계층적 셀프 어텐션 GRU

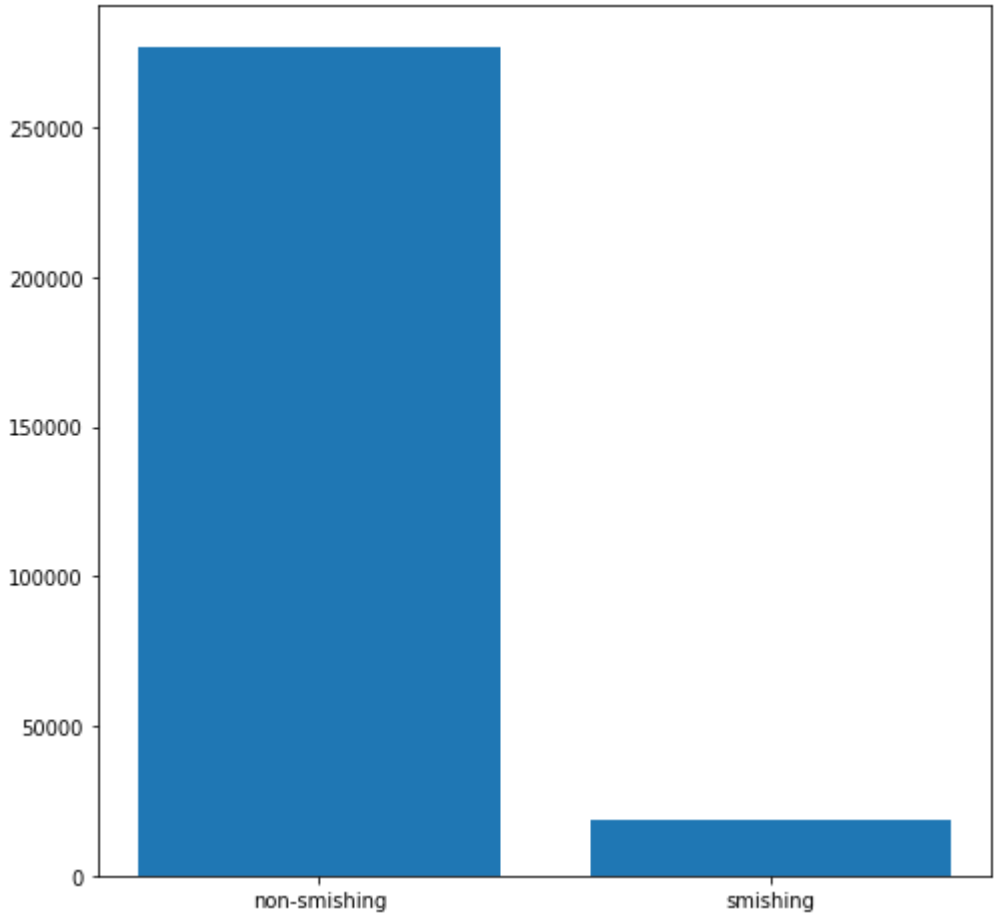
목적

스미싱 문자 탐지

1. KB에서 공개한 29만건의 정상, 스미싱 문자 데이터를 토대로 ‘해당 문자가 스미싱 문자일 확률 ' 을 예측하는 경진대회

EDA

- 타겟변수의 분포



비 스미싱 / 전체문자 = 0.063
비 스미싱 / 스미싱 = 0.067

압도적인 불균형이 특징

EDA

- 스미싱과 비스미싱 문자간 문장 길이의 차이 검정

	smishing
count	18703
mean	801
std	206.203451
min	39
25%	689
50%	876
75%	917
max	1230

	Non-smishing
count	277242
mean	133.742391
std	149.335552
min	1
25%	39
50%	75
75%	172
max	1498

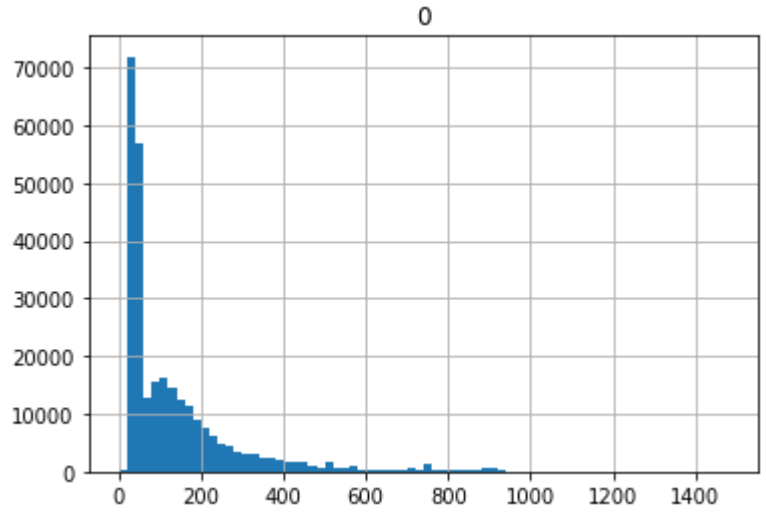
스미싱 문자의 평균 : 801

비스미싱 문자의 평균 : 133

분명 평균에 차이가 있어 보인다.

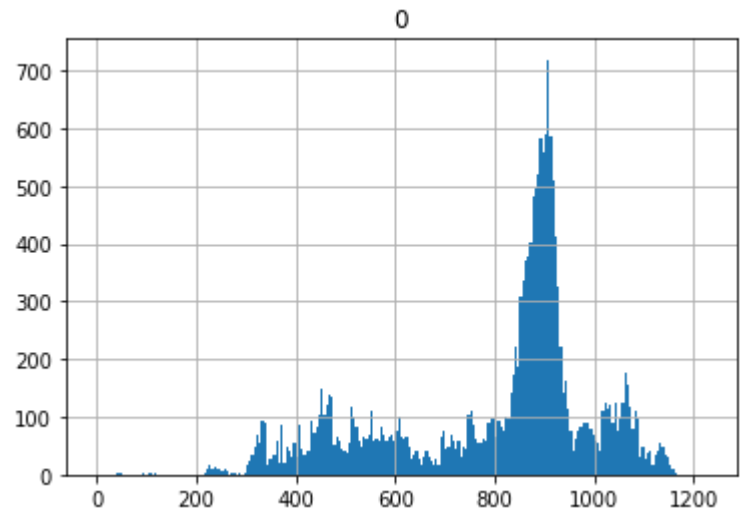
EDA

- 스미싱과 비스미싱 문자간 문장 길이의 차이 검정




```
1 stats.kstest(np.array(length_1_sample.T)[0],"norm")
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```




```
1 stats.kstest(np.array(length_0_sample.T)[0],"norm")
```

```
KstestResult(statistic=0.9999999999999993, pvalue=0.0)
```

문장 길이 분포의 경우, 둘 다 명백히 정규분포에서 추출되지 않았으므로
비모수 검정인 KS검정을 수행한다.

EDA

- 스미싱과 비스미싱 문자간 문장 길이의 차이 검정

	smishing
count	18703
mean	801
std	206.203451
min	39
25%	689
50%	876
75%	917
max	1230

	Non-smishing
count	277242
mean	133.742391
std	149.335552
min	1
25%	39
50%	75
75%	172
max	1498

```
1  ## KS검정
2
3  stats.ks_2samp(np.array(length_1_sample.T)[0],np.array(length_0_sample.T)[0])
```

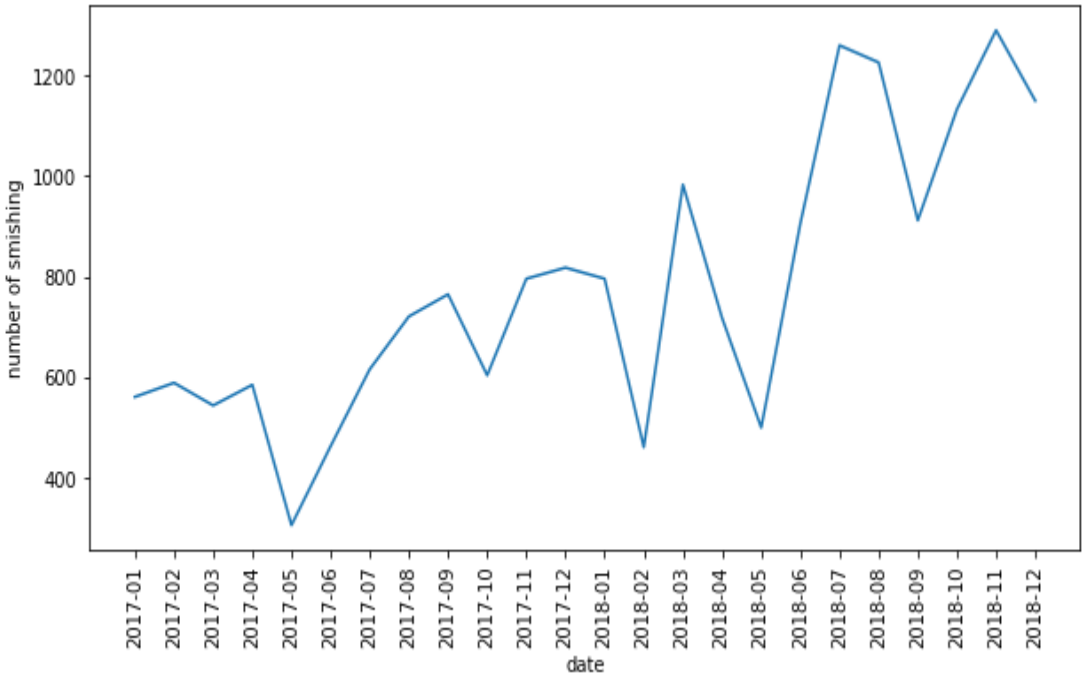
```
Ks_2sampResult(statistic=0.8889, pvalue=0.0)
```

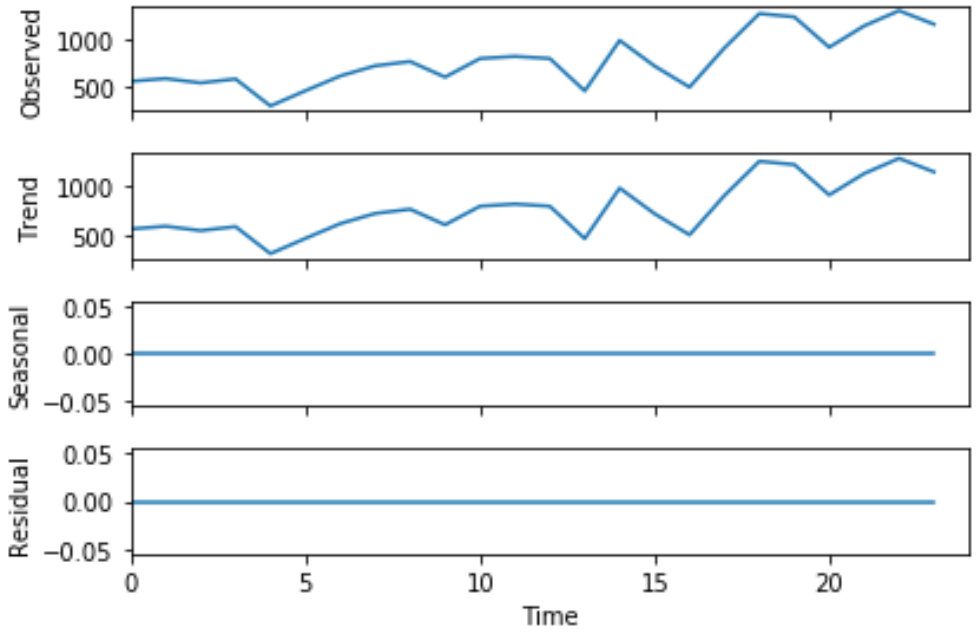
2-sample KS검정 결과, 두 분포가 동일한 분포에서 추출됐다는 귀무가설을 기각한다.

즉, 두 문장길이 분포의 경험적 누적분포는 같지 않다.

EDA

- 스미싱 문자의 시계열성





스미싱 문자의 경우, 증가하는 추세는 보이지만 다른 성분은 파악되지 않는다.

EDA 결론

1. 타겟 불균형이 심각.

1) 단순히 모델이 전부 '스미싱이 아니다 ' 라고 판단하는 경우라도 정확도는 93% 도출

2) 따라서, 해당 경진대회에선 평가 함수로 AUC를 사용하도록 규정

2. 스미싱 문자는 정상 문자에 비해 길이가 유의하게 길다.

1) 사기꾼들이 피해자들에게 혼동을 주기 위해 일부러 길게 보내는 특성이 존재?

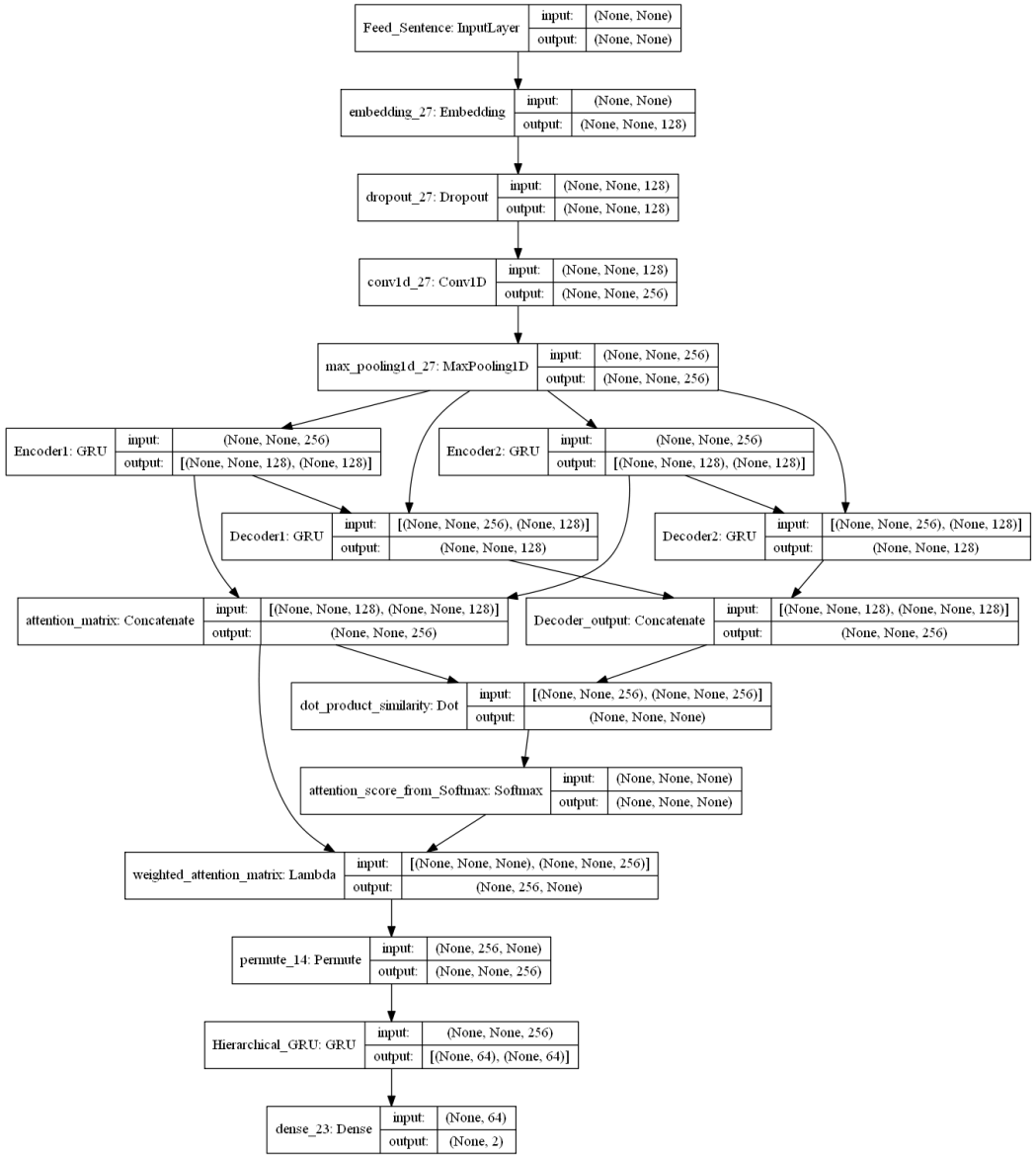
2) RNN계열 모델을 사용하는 경우, 제로패딩의 존재 자체가 단서로 작용할 수 있음

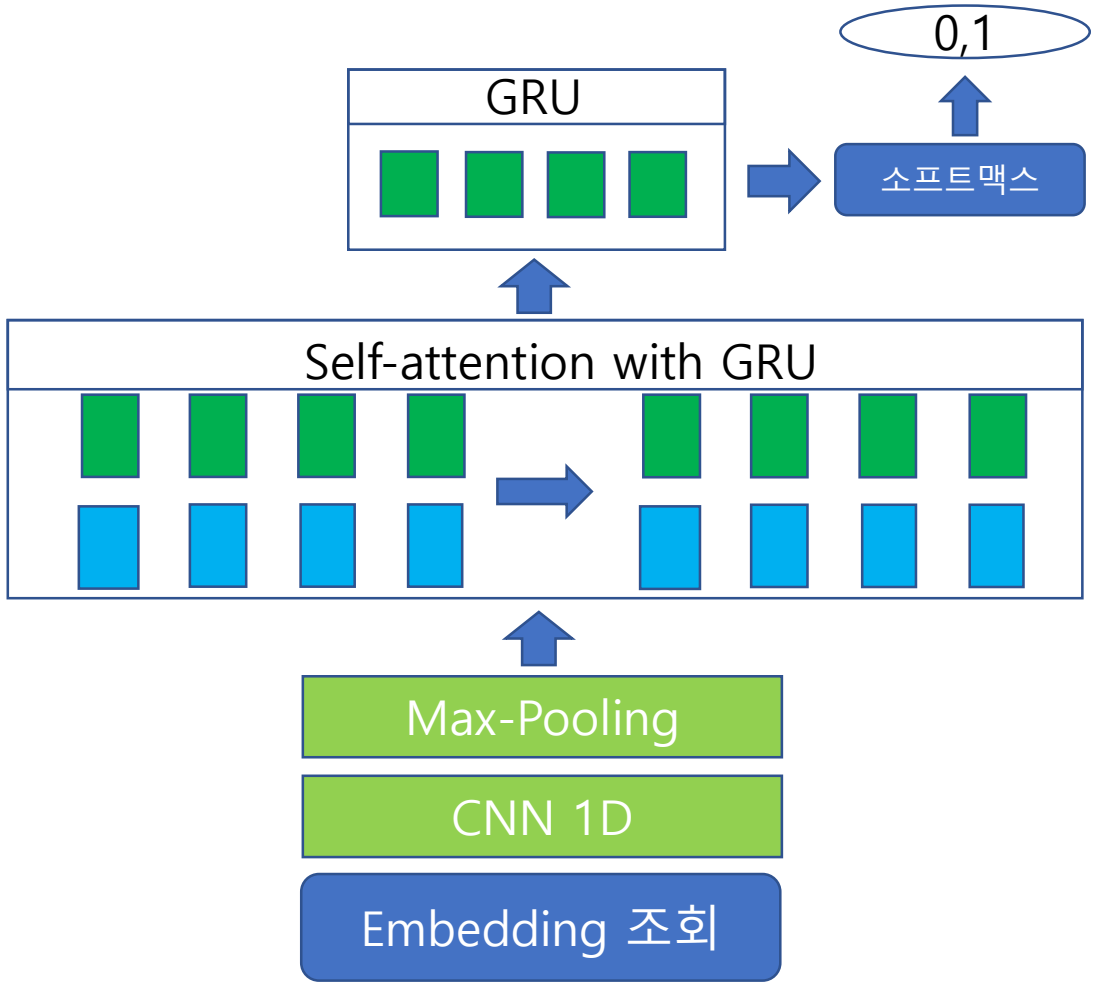
- 제로패딩이 많다 -> 문장 길이가 짧다 -> 정상문자이다

- 제로패딩이 적다 -> 문장 길이가 길다 -> 스미싱 문자이다

타겟 불균형에 강건하며, 제로 패딩 정보를 반영할 수 있는 모델 적합 필요

계층적 셀프 어텐션 GRU

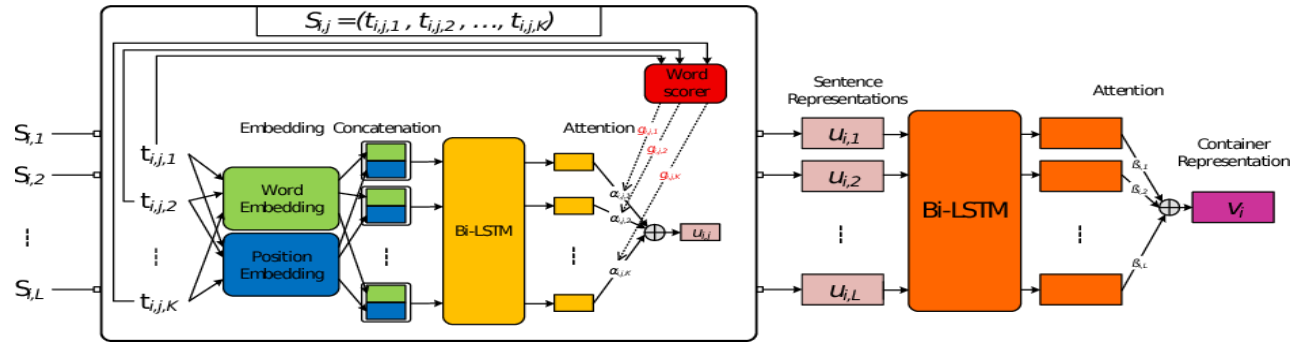




계층적 셀프 어텐션 GRU

개요

1. Self-attention을 활용한 계층적 LSTM을 활용한 논문1)을 참조



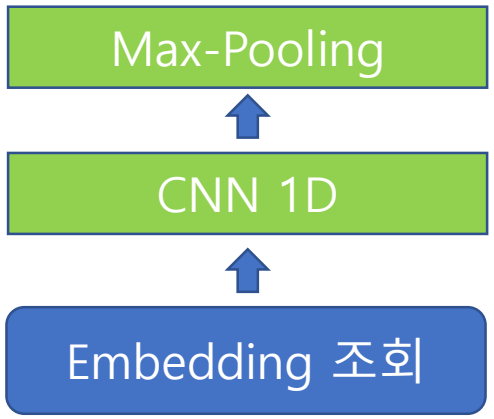
출처 : who is Killed by Police: Introducing Supervised Attention for Hierarchical LSTMs, M Nguyen

2. '스미싱인가 스미싱이 아닌가 ' 를 판단하는데 도움이 되는 단어를 강조하는 방식의 작동을 기대

➡ 심각한 타겟 불균형 상황에서도 특정 단어를 부각할 수 있음

3. RNN계열 신경망을 사용하여, 제로 패딩의 양을 모델에 반영 가능

계층적 셀프 어텐션 GRU

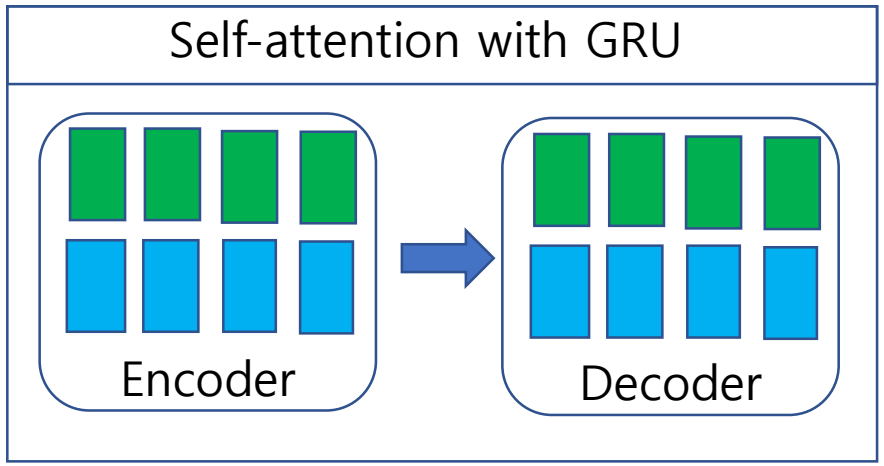


1. 특성 추출

- 1차원 컨볼루션으로 임베딩 행렬의 차원(d=256)에서 특성만 추출

- 맥스 풀링으로 차원을 대폭 축소하여 self-attention에 전달

계층적 셀프 어텐션 GRU



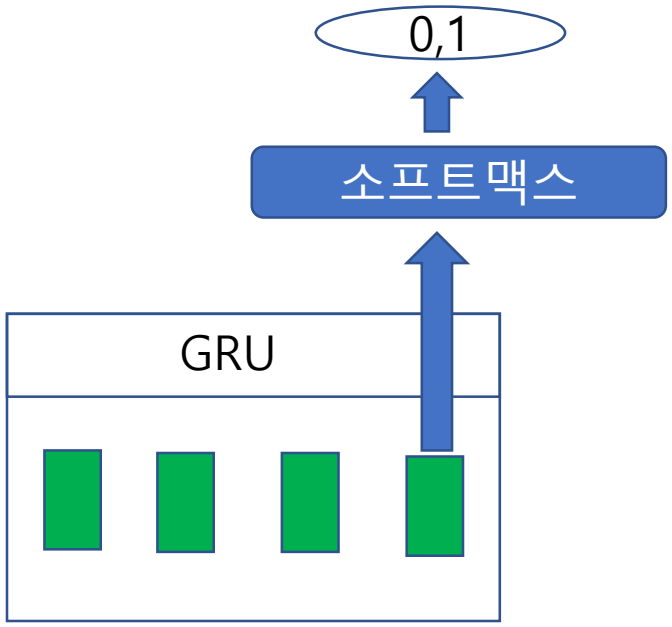
1. 셀프 어텐션

- 인코더와 디코더에 동일한 임베딩 벡터 주입
- 다음의 형식으로 셀프 어텐션을 수행

$$\{softmax(X \cdot Y^T) \cdot X\}/\sqrt{d_Y}\}$$

- 이 때, X = (인코더 임베딩) * We,
  Y = (디코더 임베딩) * Wd

계층적 셀프 어텐션 GRU



3. 계층적 GRU

- 셀프 어텐션에서 출력된 어텐션 벡터를 입력 값으로 받아 시계열적인 패턴을 학습

- 마지막 Hidden-state를 토대로 스미싱 여부를 판단

테스트 결과

- CNN – LSTM

```
Train on 221958 samples, validate on 73987 samples
Epoch 1/10
221958/221958 [==============================] - 1409s 6ms/step - loss: 5.0465e-04 - acc: 0.9999 - val_loss: 0.0010 - val_acc: 0.9999
Epoch 2/10
221958/221958 [==============================] - 1406s 6ms/step - loss: 2.5936e-04 - acc: 1.0000 - val_loss: 0.0012 - val_acc: 0.9999
Epoch 3/10
221958/221958 [==============================] - 1420s 6ms/step - loss: 2.6522e-04 - acc: 1.0000 - val_loss: 0.0025 - val_acc: 0.9995
Epoch 4/10
221958/221958 [==============================] - 1418s 6ms/step - loss: 2.6839e-04 - acc: 1.0000 - val_loss: 0.0014 - val_acc: 0.9999
```

Validation 기준 AUC 스코어 : 0.998

submission 기준 AUC 스코어 : 0.95

- 계층적 셀프 어텐션 GRU

```
Train on 221958 samples, validate on 73987 samples
Epoch 1/20
221958/221958 [==============================] - 6739s 30ms/step - loss: 0.0038 - acc: 0.9988 - val_loss: 8.6905e-04 - val_acc: 0.9998
Epoch 2/20
221958/221958 [==============================] - 6777s 31ms/step - loss: 7.1111e-04 - acc: 0.9999 - val_loss: 0.0011 - val_acc: 0.9998
Epoch 3/20
221958/221958 [==============================] - 6167s 28ms/step - loss: 3.8665e-04 - acc: 0.9999 - val_loss: 0.0010 - val_acc: 0.9998
Epoch 4/20
221958/221958 [==============================] - 6313s 28ms/step - loss: 3.7104e-04 - acc: 0.9999 - val_loss: 0.0012 - val_acc: 0.9998
```

Validation 기준 AUC 스코어 : 1

Submssion 기준 AUC 스코어 : 0.91

결론

- Validation 기준 CNN-LSTM 모델에 비해 성능은 우수

- AUC스코어 기준

- GRU(1) > CNN-LSTM(0.998)

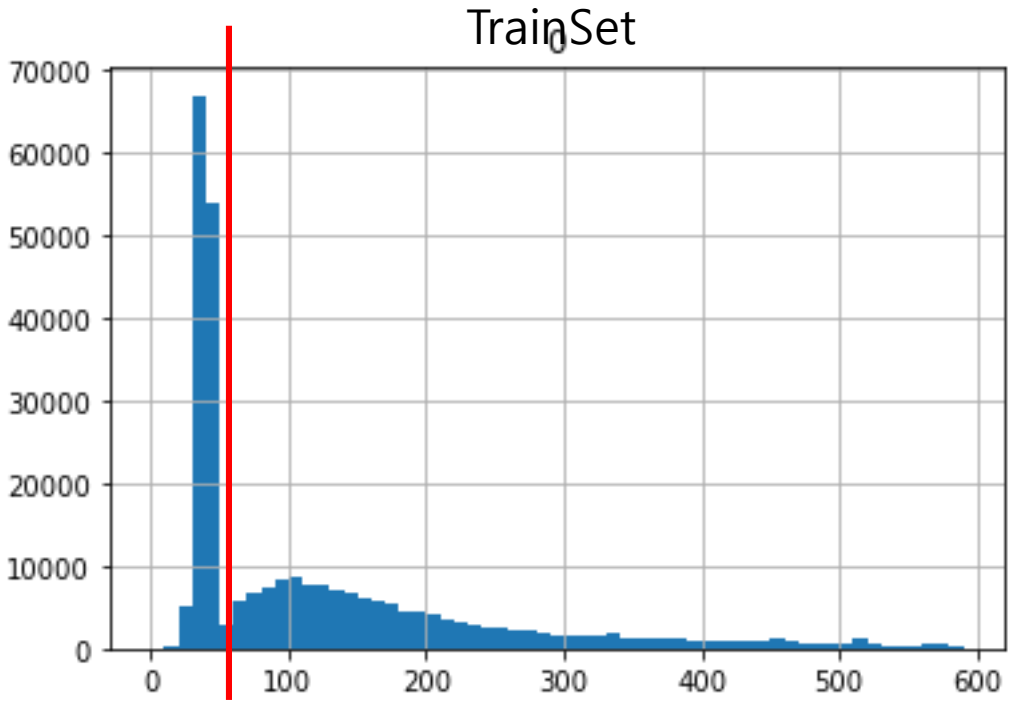
- 하지만 Submission 기준으론 CNN-LSTM 모델에 비해 성능이 뒤쳐진다.

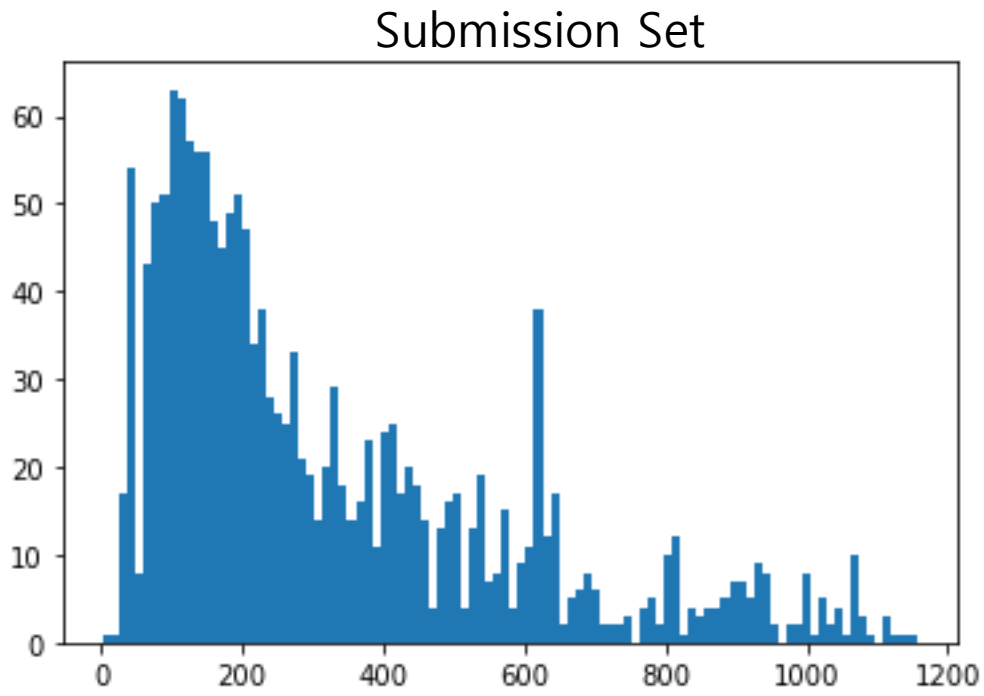
- AUC 스코어 기준

- CNN-LSTM(0.95) > GRU(0.91)

Submission 성능 하락의 이유

- Train 세트와 다른 Submission의 분포





Train Set 분포의 경우와 달리 Submission 세트의 경우 스미싱-비스미싱 구분이 뚜렷하지 않다.

개선방향

- 개선점

- 제로 패딩을 정보로 쓰는 관점의 변화 필요 -> EoS 토큰을 사용하는 역방향 문장투입 고려

- 셀프 어텐션 부분을 Transformer나 BERT와 같은 non-RNN 모델로 교체 가능
- 이를 통해 연산 Cost를 절감 및 추가적인 성능 향상 기대

감사합니다