boostcamp AI Tech

Wrap up

▼ 목차

- 💡기술적인 도전
  - 최종 순위
  - EDA 결과
  - 검증(Validation) 전략
  - Model Architecture & Hyper parameters
    - 1. 기본 모델 78.7%
    - 2등 - 77.8%
    - 3등 - 77.0%
    - Tokenizing 방식
  - 앙상블 방법
  - 시도했으나 잘 되지 않았던 것들
- 💡학습과정에서의 교훈
- 💡마주한 한계와 도전 숙제
  - 아쉬웠던 점들
  - 한계/교훈을 바탕으로 다음 스테이지에서 새롭게 시도해볼 것

2주에 걸친 Stage 2 - KLUE(관계 추출, Relation Extraction)이 끝이 났다.

그동안 시도했던 방법들, 아쉽거나 부족했던 점들에 대해 적어보고자 한다.

💡기술적인 도전

최종 순위

- LB - 50 / 135

  Accuracy 79.8%

EDA 결과

- 이번에도 stage 1과 마찬가지로 클래스간 분포가 불균형한 문제가 있었다.

  http://boostcamp.stages.ai/competitions/4/discussion/post/108

- 관계가 없는 데이터가 4400개, 관계가 있는 데이터는 최대 800개에서 최소 1개의 분포를 가지고 있는데, 이것을 어떻게 해결해 줄지에 대해 생각하다가
- 이번 compitition에서는 f1 score가 아니라 accuracy를 기준으로 하기 때문에, 데이터가 작은 클래스에서 모델의 정확도가 크게 중요하지 않다고 생각해서 따로 클래스간 분포를 맞춰주기 위해 `Oversampling`이나 `Undersampling`을 진행하지 않았다.

검증(Validation) 전략

- 적합한 parameter를 찾을 때는, train set : validation set = 8 : 2의 비율로 두고 진행했다.
- 그 중 적합한 parameter와 epoch을 찾은 후, train set : validation set = 10 : 0의 비율로 두고 제출용 모델을 만들었다.
- 제출 기회가 많이 않았다는 것을 감안하면, K-fold validation으로 validation accuracy를 확인 하고 제출하는 것이 나았을 것이라는 생각이 든다.
- Train set과 Validation set을 나눌 때 클래스의 분포를 고려해서 분배했다. (Stratified split)

  ```
  train_test_split(train_dataset, train_label, test_size=args.validation_ratio, random_state = args.seed, stratify=train_label)
  ```

Model Architecture & Hyper parameters

단일 모델 성능 순으로 나열한 순서

- 기본 모델 : `XLM-RoBERTa-LARGE`
- 학습 방법 : Huggingface의 Trainer 사용
- Batch size : 100
- learning rate : 5e-5

1. 기본 모델 78.7%

- 기본 모델에 hyper parameter를 변형하면서 tuning
- Tokenizing : 기본 tokenizing

  `[CLS] (Entity 1) [SEP] (Entity 2) [SEP] (Sentence) [SEP]`
- 여기에 `label smoothing` (76.8%), `20epochs + warmup` (77.2%) 과 같은 traininer hyper parameter를 조정해가며 실험했는데, 추가하는 것보다 추가하지 않는 것이 더 좋은 성능이 있었다.

2등 - 77.8%

- 1등 모델에서 train set : validation set = 8 : 2로 설정하고 학습시킨 결과이다.

3등 - 77.0%

- 9,000개의 기본 제공 데이터 외에 외부 데이터를 학습시킨 모델이다.
- 토론 게시판에 올라와 있는 김규진 캠퍼님의 글 (http://boostcamp.stages.ai/competitions/4/discussion/post/174)을 참고해 약 23만개의 데이터로 학습을 진행했다.
- 1 epoch당 1시간씩 총 5epoch을 학습시켰는데, validation accuracy는 98%가 나왔다.
- 학습 데이터는 많은 데에 비해 learning rate가 높게 설정했기 때문에 overfitting이 되지 않았나 생각한다.
- learning rate를 적게 하는 방향으로 진행했으면, 조금 더 유의미한 결과가 나올 수 있다고 생각한다.
- 1epoch로 진행하면, validation accuracy는 96.2%가 나오고 LB accuracy는 69.7이 나왔다.

Tokenizing 방식

기본 tokenizing 방법 이외에 다른 방법들을 적용해 보았는데, 성능이 좋게 나오지 않았다.

그래도 학습 과정에서 중요한 점이기 때문에, 유의미한 시도였다고 생각한다.

- 기본 Tokenizing : 앞에 Entity에 대한 정보를 부각하는 방식, 2개의 문장을 학습시키는 것처럼 학습

  `[CLS] (Entity 1) [SEP] (Entity 2) [SEP] (Sentence) [SEP]`
- TEM(Typed entity marker) Tokenizing : 조원 캠퍼님의 글 (http://boostcamp.stages.ai/competitions/4/discussion/post/212)과 논문 (https://arxiv.org/pdf/2102.01373.pdf)을 바탕으로 구현했다.

`Pororo` 라이브러리의 NER을 사용해 각 entity에 대해 개체명 정보를 받아와 진행했다.

`[CLS] (Sentence 1) # (α ner(Entity 1) α) # (Sentence 2) @ (β ner(Entity 2) β) @ (Sentence 3) [SEP]`

Epochs 5 validation accuracy : 73%

- TEM(Typed entity marker) Tokenizing New : 기본 TEM에 내가 생각한 부분 구현

  `[CLS] # (α ner(Entity 1) α) # [SEP] @ (β ner(Entity 2) β) @ [SEP] (Sentence) [SEP]`

  TEM 결과 성능이 오르지 않아 entity와 ner에 대한 정보를 강조하기 위해 구현

  하나 sentence에 들어가는 token의 길이를 길게 설정 (100 → 128)

  Epochs 5 validation accuracy : 74.1%

  Epochs 5 validation accuracy (Smoothing): 75.5% ⇒ 이렇게 보면 smoothing이 효과가 있는 것 같은데, 실제 제출에서는 그러지 못했다.

앙상블 방법

- 제출한 submission 기준으로 LB에서 가장 accuracy가 높았던 모델을 ensemble 시도했다.
- 3개, 5개, 가중치를 둔 10개를 모두 시도했다, 그 중 성능이 가장 좋았던 방법은 상위 5개를 앙상블 하는 방법이었다.

  단일 최고 모델 : 78.7 ⇒ 3개 : 79.6, 5개 : 79.8
- 마지막 제출은 Soft voting으로 상위 3개의 모델을 사용했는데, 오히려 hard voting보다 성능이 떨어지는 현상이 나타났다.
- 이 때 3개의 반영 비율은 5:3:2로 진행했는데, 이 부분에서 문제가 발생했을 수도 있고, 다시 모델을 학습시키다 보니 정확한 모델이 아닌 것이 이유일 수도 있다고 생각한다.

  Hard voting 79.6 ⇒ Soft voting 79.1
- 앙상블 결과를 확인하려면 LB에 제출하는 방법밖에 없는데, 그러다보니 제출 시도 횟수를 많이 소모하는 것 같다.

시도했으나 잘 되지 않았던 것들

- 많은 데이터를 적용한 결과 (약 23만개) 성능이 좋아지지 않았다. 데이터에서 발생한 결과가 아니라 데이터를 활용을 못한 것 같다.
- 기본 Tokenizing하는 방법 이외에 다른 방법들을 사용해 보았는데, 좋은 성능을 얻지 못했다.

- `TEM` 등 여러 방법을 적용했는데, 캠퍼들 마다 성능 향상을 경험한 케이스도 있고, 그렇지 않은 케이스도 있는 것 같다.
- 구현 상의 문제인지 아니면 저 방법이 모델에 적합하지 않았던 것인지 생각해볼 필요가 있는 것 같다.

💡 학습과정에서의 교훈

- 토론 게시판에서 얻은 팁들도 있고, 스스로 생각했던 방법도 몇 개 적용해 보았는데, 모두 기본 모델보다 성능이 안좋게 나와서 기대했던 결과를 얻지 못했다.
- 기본 모델에서 80%의 성능을 내는 것을 목표로 했는데 그렇지 못했기 때문에 ensemble을 진행해도 좋은 성능을 얻지 못했다.
- 마스터님께서 5일 강의 이후에 compitition에 참가하는 것을 권장하셔서 그렇게 했는데, 초반에 코드 오류 수정과 `BERT-multilingual` 모델에서 `XLM-RoBERTa-LARGE` 모델로 변경하는 과정에서 하루정도 해메다 보니 시간이 여유롭지 못했다.
- 코드는 미리미리 봐두자

💡 마주한 한계와 도전 숙제

아쉬웠던 점들

- 부스트캠프를 하면서 많은 피어세션을 하고, 많은 사람들을 만났는데, 그 중 최악의 피어세션의 2주였다고 생각한다.
- 피어세션 하는 시간이 아깝다고 생각될 정도였다. 매번 참가하지 않는 사람도 있었고, 캠과 마이크도 항상 꺼져있는 사람도 있었다.
- 그러면선도 리더보드에는 상위권에 있는 사람이 있었는데, 피어세션에 참가하지도 않고 compitition만 할거면 차라리 캐글이나 나가지 왜 부캠에 들어왔는지 모르겠다.
- 랜덤 배정되는 동안 한번도 이런 적이 없었는데, 2주간 진행되는 피어세션에서 이렇게 배정이 되다 보니 많이 아쉬웠고, 학습 과정에서 도움이 되지 못했다.
- Stage 1에서 괜찮은 성능을 얻은 데에는 피어세션에서의 팁들과 토론이 많은 도움이 되었다고 생각하는데, 이런 방법에서는 도움이 전혀 되지 못한다.

한계/교훈을 바탕으로 다음 스테이지에서 새롭게 시도해볼 것

~~저번 Stage 1 Wrap up에서도 적었던 내용인데,,, 한번 더~~

- 토론 게시판에 글 쓰기
- `nni`, `wandb` 적용해보기

  김봉진 캠퍼님의 글 (http://boostcamp.stages.ai/competitions/4/discussion/post/204) 참고
- 새로운 내용 찾아서 시도해보기
- 학습 정리 잘하고, 실험 내용 기록 잘하기 (마지막에 정리하기 어렵다)
