P2 - Relation Extraction

Wrap up Report

2021/09/28 ~ 2021/10/07

T2020 김다인

boostcamp

목차

I. 프로젝트 목표

이번 프로젝트는 개인적인 측면에서는 다양한 실험과 구현을 통한 성장에 목표를 두었고, 팀의 측면에서는 탐구나 실험의 결과를 원할히 공유하여 모든 팀원들이 현재 팀의 상황을 훤히 알 수 있도록 하는 것에 목표를 두었다.

II. 목표 달성과 상호 발전을 위한 나의 노력

1. Stratified K-fold validation set 구축 및 코드 공유

- 그간 겪었던 캐글 대회들을 통해 좋은 validation set을 구축하는 것이 필수적임을 배울 수 있었다. 특히 지난 P1 마스크 착용 여부 대회에선 팀 내 validation방법이 몇 차례 바뀌어 성능 비교가 힘들었고 일반 1-fold 결과물보다 K-fold set을 통한 OOF prediction 결과물들이 안정적인 성능을 보였다.
- 따라서 이번엔 대회가 시작되자마자 사이킷런 라이브러리를 이용해 Stratified 5-fold validation 이 가능하도록 베이스라인을 고쳤다. 더불어 이를 arg.parser 등의 편의를 위한 기능을 덧붙여 바로 팀 깃헙에 올리고 다른 모든 팀원들과 같은 방법으로 validation set을 구축할 수 있었고, 랜덤시드도 통일해 안정적인 성능 비교가 가능했다.

2. 평가 metric 분석을 통한 인사이트 공유

- 이번 대회의 주된 평가 metric은 micro F-1 score였는데, 개수가 많은 no_relation 클래스(0 번째 클래스)는 제외하고 평가한다고 공지되었다. 팀원들과 이 제외 방식에 대해 여러가지 가정을 하며 토의를 해보았지만 명확한 답은 찾지 못하였다.
- 때문에 나는 평가 metric을 파악하는 것이 매우 중요하다고 판단하여 micro F-1 score에 대해 다시 공부하였다. 이를 통해 no_relation 클래스를 옳게 맞추는 것은 점수에 반영이 되지 않지만 다른 클래스를 0으로 예측해 틀리거나 정답이 0인 클래스를 다른 클래스로 예측하는 경우는 점수에 마이너스가 됨을 알 수 있었다.
- 또한 임의의 데이터셋을 구성해 하나씩 틀려보는 실험을 진행해 일반적인 micro F-1 score와 이번 대회의 micro F-1 score의 차이점을 알 수 있었는데, 가장 큰 차이점은 다른 클래스를 0으로 예측해 깎이는 점수가 정답이 0인 클래스를 다른 클래스로 예측하여 깎이는 점수보다 크다는 것이었다. 이렇게 얻은 인사이트를 정리해 팀원들에게 공유하였고, no_relation 클래스를 제외한 다른 클래스에 가중치를 주는 실험 방법도 고안할 수 있었다.

3. Typed Entity Marker with Punctuation 실험

- *An Improved Baseline for Sentence-level Relation Extraction, 2021* 논문을 통해 Entity의 앞과 뒤를 표시해주고 Type을 표시해주는 Typed Entity marker와 이를 special token을 활용하지 않고 @, #, ^, *로 구현하는 Typed Entity marker (punct)가 효과적임을 알게되어 Typed Entity marker (punct)를 구현해 실험해보았다.

Method	Input Example	$\text{BERT}_{\text{BASE}}$	$\text{BERT}_{\text{LARGE}}$	$\text{RoBERTa}_{\text{LARGE}}$
Entity mask	`[SUBJ-PERSON]` was born in `[OBJ-CITY]`.	69.6	70.6	60.9
Entity marker	`[E1]` Bill `[/E1]` was born in `[E2]` Seattle `[/E2]`.	68.4	69.7	70.7
Entity marker (punct)	@ Bill @ was born in # Seattle #.	68.7	69.8	71.4
Typed entity marker	⟨`S:PERSON`⟩ Bill ⟨`/S:PERSON`⟩ was born in ⟨`O:CITY`⟩ Seattle ⟨`/O:CITY`⟩.	**71.5**	**72.9**	71.0
Typed entity marker (punct)	@ * person * Bill @ was born in # ∧ city ∧ Seattle #.	70.9	72.7	**74.6**

- 실험결과 팀의 최고 점수를 1.5점 정도 갱신해 팀에 기여할 수 있었다. 나는 기존 실험과의 대조를 위해 기존 쿼리문(기본적으로 주어지는 문장의 앞에 붙이는 문장)에는 Entity marker (punct) 기법을 이용했는데, 다른 팀원분께서 쿼리문에도 타입을 추가하고 per, org 등의 타입을 토큰 추가로 끊어지지 않게 구현하여 팀의 최고점을 달성할 수 있었다.

4. 각종 Masking 구현 및 실험

- 처음 Typed Entity marker (punct) 실험을 할 때 한가지 실수를 한 것을 실험 종료 및 리더보드에서 좋은 점수를 받은 후에 알게 되어 당황스러웠다. Train set 중 8개 정도의 문장에 * 기호가 들어가 논문처럼 *을 쓰지 않고 이 역할을 ₩기호가 대신하도록 하였는데, ₩로 둘러쌓인 Subject Entity 부분 전체가 [UNK] 토큰 처리 되어있던 것이다. 그럼에도 점수가 잘 나온 이유가 무엇일지 고민하여 Entity의 Masking이 효과가 있을 것이라 가정하였고, 아래 2가지 실험을 진행하였다.
- 첫번째 실험은 쿼리문과 각종 special token, @, #, ^, * 등의 punct marker 등을 제외하고 15% 확률로 문장의 토큰 일부를 마스킹하는 실험이었다. Fold 내의 Tokenized_trainset을 train.py에서 재가공하는 방식을 택해 fold가 달라질 때마다 같은 문장이라도 다른 방식으로 masking 되도록 구현하였다. 이 실험은 오히려 기존보다 점수가 하락하여 위의 실수처럼 entity 단어만 가려보는 방법을 시도해보기로 했다.
- 두번째 실험은 50% 확률로 Fold내의 Train Data 중 Entity Masking 후보를 정한 후 다시 50%의 확률로 Subject를 가릴 것인지 Object를 가릴 것인지 정하는 방법을 이용하였다. 이는 Masking만 된 단어만 학습하면 inference 과정에서 결국 Masking 이 없는 문장을 못 맞출 수 있을 것이라 생각하였기 때문이다. dataset.py의 tokenized_dataset에 함수를 추가하여 구현하였고, 사용을 원한다면 train.py에서 인자 2개만 추가하면 쓸 수 있게 하였다. 이 실험의 결과, 기존 실수보다는 점수가 올랐지만 실수를 고친 버전을 이길 수는 없었다. 그래도 점수는 높은 편에 속했기에 마지막 앙상블의 재료로 사용할 수 있었다.

5. Focal Loss 적용을 위한 MyTrainer.py 구현

- Hugging face는 편리한 기본 기능을 많이 제공했지만, 기본 기능 내에 들어있지 않은 기능을 구현하기 위해선 그 구조에 대한 공부가 수반되어야 했다. 그 예시 중 하나가 Hugging face의 Trainer가 제공하는 Loss 함수였는데, Label Smoothing Loss 와 Cross Entropy만이 기본 옵션이었다.
- 나는 Data가 Imbalance하여 Focal Loss를 통한 학습에서 좋은 결과를 기대할 수 있을 것이라 판단하였다. 이를 위해 Hugging face Document를 읽고 trainer를 상속받아 MyTrainer.py 를 만들고 Loss.py에 Focal Loss를 넣어두어 다른 팀원들이 다른 커스텀 Loss 함수를 사용하고 싶을 때면 Loss.py에 추가만 하면 쓸 수 있도록 구현했다. 실제로 다른 팀원이 Weighted Cross Entropy를 이 방법을 이용해 구현하여 뿌듯함을 느꼈다.
- 안타깝게도 Focal Loss를 이용한 실험의 결과는 Cross Entropy와 비슷하거나 리더보드 점수가 살짝 낮은 정도였기에 주로 사용하진 않았다. 최종적으론 Loss 함수로 Label Smoothing Loss을 이용했다.

6. 소프트 보팅 툴 구현

- Model의 예측 결과에 원하는 가중치를 주어 soft voting할 수 있는 .py 파일을 만들어 팀 깃헙에 공유하였다. 이를 활용해 5-fold OOF Prediction과 마지막 Soft Voting 파일을 제출할 수 있었다.

III. 결과 및 느낀 점

- 최종적으로 **public 2등, private 2등**이라는 좋은 결과를 얻었다.
- 좋은 팀원들 덕분에 이러한 결과를 얻을 수 있었다고 생각한다. 팀원들과 줌을 활용한 모각공을 통해 지속적으로 실험 상황을 공유하고 더 나은 실험 아이디어를 얻을 수 있어 너무나 행복했다.
- Wandb를 더욱 잘 활용하게 되어 좋았다. 첫 대회 때는 팀 완디비를 개설하지 않았고, 5-fold 학습을 시키면 한 그래프에 5fold 학습 과정이 뭉쳐서 보였는데, 이번엔 팀원들의 실험 과정도 모두 참고하고 각 fold 별 학습도 따로 기록하도록 코드를 구축하여 실험 관리가 용이했다.
- 다양한 실험과 코드 구현을 중점으로 뒀던 이번 목표를 나름대로 잘 이룬 것 같아 기뻤다. 내가 공유한 코드들을 팀원분들이 잘 사용해주셔서 더욱 자신감이 붙은 것 같다.
- Hugging face 라이브러리의 강력함을 깨달을 수 있었다. 원하는 추가 기능의 구현을 위해 Document를 분석하며 실력이 자연스레 늘은 것 같다.

IV. 한계 및 개선해야 할 점

- 최신 논문을 보다 적극적으로 읽고 트렌드를 파악해 모델에 적용하는 것이 필수적이다.
- 4명이 다양한 방법의 Augmentation을 적용했지만 모두 원본을 이기지 못해 결국 Augmentation결과물들은 앙상블의 보조 재료로만 쓰였다. 대회 후반에 꽤나 집중했던 영역인데 결과가 좋지 않아 아쉽다.
- Masking을 적용한 두 실험 모두 Roberta모델의 Pre-training 방식인 epoch마다 masking 방법을 달리하는Dynamic Masking 방법으로 구현하지 못한 점이 아쉽다. 또한 주어진 시간의 문제로 더욱 다양한 마스킹 방법과 마스킹될 확률을 달리하여 실험을 해보지 못한 것 또한 아쉽다.
- 대회 후반에 모델 학습 과정과 결과가 소수점 아래 둘째 자리 차이 정도로 달랐는데 끝내 원인을 찾지 못하였다. 대회 초기에는 모든 소수점과 결과가 동일했기에 아쉬운 부분이었다. 이를 위해 다음 대회에서는 완벽 재현 가능 여부를 체크하고 아예 train data set을 stratify한 5개로 분할해 팀원들에게 배포하겠다고 다짐하였다.