자연어처리(NLP) 핵심 개념 정리

1. 자연어처리(NLP)의 정의

- 자연어(한국어, 영어 등)의 이해, 생성 및 분석을 다루는 인공지능 연구 분야
- 자연어 이해(NLU): 형태/통사/의미 분석을 통해 자연어를 컴퓨터가 이해할 수 있는 형태로 변환
  - 감정 분석, 개체 독해, 상식 추론, 의미론적 유사도 측정 등
- 자연어 생성(NLG): 컴퓨터가 처리한 결과물을 텍스트나 음성으로 변환
  - 자동 완성, 스토리 생성, 생성형 언어 모델, 캡션 생성 등
- 교집합: 생성형 문서 요약, 생성형 질의응답, E2E 챗봇 등

2. 통계적 언어 모델과 N-gram

- 통계적 언어 모델: 자연어 문장이나 단어 시퀀스에 확률을 할당하는 모델
- N-gram: 연속된 N개 항목(단어, 문자, 음절)을 하나의 단위로 처리
  - Unigram(1-gram): 단일 단어/문자만 고려 (예: "나", "는", "밥")
  - Bigram(2-gram): 연속된 2개 단어/문자 (예: "나는", "는 밥", "밥을")
  - Trigram(3-gram): 연속된 3개 단어/문자 (예: "나는 밥", "는 밥을", "밥을 먹는다")

3. 로그 확률 사용 이유

- 수치적 안정성: 매우 작은 확률값들의 곱셈 시 언더플로우 방지
  - $P(A) \times P(B) \times P(C) \rightarrow \log(P(A)) + \log(P(B)) + \log(P(C))$
- 계산 효율성: 곱셈이 덧셈으로 변환되어 계산 효율 향상
- 확률값 비교: 매우 작은 확률값들의 차이가 로그 스케일에서 더 명확하게 드러남
- 학습 안정성: 딥러닝의 cross-entropy 손실 함수에서 안정적 학습 가능

4. 라플라스 스무딩(Laplace Smoothing)

- 목적: N-gram의 Zero 확률 문제(희소성 문제) 해결
- 방법: 모든 가능한 N-gram 조합에 1을 더해 등장하지 않은 조합에도 최소 확률 부여
- 수식:
  - 기존: $P(w_n \mid w_1, ..., w_{n-1}) = \text{Count}(w_1, ..., w_n) / \text{Count}(w_1, ..., w_{n-1})$
  - 스무딩 적용: $P(w_n \mid w_1, ..., w_{n-1}) = (\text{Count}(w_1, ..., w_n) + 1) / (\text{Count}(w_1, ..., w_{n-1}) + V)$

    - V는 전체 어휘 개수
- 한계: 어휘 수가 클수록 실제 등장 조합의 확률이 과도하게 낮아질 수 있음

5. 마르코프 가정(Markov Assumption)

- 정의: "현재 상태(단어)는 오직 바로 앞의 n-1개 상태(단어)에만 의존한다"는 가정
- 이유:
    - 계산의 단순화: 모든 이전 단어 고려 시 계산량 기하급수적 증가 방지
    - 데이터 희소성 문제 완화: 필요한 데이터양 감소
    - 모델 학습 및 구현 용이
- 한계:
    - 장기 의존성(Long-term dependency) 무시
    - 차원의 저주: n이 커질수록 파라미터가 기하급수적으로 증가

6. 한국어 자연어처리의 난이도

- 교착어적 특성: 조사와 어미가 풍부하여 어근에 다양한 접사 결합
- 형태소의 복잡성: 불규칙 활용, 음운 변화 다양
- 높은 문맥 의존성: 주어나 목적어 생략이 빈번
- 어순의 상대적 자유로움: 조사 사용으로 어순 변경 자유
- 어휘 희소성 문제: 형태소 결합으로 생성 가능한 단어 형태가 매우 다양
- 형태소 분석의 복잡성: 전처리 과정에서 오류 발생 가능성 높음
- OOV(Out-of-Vocabulary) 문제 심각: 조사/어미의 다양한 결합으로 미등장 단어 빈번

7. 임베딩(Embedding)과 BoW(Bag of Words)

- 임베딩: 텍스트 데이터를 고정된 크기의 실수 벡터로 변환하는 방법
    - 고차원→저차원 변환, 의미적 유사성 반영
- BoW 가정:
    - 단어 순서 무시: 문장 내 단어 순서는 중요하지 않음
    - 단어 빈도만 고려: 각 단어의 등장 횟수만 반영
    - 문서 길이 반영 가능: 단어 빈도 벡터의 합으로 간접 반영
- BoW 한계:
    - 단어 순서와 문맥 정보 미반영
    - 의미적 유사 단어 구분 불가
    - 희소하고 고차원 벡터 생성

8. Word2Vec

- 단어를 벡터로 임베딩하는 방법(CBOW, Skip-gram)
- 네거티브 샘플링(negative sampling):
  - 전체 단어집합이 아닌 일부 네거티브 샘플만 사용
  - 중심 단어와 주변 단어 쌍이 실제 문맥에 함께 등장하는지(1) 아닌지(0) 구분하는 이진 분류 문제로 변환
- 효과:
  - 계산 효율성 향상: 전체 softmax 대신 소수 샘플에 대한 sigmoid 계산
  - 학습 속도 향상: 대용량 코퍼스에서도 빠른 학습 가능
  - 임베딩 품질 유지: 적절한 샘플 수 선택 시 전체 softmax와 유사한 품질

9. FastText

- Facebook AI Research에서 개발한 단어 임베딩 및 텍스트 분류 모델
- 특징: 단어 내부의 subword(부분단어, n-gram) 정보 활용
- 장점:
  - 오타 및 희귀 단어 처리에 강함: 문자 단위 n-gram 분해로 OOV 단어도 임베딩 가능
  - 형태소가 중요한 언어(한국어 등)에 유리: 공통 subword 공유로 의미적 유사성 반영
  - 학습 및 추론 속도가 빠름
  - 텍스트 분류에도 활용 가능

10. ELMo(Embeddings from Language Models)

- 문맥(Context)에 따라 단어 임베딩이 달라지는 특징
- 기존 LSTM과의 차이:
  - 딥(Deep) 양방향 LSTM 사용: 여러 층의 BiLSTM으로 깊은 문맥 정보 학습
  - 문맥 기반 임베딩: 문장 내 위치, 주변 단어에 따라 동적 생성
  - 사전학습 언어모델의 각 층 출력을 가중합하여 최종 임베딩 생성
- 필요성:
  - 고정 임베딩(Word2Vec, GloVe)의 한계 극복: 동음이의어, 다의어 구분
  - 문맥에 따라 의미가 달라지는 자연어 특성 반영
  - 다양한 NLP 태스크에서 성능 향상

11. BERT(Bidirectional Encoder Representations from Transformers)

- 2018년 구글 개발, 양방향 트랜스포머 인코더 기반 사전훈련 언어 모델
- 주요 특징:
  - 양방향성: Masked Language Model(MLM) 방식으로 양방향 문맥 학습
  - 사전훈련-미세조정: 대규모 코퍼스로 사전훈련 후 특정 태스크에 미세조정
  - 모델 구조: 트랜스포머 인코더 층 다층 구조(BERT-Base: 12층, BERT-Large: 24층)
- 사전훈련 방식:
  - MLM: 입력 텍스트 일부 토큰(15%) 마스킹 후 예측
  - NSP: 두 문장이 연속적인지 예측하는 이진 분류
- 응용: 텍스트 분류, 개체명 인식, 질의응답, 문장 쌍 분류, 텍스트 요약 등

12. Doc2Vec

- 문서(문장, 문단, 전체 문서) 단위의 임베딩 생성 기법
- Word2Vec과 차이:
  - Word2Vec: 단어 임베딩, 문맥 정보 제한적
  - Doc2Vec: 문서 전체 임베딩, 문맥+문서 전체 의미 반영
- 주요 원리: Word2Vec 구조 확장, 문서 고유 벡터(문서 태그) 추가 학습
  - Distributed Memory(DM) 모델: 문맥 단어 벡터 + 문서 벡터로 다음 단어 예측
  - Distributed Bag of Words(DBOW) 모델: 문서 벡터만으로 문서 내 임의 단어 예측
- 활용: 문서 분류, 유사 문서 검색, 문서 군집화 및 시각화