P – Stage 2

문장 내 개체간 관계 추출 Solution

발표자 : 정익효, 이현규

시작에 앞서

최종 Submit(81.9)은 정익효(81.7), 이현규(81.5), 권태양(81.4)의 Ensemble 결과입니다.

개인보다는 Team의 Solution으로 이해해주시면 감사하겠습니다.

목차

- 정익효 Solution
  1) 개요
  2) Models
  3) Preprocessing
  4) Random Masking
  5) Random Switching
  6) Truncation
  7) Add Weight

- 이현규, 권태양 Solution
  1) 개요
  2) Preprocessing
  3) Models
  4) Back Translation

정익효's Solution

- 개요
- Models
- Preprocessing
- Random Masking
- Random Switching
- Truncation
- Add Weight

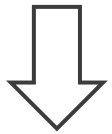
Solution 개요

Models	**XLM-RoBERTa-large,** **XLM-RoBERTa-large + LSTM Classifier,** **KoElectra**
Loss Function	Focal Loss (gamma = 0.5)
Optimizer	AdamW (lr = 1e-5)
Scheduler	Cosine Annealing LR Scheduler (T_max=10, eta_min=1e-6)
Ensemble	Hard Voting
Core Ideas	**Random Masking,** **Random Switching,** **Truncation,** **Add Weight to 0 Class**

Models

LSTM Classifier

```
  )
  (classifier): RobertaClassificationHead(
    (dense): Linear(in_features=1024, out_features=1024, bias=True)
    (dropout): Dropout(p=0.0, inplace=False)
    (out_proj): Linear(in_features=1024, out_features=42, bias=True)
  )
```



```
)
(lstm): LSTM(1024, 1024, num_layers=3, batch_first=True, dropout=0.3, bidirectional=True)
(dense_layer): Linear(in_features=2048, out_features=42, bias=True)
```

Preprocessing

기존 base line code 에서는 sep토큰으로 entity_01과 enity_02를 이어 붙여줌

sep 토큰은 이곳뿐만 아니라 entity와 문장 사이, 문장과 padding 사이에도 들어간다.
=〉 entity 들의 관계를 의미 한다기보다 분리하는 기능

sep토큰 이외의 다른 단어, 문장 사용

Ex1) 이순신 SEPARATE 조선
Ex2) 이순신 RELATION 조선
Ex3) 이순신과 조선의 관계는?

Random Masking

Ex) 이순신은 조선 중기의 무신이다. ➡ 이순신은 〈mask〉 중기의 무신이다.

train 하는 쪽 코드 부분에 추가하여 배치마다 확률 값을 가지게 하여 랜덤으로 mask생성

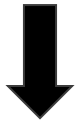
Random Switching

Ex1) 이순신은 조선 중기의 무신이다. ➡ 무신이다. 이순신은 조선 중기의

Ex2) 이순신은 조선 중기의 무신이다. ➡ 중기의 무신이다. 이순신은 조선

Truncation

Ex) 1990년 미국 알래스카주가 처음으로 출마 권리를 획득하였지만, 정치적 정당으로 초점이 맞춰지면서 어떻게 미국의 정치 시스템에 맞춰야 하는지, 녹색 통신 연락 위원회가 정치적 정당인지, 정치적 정당이 어떻게 녹색 운동에 관련이 되는지 등의 문제에 직면하였고, 이러한 문제를 해결하기 위해 1991년 투표를 통하여 녹색 통신 연락 위원회에서 ""녹색/녹색당 USA"" 으로 개명하였다.



1990년 미국 알래스카주가 처음으로 출마 권리를 획득하였지만, 정치적 정당으로 초점이 맞
〈/s〉
러한 문제를 해결하기 위해 1991년 투표를 통하여 녹색 통신 연락 위원회에서 ""녹색/**녹색당** USA"" 으로 개명하였다.

entity_01 과 entity_02 주변 사이의 문자들에 집중

Add Weight to 0 Class

주어진 데이터 셋의 분포는 관계없음 라벨 (0)이 약 50%정도로 높게 분포 + validation set의 틀린 비율 중 0이 매우 높았다.

=〉 0에 가중치를 주면 어떨까?

최종 모델 출력결과물 (softmax) 0번째 라벨에 0.1 정도의 가중치를 더하여 최종 submission 생성

위험한 방법이지만 해당 competition에서는 대부분의 모델에서 효과를 보았다.

이현규 & 권태양's Solution

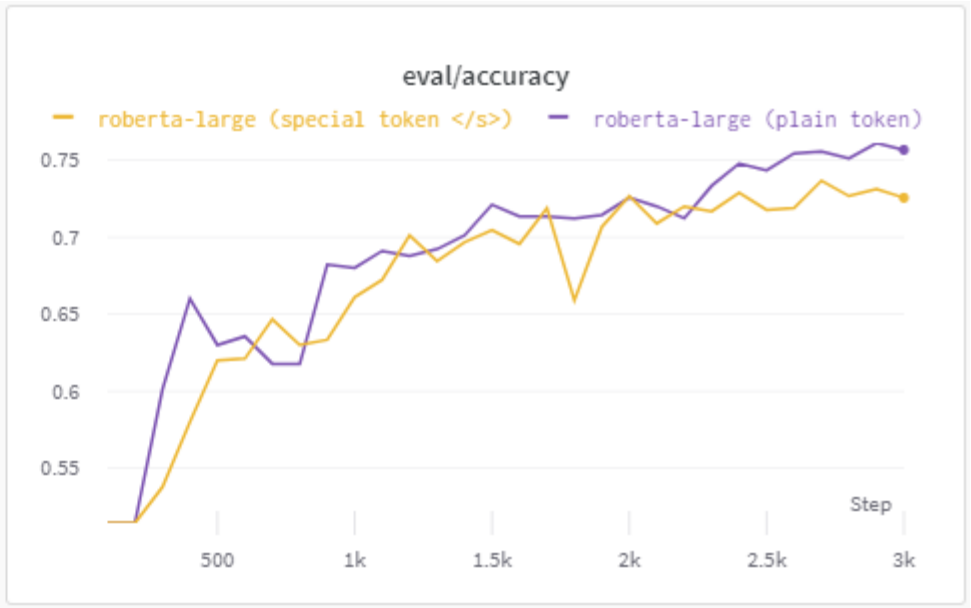
- 개요
- Preprocessing
- Models
- Back Translation

Solution 개요

Models	XLM-RoBERTa-large, XLM-RoBERTa-large + LSTM Classifier, **R-BERT (backbone = XLM-RoBERTa-large)**
Loss Function	Focal Loss (gamma = 0.5)
Optimizer	AdamP (lr = 3e-5)
Scheduler	Cosine Annealing LR Scheduler (T_max=max_steps, eta_min=1e-6), Warm-up (Step = 500)
Ensemble	Hard Voting
Core Ideas	**Back Translation,** Random Masking, Truncation, Add Weight to 0 Class

Preprocessing

Plain Text 와 Multi Sentence를 고정으로 사용



Preprocessing

기존 문장 : 이순신은 임진왜란에서 큰 활약을 한 장수로서 … 조선을 큰 위기로부터 지켜낸 영웅이다.
Entity1 : 이순신
Entity2 : 조선

Ex1) 〈s〉 이순신은 … 〈/s〉 … 조선을 … 〈/s〉 〈/s〉 이순신과 조선의 관계는? 〈/s〉

Preprocessing

기존 문장 : 이순신은 임진왜란에서 큰 활약을 한 장수로서 … 조선을 큰 위기로부터 지켜낸 영웅이다.
Entity1 : 이순신
Entity2 : 조선

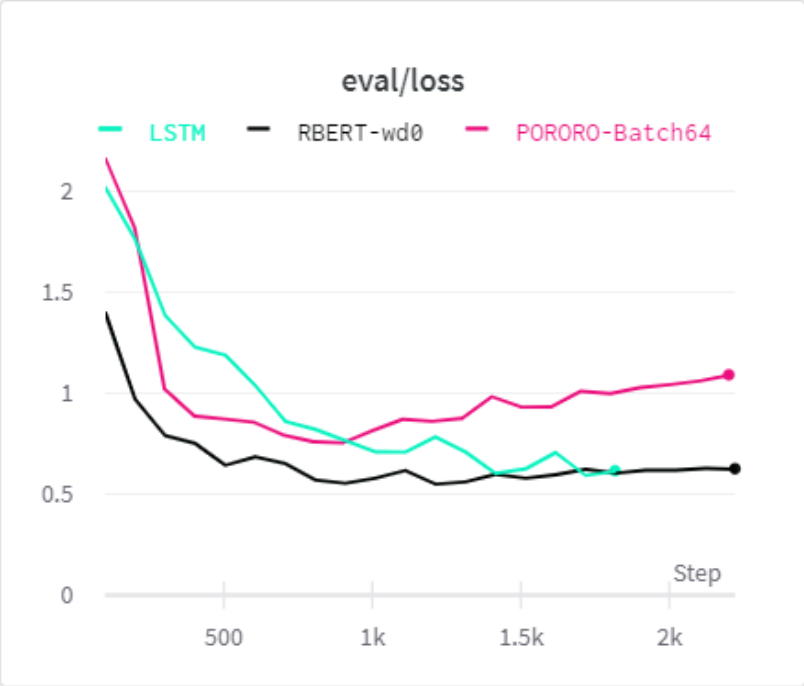
Ex2) 〈s〉 **〈e1〉**이순신**〈e1〉**은 … 〈/s〉 … **〈e2〉**조선**〈e2〉**을 … 〈/s〉 〈/s〉 이순신과 조선의 관계는? 〈/s〉

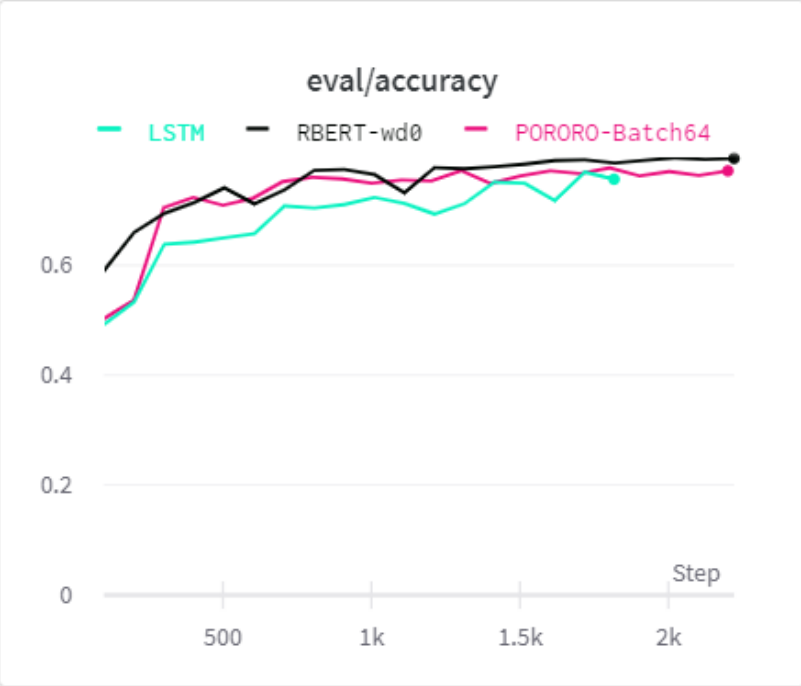
Preprocessing

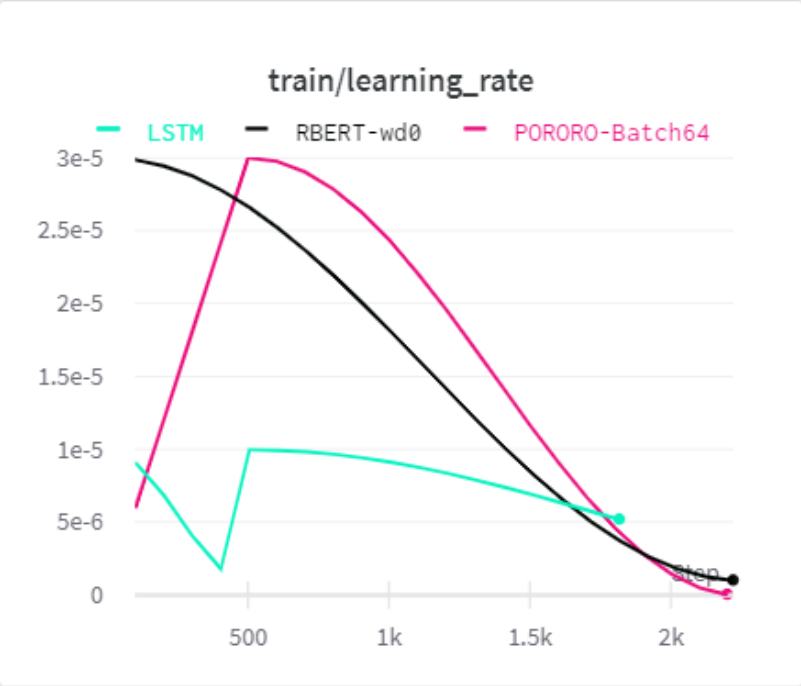
기존 문장 : 이순신은 임진왜란에서 큰 활약을 한 장수로서 … 조선을 큰 위기로부터 지켜낸 영웅이다.
Entity1 : 이순신
Entity2 : 조선

Ex3) 〈s〉 이순신 **RELATION** 조선 〈/s〉 〈/s〉 이순신은 **〈mask〉**에서 … 조선을 **〈mask〉** 위기로부터… 〈/s〉

Models

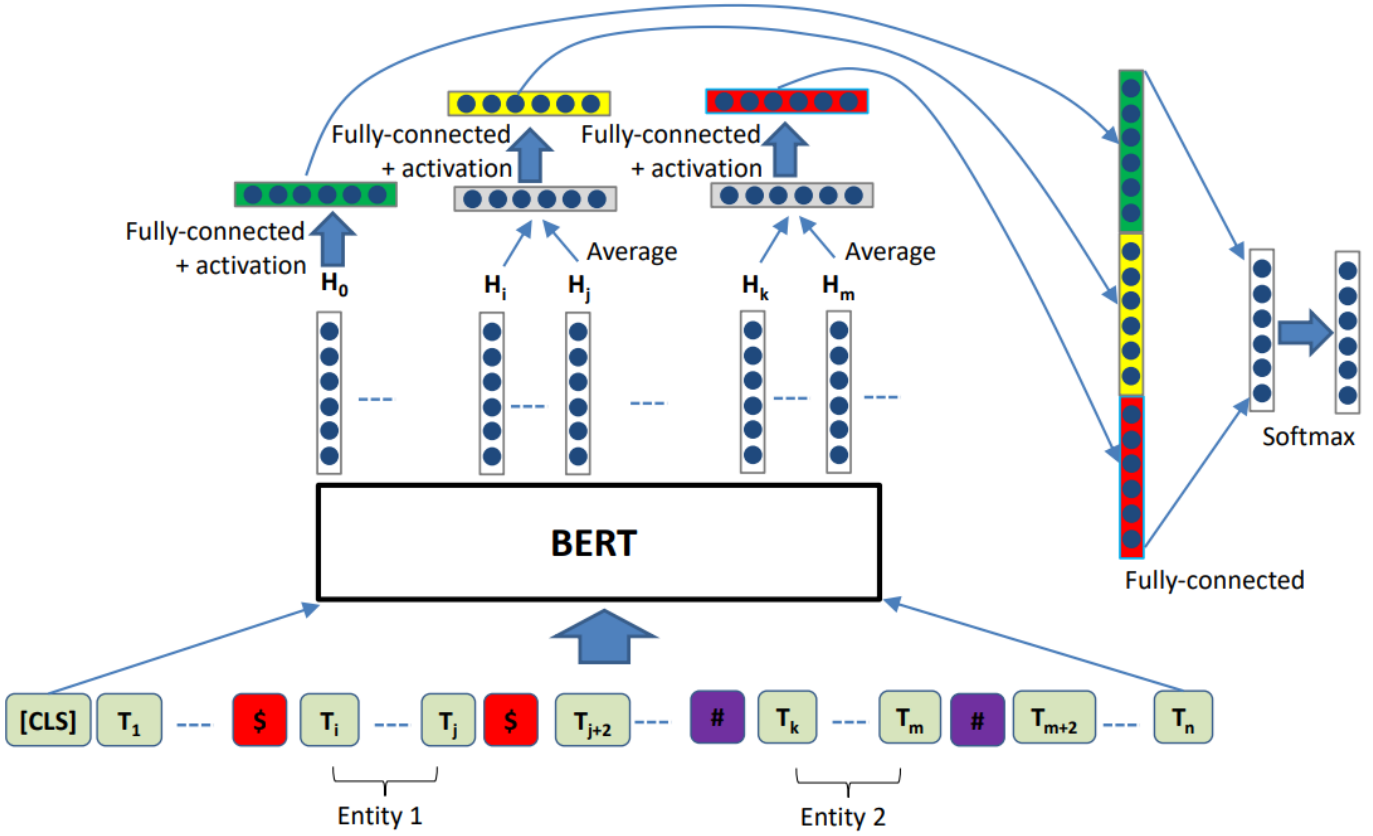






Models

R-BERT Architecture Image



R-BERT Codes

1. Model
2. Entity Mask

Models

R-BERT Architecture Image

R-BERT Codes

1. Model
2. Entity Mask

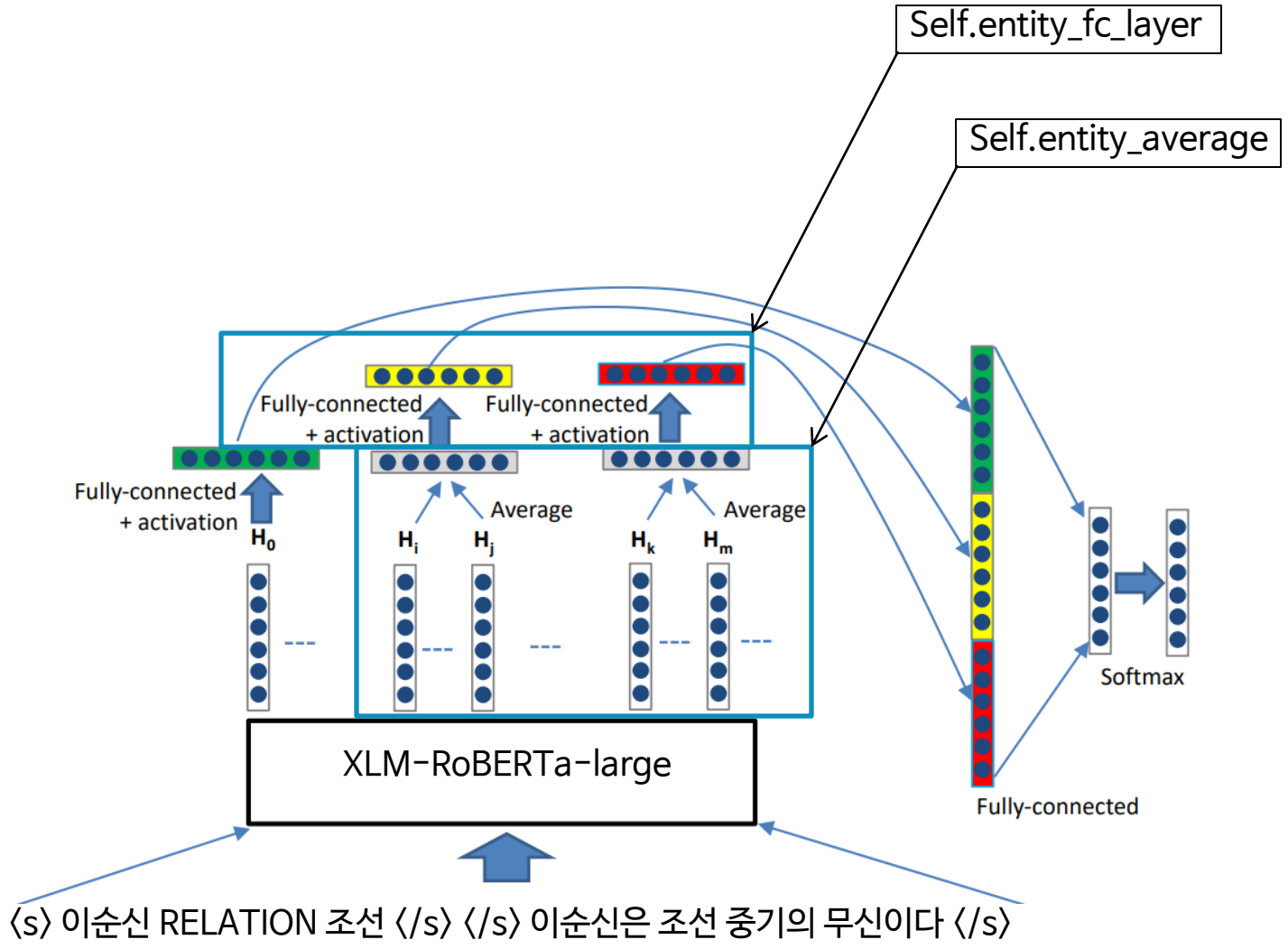
```python
class RBERT_RobertaForSequenceClassification(nn.Module):
    def __init__(self, model_name, num_classes, dr_rate=0.1):
        super(RBERT_RobertaForSequenceClassification, self).__init__()

        config = transformers.AutoConfig.from_pretrained(model_name)
        config.num_labels = config.hidden_size
        self.backbone = transformers.AutoModel.from_pretrained(model_name, config=config)
        self.num_classes = num_classes
        self.dropout_rate = dr_rate

        self.cls_fc_layer = FCLayer(config.hidden_size, config.hidden_size, self.dropout_rate)
        self.entity_fc_layer = FCLayer(config.hidden_size, config.hidden_size, self.dropout_rate)
        self.label_classifier = FCLayer(config.hidden_size*3, self.num_classes, self.dropout_rate, use_activation=False)

    def forward(self, input_ids, attention_mask, e1_mask, e2_mask, labels=None):
        outputs = self.backbone(input_ids=input_ids,
                                attention_mask=attention_mask)

        sequence_output = outputs['last_hidden_state']
        pooled_output = outputs['pooler_output']  # [CLS]

        e1_h = self.entity_average(sequence_output, e1_mask)
        e2_h = self.entity_average(sequence_output, e2_mask)

        pooled_output = self.cls_fc_layer(pooled_output)
        e1_h = self.entity_fc_layer(e1_h)
        e2_h = self.entity_fc_layer(e2_h)

        concat_h = torch.cat([pooled_output, e1_h, e2_h], dim=-1)
        logits = self.label_classifier(concat_h)
        return logits
```

Models

R-BERT Architecture Image

R-BERT Codes

1. Model
2. Entity Mask



Models

R-BERT Architecture Image

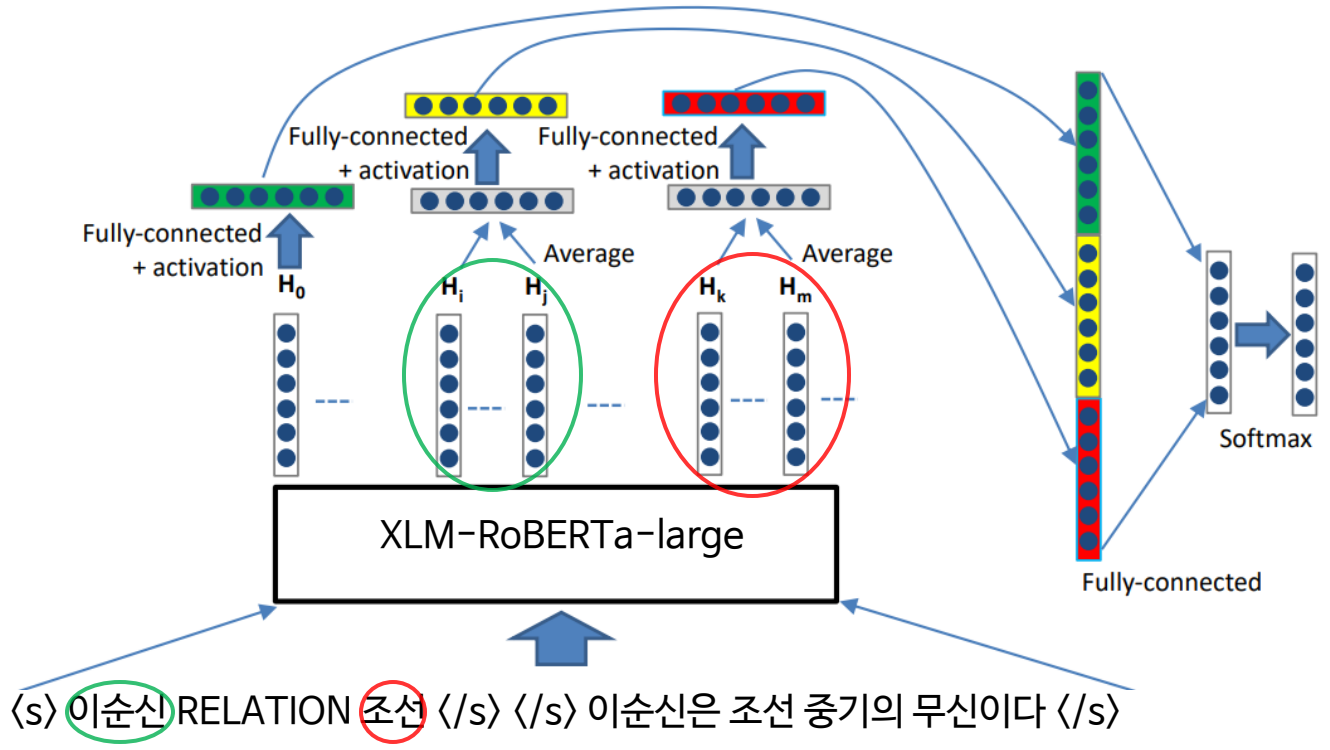
R-BERT Codes

1. Model
2. Entity Mask

```
def __getitem__(self, idx) :
    if self.training :
        sentences = [self.origin_sentence[idx], ]
        if self._is_sentence(self.en_sentence[idx]) :
            sentences.append(self.en_sentence[idx])
        if self._is_sentence(self.ja_sentence[idx]) :
            sentences.append(self.ja_sentence[idx])
        if self._is_sentence(self.zh_sentence[idx]) :
            sentences.append(self.zh_sentence[idx])
        sentence = sentences[np.random.randint(len(sentences))]
    else :
        sentence = self.origin_sentence[idx]

    e1 = self.entity_01[idx]
    e2 = self.entity_02[idx]
    e1_mask, e2_mask = self._get_ent_mask(e1, e2)

    item = tokenizer(e1+' RELATION '+e2, sentence, max_length=CFG.tokenizer_max_length,
                     padding='max_length', truncation=True, return_tensors='pt')
    item['input_ids'] = self._random_mask(item['input_ids'][0], e1, e2)
    item['attention_mask'] = item['attention_mask'].squeeze(0)
    item['e1_mask'] = torch.Tensor(e1_mask)
    item['e2_mask'] = torch.Tensor(e2_mask)
    item['labels'] = self.labels[idx]
    return item
```

Models

R-BERT Architecture Image



R-BERT Codes

1. Model
2. Entity Mask

Back Translation

Back Translation Datas

1. DataFrame
2. Dataset

origin_sentence	en_sentence	ja_sentence	zh_sentence	entity_01	entity_02	labels
영국에서 사용되는 스포츠 유틸리티 자동차의 브랜드로는 랜드로버(Land Rover)...	영국에서 사용되는 스포츠카 브랜드에는 랜드로버와 지프가 포함되어 있으며, 이들 브랜...	영국에서 사용되는 스포츠 유틸리티 자동차 브랜드에서는 랜드로버와 지프가 있고, 이 ...	영국 스포츠타운 자동차의 브랜드는 랜드로버와 제프입니다	랜드로버	자동차	17
선거에서 민주당은 해산 전 의석인 230석에 한참 못 미치는 57석(지역구 27석,...	선거에서는 민주당이 해산 전 230석을 훨씬 밑도는 57석(구 27석, 비례대표 3...	선거에서 민주당은 해산 전 의석의 230석에 아직 못 미치는 57석(지역구 27석,...	NaN	민주당	27석	0
유럽 축구 연맹(UEFA) 집행위원회는 2014년 1월 24일에 열린 회의를 통해 ...	NaN	NaN	NaN	유럽 축구 연맹	UEFA	6
용병 공격수 챠디의 부진과 시즌 초 활약한 강수일의 침체, 시즌 중반에 영입한 세르...	부진한 공격수 차디와 시즌 초반 활약을 펼친 강수일의 부진, 중반 영입한 세르비아 ...	NaN	NaN	강수일	공격수	2
람캄행 왕은 1237년에서 1247년 사이 수코타이의 왕 퍼쿤 씨 인트라팃과 쓰엉 ...	NaN	NaN	NaN	람캄행	퍼쿤 씨 인트라팃	8

Back Translation

Back Translation Datas

1. DataFrame
2. Dataset

```python
class NLPDataset(Dataset) :
    def __init__(self, dataset, tokenizer, training=False, threshold=0.1) :
        self.origin_sentence = dataset['origin_sentence']
        self.en_sentence = dataset['en_sentence']
        self.ja_sentence = dataset['ja_sentence']
        self.zh_sentence = dataset['zh_sentence']
        self.entity_01 = dataset['entity_01']
        self.entity_02 = dataset['entity_02']
        self.labels = torch.tensor(dataset['labels'])
        self.tokenizer = tokenizer
        self.training = training
        self.threshold = threshold

    def __getitem__(self, idx) :
        if self.training :
            sentences = [self.origin_sentence[idx], ]
            if self._is_sentence(self.en_sentence[idx]) :
                sentences.append(self.en_sentence[idx])
            if self._is_sentence(self.ja_sentence[idx]) :
                sentences.append(self.ja_sentence[idx])
            if self._is_sentence(self.zh_sentence[idx]) :
                sentences.append(self.zh_sentence[idx])
            sentence = sentences[np.random.randint(len(sentences))]
        else :
            sentence = self.origin_sentence[idx]
```

FAQ (by 영웅님)

앙상블을 하게 된 계기는 무엇이었나요?

대회 중 이렇게 하면 큰 일 나겠다 싶었던 순간이 있나요?
- 이를 위해 어떤 시도를 했고, 결과는 어땠나요?

학습에 도움이 될만한 Tip이 있을까요?

END

감사합니다.