GNN Applications in NLP

How are graphs used in NLP currently?

- DL-NLP representation dominated by embeddings
  - Word, sentence and document level embeddings
  - Relations are also represented by relational embeddings
- Graph information such as dependency/syntax often represented as auxiliary information
- Graphs are NOT first class citizen in NLP
  - Not involved in end to end training
- Can GNNs change some of these?

Graphs are extensively used in NLP

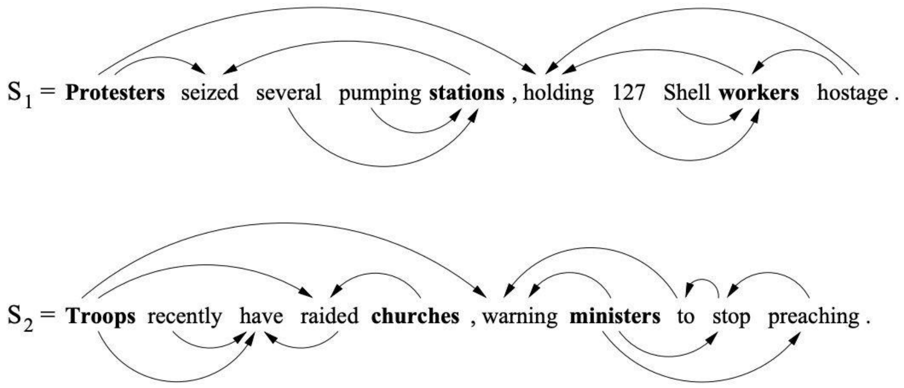
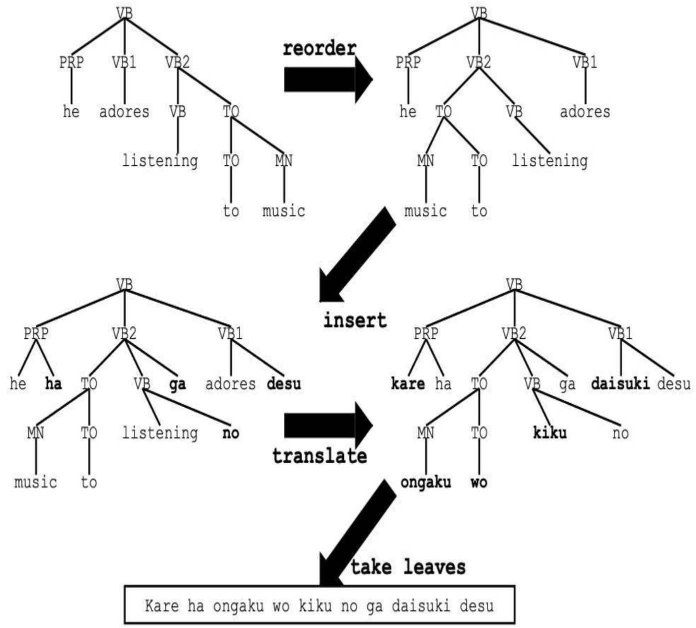


Figure 1: Sentences as dependency graphs.

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

Table 1: Shortest Path representation of relations.



[Bunescu and Mooney, 2005]

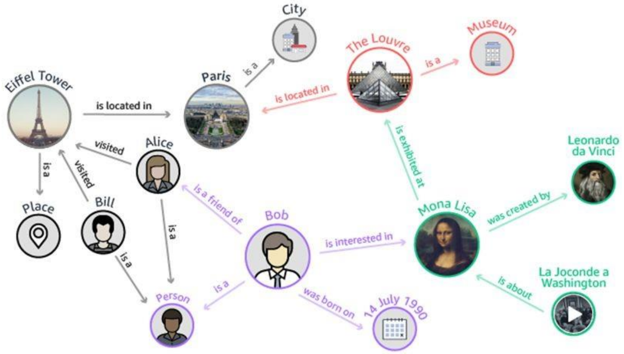
Syntax-based MT
[Yamada and Knight, 2001]

and many more ...

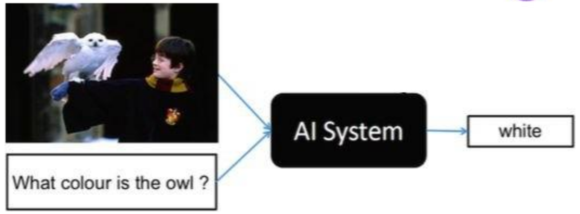
Applications of Graph Neural Nets in NLP

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

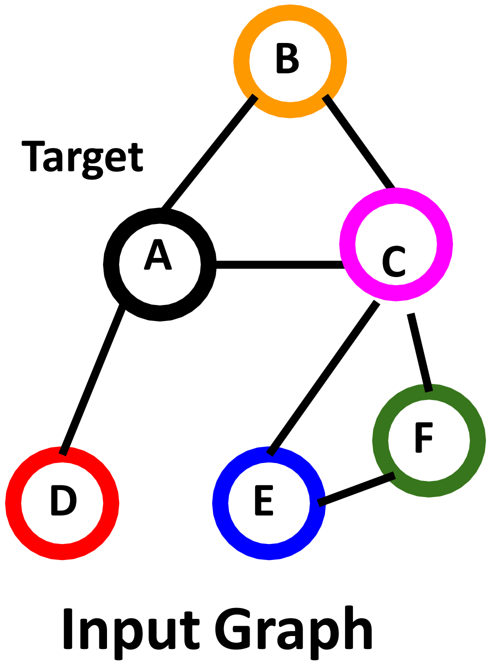
- Knowledge Graphs

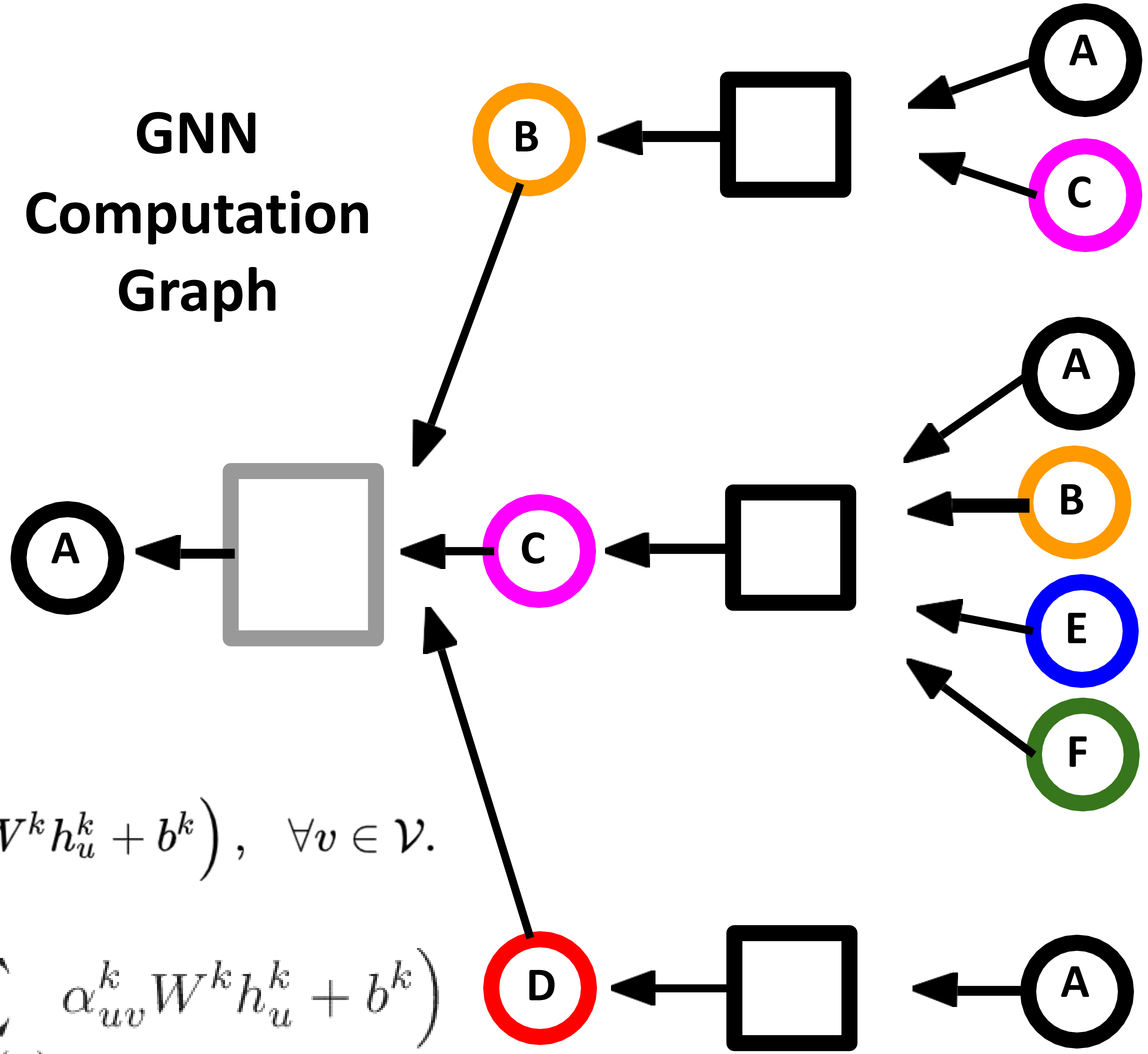


- Vision + NLP



GNNs for NLP





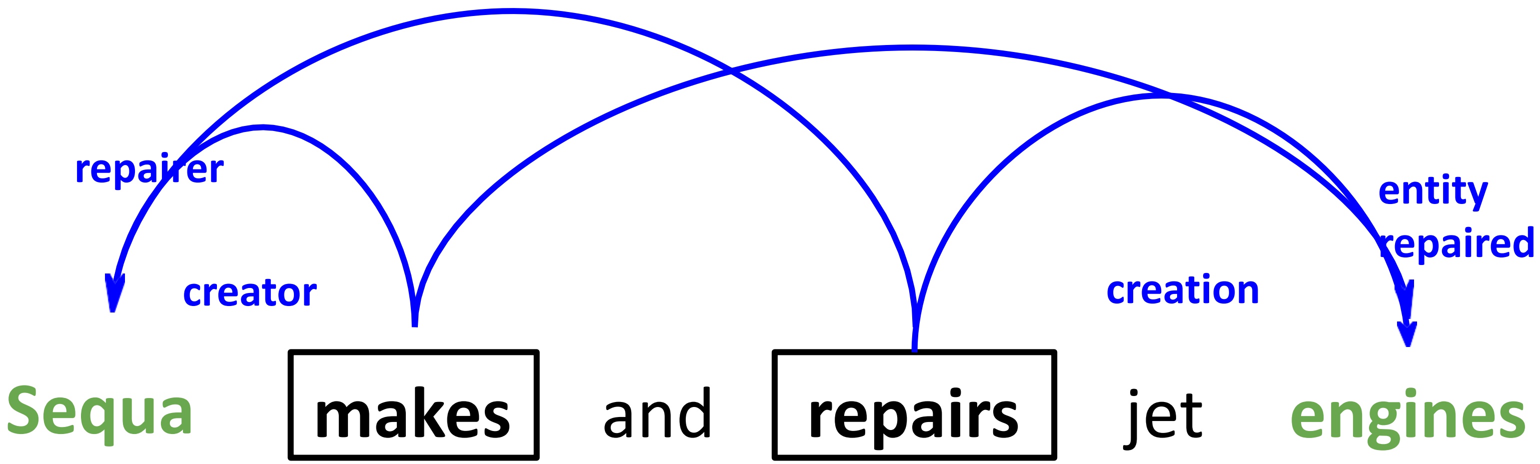
GCN $h_v^{k+1} = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W^k h_u^k + b^k\right), \quad \forall v \in \mathcal{V}.$

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

Semantic Role Labelling

Semantic Role Labelling

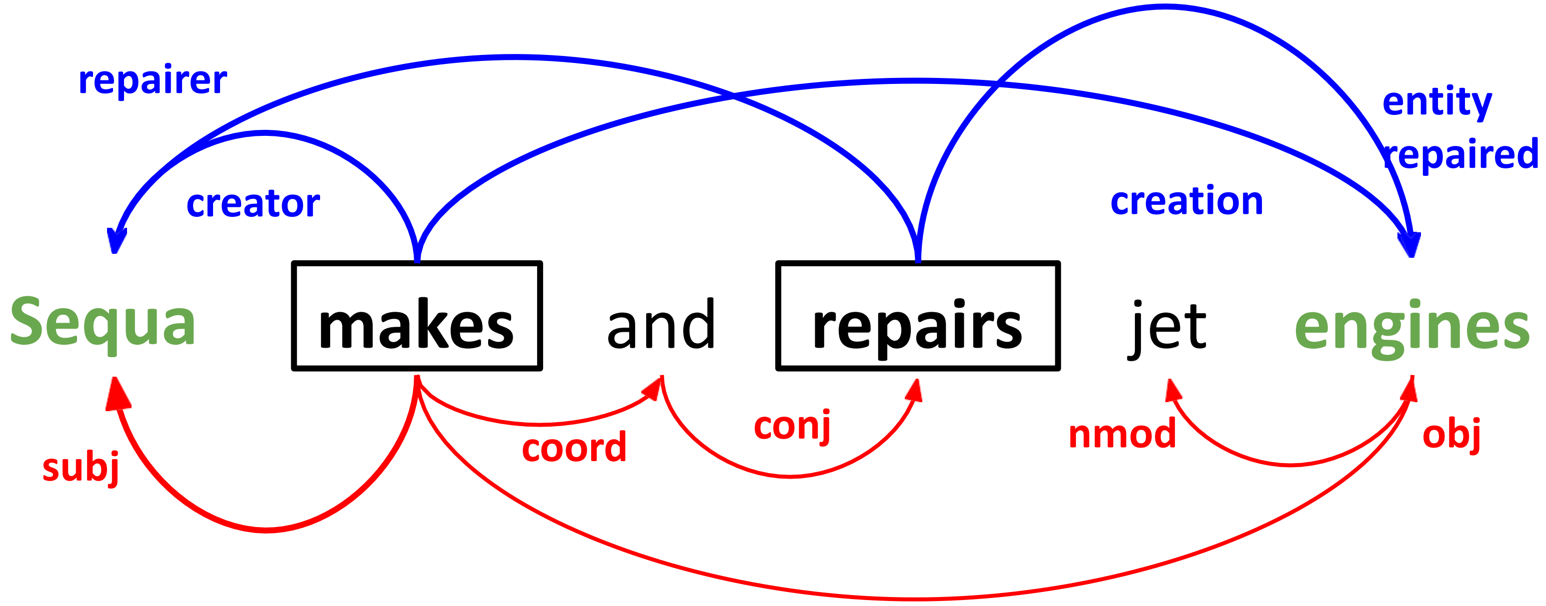
- In lay man terms, **WHO** did **WHAT** to **WHOM**?
- Important component for Natural Language Understanding
- What does SRL consist of?
  - Discover predicates
  - Identify arguments, their semantic roles
- Part of standard NLP Pipeline for QA, Information Extraction, NLU



Semantic Role Labelling

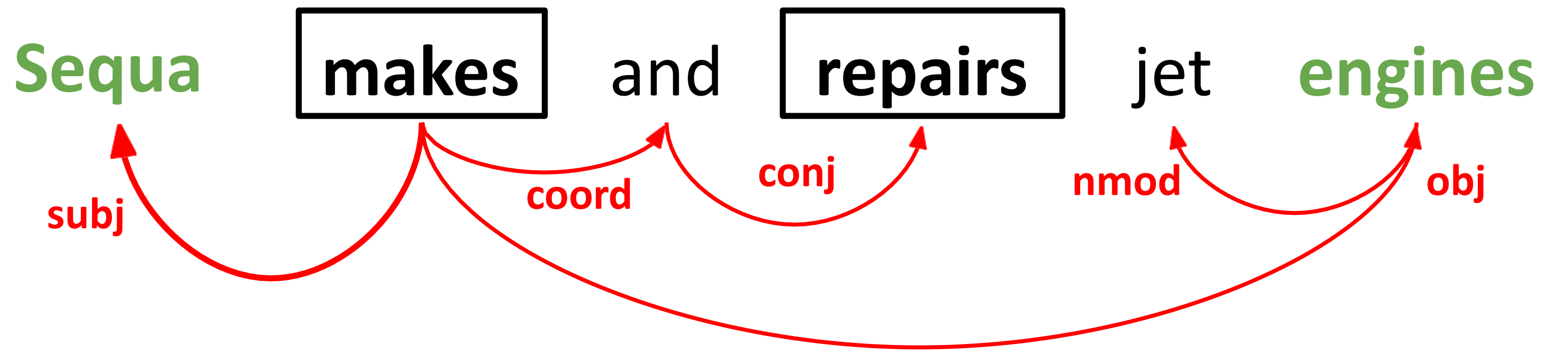
- Semantic role labeling (SRL) is the task of identifying the predicate-argument structure of a sentence.
- typically regarded as an important step in the standard NLP pipeline.
- Semantic representations are closely related to syntactic ones
- Syntactic information can be used to predict/improve semantics
- models exploit syntactic information to improve SRL performance
- How can we use GNNs to build encodings which can leverage syntactic information?

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



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

Syntactic GCNs [Marcheggiani et. al., EMNLP'17]



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

word emb of v

edge-wise gating

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

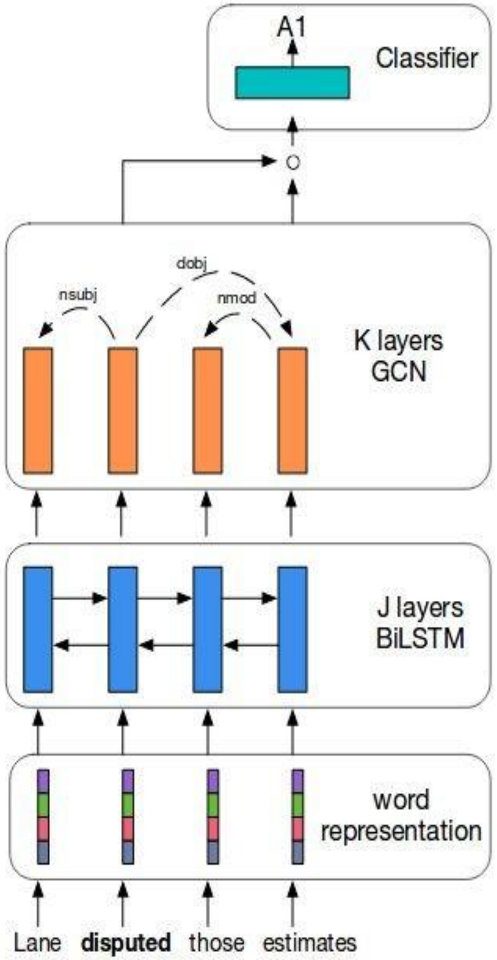
weight of direction

bias of label + direction

Syntax-Aware Neural SRL Encoder

1. look-ups of word embeddings;
2. a BiLSTM encoder that takes as input the word representation of each word in a sentence;
3. a syntax-based GCN encoder that re-encodes the BiLSTM representation based on the automatically predicted syntactic structure of the sentence;
4. a role classifier that takes as input the GCN representation of the candidate argument and the representation of the predicate to predict the role associated with the candidate word.

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



trained with cross-entropy

Arguments far away come closer because of syntactic arcs

Syntactic GCN

- We need to handle edge labels
- Separate set of parameters for each edge label

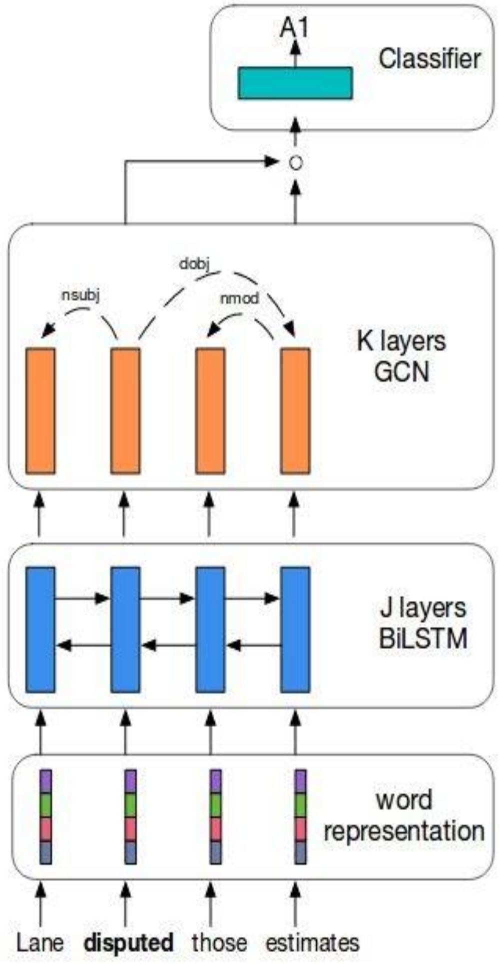
$$h_v^{(k+1)} = ReLU\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)} h_u^{(k)} + b_{L(u,v)}^{(k)}\right).$$

- Leads to over-parameterization – increases with edge labels

$$W_{L(u,v)}^{(k)} = V_{dir(u,v)}^{(k)}, \quad V_{dir(u,v)}^{(k)} \in \mathbb{R}^{m \times m}$$

- Different set of parameters for different edge types
- Different bias parameters for different edge labels
- Prevents parameter explosion

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



F1 on CoNLL-2009

BiLSTM	82.7
BiLSTM + GCN	**83.3**

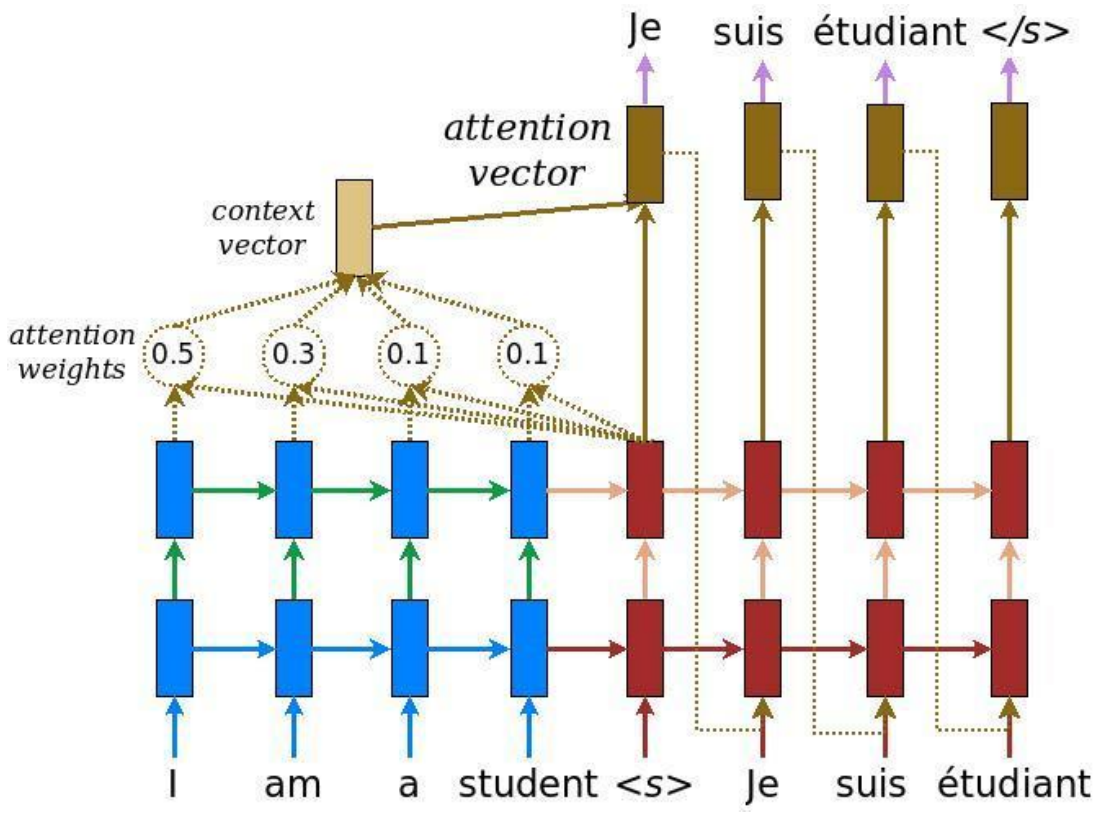
- GCN integrates syntax, context
- GCN, LSTM complement each other

Neural Machine Translation

Neural Machine Translation

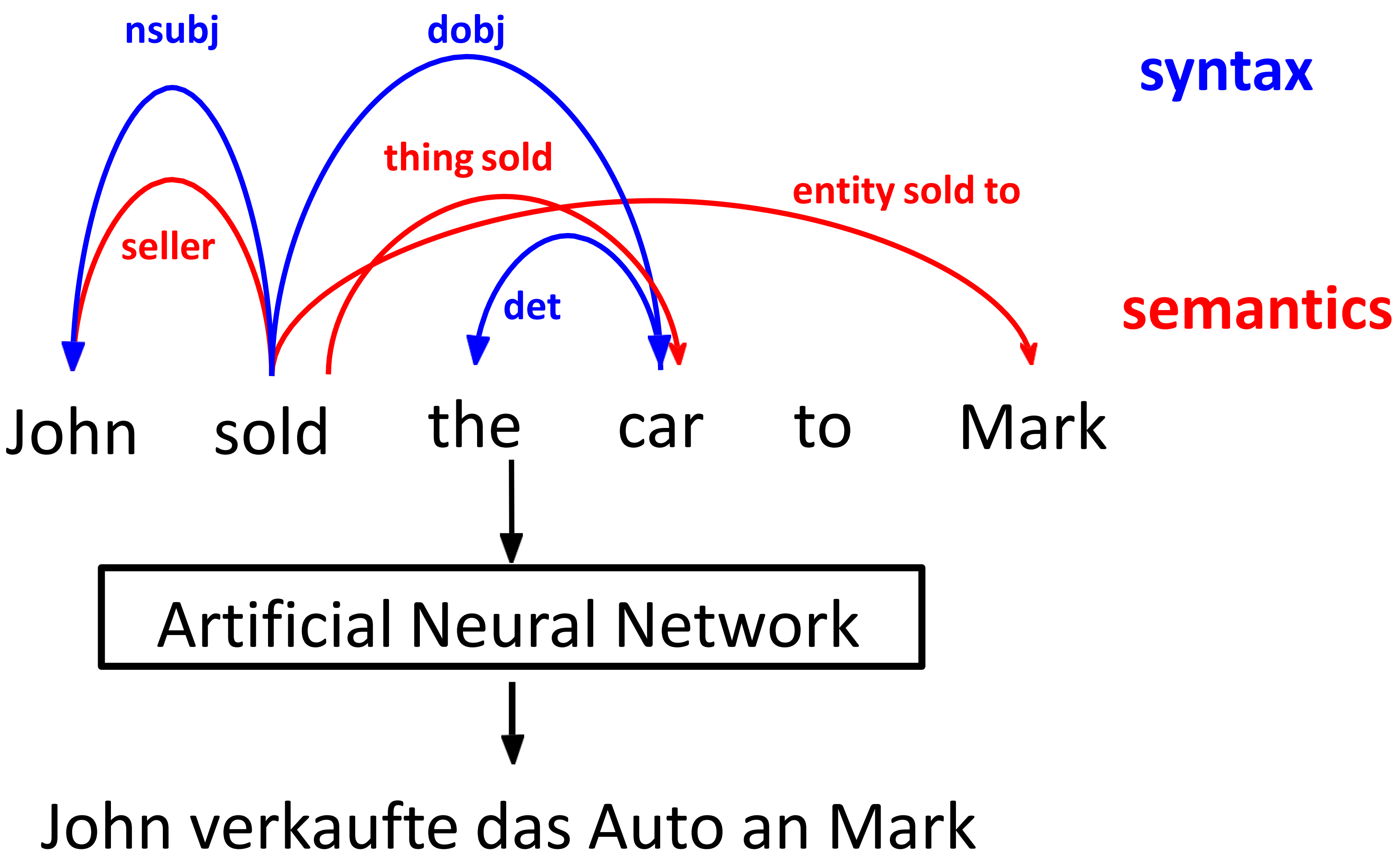
- Given example translation pairs from a parallel corpus, a neural network is trained to translate from source to target language
- it directly estimates the conditional distribution p(y[1:Ty]/x[1:Tx]) of translating a source sentence x[1:Tx] (a sequence of Tx words) into a target sentence y[1:Ty] (a sequence of Ty words)
- NMT models typically consist of an encoder, a decoder and some method for conditioning the decoder on the encoder

NMT Seq2Seq Model Overview

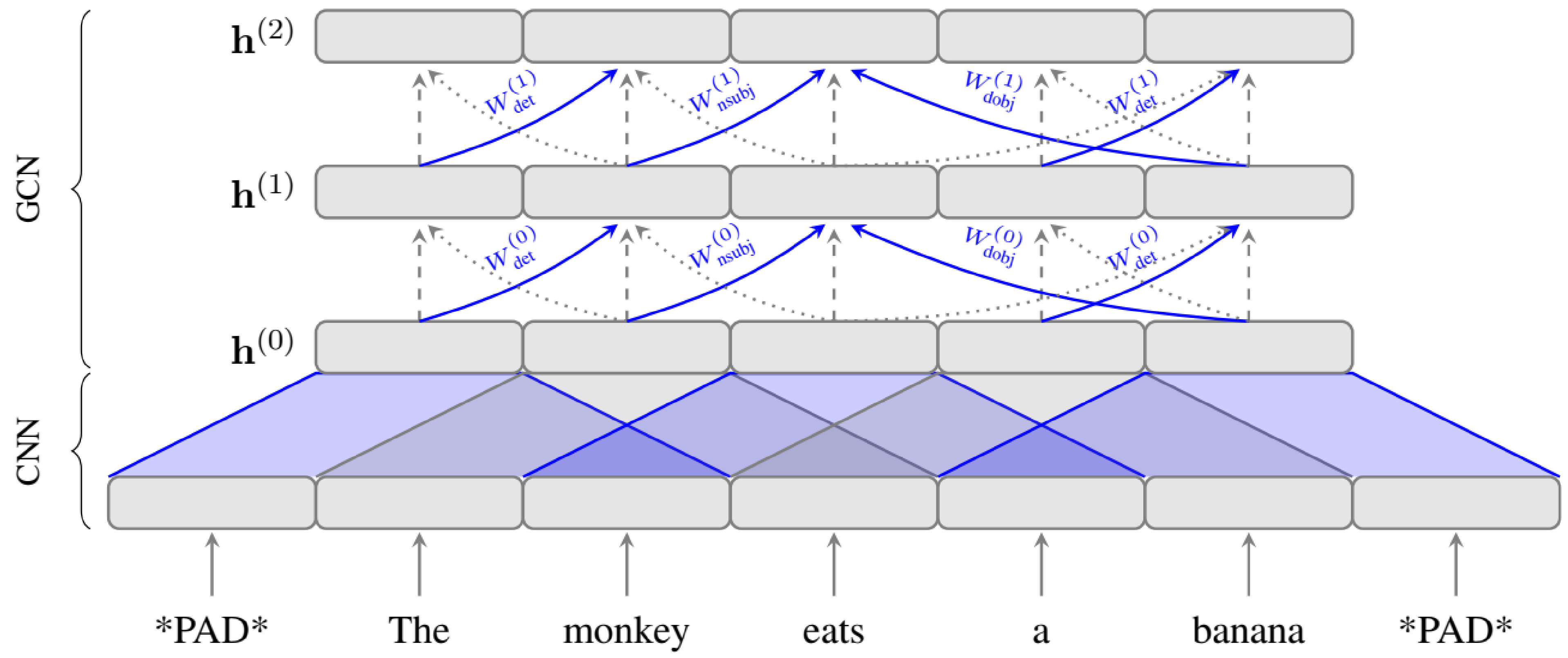


- Seq2Seq models typically contain an encoder, decoder and attention mechanism
- Encoder creates a distilled representation of input.
- Decoder generates the output based on the encoder outputs and each previously generated output symbol
- Attention weights selectively weigh the encoder outputs
- Each encoder/decoder block can be a CNN, RNN or a transformer block

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



GCNs for NMT


GCN
CNN
$\mathbf{h}^{(2)}$
$\mathbf{h}^{(1)}$
$\mathbf{h}^{(0)}$
$W^{(1)}_{\text{det}}$
$W^{(1)}_{\text{nsubj}}$
$W^{(1)}_{\text{dobj}}$
$W^{(1)}_{\text{det}}$
$W^{(0)}_{\text{det}}$
$W^{(0)}_{\text{nsubj}}$
$W^{(0)}_{\text{dobj}}$
$W^{(0)}_{\text{det}}$
PAD
The
monkey
eats
a
banana
PAD

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

English - German NMT on News Commentary

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

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

Addressing GCN Limitations

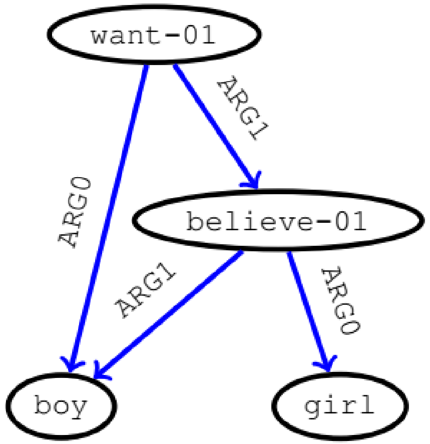
[Beck et al., ACL'18]

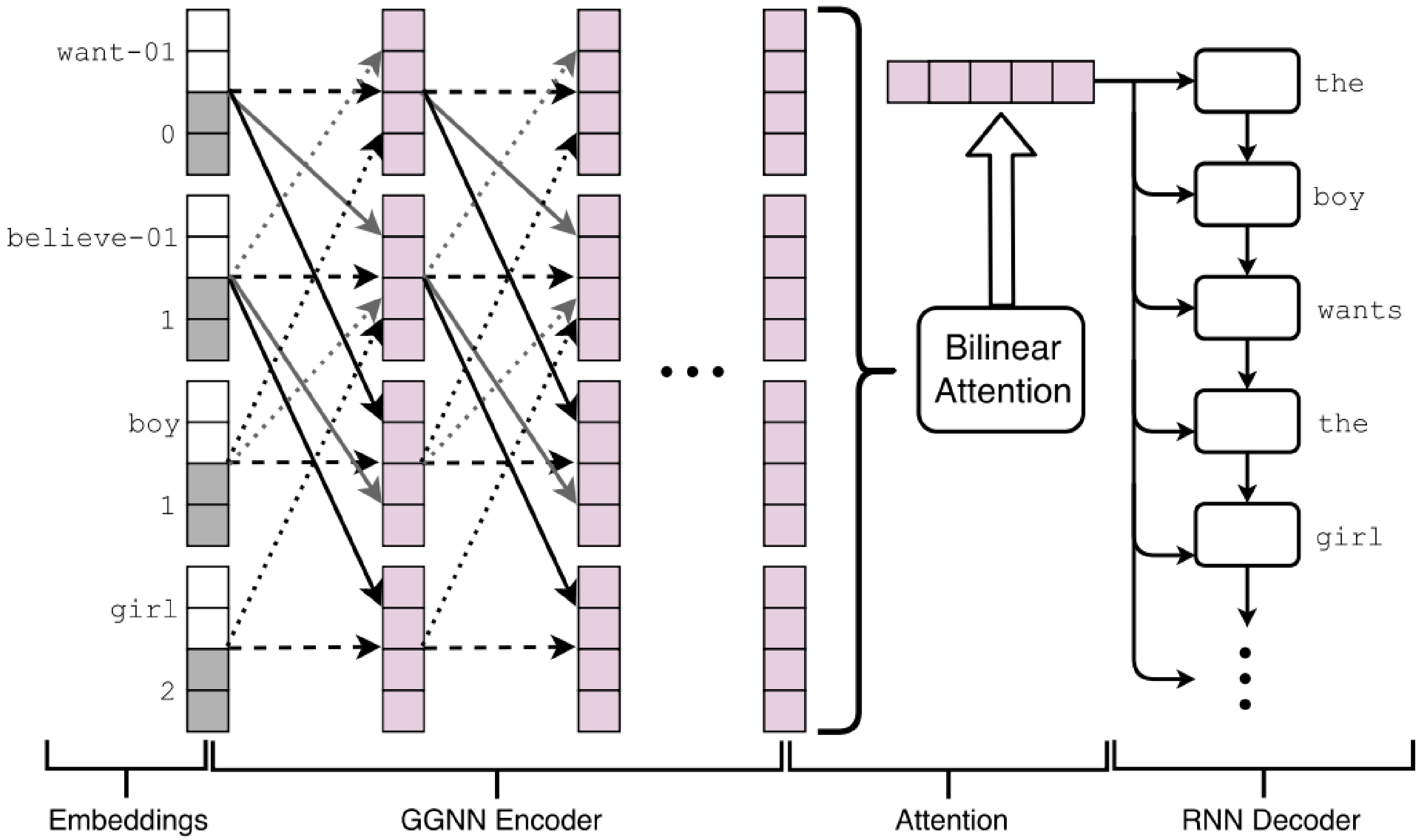
- **Limitations**
  - ✗ Parameters increase quadratically with # edge labels
  - ✗ No parameter sharing across layers
  - ✗ Edge labels are not encoded
- **GraphGRU (GGNN)**
  - ✓ Best of BiGRU + GCN worlds
  - ✓ Arbitrary # layers w/o increasing parameters

Gated Graph Neural Networks [Beck et al., ACL’18]

- Graph2Sequence learning instead of Seq2Seq learning
- Input is a graph and output is a surface form sequence
- Encoder based on Gated Graph Neural Networks (Li 2016)
- encoder is a GGNN that receives node embeddings as inputs and generates node hidden states as outputs, using the graph structure as context
- Bidirectionality and positional embeddings
  - POS embeddings are indexed by integer values representing the minimum distance from the root node and are learned as model parameters
  - Applicable for DAGs

GGNN based G2S learning





Gated Graph Neural Network

Beck et al., ACL'18

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

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

reset gate

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

update gate

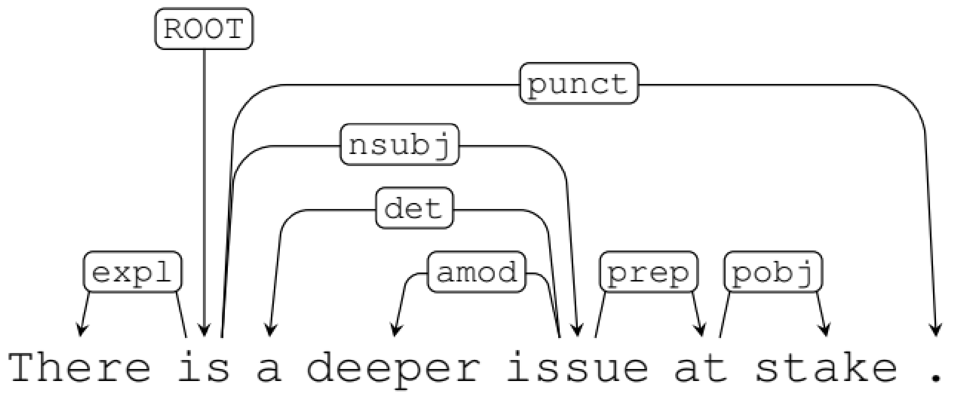
$$\widetilde{\mathbf{h}}_v^t = \rho\left(c_v \sum_{u \in \mathcal{N}_v} \mathbf{W}_{\ell_e}\left(\mathbf{r}_u^t \odot \mathbf{h}_u^{(t-1)}\right) + \mathbf{b}_{\ell_e}\right)$$

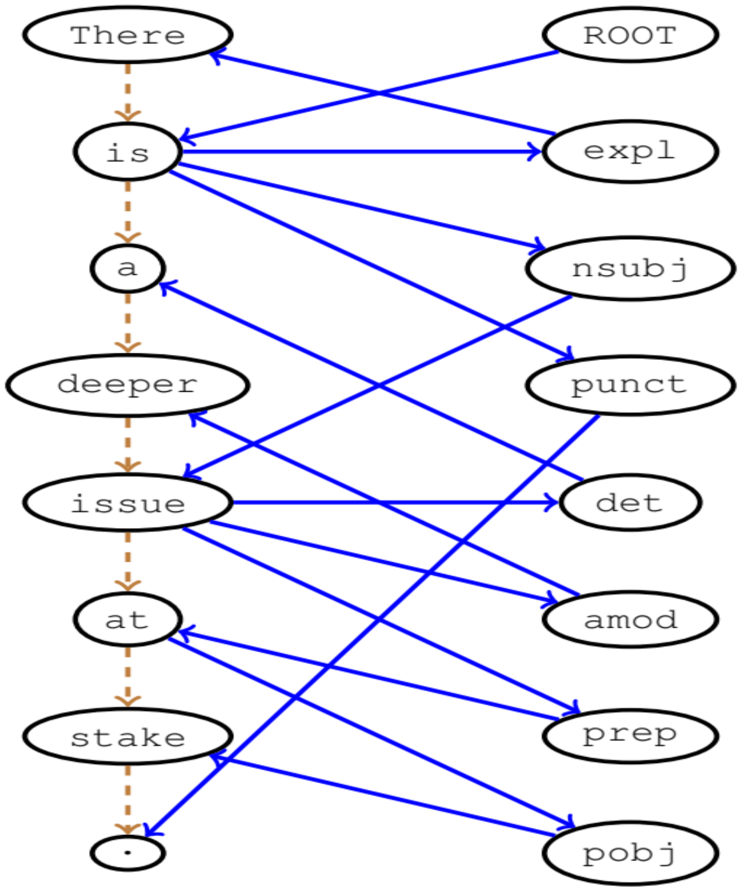
$$\mathbf{h}_v^t = (1 - \mathbf{z}_v^t) \odot \mathbf{h}_v^{(i-1)} + \mathbf{z}_v^t \odot \widetilde{\mathbf{h}}_v^t$$

Issues with Naïve G2S

- Increasing edge types can lead to parameter explosion
    - AMR, for instance has around 100 different predicates, which correspond to edge label
- Prior work combined edge labels → leads to loss of information
- edge label information encoded as GGNN parameters -> each label will have the same “representation” across all graphs.
- Ideally, edges should have instance-specific hidden states, in the same way as nodes
- Hence convert the edge label information using Levi Graph Transformation

Levi Graph Example





Levi Graph

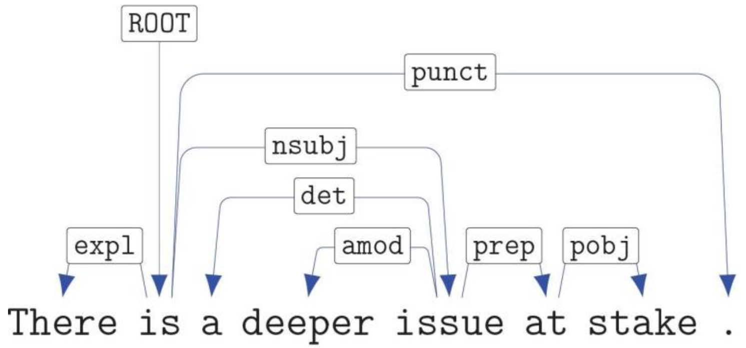
- Given a graph $G(V,E, L_V, L_E)$,

  a Levi graph[3] is defined as $\mathcal{G} = \{\mathcal{V}', \mathcal{E}', L_{\mathcal{V}'}, L_{\mathcal{E}'}\}$, where $\mathcal{V}' = \mathcal{V} \cup \mathcal{E}$, $L_{\mathcal{V}'} = L_{\mathcal{V}} \cup L_{\mathcal{E}}$ and $L_{\mathcal{E}'} = \varnothing$. The new edge set $\mathcal{E}'$ contains a edge for every (node, edge) pair that is present in the original graph. By definition, the Levi graph is bipartite.

- Since there are no labelled edges in LG, no parameter explosion
- Edges can also now have learnt embeddings
- Disad: Nodes and edge labels share the same embedding space

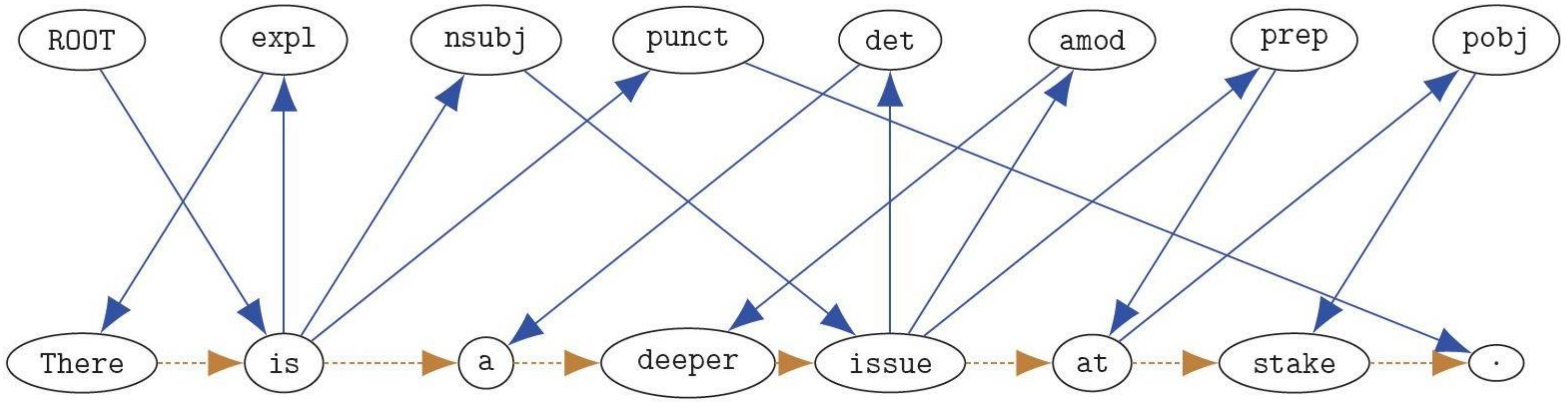
Levi graph [Beck et al., ACL'18]

E.g. syntactic dependency



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

✓ **Edge labels have hidden emb**



Levi graph

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

English - German NMT on News Commentary

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

Embedding edge labels is effective

Summary of GNNs for SRL, NMT

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

GNNs for Text Classification, Event Detection & Relation Extraction

GNNs for Text Classification, Extraction

Event Detection / Timestamping

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

Word Embedding / Text Classification

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

Sentiment Analysis

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

Relation Extraction

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

Sentiment Analysis using GNN

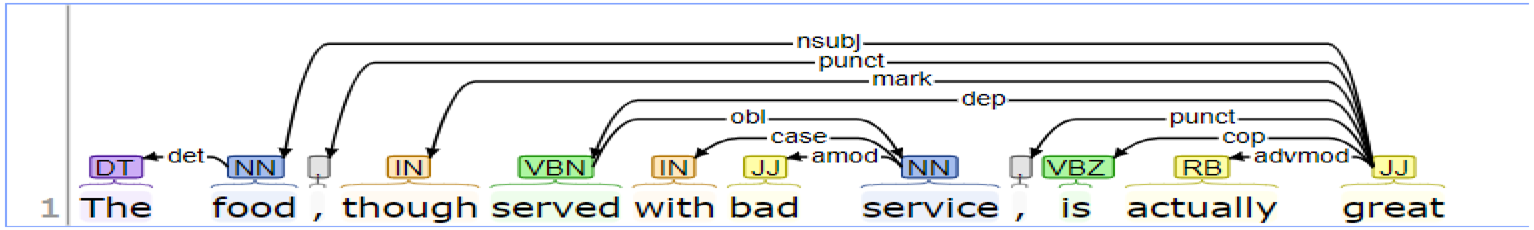
Aspect Level Sentiment Classification

- Aspect Level Sentiment Classification (ALSC)
- identify the sentiment polarity (eg. positive, negative, neutral) of an aspect target
- Sentence-level sentiment classification detect the overall sentiment in a sentence
- Aspect level sentiment is a more fine-grained task, detecting sentiment of each aspect
- ALSC distinguish sentiment polarity for multiple aspects in a sentence with various sentiment polarity

- "the service and ambience were great but the food quality was average"
- Aspects: Service (positive). Ambience (positive), Food(Negative)

Aspect Level Sentiment using GNNs

- "The food, though served with bad service, is actually great"
- Aspect food and sentiment word great are separated in word sequence
- But they are closer in the dependency graph



- Use Graph Neural Networks to capture syntax to improve ALSC

Aspect Level Sentiment using GNNs

- Transform sentence into its dependency graph with N nodes
- Each node - a word (embedding) as its local feature vector x
- Use GAT to propagate syntax features to its aspect node
  - compute node representations by aggregating neighbourhood's hidden states
  - With an L-layer GAT network, features from L hops away can be propagated

$$H_{l+1} = GAT(H_l, A; \Theta_l) \tag{3}$$

where $H_l \in R^{N \times D}$ is the stacked states for all nodes at layer l, $A \in R^{N \times N}$ is the graph adjacent matrix. Θ_l is the parameter set of the GAT at layer l.

Event Detection

Event Detection

- Event Detection (ED) - recognize instances of specified types of events (event mentions) in text.

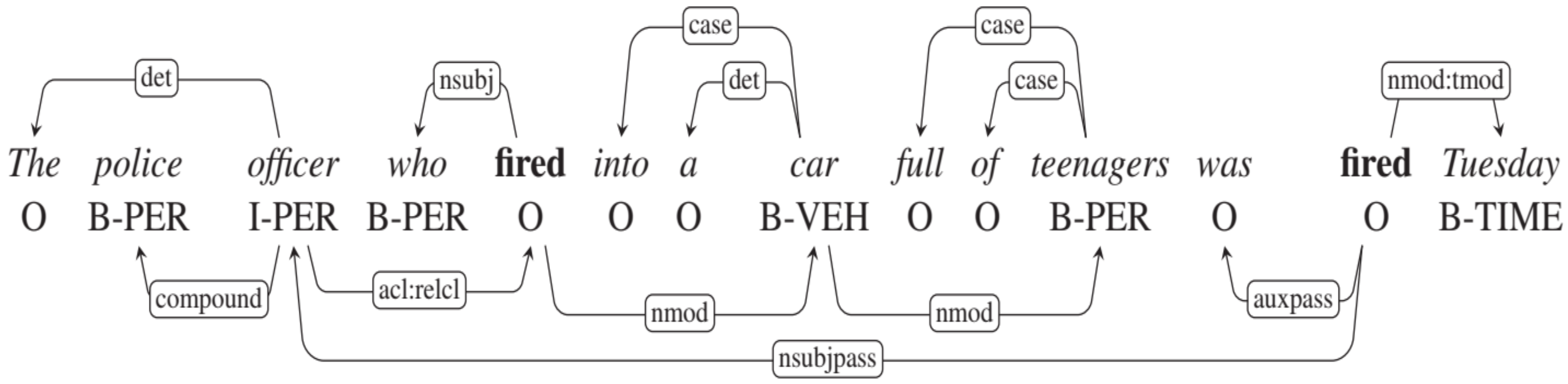
  ‘The Police Officer fired at the unruly crowd’

  Event Mention – ‘Fired”

  Event Type – “Attack”
- Each event mention is often present in a single sentence in which an event trigger is selected to associate with that event mention.
- Event triggers are generally single verbs or nominalizations
  - serve as the main words to evoke the corresponding events.
- The event detection task aims to detect event triggers and classify them into specific types of interest
- “***The police officer who fired into a car full of teenagers was fired Tuesday***”
  - In this example, an ED system should be able to realize that the first occurrence of “*fired*” is an event trigger of type *Attack* while the second “*fired*” takes *End-Position* as its event type.
- ED is complex
  - an expression might evoke different events depending on contexts (“fired”)
  - same event might be presented in various expressions (e.g, the trigger words “killed”, “shot” or “beat for the event type Attack).

Event Detection (ED) using GCN

- CNNs were used originally for ED
- But event mentions may span non-consecutive k-grams
- “The **police officer**, who **fired** into a car full of teenagers, was **fired** yesterday”
- non-consecutive 3-grams “officer was fired” should be considered to correctly identify the event type End-Position for the second word “fired”
- Considering all non-consecutive k-grams and then pooling can be noisy
- Can we perform the convolution operation over the syntactic dependency graphs?



Event Detection (ED) using GCN

- Three major modules for using GCNs in ED
- Encoding module that represents the input sentence with a matrix for GCN computation
  - Each word represented by std. word emb, position embedding and its entity embedding
  - This initial representation can then be abstracted using a BI-LSTM
- the convolution module that performs the convolution operation over the dependency graph structure of *w* for each token in the sentence
- the pooling module that aggregates the convolution vectors based on the positions of the entity mentions in the sentence to perform ED.

Event Detection (ED) using GCN

Create Initial encoding with word embeddings, position embeddings and entity type embeddings

Abstracting the initial encoding with bidirectional LSTM

Performing convolution over the dependency trees using the BiLSTM representation

Pooling over the convolution vector based on the positions of the entity mentions

Feed-forward neural networks with softmax for prediction

Event Detection using GNNs

Nguyen and Grishman, AAAI'18

ATTACK

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

END-POSITION

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

BiLSTM	70.5
BiLSTM + Syntactic GCN	**71.4**

GCN, LSTM complement each other

Multiple Event Extraction (MEE)

- Event Extraction (EE) task can be divided into two subtasks
  - event detection (identifying and classifying event triggers)
  - argument extraction (identifying arguments of event triggers and labeling their roles)
- multiple events commonly exist in the same sentence.
- MEE - certain co-occurrence phenomena of events
  - Injure and Die events are more likely to co-occur with Attack events than others, whereas Marry and Born events are less likely to co-occur with Attack event

MEE using GNNs

Liu et al., EMNLP'18

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

END-POSITION ✗

TRANSPORT ✓

TRANSPORT

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

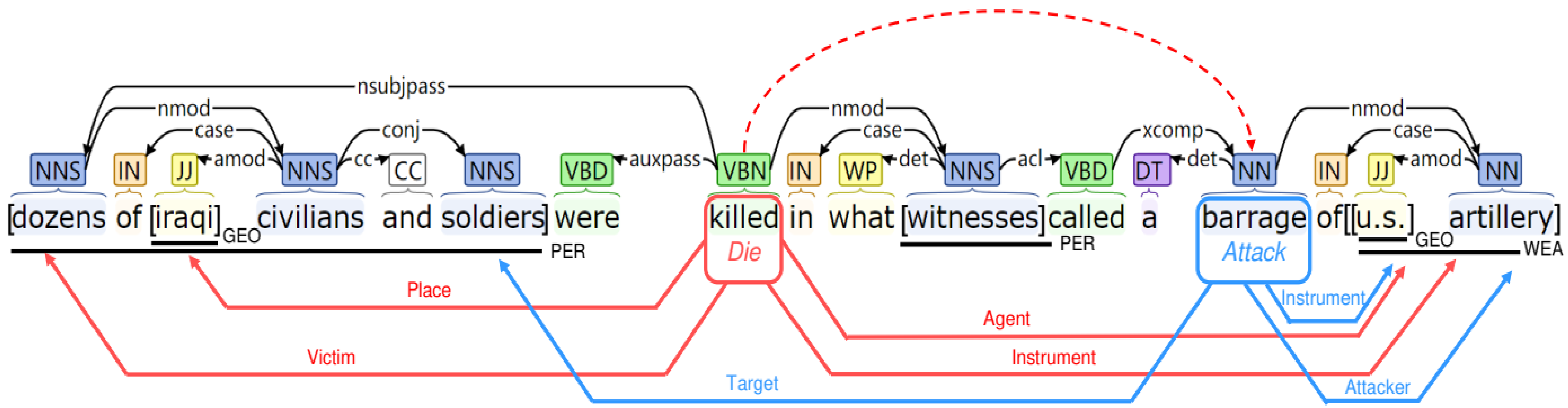
MEE techniques

- Standard sequence modelling of sentences are typically used for MEE
- Other option is to use feature based techniques
- However, sentence level sequential modeling methods suffer a lot from the low efficiency in capturing very long range dependencies
- the feature-based methods require extensive human engineering, which also largely affects model performance.
  these methods do not adequately model the associations between multiple events in sentence
- can we use dependency parsing info to enhance joint MEE?

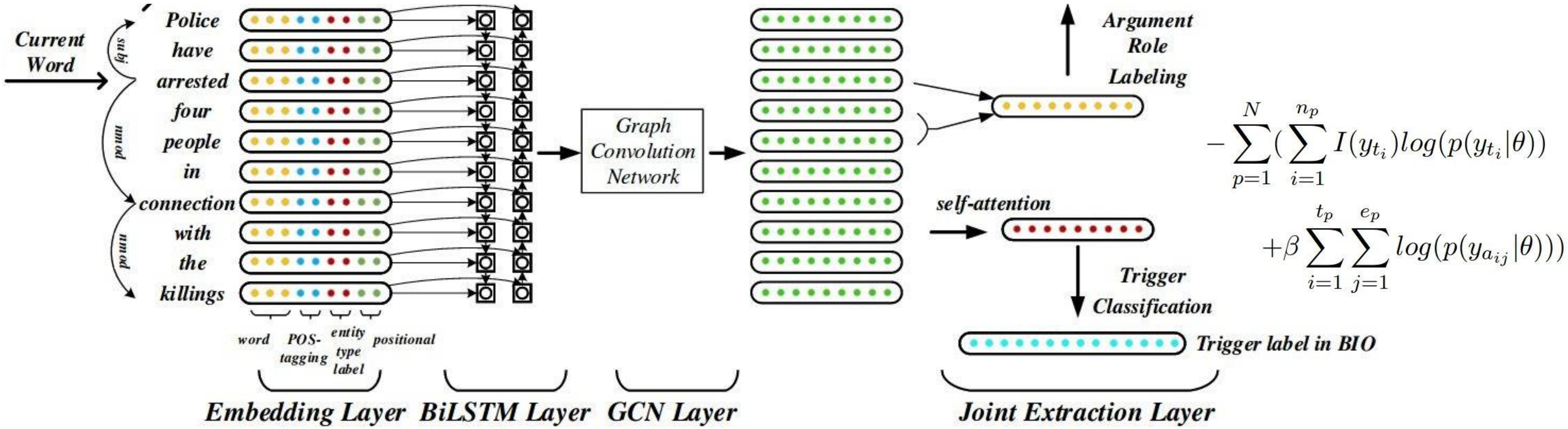
Event Extraction - Task Definition

- event extraction can be cast as a multiclass classification problem
    - whether each word in the sentence forms a part of event trigger candidate
    - whether each entity in the sentence plays a particular role in the event
- Two main approaches to event extraction:
    - the joint approach that extracts event triggers and arguments simultaneously as a structured prediction problem,
    - the pipelined approach that first performs trigger prediction and then identifies arguments in separate stages
- Joint approach avoids error propagation in the pipeline

Use of Dependency Arcs in MEE



MEE using GNNs Liu et al., EMNLP’18



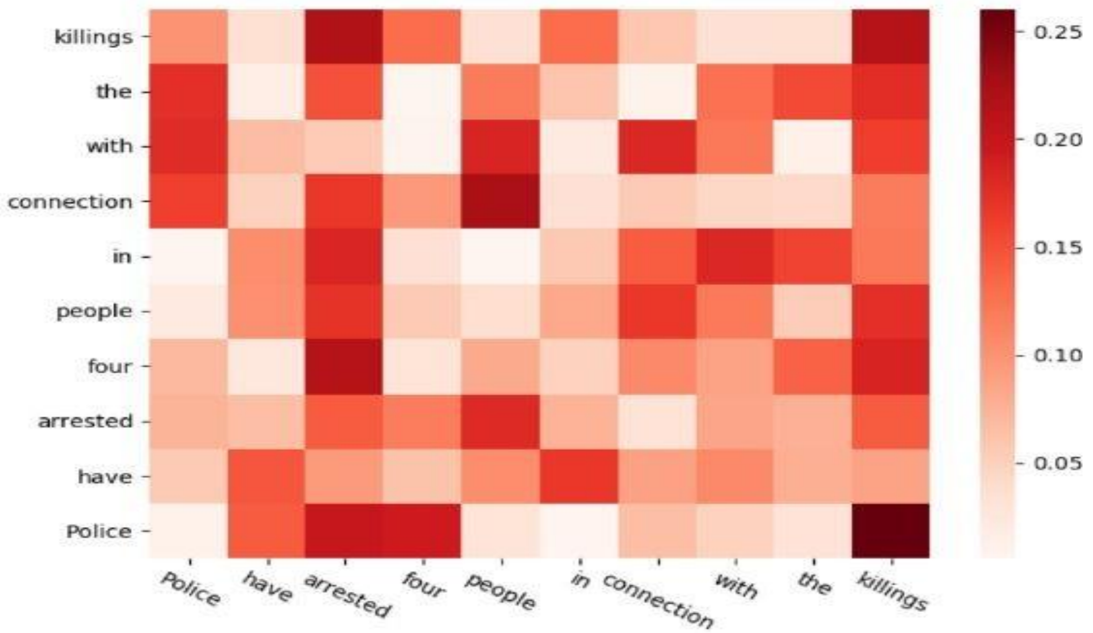
Performance Results Liu et al., EMNLP'18

F1 on the ACE 2005 dataset

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

GCN, LSTM complement each other

high



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

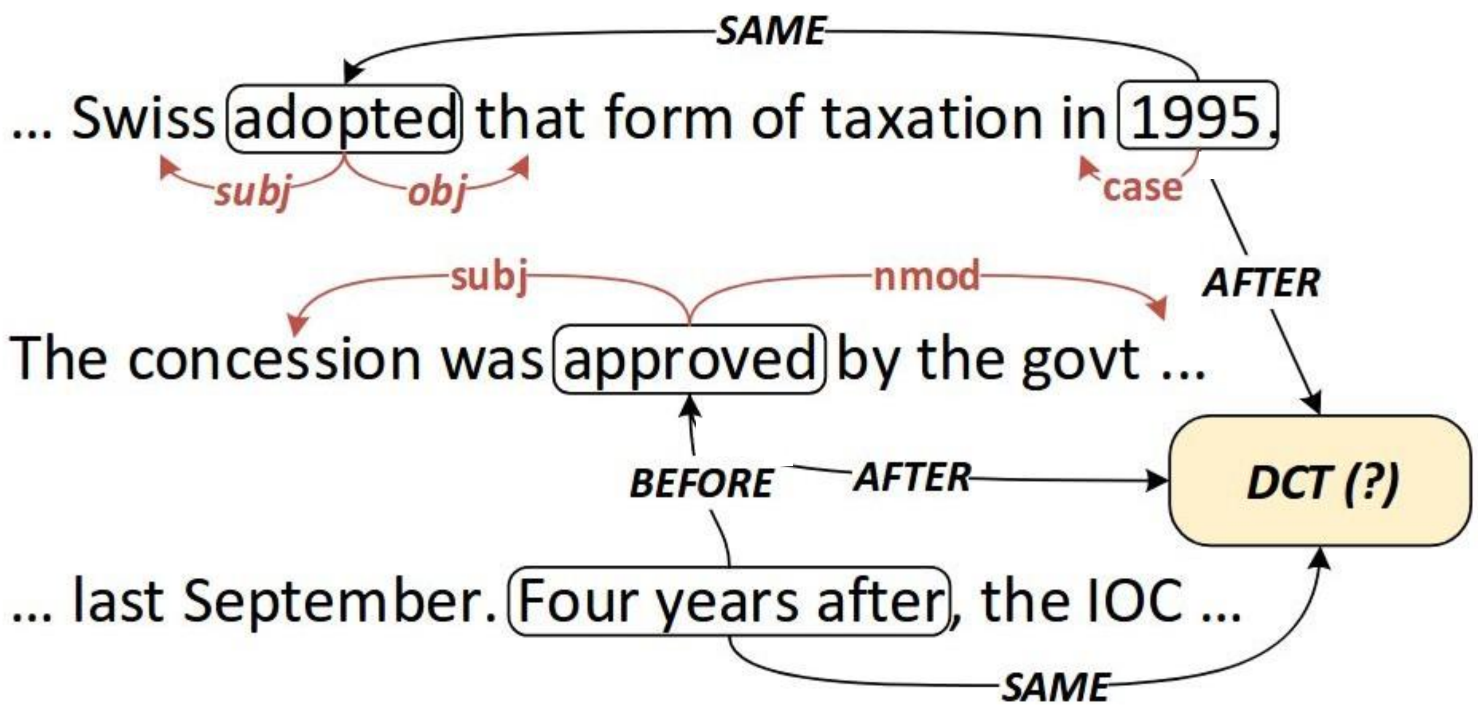
Time Stamping of Documents

Time Stamping of Documents

- Date of a document - Document Creation Time (DCT)
- Tasks such as information retrieval, temporal reasoning, event detection and analysis of historical text require correct DCT
- need to automatically predict the date of a document based on its content.
- This problem is referred to as *Document Dating*.
- requires extensive reasoning over the temporal structure of the document

Document time stamping

Vashishth et al., ACL'18



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

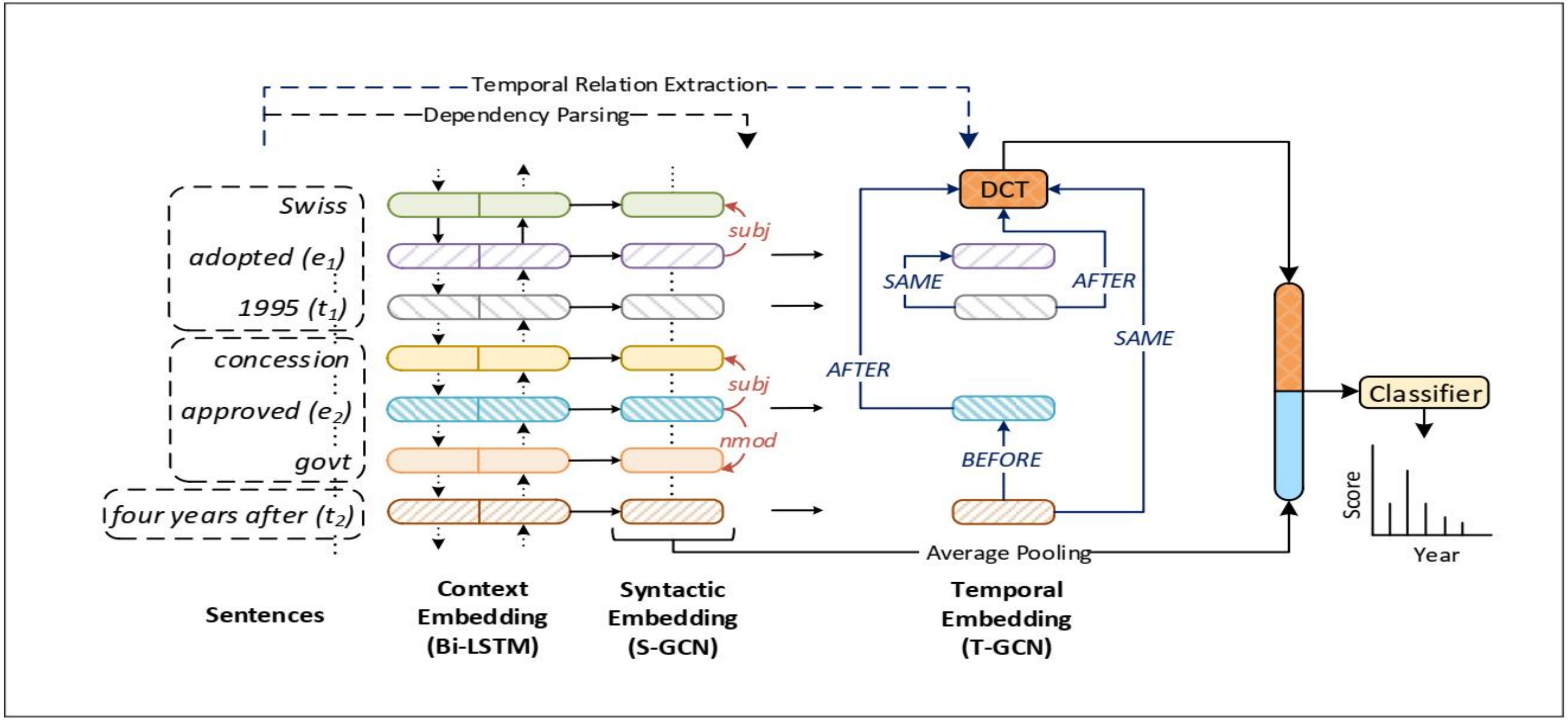
Document Dating Approaches

- Use of discriminative models using handcrafted temporal features
- Use of statistical model using term burstiness
- Application of Neural Networks
- NNs with temporal and syntactical dependences

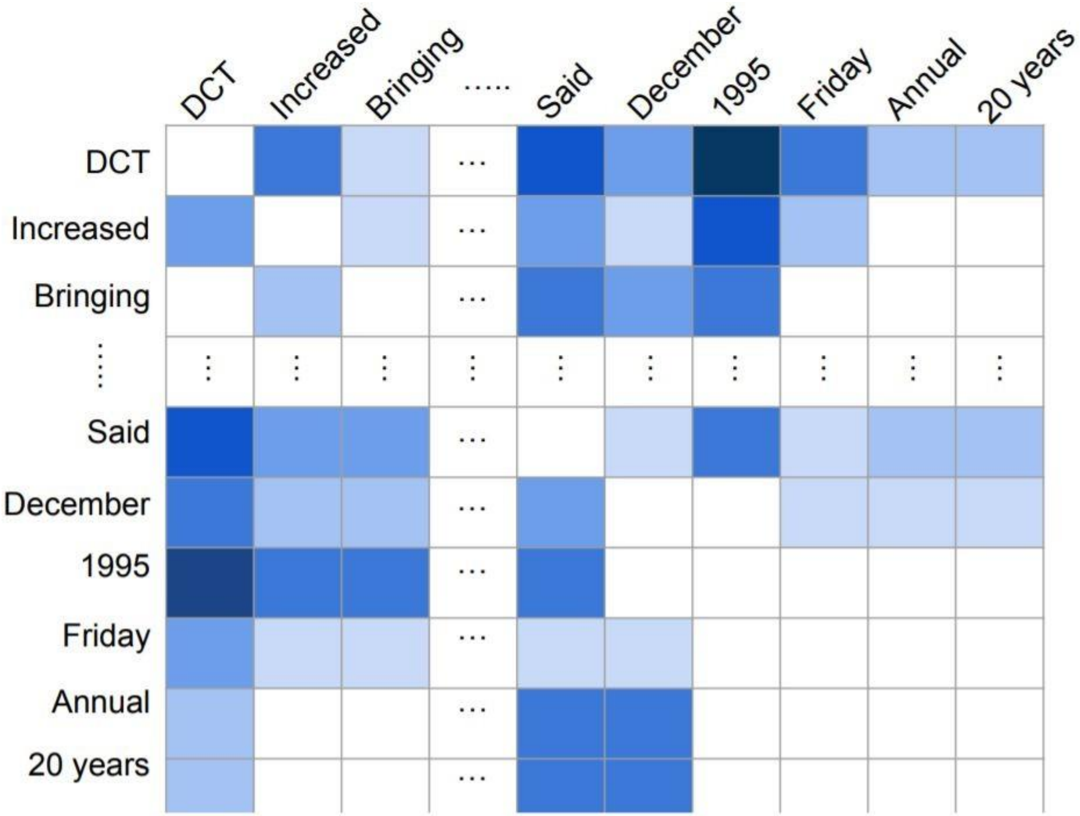
Neural Document Dater

- NeuralDater network consists of three layers
  - Context Embeddings – Flat Context (BiLSTM)
  - Syntactic Embeddings – Dependency Graph (S-GCN)
  - Temporal Embeddings – Temporal Document Graph (T-GCN)
- Temporal Graph Creation
  - Uses SUTime tagger of Stanford coreNLP for time/date annotations
  - Uses CATENA algorithm for temporal document graph creation
- Learns an embedding for the Document Creation Time (DCT) node corresponding to the document in the T-GCN
- DCT node embedding and average pooled S-GCN document embedding are concatendated
- fed to a fully connected softmax classifier which makes the final prediction about the date of the document

Neural Document Dater



Document Time Stamping



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

Associated Press Worldstream

Relation Extraction

Relation Identification Task

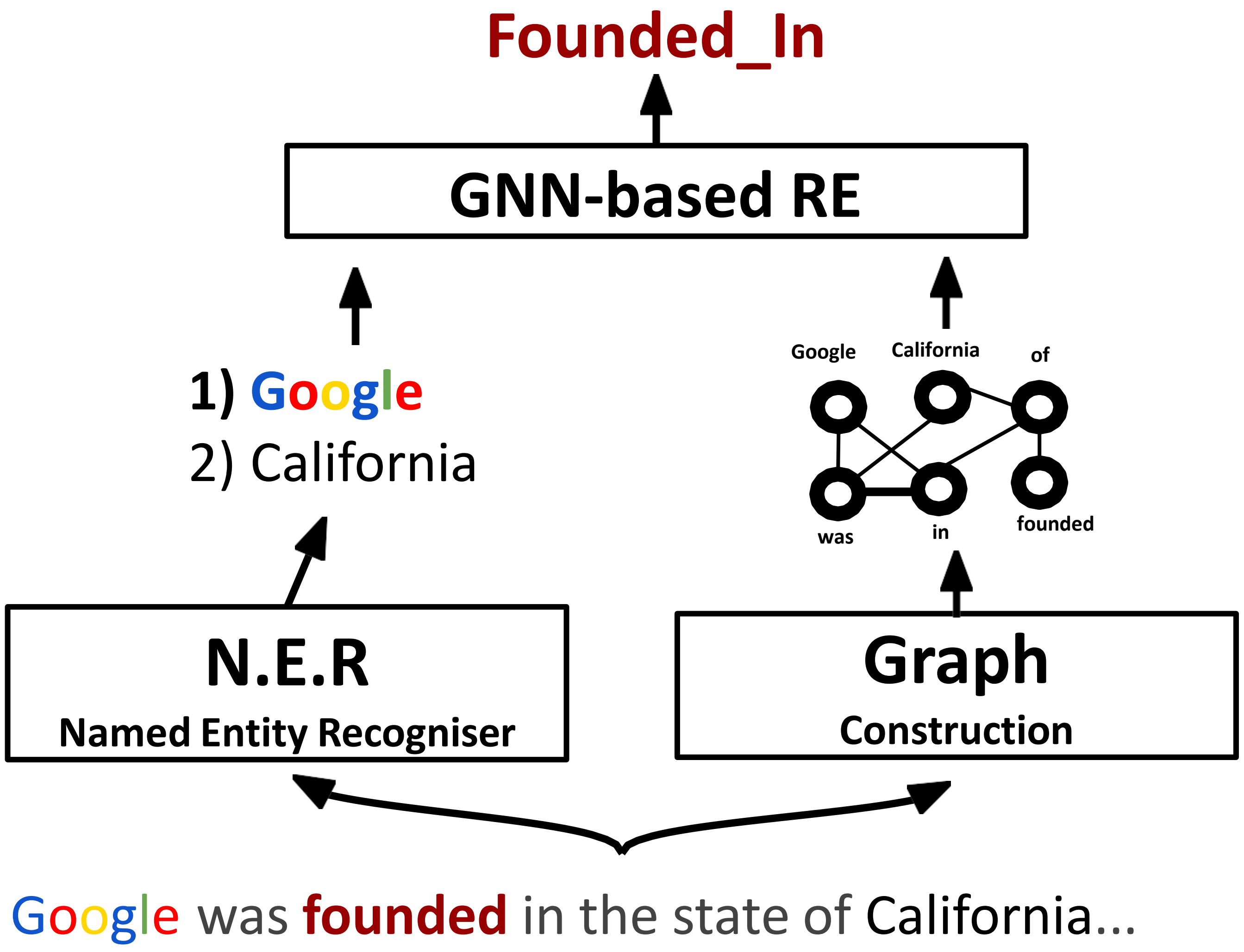
- Identify **relation** between entities.

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

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

Relation Extraction

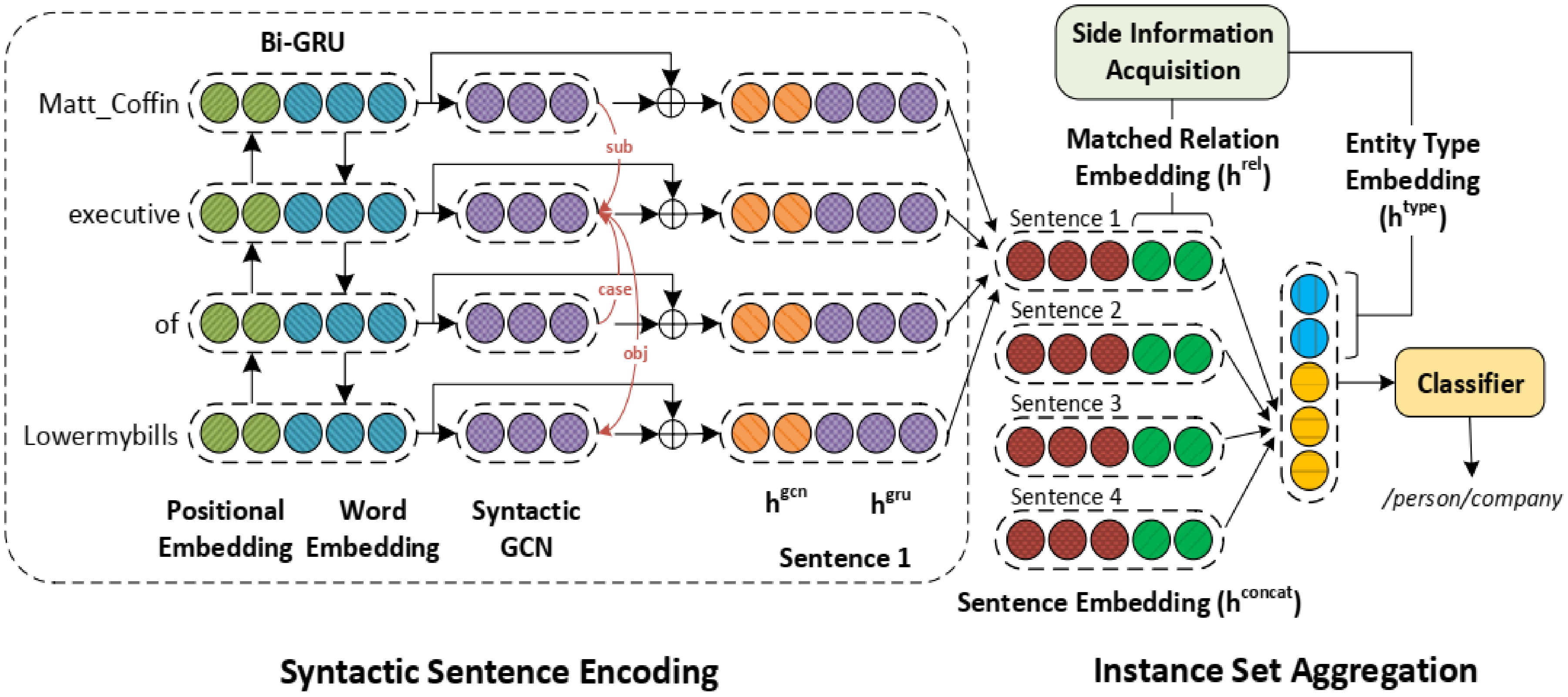
- Relation Extraction (RE) = extracting semantic relationships between entity pairs from plain text.
- This task can be modeled as a simple classification problem after the entity pairs are specified.
- Formally, given an entity pair (e1,e2) from the KB and an entity annotated sentence (or instance), we aim to predict the relation r, from a predefined relation set, that exists between e1 and e2.
- Supervised techniques require large labelled data
- Typically addressed by using distant supervision – but noisy
- Can we use syntactic information and side information from text to reduce noisiness?


Founded_In
GNN-based RE
1) Google
2) California
Google
California
of
was
in
founded
N.E.R
Named Entity Recogniser
Graph
Construction
Google was founded in the state of California...

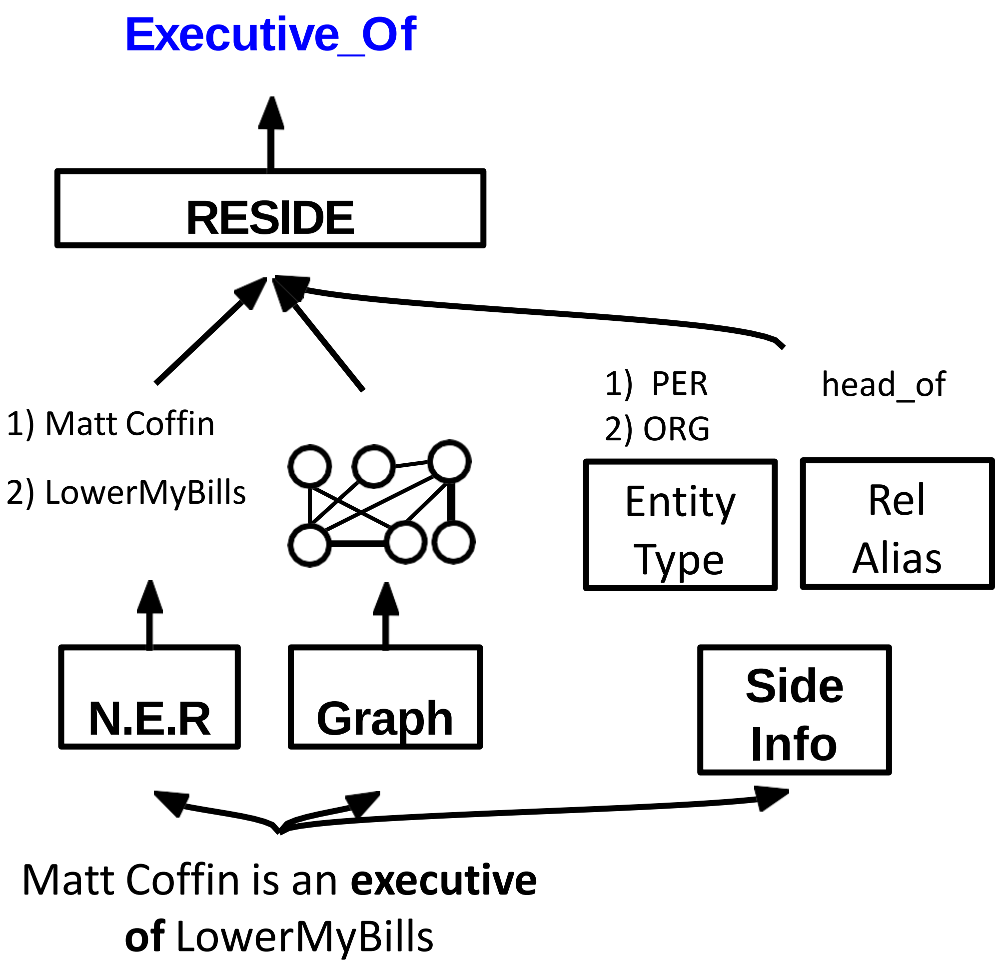
Using side information for relation extraction

- KBs often contain relevant side information
- aliases of relations
  - e.g., founded and co-founded are aliases for the relation founderOfCompany
- Types of entities involved
  - Microsoft was started by Bill Gates.
  - The type information of Bill Gates (person) and Microsoft (organization) can be helpful in predicting the correct relation founderOfCompany

RE Using Side Information



Relation Extraction using Side Info Vashishth et al., EMNLP'18

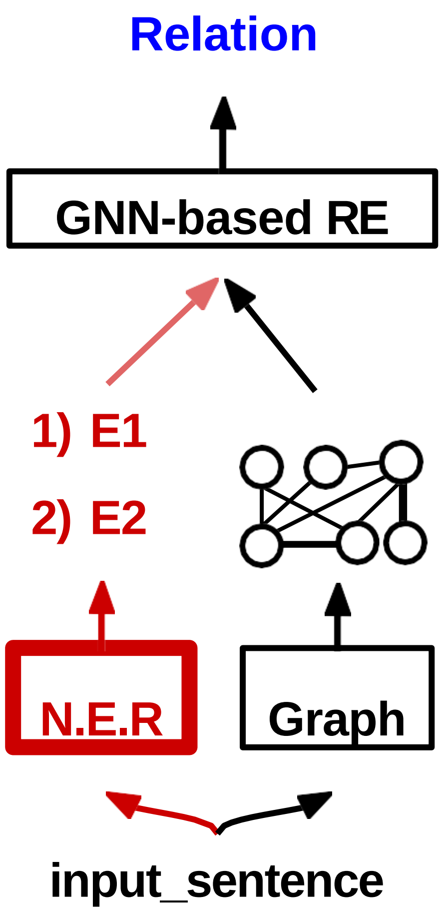


P@300 on Riedel dataset

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

Even limited side info improves performance

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



× **Assumes an N.E.R**

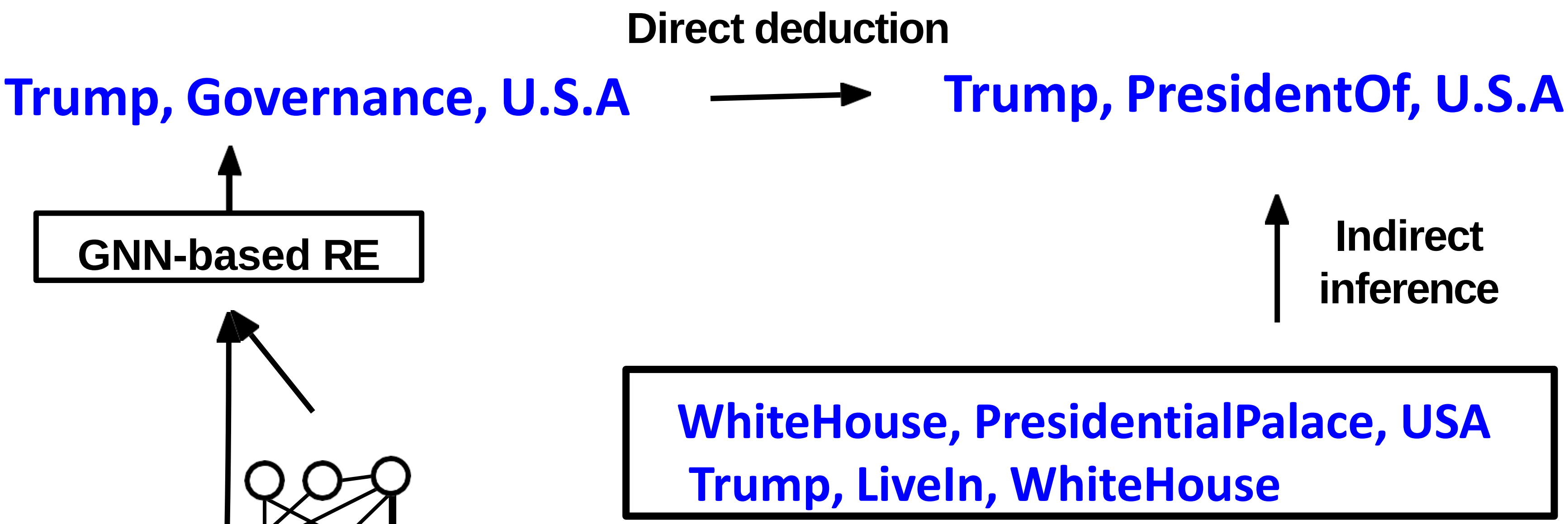
Errors are propagated without any feedback

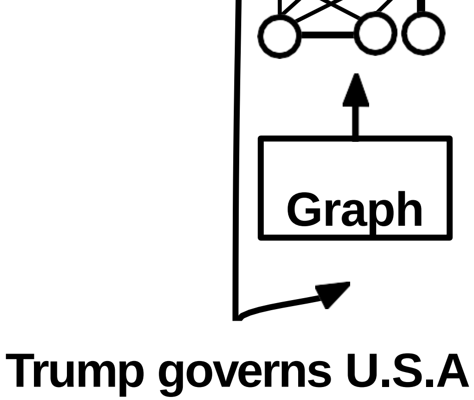
Leveraging overlapping relations

- Entity Pair Overlap
  - (*BarackObama*, *PresidentOf*, *UnitedStates*) <-> (*BarackObama*, *Governance*, *UnitedStates*)
- Single Entity Overlap
- Given the two relations (*BarackObama*, *LiveIn*, *WhiteHouse*) and (*WhiteHouse*, *PresidentialPalace*, *UnitedStates*),
  → (*BarackObama*, *PresidentOf*, *United States)*

  → (*BarackObama*,Governance, *United States)*
- Joint learning of entities and relations
- Two phase Relation Extraction to leverage overlapping relations

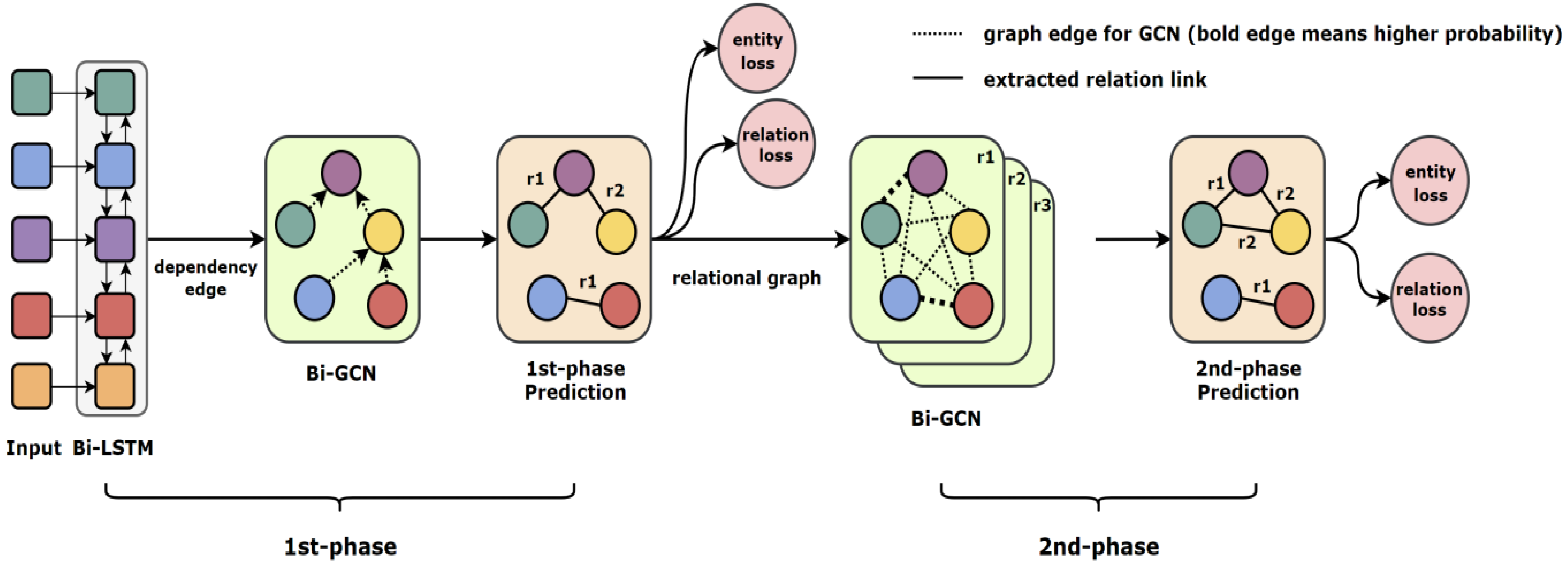
Joint Entity and RE Fu et al., ACL’19





- Facilitates Knowledge Base Completion

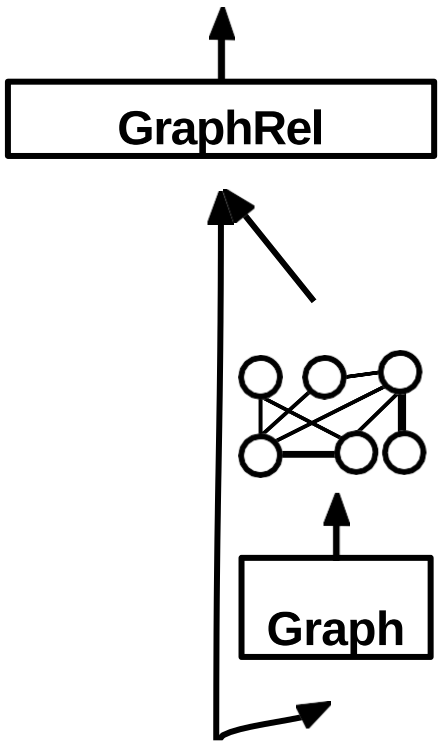
GraphRel Joint Entity and Relation Extraction



Joint Entity & RE Fu et al., ACL'19

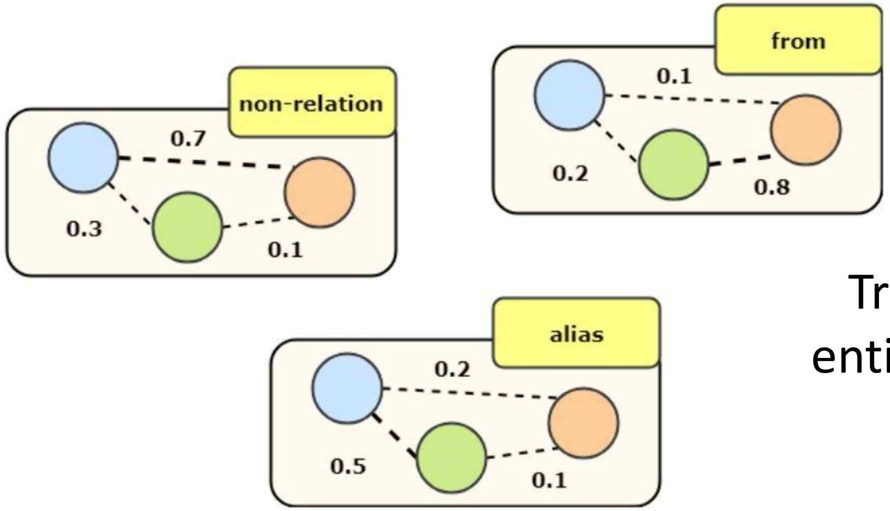
Direct deduction

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



Trump governs U.S.A

Idea: GCN on relational graphs



Trained with entity + relation losses

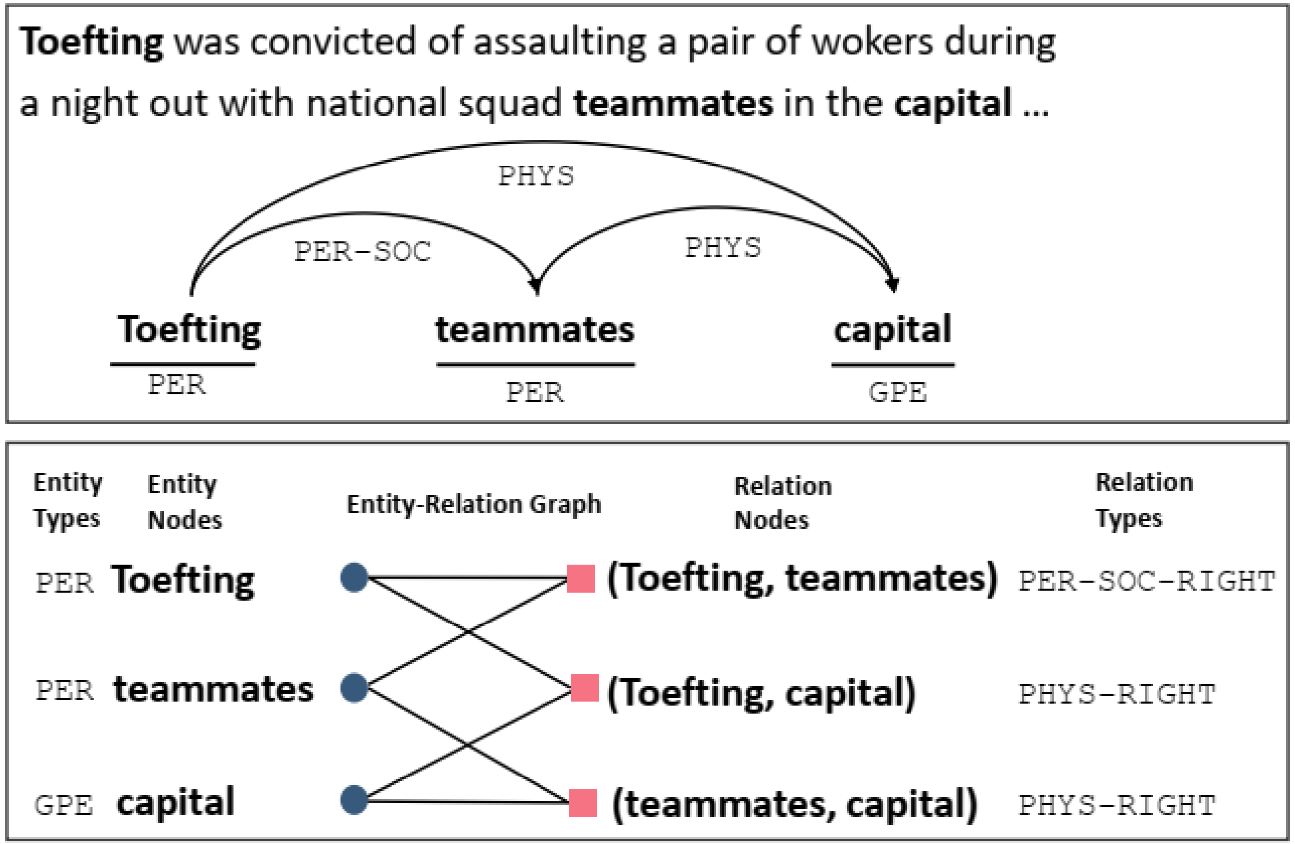
MultiDecoder	59
GraphRel	**62**

F1 on NYT datase

Joint Type Inference on Entities & Relations

- Performance of existing RE models on ACE05 dataset
- For many entities, their spans are correctly identified,
- but their entity types are wrong.
- the F1 of extracting typed entities is about 83% while the F1 of extracting entity spans is about 90%
- We need a better type inference model
- Can help with improved relation extraction

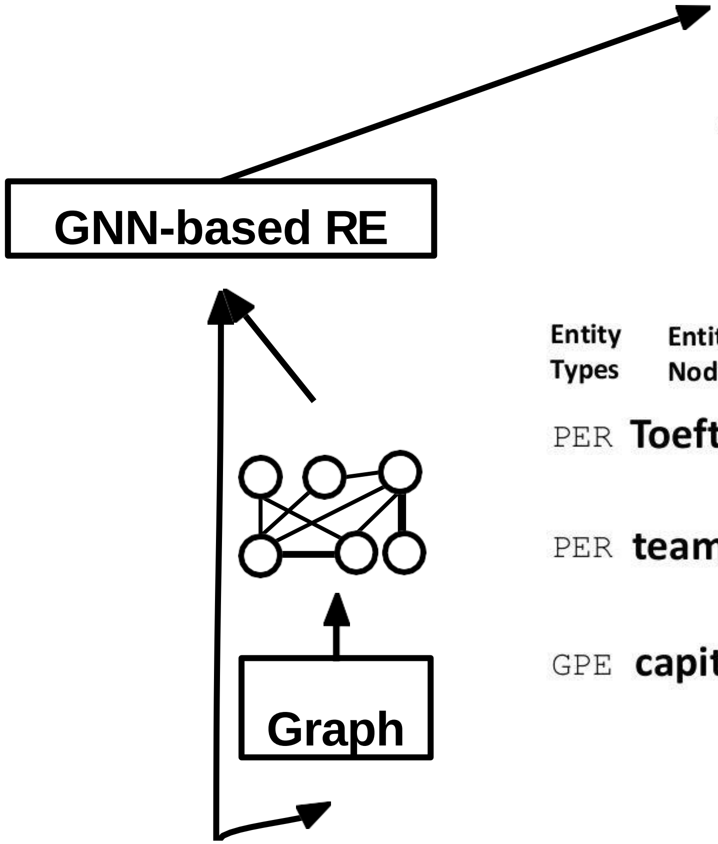
Joint Type Inference on Entities & Relations

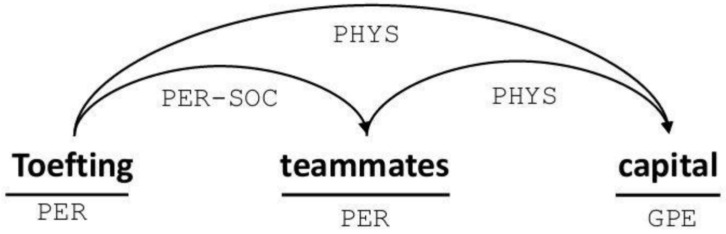


Joint Type Inference on Entities & Relations

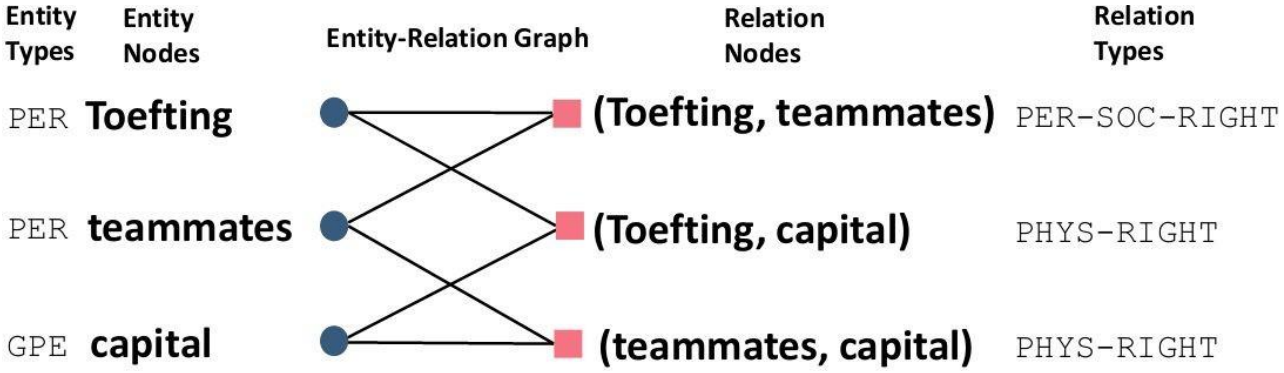
- define joint entity relation extraction into two sub-tasks:
  (1) entity span detection (2) entity relation type deduction
- entity span detection, treat it as a sequence labeling problem
- given all detected entity spans in a sentence, we define an entity-relation bipartite graph and apply GCN on the graph
- For each entity span, we assign an entity node. For each entity-entity pair, we assign a relation node.
- Edges connect relation nodes and their entity nodes
- Learn representations for entity nodes and relation nodes by recursively aggregating information from their neighborhood over the bipartite graph

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



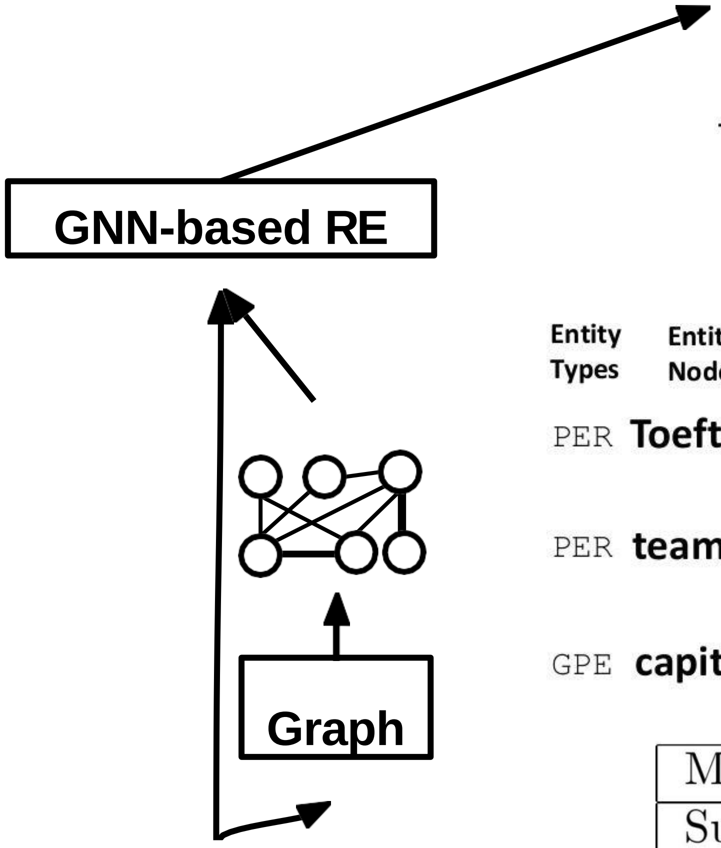


Idea: GCN on bipartite graph

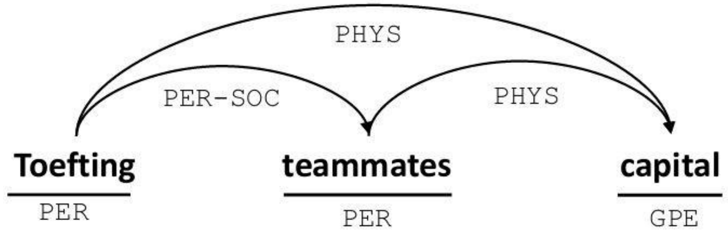


Toefting was convicted with teammates in the capital …

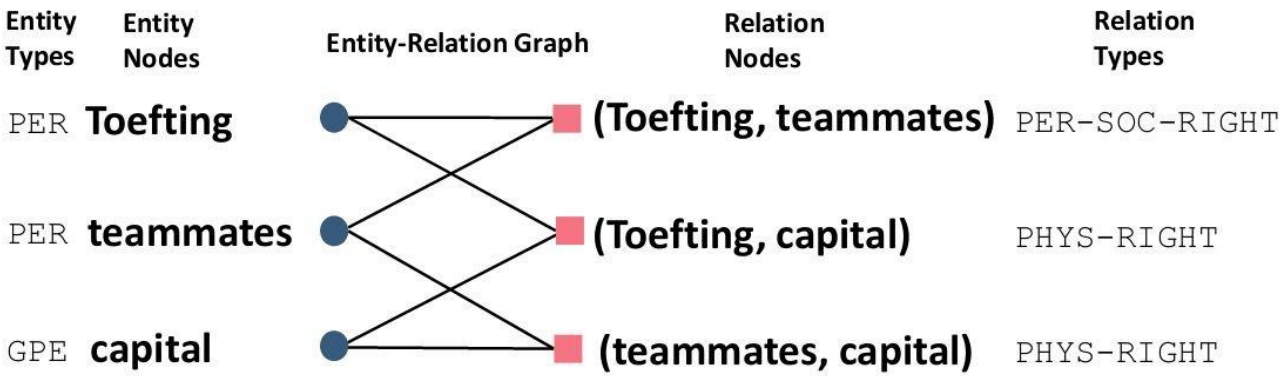
Joint Type Inference Sun et al., ACL'19]



Toefting was convicted with teammates in the capital …



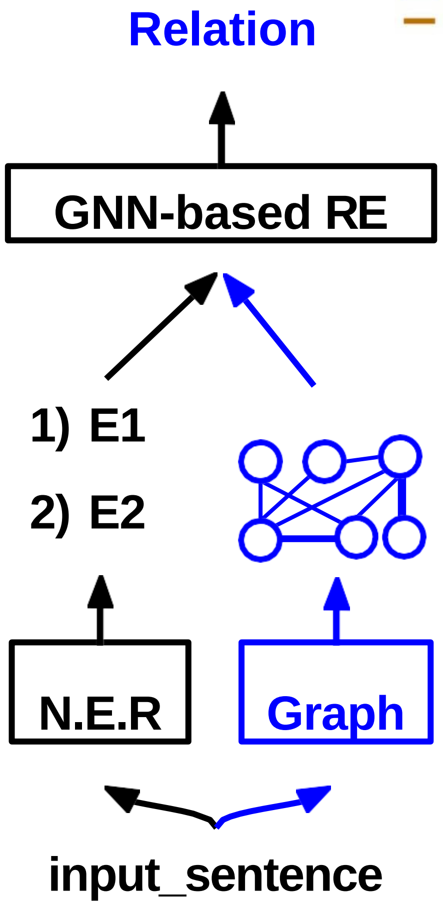
Idea: GCN on bipartite graph

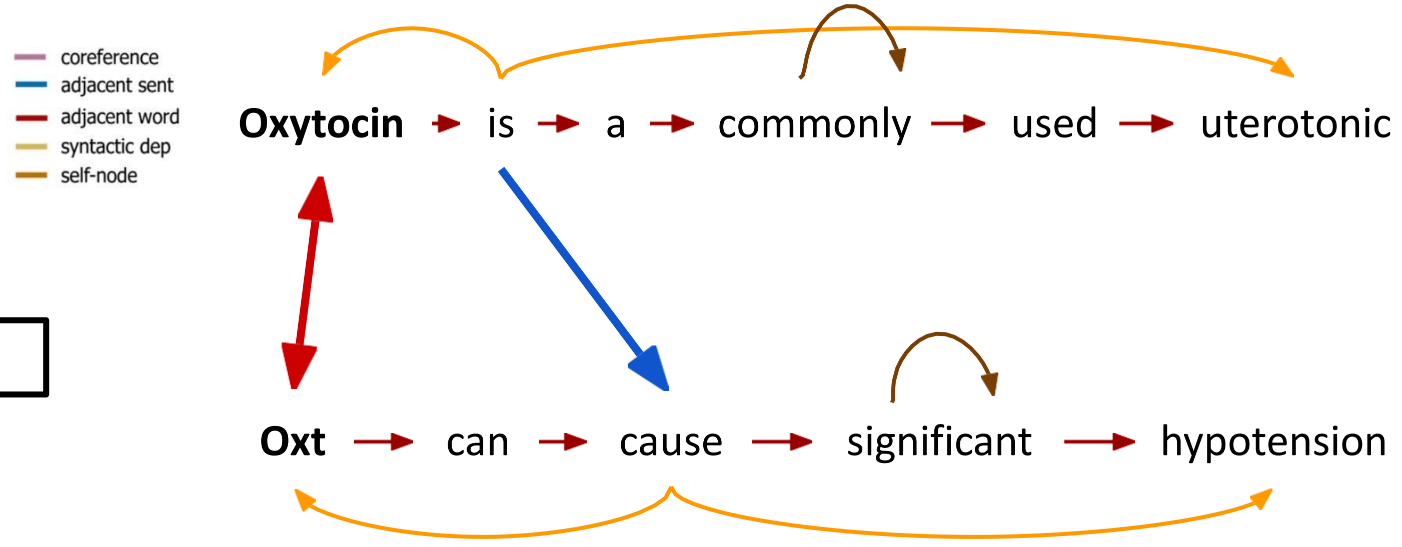


Method	Entity	Relation
Sun et al., EMNLP'18	83.4	57.8
GCN	**84.2**	**59.1**

F1 on ACE05 dataset

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



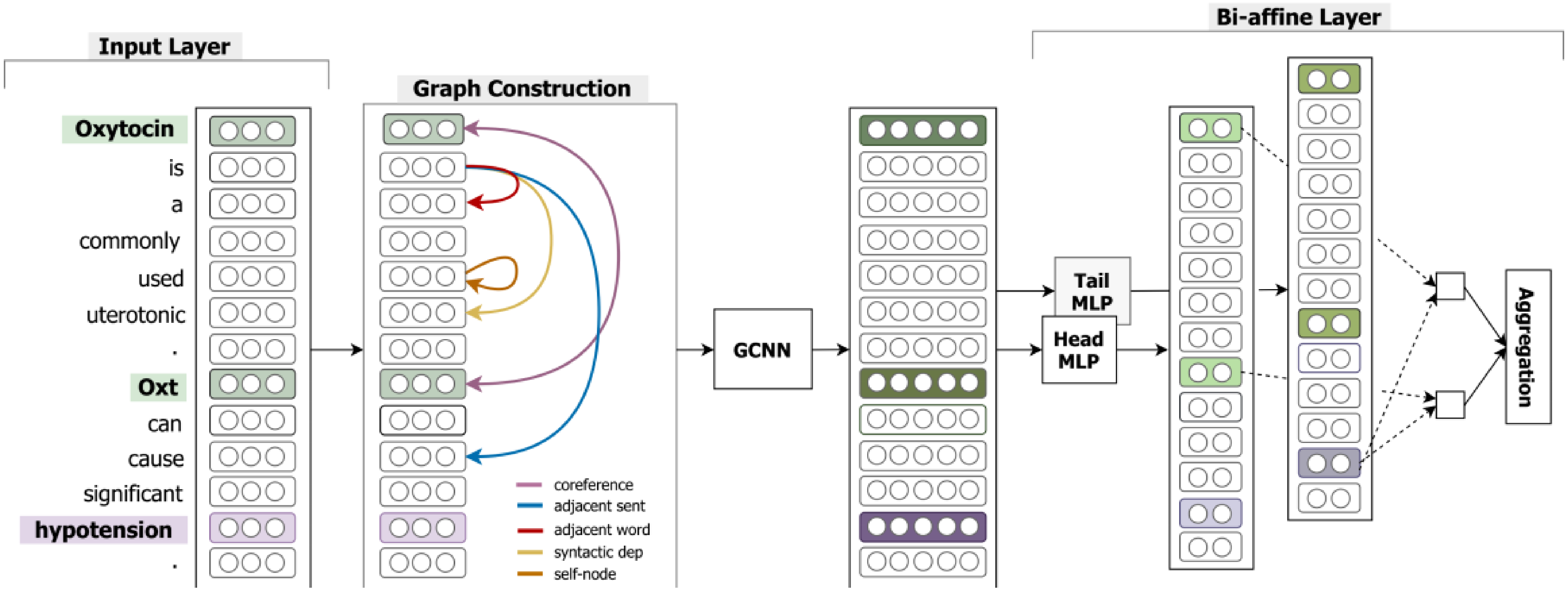


Oxytocin , ? , hypotension

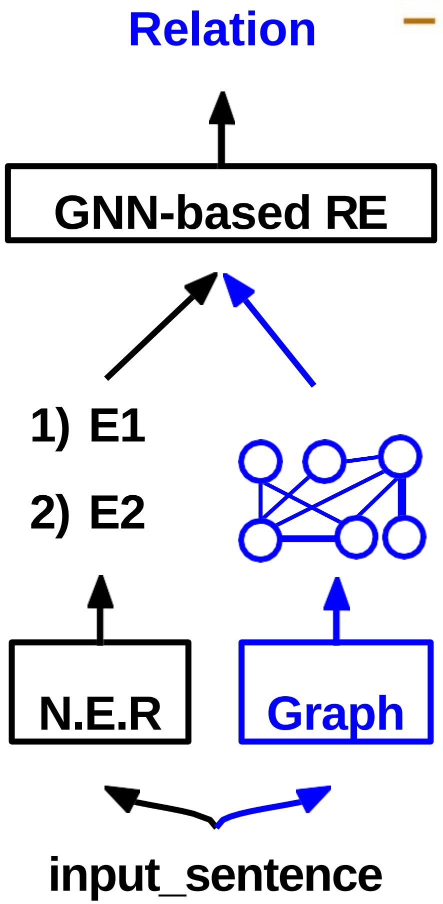
Inter-Sentence Relation Extraction

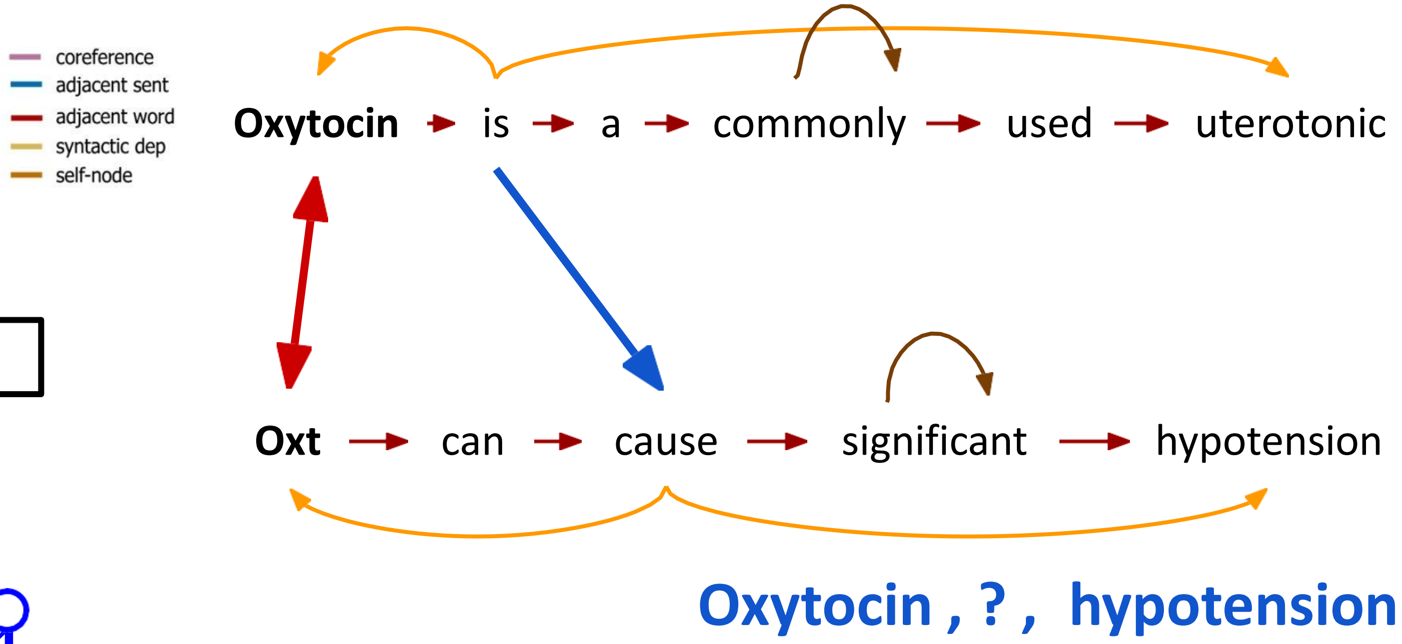
- RE model takes a triple (e1, e2, t) as input and returns a relation for the pair, where t is document containing words [w1, w2.. Wn]
- apply Multi-Instance Learning (MIL) on ‘t’ to combine all mention-level pairs and predict the final relation category of a target pair
- inter-sentence RE model builds a labelled edge Graph CNN (GCNN) model on a document-level graph.
- Graph nodes correspond to words and edges represent local and nonlocal dependencies among them.
- Encode the graph structure using a stacked GCNN layer
- infer relations between entities using MIL-based bi-affine pairwise scoring function on the entity node representations

Inter-Sentence RE



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



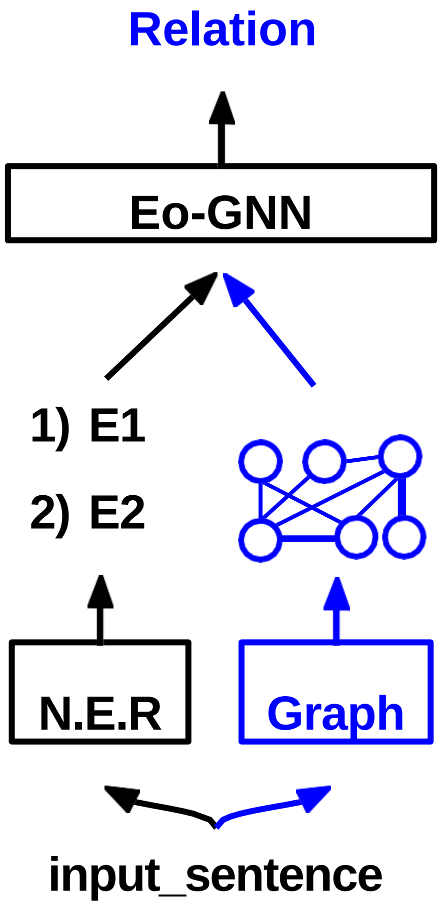


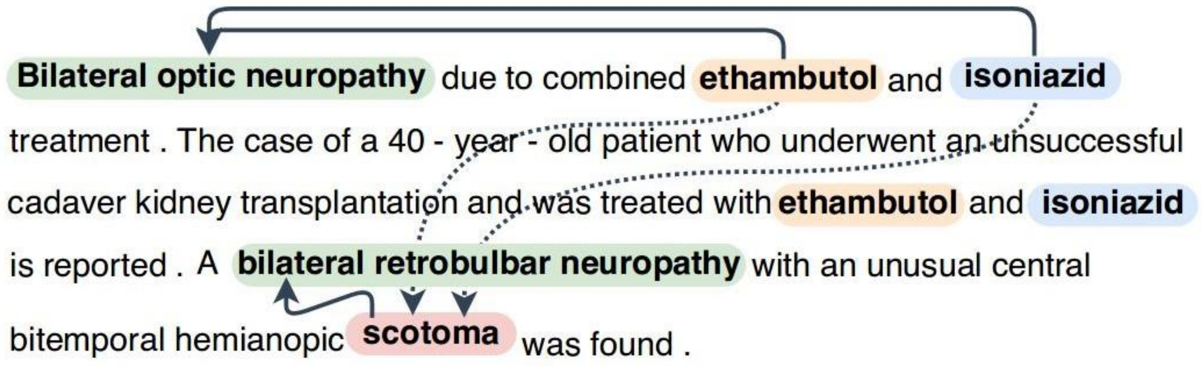
Bi-LSTM	84.1
Bi-LSTM + GCN	**87.5**

Coreference edges improve inter RE

BACKUP SLIDES

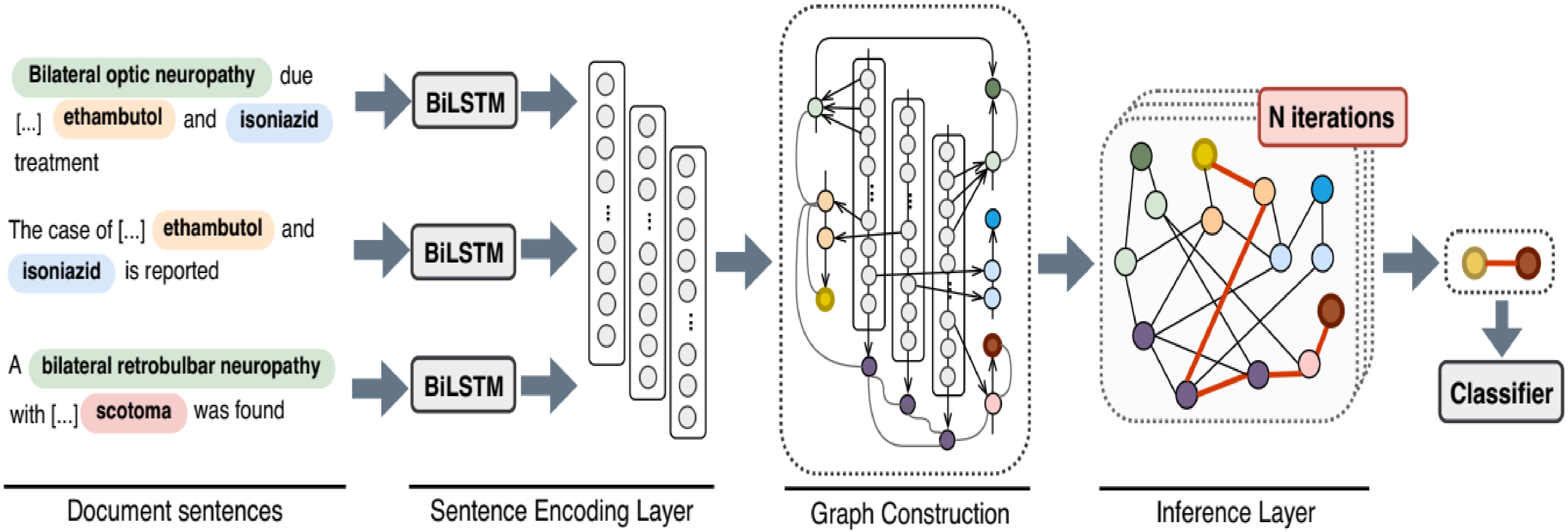
Edge Oriented Graphs [Christopoulou et al., ACL'18, EMNLP'19]





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

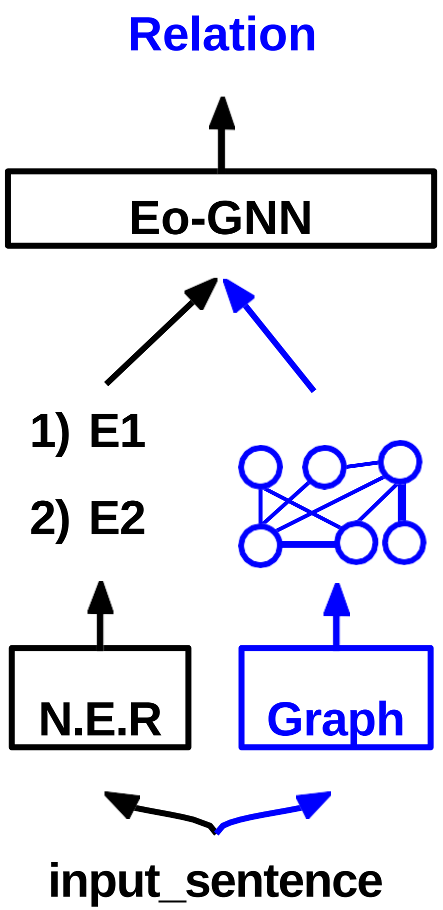
Document level RE



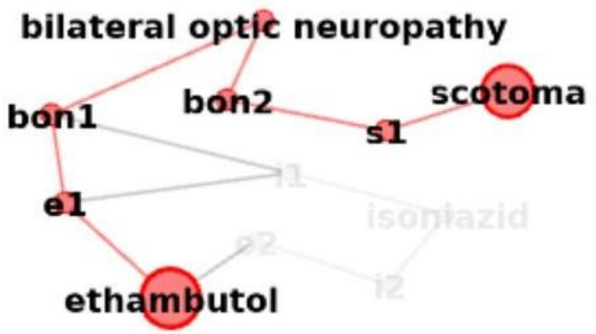
Document Level RE

- four layer model
    1. Document encoding
    2. Graph construction
    3. Inference layer
    4. Classifier
- The model receives as input a document with identified concept-level entities and their textual mentions.
- Next, a document-level graph with multiple types of nodes and edges is constructed.
- An inference algorithm is applied on the graph edges to generate concept-level pair representations.
- In the final layer, the edge representations between the target concept-entity nodes are classified into relation categories

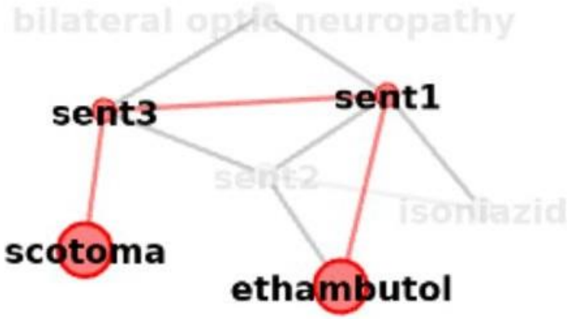
Edge oriented Graph Christopoulou et al., ACL'18, EMNLP'19



mention-mention



entity-sentence

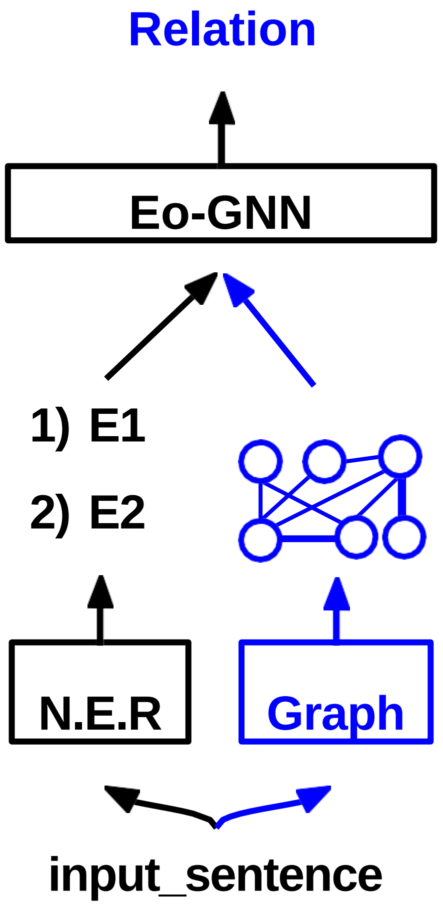


$$\mathbf{n}_m = [\mathrm{avg}_{w_i \in m}(\mathbf{w}_i); \mathbf{t}_m]$$

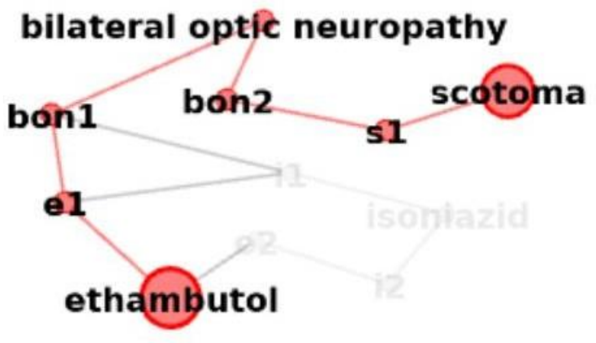
$$\mathbf{n}_e = [\mathrm{avg}_{m_i \in e}(\mathbf{m}_i); \mathbf{t}_e]$$

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

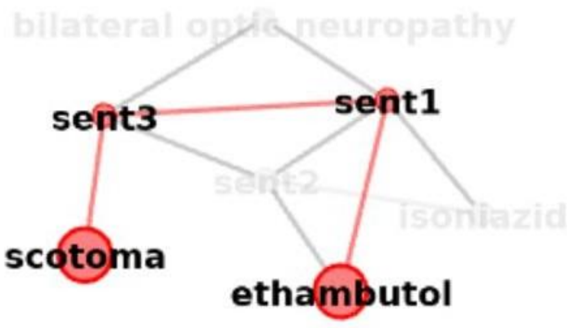
Christopoulou et al., ACL’18, EMNLP’19



mention-mention



entity-sentence



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

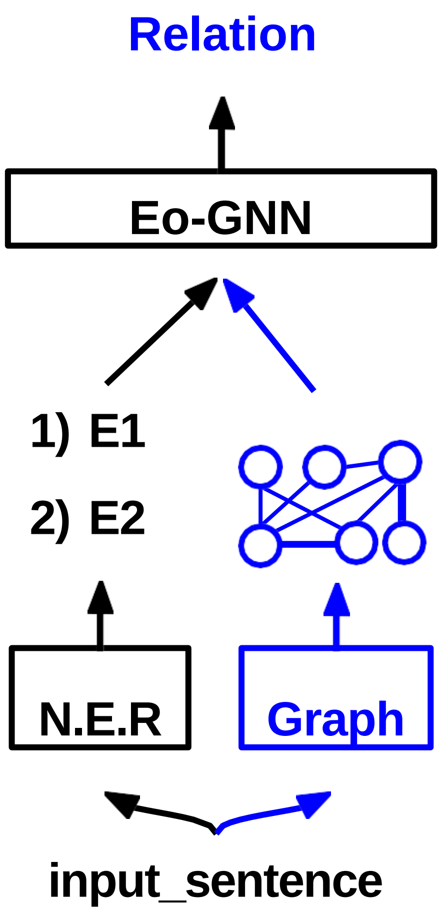
$$\mathbf{x}_{\mathrm{MS}} = [\mathbf{n}_m; \mathbf{n}_s]$$

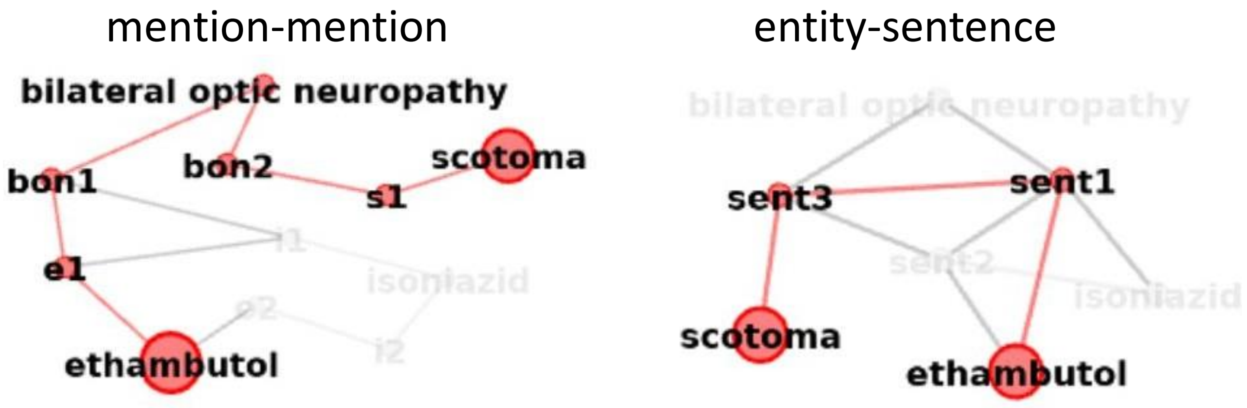
$$\mathbf{x}_{\mathrm{ME}} = [\mathbf{n}_m; \mathbf{n}_e]$$

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

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

Christopoulou et al., ACL’18, EMNLP’19





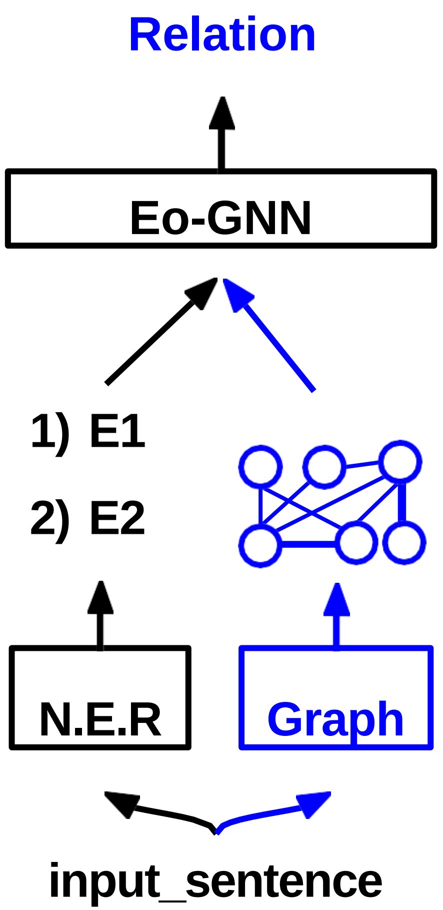
$$\mathbf{e}_z = \mathbf{W}_z\,\mathbf{x}_z \qquad z \in [\mathrm{MM}, \mathrm{MS}, \mathrm{ME}, \mathrm{SS}, \mathrm{ES}]$$

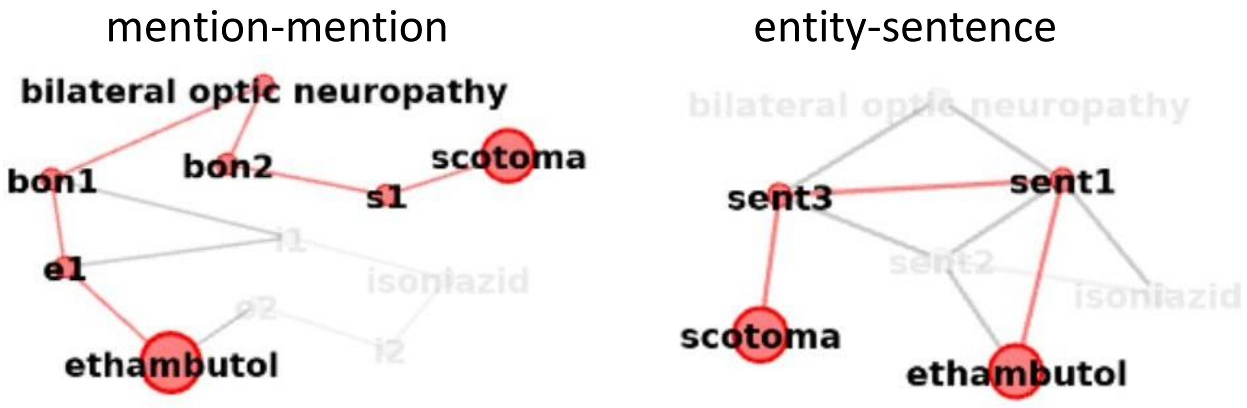
$$f\left(\mathbf{e}_{ik}^{(l)}, \mathbf{e}_{kj}^{(l)}\right) = \sigma\left(\mathbf{e}_{ik}^{(l)} \odot \left(\mathbf{W}\,\mathbf{e}_{kj}^{(l)}\right)\right)$$

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

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

Christopoulou et al., ACL’18, EMNLP’19





CDR dataset

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

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

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

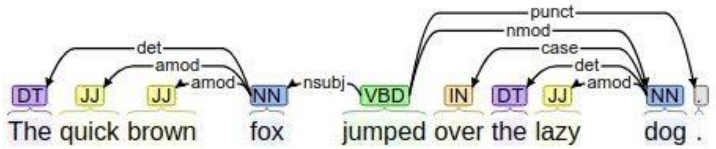
Ablation on GDA dataset

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

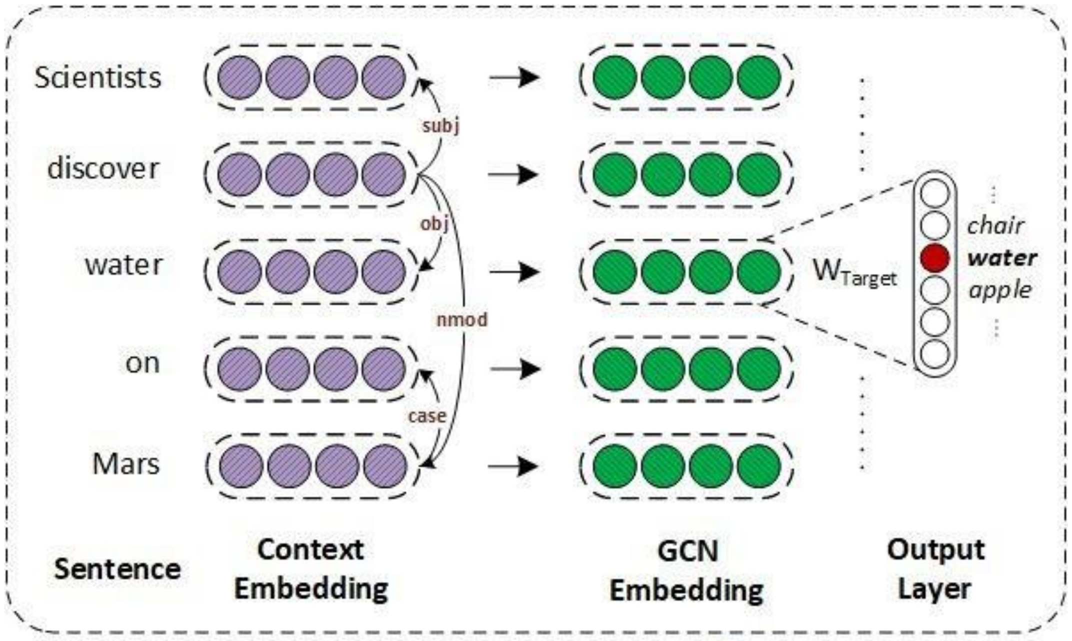
Syn GCN

Vashishth et al., ACL’19

- Given a sentence, obtain its syntax.



- Exploit syntax for predicting a word.



SynGCN (Sentence-level)

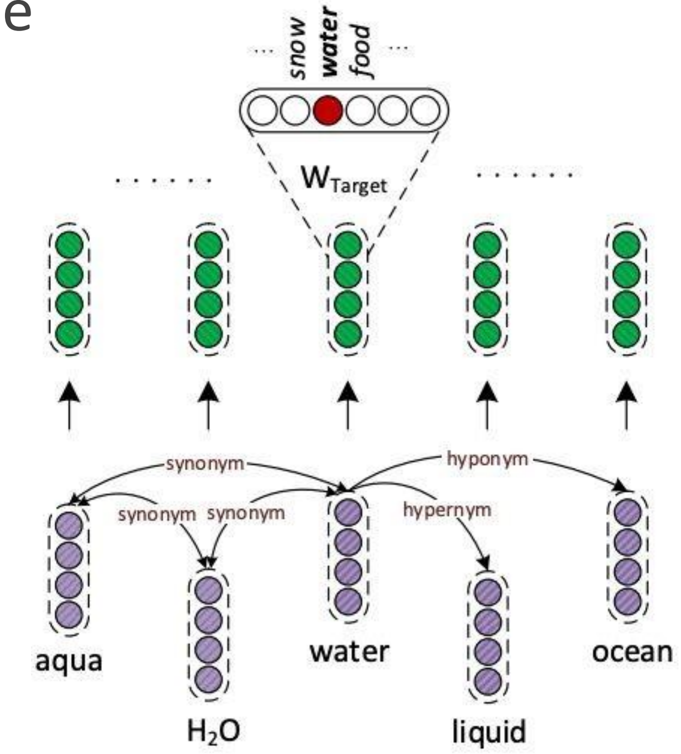
SynGCN

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

F1 Score

Sem GCN Vashishth et al., ACL'19

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



SemGCN (Corpus-level)

SemGCN

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

F1 Score

Syntax, Semantics help word embeddings