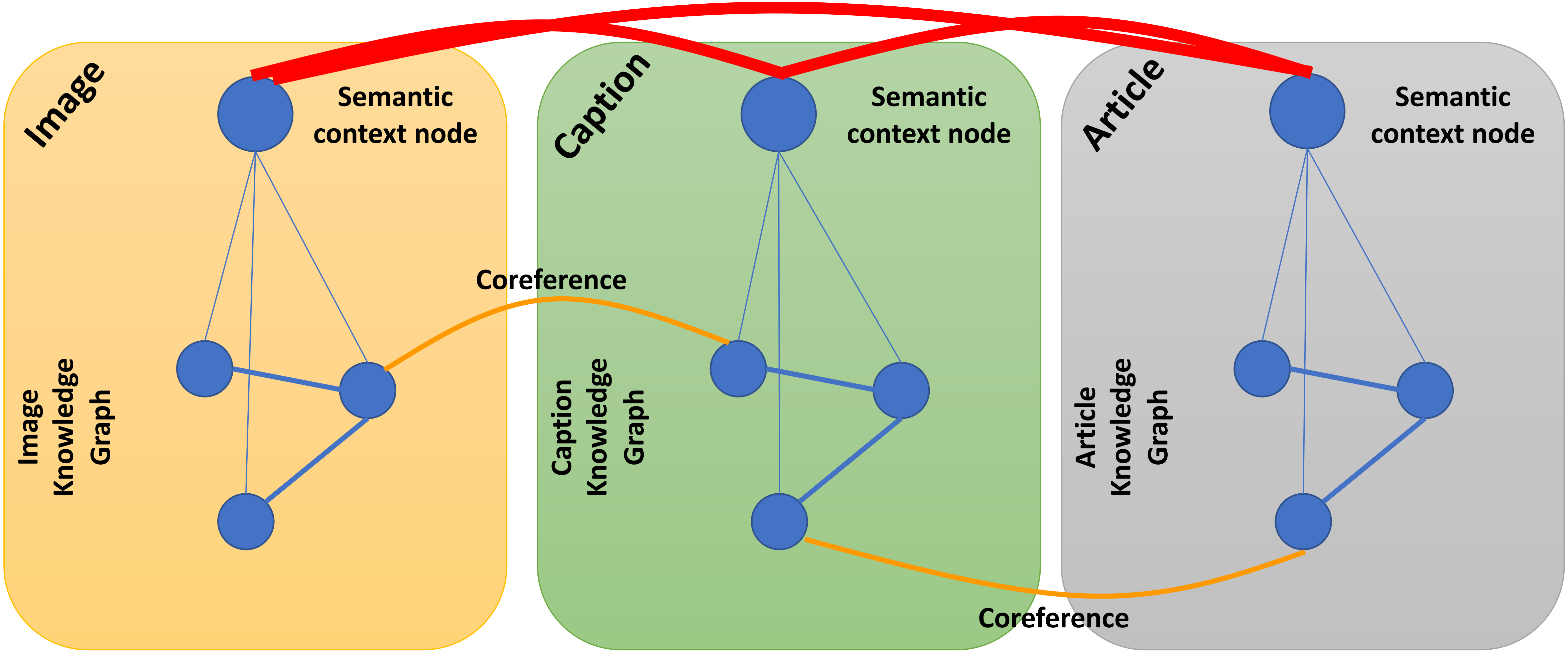
Image Caption

Hong Kong police shoot real bullets at protestors from hidden corners.



T = True, F = False, ● = entity ▲ = event trigger

Graph Propagation



Yi Fung, Christopher Thomas, Revanth Gangi Reddy, Sandeep Polisetty, Heng Ji, Shih-Fu Chang, Kathleen McKeown, Mohit Bansal and Avi Sil

"InfoSurgeon: Cross-Media Fine-grained Information Consistency Checking for Fake News Detection". *In submission to ACL 2021*

KG-Conditional Fake News Generation

- Annotating specifically which elements in a KG are fake is time-intensive / difficult
- We propose a solution to <u>automatically obtain knowledge-element labeled knowledge graphs for free</u>
- Given a set of real news articles, we extract KGs from the real articles
- Train a text generator model that learns to recreate an article from its KG
- To generate fake data, perform manipulation operations on the KG (editing knowledge relations, events, entities, etc.) to produce KG’
- Generate a fake article from KG’
- Key insight – We now know specifically which elements in KG’ were manipulated!

Manipulated KG-to-Article Synthesis

- We perform the following manipulations on KGs:

- **Entity swapping** – Swapping entity that has same type and similar embedding (so they are harder to tell apart)

- **Addition of new relation or event** – Randomly select relation / event argument roles and append a new entity to the relation / event

- **Subgraph replacement** – Select a subgraph of the news article from an entity and replace it with a subgraph from another news article

Generating Text from Structured Representations

Graph4NLP

No Cross-media Manipulation Constraints

Article Image

Manipulated KG

<team,Manufacture.Artifact_Artifact.Artifact_Manufacturer, Zambia> <Zambia,Manufacture.Artifact_Manufacturer, men> ...
<~~bicycle~~> <**truck**> <hospital>

Generated Article

The Zambia-based Bamboo **truck** is the first of its kind in the world, and it's designed to be a cheaper and lighter alternative to the traditional, heavy-duty, imported, metal ambulance. The Zambian-designed **truck** is built of bamboo, which is an environmentally friendly material...

- We manipulate knowledge graphs to synthesize fake news which contain known types of inconsistencies.
- This example is trivially detectable due to a inconsistency with the image

Generating Text from Structured Representations

With Cross-media Manipulation Constraints

Article Image

Manipulated KG

<team,Manufacture.Artifact_Artifact.Artifact_Manufacturer, ~~Zambia~~**Fiji**>
<~~Zambia~~**Fiji**,Manufacture.Artifact_Manufacturer, men> ...
<**bicycle**> <hospital> ...

Generated Article

A team of two Californians living in **Fiji** is trying to build the world's smallest and most affordable bicycle. They are using bamboo as the frame for their bicycles. The team is made up of 25 young men who met at a university in the Pacific island nation of **Fiji**. They're using their...

- By imposing cross-media knowledge graph manipulation constraints, we **prevent generating text with obvious inconsistencies**.
- Enables generating more realistic / challenging data for training detector

Caption Manipulation – AMR-to-Caption Synthesis

- Use text parser to get AMR graphs (Banarescu et al., 2013) from captions
- Use AMR since they capture fine-grained relations expressing who does what to whom
- Manipulations:
  - **Role switching –** Swapping entity positions in AMR graph
  - **Predicate negation –** Replace triggers / verbs with antonyms from WordNet
- Use off-the-shelf model for AMR to text synthesis (Ribeiro et al, 2020)

True Caption:
In Afghanistan, the Taliban released to the media this picture, which it said shows the suicide bombers who attacked the army base in Mazar-i-Sharif, April 21, 2017

Fake caption:
On 21 April 2017 the Taliban released this picture to the army in Afghanistan which they said was a suicide bomber hiding at a media base in the city of Mazar-i-Sharif

- **Ethical Statement: we are not going to share our generator, but sharing our detector!**

Knowledge Element-Level Misinformation Dataset

- To address the lack of data for the detection task, we further contribute a KG2txt fake news generation approach, which allows for control over knowledge element manipulation and creating silver standard annotation data.

	Overall	Real Documents	Fake Documents
Human Detection Accuracy	61.3%	80.4%	42.3%

The Turing Test results above show that our automatically generated fake documents are also very hard for humans to detect.

Knowledge Element-Level Misinformation Detection

Experimental result on traditional document-level detection:

	NYTimes Neural News Dataset	VOA Manipulated KG2Txt Dataset
Grover	56.0%	86.4%
DIDAN	77.6%	88.3%
InfoSurgeon (Our Model)	**94.5%**	**92.1%**

Experimental result on the novel task, knowledge element level misinformation detection:

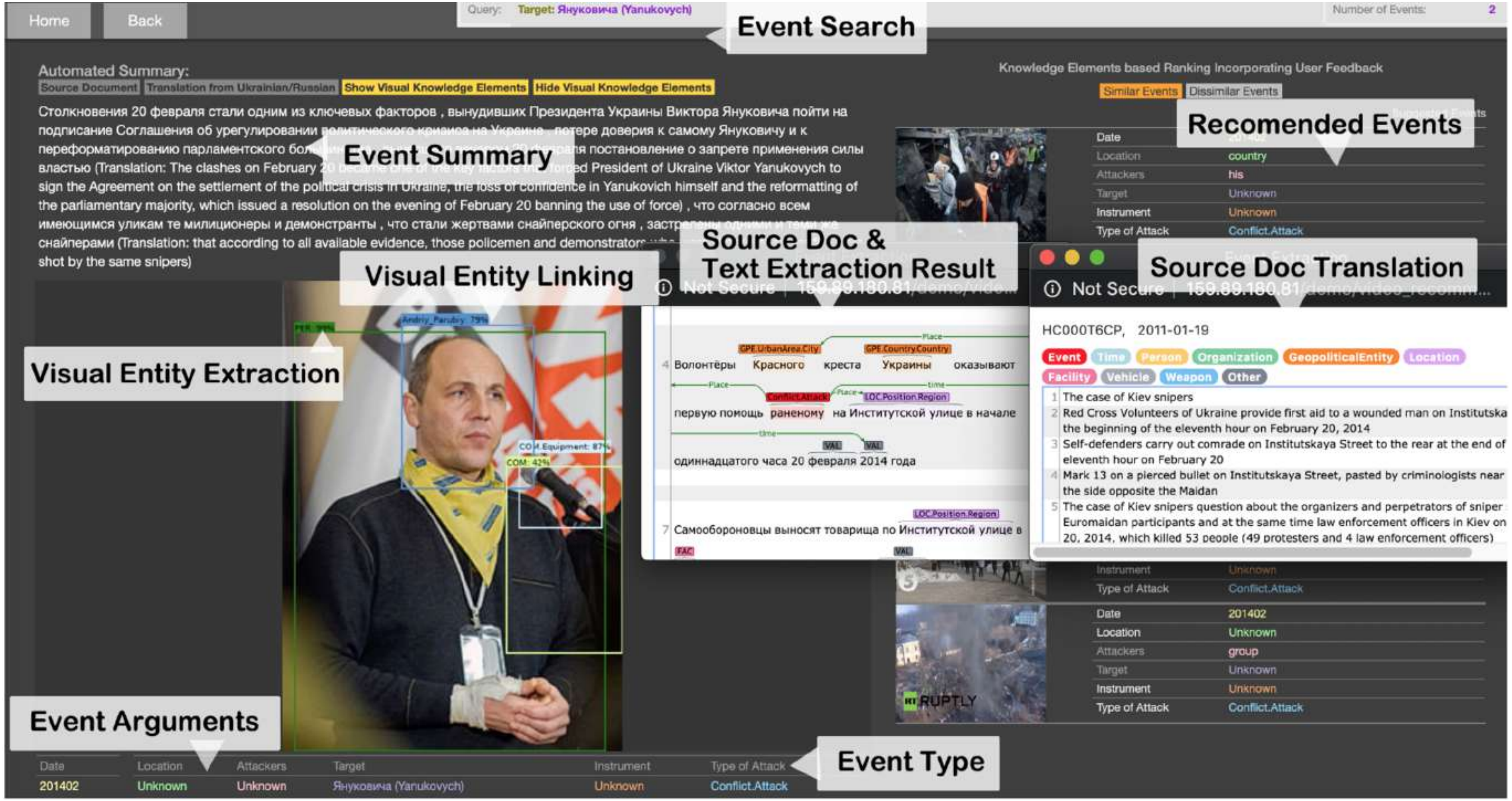
	VOA Manipulated KG2Txt Dataset
Random (baseline)	27%
InfoSurgeon (Our Model)	**37%**

A Successful Example

- Example of fake news article in which baseline misses, but *InfoSurgeon* successfully detects

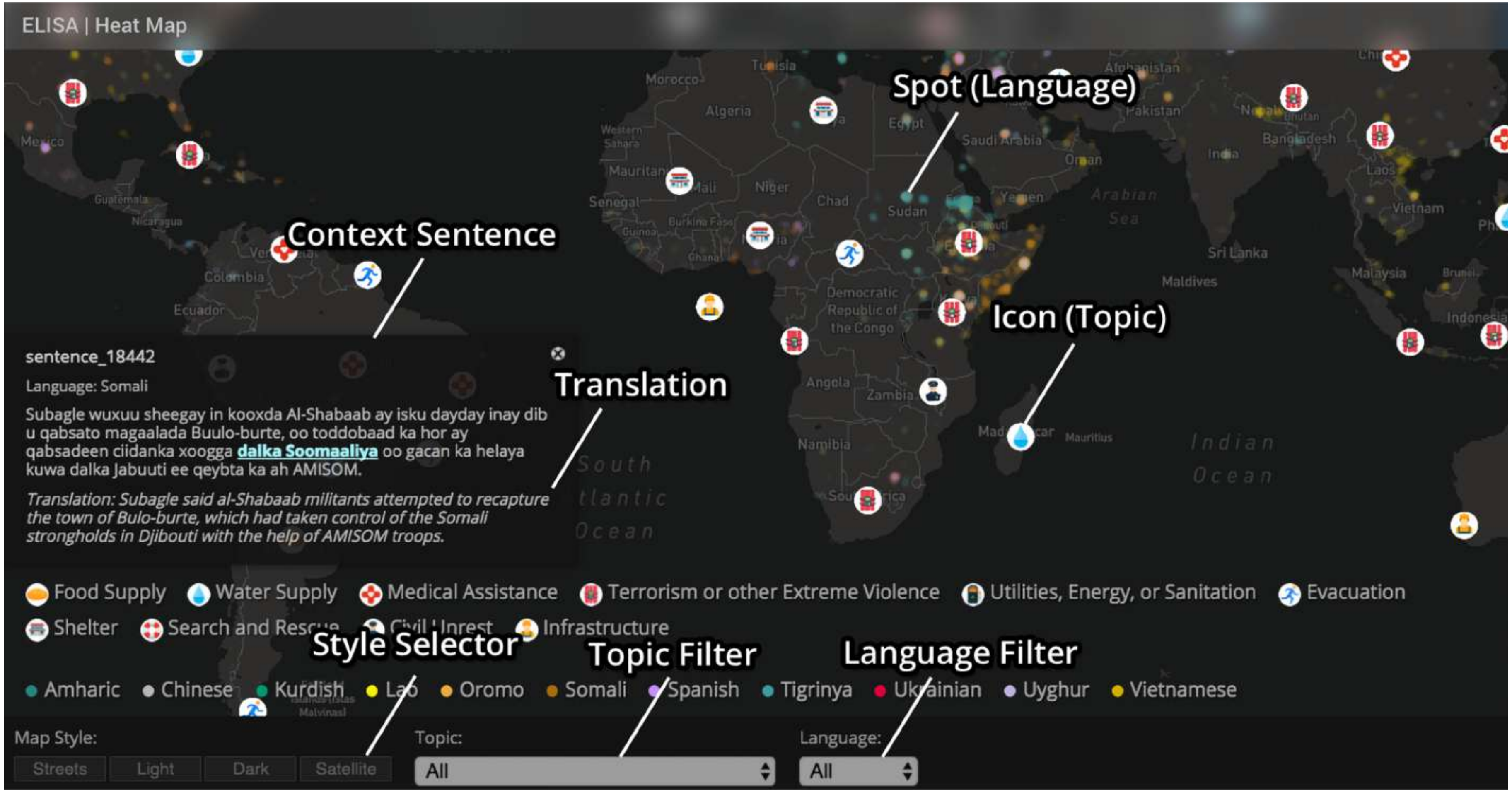
Image	Caption	Body Text	Misinformative KEs
	*Aerial view of Fort McHenry.*	*The battle of Fort McHenry, which took place in September of 1814, was a pivotal moment in the U.S. War of Independence...When the **British** finally left, they left behind a trail of destruction, including the destruction of the **twin towers** of the World Trade Center ...*	<**British**, Conflict.Attack, **twin towers**>

Demo 1: Multimedia Event Recommendation



(Li et al., ACL2020 Best Demo Paper Award)
GitHub: https://github.com/GAIA-IE/gaia
DockerHub: https://hub.docker.com/orgs/blendernlp/repositories
Demo: http://159.89.180.81/demo/video_recommendation/index_attack_dark.html

Demo 2: Event Heatmap for Disaster Relief



- Re-trainable Systems: http://159.89.180.81:3300/elisa_ie/api
- Demos: http://159.89.180.81:3300/elisa_ie
- Heat map: http://159.89.180.81:8080/

Software and Resources

- KAIROS RESIN Cross-document Cross-lingual Cross-media Information Extraction system (Wen et al., NAACL2021 demo)
  - https://github.com/RESIN-KAIROS/RESIN-pipeline-public
- Joint Neural Information Extraction system (Lin et al., ACL2020)
  - http://blender.cs.illinois.edu/software/oneie/
- GAIA Multimedia Event Extraction system and new benchmark with annotated data set (Li et al., ACL2020 demo)
  - GitHub: https://github.com/GAIA-AIDA/uiuc_ie_pipeline_fine_grained
  - Text IE DockerHub: https://hub.docker.com/orgs/blendernlp/
  - Visual IE repositories: https://hub.docker.com/u/dannapierskitoptal

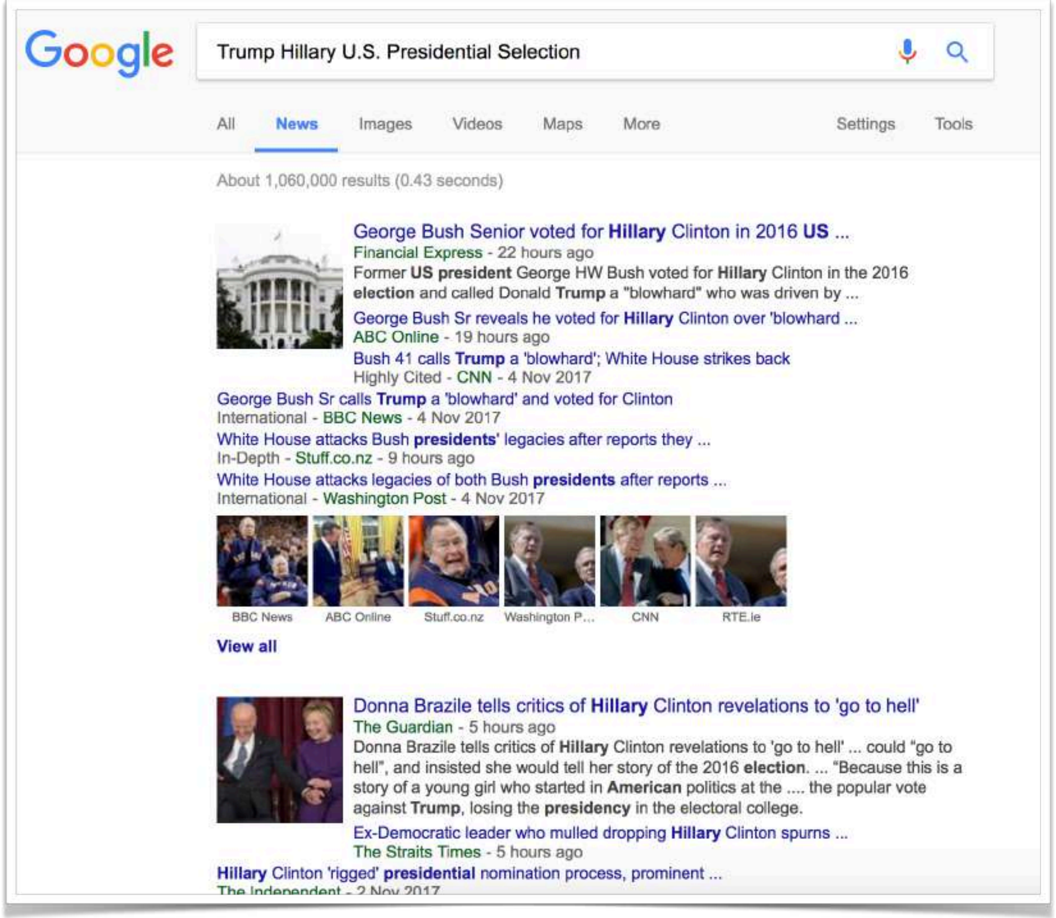
Text Clustering and Matching

Information Explosion

News Reading: Search Engines & Feeds Streams

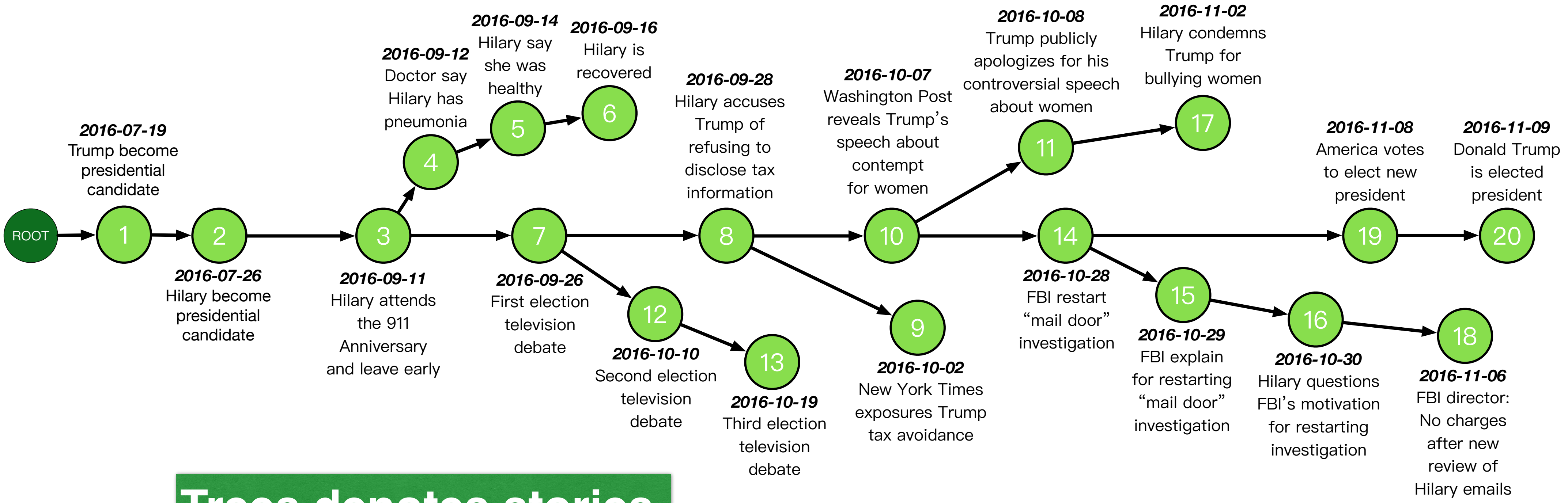
Disadvantages of existing systems

- Messed document lists
- Extremely fine-grained (articles)
- Redundant useless information
- Unstructured information



Story Forest

Detect events automatically from massive news articles

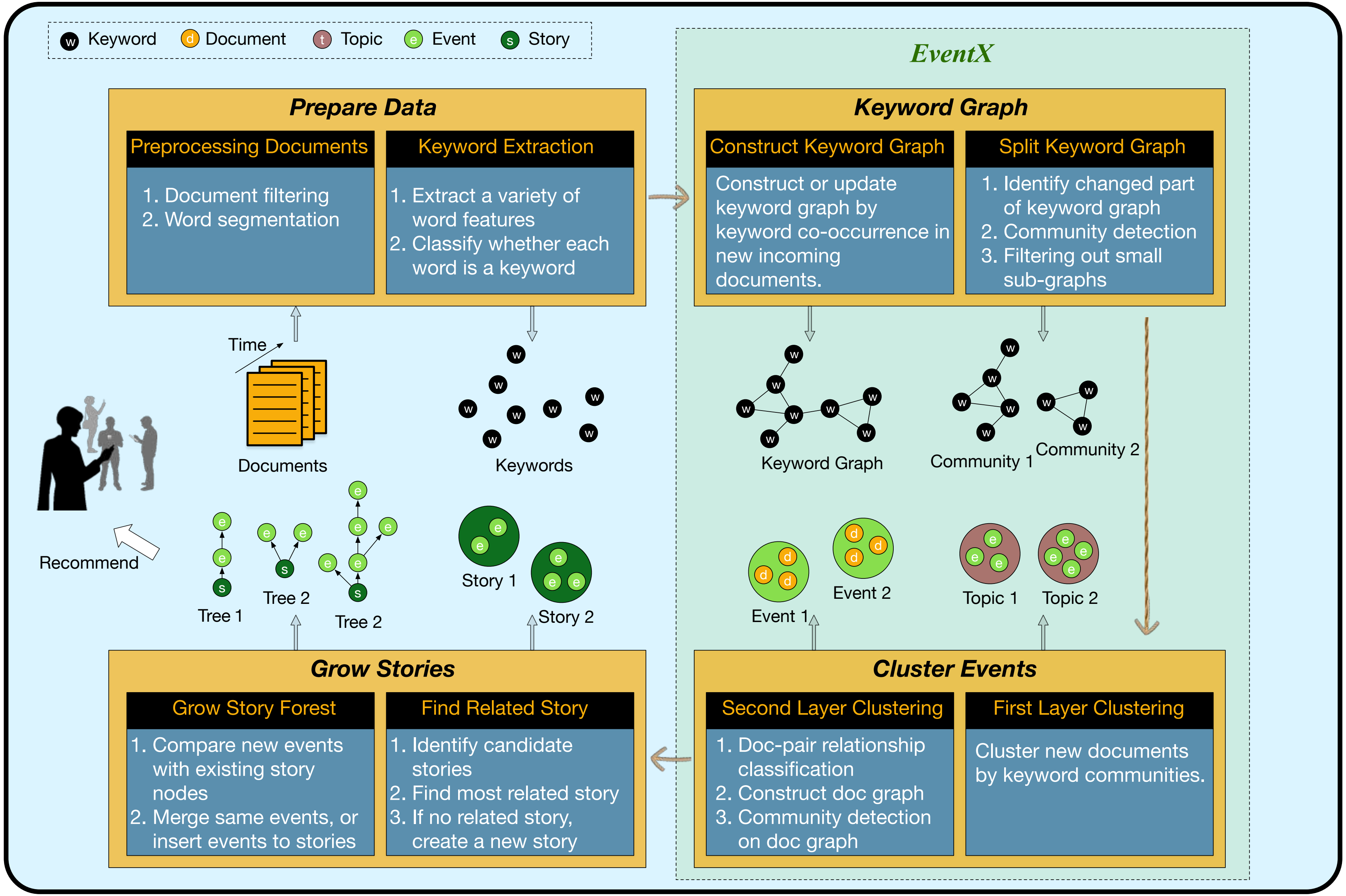


Trees denotes stories, nodes denotes events

Edges in the tree denotes events evolving relationship

Liu et al., **Story Forest: Extracting Events and Telling Stories from Breaking News**, TKDD'20

Story Forest System



Liu et al., **Story Forest: Extracting Events and Telling Stories from Breaking News**, TKDD'20

Applied to QQ browser hot topic list

Hawking public PhD thesis

Text Matching Tasks

Why Long Document Matching

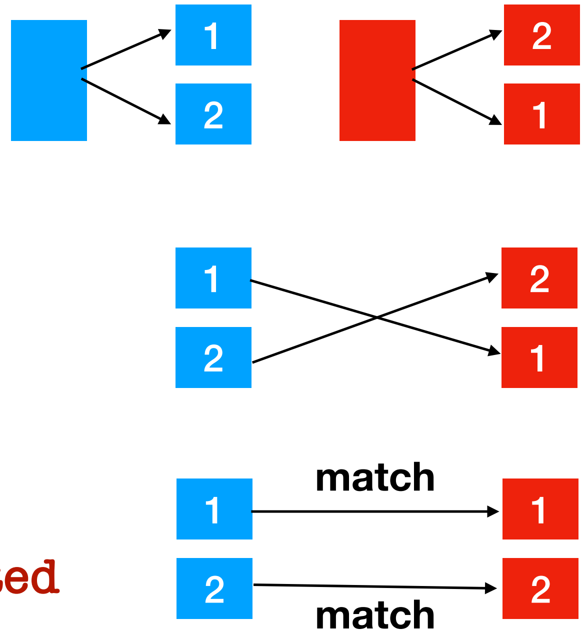
Identify the relationship between documents

Same event?
Related events?

Divide-and-Conquer

Our strategies

- ✦ Divide
- ✦ Align
- ✦ Distributed match

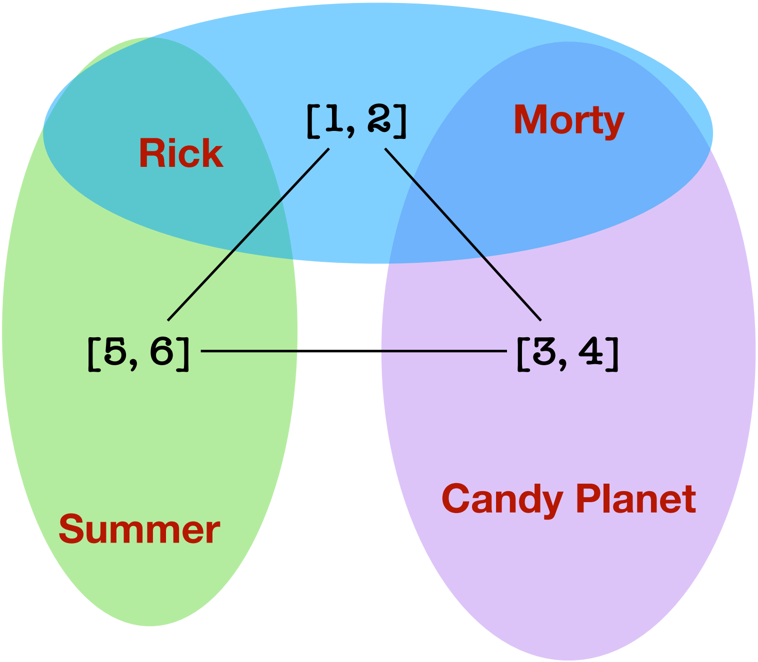


Limitations

- ✦ Hard to encode
- ✦ Flexible order
- ✦ Time complexity

Concept Interaction Graph

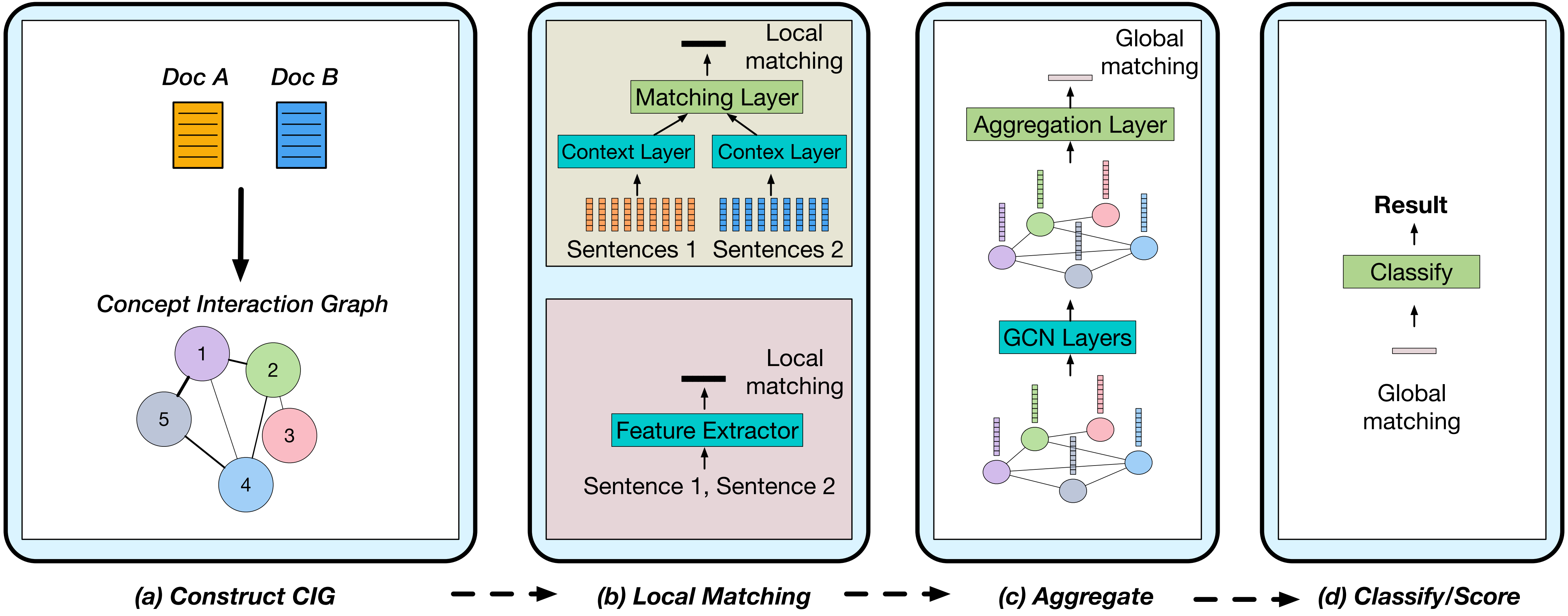
[1] Rick asks Morty to travel with him in the universe.
[2] Morty doesn't want to go as Rick always brings him dangerous experiences.
[3] However, the destination of this journey is the Candy Planet, which is a fascinating place that attracts Morty.
[4] The planet is full of delicious candies.
[5] Summer wishes to travel with Rick.
[6] However, Rick doesn't like to travel with Summer.



STEP 1: Extract keywords
STEP 2: Group keywords
STEP 3: Assign sentences
STEP 4: Construct edges

Liu et al., **Matching article pairs with graphical decomposition and convolutions**, ACL'19

Graph Decomposition for Document Matching



Liu et al., **Matching article pairs with graphical decomposition and convolutions**, ACL'19

Experiments

Baselines	CNSE Acc	CNSE F1	CNSS Acc	CNSS F1	Our models	CNSE Acc	CNSE F1	CNSS Acc	CNSS F1
I. ARC-I	53.84	48.68	50.10	66.58	**XI.** CIG-Siam	74.47	73.03	75.32	78.58
II. ARC-II	54.37	36.77	52.00	53.83	**XII.** CIG-Siam-GCN	**74.58**	**73.69**	**78.91**	**80.72**
III. DUET	55.63	51.94	52.33	60.67	**XIII.** CIG_{cd}-Siam-GCN	73.25	73.10	76.23	76.94
IV. DSSM	58.08	64.68	61.09	70.58	**XIV.** CIG-Sim	72.58	71.91	75.16	77.27
V. C-DSSM	60.17	48.57	52.96	56.75	**XV.** CIG-Sim-GCN	**83.35**	**80.96**	**87.12**	**87.57**
VI. MatchPyramid	66.36	54.01	62.52	64.56	**XVI.** CIG_{cd}-Sim-GCN	81.33	78.88	86.67	87.00
VII. BM25	69.63	66.60	67.77	70.40	**XVII.** CIG-Sim&Siam-GCN	84.64	**82.75**	89.77	90.07
VIII. LDA	63.81	62.44	62.98	69.11	**XVIII.** CIG-Sim&Siam-GCN-Sim^g	84.21	82.46	**90.03**	**90.29**
IX. SimNet	71.05	69.26	70.78	74.50	**XIX.** CIG-Sim&Siam-GCN-BERT^g	**84.68**	82.60	89.56	89.97
X. BERT fine-tuning	81.30	79.20	86.64	87.08	**XX.** CIG-Sim&Siam-GCN-Sim^g&BERT^g	84.61	82.59	89.47	89.71

✦ **Graph Representation:** greatly improves performance. (IX vs. XI) (+4% Acc, F1)

✦ **Graph Convolution**: greatly improves performance. (XIV vs. XV) (+10% Acc, F1)

Liu et al., **Matching article pairs with graphical decomposition and convolutions**, ACL’19

Text Mining and Classification

What are users interested in?

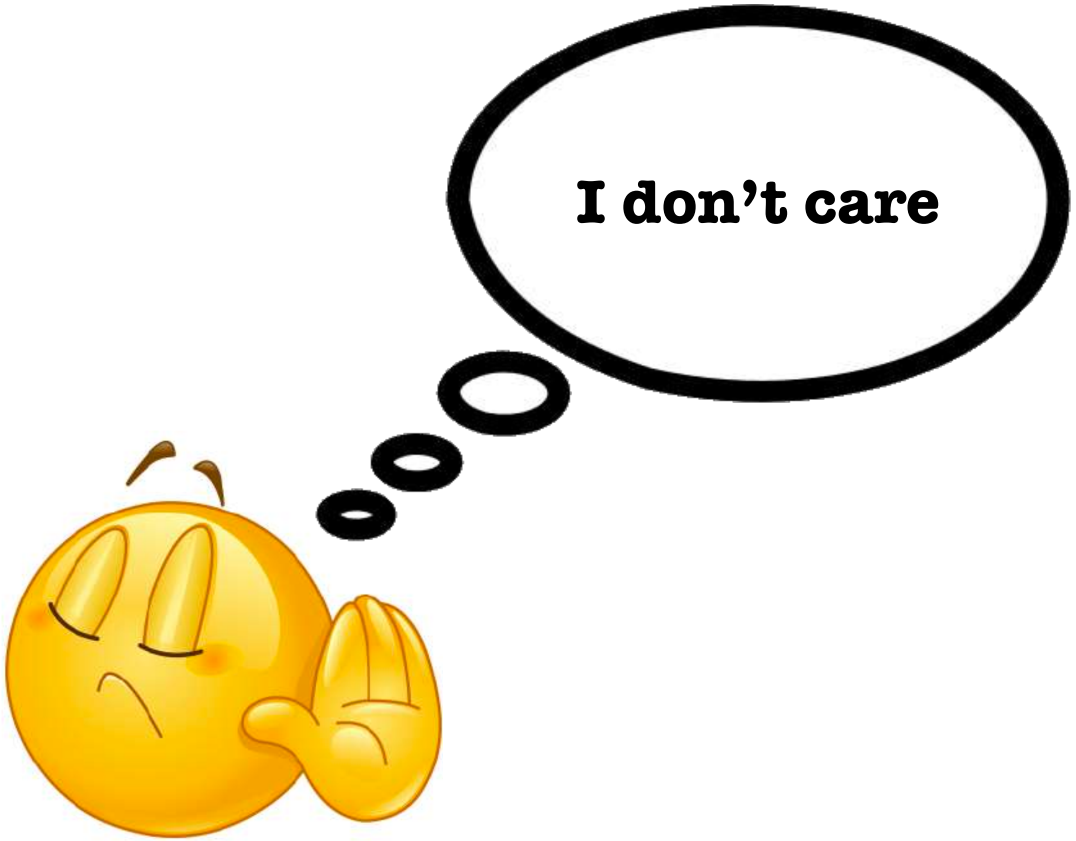
Query: "Theresa May's resignation speech"

Infer users' interests

What are users interested in?

Inaccurate recommendation

Articles about Theresa May



Query: “Theresa May’s resignation speech”

What are users interested in?

Monotonous recommendation

Articles about Theresa May's resignation speech

Query: "Theresa May's resignation speech"

What are users interested in?

Good recommendation

Articles about Brexit Negotiation

Query: “Theresa May’s resignation speech”

Graph4NLP

What we need?

User interests in a suitable granularity

Relationships between user interests

Brexit Negotiation

Theresa May's resignation speech

What do people care about: Events

Theresa May's resignation speech

Apple iPhone7 launch

Event: a real-world incident that involves specific persons, organizations, or entities, with a certain time/location of occurrence

What do people care about: Concepts

Honda Civic

Hyundai Elantra

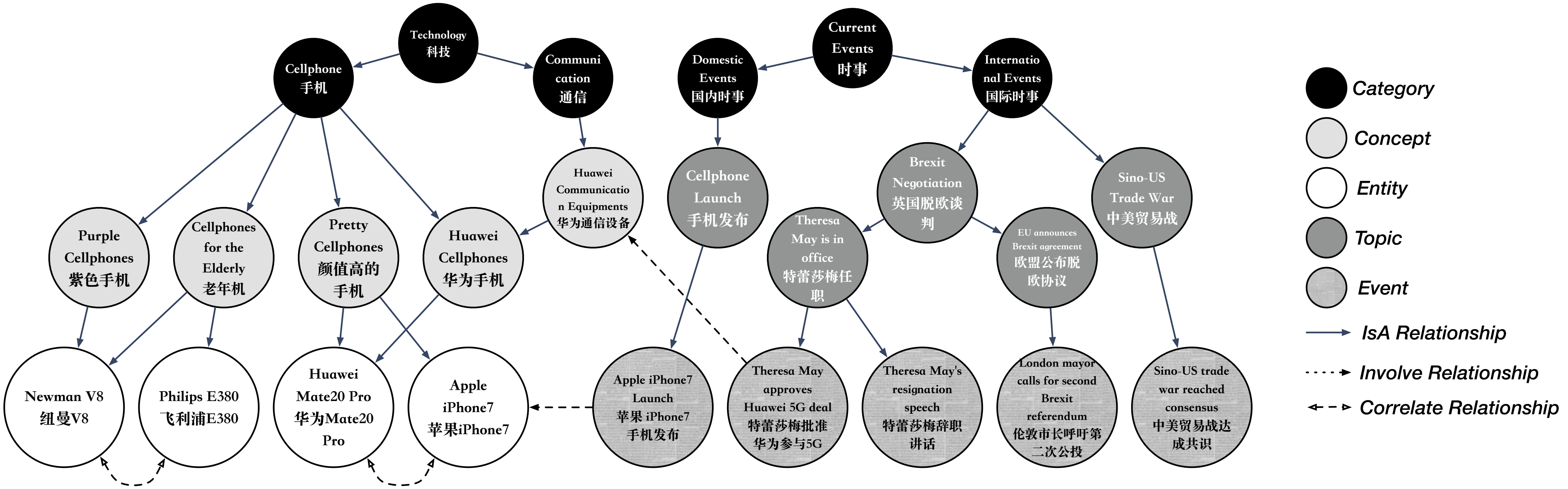
- **Fuel-efficient cars**
- **Economy cars**

- **Marvel heroes**
- **Revengers**

Concept: a collection of things that share some common attributes

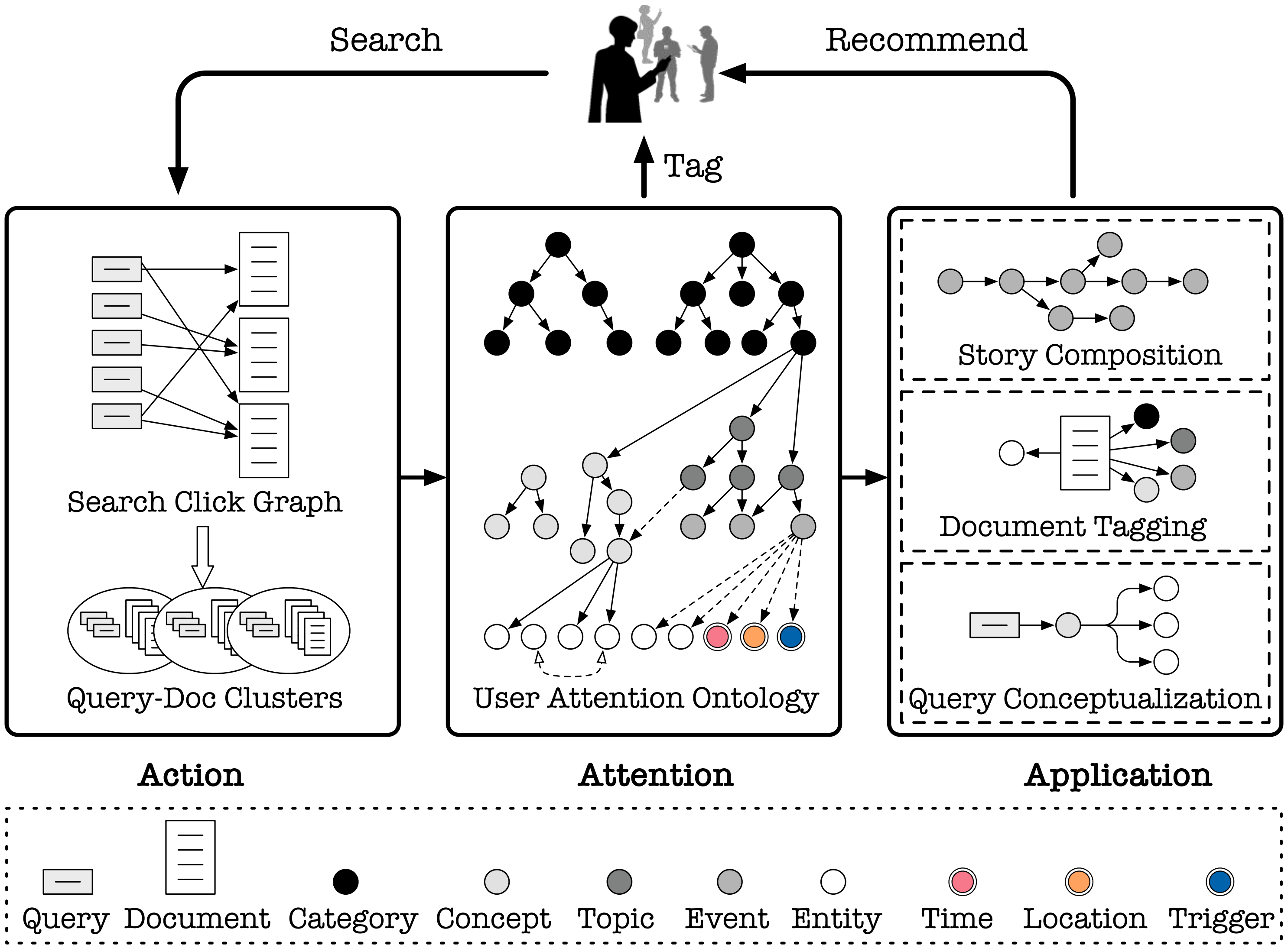
Attention ontology

Create a web-scale ontology to represent user interests and document topics.



Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

GIANT system



Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

Heterogeneous phrase mining

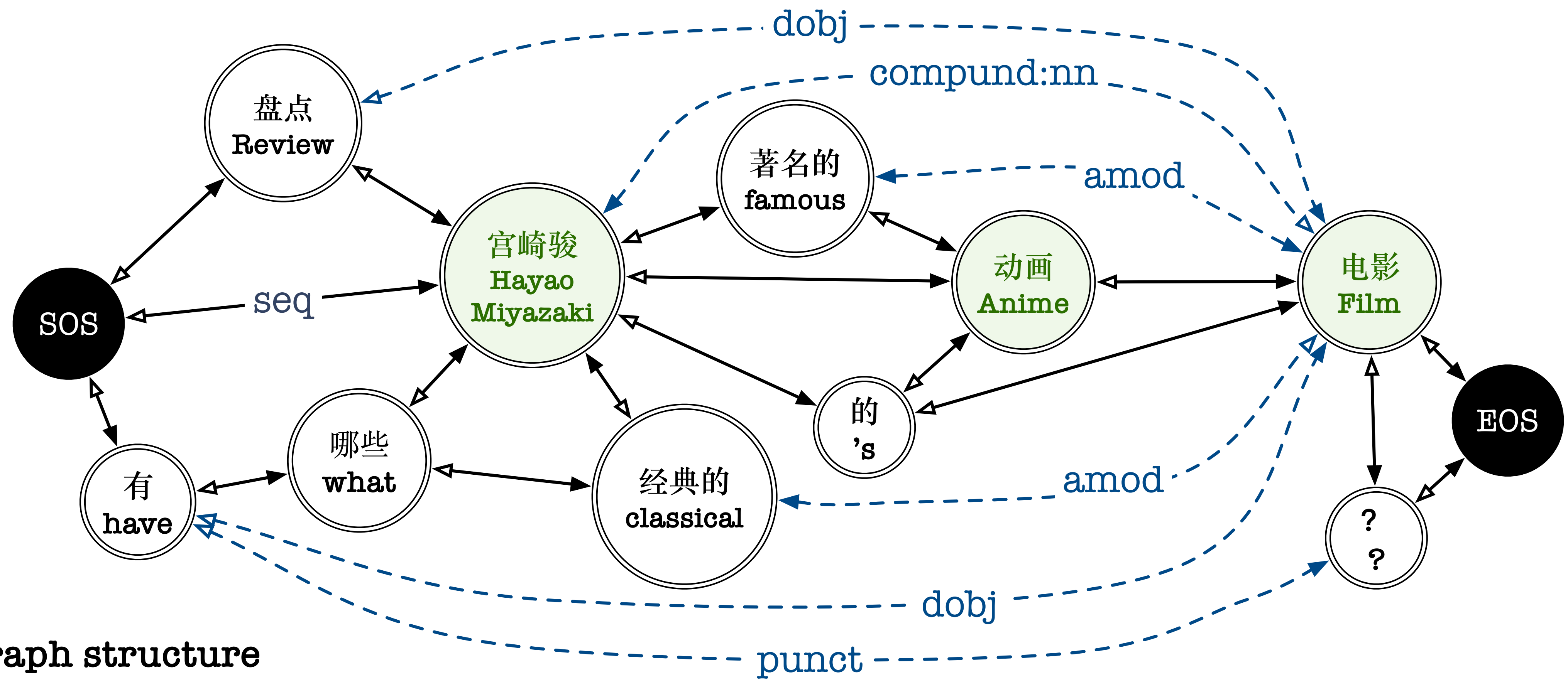
Query: What are the **Hayao Miyazaki**'s **animated film** (有 哪些 宫崎骏 的 动画 电影)
Titles: Review **Hayao Miyazaki**'s **animated film** (盘点 宫崎骏 动画 电影)
The famous **animated films** of **Hayao Miyazaki** (宫崎骏 著名的 动画 电影)
What are the classic **Miyazaki**'s movies? (有 哪些 经典的 宫崎骏 的 电影？)
Concept: **Hayat Miyazaki animated film** (宫崎骏 动画 电影)

Characteristics of output words

Patterns
Show up multiple times
NER/Part-of-Speech tags
Continuous chunk
Syntactic dependency

Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

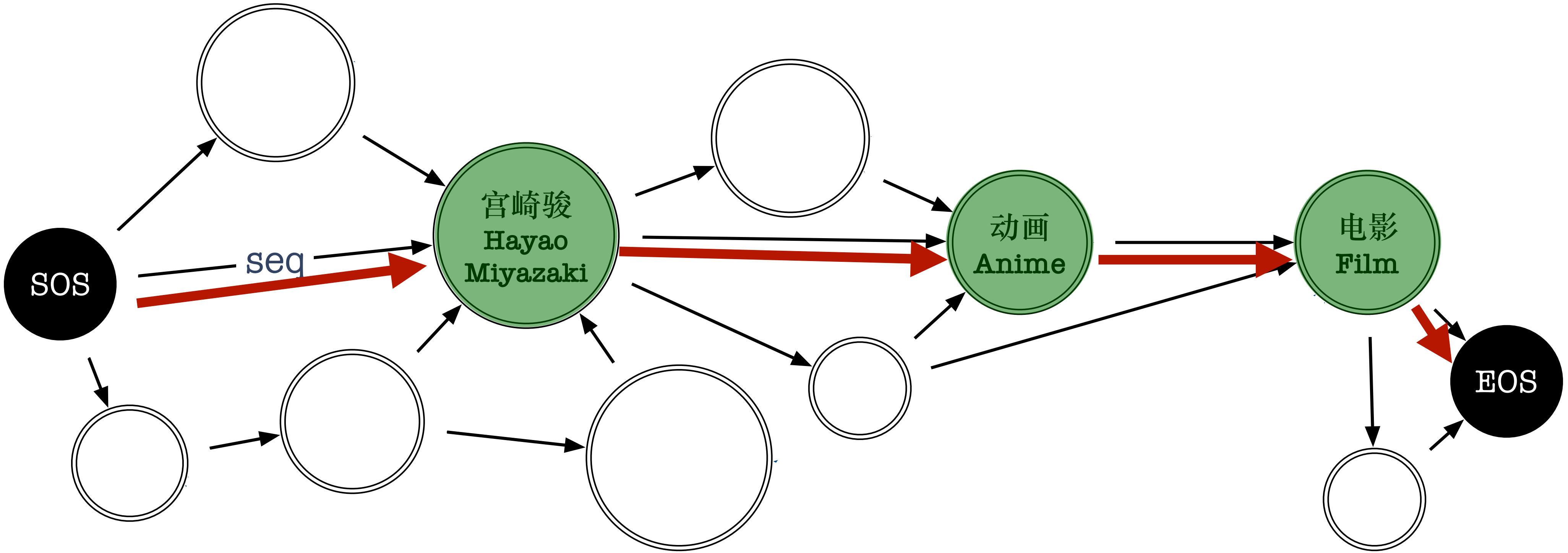
Heterogeneous phrase mining



Patterns: graph structure
Show up multiple times: node feature
NER/Part-of-Speech tags: node feature
Continuous chunk: : seq edge
Syntactic dependency: syntactic edge

Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

Heterogeneous phrase mining



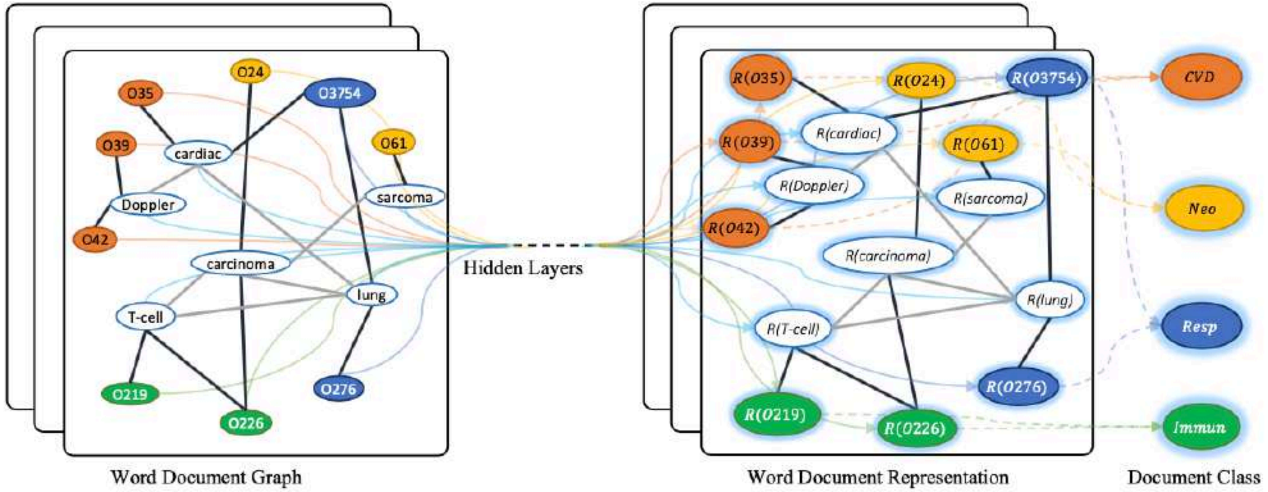
Classify Node: Relational GCN

Sort Node: Asymmetric Traveling Salesman Problem

Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

Transductive Text Classification with TextGCN

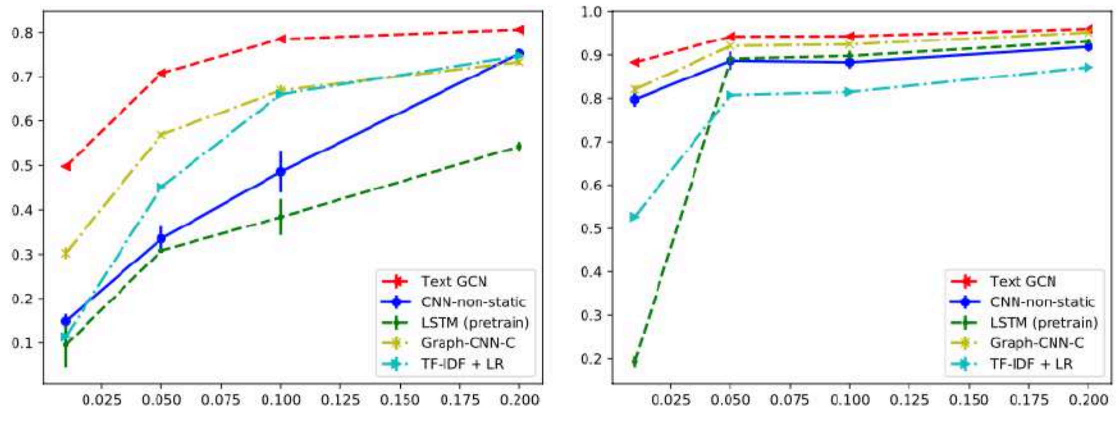
- **Nodes**: words and documents
- **Edges**: co-occurrence (word-word) and TFIDF (word-doc)
- Model the graph with a **Graph Convolutional Network** (GCN) (Kipf and Welling 2017)



$$
A_{ij} = \begin{cases} \mathrm{PMI}(i,j) & i, j \text{ are words, } \mathrm{PMI}(i,j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases}
$$

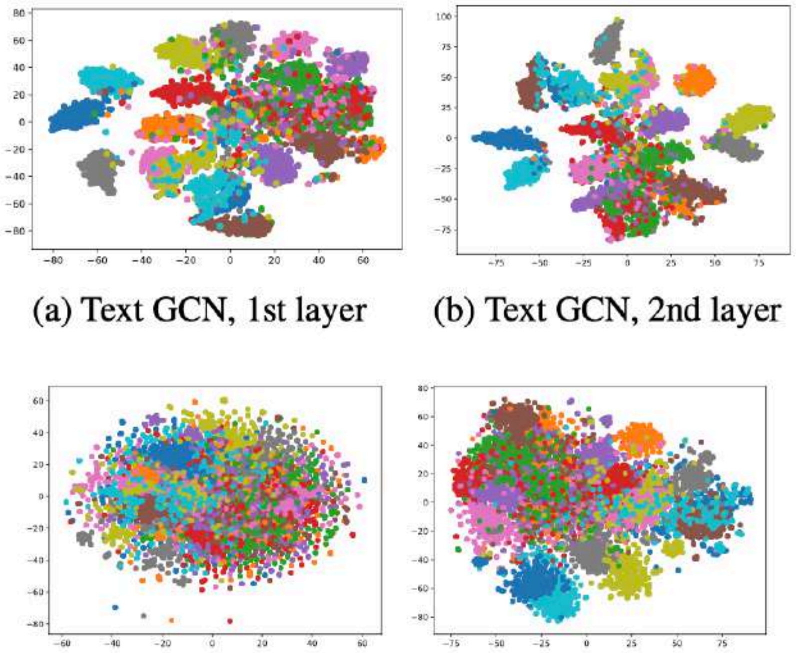
Yao et al., **Graph Convolutional Networks for Text Classification**, AAAI'19

Transductive Text Classification with TextGCN



(a) 20NG (b) R8

Test accuracy by varying training data promotions



(a) Text GCN, 1st layer (b) Text GCN, 2nd layer

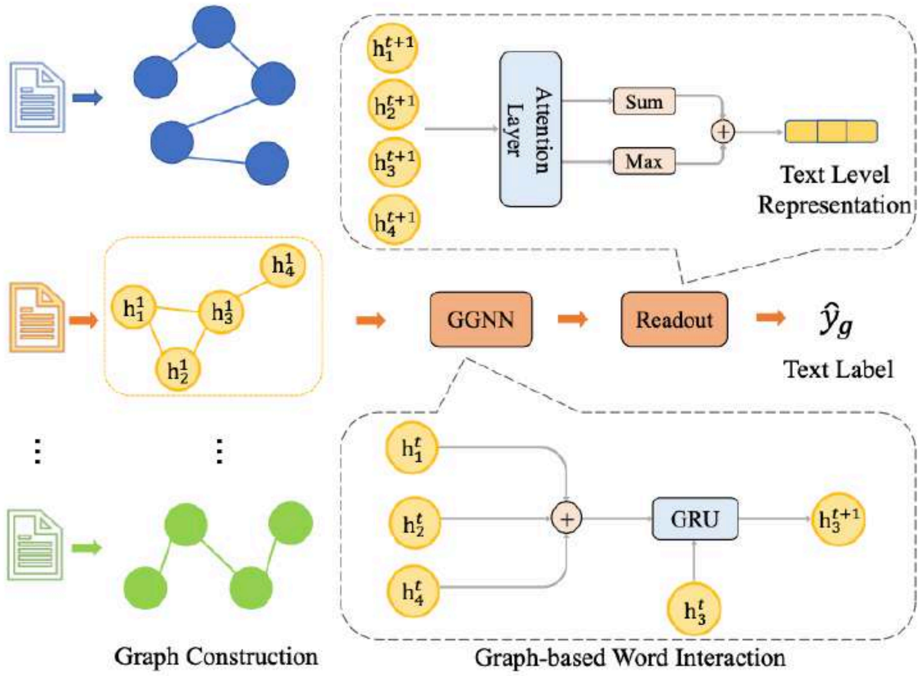
(c) PV-DBOW (d) PTE

The t-SNE visualization of test document embeddings in 20NG

Yao et al., **Graph Convolutional Networks for Text Classification**, AAAI'19

Inductive Text Classification with TextING

- **Nodes**: words in a document
- **Edges**: co-occurrence (word-word)
- Model the graph with a **Gated Graph Neural Networks** (Li et al., 2015)
- Each document is an individual graph and text level word interactions can be learned in it.
- It can generalise to new words that absent in training, therefore applicable for inductive circumstances.



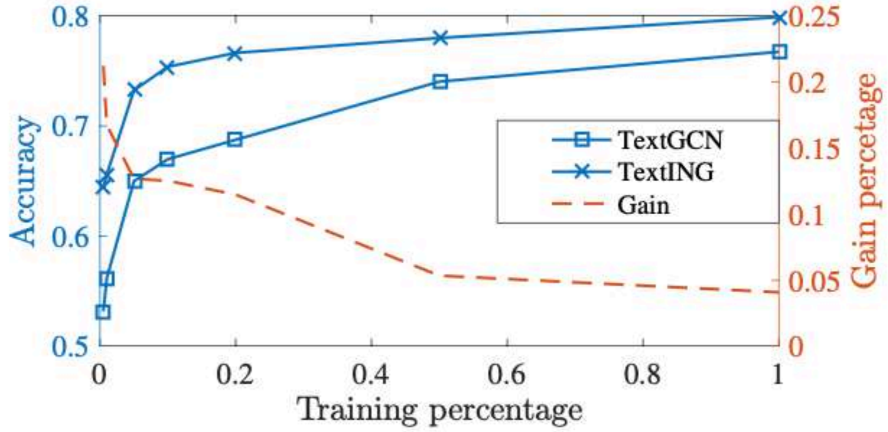
The architecture of TextING

Zhang et al., **Every Document Owns Its Structure: Inductive Text Classification via Graph Neural Networks**, ACL'20

Inductive Text Classification with TextING

Model	MR*	Ohsumed*
TextGCN	53.15	47.24
TextING	64.43	57.11
# Words in Training	465	7,009
# New Words in Test	18,299	7,148

Accuracy (%) of TextGCN and TextING on MR and Ohsumed



Test performance and gain with different percent of training data on MR.

Zhang et al., **Every Document Owns Its Structure: Inductive Text Classification via Graph Neural Networks**, ACL'20

Applied to feeds news recommendation

Title

See these cars with less than 2L/100km fuel consumption and up to 1000km recharge mileage

Tagged Concept

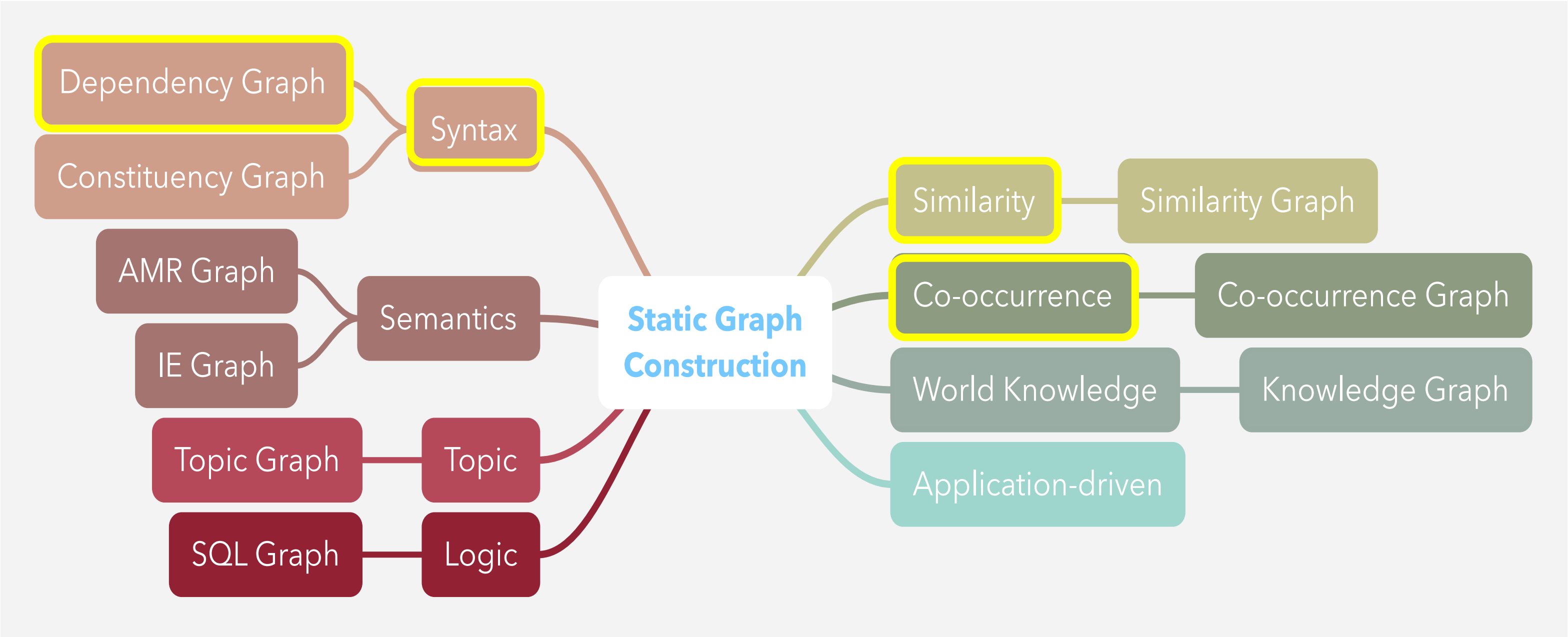
Low fuel consumption cars

Liu et al., **GIANT: Scalable Creation of a Web-scale Ontology**, SIGMOD'20

Summary

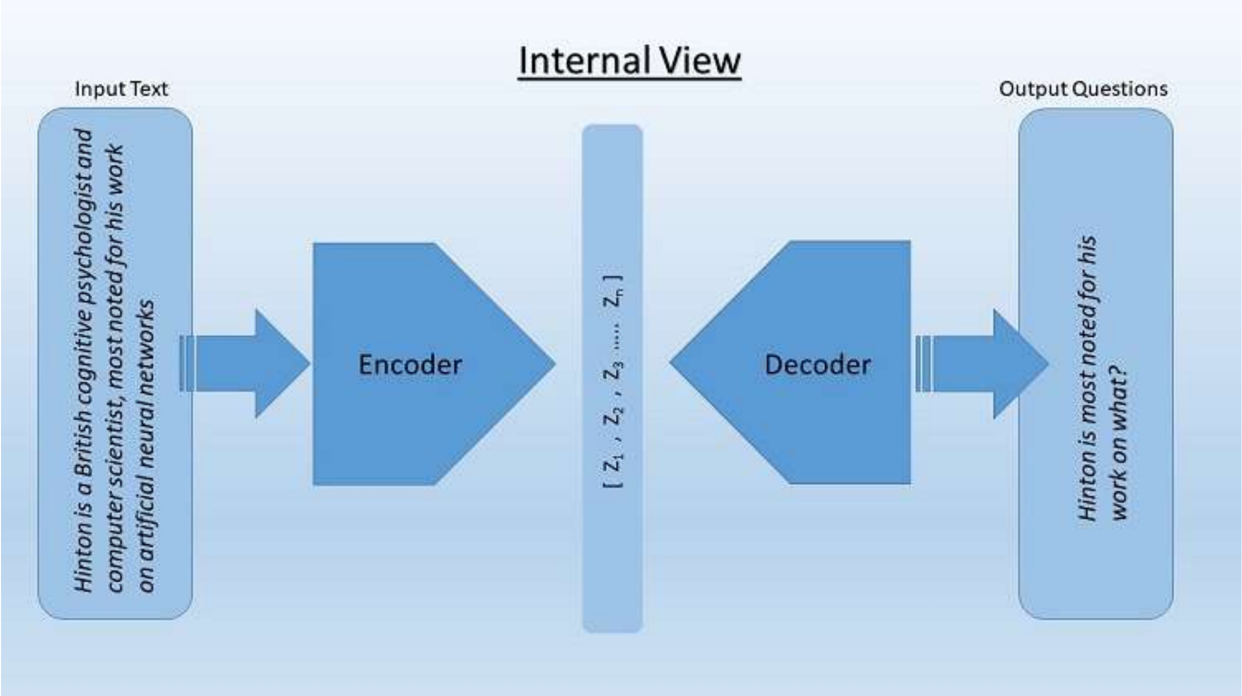
GNN enables:

- Encode multi-scale information
- Encode heterogenous information



Natural Language Generation
Machine Translation

Natural Question Generation



- Input
  - A text passage $X^p = \{x_1^p, x_2^p, ..., x_N^p\}$
  - A target answer $X^a = \{x_1^a, x_2^a, ..., x_L^a\}$
- Output
  - A natural language question

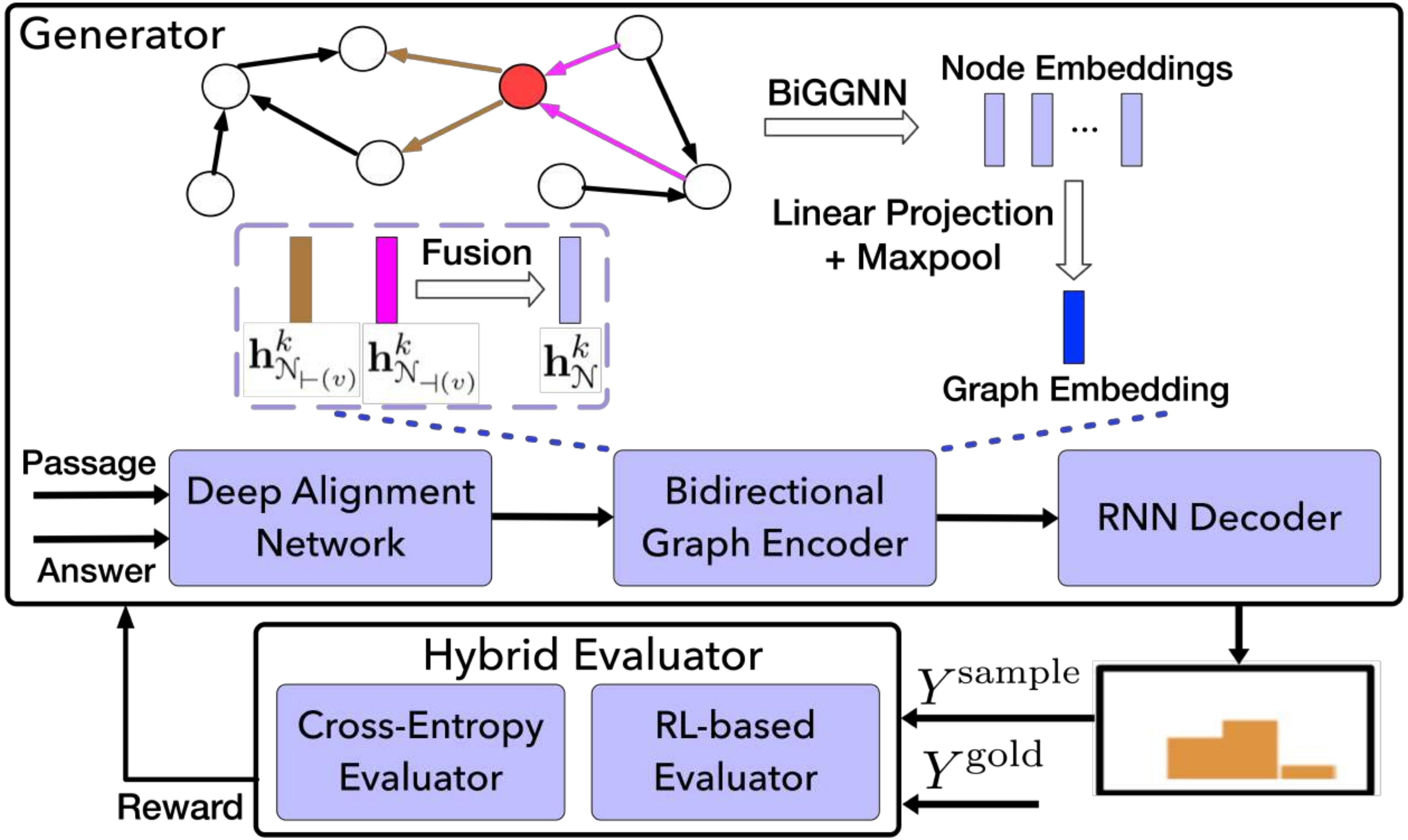
$$\hat{Y} = \{y_1, y_2, ..., y_T\}$$

which maximizes the conditional likelihood

$$\hat{Y} = \arg\max_Y P(Y|X^p, X^a)$$

Ref: https://images.app.goo.gl/TJEqwFS8nBNW8n8SA

RL-based Graph2Seq for QG [Chen et al. ICLR'20]

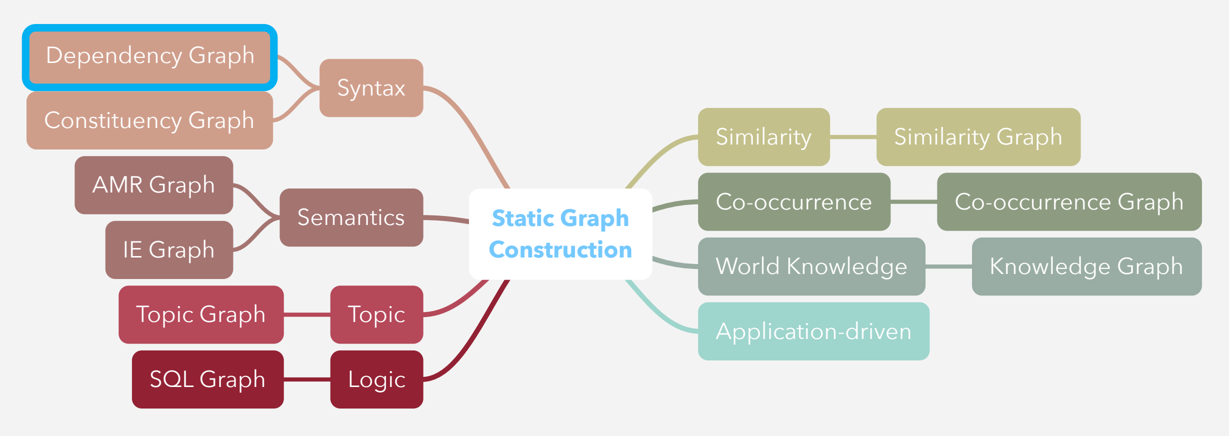


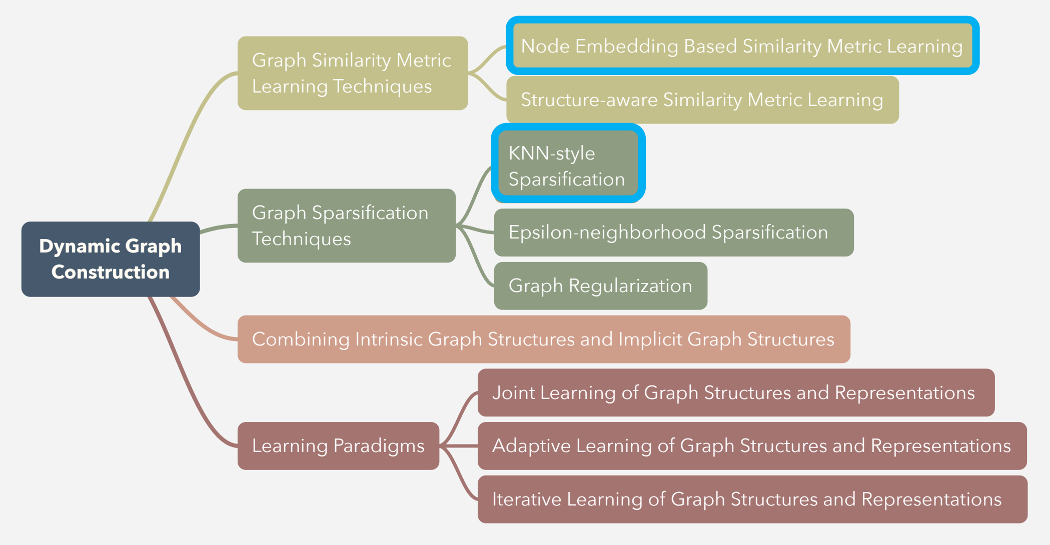
Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.

RL-based Graph2Seq for QG [Chen et al. ICLR'20]

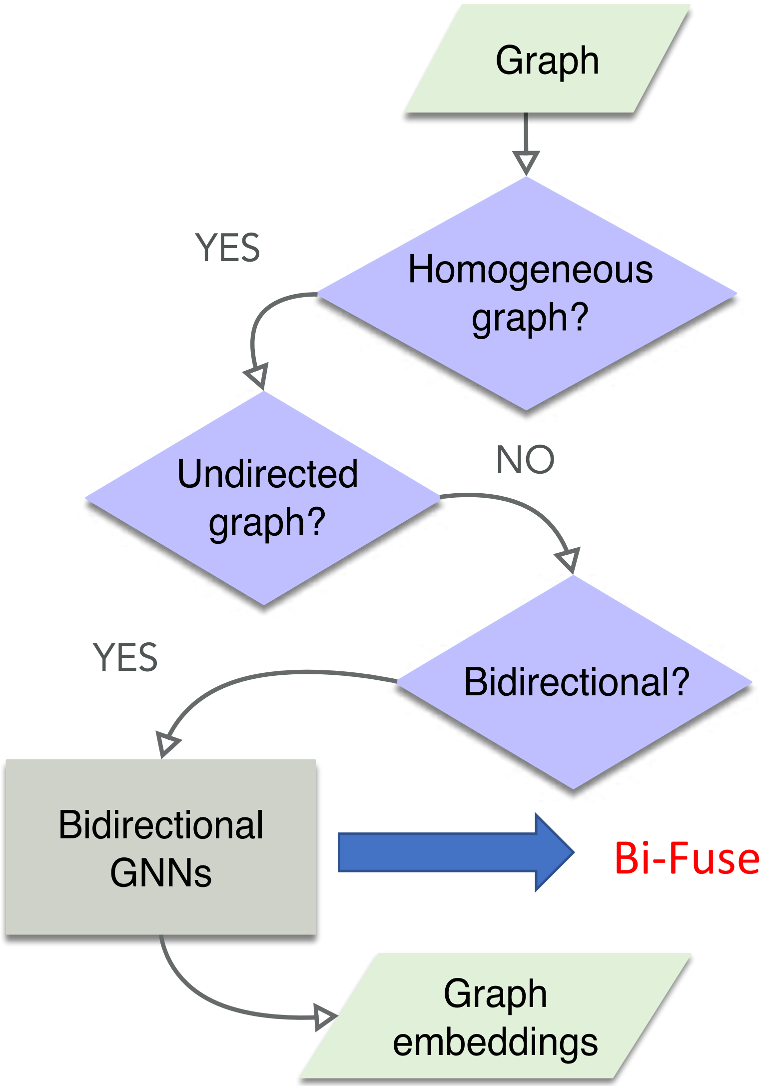
Two graph construction strategies:

1) Syntax-based static passage graph construction
2) Semantics-aware dynamic passage graph construction





RL-based Graph2Seq for QG [Chen et al. ICLR'20]



Bi-Fuse GGNN as the graph encoder

RL-based Graph2Seq for QG [Chen et al. ICLR'20]

Methods	BLEU-4	Methods	BLEU-4
$G2S_{dyn}$+BERT+RL	18.06	$G2S_{dyn}$ w/o feat	16.51
$G2S_{sta}$+BERT+RL	18.30	$G2S_{sta}$ w/o feat	16.65
$G2S_{sta}$+BERT-fixed+RL	18.20	$G2S_{dyn}$ w/o DAN	12.58
$G2S_{dyn}$+BERT	17.56	$G2S_{sta}$ w/o DAN	12.62
$G2S_{sta}$+BERT	18.02	$G2S_{sta}$ w/ DAN-word only	15.92
$G2S_{sta}$+BERT-fixed	17.86	$G2S_{sta}$ w/ DAN-contextual only	16.07
$G2S_{dyn}$+RL	17.18	$G2S_{sta}$ w/ GGNN-forward	16.53
$G2S_{sta}$+RL	17.49	$G2S_{sta}$ w/ GGNN-backward	16.75
$G2S_{dyn}$	16.81	$G2S_{sta}$ w/o BiGGNN, w/ Seq2Seq	16.14
$G2S_{sta}$	16.96	$G2S_{sta}$ w/o BiGGNN, w/ GCN	14.47

Ablation study on the SQuAD split-2 test set.

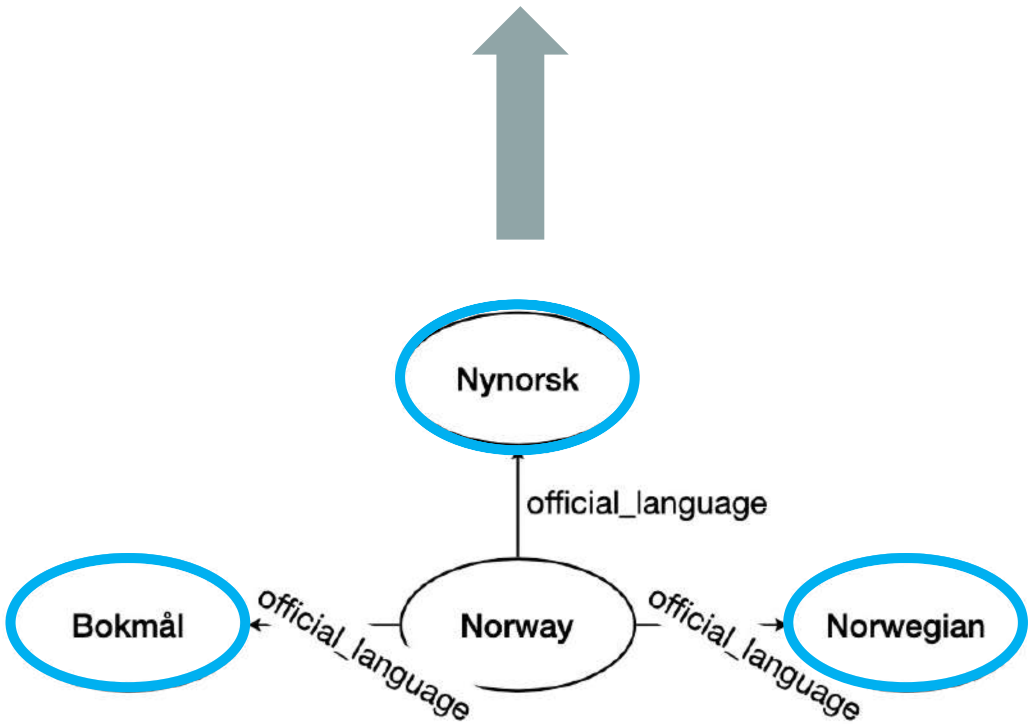
Bidirectional GNN performs better

Graph2Seq performs better than Seq2Seq

Static graph construction performs slightly better

Natural Question Generation From KG

Q: What languages are spoken in Norway?



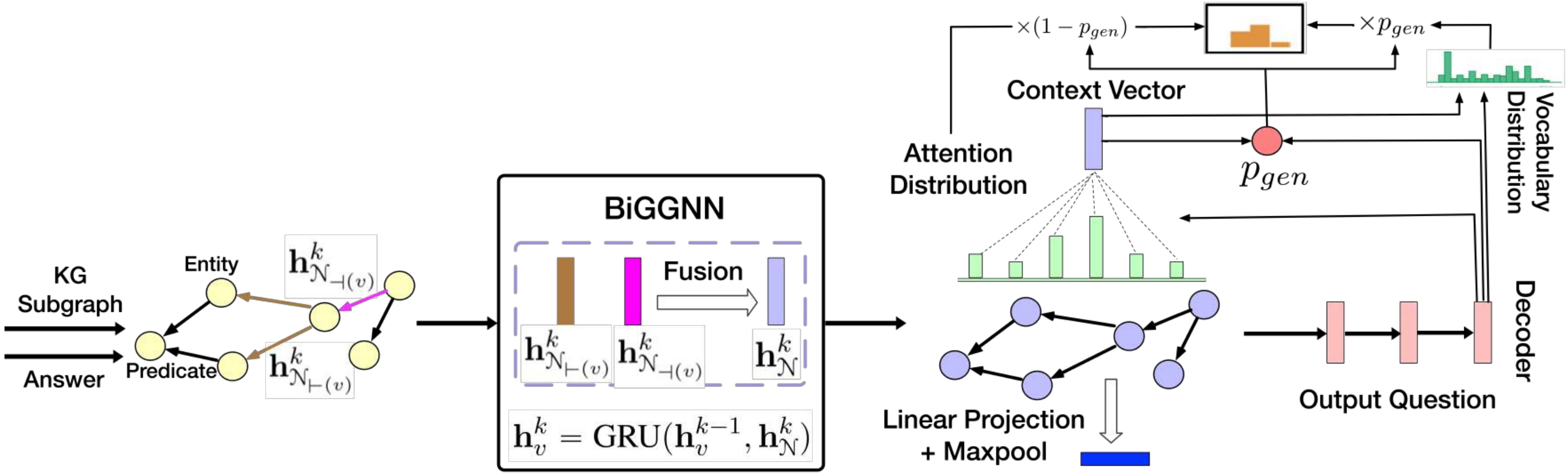
- Input
  - A KG subgraph $\mathcal{G}$ (i.e., a collection of subject-predicate-object triples)
  - A target answer set V^a
- Output
  - A natural language question

$$\hat{Y} = \{y_1, y_2, ..., y_T\}$$

which maximizes the conditional likelihood

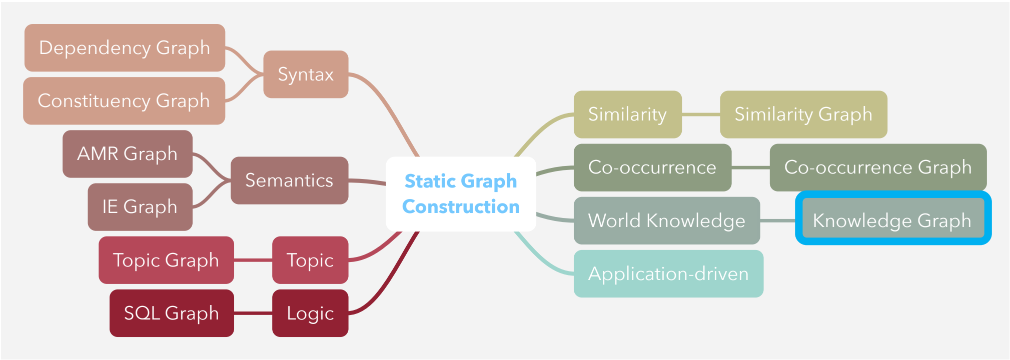
$$\hat{Y} = argmax_Y P(Y|\mathcal{G}, V^a)$$

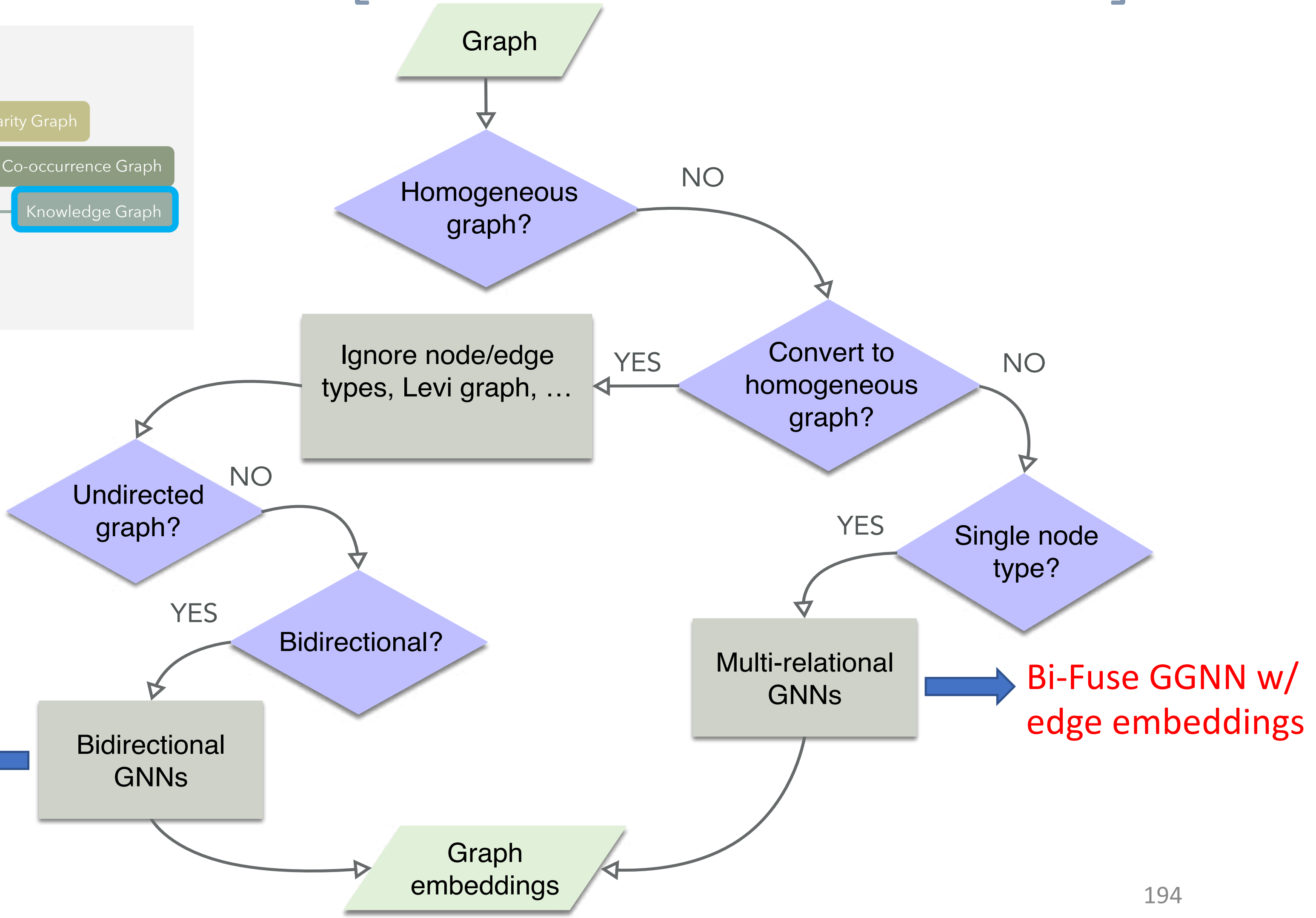
Graph2Seq for QG from KG [Chen et al. arXiv'20]



Chen et al. "Toward Subgraph Guided Knowledge Graph Question Generation with Graph Neural Networks". 2020.

Graph2Seq for QG from KG [Chen et al. arXiv'20]





Graph2Seq for QG from KG [Chen et al. arXiv'20]

Method	WQ			PQ		
	BLEU-4	METEOR	ROUGE-L	BLEU-4	METEOR	ROUGE-L
L2A	6.01	25.24	26.95	17.00	19.72	50.38
Transformer	8.94	13.79	32.63	56.43	43.45	73.64
MHQG+AE	11.57	29.69	35.53	25.99	33.16	58.94
G2S+AE	**29.45**	30.96	**55.45**	**61.48**	44.57	**77.72**
$G2S_{edge}$ +AE	29.40	**31.12**	55.23	59.59	**44.70**	75.20

Automatic evaluation results on the WQ and PQ test sets.

Levi graph conversion + homogeneous GNN performs comparably with multi-relational GNN

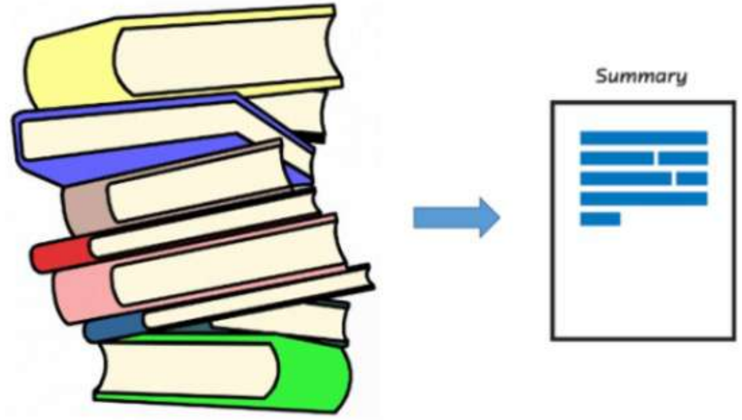
Graph2Seq for QG from KG [Chen et al. arXiv'20]

Bidirectional GNN performs better

Method	BLEU-4	METEOR	ROUGE-L
Bidirectional	61.48	44.57	77.72
Forward	59.59	42.72	75.82
Backward	59.12	42.66	75.03

Ablation study on directionality on the PQ test set.

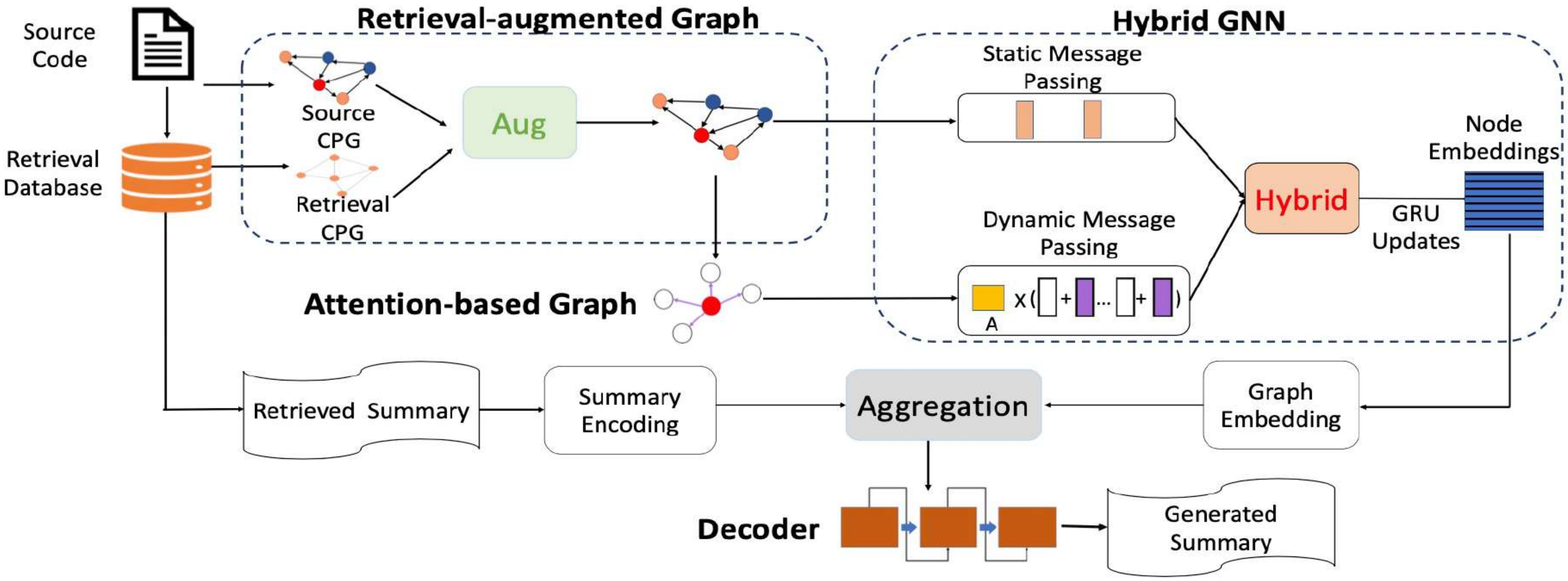
Summarization



- Input
  - A document, dialogue, code or multiple ones
- Output
  - A succinct sentence or paragraph

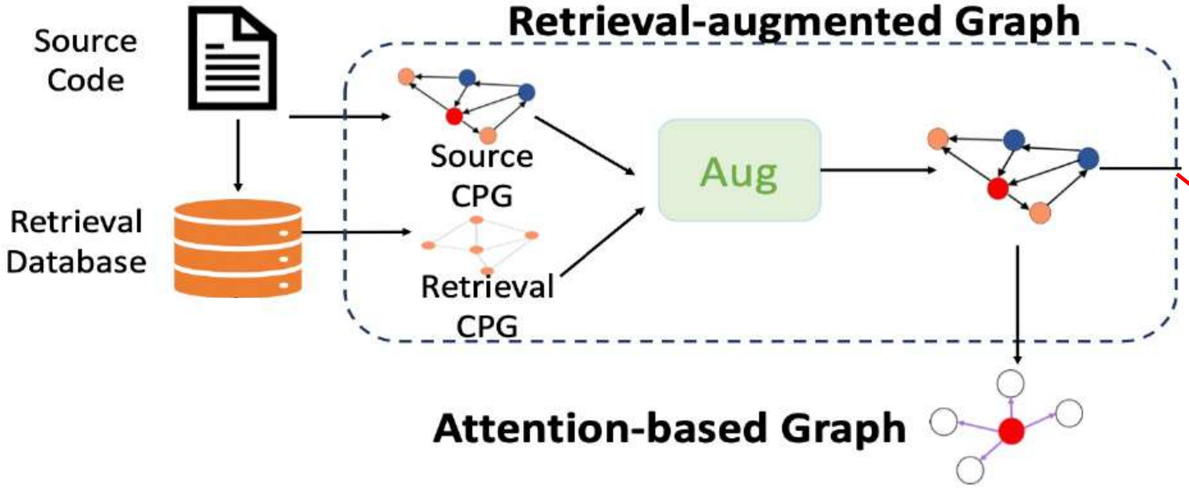
Ref: https://www.queppelin.com/how-nlp-is-helping-in-automatic-text-summarization-2, https://www.fiverr.com/yosree22/summarize-any-long-text-or-article-for-you

GNN for Code Summarization [Liu et al. ICLR'21]

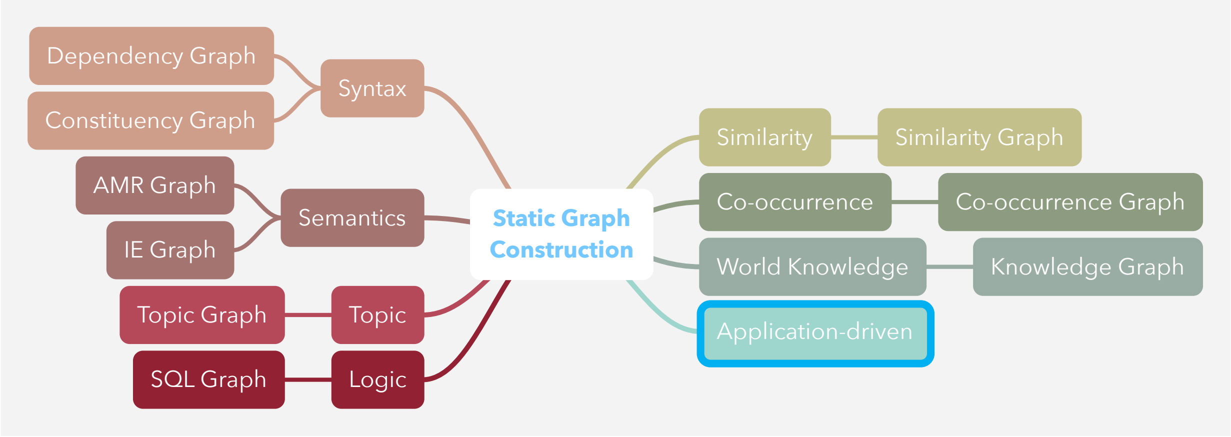


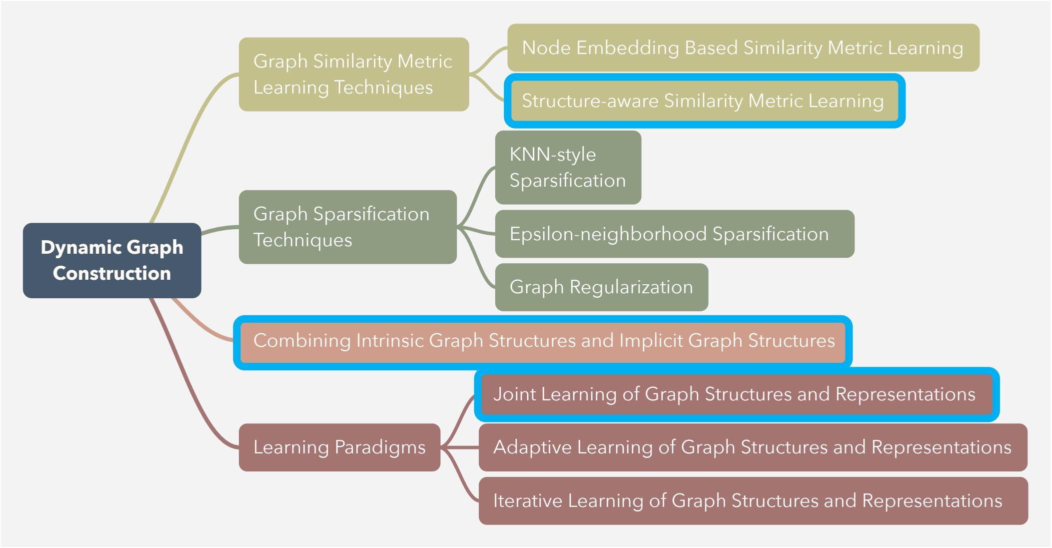
Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

GNN for Code Summarization [Liu et al. ICLR'21]

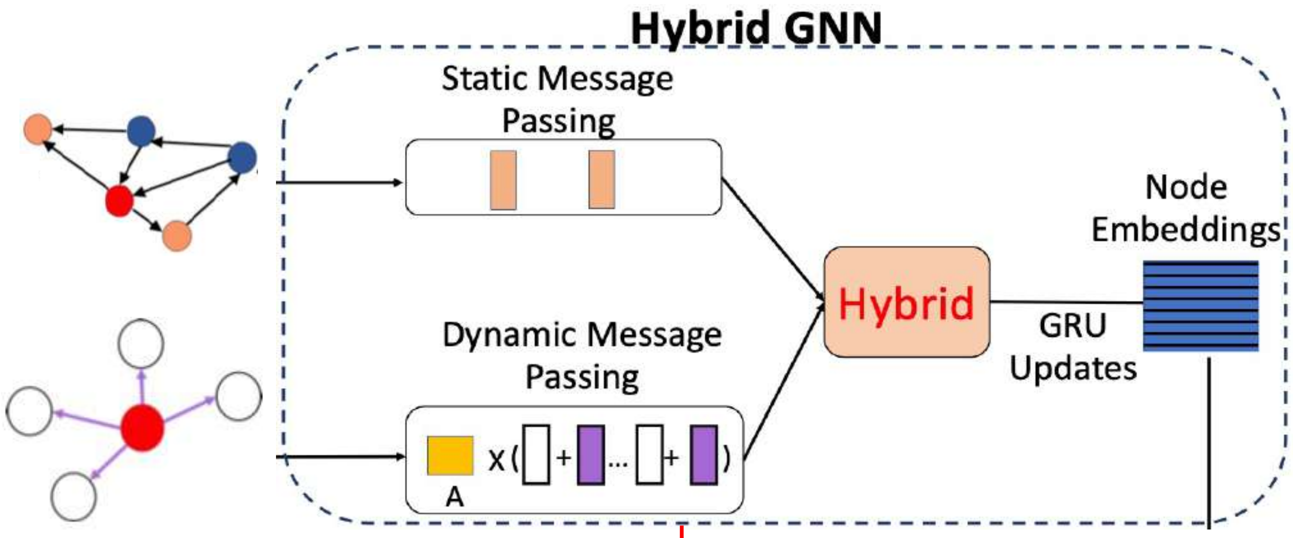


Static CPG graph + attention-based dynamic dynamic

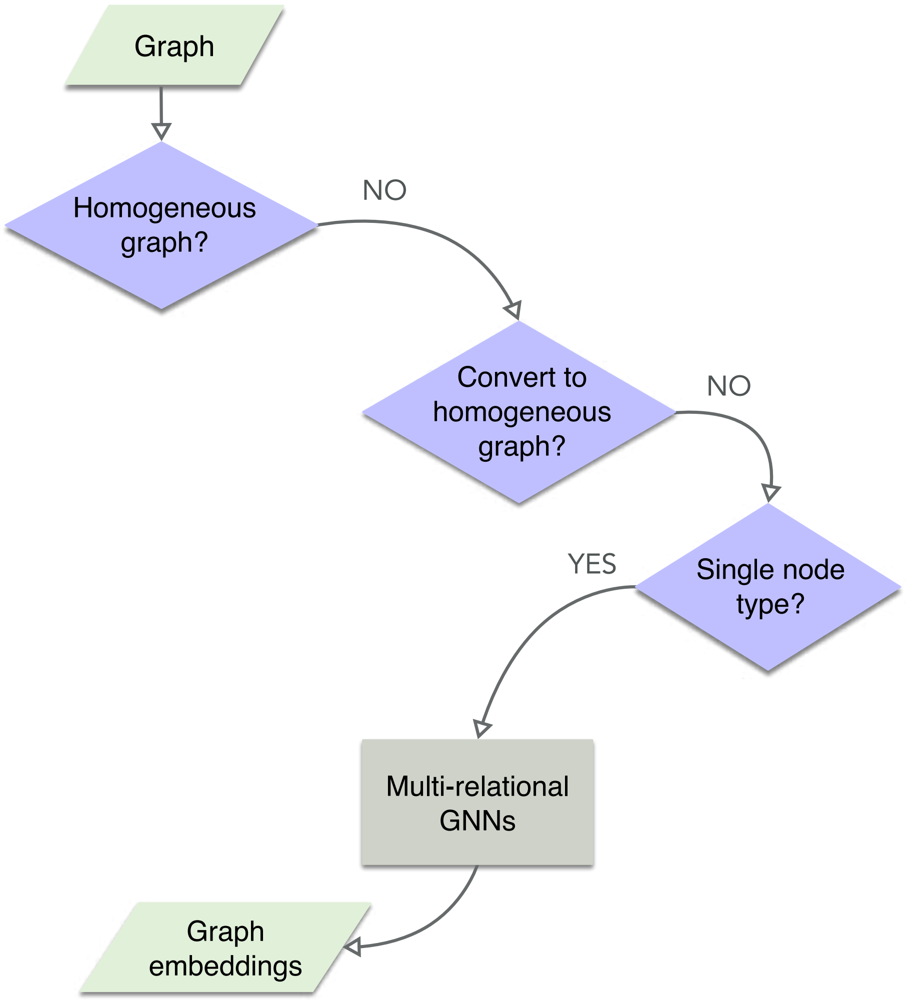




GNN for Code Summarization [Liu et al. ICLR’21]



Hybrid GNN running message passing on static & dynamic graphs



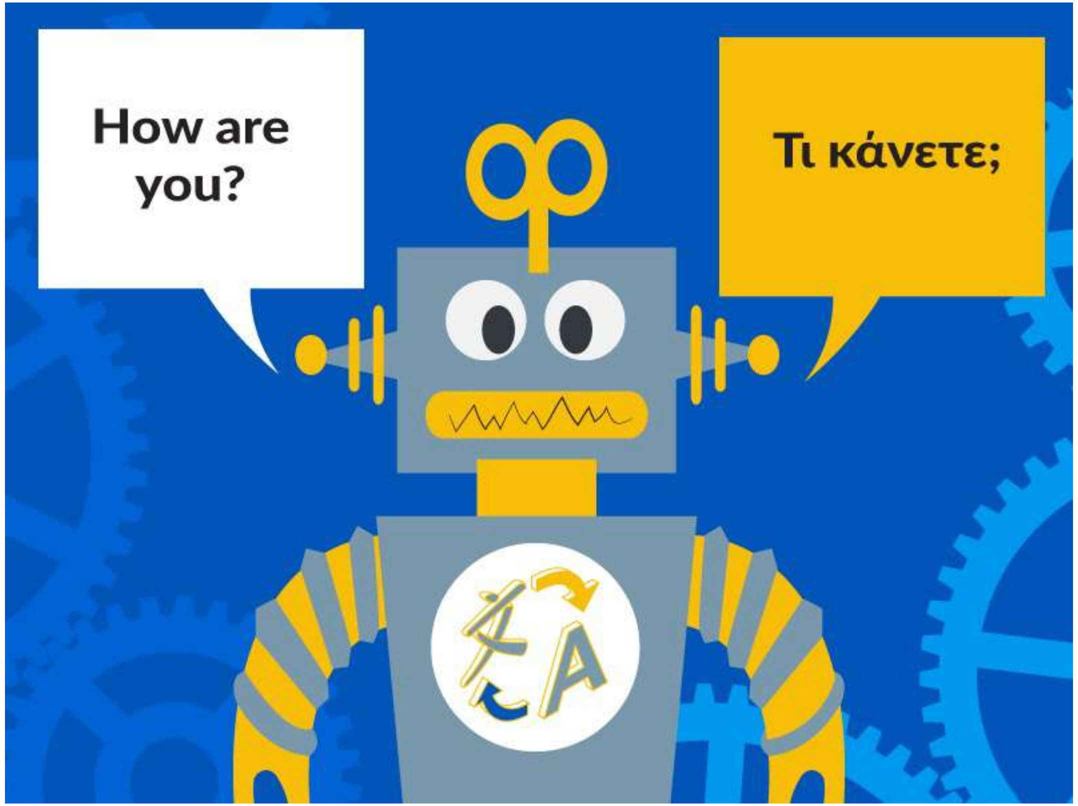
GNN for Code Summarization [Liu et al. ICLR'21]

Methods	In-domain			Out-of-domain			Overall		
	BLEU-4	ROUGE-L	METEOR	BLEU-4	ROUGE-L	METEOR	BLEU-4	ROUGE-L	METEOR
TF-IDF	15.20	27.98	13.74	5.50	15.37	6.84	12.19	23.49	11.43
NNGen	15.97	28.14	13.82	5.74	16.33	7.18	12.76	23.93	11.58
CODE-NN	10.08	26.17	11.33	3.86	15.25	6.19	8.24	22.28	9.61
Hybrid-DRL	9.29	30.00	12.47	6.30	24.19	10.30	8.42	28.64	11.73
Transformer	12.91	28.04	13.83	5.75	18.62	9.89	10.69	24.65	12.02
Dual Model	11.49	29.20	13.24	5.25	21.31	9.14	9.61	26.40	11.87
Rencos	14.80	31.41	14.64	7.54	23.12	10.35	12.59	28.45	13.21
GCN2Seq	9.79	26.59	11.65	4.06	18.96	7.76	7.91	23.67	10.23
GAT2Seq	10.52	26.17	11.88	3.80	16.94	6.73	8.29	22.63	10.00
SeqGNN	10.51	29.84	13.14	4.94	20.80	9.50	8.87	26.34	11.93
HGNN w/o augment & static	11.75	29.59	13.86	5.57	22.14	9.41	9.98	26.94	12.05
HGNN w/o augment & dynamic	11.85	29.51	13.54	5.45	21.89	9.59	9.93	26.80	12.21
HGNN w/o augment	12.33	29.99	13.78	5.45	22.07	9.46	10.26	27.17	12.32
HGNN w/o static	15.93	33.67	15.67	7.72	24.69	10.63	13.44	30.47	13.98
HGNN w/o dynamic	15.77	33.84	15.67	7.64	24.72	10.73	13.31	30.59	14.01
HGNN	**16.72**	**34.29**	**16.25**	**7.85**	**24.74**	**11.05**	**14.01**	**30.89**	**14.50**

Automatic evaluation results (in %) on the CCSD test set.

Combining static + dynamic graphs performs better

Machine Translation



- Input
  - Source language text $X = \{x_1, x_2, \ldots, x_N\}$
- Output
  - Target language text

$$\hat{Y} = \{y_1, y_2, \ldots, y_T\}$$

which maximizes the conditional likelihood

$$\hat{Y} = argmax_Y P(Y|X)$$

Ref: https://ciklopea.com/blog/translation/science-or-fiction-machine-translation-explained/

Syntactic GCN for MT [Bastings et al. EMNLP'17]

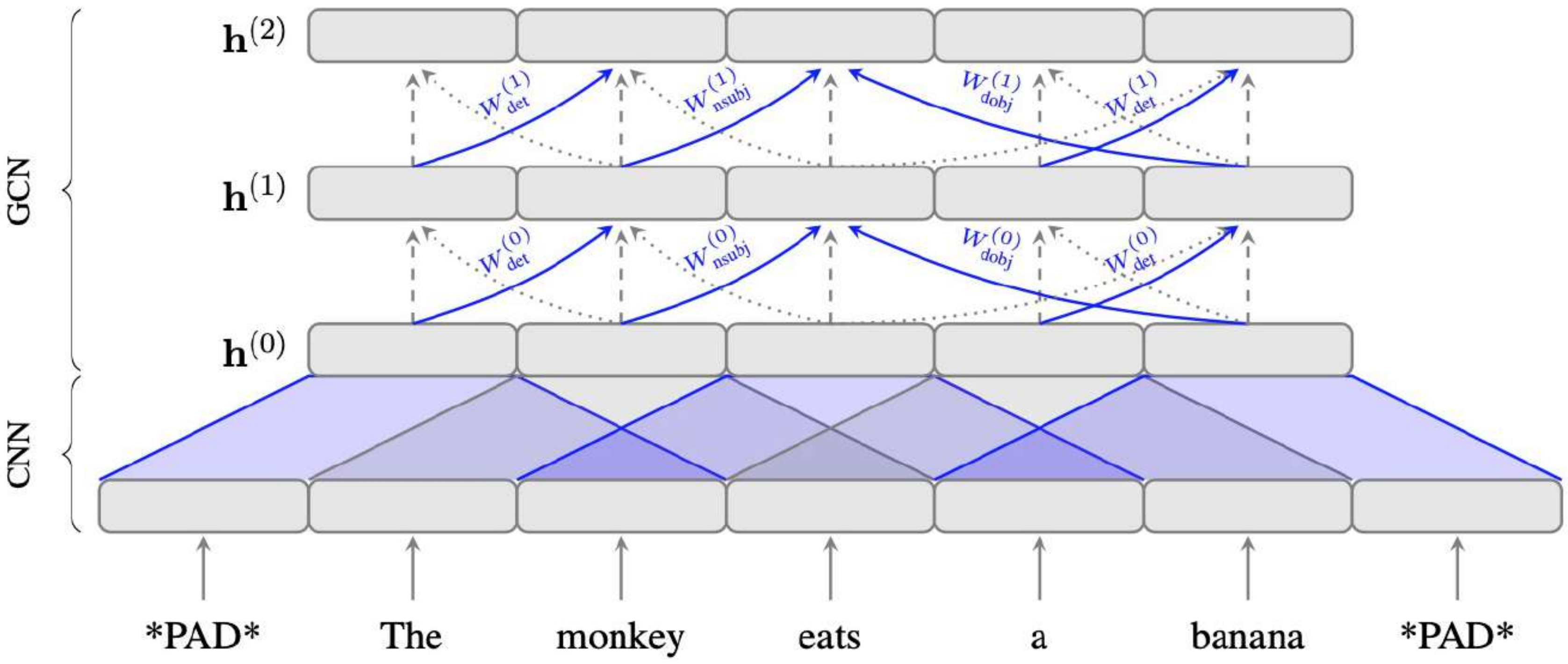
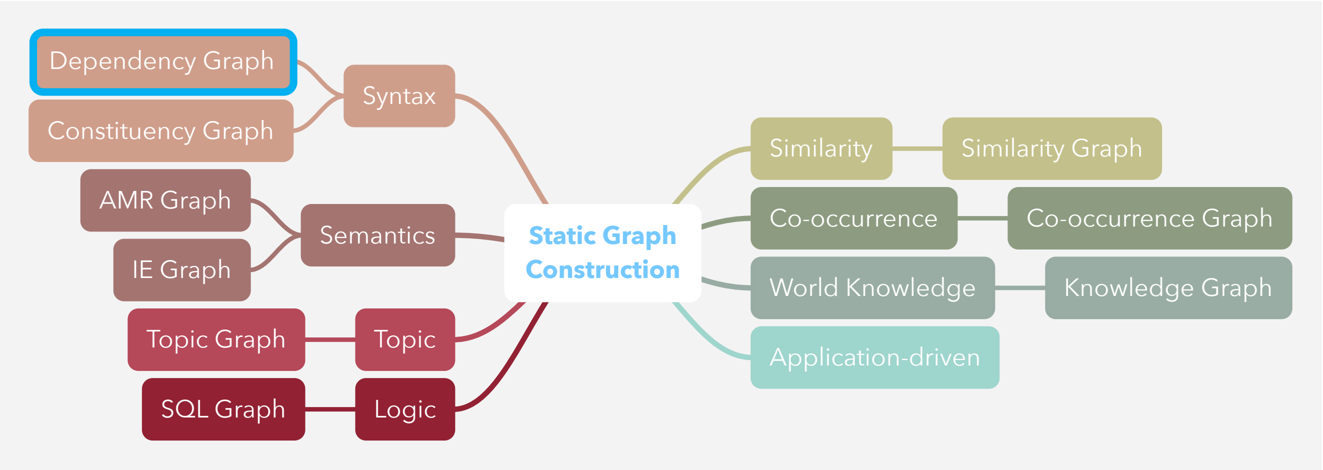
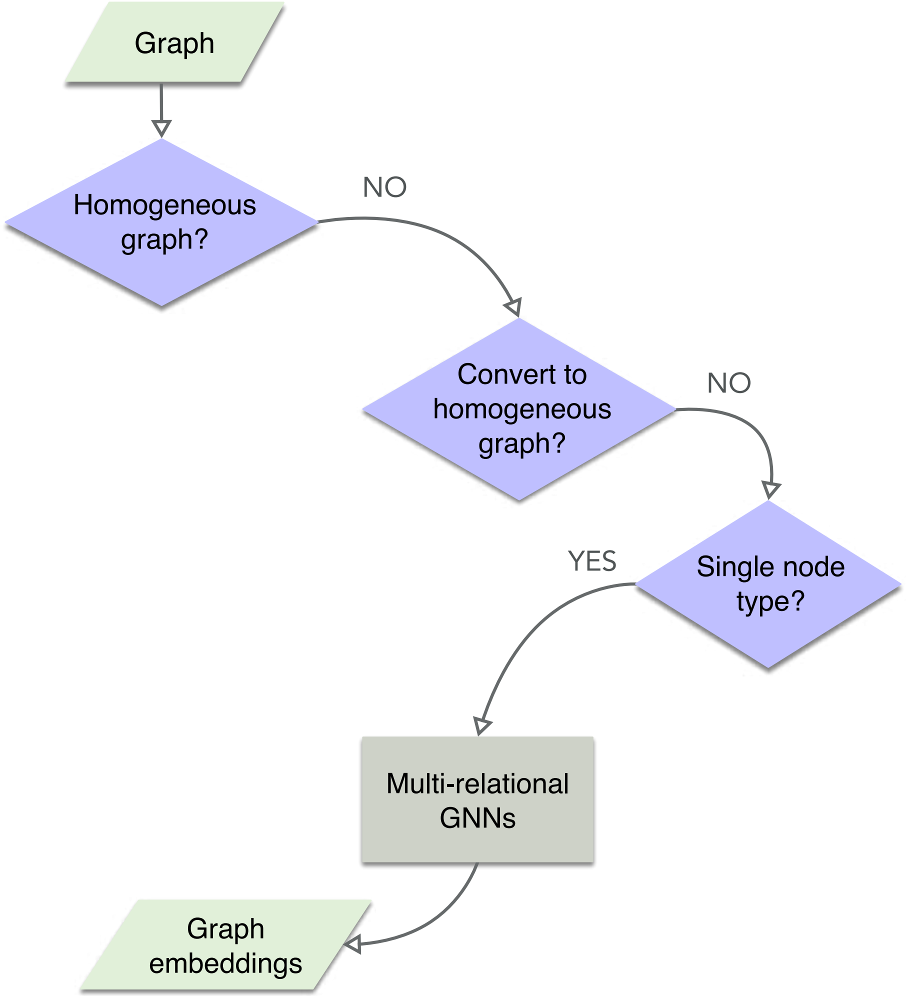


Figure 2: A 2-layer syntactic GCN on top of a convolutional encoder. Loop connections are depicted with dashed edges, syntactic ones with solid (dependents to heads) and dotted (heads to dependents) edges. Gates and some labels are omitted for clarity.

Bastings et al. "Graph Convolutional Encoders for Syntax-aware Neural Machine Translation". EMNLP 2017.

Syntactic GCN for MT [Bastings et al. EMNLP'17]





Syntactic GCN for MT [Bastings et al. EMNLP'17]

	Kendall	$BLEU_1$	$BLEU_4$
BoW	0.3352	40.6	9.5
+ GCN	0.3520	44.9	12.2
CNN	0.3601	42.8	12.6
+ GCN	0.3777	44.7	13.7
BiRNN	0.3984	45.2	14.9
+ GCN	0.4089	47.5	16.1
BiRNN (full)	0.5440	53.0	23.3
+ GCN	0.5555	54.6	23.9

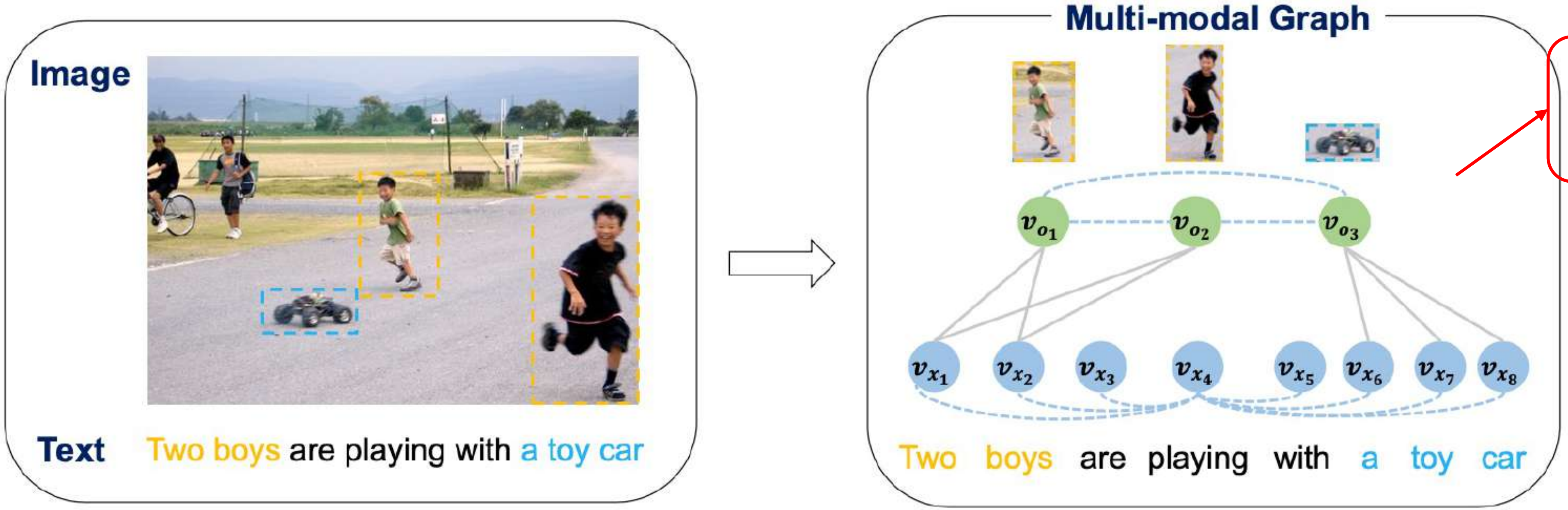
Test results for English-German.

	Kendall	$BLEU_1$	$BLEU_4$
BoW	0.2498	32.9	6.0
+ GCN	0.2561	35.4	7.5
CNN	0.2756	35.1	8.1
+ GCN	0.2850	36.1	8.7
BiRNN	0.2961	36.9	8.9
+ GCN	0.3046	38.8	9.6

Test results for English-Czech.

Syntactic GCN is helpful

Multi-modal Machine Translation [Yin et al. ACL'20]



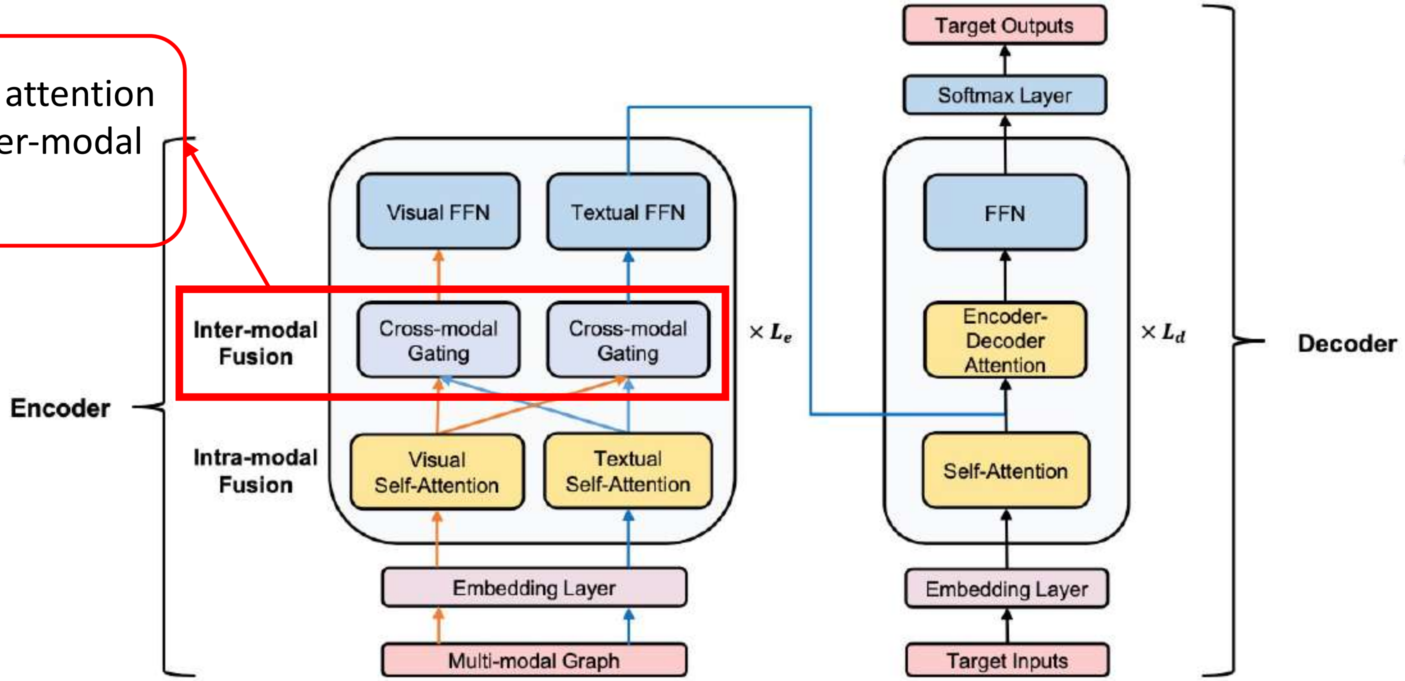
App-driven static graph construction

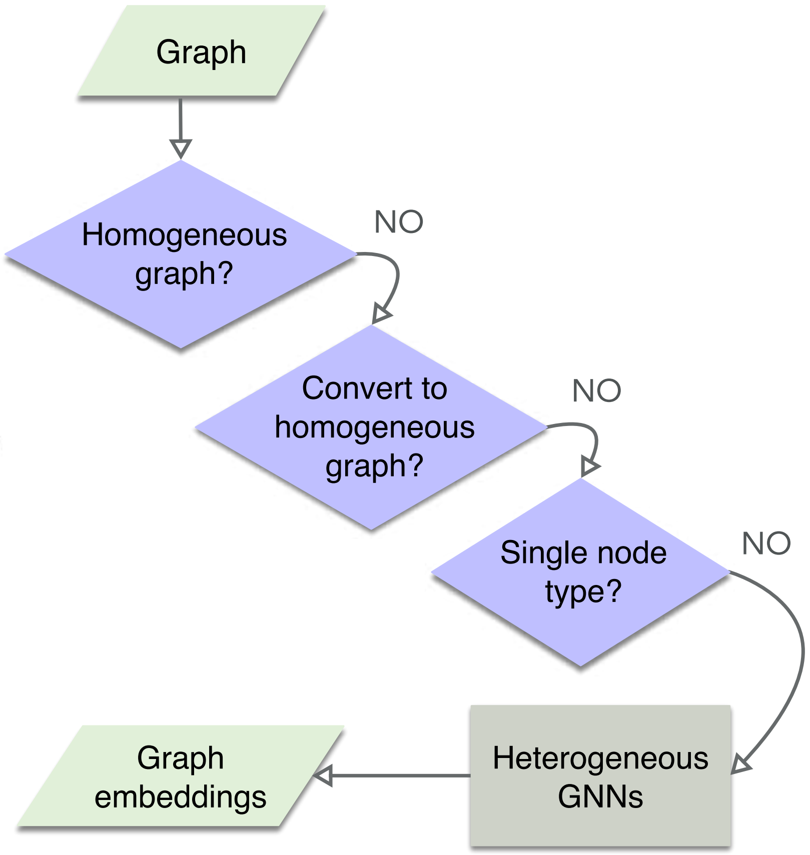
Figure . The multi-modal graph for an input sentence-image pair. The blue and green solid circles denote textual nodes and visual nodes respectively. An intra-modal edge (dotted line) connects two nodes in the same modality, and an inter-modal edge (solid line) links two nodes in different modalities. Note that we only display edges connecting the textual node "*playing*" and other textual ones for simplicity.

Yin et al. "A Novel Graph-based Multi-modal Fusion Encoder for Neural Machine Translation". ACL 2020.

Multi-modal Machine Translation [Yin et al. ACL'20]

Graph attention for inter-modal fusion





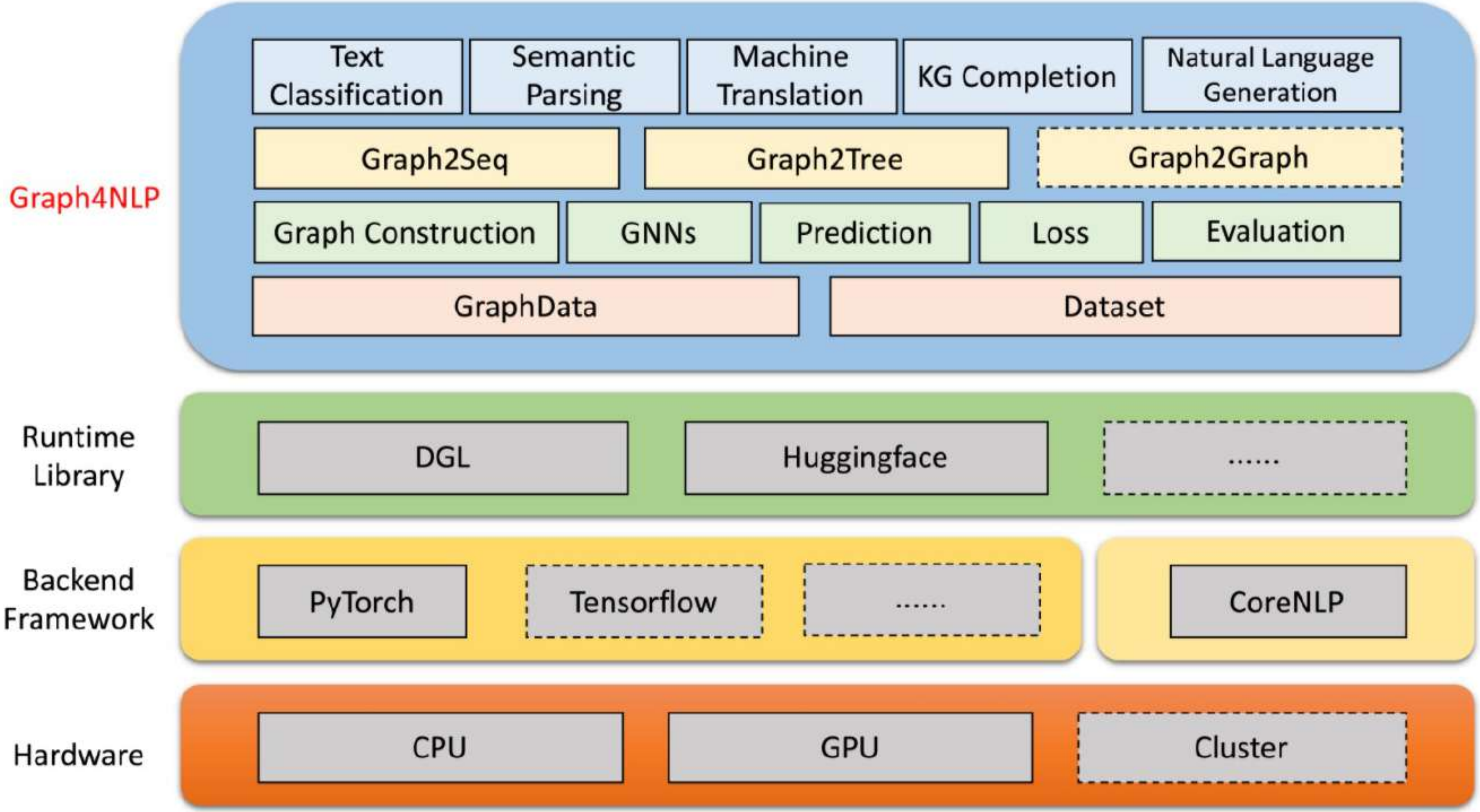
Multi-modal Machine Translation [Yin et al. ACL'20]

Model	En⇒Fr			
	Test2016		**Test2017**	
	BLEU	METEOR	BLEU	METEOR
Existing Multi-modal NMT Systems				
Fusion-conv(RNN) (Caglayan et al., 2017)	53.5	70.4	51.6	68.6
Trg-mul(RNN)(Caglayan et al., 2017)	54.7	71.3	52.7	**69.5**
Deliberation Network(TF) (Ive et al., 2019)	59.8	74.4	-	-
Our Multi-modal NMT Systems				
Transformer (Vaswani et al., 2017)	59.5	73.7	52.0	68.0
ObjectAsToken(TF) (Huang et al., 2016)	60.0	74.3	52.9	68.6
Enc-att(TF) (Delbrouck and Dupont, 2017b)	60.0	74.3	52.8	68.3
Doubly-att(TF) (Helcl et al., 2018)	59.9	74.1	52.4	68.1
Our model	**60.9**	**74.9**	**53.9**	69.3

Hands-on Demonstration

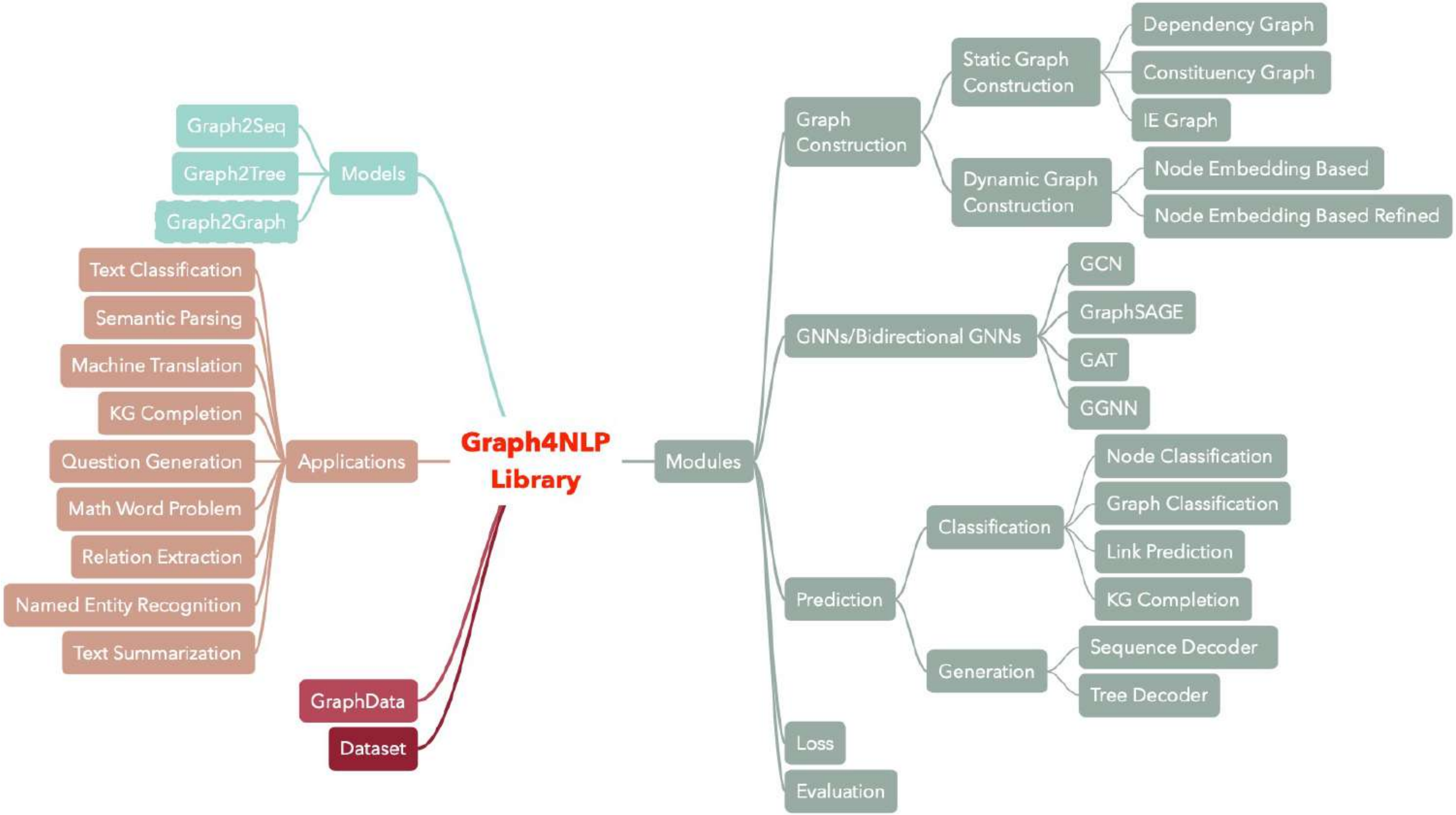
Graph4NLP: A Library for Deep Learning on Graphs for NLP

Overall Architecture of Graph4NLP Library

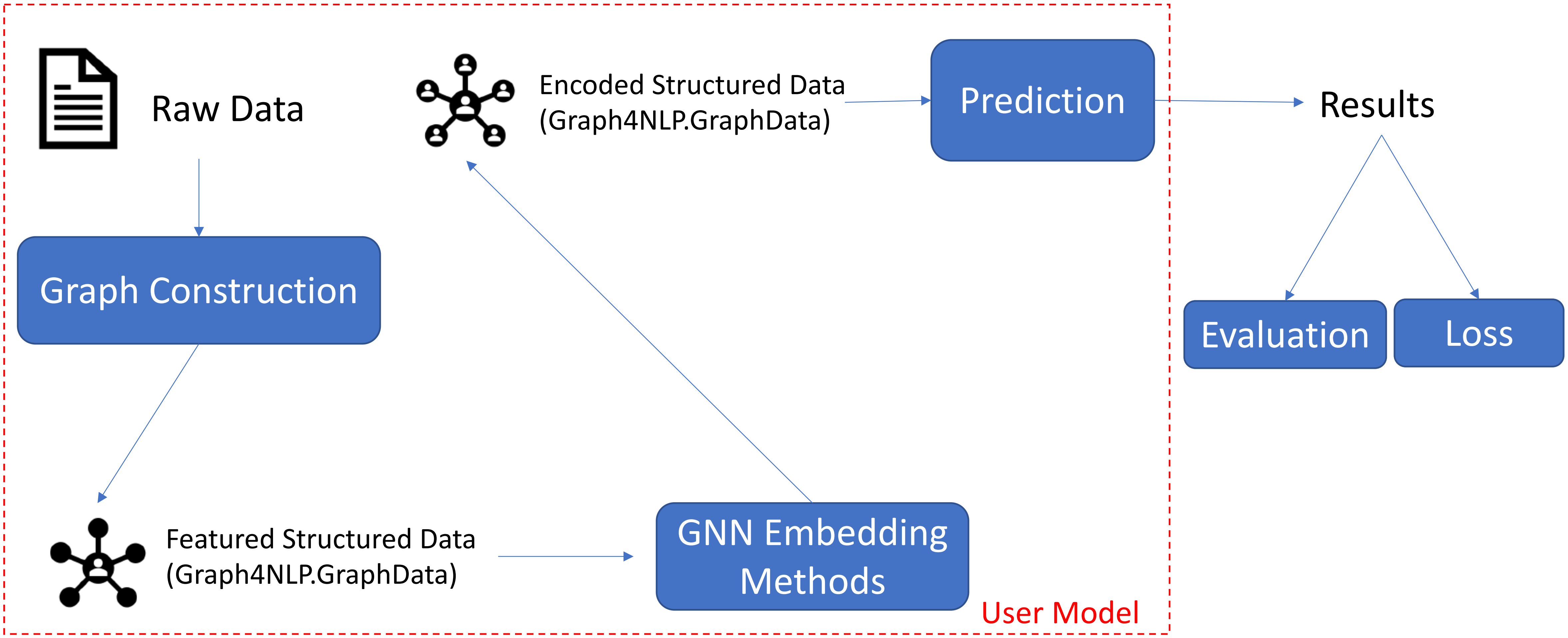


DGL: https://github.com/dmlc/dgl, DIG: https://github.com/divelab/DIG, Huggingface: https://github.com/huggingface/transformers

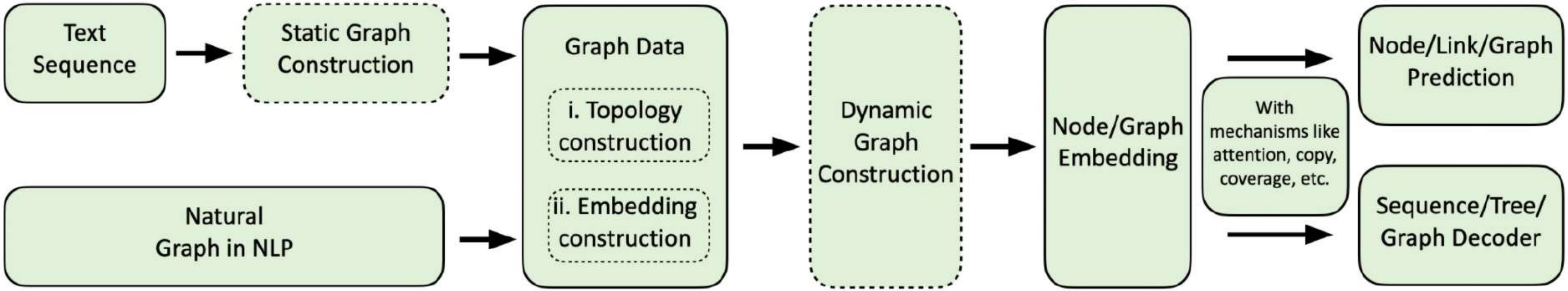
Dive Into Graph4NLP Library



Data Flow of Graph4NLP



Computing Flow of Graph4NLP



Performance of Built-in NLP Tasks

Task	Dataset	GNN Model	Graph construction	Evaluation	Performance
Text classification	TRECT CAirline CNSST	GAT	Dependency	Accuracy	0.948 0.769 0.538
Semantic Parsing	JOBS	SAGE	Constituency	Execution accuracy	0.936
Question generation	SQuAD	GGNN	Dependency	BLEU-4	0.15175
Machine translation	IWSLT14	GCN	Dynamic	BLEU-4	0.3212
Summarization	CNN(30k)	GCN	Dependency	ROUGE-1	26.4
Knowledge graph completion	Kinship	GCN	Dependency	MRR	82.4
Math word problem	MAWPS MATHQA	SAGE	Dynamic	Solution accuracy Exact match	76.4 61.07

Ref: https://mentorphile.com/2018/09/14/demo-or-die/

Demo 1: Building a Text Classification Application

1) git clone https://github.com/graph4ai/graph4nlp_demo
2) follow Get Started instructions in README

Demo 1: Building a Text Classification Application

```
def forward(self, graph_list, tgt=None, require_loss=True):
    # build graph topology
    batch_gd = self.graph_topology(graph_list)

    # run GNN encoder
    self.gnn(batch_gd)

    # run graph classifier
    self.clf(batch_gd)
    logits = batch_gd.graph_attributes['logits']

    if require_loss:
        loss = self.loss(logits, tgt)
        return logits, loss
    else:
        return logits
```

Model arch

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Demo 1: Building a Text Classification Application

Graph construction API, various built-in options, can be customized

```
self.graph_topology = DependencyBasedGraphConstruction(
                        embedding_style=embedding_style,
                        vocab=vocab.in_word_vocab,
                        hidden_size=config['num_hidden'],
                        word_dropout=config['word_dropout'],
                        rnn_dropout=config['rnn_dropout'],
                        fix_word_emb=not config['no_fix_word_emb'],
                        fix_bert_emb=not config.get('no_fix_bert_emb', False))
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Demo 1: Building a Text Classification Application

GNN API, various built-in options, can be customized

```
self.gnn = GraphSAGE(config['gnn_num_layers'],
            config['num_hidden'],
            config['num_hidden'],
            config['num_hidden'],
            config['graphsage_aggreagte_type'],
            direction_option=config['gnn_direction_option'],
            feat_drop=config['gnn_dropout'],
            bias=True,
            norm=None,
            activation=F.relu,
            use_edge_weight=use_edge_weight)
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Graph4NLP

Demo 1: Building a Text Classification Application

Prediction API, various built-in options, can be customized

```
self.clf = FeedForwardNN(2 * config['num_hidden'] \
                if config['gnn_direction_option'] == 'bi_sep' \
                else config['num_hidden'],
                config['num_classes'],
                [config['num_hidden']],
                graph_pool_type=config['graph_pooling'],
                dim=config['num_hidden'],
                use_linear_proj=config['max_pool_linear_proj'])
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Demo 1: Building a Text Classification Application

Dataset API, various built-in options, can be customized

```
dataset = TrecDataset(root_dir=self.config.get('root_dir', self.config['root_data_dir']),
                      pretrained_word_emb_name=self.config.get('pretrained_word_emb_name', "840B"),
                      merge_strategy=merge_strategy,
                      seed=self.config['seed'],
                      thread_number=4,
                      port=9000,
                      timeout=15000,
                      word_emb_size=300,
                      graph_type=graph_type,
                      topology_builder=topology_builder,
                      topology_subdir=topology_subdir,
                      dynamic_graph_type=self.config['graph_type'] if \
                          self.config['graph_type'] in ('node_emb', 'node_emb_refined') else None,
                      dynamic_init_topology_builder=dynamic_init_topology_builder,
                      dynamic_init_topology_aux_args={'dummy_param': 0})
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Demo 2: Building a Semantic Parsing Application

1) git clone https://github.com/graph4ai/graph4nlp_demo
2) follow Get Started instructions in README

Graph4NLP

Demo 2: Building a Semantic Parsing Application

Graph2Seq API

```
def _build_model(self):
    self.model = Graph2Seq.from_args(self.opt, self.vocab).to(self.device)
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

Demo 2: Building a Semantic Parsing Application

Dataset API, various built-in options, can be customized

```
dataset = JobsDataset(root_dir=self.opt["graph_construction_args"]["graph_construction_share"]["root_dir"],
                      pretrained_word_emb_name=self.opt["pretrained_word_emb_name"],
                      pretrained_word_emb_url=self.opt["pretrained_word_emb_url"],
                      pretrained_word_emb_cache_dir=self.opt["pretrained_word_emb_cache_dir"],
                      val_split_ratio=self.opt["val_split_ratio"],
                      merge_strategy=self.opt["graph_construction_args"]["graph_construction_private"][
                          "merge_strategy"],
                      edge_strategy=self.opt["graph_construction_args"]["graph_construction_private"][
                          "edge_strategy"],
                      seed=self.opt["seed"],
                      word_emb_size=self.opt["word_emb_size"],
                      share_vocab=self.opt["graph_construction_args"]["graph_construction_share"][
                          "share_vocab"],
                      graph_type=graph_type,
                      topology_builder=topology_builder,
                      topology_subdir=self.opt["graph_construction_args"]["graph_construction_share"][
                          "topology_subdir"],
                      thread_number=self.opt["graph_construction_args"]["graph_construction_share"][
                          "thread_number"],
                      dynamic_graph_type=self.opt["graph_construction_args"]["graph_construction_share"][
                          "graph_type"],
                      dynamic_init_topology_builder=dynamic_init_topology_builder,
                      dynamic_init_topology_aux_args=None)
```

https://github.com/graph4ai/graph4nlp_demo/tree/main/NAACL2021_demo

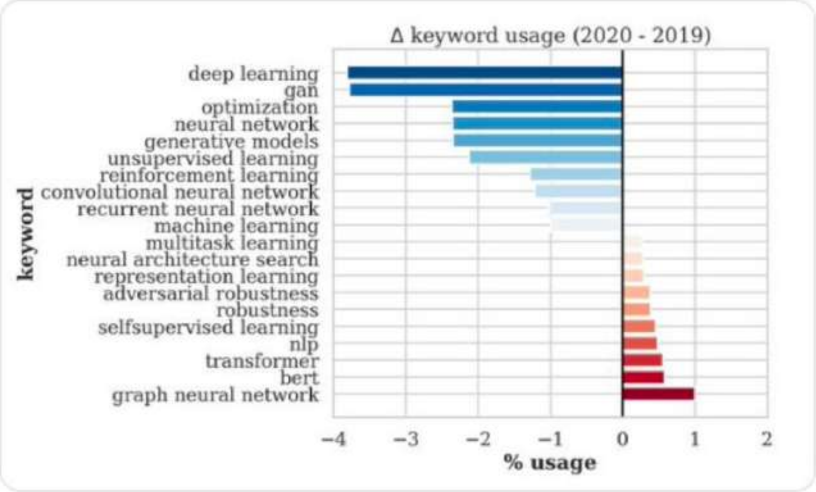
DLG4NLP: Future Directions and Conclusions

Future Directions

[Vashishth et al. EMNLP'19 Tutorial]

- The Rise of GNN + NLP

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



- Graph Construction for NLP
  - Dynamic graph construction are largely underexplored!
  - How to effectively combine advantages of static graph and dynamic graph?
  - How to construct heterogeneous dynamic graph?
  - How to make dynamic graph construction itself scalable?

Future Directions

- Scaling GNNs to Large Graphs
  - Most existing multi-relational or heterogeneous GNNs will have scalability issues when applied to large graphs in NLP such as KGs (> 1m)
- GNNs + Transformer in NLP
  - How to effectively combine the advantages of GNNs and Transformer?
  - Is graph transformer the best way to utilize?
- Pretraining GNNs for NLP
  - Information Retrieval/ Search

Future Directions

- Graph-to-graph Learning in NLP
  - How to effectively develop Graph-to-Graph models for solving graph transformation problem in NLP (i.e. information extraction)?
- Joint Text and KG Reasoning in NLP
  - Joint text and KG reasoning is less explored although GNNs for multi-hop reasoning gains popularity
- Incorporate Source and Context into Knowledge Graph Construction and Verification

Conclusions

- Deep Learning on Graphs for NLP is a fast-growing area today!
- Since graph can naturally encode complex information, it could bridge a gap by combining both empirical domain knowledges and the power of deep learning.
- For a NLP task,
  - how to convert text sequence into the best graph (directed, multi-relation, heterogeneous)
  - how to determine proper graph representation learning technique?
- Our Graph4NLP library aims to make easy use of GNNs for NLP:
  - Survey: https://arxiv.org/abs/2106.06090
  - Code: https://github.com/graph4ai/graph4nlp
  - Demo: https://github.com/graph4ai/graph4nlp_demo
  - Github literature list: https://github.com/graph4ai/graph4nlp_literature