LSTM을 활용한 난독화된 한글 리뷰 복원

2025.3.1

윤희찬, 장덕재, 조태연

문제 정의

문제 정의

1. 기존 토크나이저의 문제점

형태소 분석기, BPE 기반 토크나이저 등이 난독화된 텍스트를 잘 분해하지 못함.

ex) 고유명사, 구어체 및 비표준 표현 처리 등이 어려움,

2. 생성형 모델 vs 고전적 방법

초기 시도에는 Transformer 기반 생성 모델이 유망해 보였으나,

한계를 보임.

기존 방법론 분석

기존 방법론 분석

- 처음 사용한 방법: 생성형 모델(Llama3, DeepSeek)

  트랜스포머 기반 LLM 사용 (Llama3, DeepSeek 활용)

  기대: 문맥을 이해하고 자연스러운 문장 생성

  문제점:

  1. 노이징을 해석하는 능력이 부족
     - 트랜스포머는 문맥 이해에 강하지만, 문자 수준의 변형에 취약
     - ex) “쫏쉭닿깝쥐” -> “조식당까지”를 정확히 매핑하기 힘듦.
  2. 추론 속도 & 계산량 문제
     - LLM은 복잡한 연산을 수행하여 과도한 연산 비용 발생.
     - 파라미터 튜닝시, 많은 시간 소요

기존 방법론 분석

- 처음 사용한 방법: 생성형 모델(LLAMA3, DeepSeek)

  3. Query Counting 문제
  - 참고 논문에서는 Transformer가 특정 단어가 몇 번 등장했는지를 세는 문제(Query Counting)에 약하다는 점을 보임. 이를 통해 Transformer가 정확한 토큰 빈도를 추적하는 것이 어렵다는 점을 추론할 수 있음.
  - 리뷰 복원 작업에서는 글자 수를 고정하고, 유사한 패턴을 기반으로 학습해야하지만, 생성 모델은 이를 학습하기 어려움.

참고 논문:
Yehudai, G., Kaplan, H., Ghandeharioun, A., Geva, M., & Globerson, A. (2024). When Can Transformers Count to n?
Retrieved from https://arxiv.org/abs/2407.15160

기존 방법론 분석

- 기존 형태소 분석기(Kiwi)
  - 본 문장: 주변이 조용해서 좋았고 이 정도 상태면 정말 훌륭했습니다.
  - 형태소 분석 결과:
    ['주변', '이', '조용', '하', '어서', '좋', '았', '고', '이', '정도', '상태', '이', '면', '정말',
    '훌륭', '하', '었', '습니다', '.']
  - 난독화된 문장: 죽펴닛 쪼옹헤셔 좋앝교 임 쩡뚜 샹탬먼 졍말 휼룽햇숲뉘댜.
  - 형태소 분석 결과:
    ['죽펴닛', '쪼옹헤셔', '좋앝교', '임', '쩡뚜', '샹탬먼', '졍말', '휼룽햇숲뉘댜', '.']
  - 일반 문장은 정확히 분해하지만, 난독화된 문장은 형태소 분석기에 없는 단어(OOV)로 처리되어 제대로 분석되지 않음.
  - 기존 토크나이저는 난독화 문장 복원에 최적화되지 않았으며, 새로운 토크나이저가 필요.

해결 방향

해결 방향

- 왜 커스텀 토크나이저가 필요한가?
  - 기존 토크나이저들은 형태소 분석에 집중 -> 난독화된 문장은 형태소 단위로 복원 불가
  - 데이터 특성: 입출력 음절 수 동일 및 글자 위치 동일
  - 커스텀 토크나이저: 음절 단위 & Noise-aware Tokenizer
- 왜 LSTM을 선택했는가?
  - Transformer보다 계산량이 현저히 적음 -> 훈련 시간 감소
  - 음절 시퀀스에 대한 강한 학습 능력 -> 글자 수, 반복 패턴, 변형된 형태 복원에 유리
  - 문맥을 길게 유지할 필요가 적음 -> 복잡한 Attention 구조가 필요하지 않음
  - 최종적으로, LSTM이 트랜스포머 기반 모델보다 성능이 높았음.

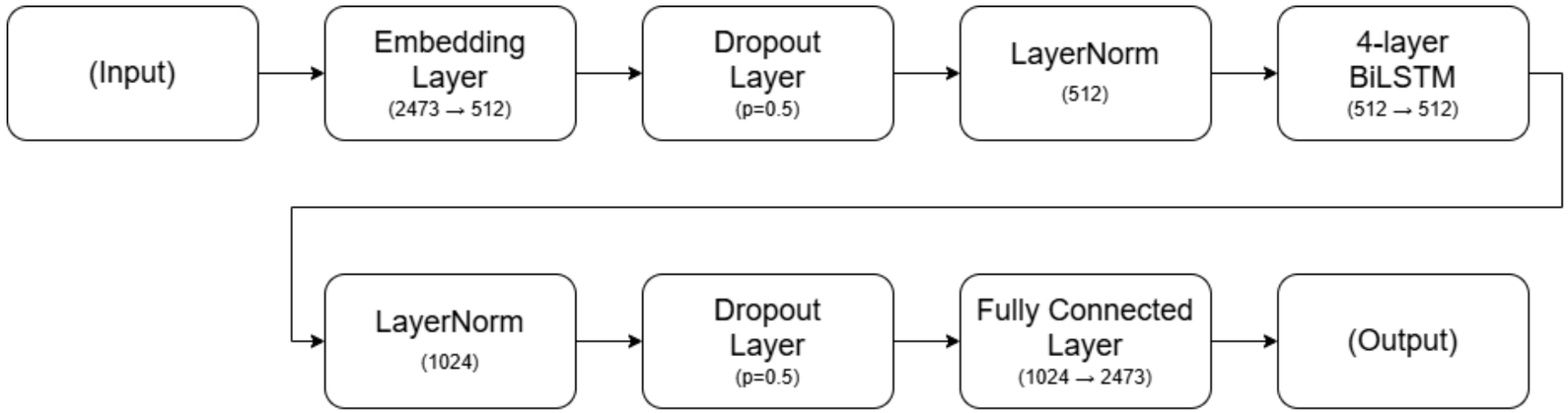
최적화 과정 & 실험결과

최적화 과정 & 실험결과

- Baseline 모델 성능 비교
  - LSTM 파라미터 수: 25 M
  - DeepSeek-R1-Distill-Qwen-1.5B 파라미터 수: 1.5 B
  - Llama 3 8B 파라미터 수: 8 B
  - Gemma 2 파라미터 수: 2 B
- 최소 80배에서 최대 320배 적은 파라미터 사용
  - LSTM 모델이 복원 정확도에서 생성 모델을 웃돌았음
  - LSTM이 상대적으로 가벼운 연산량으로도 효과적인 성능을 낼 수 있음을 의미
  - 대형 모델 대비 자원 효율성이 매우 뛰어남

최적화 과정 & 실험결과

- LSTM 모델 개선 과정
  - BiLSTM 도입 → 좌우 문맥 활용
  - Pre-trained Word2Vec 임베딩 활용
  - Layer Normalization & Dropout 적용
- 모델 형태 시각화



최적화 과정 & 실험결과

- 데이터 증강
  - 난독화된 문장과 본 문장을 50% 비율로 섞어서 학습
  - ex) input: 잚많ᆭ 쟉꼬 갉 태 좋눼욥. 차못동 줆 ㅋ

    output: 잠만 자고 갈 때 좋네요. 잠옷도 줌 ㅋ

    input: 편안히 잘 쉬고 왔습니다.

    output: 펀냔휜 잘 쉭곶 왔쑵닝따.

최적화 과정 & 실험결과

- 최종 결과
  - [입력] 윌뱐 잎츔민든릿 샤있샤윔엡 위썬 호뗄첨렴 관뤽갉 찰 앉 뙨는 누뀜뮈넬오.
    [예측] 일반 입주민들이 사이사이에 있어 호텔처럼 관리가 잘 안 되는 느낌이네요.
  - [입력] 쾅얀땜고카 잚 뵤없 쑥쇼엔썬 편난학께 쀼룰 캄쌍할 쑤 위써셔 좋앝숩니탸.
    [예측] 광안대교가 잘 보여 숙소에서 편안하게 뷰를 감상할 수 있어서 좋았습니다.
  - [입력] 뺨튈륨 괜찬았옅욤! 쿄테끼량 둥랐임깆또 인셔셔 좋았서온.
    [예측] 파티룸 괜찮았어요! 고데기랑 드라이기도 있어서 좋았어요.

결론

결론

- 단순히 크고 좋다고 알려진 모델만이 정답이 아닐 수 있음.
  - 과도한 컴퓨팅 자원 대신 데이터 특성에 최적화된 솔루션 적용.
  - LSTM + 음절 단위 토크나이저가 문제에 최적화된 솔루션임을 보임.
  - 경량 모델 + 데이터 특화 기법으로 최적 성능을 달성.

thank you!

https://github.com/quant-jason/NKTL-Noised-Korean-Translator-by-LSTM