프로젝트 수행 필요 기술
: 자연어처리

@ 자연어 처리

- 컴퓨터를 이용하여 인간 언어의 이해, 생성 및 분석을 다루는 인공지능 기술
- 자연어 처리 = 특징 찾기 = 분류 = 그래프 위에 표현 = 벡터화

@ 자연어 처리 종류

1. 규칙/지식 기반 접근법(Symbolic approach)
- 현업에서 가장 많이 사용하고, 인간의 문법이나 사용하는 기호의 형식을 코드로 바꾼것
2. 확률/통계 기반 접근법(Statistical approach)
- tf-idf를 활용하는 접근법

@ 자연어 처리 방법

1. One-hot encoding
- 장점 : 가장 단순하고 쉽게 표현 가능
- 단점 : 차원 지옥에 빠질 수 있음, 단어의 의미를 벡터 공간을 표현 불가능
-
2. Word2Vec
- 정의 : 한단어의 주변 단어들을 통해, 그 단어의 의미를 파악
- 장점 : 한정된 차원으로 표현 가능, 비지도 학습 가능, 벡터 연산 가능
- 단점 : subword information 무시, OOV에서 적용 불가능, 문맥 고려 불가능
-
3. FastText
- 정의 : n-gram으로 단어를 분리 한 뒤, 모든 n-gram vector를 합산한 후 평균을 통해 단어 벡터를 획득
- 장점 : Word2Vec의 단점을 보완한 모델
- 단점: 문맥 고려 불가능

@ 자연어 모델의 발전

기존 word embedding 방식 : 주변 단어로 학습 ~> 문맥 고려 x
⇒ 언어모델

언어모델

@ 모델
: 연구 대상 주제를 도면, 사진 사용 or 수식, 기호로 표현.
즉, 자연 법칙을 컴퓨터로 모사함으로써 시뮬레이션 가능

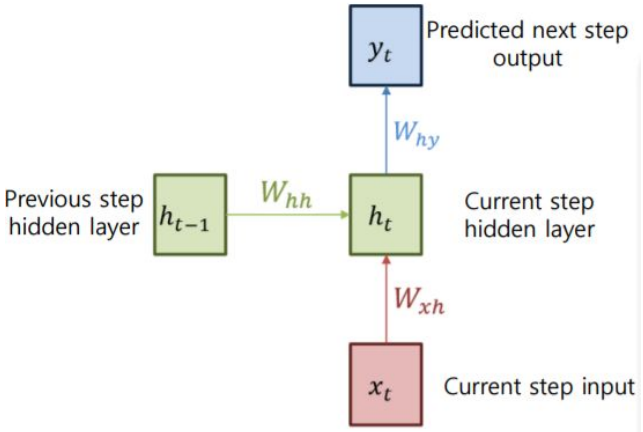
* 언어모델
` 학습 방향
- 주어진 단어들로부터 다음 등장 단어의 확률을 예측

=> 예측 good ~> 언어의 특성이 잘 반영된 모델

* Markov 확률 기반 언어 모델 : Markov Chain Model
` 학습방법
- 문장들에서 각 단어마다 다음 단어의 확률 학습 ~> 확률을 최대로 하도록 학습
- Markov 테이블 작성
- 같은 방식으로 문장들의 확률도 계산 가능
` 예시
- p.42

* Recurrent Neural Network(RNN) 기반의 언어 모델
` 학습방법
- 현재 단계에서 다음 단계를 예측하는 과정에서, 이전 단계의 정보를 반영함.

기본적인 RNN의 구조



` 예시
- p.43
` 출력결과
- 앞선 단어들의 '문맥' 고려 ~> 마지막 출력
- Context vector 형성 가능
- Context vector + classification layer ⇒ 예측 모델
` 구조적 문제점
- 입력 sequence 길이가 매우 긴 경우 ~> 처음 token 정보 ↓
- 중요 x 토큰도 중요 토큰과 같은 영향력

* Attention 모델
` 배경
- 중요 feature은 더욱 중요하게 고려
` 학습단계
1. 기존 context vector + 각 step의 output
2. Feed Forward Fully Connected Layer에 입력 + softmax
3. 각 단계별 weight 출력 (중요도)
4. 각 step의 output * weight => 최종 context vector
5. 예측 결과 bad => Feed Forward Fully Connected Layer & weight 수정
6. 반복
` 장점
- 문맥에 따라 동적으로 할당되는 dynamic context vector 획득

- Attention 시각화를 통해 설명력 ↑
- p. 58

` 단점
- RNN을 순차적으로 연산 => 느림

* Self-attention 모델
` 배경
- 결국 필요한건 각 단어의 중요도(weight) 파악
- RNN 과정을 생략하고

` 학습과정
- p.65 ~ p.68

* Multi-head Self Attention 모델 (Bert 영역)
- Query, Key, Value로 구성된 attention layer을 동시에 여러 개 수행

* Transformer 모델
- multi-head attention으로 이루어진 encoder를 여러 층 쌓아서 encoding을 수행

@ Bert

1. Bert 란

- Bert 는 이전 단계를 보고 다음 단어를 예측하는 형태가 변형된 bi-directional Transformer로 이루어진 언어모델
- 잘 만들어진 Bert 언어모델 위에 1개의 classification layer만 부착하여 다양한 NLP task 수행
-

2. 처리 순서

- BPE 알고리즘으로 WordPiece tokenizing 수행
- input된 token을 일정 확률로 masking처리
- 주변 단어들을 보고 masked 단어를 예측
- 문장들 사이의 관계를 학습하기 위해 '다음 문장' 이라는 label을 추가

2-1. Task 1

- 15 %dml [MASK] token을 만들어 냄
  - 80%의 경우 : token을 [MASK]로 바꿈
  - 10%의 경우 : token을 random word로 바꿈
  - 10%의 경우 : token을 원래의 단어 그대로 놔둬, 관측단어와 비교

2-2. Task 2

- corpus에서 두 문장을 이어 붙여 이것이 원래의 corpus에서 바로 이어 붙여져 있던 문장인지를 맞추는 binarized next sentence prediction task를 수행
  - 50% : sentence A, B가 실제 next sentence
  - 50% : sentence A, B가 corpus에서 random으로 뽑힌(관계가 없는) 두 문장

@KorBert

1. Korbert

- 한국어 bert
- tokenizing 이 기존의 bert와 조금 다름 형태소 분석을 한 다음 줌

- wordpiece tokenizing 이전에 형태소 태그를 다 붙인 다음 wordpiece를 타게 되고 텍스트가 더 쪼개짐
- 동음이의어인 경우 다른 vocab이라고 판단 가능
- KorBERT를 사용하기 위해선 형분석기를 사용해, 형분석 이후 데이터가 입력되야 함
- 공개된 형분석기의 경우 Kakao의 Khaii의 형분석기가 있음

서브 워드 분리 작업이란?
하나의 단어를 여러 서브워드로 분리해서 단어를 인코딩 및 임베딩하겠다는 의도를 가진 전처리 작업
이를 통해 OOV나 희귀 단어, 신조어와 같은 문제를 완화 (영어권 언어나 한국어의 경우 어느정도 의미있는 단위로 나누는 것이 가능)

BPE (Btye Pari Encoding) 란?
빈도수가 많은 글자의 쌍을 찾아서 하나로 병합하는 방식을 수행하는 서브 워드 분리 알고리즘 (기존에 있던 단어를 분리한다는 의미)

- 기계에 아무리 많은 단어를 학습 시켜도 세상의 모든 단어를 알려줄 수 없음
- 기계가 모르는 단어가 등장하면 그 단어를 단어 집합에 없는 단어란 의미로 OOV(Out-Of-Vocabulary) 또는 UNK(Unknown Token)라고 표현
- 이를 위해 서브워드 분리 작업을 함

참고자료 : T 아카데미 자연어 언어 모델 'Bert' 강의 자료.