AILab

Attention Guided Graph Convolutional Networks for Relation Extraction

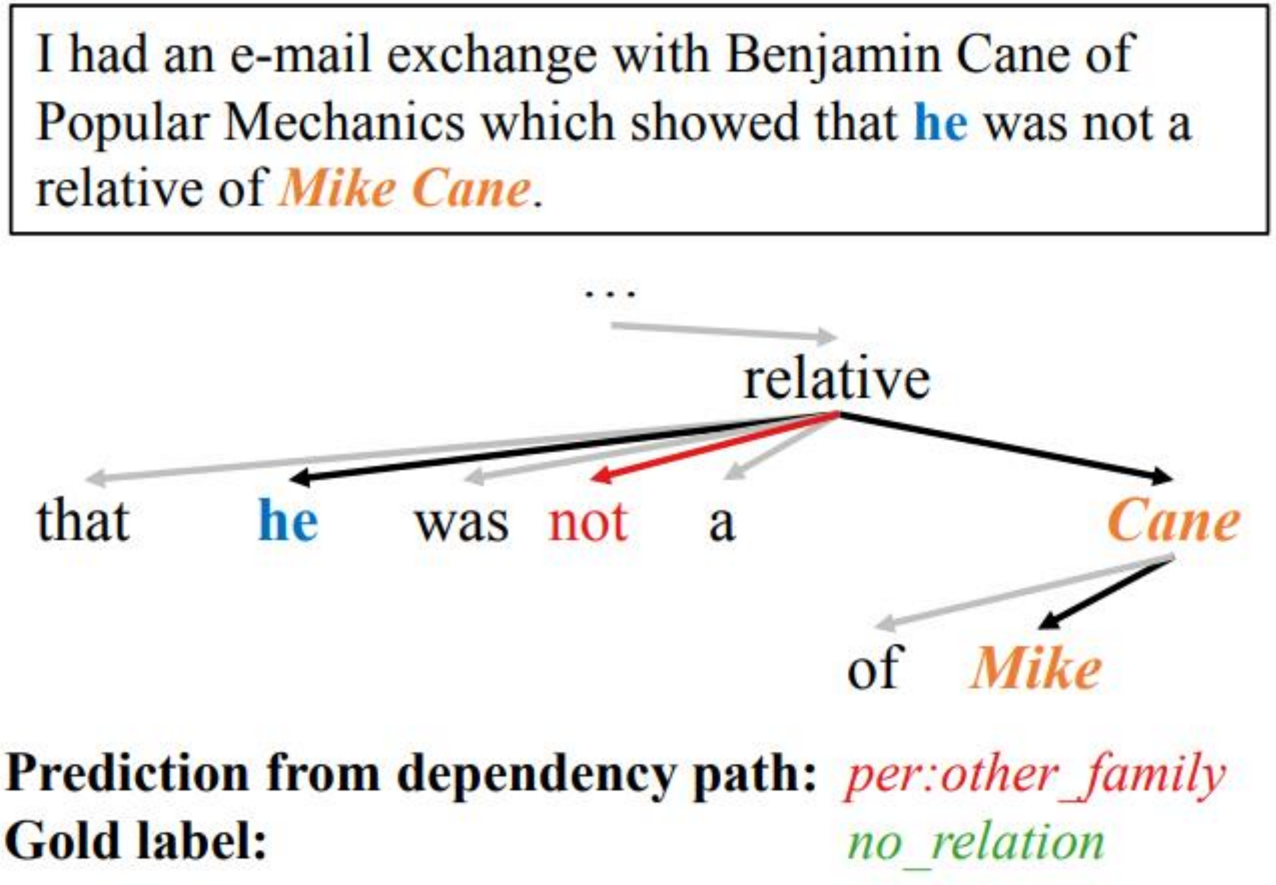
석사 3기 조충현

Introduction

Introduction

- Relation Extraction 개요
  - 관계 추출은 문장에서 엔티티 간의 관계를 추출하는 것을 목표로 하는 classification task
  - 이 Task는 다양한 분야의 Application(QA, KBP etc.)에서 중요한 역할을 한다.
  - 대부분의 존재하는 Relation Extraction model들은 두가지로 분류가능
    - Sequence-based : word sequence 만 이용함
    - Dependency-based : dependency tree를 이용함 (눈으로 보기에는 모호한 관계 포착 가능!)
  - Dependency 정보를 더 잘 사용하기 위해 다양한 Pruning 전략 제안
  - 더 나아가서 Dependency tree에 Graph Convolutional networks(GCNs) 적용
  - 그러나 rule-based로 된 pruning 전략은 Tree에서 중요한 정보를 제거할 수 있는 단점 존재
  - 이 논문은 그러한 문제를 해결하기 위한 방법 제시

Introduction



I had an e-mail exchange with Benjamin Cane of Popular Mechanics which showed that **he** was not a relative of ***Mike Cane***.

Prediction from dependency path: *per:other_family*
Gold label: *no_relation*

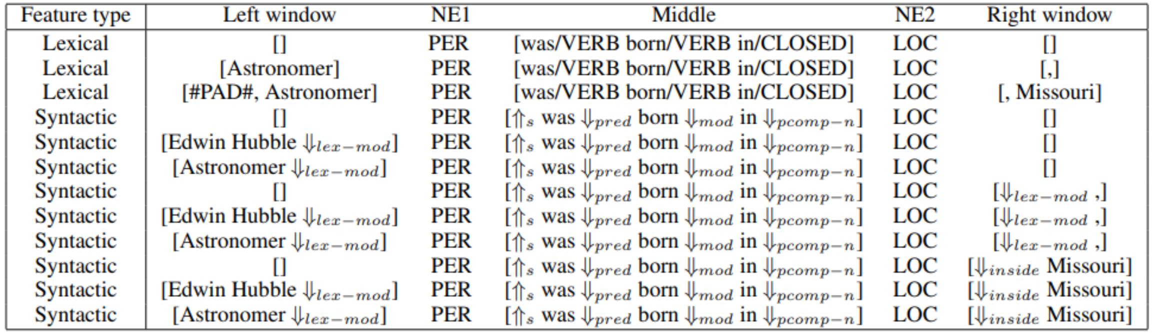
Related Work

Related Work

- Relation Extraction
  - 엔티티들 간에 관계를 찾는 것
  - 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

Related Work

- Relation Extraction
  - 초기에는 statistical method들을 기반으로 연구
    - Tree-based kernel (Zelenko et al., 2002)
    - Dependency path-based kernel (Bunescu and Mooney, 2005)
    - Syntactic features를 포함한 statistical classifier (Mintz et al. 2009)



Feature type	Left window	NE1	Middle	NE2	Right window
Lexical	[]	PER	[was/VERB born/VERB in/CLOSED]	LOC	[]
Lexical	[Astronomer]	PER	[was/VERB born/VERB in/CLOSED]	LOC	[,]
Lexical	[#PAD#, Astronomer]	PER	[was/VERB born/VERB in/CLOSED]	LOC	[, Missouri]
Syntactic	[]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[]
Syntactic	[Edwin Hubble $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[]
Syntactic	[Astronomer $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[]
Syntactic	[]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{lex-mod}$,]
Syntactic	[Edwin Hubble $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{lex-mod}$,]
Syntactic	[Astronomer $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{lex-mod}$,]
Syntactic	[]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{inside}$ Missouri]
Syntactic	[Edwin Hubble $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{inside}$ Missouri]
Syntactic	[Astronomer $\Downarrow_{lex-mod}$]	PER	[$\Uparrow_s$ was $\Downarrow_{pred}$ born $\Downarrow_{mod}$ in $\Downarrow_{pcomp-n}$]	LOC	[$\Downarrow_{inside}$ Missouri]

Table 3: Features for ‘Astronomer Edwin Hubble was born in Marshfield, Missouri’.

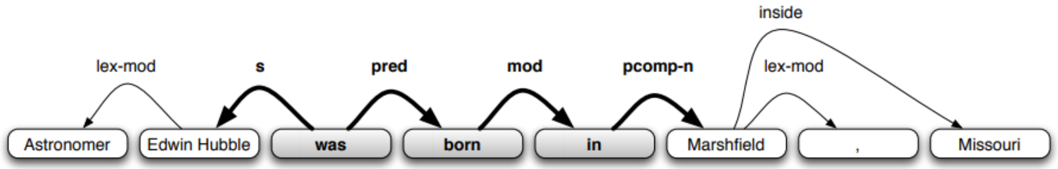


Figure 1: Dependency parse with dependency path from ‘Edwin Hubble’ to ‘Marshfield’ highlighted in boldface.

Related Work

- Relation Extraction
    - 최근에는 sequence-based models은 다른 neural networks (CNN [Wang et al., 2016], RNN [Zhou et al., 2016; Zhang et al., 2017], CNN+RNN [Vu et al., 2016], Transformer [Verga et al., 2018])들을 활용한 모델이 등장
    - Dependency-based 방식은 structural information을 neural model에 포함
        - Graph LSTM (Peng et al. 2017)

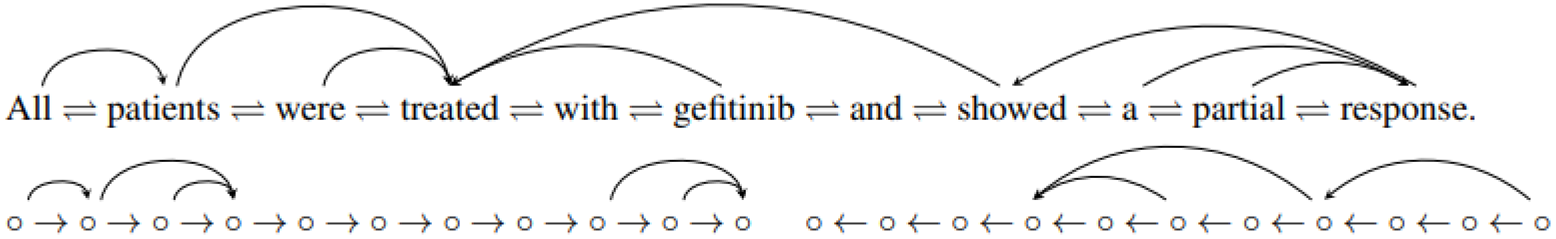
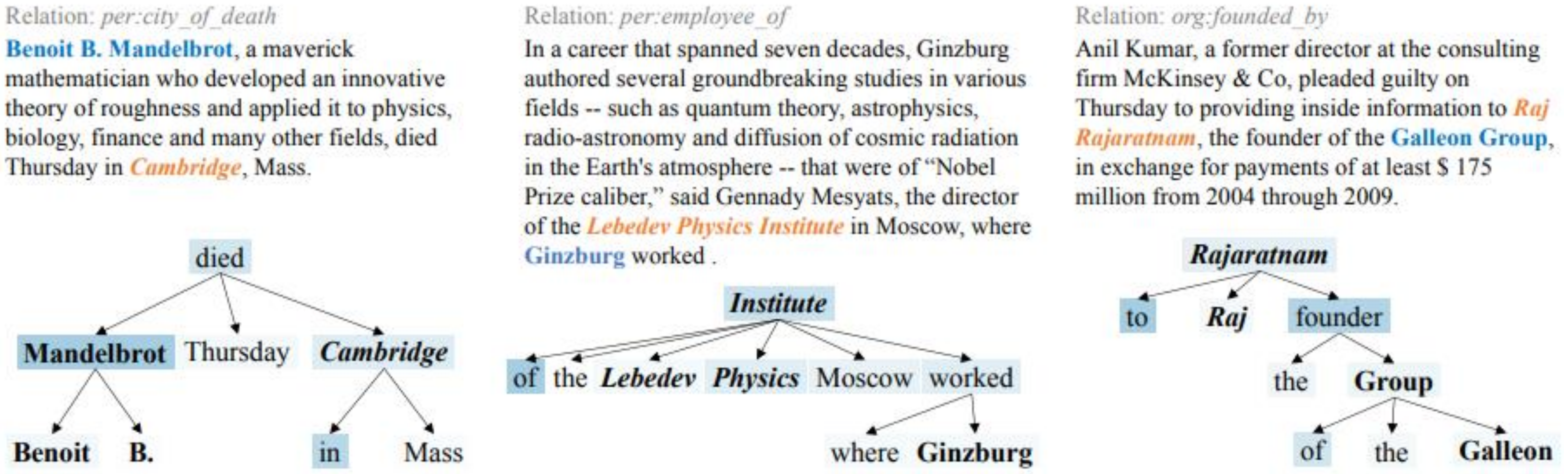

All ⇌ patients ⇌ were ⇌ treated ⇌ with ⇌ gefitinib ⇌ and ⇌ showed ⇌ a ⇌ partial ⇌ response.

Figure 3: The graph LSTMs used in this paper. The document graph (top) is partitioned into two directed acyclic graphs (bottom); the graph LSTMs is constructed by a forward pass (Left to Right) followed by a backward pass (Right to Left). Note that information goes from dependency child to parent.

Related Work

- Relation Extraction
  - 다양한 pruning 전략으로 dependency information의 품질을 올려 성능을 높임
  - [Xu et al. 2015]은 shortest dependency path를 encode하여 neural model에 적용
  - [Miwa and Bansal 2016]은 두 엔티티의 LCA subtree를 LSTM에 적용
  - [Zhang et al. 2018] path-centric pruning 전략을 적용



Relation: *per:city_of_death*

Benoit B. Mandelbrot, a maverick mathematician who developed an innovative theory of roughness and applied it to physics, biology, finance and many other fields, died Thursday in ***Cambridge***, Mass.

Relation: *per:employee_of*

In a career that spanned seven decades, Ginzburg authored several groundbreaking studies in various fields -- such as quantum theory, astrophysics, radio-astronomy and diffusion of cosmic radiation in the Earth's atmosphere -- that were of "Nobel Prize caliber," said Gennady Mesyats, the director of the ***Lebedev Physics Institute*** in Moscow, where **Ginzburg** worked .

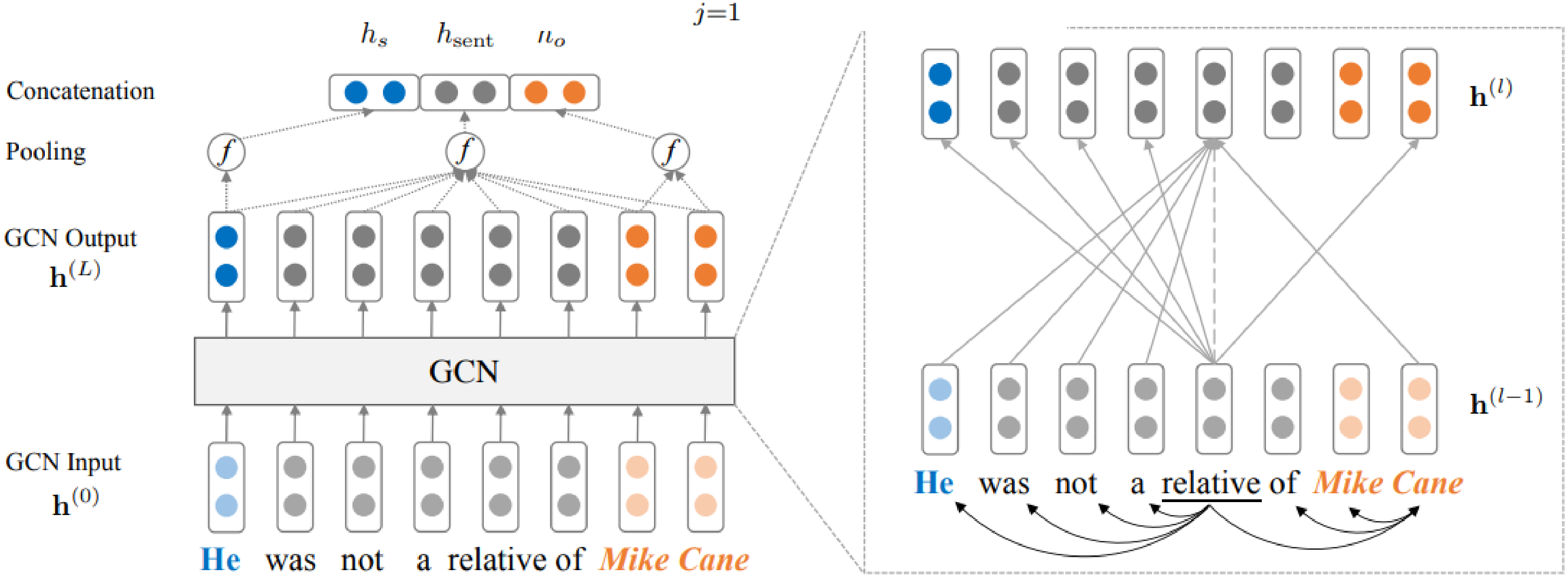
Relation: *org:founded_by*

Anil Kumar, a former director at the consulting firm McKinsey & Co, pleaded guilty on Thursday to providing inside information to ***Raj Rajaratnam***, the founder of the **Galleon Group**, in exchange for payments of at least $ 175 million from 2004 through 2009.

Figure 5: Examples and the pruned dependency trees where the C-GCN predicted correctly. Words are shaded by the number of dimensions they contributed to h_{sent} in the pooling operation, with punctuation omitted.

Related Work

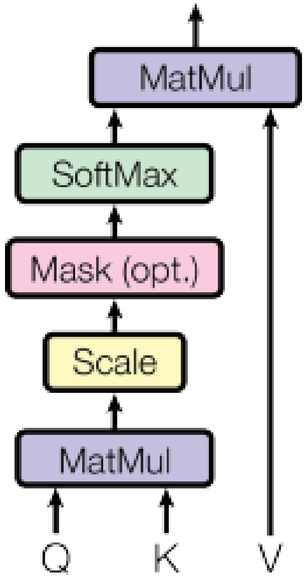
- Graph Convolutional Networks
  - 초기에는 neural network에 구조화된 graph를 적용 (Gori et al., 2005; Bruna 2014)
  - 부수적으로 local spectral convolution technique로 계산 효율성을 높임 (Henaff et al., 2015; Defferrard et al., 2016)
  - 우리 approach는 GCNs과 비슷함 $h_i^{(l)} = \sigma\left(\sum_{j=1}^{n} A_{ij} W^{(l)} h_j^{(l-1)} + b^{(l)}\right)$



Related Work

- Graph Convolutional Networks
  - 최근 이웃 node들을 masked self-attentional layers를 통해 summarize하는 graph attention network (GATs)를 제안 (Velickovic et al., 2018)
  - 이 논문은 이 방식을 채택하여 모든 노드들 사이의 관계성을 측정

Scaled Dot-Product Attention



$$\mathrm{Attention}(Q,K,V)=\mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Method

Method

- GCNs
  - N개의 노드를 가지는 Graph, $n * n$ 인접행렬 A
  - i 번째 노드에 대한 l번째 layer에 대한 convolution computation의 input은 $h^{(l-1)}$이고 output은 $h_i^{(l)}$

$$\mathbf{h}_i^{(l)} = \rho\Big(\sum_{j=1}^{n} \mathbf{A}_{ij}\mathbf{W}^{(l)}\mathbf{h}_j^{(l-1)} + \mathbf{b}^{(l)}\Big)$$

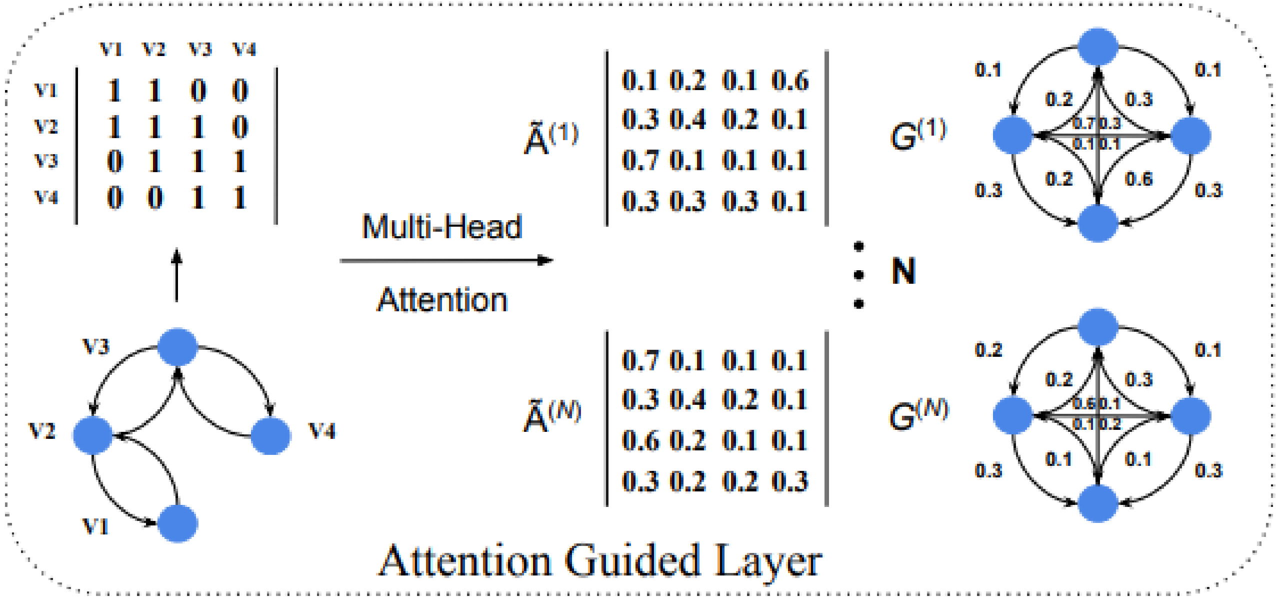
  - $W^{(l)}$은 weight matrix, $b^{(l)}$은 bias vector, ρ는 activation function (e.g., RELU), $h_i^{(0)}$은 initial input x_i

Method

- Attention Guided Layer
  - 대부분의 존재하는 pruning 전략은 predefined 된 것, 인접행렬에 의해서 full tree에서 subtree로 가지치기 함
  - 이러한 전략은 hard attention (1 아니면 0) 형태로 볼 수 있다.
    - 그렇기 때문에 관계가 있는 정보를 제거할 수 가 있다!!
  - 이 논문은 attention guided layer를 통해 soft pruning 전략을 개발
  - 모든 edge에 weight를 할당하고, 이 weight는 모델에 의하여 end-to-end로 학습된다.

Method

- Attention Guided Layer
  - 먼저 기존의 dependency tree를 attention guided 인접행렬 Ã $(n * n)$에 의해 fully connected edge-weighted graph로 변환
  - Ã는 self-attention에 의해 구성되어진 것으로, 두 단어의 상호작용을 capture한다.



$$\begin{array}{c|cccc|} & \text{V1} & \text{V2} & \text{V3} & \text{V4} \\ \text{V1} & 1 & 1 & 0 & 0 \\ \text{V2} & 1 & 1 & 1 & 0 \\ \text{V3} & 0 & 1 & 1 & 1 \\ \text{V4} & 0 & 0 & 1 & 1 \end{array}$$

$$\tilde{A}^{(1)} \left|\begin{array}{cccc} 0.1 & 0.2 & 0.1 & 0.6 \\ 0.3 & 0.4 & 0.2 & 0.1 \\ 0.7 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.3 & 0.3 & 0.1 \end{array}\right|$$

$$\tilde{A}^{(N)} \left|\begin{array}{cccc} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.4 & 0.2 & 0.1 \\ 0.6 & 0.2 & 0.1 & 0.1 \\ 0.3 & 0.2 & 0.2 & 0.3 \end{array}\right|$$

Attention Guided Layer

Method

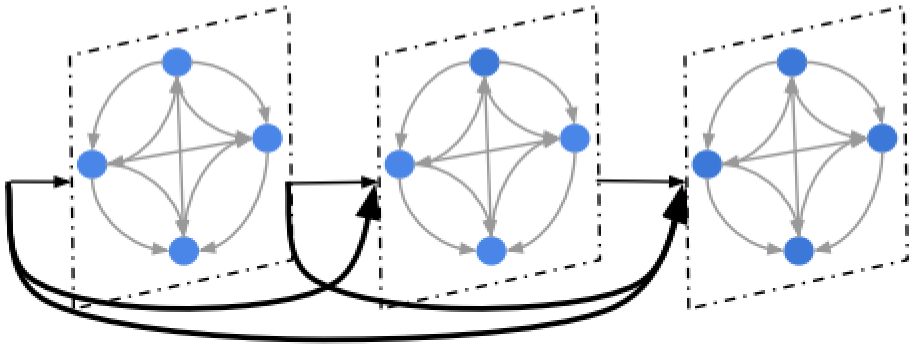
- Attention Guided Layer
  - Key idea는 노드 사이(특히,indirect로 연결된, multi-hop path)의 relation을 얻는 것
  - Ã를 multi-head attention을 사용하여 다양한 표현의 attend 정보를 이용한다.

$$\tilde{\mathbf{A}}^{(\mathbf{t})} = softmax(\frac{Q\mathbf{W}_i^Q \times (K\mathbf{W}_i^K)^T}{\sqrt{d}})V$$

  - Q와 K는 $l-1$ layer에서의 $h^{(l-1)}$로 동일, W_i^Q와 W_i^K는 parameter matrices, t는 t번째 head라는 것을 의미

Method

- Densely Connected Layer
  - 이전의 pruning 전략과는 달리 이 논문의 전략은 기존 구조보다 더 큰 fully connected graph 생성
  - 그래서 큰 그래프에서 더욱 잘 구조적 정보를 알아내기 위해 dense connection을 소개
  - 이 dense connection을 통하여 더 깊은 model을 학습할 수 있으므로 더 풍부한 local and non-local information을 가져서 더 나은 graph representation을 capture 할 수 있음



Densely Connected Layer (*number of sub-layers is 3*)

$$\mathbf{g}_j^{(l)} = [\mathbf{x}_j; \mathbf{h}_j^{(1)}; \ldots; \mathbf{h}_j^{(l-1)}]$$

$$\mathbf{h}_{t_i}^{(l)} = \rho\Big(\sum_{j=1}^{n} \tilde{\mathbf{A}}_{ij}^{(t)} \mathbf{W}_t^{(l)} \mathbf{g}_j^{(l)} + \mathbf{b}_t^{(l)}\Big)$$

Method

- Linear Combination Layer
  - N개의 다른 densely connected layer를 통합하는 linear combination

$$\mathbf{h}_{comb} = \mathbf{W}_{comb}\mathbf{h}_{out} + \mathbf{b}_{comb}$$

  - h_{out}은 N개의 서로 다른 densely connected layer의 output들을 concatenating한 output

$$\mathbf{h}_{out} = [\mathbf{h}^{(1)}; ...; \mathbf{h}^{(N)}]$$

  - W_{comb}는 weight matrix, b_{comb}는 bias vector

Method

- AGGCNs for Relation Extraction
  - [Zhang et al., 2018]에서와 같이 sentence representation과 entity representation들을 concatenate 해서 최종 representation을 얻는다.

$$h_{sent} = f(\mathbf{h_{mask}}) = f(\mathrm{AGGCN}(\mathbf{x}))$$

  - h_{mask}는 entity token을 제외한 모든 token들의 representatio이다.
  - f는 max pooling 함수
  - 똑같이 entity representation을 얻는다. $h_{e_i} = f(\mathbf{h_{e_i}})$

$$h_{final} = \mathrm{FFNN}([h_{sent}; h_{e_1}; ...h_{e_i}])$$

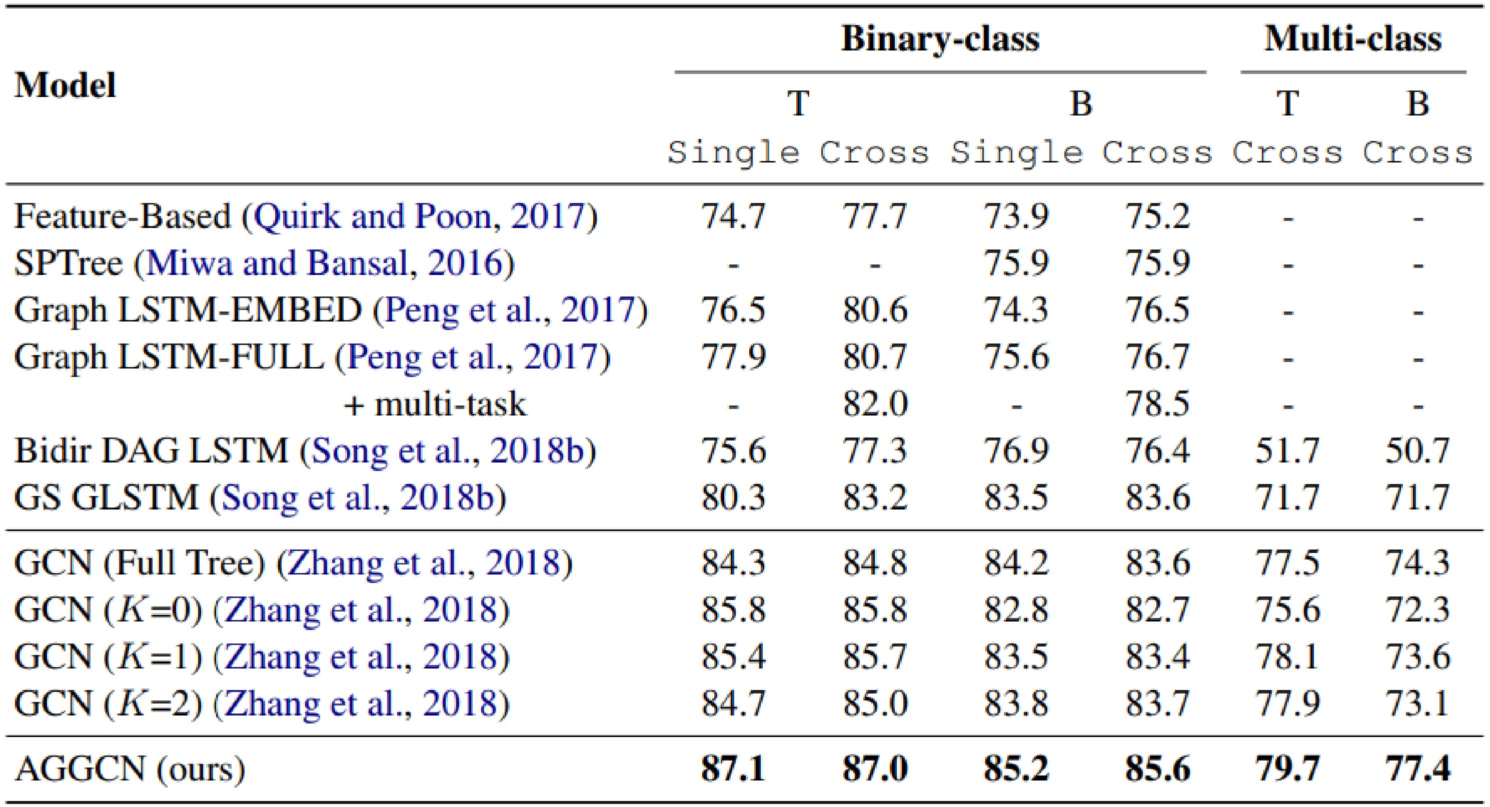
  - h_{final}은 logistic regression classifier의 input으로 들어가 prediction을 함

Experiments

Experiments

- Data
  - Cross-sentence n-ary relation extraction
    - [Peng et al., 2017]에서 소개된 dataset 사용
    - PubMed로 부터 추출한 6,987개의 ternary relation instance와 6,087개의 binary relation instance로 구성
    - 대부분의 instance는 multiple sentence 로 구성
  - Sentence-level relation extraction
    - TACRED dataset과 Semeval-10 Task 8 사용
    - TACRED는 106K의 instance 와 41개의 relation type으로 구성 (special type : 'no relation')
    - Semeval-10 Task 8은 10,717 instance 와 9개의 relation으로 구성 (special type : 'other')

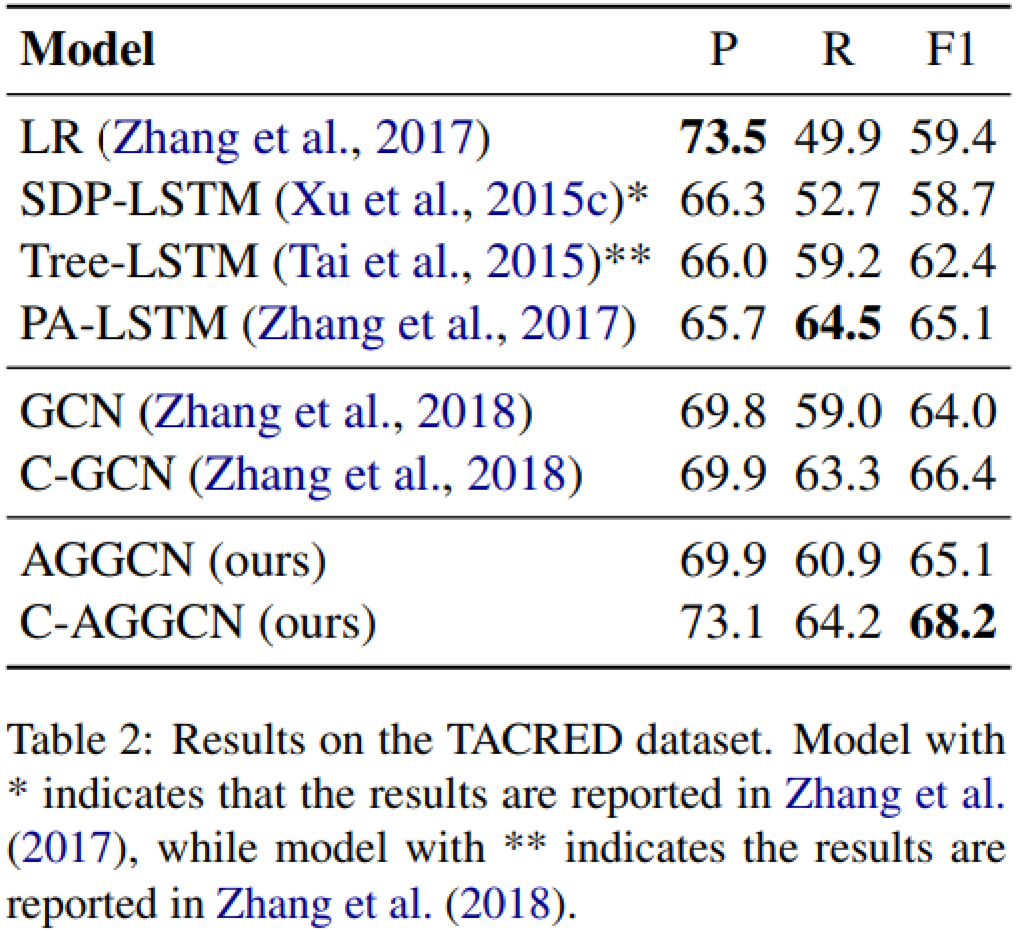
Experiments



Model	Binary-class				Multi-class	
	T		B		T	B
	Single	Cross	Single	Cross	Cross	Cross
Feature-Based (Quirk and Poon, 2017)	74.7	77.7	73.9	75.2	-	-
SPTree (Miwa and Bansal, 2016)	-	-	75.9	75.9	-	-
Graph LSTM-EMBED (Peng et al., 2017)	76.5	80.6	74.3	76.5	-	-
Graph LSTM-FULL (Peng et al., 2017)	77.9	80.7	75.6	76.7	-	-
+ multi-task	-	82.0	-	78.5	-	-
Bidir DAG LSTM (Song et al., 2018b)	75.6	77.3	76.9	76.4	51.7	50.7
GS GLSTM (Song et al., 2018b)	80.3	83.2	83.5	83.6	71.7	71.7
GCN (Full Tree) (Zhang et al., 2018)	84.3	84.8	84.2	83.6	77.5	74.3
GCN (K=0) (Zhang et al., 2018)	85.8	85.8	82.8	82.7	75.6	72.3
GCN (K=1) (Zhang et al., 2018)	85.4	85.7	83.5	83.4	78.1	73.6
GCN (K=2) (Zhang et al., 2018)	84.7	85.0	83.8	83.7	77.9	73.1
AGGCN (ours)	**87.1**	**87.0**	**85.2**	**85.6**	**79.7**	**77.4**

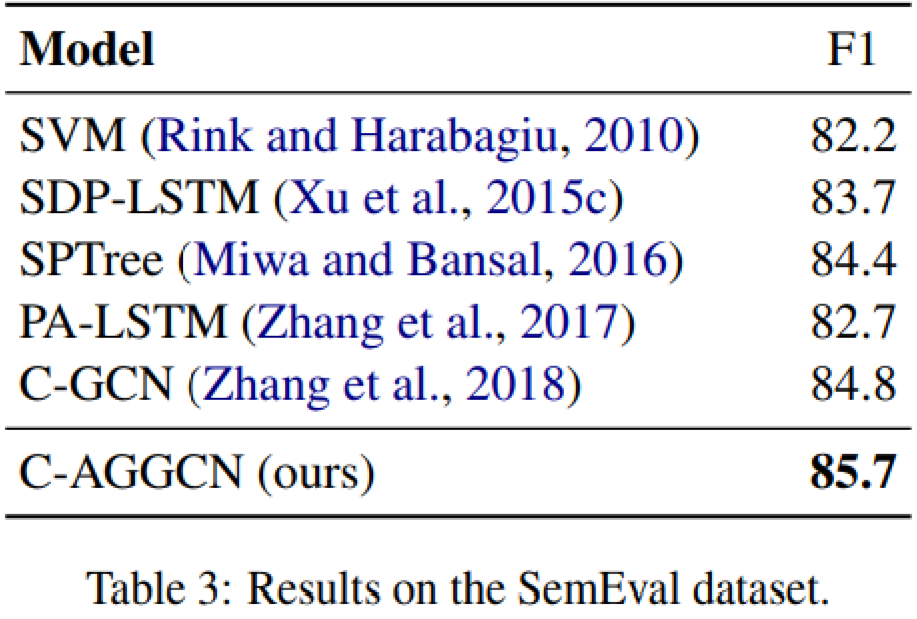
Table 1: Average test accuracies in five-fold validation for binary-class n-ary relation extraction and multi-class n-ary relation extraction. "T" and "B" denote ternary drug-gene-mutation interactions and binary drug-mutation interactions, respectively. Single means that we report the accuracy on instances within single sentences, while Cross means the accuracy on all instances. K in the GCN models means that the preprocessed pruned trees include tokens up to distance K away from the dependency path in the LCA subtree.

Experiments



Model	P	R	F1
LR (Zhang et al., 2017)	**73.5**	49.9	59.4
SDP-LSTM (Xu et al., 2015c)*	66.3	52.7	58.7
Tree-LSTM (Tai et al., 2015)**	66.0	59.2	62.4
PA-LSTM (Zhang et al., 2017)	65.7	**64.5**	65.1
GCN (Zhang et al., 2018)	69.8	59.0	64.0
C-GCN (Zhang et al., 2018)	69.9	63.3	66.4
AGGCN (ours)	69.9	60.9	65.1
C-AGGCN (ours)	73.1	64.2	**68.2**

Table 2: Results on the TACRED dataset. Model with * indicates that the results are reported in Zhang et al. (2017), while model with ** indicates the results are reported in Zhang et al. (2018).

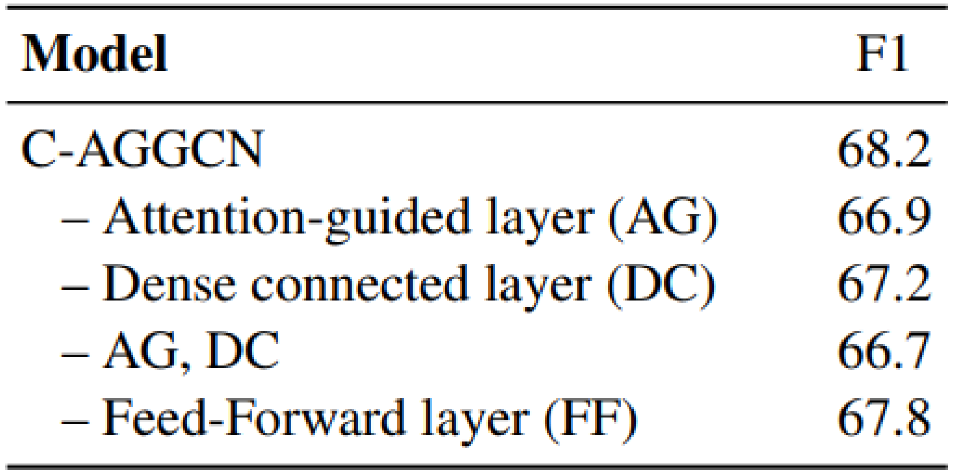


Model	F1
SVM (Rink and Harabagiu, 2010)	82.2
SDP-LSTM (Xu et al., 2015c)	83.7
SPTree (Miwa and Bansal, 2016)	84.4
PA-LSTM (Zhang et al., 2017)	82.7
C-GCN (Zhang et al., 2018)	84.8
C-AGGCN (ours)	**85.7**

Table 3: Results on the SemEval dataset.

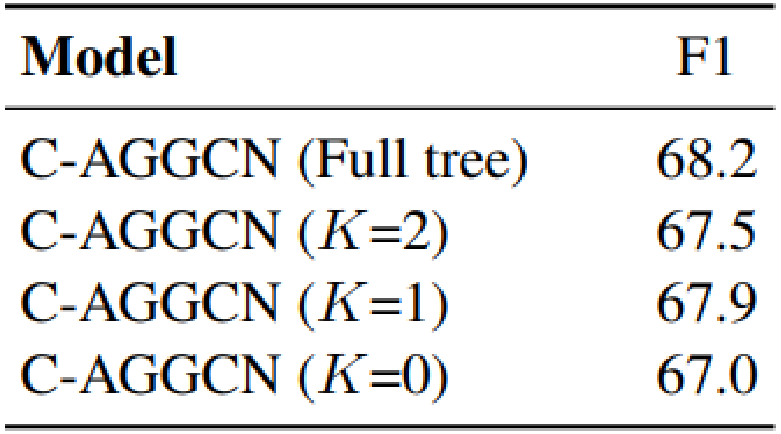
Analysis

Analysis



Model	F1
C-AGGCN	68.2
– Attention-guided layer (AG)	66.9
– Dense connected layer (DC)	67.2
– AG, DC	66.7
– Feed-Forward layer (FF)	67.8

Table 4: An ablation study for C-AGGCN model.



Model	F1
C-AGGCN (Full tree)	68.2
C-AGGCN (K=2)	67.5
C-AGGCN (K=1)	67.9
C-AGGCN (K=0)	67.0

Table 5: Results of C-AGGCN with pruned trees.

Analysis

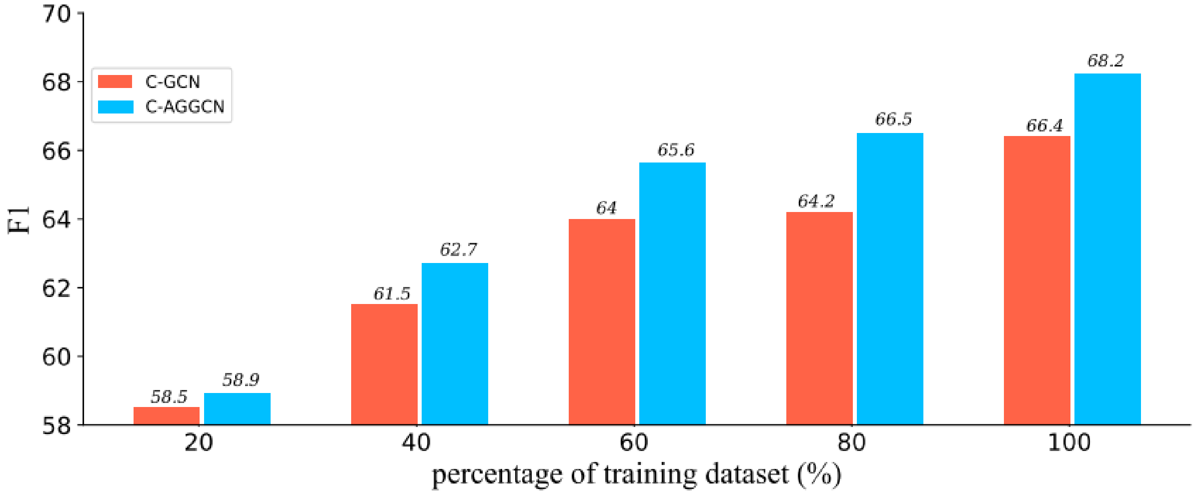


Figure 3: Comparison of C-AGGCN and C-GCN against different training data sizes. The results of C-GCN are reproduced from (Zhang et al., 2018).

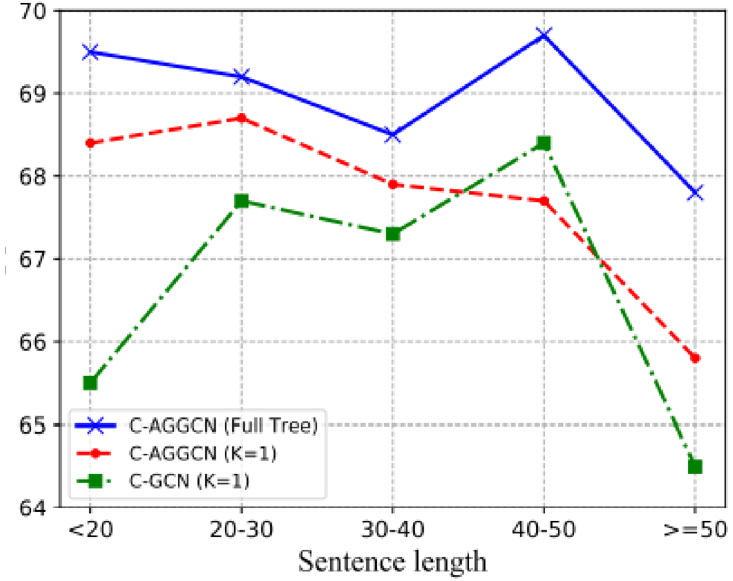


Figure 4: Comparison of C-AGGCN and C-GCN against different sentence lengths. The results of C-GCN are reproduced from (Zhang et al., 2018).

Conclusion

Conclusion

- 새로운 Attention Guided Graph Convolutional Networks (AGGCNs) 소개
- 실험에서 보이듯이 AGGCNs은 SOTA의 성능을 보임
- 이전의 접근법과는 달리, AGGCNs은 full tree를 직접적으로 사용하고 soft prunin을 이용하여 자동적으로 graph representation의 정보를 추출