LangCon 2021

한국어 토크나이징의 현재와 미래

Tokyo Institute of Technology I 정보이공학원 문상환

토크나이징의 정의 및 한국어 적용

LangCon 2021

토크나이징이란?

- 토크나이징이란 문자열을 특정 규칙에 의해 "토큰"으로 분절하는 여러 기법들로, **자연어 처리에서 알고리즘 입력 직전에 반드시 이루어지는 필수적인 과정**
- **문장을 단어로 자르는것이 가장 간단한 방식**

  e.g. Hello world → Hello | world
- "토큰"은 모델 등 알고리즘에서 처리를 위한 **기본 입력 단위**
  - 길이나 경계는 알고리즘에 따라 다르나 전통적으로 대부분 언어는 (e.g. 영어) 공백, 개행, 그리고 구두점을 경계로 하고, 이에 따라 **구현이 간단**
  - Scriptio Continua인 언어들은 (e.g. CJK) 형태소 등의 단위로 분절하는 알고리즘을 이용, 따라서 **데이터셋이 필요하고 구현 또한 복잡**
  - Subword 이전 **한국어의 de-facto 표준은 형태소 기반 분절**
- **전이학습에 있어 전이 학습 후 최종 평가 성능을 좌우하는 첫번째 요소**
  - 다국어BERT(mBERT)가 한국어에 애매한 모델로 평가 절하 된 근본적 원인이기도 함

토큰의 단위와 장단점

- **한국어의 경우 토큰은 어절, 형태소, 서브워드, 음절, 또는 그 보다 작은 단위로 분절하여 사용 가능**
- 표현 능력 (Representation Robustness), 토큰의 길이, OOV에 대한 안전성 **3가지 측면에서 균형이 필요**
- 표현 능력은 의미 경계가 (e.g. 형태소) 우수한 성향을 보이나 형태소 기반 토크나이징이 완벽하지 않아 애매한 representation을 학습할 위험이 있음
- **토큰의 길이와 OOV에 대한 안전성은 반비례 양상을 가짐**
  - 토큰을 문자나 서브워드 레벨로 둘 경우 OOV에 대한 민감도는 줄어듬
    - 당연히 표현이 갖는 **내용이 줄고, 어텐션(Context)에 대한 의존도가 높아짐**
    - 이에 따라 **컨텍스트 없는 임베딩이 가지는 의미는 거의 없어짐**
  - 길이를 늘릴수록 OOV 빈도가 올라감 (특히 한국어의 경우)

단위별 알고리즘

- **어절:** `input_string.split()`
- **형태소:** 꼬꼬마, 한나눔, 코모란, Okt, MeCab-ko, Khaiii 등
  - 유저 사전 커스터마이징이 수월하여 MeCab 사용을 현재도 많이 함
- **서브워드:** Subword-NMT/BPE (Sennrich et al., 2015), HuggingFace Tokenizers, fastBPE, YouTokenToMe, WordPiece (Wu et al., 2016), SentencePiece (Kudo and Richardson., 2018) 등
  - Pre-tokenization시, 방식에 따라 단어나 형태소 중간에 잘리는 경우 (e.g. 신조어) 서브워드의 표현력이 떨어짐
- **음절:** `list(input_string)`
- **그 외:** BBPE (Radford et al., 2019), FastText (Bojanowski et al., 2016), Byte (Gillick et al., 2016) 등
  - 완전한 유니코드 "문자" 이하의 단위 또는 애매한 경우

사전 분절

- **서브워드 기반 기법의 경우, 사전 토크나이징 (pre-tokenization) 또는 전처리를 (pre-processing) 본 알고리즘 학습 전에 수행하는 경우가 있음**
- BERT (Devlin et al., 2018) Tokenizer의 경우, **어절 단위 분절 후에 WordPiece 알고리즘 적용**
  - 중국어, 일본어의 경우 한자는 문자 단위로 토크나이징하는 편법 사용 (논문에는 기재 안된 내용)
- 일본어에서는 MeCab (Kudo, 2015), JUMAN++ (Tolmachev et al., 2018), Kytea 등 형태소 분석기로 **사전 분절 후 서브워드 학습이 일반적**
- 일본어와 같이 형태소 기반 **사전 분절 기법이 한국어에서도 효과적**인 것으로 확인 (Park et al., 2020)
- 전처리로 **자모 단위 후 BPE를 학습하는 기법** 또한 제안된 바 있음 (Moon and Okazaki, 2020)

한국어 토크나이징의 어려움

Language Conference

- **한국어는 여러모로 NLP에 있어서 어려운 언어**
- 띄어쓰기가 어려워서 대부분의 사용자들은 **옳은 띄어쓰기를 포기했고**
  - 이로 인해 pre-tokenization을 공백으로 하면 애매한 결과가 나온다
- 맞춤법도 어려워서 위키나 뉴스가 아닌 이상 **맞춤법 오류는 있다고 가정을 해야 하며**
  - 일부러 맞춤법을 틀려주시는 분들도 계신다 (e.g. 덧글)
- 결국 의미있는 단위로 자르려면 **형태소 분석기가 필요함**
  - 허나 이 또한 오래된 코퍼스로 학습되어 신조어와 구어식 표현에 취약
- 완성형 한글 인코딩을 택한 덕분에 **"글자수" 또한 어마무시하게 많음**
  - 한글의 alphabetic한 특성을 서브워드 기반 기법들이 제대로 보지 못함
  - 서브워드 알고리즘이 보기엔 (인명 제외시) 중국어보다 더 어려운 언어

국가표준 MeCab과 21세기 세종 계획

- 원래 일본어 형태소 분석기였던 MeCab을 약간 변형한 Mecab-ko가 현재도 가장 일반적으로 사용
- **세종 코퍼스를 (21세기 세종 계획) 이용하여 학습**
  - 일부를 사용한것으로 추정되나 확인 불가 (아시는 분 있으면 연락을)
  - 2018년 7월 최신 모델 출시가 마지막
- **단, 세종 코퍼스는 수정/재배포가 엄격히 금지되어 있음**
  - 이에 따라 데이터셋이 최초 배포 시점 이후로 개선이 없음
  - 이에 따라 신조어/새로운 인명 대응이 어려움
  - 모두의 말뭉치 또한 유사한 라이센스 제약이 있음 (개선해주세요!)
  - 해당 라이센스 문제로 형태소 기반 토크나이징의 발전이 정체된 상태

(개인적인) 한국어 토크나이징 연구 소개

LangCon 2021

자모 기반 서브워드, Jamo Pair Encoding

- Moon & Okazaki, 2020 (LREC 2020)
- 인구 언어들은 다 알파벳 단위로 서브워드를 학습하는데 **왜 한글은 알파벳이 (자모) 아닌 음절을 기반으로 서브워드를 학습하는가에 의문**을 가짐
- **KSC5601-1987로 2음절 서브워드 구성시 총 11,045,000 서브워드**
  - 하지만 쏭돈까스나 똠방각하는 표현 불가
- 서브워드 학습 (Unigram LM) 전에 **자모 단위로 분해 후 학습, 생성의 경우 자모 단위로 출력된 것을 다시 후처리**
- 기존에도 자모 기반 표현을 제안한 바 있으나 (Park et al., 2018, Stratos., 2017) **출력이 자모 단위로 나올 경는 (생성) 대응이 안되는 제약이 있음**
- **입력기 오토마타를 응용**하여 생성 문제에 대한 해결책을 포함하여 제안
  - 사람이 입력하는것처럼 하면 되겠지라는 다소 안이한 생각
  - 출력 하나 잘못 나오면 술 취한 사람이 백스페이스 없이 눈 감고 친거처럼 결과가 나옴

모르는 단어 나중에 배우기, PatchBERT

- Moon & Okazaki, 2020 (EMNLP 2020)
  Moon & Okazaki, 2021 (Journal of Information Processing, Vol. 29)
- 기 학습 모델 단어장에서 OOV 발생시 **기존 사전을 보강하는 방법론**
- 최종 태스크 파인튜닝 직전에 **"모르는 단어 학습" 단계 추가**
- 사실은 mBERT 성능이 너무 안나와서 들여다본 **우연의 성과물**
  - mBERT에서 "동해물과 백두산이 마르고 닳도록"에서 UNK가 나오는 황당한 현상에서 시작
- 모르는 단어 파악 후, 기존 단어장에 존재하는 **다른 서브워드의 임베딩과 weight sharing을 하거나 덮어쓰는 식**
- 심각하게 망가진 모델도 어느 정도 복구 가능한 것 또한 확인
  - 랜덤으로 단어장에서 단어 제거 후 성능 복구 실험
  - (사족: NSMC 태스크의 난이도에 대해서 다소 의구심이 들기는 했습니다)

한국어 토크나이징의 미래

LangCon 2021

오픈 형태소 분석 데이터셋, OpenKorpos

- (졸업 논문과 무관하게) **현재 진행중인 연구** (ACL 2021에서 고배를...)
  - 공범들은 당 행사 관계자들 안에 있습니다 (Anonymity 풀리면 알게 될겁니다)
- **기존 형태소 말뭉치의 라이센스 제약에 대한 근본적인 해결책을 제안**
- **기 학습 형태소 분석기의 Consensus를 이용, 기계 태깅하여 제작**
  - 현재 (평가셋으로 사용할) 일부를 수작업 평가 진행중
  - 5천5백만 어절 규모로, 전량 수동 평가는 불가
- 수정을 자유롭게 할 수 있도록 **라이센스 문제가 없는 원문만 사용**
- **저작권에 대해 자유로운 문장을 투고할 수 있고, 문제점들을 Github PR로 수정할 수 있도록 오픈소스 프로젝트로 Living Corpus를 지향**
  - 특정 스냅샷 기준으로 성능 평가가 비교 가능하도록 할 예정
- 조만간에 공개 예정입니다

형태소 분석 태스크를 분리해서 본다면?

- 형태소 분석용 말뭉치를 만드는것은 공수가 많이 들어가는데, **분절과 POS 태깅을 같이 해야만 하는 특성 때문**
- **특히 POS태깅은 일정 수준 이상의 국어학 지식이 필요**하고, 이에 따라 작업 가능한 사람이 제한적 (e.g. JKG와 JKO의 차이는?)
- 그러나 **현실에서 형태소 분석기를 사용할 경우 분절된 형태소만 사용하고 POS 태그는 버려지는 경우가 대부분** (심지어 대부분은 quasi-형태소)
- 형태소 분절만 하는 데이터셋 / 모델을 만드는것을 목표로 하면 조금 더 저렴하고 효율적으로 만들 수 있을까에 대한 고민을 해볼 시기
  - 데이터와 모델 규모가 주력인 동향에 따라갈 필요성에 대한 고찰 필요
  - 어노테이션 툴링과 데이터셋의 "완성"의 정의 또한 바뀌어야
  - 완벽한 형태소가 (lemma포함) 필요한 경우는 언제인지 생각해볼 시간

신조어에 대한 대응

- 일본어의 경우 neologd라는 프로젝트가 있으며, **신조어 학습을 하기 위한 원부 데이터와 신조어 기학습 사전으로 제공하는 오픈소스 프로젝트**
- 비교적 최근까지 (2020/9) 업데이트가 되고 있고 업계에서도 꽤 사용되고 있음
- **원부 (seed) 데이터가 Github에 압축된 형태로 올라가 있어 PR을 보내는게 불가하여, 외부 패치 기여가 어려움**
  - 현재는 Github Issues를 통해 오류 제보를 받는 식
  - 프로젝트 오너가 사실상 1명이라 바쁘면 진행이 멈추는 문제가 있음 - 프로세스 비효율성이 도움이 되지는 않는 듯
- **한국어 NLP에서도 유사한 프로젝트가 필요할것으로 생각**
  - 일본판의 프로세스 문제를 해결해서 접근할 필요가 있어 보임
  - 관심 있으신 분 있으면 연락 주세요

두서 없는 다른 아이디어들

- Subword Regularization이 한국어에 적용할 때와 하지 않았을 때의 차이는?
- Differentiable Tokenization을 한국어에 적용한다면?
- JUMAN++ 같이 MeCab보다 좋은 형태소 분석기를 한국어 데이터셋에 학습을 시도해본다면?
- 자모 기반 서브워드에서 형태소 비스무레한게 나오는 현상을 보았으나 현재는 이야기하기에 이르나, 좀 더 들여다 볼 필요가 있을지?
- 자모 기반 서브워드를 생성 모델에서 사용시, 다음 서브워드가 잘못 나오는 경우에 대한 대책은?
  - 이는 Byte 단위에서도 같은 이슈가 있음 (Unicode orphan 문제)
- Hash embedding을 (Svenstrup etc) 이용한 접근에 대해서 시도를 해볼 필요가 있을까?

긴 발표 들어주셔서 감사합니다!

(행사 후 질문은 sangwhan@iki.fi 또는 트위터 @sangwhanmoon)

LangCon 2021