

ING LAB Seminar

Story Ending Generation with Multi-Level Graph Convolutional Networks over Dependency Trees

목차



- 사전 지식
- Contribution
- 모델 구조
- Experiments

사전 지식

- Story Ending Generation (SEG)
- Dependency Tree
- Graph Convolutional Network (GCN)

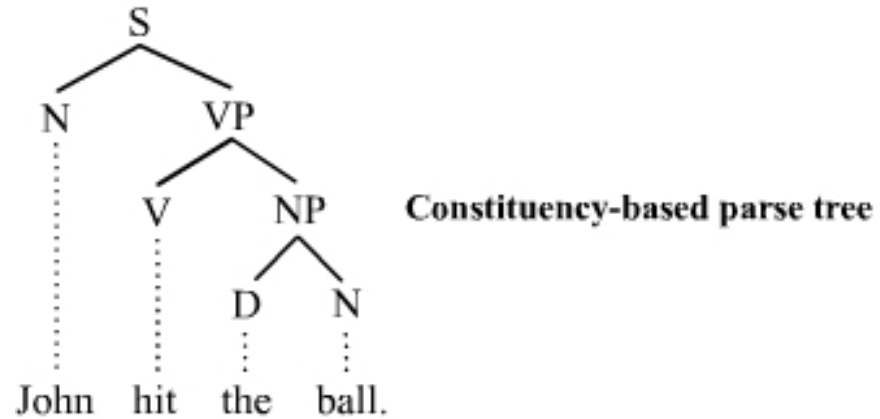
Story Ending Generation (SEG)

- ROCStories

Title	Five-sentence Story
The Test	Jennifer has a big exam tomorrow. She got so stressed, she pulled an all-nighter. She went into class the next day, weary as can be. Her teacher stated that the test is postponed for next week.  
The Hurricane	Morgan and her family lived in Florida. They heard a hurricane was coming. They decided to evacuate to a relative's house. They arrived and learned from the news that it was a terrible storm. They felt lucky they had evacuated when they did.
Spaghetti Sauce	Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting. She was touched that he pretended it was good to spare her feelings.

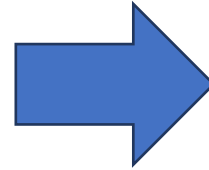
Dependency Parsing

- Dependency Tree



Dependency Parsing

The strongest rain ever recorded in India shut down the financial hub of Mumbai, snapped communication lines, closed airports and forced thousands of people to sleep in their offices or walk home during the night, officials said today.



det(rain-3, The-1)
amod(rain-3, strongest-2)
nsubj(shut-8, rain-3)
nsubj(snapped-16, rain-3)
nsubj(closed-20, rain-3)
nsubj(forced-23, rain-3)
advmod(recorded-5, ever-4)
partmod(rain-3, recorded-5)
prep_in(recorded-5, India-7)

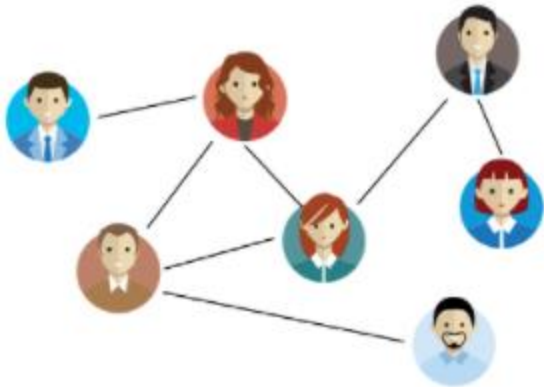
·
·
·
·
·

Graph Convolutional Network (GCN)

- Semi-Supervised Classification with Graph Convolutional Networks (ICLR 2017)

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

Node Edge

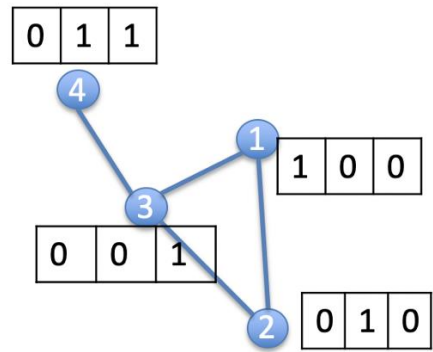


$$f(X, A)$$

X: Feature Matrix

A: Adjacency Matrix

Graph Convolutional Network (GCN)

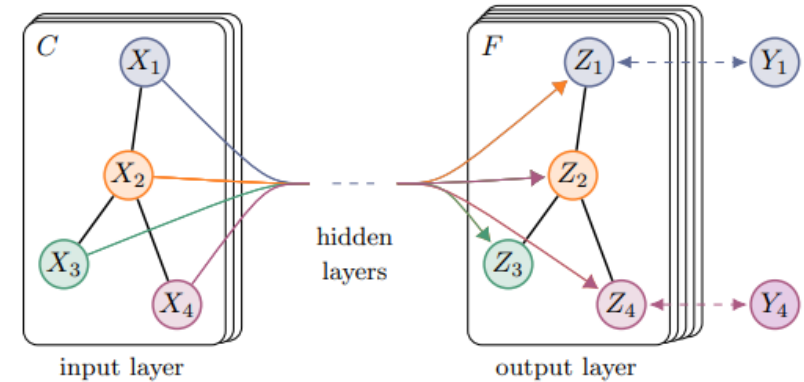


1	1	1	0
1	1	1	0
1	1	1	1
0	0	1	1

Adjacency matrix (A)

1	0	0
0	1	0
0	0	1
0	1	1

Feature matrix (X)



Graph R-CNN for Scene Graph Generation (2018)

$$f(X, A) = \text{softmax} \left(\hat{A} \text{ReLU} \left(\hat{A} X W^{(0)} \right) W^{(1)} \right)$$

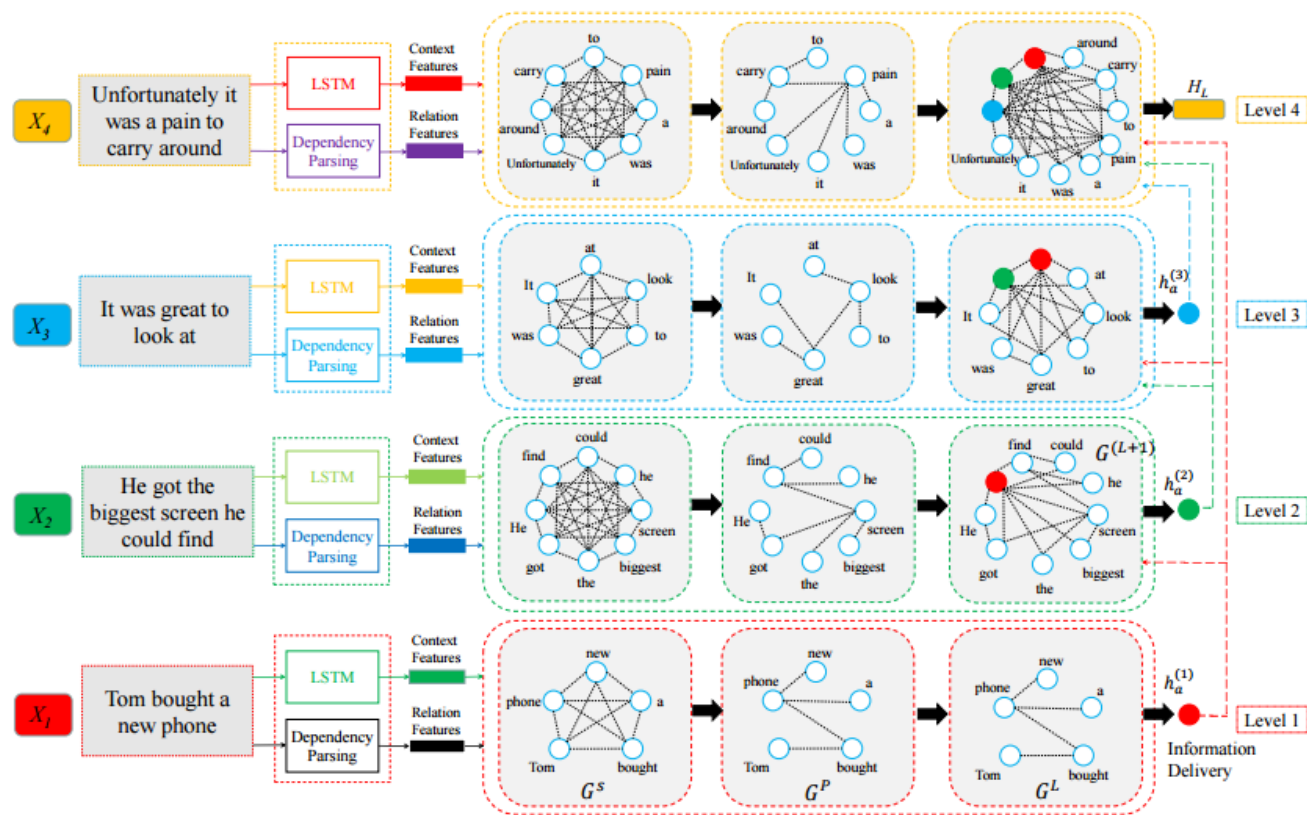


$$z_i^{(l+1)} = \sigma \left(z_i^{(l)} + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} W z_j^{(l)} \right)$$

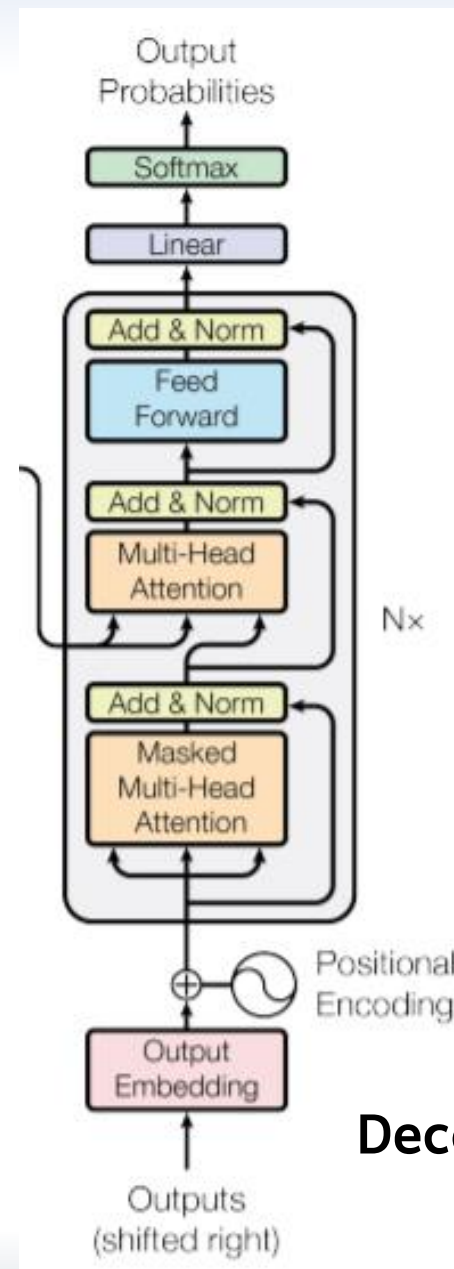
Contribution

- Multi-level Graph Convolutional Networks 방식 제안
- Dependency tree를 SEG에 적용한 첫 사례
- External resource 없이 State-of-the-art에 근접함

모델 구조

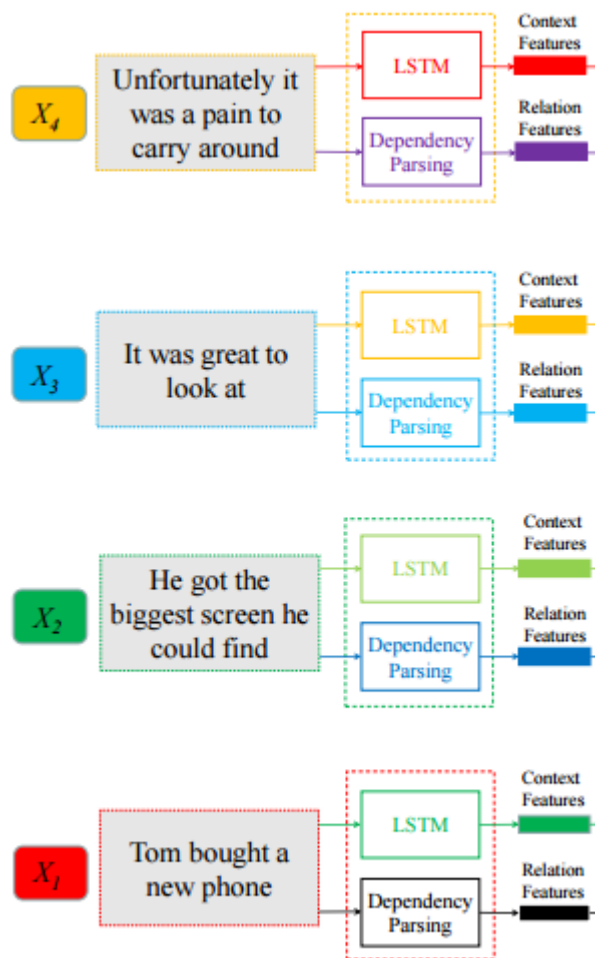


Encoder



Decoder

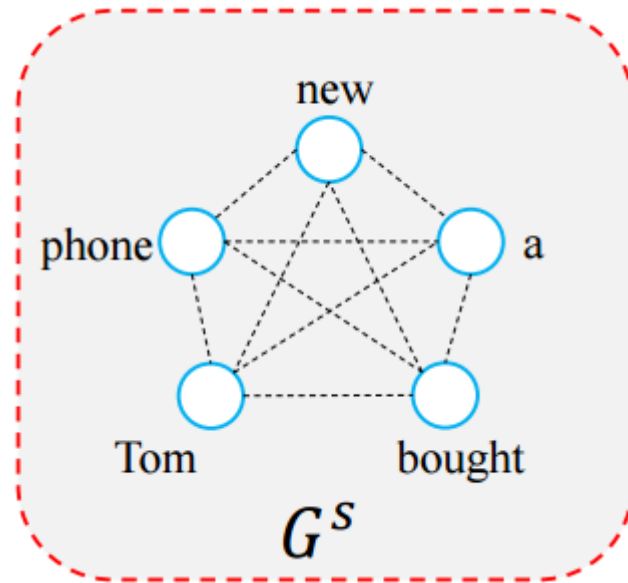
Encoder - Feature 추출



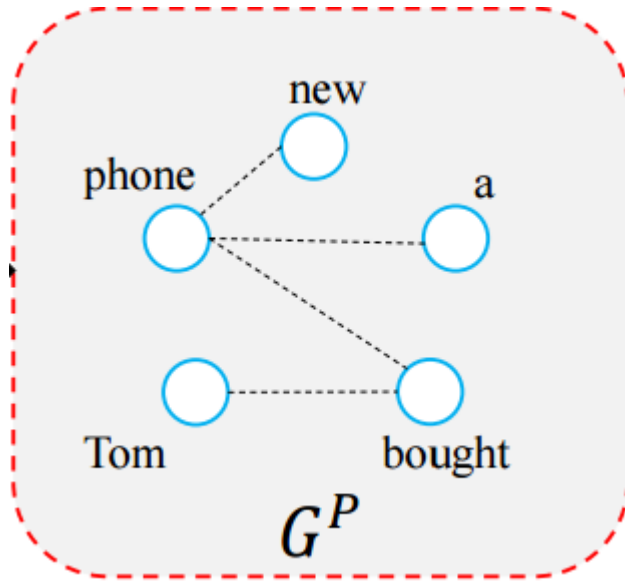
$$e_k^{(s)} = \mathbf{e}^w(x_k^{(s)})$$

$$h_{wk}^{(s)} = LSTM(e_k^{(s)})$$

Encoder - Fully connected Graph Construction



Encoder - Pruned Graph and Update



correlation score

$$w_{ij} = w_0^T \sigma(W_0[h_{wi}; h_{wj}] + b_0)$$

weight

$$\alpha_{ij} = \frac{\exp(w_{ij})}{\sum_{j \in \mathcal{N}(i)} \exp(w_{ij})}$$

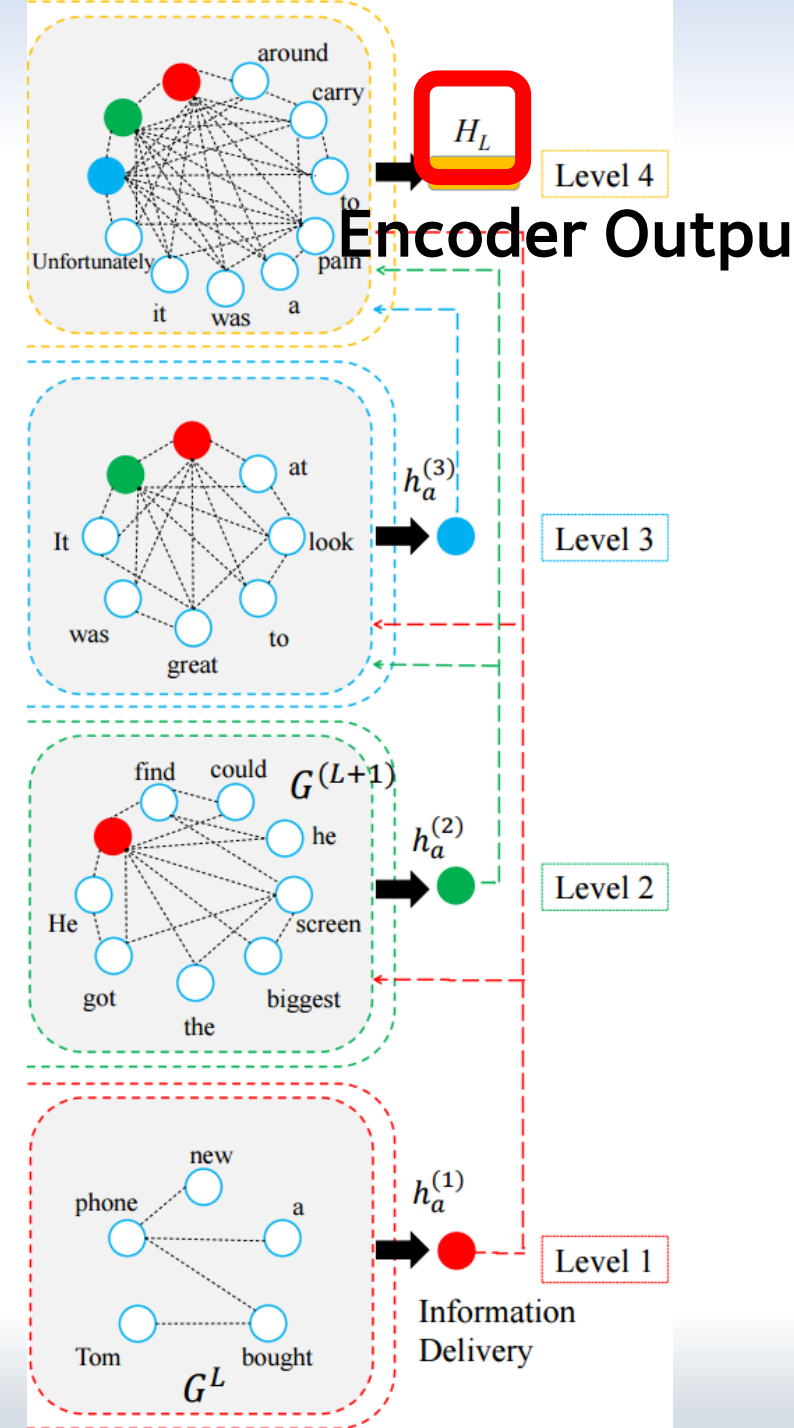
$$h_{wi}^{(l+1)} = \sigma(h_{wi}^{(l)} + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} (W_1 h_{wj}^{(l)} + b_1))$$

$$H_w = h_{wi}^{(l+1)}$$

Encoder - Multi-level GCN

$$\begin{aligned}
 \mathcal{V}^L &= [h_{w1}^{(s)} \cdots h_{wn}^{(s)}] \\
 \beta &= \text{softmax}(W_2 \mathcal{V}^L + b_2) \\
 h_a^{(L)} &= \sum_{L=1}^n \beta \mathcal{V}^L \\
 \mathcal{V}^{(L+1)} &= [h_{w1}^{(s+1)} \cdots h_{wm}^{(s+1)}; h_a^{(1)}, \dots, h_a^{(L)}]
 \end{aligned}$$

Attention 매커니즘



Encoder - Code

문장에서
context feature 추출

Dependency Parsing을 통해 얻은
Adjacency matrix

5번의 layer 업데이트

Level 1
representation 추출

Level 2에 추가

```
sent1 = self.src_word_emb(sent1)[: , :n_words]
sent2 = self.src_word_emb(sent2)[: , :n_words]
sent3 = self.src_word_emb(sent3)[: , :n_words]
sent4 = self.src_word_emb(sent4)[: , :n_words]
```

```
sent1_feat = self.intra_sent_gcn1(sent1, adj1) # (batch, n_word, hidden_size)
```

```
sent1_feat, att_1 = self.sent1_level_feat(sent1_feat, sent1_mask)
```

```
sent1_feat = sent1_feat.unsqueeze(1) # (batch, hidden_size)
```

```
sent2 = torch.cat((sent2, sent1_feat), dim=1)
```

```
sent2_feat = self.intra_sent_gcn2(sent2, adj2) # (batch, n_word+1, hidden_size)
```

```
sent2_feat, att_2 = self.sent2_level_feat(sent2_feat, sent2_mask)
```

```
sent2_feat = sent2_feat.unsqueeze(1)
```

```
sent3 = torch.cat((sent3, sent1_feat, sent2_feat), dim=1)
```

```
sent3_feat = self.intra_sent_gcn3(sent3, adj3) # (batch, n_word+2, hidden_size)
```

```
sent3_feat, att_3 = self.sent3_level_feat(sent3_feat, sent3_mask)
```

```
sent3_feat = sent3_feat.unsqueeze(1)
```

```
sent4 = torch.cat((sent4, sent1_feat, sent2_feat, sent3_feat), dim=1)
```

```
sent4_feat = self.intra_sent_gcn4(sent4, adj4) # (batch, n_word+3, hidden_size)
```

```
encoder_output = sent4_feat
```

```
return encoder_output
```

Decoder

$$\tilde{D}_{in} = \text{MultiHead}(\underbrace{D_{in}}_{\text{Decoder Input}}, \underbrace{H_L, H_L}_{\text{Encoder Output}})$$

$$D_o = FFN(\tilde{D}_{in})$$

$$P(y_t | \hat{y} < t, X) = \text{softmax}(W_5 D_o + b_5)$$

Experiments – Baseline

- Seq 2 Seq
- Transformer
- IE+MSA
- T-CVAE
- GCN
- KE-GPT2
- Plan&Write

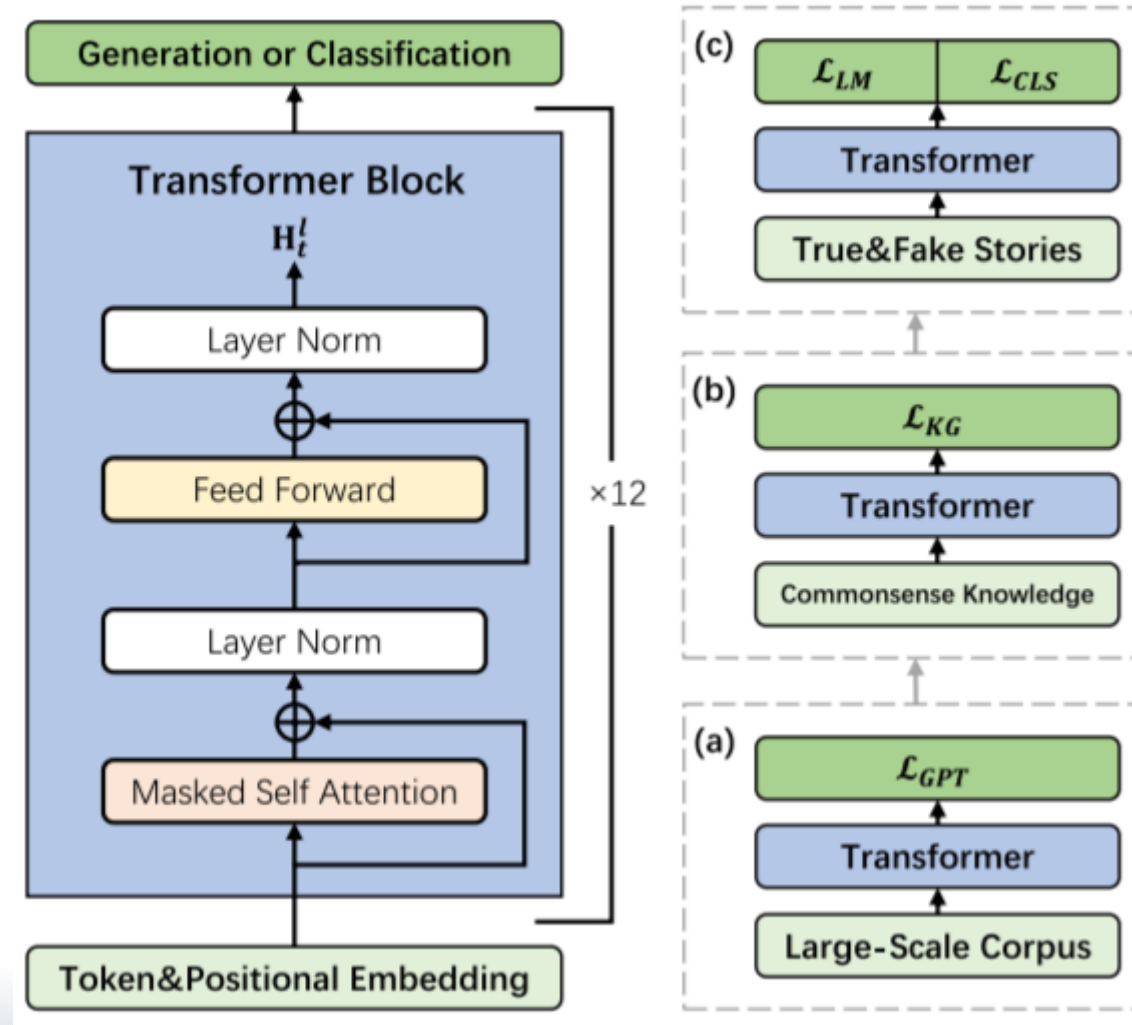
Experiments – Metric

- Automatic Evaluation Metric
 - BLEU
- Human Evaluation Metric
 - Grammaticality, Logicality

Experiments – Result

Model	B1%	B2%	Gram	Logic
Seq2Seq	18.5	5.9	2.57	1.41
Transformer	17.4	6.0	2.54	1.62
GCN	17.6	6.2	2.62	1.70
IE+MSA	24.4	7.8	2.64	1.80
T-CVAE	24.3	7.7	2.58	1.71
Plan&Write	24.4	8.4	<u>2.65</u>	1.73
KE-GPT2*	26.5	9.4	<u>2.65</u>	1.92
MGCN-DP(ours)	<u>24.6</u>	<u>8.6</u>	2.67	<u>1.86</u>

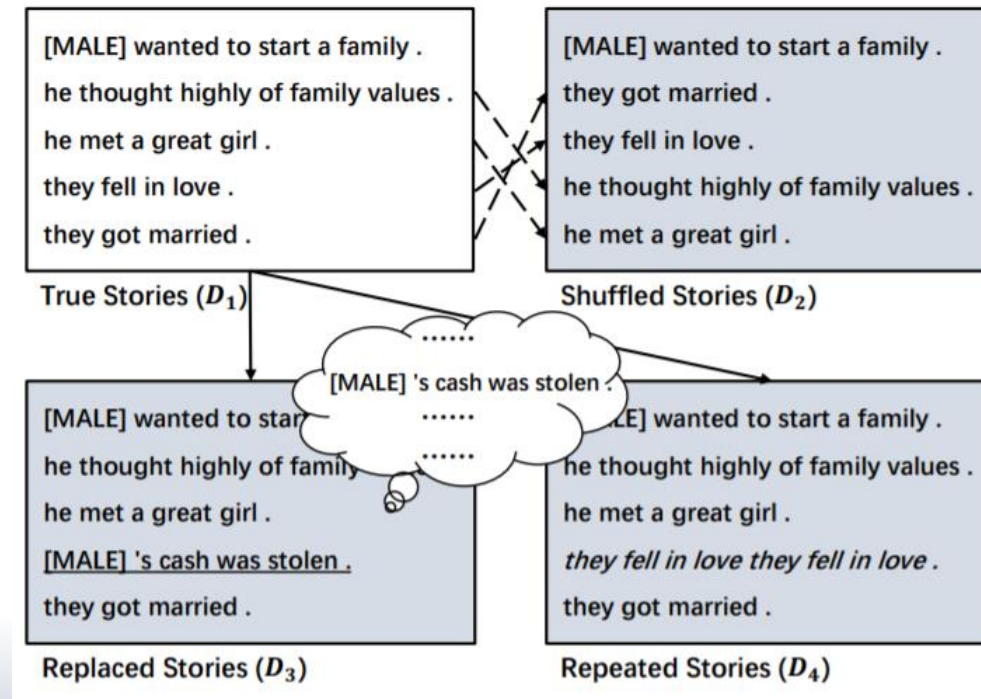
A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation



A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation

Atomic ConceptNet

Knowledge Bases	Original Triples	Examples of Transformed Sentences
ConceptNet	(eiffel tower, AtLocation , paris) (telephone, UsedFor , communication)	eiffel tower is at paris. telephone is used for communication.
ATOMIC	(PersonX dates for years, oEffect , continue dating) (PersonX cooks spaghetti, xIntent , to eat)	PersonX dates for years. PersonY will continue dating. PersonX cooks spaghetti. PersonX wants to eat.



Ablation

Model	B1%	B2%	Gram	Logic
MGCN-DP	24.6	8.6	2.67	1.86
w/o DP	22.0	7.8	2.65	1.79
w/o ML	21.7	7.6	2.64	1.77
w/o DP, ML	17.6	6.2	2.62	1.70

Relation	Description	B1%	B2%
MGCN-DP	full model	24.6	8.6
w/o <i>nsubj</i>	nominal subject	23.7	7.5
w/o <i>dobj</i>	direct object	23.7	7.5
w/o <i>acl</i>	clausal modifier	23.7	7.5
w/o <i>iobj</i>	indirect object	23.8	7.6
w/o <i>dep</i>	dependent	23.8	7.6
w/o <i>amod</i>	adjective modifier	23.8	7.7
w/o <i>case</i>	pre/post-positions	23.9	7.8
w/o <i>advmod</i>	adverbial modifier	23.9	7.9
w/o <i>det</i>	determiner	24.0	8.0
w/o <i>cop</i>	copula	24.0	8.1
w/o <i>mark</i>	marker	24.1	8.1
w/o <i>xcomp</i>	open clausal	24.2	8.2

Q&A