공학박사 학위논문

언어 지식과 문맥 정보를 융합한 형태소 단위의 언어 모델

A Language Model of Morpheme Unit that Ensemble Knowledge of Language and Syntactic Information

울산대학교 대학원
전기전자컴퓨터공학과
이 주 상

언어 지식과 문맥 정보를 융합한 형태소 단위의 언어 모델

A Language Model of Morpheme Unit that Ensemble Knowledge of Language and Syntactic Information

지도교수 옥철영

이 논문을 공학박사 학위논문으로 제출함

2023년 8월

울산대학교 대학원
전기전자컴퓨터공학과
이 주 상

이주상의 공학박사 학위논문을 인준함

심사위원장 윤 석 훈 (인)

심사위원 옥 철 영 (인)

심사위원 안 대 한 (인)

심사위원 허 정 (인)

심사위원 김 민 호 (인)

울산대학교 대학원

2023년 8월

감사의 글

인생에 있어 많은 인연들을 만나고, 도움을 받아 성장한다고 생각합니다. 박사 학위를 하는 동안 많은 분들을 만나서 따뜻한 조언과 도움을 통해 한 사람의 인간으로서 한 단계 성장하는 계기가 되었으며, 혼자만의 노력이 아닌 많은 이들의 노력으로 박사 학위를 받을 수 있었다고 생각합니다.

먼저 긴 시간 고생해준 가족들에게 감사의 말을 전하고 싶습니다. 저를 지지해주고 받쳐준 가족들 덕분에 무사히 학위를 마칠 수 있었습니다. 그리고 저를 한 사람의 연구자로 성장시켜 주시면서 인생에 대해 아낌없는 지도해주신 옥철영 교수님께 감사의 말씀을 전합니다. 또한 부족한 저의 박사 학위 논문을 심사해주시고 냉철한 평가를 해주신, 윤석훈 교수님, 안대한 교수님, 허정 박사님, 김민호 교수님께 감사합니다. 심사 과정에서 지적하신 많은 부분과 조언을 통해 제 논문을 완성하는데 큰 도움이 되었습니다.

연구실에서 생활하며 만난 많은 분들에게도 감사의 말을 전하고 싶습니다. 활기찬 연구실 분위기를 만들어 연구하기 편한 환경을 조성해주며 많은 조언을 해준 선배님들과, 부족한 선배를 따라준 후배님들에게 감사합니다.

박사 학위를 마치면서 하나의 이정표에 마침표를 찍는 것이 아닌 다른 새로운 세계로 가는 길에 도달했다고 생각합니다. 이 과정 중에 많은 분들의 도움과 가르침을 본받아 저도 베풀며 살아가가겠다고 다짐을 하는 계기가 된 것 같습니다.

감사합니다.

이주상

[국문 요약]

언어 지식과 문맥 정보를 융합한 형태소 단위의 언어 모델

자연어는 다양하게 쓰이며 매우 복잡한 구조를 가진다. 자연어 표현은 컴퓨터가 이해할 수 있는 형태로 자연어를 표현하는 방식으로 자연어 처리 영역에서 선결되어야 하는 문제로 자리 잡았다. 자연어를 표현하기 위해 다양한 방식을 사용했지만, 현재는 딥 러닝에 기반을 둔 언어 모델이 대표적인 자연어 표현 방식이다. 언어 모델은 주변의 문맥 정보를 이용해 현재 위치에 사용될 수 있는 단어를 유추하는 학습을 통해 모델 자체가 자연어 표현 방식이 된다. 언어 모델의 발전으로 인해 다양한 쓰임새와 성능의 향상을 통해 자연어 처리 분야의 발전을 견인하고 있다.

언어 모델은 자연어를 효율적으로 표현하기 위해 토큰이라는 표현 단위를 사용한다. 토큰은 음절보다 크거나 같으며, 어절보다 작거나 같은 표현 단위이다. 하지만 토큰 표현 방식은 등장 빈도를 이용하여 생성하는 방식으로, 학습 데이터에서 등장 빈도가 낮은 단어에 대한 의미나, 표현을 이해하기 어려운 문제를 가지고 있다. 그리고 해당 단어는 여러 토큰으로 분리될 가능성이 높아 문맥의 정보를 온전히 해당 단어에 적용하지 못하며, 원래 단어의 의미와 다른 해석을 모델이 실시할 가능성이 있다. 토큰 표현 방식으로 인한 학습 빈도가 낮은 단어들에 대한 이해력을 높이기 위해서는 다양한 형태와 의미로 단어를 사용한 학습 말뭉치를 필요로 한다. 하지만, 이러한 말뭉치를 확보하기 어렵고, 구축에는 많은 비용이 발생한다.

본 논문에서는 다양한 언어 지식과 문맥 정보를 융합한 형태소 단위의 언어 모델인 UKnowBERT(Ulsan Knowledge ensemble BERT)를 제안한다. UKnowBERT는 형태소 단위를 유지하기 위해 다중-핫 표현(multi-hot representation) 방식을 사용한다. 다중-핫 표현은 형태소를 하나의 토큰으로 표현할 수 없는 경우 음절 토큰의 집합으로 표현하며, 다중-핫 입력 생성기를 통해 형태소 단위를 유지하여 언어 모델의 입력에 사용한 방식이다.

다중-핫 표현 방식을 사용하는 경우 마스킹 학습(Masked LM)에서 정답의 수가 1개 이상이 가능하기 때문에 기존의 손실 함수가 아닌 정답의 수에 따른 목표 확률을 조정하는 손실 함수를 개발했다. 다양한 손실 함수와의 비교 실험을 통해 본 논문에서 제안한 손실 함수가 기존의 다른 손실 함수보다 높은 성능을 보였다. 그리고 기존 토큰 방식을 사용한 BERT 모델과의 비교 실험을 통해 개체명 인식, 감성 분석 영역에서 기존 토큰 표현 방식에 비해 1% 포인트 높은 성능을 보였다. 하지만, 기계 독해, 의미역 영역과 같이 단어와 문장 간의 관계를 이해하는 영역에서는 기존 모델이 앞서는 결과를 보였다.

형태소 단위를 유지하는 다중-핫 표현만으로는 단어가 가진 다양한 의미와 표현을 언어 모델이 이해하지 못하는 현상을 해결하기 위해 UKnowBERT는 다양한 언어 지식을 함께 사전 학습(pre-training)했다. 한국어 어휘 의미망인 UWordMap에서 추출한 상위어 정보, 용언의 의미제약정보를 이용한 관계성 정보, 단어가 가진 뜻풀이를 벡터로 표현한 정보를 사용했다. 상위어 정보의 경우 명사가 가진 상위어를 해당 단어의 입력 벡터가 추론하는 학습을 진행했으며, 이를 통해 동일한 상위어를 가진 단어끼리 군집화를 이루도록 설계했다. 그리고 사전(dictionary)에 등재된 단어의 뜻풀이를 벡터로 표현한 USenseVector를 활용하여 단어가 가진 뜻풀이를 해당 단어를 표현한 입력 임베딩 벡터에 포함되도록 학습했다. 마지막으로 용언의 의미제약 정보를 사용해 용언이 가리키는 명사의 위치정보를 추론하는 학습을 사용했다. 해당 학습을 통해 언어 모델이 특정 용언과 올 수 있는 다양한 명사들의 관계를 이해하도록 유도했다. 상위어 추론과 뜻풀이 기반의 벡터 학습의 경우 인간처럼 미리 습득한 정보를 활용하는 것과 같이 해당 단어를 표현하는 벡터에 적용했으며, 언어 모델이 해당 단어에 대한 다양한 의미들 중에 문맥을 해석하여 적절한 정보를 찾아내도록 유도했다. 또한 단어에 대한 학습 빈도가 부족할 경우 미리 해당 단어에 대한 정보를 기억할 수 있도록 설계했다.

언어 지식을 사용한 언어 모델에 대한 다양한 실험을 통해 용언의 의미제약 정보를 활용할 경우 기계 독해, 의미역 인식 영역에서 사용하지 않은 다른 언어 모델에 비해 높은 성능을 보였다. 그리고 상위어 추론, 뜻풀이 벡터 학습의 경우 개체명 인식처럼 단어에 대한 의미가 중요한 영역에서 다른 모델에 비해 높은 성능을 나타냈다. 하지만, 특정한 언어 지식만 추가한 경우 다른 영역에서 낮은 성능을 기록했다. 반면, 모든 언어 지식을 함께 사전 학습한 경우 모든 실험 영역에서 높은 성능을 보였다. 또한 small 크기의 모

델에서는 기존 BERT 모델과 성능이 유사하거나 낮은 영역이 발생했지만, base 크기를 사용할 경우 더 많은 파라미터에 많은 정보를 기억할 수 있어 기존의 BERT 모델보다 UKnowBERT의 성능이 모든 영역에서 앞섰다.

실험 영역의 학습 데이터를 축소하여 제한된 환경에서의 실험을 통해 언어 지식이 해당 응용 영역이 학습하지 못한 부분에 대해 추가적인 정보를 제공하기 때문에 기존 BERT 모델보다 성능이 높았으며, 제한하지 않은 환경에 대비해서 성능의 하락이 적었다. 마지막으로 사전 학습 단계별 성능 비교 실험을 통해 언어 지식을 학습하는 경우 언어 모델이 일정 성능에 빨리 도달했으며, 사용하지 않은 모델 대비 1% 포인트 높은 성능을 보였다.

본 논문에서 제안한 UKnowBERT는 다중-핫 표현 방식을 통해 토큰의 수를 줄여도 더 많은 단어들을 토큰으로 표현하며, 전체 언어 모델의 크기를 줄일 수 있는 장점을 가지며, 성능을 유지할 수 있었다. 그리고 언어 지식을 함께 사전 학습함으로 인해 말뭉치 학습만으로는 얻을 수 없는 부족한 정보를 언어 모델이 습득할 수 있었으며, 단어에 대한 다양한 표현과 의미를 학습하는 효과를 가져와 기존 BERT에 비해 높은 성능을 보였다. 본 논문에서 제안한 모델을 통해 형태소 단위를 필요로 하는 다른 다양한 자연어 처리 응용 영역에 대해서 UKnowBERT가 사용한 다중-핫 표현 방식이 하나의 해결 방법이 될 수 있으며, 응용 영역이나, 언어 모델의 사전 학습에 필요한 말뭉치의 구축에 필요한 비용이나 시간을 언어 지식을 통해 해소할 수 있었다. 마지막으로 본 연구의 결과를 통해 자연어 이해 영역을 넘어서 자연어 생성 모델에도 언어 지식의 활용 가능성에 대한 연구가 필요하다.

목차

그림 차례

표 차례

1 서론

1.1 연구 배경

인간은 오랜 세월 동안 자연어를 사용하면서 얻은 지식과 다양한 연구를 통해 자연어에 대한 규칙과 활용법을 정립했다. 인간은 규칙과 경험을 통해 자연어를 다각도로 이해할 수 있으며, 창의적인 활용이 가능하다. 인간처럼 자연어에 대해 이해하고 활용하는 기계에 대한 관심은 인공지능 연구의 원동력이 되었다. 인공지능의 목표는 인간의 지식을 모방하고 나아가 인간 이상의 사고가 가능하도록 함이며, 많은 연구가 진행되고 있다.

자연어를 이해하는 인공지능에 대한 연구는 딥 러닝(Deep learning) 모델과 하드웨어의 발전으로 인해 다시 큰 관심을 가지게 되었다. 딥 러닝 모델을 이용한 자연어 처리 모델들은 기존의 통계와 규칙 기반보다 개선된 성능을 가져왔다. 이후 대규모 자원을 사용하는 언어 모델의 등장으로 자연어 처리 분야는 성능의 향상과 함께 기존에 접근할 수 없었던 새로운 응용 영역으로 접근할 수 있었다. 단순히 자연어의 의미 분석 및 이해 영역을 넘어서서 다른 언어로의 번역, 문서의 요약 및 생성, 효율적인 교육을 위한 학습 커리큘럼 제공, 질의응답이 가능한 챗봇(Chatbot) 등의 복잡한 수준의 자연어 서비스를 제공하고 있다.

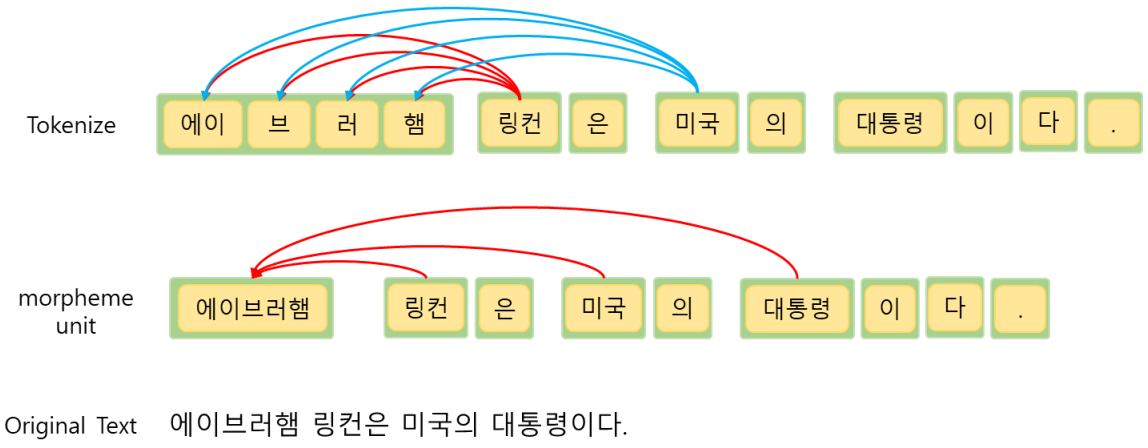
딥 러닝을 사용하는 자연어 처리에서 선결되어야 하는 과제는 다양한 자연어들을 컴퓨터가 이해할 수 있는 형태로 표현하는 것이다. 이미지의 경우 하나의 픽셀에 위치할 수 있는 RGB 표현은 한정되어 있으며, 음성의 경우 디지털 수치로 변환 가능하기 때문에 쉽게 컴퓨터가 이해할 수 있는 숫자의 형태로 표현 가능하다. 그러나 자연어의 경우 글자를 어떻게 숫자로 표현할 것인가에 대해 쉽게 정의하기 어렵다. 자연어의 경우 사용된 단어의 쓰임이나 조합에 따라 의미가 달라지며, 새로운 단어가 계속 추가되기 때문이다. 이러한 이유로 자연어를 표현하는 방법이 진화함에 따라 컴퓨터가 자연어를 이해하고 표현하는 수준의 변화를 가져왔다.

자연어 표현(Language representation)은 자연어를 표현하는 단위와 표현을 위해 사용한 모델에 따라 발전해왔다. 자연어 표현에서 최소 표현 단위는 중요한 요소로 표현 가능한 자연어의 수를 결정하며, 컴퓨터가 자연어를 이해하는 최소 단위로 사용된다. 이전

의 자연어 표현은 Word2Vec[1]처럼 학습 모델을 통해 단어들을 특정한 차원의 벡터로 표현하는 방법을 사용했지만, 해당 단어가 다른 의미로 사용된 문장에서도 동일한 벡터로 표현되는 문제를 가진다. 트랜스포머 모델(Transformer model)[2]의 등장과 이를 이용한 언어 모델(Language model)이 등장으로 동일한 단어가 문맥에 따라 다른 해석을 할 수 있게 되었다. 현재에는 언어 모델 자체가 자연어를 표현하는 대표적인 방식으로 자리 잡았다.

언어 모델은 토큰 표현 방식과 대용량의 말뭉치를 이용하여 학습한 모델이다. 토큰 표현 방식은 언어 모델에 효과적으로 많은 자연어를 표현하기 위한 방법이다. 토큰은 음절보다 크거나 같으며 어절보다는 작거나 같은 표현 단위이다. 토큰의 구성을 위해 학습 말뭉치에 등장한 단어의 빈도를 사용했으며, 빈도가 높은 단어의 경우 해당 단어 형태 그대로 토큰으로 등록되는 방식이다. 언어 모델은 다양한 형태로 사용된 자연어들을 이해하기 위해 대용량의 말뭉치를 사용했다. 말뭉치에 쓰인 다양한 표현들을 주변 문맥 정보를 이용하여 이해하도록 설계했다.

언어 모델이 사용한 토큰 표현 방식은 자연어를 효율적으로 표현하는 방법이지만, 단어들의 사용 빈도를 토큰으로 표현한 방식은 여러 문제점을 가지고 있다. 토큰 생성에 빈도를 사용하기 때문에 학습 데이터에 등장 빈도가 낮은 단어들은 여러 토큰으로 분리될 가능성이 높다. 하나의 단어가 여러 토큰으로 분리될 경우 주변의 문맥 정보가 해당 단어에 온전히 반영되기 어렵다. 그리고 단어를 구성하는 토큰들 중에 특정 토큰들은 다른 의미와 사용법으로 많은 학습을 진행한 경우 해당 단어에 대한 언어 모델의 결과가 오류를 가질 수 있다.



[그림 1] 토큰 표현 방식에 의한 단어의 분리 예시

[그림 1]에서 '에이브러햄'라는 단어가 토큰으로 표현된 경우와 단어 단위를 유지한 경우를 비교한 그림이다. 학습 과정에서 '에이브러햄'이라는 단어의 등장 빈도가 높은 경우 문제가 발생하지 않는다. 하지만, 학습에 많이 등장하지 않고 '에이'라는 토큰이 학습 과정에서 영어 단어 'A'를 한글로 표기한 데이터를 많이 학습했으면, '에이브러햄'을 해석하는 경우 '에이'라는 토큰 때문에 잘못된 정보를 제공할 수 있다. 또한 단어 단위가 유지되는 경우 주변 문맥의 정보가 '에이브러햄'이라는 단어에 직접 적용이 되지만, [그림 1]에서의 관계를 나타낸 간선처럼 주변 문맥의 정보가 각각의 토큰에 적용되는 현상이 발생하며, 단어가 가진 관계와 다른 해석이 발생하는 경우가 생긴다.

토큰 표현 방식에서 발생하는 빈도가 낮은 단어의 분리와 단어가 가진 다양한 의미를 학습하기 어려운 문제를 해결하는 대표적인 방법으로는 단어를 다양한 형태로 표현된 많은 양의 학습 말뭉치를 사용하는 방법이 있다. 하지만, 다양한 표현과 의미를 사용한 말뭉치의 확보는 어렵다. 예를 들어 인터넷에 있는 다양한 자료나, 서적, 출판물에서 자료를 확보하는 경우 저작권의 문제가 존재하며, 다양한 형태로 문서가 작성되어 있기 때문에 학습을 위한 정제 과정이 필요하다. 그리고 만약 하나의 환경에서만 구축된 문서를 수집하는 경우 저자의 성향이나, 문체에 대해 언어 모델이 편향된 학습 결과를 가져올 수도 있다. 학습 말뭉치의 편향성을 줄이면서 다양한 자연어를 사용한 학습 데이터를 구축하기 위해서는 많은 비용이 필요하다.

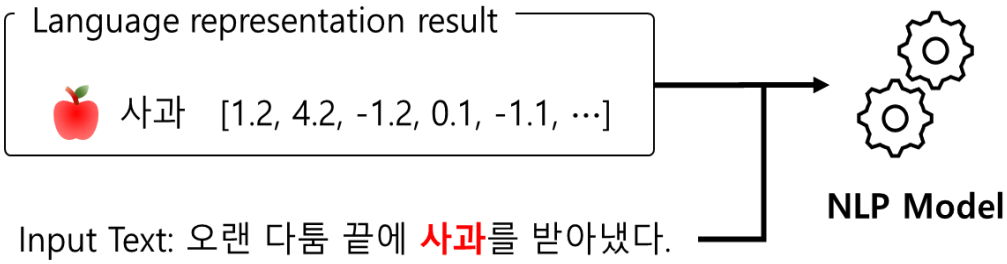
본 논문에서는 토큰 표현 방식에 따른 단어의 분리를 막기 위해 다중-핫 표현(multi-hot representation) 방식을 사용하며, 다양한 단어들의 표현을 모델이 이해하면서, 빈도가 낮은 단어에 대한 이해력을 높이기 위해 언어 지식을 활용한 UKnowBERT(Ulsan Knowledge ensemble BERT)을 제안한다. 본 논문에서 제안한 모델은 다중-핫 표현 방식을 통해 형태소 단위 유지하여 주변 문맥 정보가 고스란히 해당 형태소에 전달될 수 있으며, 빈도가 낮은 형태소에 대한 부족한 정보를 언어 지식에서 습득할 수 있다. 이를 통해 학습 말뭉치를 추가적으로 구축하지 않아도 언어 지식을 통해 형태소가 가진 다양한 의미와 표현들을 습득하고 이해하는 언어 모델을 개발한다.

2 관련 연구

2.1 자연어 표현 방식의 변화

자연어를 표현하는 방법에 따라 딥 러닝을 사용하는 자연어 처리에 대해 새로운 패러다임을 제공했다. 어떠한 방식으로 자연어를 표현하는가에 따라 표현 가능한 자연어의 수가 달라지며, 자연어가 가진 의미를 컴퓨터가 이해하는 수준을 결정하게 된다. 딥 러닝의 발전으로 인해 자연어 표현 방식도 인공 신경망(Artificial Network)을 이용한 모델들이 연구되었다.

자연어 표현은 순방향 신경망(Feedforward Neural Network)을 사용한 NNLM(Neural Network Language model)[3]를 시작으로 발전되었다. 초기 자연어 표현은 단어들을 하나의 벡터로 표현함으로 인해 벡터 간의 거리나 위치에 따라 단어 간의 관계성을 파악할 수 있었다. 인공 신경망과 학습 말뭉치를 사용하여 단어들을 고정된 벡터로 표현하는 대표적인 모델로 Word2Vec[1], Glove[4], fastText[5] 등이 있다. 벡터로 표현된 단어들을 이용해 자연어 처리 분야의 성능이 개선되었지만, 고정된 벡터 형태로 표현된 단어는 해당 단어가 가진 다양한 의미를 표현하지 못하는 문제를 가진다. 단어의 경우 동일한 형태라도 문장에 따라 다른 의미로 사용될 수 있지만, 학습을 통해 생성된 고정된 벡터로 표현된 경우 문장에 따라 의미를 반영하지 못하며, 응용 모델에서 오류가 발생할 수도 있다.



[그림 2] 자연어 표현에서 고정형 벡터 표현의 문제점

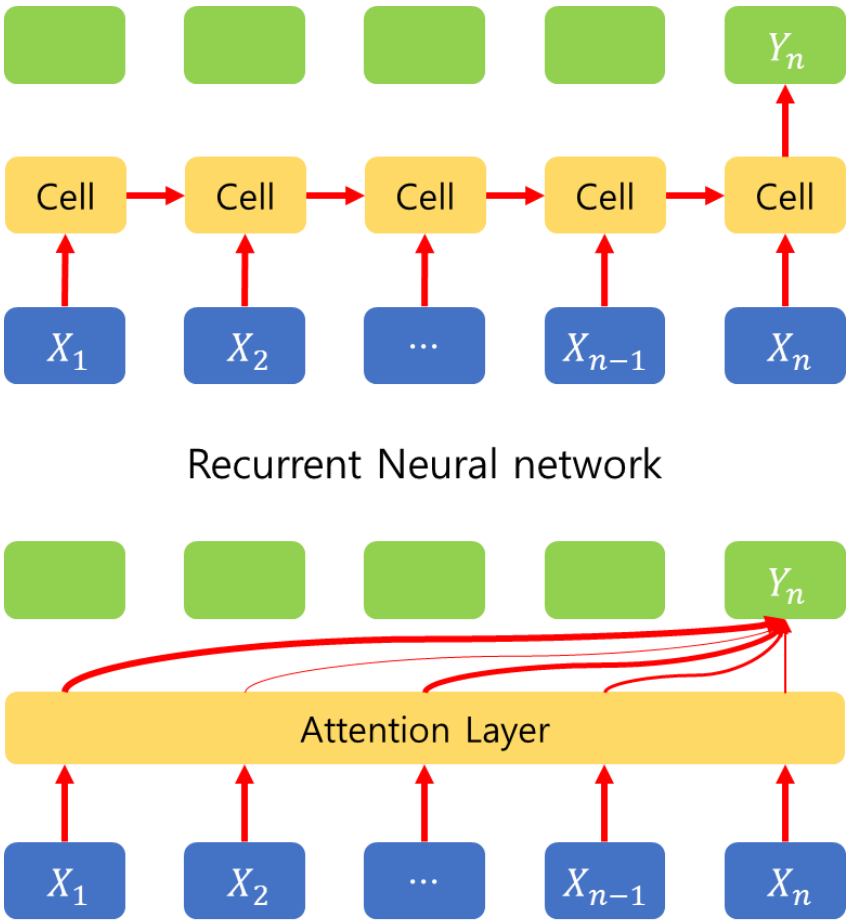
[그림 2]는 하나의 단어를 고정된 벡터로 표현할 경우 발생하는 문제의 예시이다. 만약 학습을 통해 '사과'라는 단어의 벡터가 "사과나무의 열매"라는 의미로만 학습된 경우 [그림 2]의 예문처럼 "오랜 다툼 끝에 사과를 받아냈다."라는 문장을 처리할 때 잘못된 분석

결과로 이루어질 가능성이 높다. 예문에서 '사과'의 의미는 "잘못을 용서함"으로 사용됐지만, '사과'라는 단어를 표현한 벡터에는 해당 의미가 학습되지 않았기 때문이다. 그리고 단어 단위의 고정된 벡터 표현 방식은 많은 단어를 벡터로 표현하거나, 해당 단어가 가진 다양한 의미를 벡터로 표현하기 위해서는 다양한 의미로 사용된 예제들이 포함된 대용량의 말뭉치가 필요하다.

2023년 현재에는 2017년에 등장한 ELMo(Embeddings from Language Model)[6]를 시작으로 딥 러닝 모델 자체가 자연어를 표현하는 방식이 대세가 됐다. ELMo는 순환 신경망(Recurrent Neural Network)인 LSTM(Long Short-Term Memory)[7]을 양방향으로 사용해 입력한 문장에 따라 자연어를 벡터로 표현하는 모델이다. 이후 트랜스포머 모델(transformer model)[2]의 등장으로 어텐션(attention)을 사용한 언어 모델(Language Model)은 대표적인 자연어 표현 방법이 됐다. 대표적으로 트랜스포머 모델을 사용한 언어 모델에는 BERT[8]와 GPT[9]가 있다. 이후 다양한 연구들을 통해 거대 언어 모델(Large Language Model)에 대한 발전이 이루어졌다.

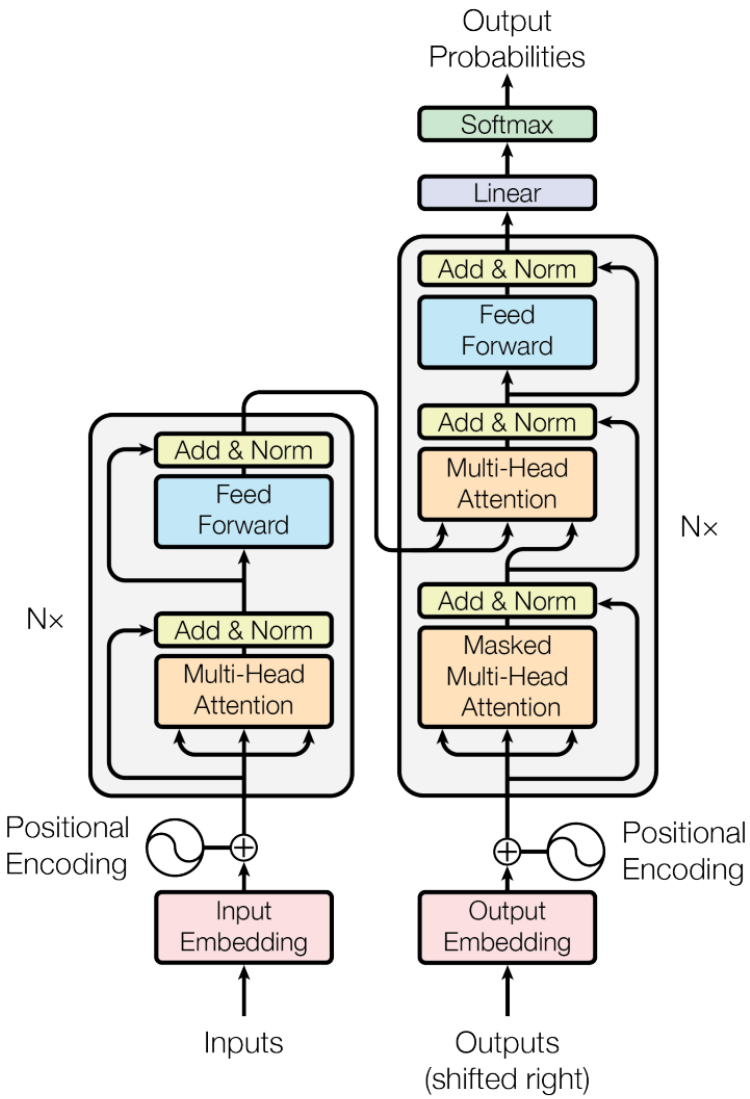
2.2 트랜스포머 모델

트랜스포머 모델[2]의 등장은 언어 모델에 큰 변화를 가져왔다. 트랜스포머 모델은 인코더-디코더(encoder-decoder) 형태의 모델로 어텐션(self-attention)을 사용한 모델이다. 어텐션 방식은 기존의 순환 신경망의 문제점인 거리에 따른 정보 손실을 해결하며, 모델의 학습 속도를 개선하기 위한 방법이다. 다음 그림은 순환 신경망 방식(Recurrent Neural Network)과 어텐션 방식(attention)이 출력을 생성하는 방식을 비교한 그림이다.



[그림 3] 순환 신경망과 어텐션 방식의 출력 값 Y_n의 생성 방법

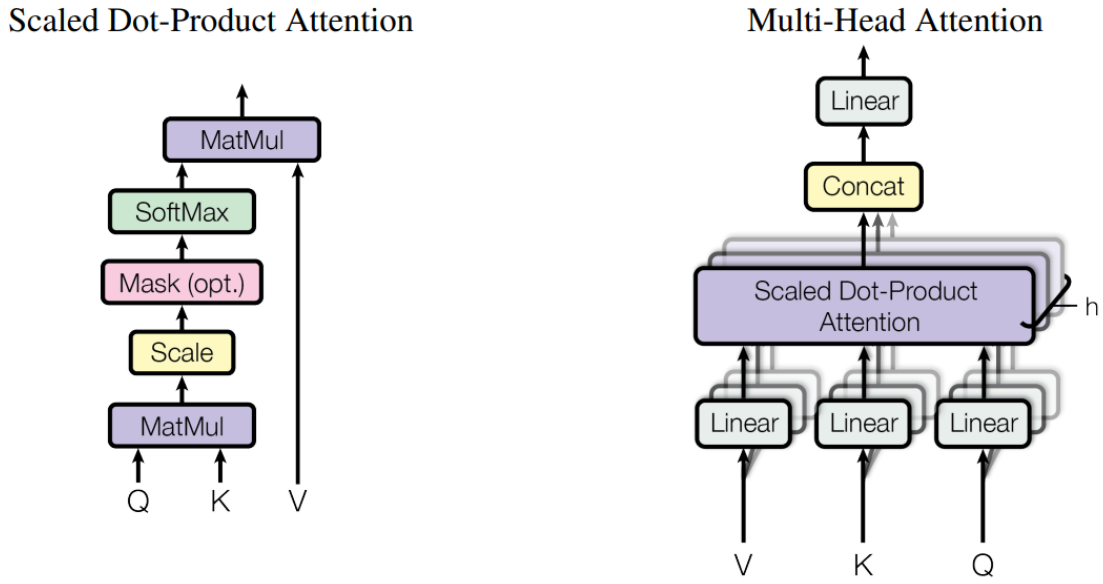
[그림 3]은 순환 신경망과 어텐션 방식에서 n번째 위치의 출력 값인 Y_n이 생성되는 과정을 보여준다. 순환 신경망에서는 첫 번째 입력 값인 X_1의 정보가 Y_n에 적용되기 위해서는 n번째 위치까지 순환 신경망의 셀(Cell)들을 통과해야 한다. 하지만, 많은 셀을 통과하는 과정에서 X_1가 가진 정보가 소실되거나 희미하게 전달된다. 하지만 어텐션 방식은 어텐션 점수(attention score)를 이용한다. Y_n에 대한 각 입력의 어텐션 점수에 따라 입력 정보의 사용량을 결정한다. 예를 들어 X_1의 어텐션 점수가 다른 입력에 비해 높을 경우 Y_n의 값에는 다른 입력 값보다 X_1의 정보가 많이 포함된다.



[그림 4] 트랜스포머 모델의 구조

(출처: Attention Is All You Need. Advances, Vaswani, A., et al[2])

[그림 4]는 트랜스포머 모델의 구조를 보여준다. 기존의 Sequence to Sequence 모델처럼 입력에 대한 인코더 부분과 출력을 위한 디코더 부분으로 구성되어 있다. 트랜스포머 모델은 어텐션이라는 방식을 사용해 구현한 모델로 순환 신경망 계열 모델의 느린 속도와 정보 소실을 해결하기 위해 어텐션 구조를 사용했다. 트랜스포머 모델은 멀티-헤드 어텐션(multi-head attention)을 사용해 여러 모델이 앙상블을 이루도록 구현했다. 다음은 트랜스포머 모델의 멀티-헤드 어텐션과 어텐션의 세부 구조를 표현한 그림이다.



[그림 5] 어텐션의 세부 구조와 멀티-헤드 어텐션의 구조

(출처: Attention Is All You Need. Advances, Vaswani, A., et al[2])

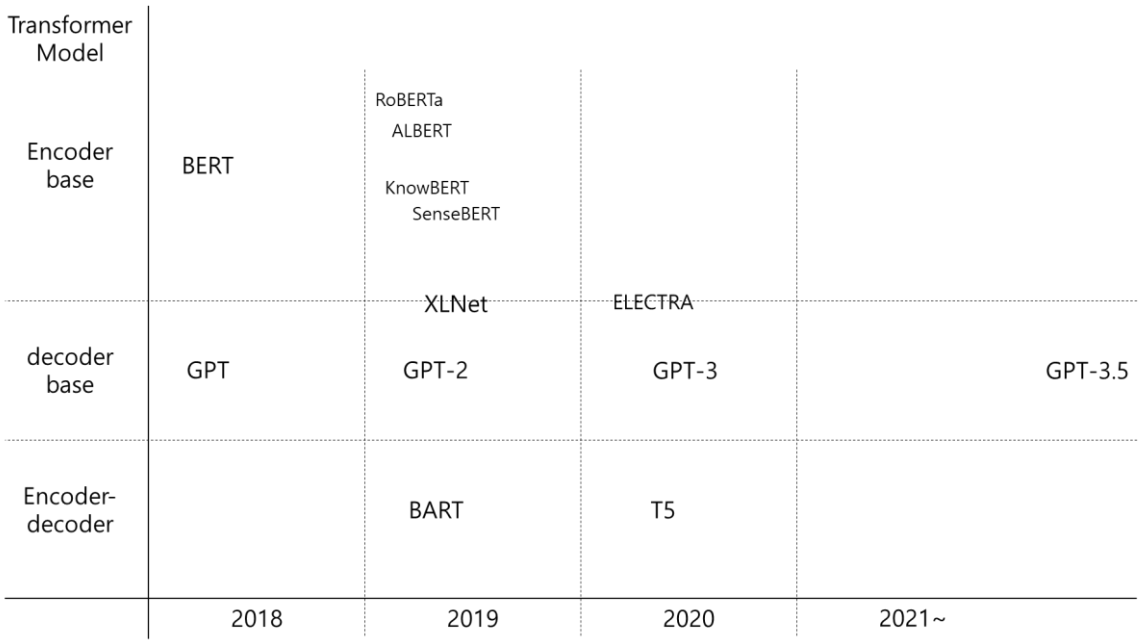
[그림 5]는 멀티-헤드 어텐션과 어텐션의 세부 구조를 보여준다. 트랜스포머 모델은 학습 속도의 개선을 위해 어텐션 계층의 구조에 선형 순방향 신경망(Feedforward Neural Net)을 사용했다. 어텐션은 입력으로 key, query, value의 세 가지 값을 사용한다.

$$Attention(Q, K, V) = softmax(QK^T)V \quad \text{[수식 1]}$$

[수식 1]은 어텐션 점수를 통해 입력에 대한 출력을 생성하는 수식이다. query 벡터와 key 벡터를 소프트맥스 함수(softmax function)를 사용하여 어텐션 점수를 계산한다. 계산된 점수와 value 벡터의 행렬 곱을 통해 어텐션 계층의 출력 값을 구하는 방식이다. 멀티-헤드 어텐션 방식을 통해 현재의 입력에 대해 다양한 어텐션 결과가 앙상블을 이루는 효과를 위해 사용했다. 하나의 key, query, value 벡터를 일정한 크기로 분리하여 각 어텐션에 입력으로 사용하며, 출력 결과를 묶어 멀티-헤드 어텐션의 결과로 표현된다.

2.3 트랜스포머 기반의 언어 모델

트랜스포머 기반의 언어 모델은 사용한 모델의 구조에 따라 크게 3가지 부류로 구분된다. 트랜스포머의 인코더만 사용하거나, 디코더만 사용한 모델, 마지막으로 트랜스포머 전체를 사용한 모델로 구분할 수 있다. 모델의 구조에 따라 구분된 언어 모델들은 각자의 모델 구조에서의 성능 향상을 위해 다양한 연구가 진행되었다. 아래의 그림은 연도별 발표된 사용된 모델의 구조의 분류에 따른 언어 모델을 보여준다.

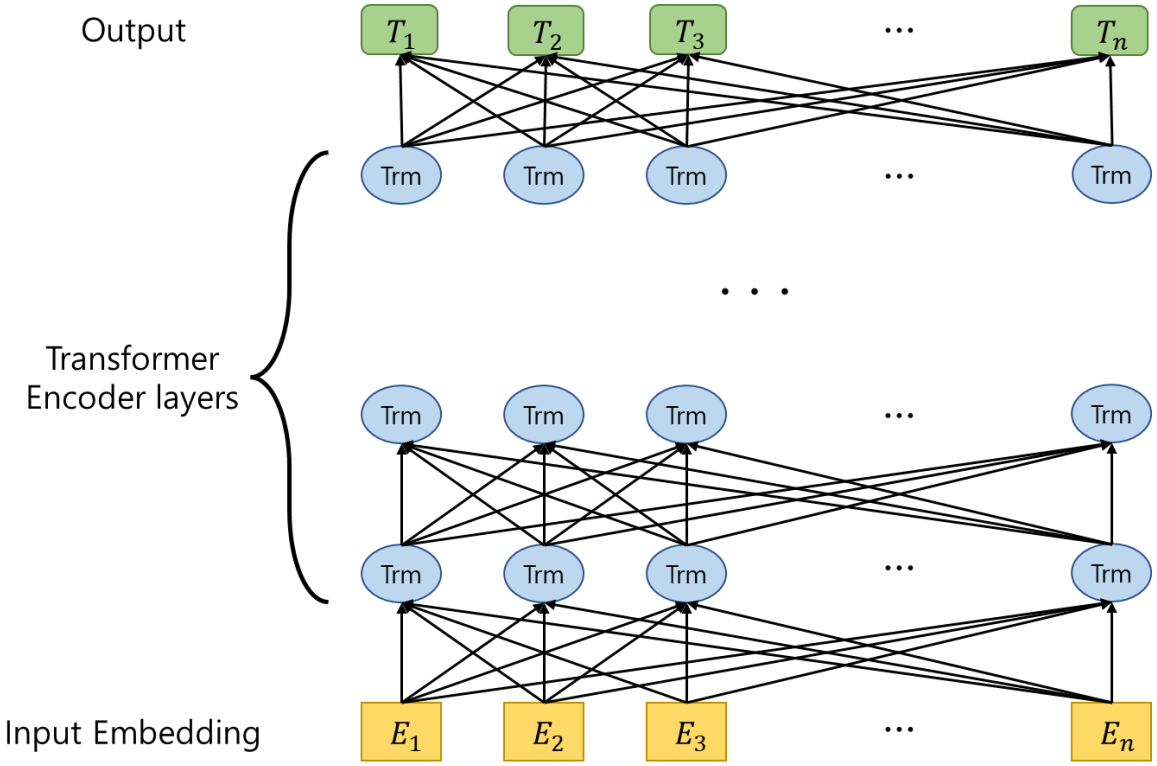


[그림 6] 모델 구분과 년도에 따라 발표된 언어 모델

2018년에 트랜스포머 모델의 인코딩 부분을 사용한 BERT(Bidirectional Encoder Representations from Transformers)[8]와 디코더 부분을 사용한 GPT(Generative Pre-Training)[9]를 시작으로 다양한 언어 모델이 등장했다. BERT와 GPT로 인해 자연어 처리 응용 영역의 성능이 한 단계 도약했다. 트랜스포머 인코더 기반의 언어 모델은 자연어의 이해에 특화되었으며, 트랜스포머 디코더 기반과 트랜스포머 전체를 사용한 언어 모델은 자연어 생성에 초점을 두고 있는 모델이다.

2.3.1 트랜스포머 인코더 기반 언어 모델

트랜스포머의 인코더 기반의 언어 모델은 자연어의 이해에 많이 사용된다. 대표적인 인코더 기반의 언어 모델인 BERT[8]는 어텐션을 통해 먼 곳에 있는 토큰의 정보들을 사용할 수 있으며, 문장에서 현재 위치의 토큰을 해석하기 위해 주변 문맥 정보를 활용하는 방식으로 설계된 모델이다. 토큰은 언어 모델에서 사용하는 자연어를 표현하는 단위로 음절보다 크거나 같으며, 어절보다는 작거나 같은 크기이다. BERT는 입력 받은 토큰들에 대해 양방향의 정보를 사용했다.



[그림 7] BERT 모델의 구조

(출처: Bert: Pre-training of deep bidirectional transformers for language understanding.[8])

[그림 7]은 BERT의 구조를 보여준다. BERT는 토큰과 여러 정보를 사용해 입력을 생성한다. 각 토큰의 입력 값은 트랜스포머 인코더 계층들을 통과하여 최종적인 출력 값을 생성한다.

$$E_k = T_k + S_k + P_k \quad \text{[수식 2]}$$

[수식 2]는 BERT에서 트랜스포머 인코더 계층의 k번째 입력 값인 E_k가 생성되는 수식이다. [수식 2]에서 T_k는 k번째 입력된 토큰에 대한 임베딩(token embedding) 값이며, S_k는 k번째 위치의 토큰이 속한 문장의 번호를 임베딩(sentence embedding) 값이다. 마지막으로 P_k는 k번째 토큰의 절대 위치를 임베딩(position embedding) 값이다. 세 가지 값을 모두 더한 벡터가 k번째 토큰의 입력 값인 E_k가 된다.

BERT는 두 가지 학습 방법을 사용한다. 마스킹 방식(masked LM)은 학습에 사용할 문장에서 일부의 토큰을 특정한 마스킹 토큰으로 치환하여 모델의 입력으로 사용하며, 모델이 치환된 부분의 토큰을 유추하는 학습 방법이다. 마스킹 방식의 학습을 통해 언어 모델이 해당 위치의 토큰을 추론하면서 토큰들의 조합에 따른 다양한 패턴을 이해하도록 설계했다. BERT에서는 전체 입력에 대해 15%에 대해 마스킹 처리를 하여 학습했다. 이러한 마스킹 학습 방식은 마스킹의 대상이 되는 토큰들을 선택하는 방법에 따라 다양한 마스킹 기법이 존재한다. 다음의 표는 다양한 마스킹 방법을 보여준다.

[표 1] 언어 모델의 학습을 위한 다양한 마스킹 방식

입력 원문	작품을 원작으로 하거나 모티브로 삼은 영화만도 100편 가까이 될 정도다.
기본 방식	작품을 원작으로 하거나 [MASK]로 삼은 영화만도 100편 가까이 될 정도다 .
n-gram, Span 방식	작품을 원작으로 하거나 [MASK][MASK] [MASK] 영화만도 100편 가까이 될 정도다 .

[표 1]과 같이 다양한 마스킹 방식은 자연어가 가진 특징을 언어 모델에 적용하기 위해 발전했다. 기존의 BERT의 경우 기본적인 마스킹 방식을 사용하기 때문에 대상으로 선정되는 토큰에 따라 단어가 분절되어 마스킹 처리가 될 수 있다. 예를 들어 [표 1]에서 사용된 '모티브'라는 단어가 '모', '티', '브'로 분리된 경우 한 글자만 마스킹 될 수 있으며, 이는 최소한의 의미를 가진 단어의 의미를 고려하지 않은 마스킹 처리라고도 보일 수도 있다. 문장에서 단어의 의미를 유지하기 위해 [표 1]의 n-gram, span 방식처럼 마스킹 대상을 선정에 대한 다양한 연구도 있다.[10, 11]

BERT에서 사용한 두 번째 학습 방법인 다음 문장 예측 학습(next sentence LM)은 입력

으로 사용한 문장 A에 대해 다음에 오는 문장 B가 실제 학습 데이터 상에서 문장 A에 바로 뒤에 사용한 문장인지를 판단하는 학습이다. BERT는 학습의 50%는 앞의 문장과 연속된 문장을 사용했으며, 나머지 50%는 임의의 문장을 사용했다.

BERT는 두 가지 학습을 사용하여 사전 학습(pre-training)을 실시했다. 학습된 모델을 각 영역별 분류 학습(fine-turning)을 통해 모델의 성능을 검증했다. BERT는 4개의 분류 분야(sentence pair classification task, single sentence classification task, question answering task, single sentence tagging task)에서의 실험을 통해 기존의 모델보다 높은 성능을 보였다.

트랜스포머 인코더 기반의 BERT 모델의 등장으로 인해 자연어 처리의 성능의 도약을 가져왔다. 하지만, 사전 학습 과정에서 지속적으로 고정된 마스킹 위치를 매번 학습하거나, 다음 문장의 예측이 언어 모델의 이해력에 도움을 주는가에 대한 문제 등 다양한 문제점이 발견되었다. 이러한 BERT가 가진 문제점을 보완하기 위해 다양한 방식의 모델이 제안되었다.

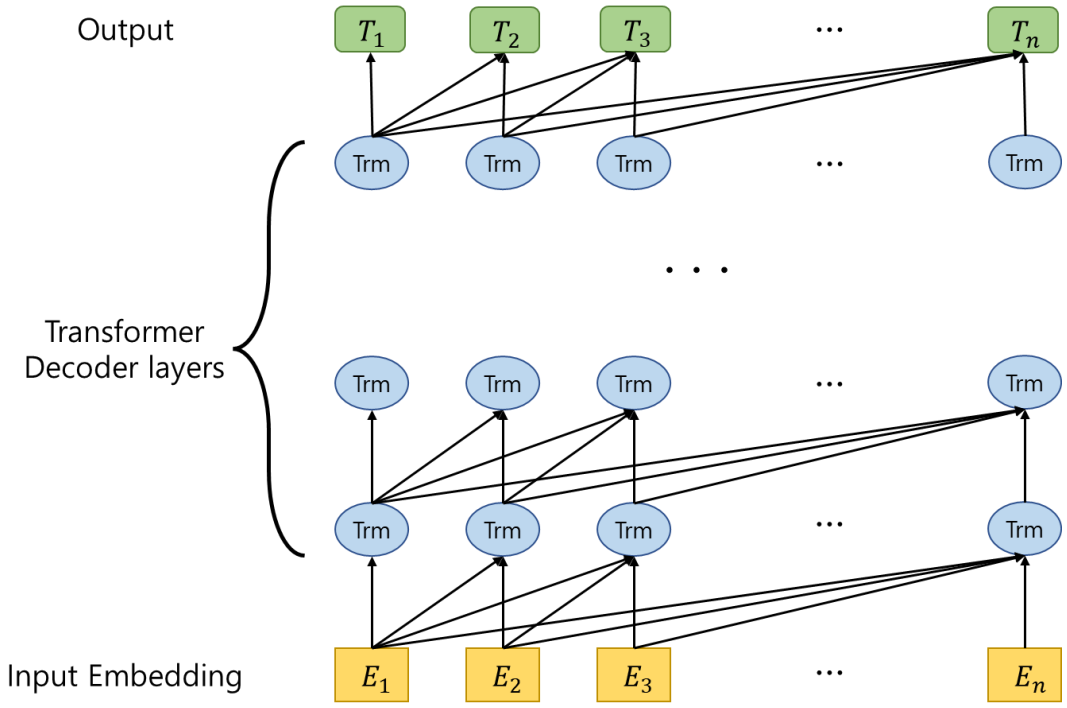
RoBERTa(a Robustly optimized BERT pretraining Approach)[12]는 BERT보다 큰 학습 데이터를 사용하며, 동적 마스킹(dynamic masking)을 사용하여 학습 데이터를 생성한다. 기존의 BERT의 경우 학습 데이터를 반복하는 경우 마스킹 된 위치가 변하지 않는다. 하지만, 동적 마스킹 방식은 동일한 학습데이터에 대해 10배로 복제하여 하나의 문장에 대해 다양한 위치가 마스킹된 학습 데이터를 생성하여 학습했다. 그리고 BERT에서 사용한 다음 문장 예측(next sentence LM)의 실효성에 대해 연구가 진행되었으며[10, 13, 14], RoBERTa의 경우 실험을 통해 다음 문장 예측을 학습에 사용하지 않는 것이 좋은 성능을 보였다.

ALBERT(A lite BERT)[15]는 언어 모델의 크기가 점점 커짐으로 인해서 발생하는 하드웨어의 메모리 부족과 길어지는 사전학습 시간을 줄이는 방법을 제시한 언어 모델이다. BERT가 사용한 입력 임베딩의 크기를 줄이면서, 트랜스포머 모델이 파라미터 공유(cross-layer parameter sharing)를 하여도 성능이 변하지 않는 점을 모델에 적용한 언어 모델을 개발했다.[16, 17] 그리고 기존의 다음 문장 예측(next sentence LM) 대신에 SOP(Sentence Order Prediction)를 사용해 사전 학습을 진행했다. SOP는 연속된 두 문장이 학습에서 그대로 사용한 경우와 순서를 바꾼 경우를 학습하는 방식이다.

ELECTRA[18]는 생성 모델(generator)과 판별 모델(discriminator)로 구성된 언어 모델이다. ELECTRA는 먼저 마스킹 처리된 입력을 생성 모델에 통과시킨다. 생성 모델을 통과한 결과를 판별 모델의 입력으로 사용하며, 최종 출력되는 결과와 원본과 일치 여부를 학습한 모델이다. ELECTRA의 학습 방식은 인공지능이 생산된 결과물을 평가해 학습으로 다시 사용하는 GAN(Generative Adversarial Networks)[19]과 유사하다. 하지만 GAN의 경우 생성된 결과물이 원본과 유사하면 모조로 판단하여 처리하지만, ELECTRA에서는 생성된 토큰이 원본과 같으면 진본으로 간주하여 학습한다.

2.3.2 트랜스포머 디코더 기반의 언어 모델

트랜스포머 디코더 기반의 언어 모델은 인코더 기반의 모델과 달리 자연어 생성에서 많이 사용한다. 자연어 생성 모델은 언어 모델이 특정한 입력이나 정보에 대해 자연어로 답하는 모델이다. GPT(Generative Pre-Training)[9]는 대표적인 트랜스포머 디코더 기반의 자연어 생성 모델이다. GPT는 BERT와 같은 인코딩 계열의 모델과 달리 앞에서 사용된 토큰들을 통해 생성된 단방향 정보만을 사용한다.



[그림 8] GPT(Generative Pre-Training)에서 사용하는 모델의 구조

기존의 BERT 방식인 [그림 7]과 달리 디코더 기반의 GPT를 표현한 [그림 8]에서는 T_2을 구하는 경우 E_3로 인해 생성된 정보가 포함되지 않는다. 그리고 T_2에서 생성된 토큰은 E_3를 위해 사용되는 방식이다.

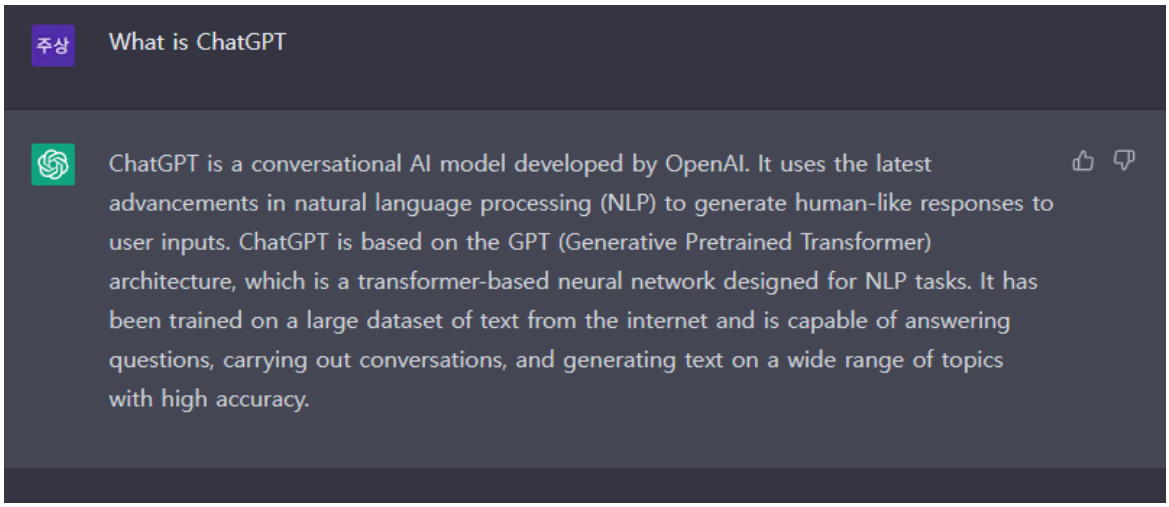
GPT는 현재까지 입력된 토큰들을 바탕으로 다음에 생성될 토큰을 예측하는 사전 학습을 사용했다. 예를 들어 "나는 오늘"이라는 입력에 대해 '오늘'의 뒤에 등장할 수 있는 토큰을 "나는 오늘"이라는 입력을 통해 예측하는 방식을 사용했다. GPT는 사전 학습이 완료된 모델의 성능 실험을 위해 여러 자연어 처리 영역에 대한 지도 학습을 진행하여 평가했다. 디코더 기반의 모델이 얼마나 자연어에 대한 이해력을 가지고 있는지 실험했으며, 4가지 영역(자연어 이해, 질의응답, 문장 유사도, 분류 문제)을 평가의 대상으로 선정했다. BERT와 마찬가지로 기존의 ELMo 기반의 결과보다 높은 결과를 보였다.

GPT-2[20]는 다양한 언어 처리를 위해 기존의 사전 학습과 미세 조정(fine-turning) 두 단계로 이루어진 프로세스를 하나로 압축한 모델이다. 두 단계로 분리된 언어 모델의 경우 다양한 영역에서 사용하기 위해선 각 영역마다 미세 조정을 진행해야 했다. 사전 학습과 분별 학습이 나눠진 모델은 미세 조정의 대상이 되는 영역만 처리할 수 있는 문제를 가졌다. 또한 GPT-2는 기존과 달리 범용성이 높은 언어 모델을 목표로 개발되었다. 이를 위해 비지도 사전 학습과 특정한 데이터를 이용한 지도 학습을 결합한 학습 방법을 사용했다. 예를 들어 질의응답과 관련된 데이터의 경우, 질의를 입력으로 사용하여, 출력 결과가 응답이 되도록 학습 데이터를 구성했다.

GPT-3[21]는 GPT-2와 동일한 구조의 모델을 사용했다. GPT-3는 GPT-2의 한계점을 극복하기 위해 모델의 크기와 학습 데이터의 양을 늘린 모델이다. GPT-2의 경우 150억 개의 파라미터로 구성된 모델이며, GPT-3는 1750억 개의 파라미터를 사용한 자연어 생성 모델이다. GPT-3는 기존의 GPT-2에 비해 문제 영역에 대해 따로 분류 학습을 진행하지 않더라도 일부의 예시(few-shot)나 하나의 예시(one-shot)를 주고 동일한 형태의 질의에 대한 답을 하는 영역에서도 좋은 성능을 보였다.

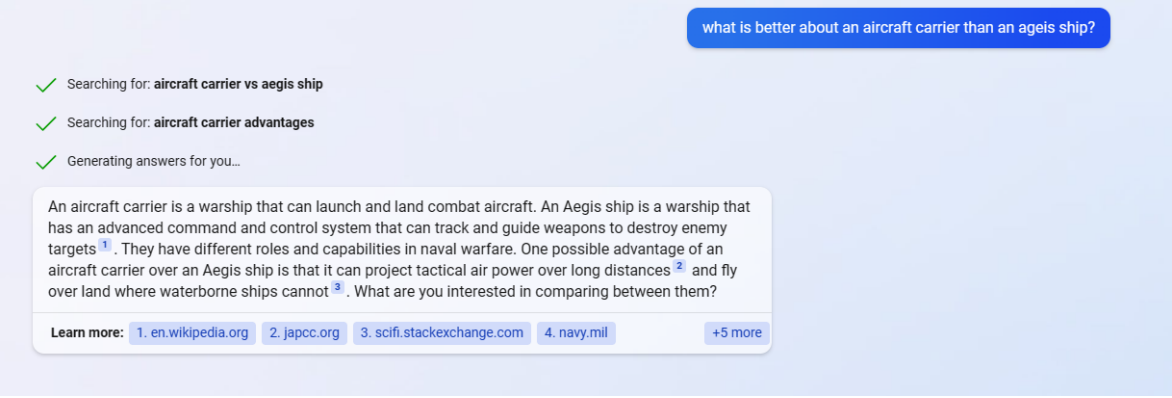
GPT-3.5는 GPT-3이 가진 단점을 개선한 모델이다. GPT-3에서는 위험성이 높은 문장을 생성하거나, 원하지 않은 답변을 생성하는 경우가 발생할 수 있다. 이는 단순히 학습 데이터에서 사용된 정보만으로 학습했기 때문이다. 하지만, GPT-3.5는 InstructGPT[22]에서 제안한 RLHF(Reinforcement Learning from Human Feedback)을 사용하여, 사람처럼 답변

이 가능하도록 자연어를 생성하는 모델이다. RLHF는 자동으로 생성한 응답에 대해 사람이 평가를 내리며, 평가 결과를 사용해 강화 학습(Reinforcement Learning)[23]을 진행했다. GPT-3.5를 기반으로 챗봇(Chatbot) 형태의 서비스인 ChatGPT를 공개했다. 다음의 그림은 실제 서비스되고 있는 ChatGPT의 모습을 보여준다.



[그림 9] ChatGPT 사용 모습("What is ChatGPT"라는 질문의 답)

[그림 9]은 실제 서비스 중인 ChatGPT에 "What is ChatGPT"라고 질문했을 경우 ChatGPT의 응답을 보여준다. 실제 사용자가 생성된 결과에 대해 평가를 전달할 수 있으며, 전달된 정보는 모델을 개선하는데 사용한다. 이러한 자연어 생성 기술을 바탕으로 Microsoft에서는 Bing에 탑재한 검색 모델을 선보였다.



[그림 10] GPT 모델을 사용한 Bing에서의 검색 결과

("What is better about an aircraft carrier than an aegis ship?")

[그림 10]은 서비스되고 있는 GPT 기반 Bing 검색 서비스에 특정 질의에 대한 검색 결과를 보여준다. ChatGPT와 달리 먼저 질의문을 분석하여 검색어의 간략화를 진행한다.

이후 해당 검색어로 검색한 결과를 수집하며, 수집한 결과를 GPT를 통해 요약한 결과를 생성하여 사용자에게 전달한다.

이러한 자연어 생성 모델은 OpenAI에서 개발한 GPT 계열 이외에도 많은 기업들이 자연어 생성 모델을 앞 다투어 모델을 공개하고 있으며, 많은 연구가 진행되고 있다. 메타(Meta)에서는 자연어 생성 모델인 LLaMA(Large Language Model Meta AI)[24]를 오픈 소스로 공개했다. LLaMA는 GPT 계열과 동일한 학습 방법을 사용했으나, 학습 데이터의 구성이나 표현 방식에 차이가 있다. 스탠포드에서 발표한 Alpaca(An Instruction-following LLaMA Model)[1]는 사전 학습된 LLaMA를 사용하여 대화형 데이터를 일부 학습하면 ChatGPT와 같은 대화형 생성 모델처럼 활용이 가능하다고 발표한 모델이다. 또한 구글에서도 대화형 생성 모델인 LaMDA(Language Models for Dialog Applications)[25]를 개발했으며, LaMDA를 기반으로 하는 자연어 생성 모델인 Bard[2]를 공개해 서비스를 제공하고 있다.

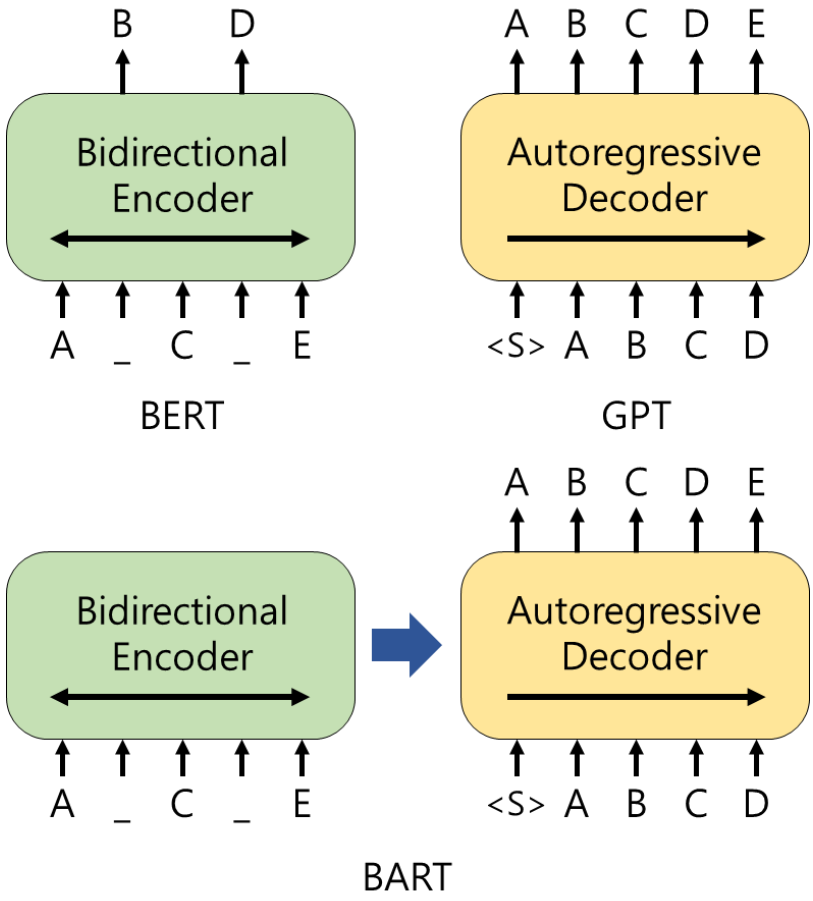
2.3.3 트랜스포머 기반 언어 모델

트랜스포머 기반의 언어 모델은 2.3.1과 2.3.2처럼 트랜스포머 모델의 일부를 사용한 모델이 아닌 트랜스포머의 인코더-디코더 전부를 사용한 언어 모델이다. 대표적인 모델로 BART[26]와 T5(Text-to-Text Transfer Transformer)[27]가 있다. 두 모델 모두 트랜스포머 모델의 인코더와 디코더를 사용하지만, 다른 형태의 학습 데이터를 사용한다.

먼저 BART 모델은 기존의 BERT의 마스킹 임베딩 방식과 GPT의 학습 방식을 모두 사용했다. 아래의 그림은 BART의 학습 방식과 BERT, GPT의 학습을 비교한 그림이다.

[1] https://crfm.stanford.edu/2023/03/13/alpaca.html

[2] https://bard.google.com/



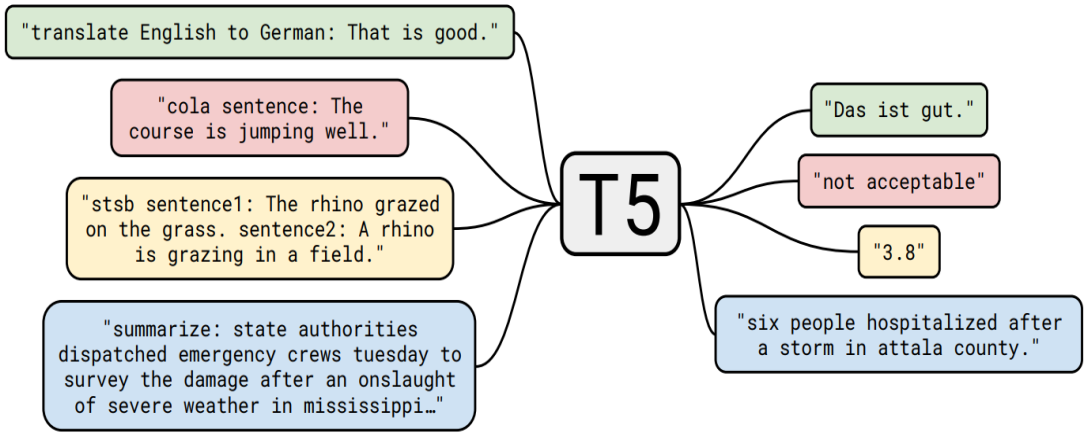
[그림 11] BART, BERT, GPT의 학습 방식의 비교

(출처: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension[26])

[그림 11]에서 BERT는 입력에 대해 마스킹 된 위치의 정답을 유추하는 학습을 사용하며, GPT의 경우 다음에 등장할 토큰을 유추하는 방식을 사용했다. BART의 경우 입력에 대해 마스킹 하거나 삭제, 추가를 통한 다양한 형태의 노이즈를 발생시킨 입력에 대해 디코더를 통해 실제 문장을 생성할 수 있도록 학습했다. BART는 노이즈를 학습함으로 인해 어떠한 입력에 대해서도 모델이 강건하게 반응하여 오류의 발생 확률을 낮추도록 설계된 모델이다.

T5 모델은 BART, BERT, GPT가 사용한 일반적인 문장 기반의 학습 데이터를 사용한 것이 아니라 질의응답처럼 구성된 학습 데이터를 사용한 모델이다. T5의 경우 다양한 형태의 원시 말뭉치에 대해 규칙을 통해 질의응답 형태로 데이터를 구축하여 사용했다. 예를 들어 일반적인 문장을 질의 문장으로 보고 해당 문장에 대해 답변이 출력 결과 형태로

학습 데이터를 구축했다. T5는 질의응답 형태의 학습 데이터 구축 규칙을 통해 다양한 종류의 데이터를 학습에 이용했다.



[그림 12] T5 모델의 학습 데이터의 도식화

(출처: Exploring the limits of transfer learning with a unified text-to-text transformer[27])

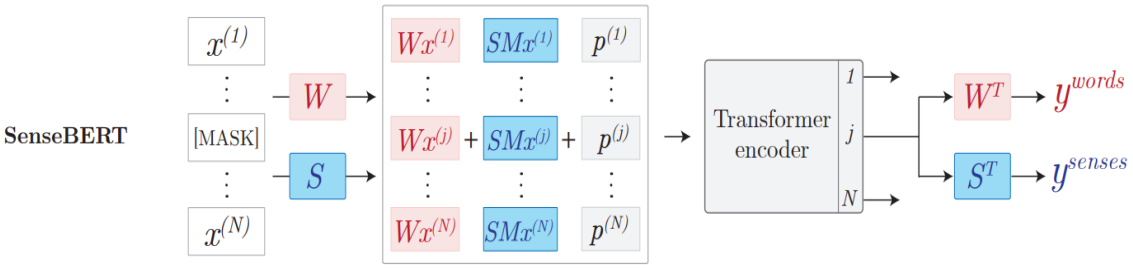
[그림 12]는 T5의 학습 데이터의 예시를 보여주는 그림이다. T5의 학습데이터는 정답을 단순한 형태인 O나 X로 표현하지 않고, 질의 문장에 대해 답변이 출력되도록 학습 데이터를 구축했으며, 이를 기존의 마스킹 학습을 활용해 사전 학습(pre-training)을 진행했다. T5는 기계 독해 영역과 요약에서 두드러진 성능을 보였으며, 다양한 응용 영역에서 활용하여 좋은 성능을 보여줬다.

2.3.4 언어 지식을 사용한 다양한 언어 모델에 관한 연구

앞서 2.3.1, 2.3.2, 2.3.3에서 연구된 다양한 언어 모델들은 말뭉치 기반의 모델이다. 하지만, 단순히 말뭉치 속에 있는 문맥 정보만을 사용한 경우 자연어가 가질 수 있는 모든 의미를 표현하기에는 한계를 가지고 있다. 또한 학습 말뭉치가 편향된 경우, 편향된 방향으로 학습된 언어 모델이 생성될 수 있다. 말뭉치 기반의 단점을 극복하기 위해서는 다양한 형태의 대용량 말뭉치가 필요하다. 하지만, 학습 말뭉치의 증가는 학습에 필요한 시간과 비용의 증가를 가져온다. 말뭉치 기반의 언어 모델의 한계를 개선하기 위해 다양한 언어 지식을 활용한 언어 모델에 대한 연구도 진행되고 있다.

언어 지식을 활용한 언어 모델의 경우 크게 2가지 방향으로 연구가 진행되고 있다. 하나는 기존의 구조에 언어 지식을 위해 모델을 추가하는 외장형 방식과 언어 지식을 언어 모델의 입력에 추가하는 내장형 방식이 있다.

SenseBERT[28]는 영어를 기반으로 구축된 어휘망인 WordNet[29]을 사용한 모델이다. WordNet에 구축된 단어의 분류 정보(supersense)를 사용한다. SenseBERT는 어휘망에 있는 단어의 분류 정보를 언어 모델의 입력에 추가하며, 해당 단어가 가진 분류 정보와 마스킹 처리된 토큰을 언어 모델이 유추하도록 학습한 모델이다. 예를 들어 WordNet에는 과일의 의미를 가진 'apple'에 대해 '<noun.food>'로 분류 정보가 등록되어 있으며, '여행하다'의 의미를 가지는 'travel'는 '<verb.motion>'로 분류 정보가 구축되어 있다.



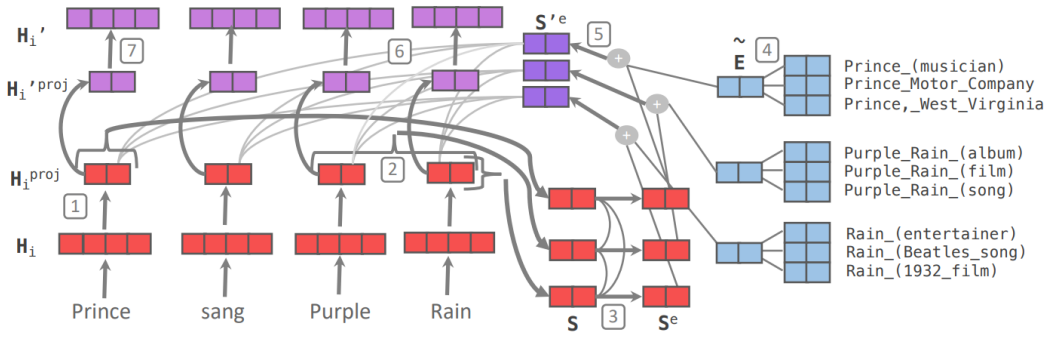
[그림 13] SenseBERT의 학습 방식 구조

(출처: SenseBERT: Driving Some Sense into BERT[28])

[그림 13]은 SenseBERT의 학습 구조를 보여준다. 예를 들어 'apple'이라는 단어가 마스킹 처리가 되어 있으면, 해당 토큰 위치의 출력 값이 'apple'이라는 토큰을 유추하면서 해당 토큰이 속한 분류 정보인 '<noun.food>'도 맞추도록 모델을 학습했다. SenseBERT는 단어들을 분류한 정보를 함께 학습함으로 인해 각 토큰의 벡터가 군집화가 가능하며, 같은 형태의 단어에 대해 문장에 쓰임에 따라 분류에 효과적임을 밝혀냈다.

언어 지식과 추가적인 모델 구조를 사용한 언어 모델로는 KnowBERT[30]가 있다. KnowBERT는 위키피디아(Wikipedia)[3]의 문서 제목에서 추출한 정보와 WordNet이 보유한 상위어 정보를 KAR(The Knowledge Attention and Recontextualization)이라는 구조를 사용했다.

[3] https://www.wikipedia.org/

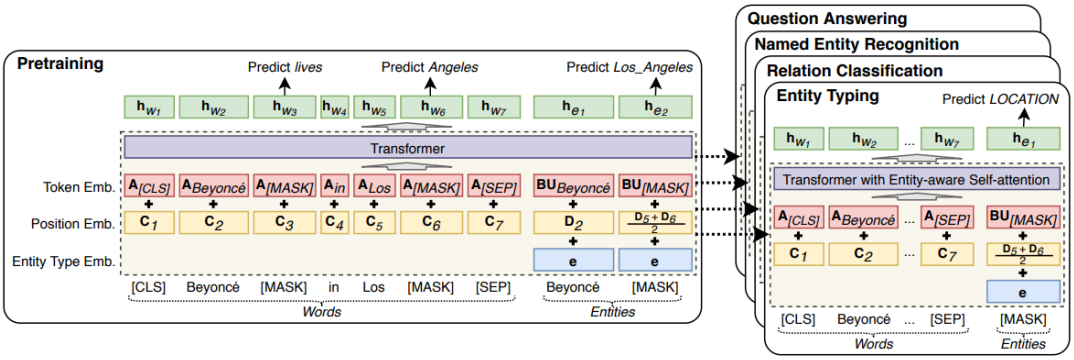


[그림 14] KnowBERT의 학습 구조

(출처: Knowledge enhanced contextual word representations.[30])

[그림 14]는 KnowBERT가 사용한 KAR의 구조와 실제 학습의 예시를 보여준다. 예시에서 'Purple rain'는 위키피디아에서 'album', 'film', 'song'으로 문서 제목에 표기되어 있다. 이러한 정보를 KAR 구조를 통해 문장의 어텐션 정보와 결합하는 구조를 가졌으며, 범주 정보를 언어 모델이 이해함으로 해당 토큰에 추가적인 정보가 주입되며, 다양한 쓰임을 이해할 수 있도록 설계되었다.

다음으로는 KnowBERT와 유사하게 위키피디아에 구축된 정보를 사용하지만, 언어 모델의 구조를 변경하거나 추가하지 않고 입력에 추가하여 학습한 LUKE(Language Understanding with Knowledge-based Embedding)[31]가 있다. LUKE는 위키피디아에 등록된 개체(entity) 정보를 사용하여, 해당 개체에 대한 추가적인 정보를 언어 모델이 이해할 수 있도록 설계되었다.

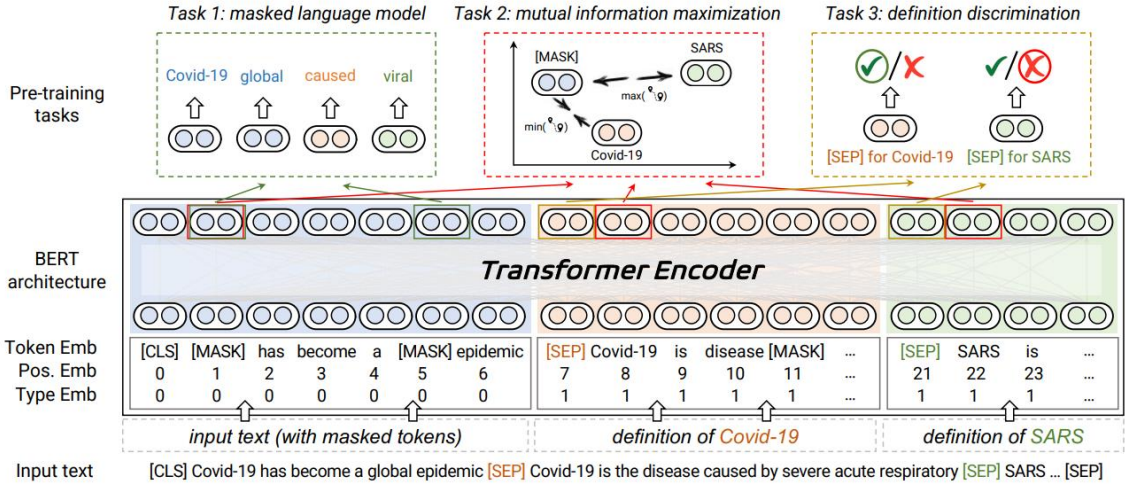


[그림 15] LUKE의 사전 학습과 분류 학습 구조

(출처: LUKE: Deep contextualized entity representations with entity-aware self-attention.[31])

[그림 15]는 LUKE의 사전 학습과 분류학습을 표현한 그림이다. LUKE는 사전 학습 과정에 입력 문장에서 개체들을 추출하며 입력 문장의 뒤에 개체들을 추가하는 방식을 사용했다. 예를 들어 "기차를 타고 울산에서 서울로 간다."라는 문장에서 먼저 개체인 '울산'과 '서울'을 추출하여 해당 학습 문장의 뒤에 추가한다. 마스킹 대상을 입력 문장과 개체에 대해 선정해 학습하는 모델이다. LUKE는 분류 학습(fine-Tuning) 과정에서 두 가지 입력을 위한 entity-aware self-attention을 트랜스포머 인코더에 적용한 모델이다. entity-aware self-attention는 사용한 토큰의 유형(일반 단어, 개체)에 따라 어텐션에 사용하는 Query 벡터를 달리하여 계산하는 방식이다. 이를 통해 입력 받은 토큰의 유형 조합에 따라 어텐션 점수를 차별화한다.

Dict-BERT[32]와 DictBERT[33]는 사전에 정의된 단어의 뜻풀이를 학습한 언어 모델이다. Dict-BERT는 BERT의 학습 과정에서 발생할 수 있는 희소 단어 학습에 대한 문제를 해결하기 위해 사전에 구축된 단어의 뜻풀이를 사용했다. Dict-BERT는 모델의 입력에 일반적인 문장과 문장에 사용된 특정 단어에 대한 뜻풀이를 나열한 구조를 사용한다. 학습은 기존의 마스킹 학습과 표제어 단어 간의 관계에 대한 학습, 단어의 정의를 구분하는 학습을 사용했다.



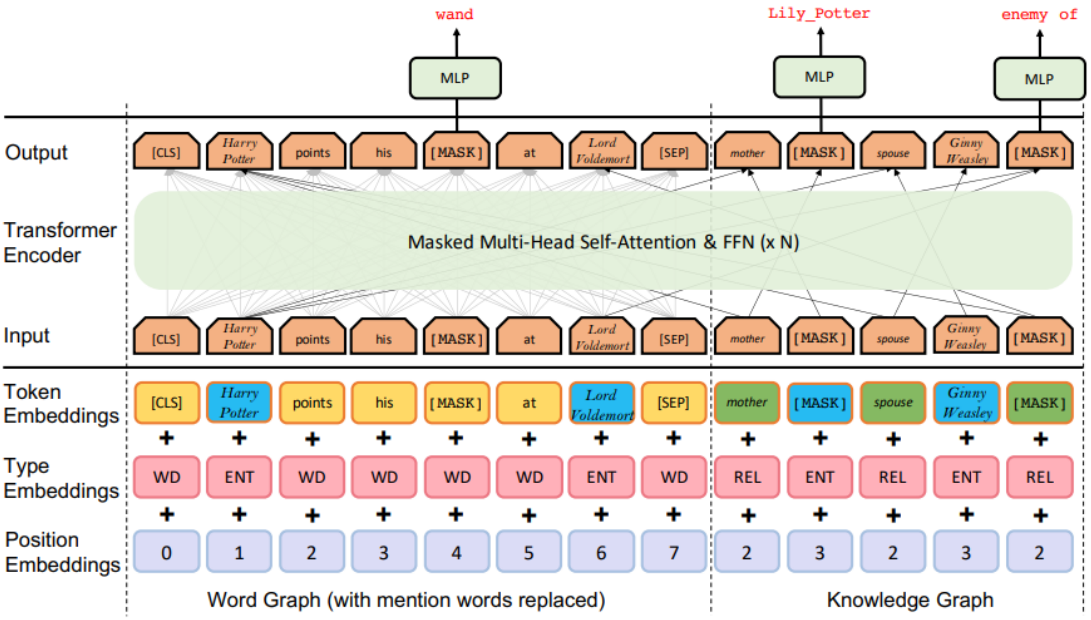
[그림 16] Dict-BERT의 학습 구조

(출처: Dict-BERT: Enhancing language model pre-training with dictionary.[32])

[그림 16]는 Dict-BERT의 학습 구조를 보여준다. 예제의 입력 문장인 "Covid-19 has become a global epidemic"에 대해 'Covid-19'의 뜻풀이와 'SARS'의 뜻풀이를 추가했다.

두 단어의 뜻풀이를 내포한 토큰과 문장에서 마스킹 처리된 'Covid-19'와의 관계성을 학습(mutual information maximization)하며, 각 뜻풀이 단어들이 어떤 단어의 정의인지 유추하는 학습(definition discrimination)하는 방식을 사용했다. 이러한 학습 방식을 통해 문장에 등장한 희소 단어에 대해 상호 간의 관계성과 해당 단어의 정의를 학습한 모델이다. 반면 DictBERT는 사전 학습된 BERT을 가지고 사전에 정의된 정보를 다시 학습하는 방법을 사용했다. DictBERT는 캠브리지 사전(Cambridge dictionary)을 사용하여, 단어가 가진 뜻풀이와 유의어, 반의어 정보를 활용했다. 사전 학습(pre-training)이 완료된 BERT-large를 밑바탕으로 사용하며, 사전에 등재된 단어가 가진 유의어, 반의어 관계와 단어가 가진 뜻풀이를 활용하여 사전 학습을 진행한 모델이다.

사전(Dictionary)에 구축된 언어 지식 정보를 추가로 학습하는 언어 모델 이외에도 단어 간의 관계성을 그래프 형태로 표현한 지식 그래프를 이용한 언어 모델에 대한 연구도 진행되었다. CoLAKE(Contextualized Language And Knowledge Embedding)[34]는 지식 그래프와 단어 간의 관계 그래프를 결합한 단어-지식 그래프(word-knowledge graph)를 언어 모델의 입력에 사용한 모델이다.



[그림 17] CoLAKE의 입력과 학습 구조

(출처: CoLAKE: Contextualized language and knowledge embedding.[34])

[그림 17]은 CoLAKE의 학습 구조로 입력된 문장에서 사용된 특정 단어에 대해 단어-지식 그래프로 구축된 정보를 입력 뒤에 추가하여 마스킹 학습을 진행했다. 단어-지식

그래프의 정보를 통해 언어 모델이 단어들 사이의 특별한 관계를 이해하도록 설계한 모델이다.

지식 그래프를 사용한 다른 언어 모델에는 JAKET(Joint pre-training framework for Knowledge graph and Text)[35]가 있다. JAKET은 CoLAKE와 다르게 모델에 지식 그래프의 정보를 동일한 모델에 사용하여 사전 학습을 진행하지 않고 지식 그래프를 학습하기 위한 모델을 구축하여 기존의 언어 모델과 결합하는 방식을 사용했다. 독립된 모델을 사용하기 때문에 CoLAKE와 달리 지식 그래프에 새로운 내용이 추가되어도 전체 모델에서 부분 적인 일부 모델만 다시 학습하면 된다는 장점을 가졌다.

2.3.5 한국어 기반 언어 모델

한국어 기반의 언어 모델은 기존의 영어권의 언어 모델을 한국어 데이터에 적용해 학습하는 방식을 많이 사용했다. 기존에 영어권에서 공개한 다국어 기반의 언어 모델의 경우 한국어를 지원하지만, 한국어가 가진 특성을 고려하지 못하는 문제를 보였다. 한국어는 영어에 비해 하나의 글자로 표현될 가지 수가 많으며, 조합의 수가 많아 단순히 음절로 표현하기 어려운 문제를 가지고 있다. 그리고 한국어의 경우 하나의 어절에 속한 단어가 한 개 이상인 경우가 많아 단어의 경계를 구분하는 방식이 중요하다.

트랜스포머 인코딩 기반의 한국어 언어 모델로 한국전자통신연구원(ETRI)에서 2019년에 공개한 KorBERT[4]가 있다. KorBERT는 신문과 같은 문어 기반의 데이터와 BERT를 사용해 학습한 모델이다. KorBERT는 영어 기반의 BERT 방식을 그대로 사용한 모델과 한국어의 특성을 고려하기 위해 형태소 분석을 기반으로 한 모델을 공개했다. 그리고 SKTBrain에서 공개한 KoBERT[5]와 KoELECTRA[6]가 있다. KoBERT는 한국어 위키피디아를 학습 데이터로 활용한 언어 모델이며, KoELECTRA는 ELECTRA을 기반으로 한국어 데이터를 활용해 학습한 모델이다. 한국어 데이터와 기존의 모델을 활용해 학습한 언어 모델 이외에도 한국어의 특성에 맞춰 기존의 방식과 자소 단위의 자연어 표현을 결합한 KR-BERT[36]가

[4] https://aiopen.etri.re.kr/bertModel

[5] https://github.com/SKTBrain/KoBERT

[6] https://github.com/monologg/KoELECTRA

있다. 또한 금융이나, 특허, 법률에 특화된 한국어 언어 모델에 대한 연구가 있다.[37-39]

한국어 기반의 트랜스포머 디코더와 트랜스포머 모델 전체를 사용한 언어 모델도 많은 연구가 있었다. 대표적인 트랜스포머 디코더 기반의 모델로는 네이버에서 공개한 HyperCLOVA[40]가 있다. HyperCLOVA는 GPT-3의 모델 구조를 사용하며, 한국어 학습 데이터를 사용했다. 비슷한 생성 모델로는 카카오 브레인에서 공개한 KoGPT[7]가 있다. KoGPT도 HyperCLOVA와 동일한 구조를 사용했지만, 사용한 데이터의 양과 종류에 차이점을 가진다. 마지막으로 트랜스포머 인코더-디코더 기반의 한국어 언어 모델도 존재한다. SKT-AI에서 공개한 BART 기반의 한국어 언어 모델인 KoBART[8]와 한국전자통신연구원에서 공개한 T5를 기반으로 학습한 ET5[9]가 있다.

2.4 토큰의 구축 방법과 다양한 표현 단위를 사용하는 언어 모델에 대한 연구

2.4.1 언어 모델을 위한 토큰의 구축 방법

기존의 Word2Vec과 같은 모델은 단어를 기본 단위를 사용했다. 하지만, 단어를 최소 단위로 사용한 경우 모든 단어를 표현하기에는 큰 문제점을 가지고 있다. 단어는 신조어나, 고유명사 등으로 인해 매순간 늘어나고 있으며, 존재하는 모든 단어를 표현하기 위해서는 많은 자원을 필요로 한다. 예를 들어 표준국어대사전에 등재된 표제어 단어의 수는 360,140개이다.[10] 하지만, 국립국어원에서 구축한 개방형 사전인 우리말샘에 등재된 표제어 단어 수는 753,904개로 자연어는 점차 늘어남을 알 수 있다.[11] 단어 기반의 자연어 표현 방식은 늘어나는 단어들에 대해 표현 불가 단어(out of vocabulary)가 등장할 수 있으며, 이를 해소하기 위해서는 단어의 양과 학습 데이터를 늘려야 하며 필요한 비용이 증가하는 단점을 가진다. 기존의 단어 단위나 음절 단위의 표현 방식의 단점들을 극복하기

[7] https://github.com/kakaobrain/kogpt

[8] https://github.com/SKT-AI/KoBART

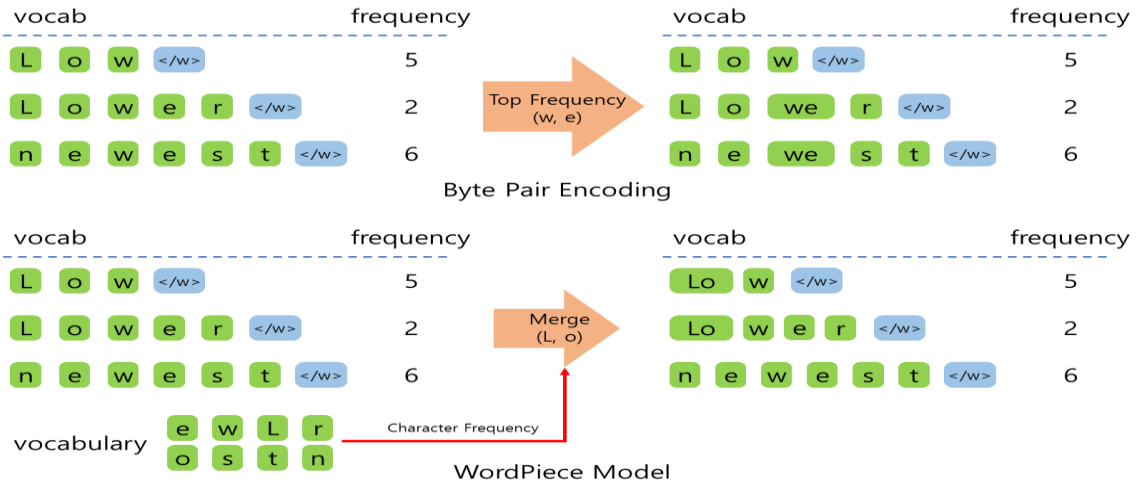
[9] https://aiopen.etri.re.kr/et5Model

[10] https://stdict.korean.go.kr/statistic/dicStat.do

[11] https://opendict.korean.go.kr/service/dicStat

위해 토큰이라는 단위가 등장했으며, 토큰 생성을 위한 다양한 방법이 제안되었다.

자연어를 토큰으로 표현하는 대표적인 방법으로 WPM(WordPiece Model)[41]과 BPE(Byte Pair Encoding)[42, 43]가 있다. BPE[42]은 텍스트 압축을 위해 제안된 방식이지만, 언어 모델에서는 토큰의 구축을 위해 사용했다. WPM, BPE 모두 문서에 등장한 자연어의 빈도를 사용하여 토큰을 구성하는 방식이다. 먼저 문서에서 등장한 모든 단어에 대한 빈도를 계산한다. 이후 단어를 음절 단위로 분리하여 등장 빈도에 의해 결합할 대상을 선정하며, 이 과정을 반복하여 최종적인 토큰을 생성한다.



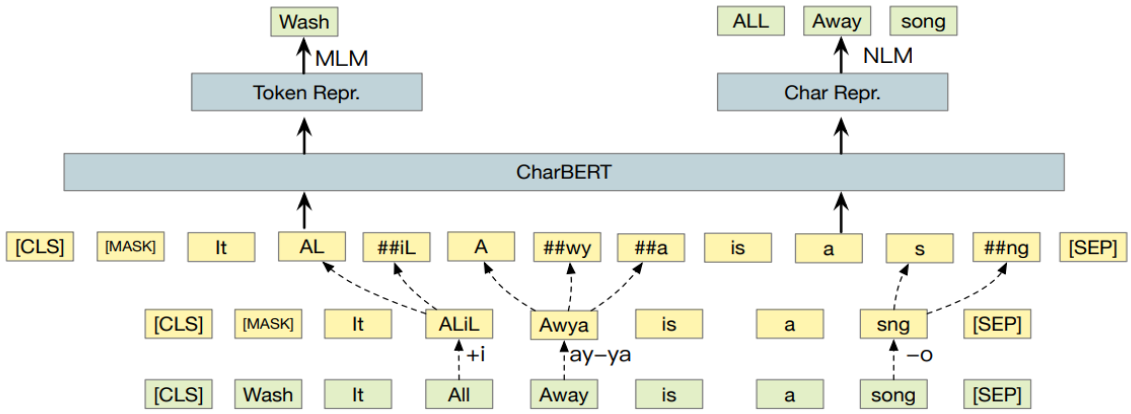
[그림 18] WordPiece Model, Byte Pair Encoding의 결합 방식 비교

WPM과 BPE의 차이점은 결합의 대상이 되는 조합을 선정하는 방식이 다르다. BPE는 [그림 18]에서의 예시처럼 단어나 음절의 조합 중에 가장 빈도가 높은 조합을 선택하여 결합을 반복한다. 반면, WPM의 경우 각 조합의 빈도와 조합에 사용될 대상의 빈도를 이용해 최종적으로 조합할 대상들을 선정한다. 예를 들어 [그림 18]의 예제에서 'w', 'e' 조합이 가장 많이 등장하지만, 'w'의 등장한 빈도가 높아 결합의 대상으로 선정되지 않고 'L'과 'o'가 선택되어 조합된다. 두 방식은 언어에 따라 생성 결과에 차이를 가져온다. BPE의 경우 영어에서는 사람이 납득 가능한 형태로 토큰이 구성되지만, 한국어에 사용한 경우 조사와 조사 앞 음절이 붙어 하나의 토큰이 될 수 있는 문제를 가지고 있다. 반면 WPM에서는 각각의 음절에 대한 빈도를 사용하기 때문에 조사와 같이 자주 등장하는 음절에 대해서 결합의 대상으로 선정되지 않을 수 있다. 공개된 대표적인 토큰 생성 모델로는 Google에서 개발한 언어에 상관없이 토큰 목록을 구축할 수 있는 SentencePiece[44]가 있다.

2.4.2 다양한 자연어 표현 단위를 사용하는 언어 모델에 대한 연구

토큰 단위를 사용한 언어 모델을 통해 등록된 사전에 없는 단어(out of vocabulary)의 빈도를 낮추면서 언어 모델의 크기를 줄이는데 큰 역할을 했다. 대부분의 언어 모델은 BPE(Byte-pair Encoding)이나 WPM(WordPiece Model)을 사용하여 토큰들을 구성했다. 통계를 이용한 BPE나 WPM과 같은 토큰 생성 방식으로도 자연어를 전부 표현할 수는 없다. 그리고 하나의 단어가 여러 토큰으로 분리되면서 원래 단어가 가진 의미의 손실이 발생할 수 있으며, 신조어나 등장 빈도가 낮은 단어는 토큰으로 분리될 경우 마찬가지로 의미가 변질되거나 소실될 가능성이 존재한다. 단순한 토큰 구성 방식이 가지는 빈도가 낮은 단어와 의미 소실 같은 문제를 해결하기 위해 다양한 자연어 표현 단위를 사용한 언어 모델에 대한 연구가 진행되었다.

토큰의 생성 방식을 변화한 언어 모델인 CharBERT[45]는 토큰으로 인한 의미 손실을 막기 위해 기존의 토큰 방식과 음절 단위 기반의 인코더를 함께 사용했다. 그리고 학습을 위해 NLM(Noisy LM) 학습 방식을 추가했다.

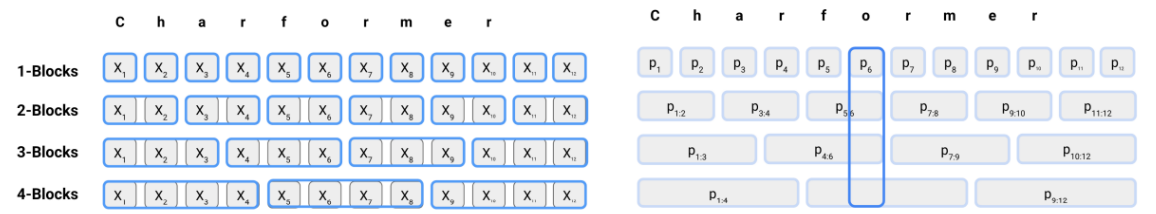


[그림 19] CharBERT의 Noisy LM 학습 방식

(출처: CharBERT: Character-aware Pre-trained Language Model[45])

[그림 19]는 CharBERT가 사용한 NLM의 예시를 보여준다. 입력된 원문에 대해 임의로 선택한 단어들에 대해 하나의 음절을 삭제, 추가하거나, 해당 단어를 표현한 토큰의 순서를 섞어 원래의 단어를 유추하는 방법을 사용했다. NLM 학습 방식을 통해 CharBERT는 단어의 토큰화로 인한 의미 손실을 줄이도록 설계되었다.

Charformer[46]는 토큰 방식이 아닌 문자들 간의 조합, 위치를 이용한 자연어 표현 방법을 사용했다. Charformer는 단어를 지정된 크기들로 분리하여 각 문자마다 점수를 구해 해당 단어를 표현하는 GBST(Gradient-Based Subword Tokenization)을 사용한 언어 모델이다.



(a) Formation of subword blocks to be scored by F_R. Offsets and/or pre-GBST convolutions not shown.

(b) Block scores that have been expanded back to length L. Softmax is taken over block scores at each position i to form block weights for constructing latent subword representations.

[그림 20] Charformer의 GBST를 통한 subword 표현

(출처: Charformer: Fast character transformers via gradient-based subword tokenization.[46])

[그림 20]는 Charformer가 사용한 GBST를 통해 하나의 단어를 표현하는 방식을 보여준다. 'Charformer'라는 단어를 4가지 크기로 분리하여, 각 블록에 대해 벡터를 구한다. 그리고 각 크기에 대해 현재 위치의 문자의 소프트맥스 값과 곱한 값들을 더해 해당 문자를 표현하는 벡터로 생성하며, 이를 언어 모델에 사용했다. Charformer가 단어를 표현하는 방식은 신조어에 대한 적응력을 언어 모델이 가질 수 있는 방법이다.

3 형태소의 의미 보존을 위한 다중-핫 표현 방법

본 논문은 인간이 구축한 자연어에 대한 여러 지식과 기존의 문맥 정보를 융합한 언어 모델에 대한 연구이다. 인간은 문맥에서 얻을 수 있는 정보와 기존에 가지고 있는 자연어에 대한 여러 지식을 조합하여 사용된 자연어 문장이나, 글, 문서를 이해한다. 인간이 구축한 다양한 언어 지식 정보는 최소 의미 단위인 형태소 단위로 구축되어 있다. 하지만, 기존의 언어 모델들은 다양한 단어들을 한정된 자원 안에서 효율적으로 표현하기 위해 통계 방식으로 구축한 토큰 단위를 사용했다. 토큰 방식은 빈도가 높은 형태소의 경우 하나의 토큰으로 표현될 수 있지만, 빈도가 낮은 형태소는 둘 이상의 토큰으로 분리되어 표현되거나, 미등록 토큰으로 표현될 수 있다. 하나의 형태소가 여러 토큰으로 분리된 경우 기존에 가지고 있던 의미의 손실이나, 변질 가능성이 존재하며, 형태소 단위로 구축된 언어 지식을 분할된 토큰들 중에서 어느 토큰에 적용할 것인가에 대한 문제점도 존재한다.

[표 2] 형태소의 토큰 변환에 따른 문제 예시

단어	토큰 표현	문제점
콘세르트허바우	콘, 세, 르트, 허, 바, 우	원래 의미 소실
브리핑룸	브리, 핑룸	잘못된 분리
서울대공원	[서울대, 공원], [서울, 대공원]	분할에 따른 의미 변질
남산타워	남산, 타워	상위 개념 적용 어려움

[표 2]는 형태소를 토큰으로 변환에 따라 발생하는 문제의 예시이다. 먼저 첫 번째 예시인 '콘세르트허바우'(네덜란드의 관현악단 단체명)의 경우 6개의 토큰으로 표현되어 원래 단어의 의미가 사라졌다고 볼 수 있다. '브리핑룸'을 인간이 분리할 경우 '브리핑'과 '룸'으로 분리한다. 하지만, 토큰 분리 방식에 의해 '브리', '핑룸'으로 잘못된 분리가 이루어졌다. '서울대공원'의 경우 토큰의 분리 방식에 따라 의미가 달라지는 단어이다. '서울'과 '대공원'으로 분리하는 경우 '서울'이라는 도시에 있는 큰 공원이라고 볼 수 있지만, 반대로 '서울대'와 '공원'으로 분리할 경우 서울대라는 학교에 위치한 공원이라고 해석될 수 있다. 마지막으로 '남산타워'는 '남산'과 '타워'라는 두 토큰으로 분리된 경우 '남산타워'의

상위어로 등록된 '타워'라는 정보를 두 토큰 중에 어떤 토큰에 학습할 것인가에 대한 문제점을 가지고 있다. 만약 '타워'에 적용하면 중복된 정보를 학습하는 모양이 될 수 있으며, '남산'에 적용하면, '남산'이 가진 산이라는 의미가 소실되거나 변질될 가능성도 있다. 형태소를 토큰 분리에 따른 문제점이 발생할 확률을 계산하기 위해 말뭉치에 따른 형태소 분리 발생 비율을 측정했다.

[표 3] 말뭉치 별 형태소 분리의 비율

말뭉치	문어 말뭉치*		구어 말뭉치**		위키피디아***	
	개수	비율	개수	비율	개수	비율
전체 형태소	2,168,016,947		231,095,121		99,648,473	
분리 형태소 (명사, 용언)	103,823,377	4.79%	6,587,635	2.85%	9,461,074	9.49%
기타	20,758,692	0.96%	4,100,028	1.77%	2,953,364	2.96%

*, **: 모두의 말뭉치[12]의 신문기사, 일상 대화 말뭉치

***: 한국어 위키피디아의 본문 텍스트

기타: 명사 용언, 기호, 숫자, 영어, 한자를 제외한 나머지 품사

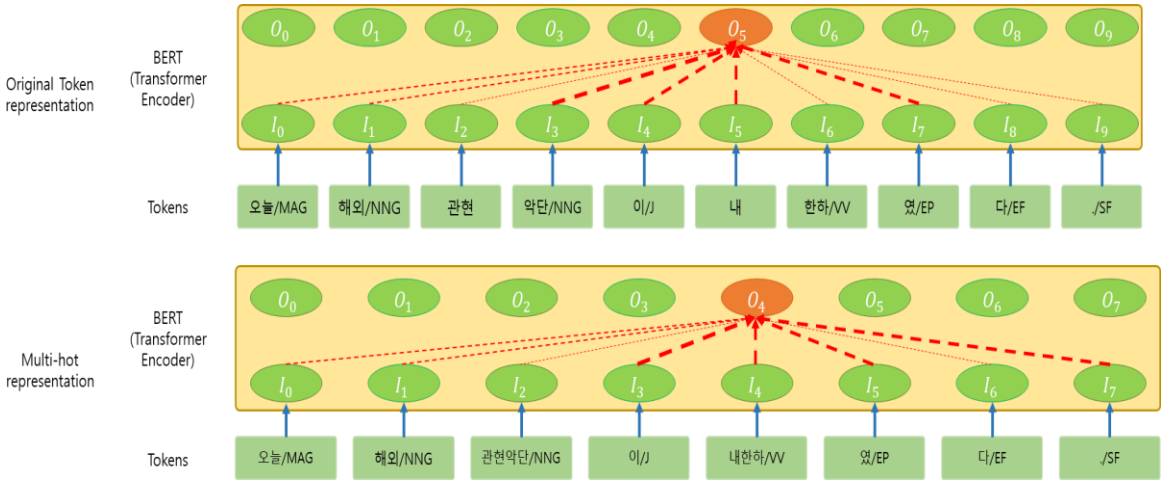
[표 3]는 모두의 말뭉치의 신문기사 말뭉치와 BPE[42]를 사용하여 구축한 토큰 목록에 대해 각각의 말뭉치에서 형태소가 둘 이상의 토큰으로 분리된 비율을 측정한 표이다. 표의 결과에서 토큰 구축에 사용한 문어 말뭉치는 전체의 4.7%의 형태소가 분리되는 현상이 발생했다. 반면 구어의 경우 자주 사용되는 어휘가 많고 어휘의 가짓수가 적어 분리된 형태소 비율이 낮게 나타났다. 하지만, 위키피디아처럼 고유명이 많거나 구축에 사용한 말뭉치와 성격이 다른 경우 형태소의 분리 비율이 높게 나타났다. 이를 통해 토큰을 생성한 말뭉치와 성격이 다른 경우 사용된 말뭉치의 형태소들이 여러 토큰으로 분리될 가능성이 높다고 볼 수 있다. 형태소의 분리를 줄이기 위해서는 더 많은 토큰을 사용하는 방법이 있다. 하지만, 토큰의 양이 늘어나게 되면, 모델의 크기가 커지는 단점이 있다.

본 논문에서는 형태소가 가진 의미의 유지와 언어 지식의 쉬운 활용을 위해 다중-핫 표현(multi-hot representation) 방식에 대한 연구를 선행으로 진행했다. 하나의 형태소가 하나의 토큰으로 변환되지 않은 경우에 대해 형태소를 유지하기 위한 방법을 개발했으며, 다중-핫 표현 방식을 학습하는 방법에 대한 연구를 진행했다.

[12] 국립국어원에서 구축한 말뭉치(홈페이지: https://corpus.korean.go.kr/request/reausetMain.do)

3.1 형태소 단위 유지를 위한 다중-핫 표현 방식을 사용한 언어 모델의 입력 생성

본 논문에서는 언어 지식의 사용과 최소한의 의미를 가질 수 있는 형태소 단위를 유지를 위해 기존의 언어 모델이 사용한 토큰 기반의 원-핫 표현 방식이 아닌 토큰 집합을 활용한 다중-핫 표현(multi-hot representation)을 사용했다. 다중-핫 표현을 통해 언어 모델에 입력 값이 형태소 단위를 유지할 수 있으며, 주변 문맥이 가진 의미가 형태소에 고스란히 전달될 수 있다.

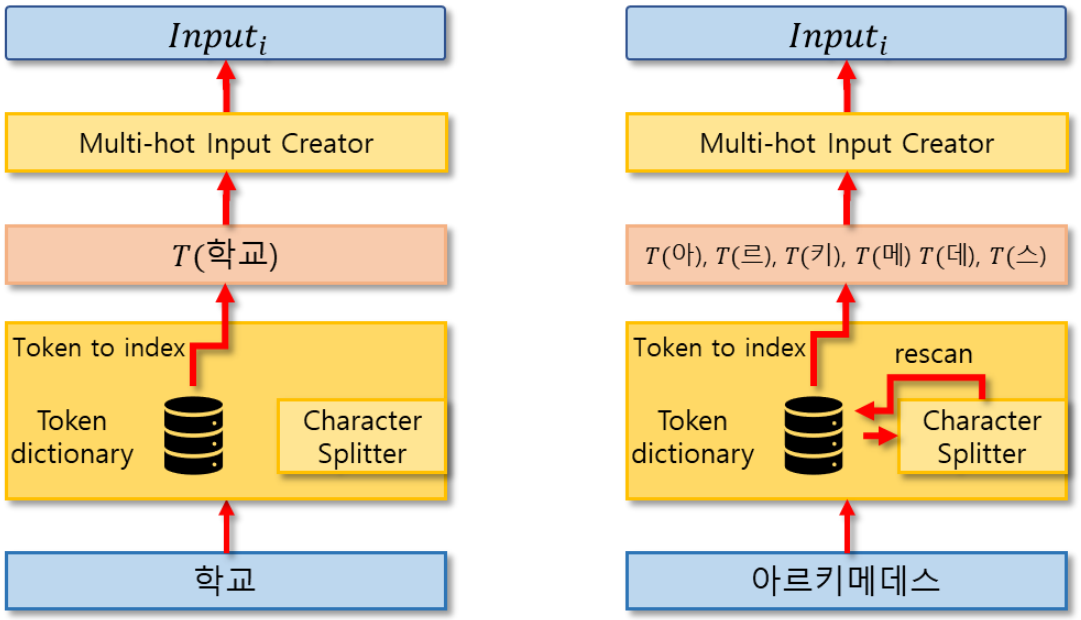


[그림 21] 형태소 단위의 유지가 언어 모델에 가져오는 영향 비교

(예시: 오늘/MAG 해외/NNG 관현악단/NNG+이/JKS 내한하__01/VV+였/EP+다/EF+./SF)

[그림 21]는 기존의 토큰 방식과 형태소 단위를 유지한 경우 트랜스포머 인코더에 적용되는 현상의 차이를 보여주는 그림이다. 기존의 언어 모델의 경우 '내한하/VV'라는 형태소가 '내'와 '한하/VV'로 분리된다. 분리로 인해 주변 문맥의 정보가 '내'와 '한하/VV' 각각에 전달되는 형태로 표현되며, 각각의 토큰이 값을 가진다. 반면, 형태소 단위를 유지한 경우 '내한하/VV'가 하나의 토큰으로 사용되며, 주변 문맥의 정보가 고스란히 해당 형태소에 담기며, 최종적으로 '내한하/VV'에 대한 출력 결과를 생성할 수 있다.

본 논문에서 사용한 다중-핫 표현 방식은 구축된 토큰 사전에 등록 여부에 따라 표현 방식이 다르다. 형태소가 토큰 사전에 존재하는 경우 기존의 원-핫 표현을 사용하지만, 존재하지 않은 경우 형태소를 음절 단위로 분리하여, 음절 토큰들의 집합으로 표현했다.

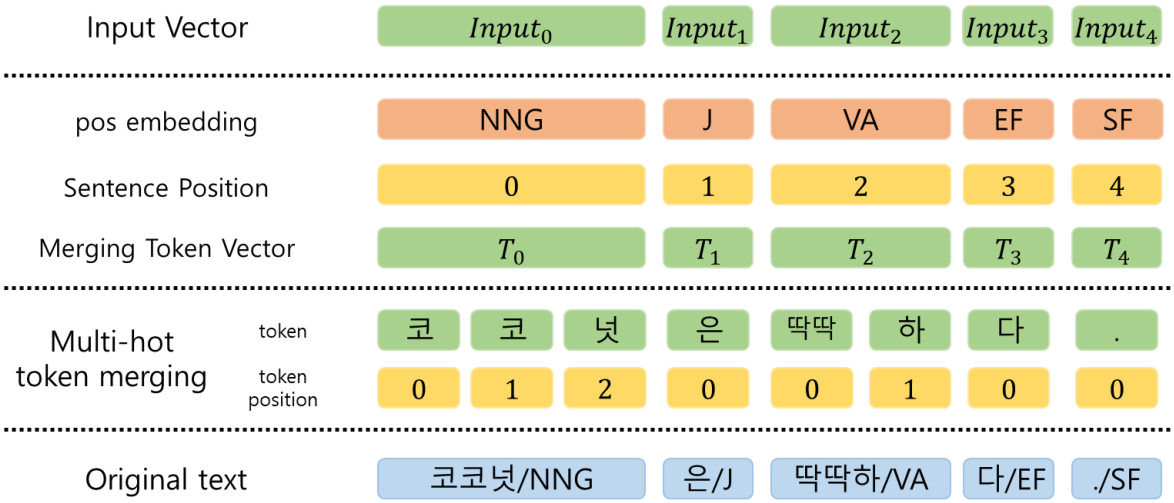


토큰 사전에 존재한 경우　　토큰 사전에 존재하지 않는 경우

[그림 22] 토큰 사전의 존재 유무에 따른 입력 토큰의 생성 방식의 차이

[그림 22]는 다중-핫 표현 방식이 형태소 단위를 유지하는 방법을 보여준다. 형태소가 토큰 사전에 존재 유무에 따라 토큰 표현 방식이 다르다. [그림 22]에서 '학교'라는 단어는 토큰 사전에 등재되어 있어 '학교'라는 토큰으로 반환된다. 여기서 $T(학교)$는 '학교'라는 토큰의 인덱스 번호이며, 입력 생성기에 의해 언어 모델의 입력 값으로 표현된다. 반면, 토큰 사전에 등재되지 않은 '아르키메데스'는 음절 단위로 분리하여 6개의 음절 단위의 토큰 인덱스 집합으로 표현된다. 이 과정에서 해당 음절이 존재하지 않은 경우 미등록 토큰(unknown 토큰)으로 처리된다. 단어를 표현한 각각의 음절에 대한 인덱스 번호들을 다중-핫 입력 생성기에 입력하여 '아르키메데스'라는 단어의 벡터로 표현했다.

형태소의 토큰 사전 등재 여부에 의해 변환된 토큰들은 다중-핫 입력 생성기에 의해 하나의 벡터로 표현된다. 다중-핫 입력 생성기는 각 토큰의 임베딩 결과와 토큰의 위치 정보를 통해 하나의 벡터로 먼저 표현되며, 최종적으로 품사 정보와 문장에서의 위치 정보를 융합하여 언어 모델의 입력 값으로 표현된다.



[그림 23] 형태소 단위 유지를 위한 다중-핫 입력 생성기 도식화

[그림 23]은 다중-핫 입력 생성기의 구조를 보여준다. 입력된 형태소가 실제 언어 모델의 입력에 사용하기 위해서는 2단계를 거쳐 입력으로 변환된다. 1단계에서는 형태소를 변환한 토큰들을 하나의 벡터로 표현한다. 변환된 토큰들의 임베딩 벡터와 현재 토큰이 형태소 안에서의 위치 정보를 조합했다. 형태소 내의 토큰의 위치 정보는 해당 형태소가 특정 음절이 중복된 경우를 분별하기 위해 사용했다. 예를 들어 '코코넛'이라는 단어가 음절로 분리되어 합한 토큰을 생성한 경우 단순히 '코'에 해당하는 임베딩 값이 2배로 적용된 모습이 되며, '넛코코'나 '코넛코'라는 단어가 동일한 벡터로 표현되기 때문에 구분을 위해 형태소 안에서의 각 토큰의 위치 정보를 사용했다.

$$T_k = \sum_{i=0}^{j} t_i \, tp_i \qquad \text{[수식 3]}$$

[수식 3]은 1단계에서 각 형태소가 단일 토큰으로 구성되거나, 토큰들의 집합으로 표현된 경우에 대해 해당 형태소를 벡터로 표현하는 수식이다. [수식 3]에서 t_i는 해당 토큰의 임베딩 벡터 값이며, tp_i는 해당 토큰의 형태소 안에서의 위치를 임베딩한 정보이다. 이렇게 두 정보를 곱하여 생성된 벡터들을 전부 더하여 형태소에 대한 벡터로 표현했다.

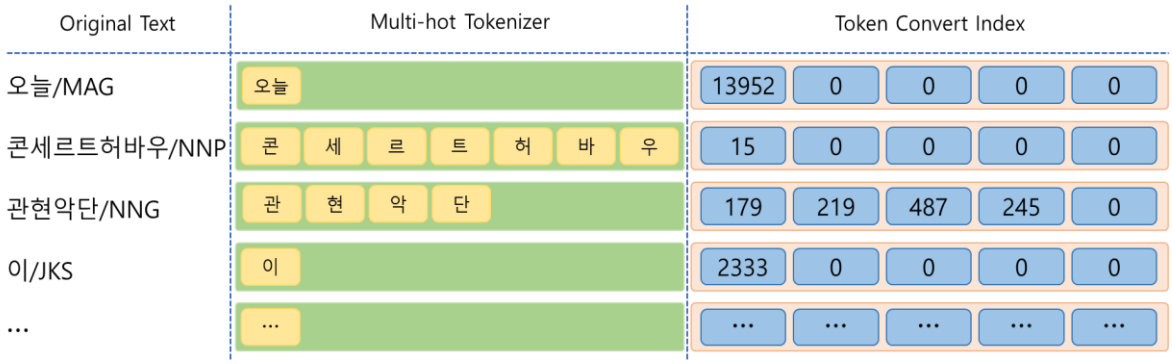
다중-핫 입력 생성기의 2단계에서는 1단계에서 형태소를 통해 생성한 벡터 값에 추가 정보를 더하여 최종적인 언어 모델의 입력 벡터로 생성했다. 현재 문장에서 해당 형태소의 위치 정보 임베딩 값과 각 형태소가 가진 품사를 임베딩 값을 더해 생성했다.

$$Input_k = T_k + P_k + pos_k \quad \text{[수식 4]}$$

[수식 4]는 언어 모델에 입력으로 사용될 $Input_k$를 생성하는 수식이다. [수식 3]에서 생성된 형태소의 벡터 T_k와 문장에서의 형태소의 절대 위치 정보 임베딩 결과인 P_k와 해당 형태소가 가지는 품사의 임베딩 정보인 pos_k를 더했다. 이렇게 생성된 입력 벡터는 언어 모델의 트랜스포머 인코더에 사용된다.

3.2 다중-핫 표현 방식을 위한 입력 생성기의 최적화

다중-핫 표현 방식은 원-핫 방식보다 더 많은 양의 데이터를 필요로 한다. 기존의 원-핫 표현 방식의 경우 토큰 단위가 모델에서 입력으로 사용되며, 임베딩의 결과와 추가 정보의 결합만으로 쉽게 사용이 가능했다. 하지만, 다중-핫 표현 방식은 사용된 형태소에 따라 토큰으로 표현되는 개수가 다르다. 예를 들어 토큰 목록에 등재된 '수학'이라는 단어는 하나의 토큰으로 표현되지만, 등재되지 않은 '코코넛'이라는 단어는 3개의 토큰으로 표현된다. 만약 해당 형태소를 음절 단위 토큰의 집합으로 표현하기 위해 최대 크기로 고정된 토큰 집합을 강제하게 되면 원시 문장을 토큰으로 변환했을 때 형태소 수와 최대 토큰 집합 크기의 곱으로 표현된다. 모든 형태소가 최대 토큰 길이를 가지지 않기 때문에 빈 공간이 발생하게 되며, 토큰으로 표현된 입력에 대해 빈 공간이라는 정보를 전달하기 위한 추가적인 자원이 필요하다.

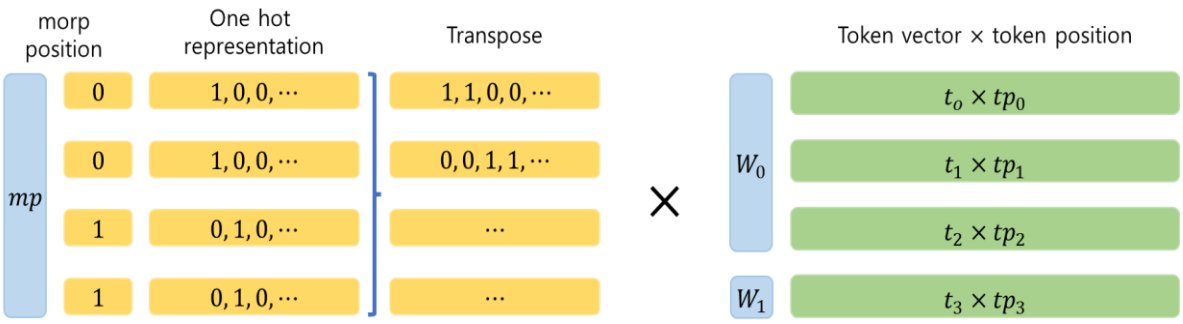


[그림 24] 다중-핫 표현 방식에서 고정된 길이의 인덱스 표현의 문제점

[그림 24]는 다중-핫 표현을 통해 생성된 형태소를 구성하는 토큰들에 대해 고정된 길이의 인덱스로 변환하는 경우 발생할 수 있는 문제점들을 보여준다. [그림 24]에서는 형

태소 하나를 표현하기 위해 최대 5개의 크기로 고정된 길이를 사용했다. '오늘/MAG'의 경우 토큰 사전에 등재되어 있는 형태소이기 때문에 한 개의 토큰으로 표현되지만, 하나의 형태소가 고정된 길이의 표현 형태를 가지기 때문에 '오늘'의 인덱스 번호와 4개의 0번 인덱스로 표현된다. 이 경우 4개의 공간이 낭비되는 현상이 발생한다. '콘세르트허바우/NNP'의 경우 7음절의 형태소로 다중-핫 표현에 의해 7개의 음절 토큰의 집합으로 표현되었다. 하지만, 입력을 위한 변환에서 형태소의 최대 표현 길이를 5로 설정한 경우 초과하기 때문에 해당 형태소를 모델에 표현할 수 없기 때문에 미등록 토큰으로 변환되는 문제가 발생하게 된다.

본 연구에서는 다중-핫 표현 방식을 위한 입력 토큰의 표현 방식 최적화를 위해 입력 문장을 변환한 토큰들과 해당 토큰이 속한 형태소의 위치 정보를 활용했다. 예를 들어 "학교", "에"라는 두 형태소가 '학', '교', '에'로 분리되어 3가지 토큰을 가지는 경우 각 토큰들이 위치한 형태소 번호는 [0, 0, 1]로 표현된다. '학', '교'는 첫 번째 형태소인 '학교'에서 분리된 토큰이기 때문에 위치 번호가 0으로 표기되며, '에'의 경우 문장에서 두 번째 형태소여서 위치 번호가 1로 표기됐다. 이렇게 구축된 위치 정보를 사용해 다양한 길이의 형태소를 표현할 수 있었다.



[그림 25] 다중-핫 표현 변환 방식의 최적화를 위한 연산 방법 도식화

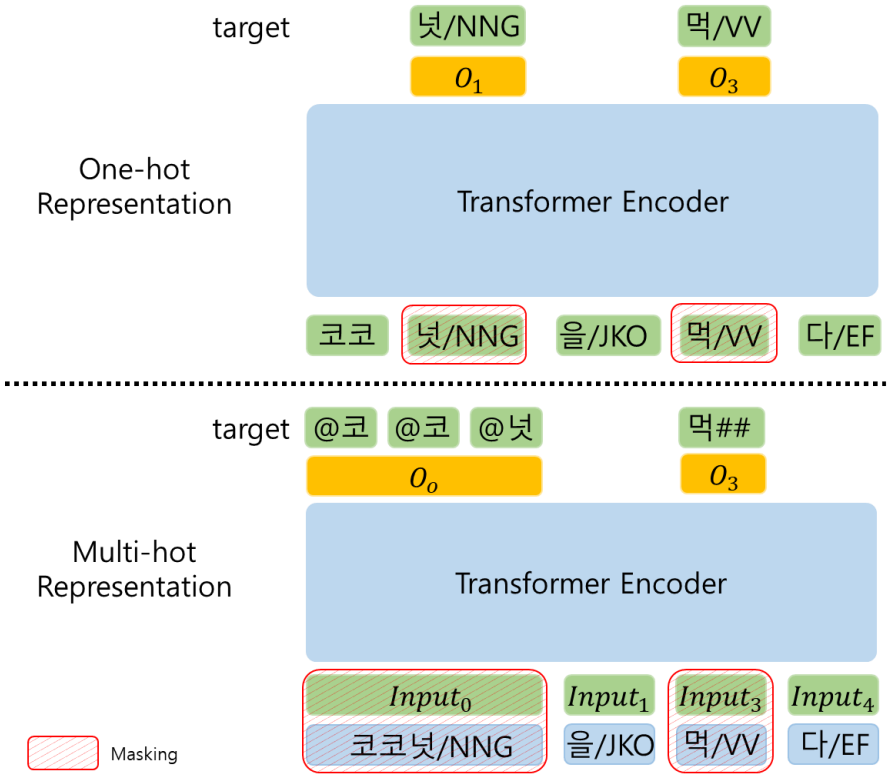
[그림 25]은 각 토큰이 속한 형태소 번호와 [수식 3]에서 1단계에 해당하는 토큰 집합을 하나의 벡터로 표현하는 방법을 보여준다. 문장에서의 형태소 위치 정보를 원-핫 방식으로 표현한 2차원 행렬에 대해 전치행렬(transpose matrix)을 실시했다. 원-핫으로 표현된 행렬을 전치(transpose)하면, 동일한 입력 값을 가지는 위치들이 모여서 표현되게 된다. 형태소 위치 번호의 전치행렬과 각 토큰의 벡터와 형태소 안의 위치를 곱한 정보에 대해 행렬 곱을 시행하여 최종적으로 하나의 형태소에 대한 [수식 3]에서의 T_k를 생

성했다. 이러한 방식을 통해 글자 수가 다양한 형태소들로 구성된 입력에 대해 다중-핫 표현 방식으로 인해 발생할 수 있는 공간의 낭비를 줄일 수 있다.

3.3 다중-핫 표현 방식을 학습하기 위한 손실 함수

3.3.1 다중-핫 표현 방식과 기존의 손실 함수

다중-핫 표현 방식은 형태소 단위의 유지와 언어 지식을 쉽게 사용하기 위한 바탕이 된다. 다중-핫 표현을 통해 언어 모델의 출력이 형태소 단위를 유지하게 되며, 주변의 문맥을 온전히 형태소에 담을 수 있기 때문에 문맥의 이해와 해석에서도 장점을 가질 수 있다. 본 논문에서는 다중-핫 표현 방식을 통해 형태소를 표현하기 때문에 마스킹 학습도 적용된다. 하지만, 다중-핫 표현을 사용한 경우 마스킹 처리된 위치가 여러 토큰으로 구성될 수 있다. 이러한 경우 하나의 문제에 대해 여러 정답을 가지는 문제(multi-class classification)가 되며, 이에 맞춘 학습 방식이 필요하다.



[그림 26] 원-핫 표현과 다중-핫 표현의 마스킹 학습에서의 차이점

[그림 26]은 원-핫 표현과 다중-핫 표현을 사용한 언어 모델에 대해 마스킹 학습에서 정답의 차이점을 보여준다. 원-핫 표현 방식의 경우 마스킹 대상이 된 '넛/NNG'라는 토큰에 대해 모델이 유추하면 된다. 하지만, 다중-핫 표현 방식의 경우 마스킹 된 위치의 형태소가 하나 이상의 토큰으로 구성될 수 있다. 예를 들어 [그림 26]에서 '코코넛/NNG'라는 형태소에 대해 출력 값이 3개의 정답을 유추해야 한다. 다중-핫 표현 방식의 경우 정답이 여러 개로 구성 가능하기 때문에 이에 적합한 손실 함수의 선택이 중요하다. 적합한 손실 함수의 선택을 위해 기존의 대표적인 손실 함수들을 분석했다.

$$\ell_i = y_i\big(log\big(p(y_i)\big) + (1-y_i)\,log\big(1-p(y_i)\big)\big)$$
$$BCE = -\frac{1}{N}\sum_i \ell_i \qquad \text{[수식 5]}$$

[수식 5]은 바이너리 크로스 엔트로피의 수식이다. 바이너리 크로스 엔트로피는 정답이 1개 이상인 문제(multi label classification)에서 대표적으로 사용되는 손실 함수이다. 바이너리 크로스 엔트로피는 각각의 후보에 대한 손실 값인 ℓ_i를 구하여, 평균 손실을 사용하는 방식이다. 복수의 정답을 가지는 다중-핫 표현 방식의 언어 모델에 적합해 보이지만, 후보가 많은 경우에 적합하지 않다. 바이너리 크로스 엔트로피는 손실 값인 ℓ_i을 모든 후보에 대해 계산하는 손실 함수로 후보가 많아 질수록 평균 손실 값이 낮아지며, 모든 후보의 정답의 확률이 0으로 수렴하게 되는 문제점을 가진다. 언어 모델의 경우 마스킹 학습으로 사용하는 후보의 가짓수는 구축된 토큰 사전의 양과 같기 때문에 후보의 수가 일반적인 분류 문제보다 많다.

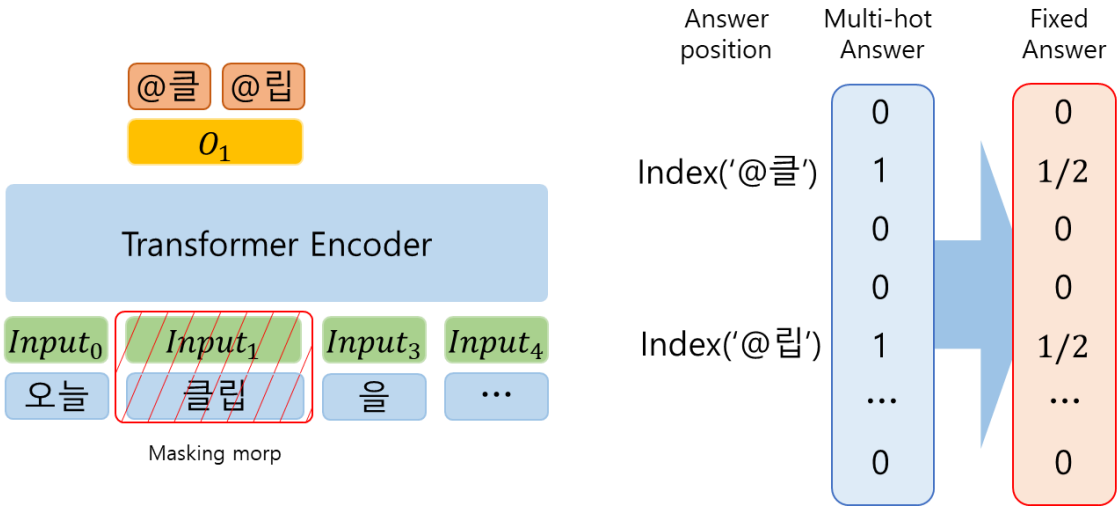
$$SCE = -\sum_i log\ (S(O_i))y_i \qquad \text{[수식 6]}$$

[수식 6]는 소프트맥스 크로스 엔트로피(Softmax Cross Entropy)의 수식이다. 소프트맥스 크로스 엔트로피는 바이너리 크로스 엔트로피와 달리 정답이 하나인 분류 문제에서 사용한다. 손실 값을 구하기 위해 로그-소프트맥스 함수를 이용한다. 하지만 소프트맥스 크로스 엔트로피 함수도 다중-핫 표현에는 적합하다고 볼 수 없다. 다중-핫 표현 방식의 경우 정답이 되는 후보가 하나 이상이 가능하다. 현재 문제에 대해 정답의 수가 많아질수록 손실 값이 증가하는 문제점이 발생한다. [수식 6]을 통해 계산된 각 후보의 손실 값을 다 더하기 때문에 정답이 여러 개인 경우 커진다. 예를 들어 마스킹으로 선정된 형태

소가 5개의 음절 토큰으로 구성된 경우 $\sum_i y_i$의 값이 5가 된다. 다중-핫 표현으로 인해 $\sum_i y_i$의 값이 5인 경우 정답 토큰의 수가 1개인 경우와 출력되는 손실 값이 최대 5배 큰 값으로 계산될 수 있다. 학습의 정답이 되는 토큰들의 수에 따라 각 손실 값 간의 편차가 커지게 되는 문제가 발생한다.

3.3.2 정답의 개수에 의한 목표 확률을 조절하는 소프트맥스 크로스 엔트로피 손실 함수

다중-핫 표현 방식의 학습을 위해 목표 확률을 조절하는 소프트맥스 크로스 엔트로피 손실 함수를 사용했다. 복수의 정답을 가지는 학습에 대해 정답 위치에 있는 후보의 목표 확률을 1이 아닌 y_i/n로 유도했다. 여기서 n은 정답의 수이며 y_i는 해당 후보의 정답 여부이다. 정답의 확률 조정 방식은 레이블 스무딩(label smoothing)[47]과 유사하다. 레이블 스무딩은 원-핫 형태의 정답에 대해 나머지 후보들도 낮은 확률을 가지도록 정답에 대한 정규화를 실시하는 방법이다. 하지만, 레이블 스무딩의 경우 정답이 아닌 후보가 계산의 영향을 주기 때문에 전체 후보의 가짓수가 많고 정답의 개수가 작기 때문에 모델이 정답을 위해 목표하는 확률 값이 너무 작아지는 문제를 가진다. 본 논문에서 사용한 목표 확률 조절의 예시로는 마스킹된 형태소가 2개의 토큰으로 구성된 경우 정답의 개수는 2개이며, 각각의 정답 위치 후보의 목표 확률을 1이 아닌 $y_i/2$이 되도록 유도했다.

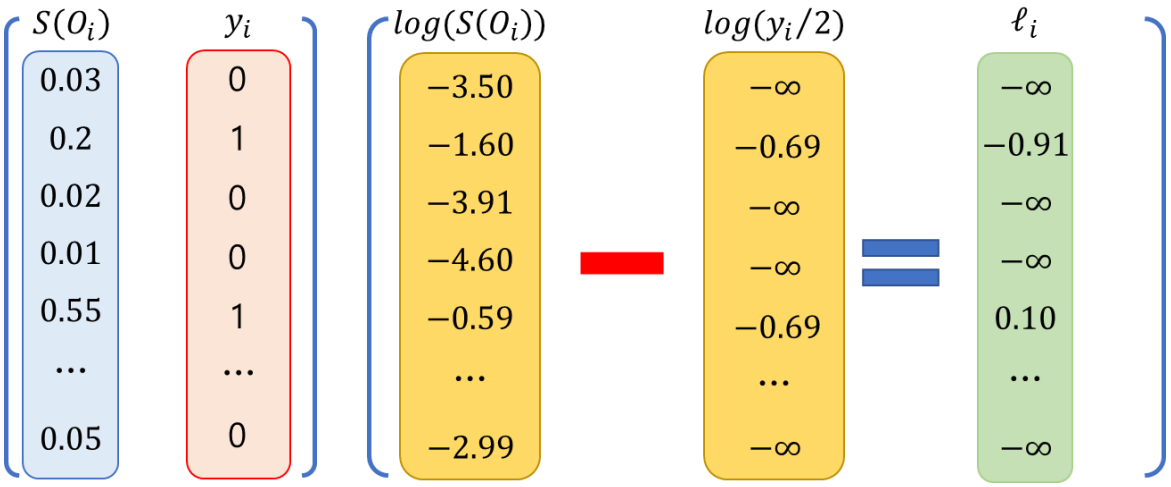


[그림 27] 정답의 수에 따른 학습 목표 확률의 변경

[그림 27]은 정답 수에 따른 목표 확률의 변화의 예시이다. 마스킹 처리된 '클립'이라는 형태소는 2개의 토큰으로 표현되며 정답이 된다. 정답을 원-핫으로 표현한 값을 사용하지 않고 현재 마스킹 처리된 형태소의 토큰 수인 2로 나눈 $y_i/2$이 되도록 모델을 학습한다.

$$\ell_i = (log(S(O_i)) - log\ (y_i/n)) \qquad \text{[수식 7]}$$

[수식 7]은 특정 후보에 대한 손실 값을 계산하는 수식이다. 후보가 목표로 하는 확률을 y_i/n로 유도하도록 수식을 구성했다. [수식 7]은 $\log(S(O_i))$를 통해 계산된 현재 후보의 손실 값이 목표 확률에 도달하도록 설계했으며, 목표 확률에 도달한 경우 후보의 손실 값이 0으로 표현하기 위해 $\log\,(y_i/n)$를 뺀 값을 사용했다. 여기서 n은 정답의 수이며 y_i는 현재 후보의 정답 여부이다. 수식의 계산 과정에서 y_i/n가 0이 되는 경우 로그로 인해 $-\infty$로 값이 발산하는 문제를 가지지만, 이후 계산 과정을 통해 학습에서 발산하는 문제를 제거했다.



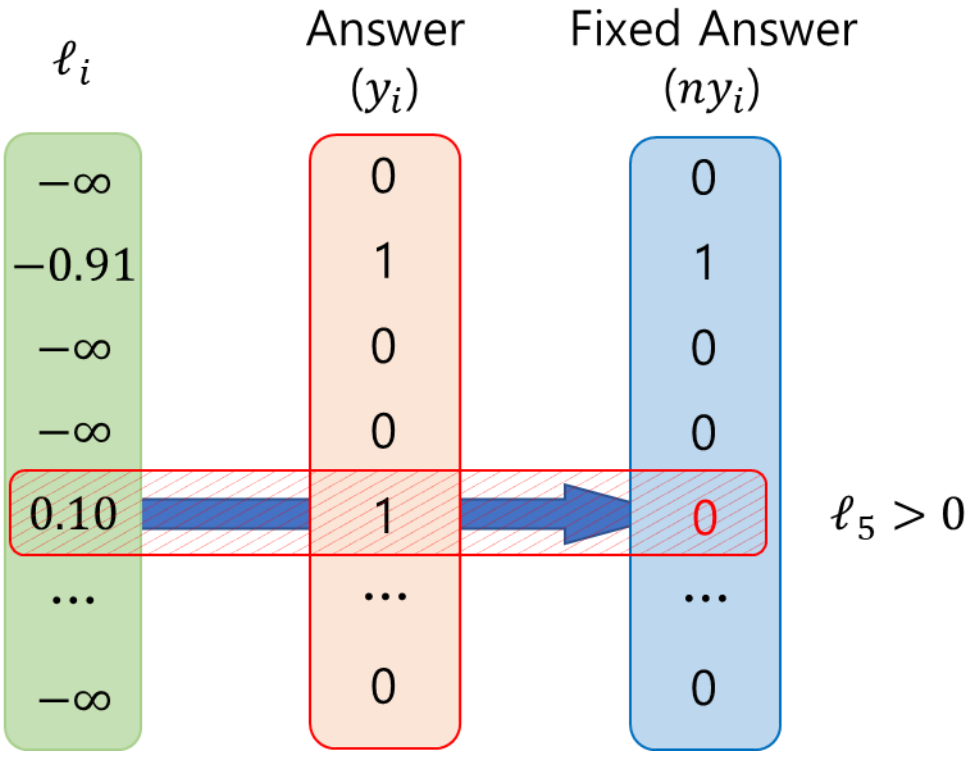
[그림 28] 각 후보의 손실 값 계산 예제

[그림 28]는 [그림 27]에서 사용한 예시에 대해 각각 후보들에 대한 손실 값을 계산 결과이다. 언어 모델이 출력한 결과의 각 후보들의 소프트맥스 값과 목표 확률을 변형한 정답에 대해 [수식 7]을 사용해 각각의 후보의 손실 값을 계산하게 된다. 이 과정에서 5번째 후보의 손실 값인 ℓ_5가 0.1로 양수가 되며 이는 해당 후보가 이미 목표 확률을 달성한 것을 의미한다. 각각의 후보가 목표 확률을 달성한 경우를 학습에서 제외하기 위해 정답을 재정의 하는 과정을 진행한다.

각 후보에 대한 정답을 재정의 하는 과정은 정답의 위치에 있는 후보가 이미 목표하는 확률인 y_i/n를 달성한 경우를 학습에서 제외하기 위해 사용했다. 목표 확률을 넘어서는 경우 [수식 7]을 통해 계산된 각 후보의 손실 값인 ℓ_i가 양수가 된다. 최종적인 손실 값을 구하기 위해 모든 후보의 손실 값을 더하기 때문에 목표 확률을 초과한 경우 최종 손실 값이 양수로 표현될 수 있으며, 오차를 가지는 후보와 목표 확률을 초과한 후보들로 인해 손실 값이 확률에 비해 너무 작아지는 문제를 가져오게 된다.

$$ny_i = if\ \ell_i < 0 : y_i\ else : 0 \qquad \text{[수식 8]}$$

[수식 8]은 다중-핫 표현 모델에서 정답의 재설정하는 수식이다. [수식 8]의 ny_i는 정답을 재정의 한 결과이다. 수식에서 ℓ_i의 값이 음수로 표현된 경우 기존의 정답인 y_i을 그대로 사용했다.



[그림 29] 정답의 재설정 과정 예시

[그림 29]에서 두 번째 후보의 ℓ_1는 음수이며, 실제 정답은 1이다. 이 경우 아직 학습이 필요한 부분이기 때문에 [수식 8]에 의해 ny_1가 1로 유지된다. 반면 5번째 후보의 경우 계산된 손실 값 ℓ_5가 0.1인 양수이며 이는 목표 확률을 초과했다고 볼 수 있다. 이 경우 정답이 1에서 ny_5가 0으로 바뀌게 된다.

최종적으로 하나의 토큰 집합에 대한 손실 함수는 [수식 8]에서 재설정한 정답과 [수식 7]에서 구한 손실 값을 곱한 값의 평균을 사용한다. 다중-핫 표현의 경우 정답의 수가 1개 이상이기 때문에 전체 후보가 가진 손실 값의 평균을 사용했다.

$$\Delta Loss = -\sum_i(\ell_i \cdot ny_i)/\sum_i(ny_i) \quad \text{[수식 9]}$$

[수식 7]에서 구한 ℓ_i의 평균을 해당 문제에 대한 손실 값으로 사용했다. [수식 8]에서 정답을 재설정한 ny_i을 합산한 값을 분모로 사용하며, ℓ_i와의 곱으로 [수식 9]을 구성했다. [수식 7]에서 문제가 될 수 있는 ℓ_i의 결과가 $-\infty$인 경우에 대해 [수식 9]의 연산 과정에서 ny_i가 0의 값을 가져 두 값을 곱하게 되면 0이 되므로 최종적으로 손실 값에는 영향을 주지 않는다.

4 언어 모델을 위한 다양한 언어 지식의 활용

본 논문에서는 인간이 구축한 다양한 자연어에 대해 구축한 지식 정보와 기존의 문맥 정보를 함께 언어 모델에 사전 학습(pre-training)했다. 기존의 언어 모델들은 말뭉치에서 얻을 수 있는 문맥 정보만을 가지고도 자연어에 대한 높은 이해력을 보여주었다. 하지만, 말뭉치에서만 습득한 정보의 경우 단편적이며, 편향적일 수도 있다. 그리고 단순히 대량의 말뭉치를 확보하기 어려운 문제도 존재한다. 언어 모델과 달리 인간은 자연어에 대해 구축한 다양한 언어 지식을 학습하고 활용하여 자연어에 대해 다양한 정보를 생성하고 이해할 수 있다. 반면, 인간은 습득할 수 있는 지식의 양이나, 정보의 다양성에는 한계를 가지고 있으며, 언어 모델처럼 거대한 양의 정보를 학습하지는 못한다.

본 논문에서는 여러 언어 지식 정보와 기존의 문맥 정보를 융합한 언어 모델인 UKnowBERT(Ulsan Knowledge ensemble BERT)을 개발했다. 기존의 BERT에 추가적인 구조를 사용하지 않고, 구축한 다양한 언어 지식 정보를 학습에만 사용하여, 언어 모델이 문맥적, 의미적 이해를 함께 함과 동시에 모델의 크기를 유지하도록 설계했다. 울산대학교에서 구축한 한국어 어휘 의미망 UWordMap[48]을 사용하여 학습에 사용할 언어 지식을 생성했다. UWordMap은 표준국어대사전에 등재된 표제어를 다의어 수준으로 어휘 간의 관계성을 정의한 어휘망이다. 울산대학교에서 구축한 UWordMap은 전체 데이터가 공개되어 있다.[13] UWordMap에는 명사들의 상하 관계 정보, 용언의 의미제약 정보 등 다양한 자연어 지식 정보가 구축되어 있다. 본 논문에서는 명사의 상위어 정보, 어휘의 뜻풀이, 용언의 의미 제약 정보를 활용하여 언어 지식 정보를 구축했으며, 각각의 데이터의 성격에 맞는 학습 방식을 사용했다.

4.1 명사의 상위어 추론 학습을 통한 어휘의 군집화

명사는 특정한 대상의 명칭이나, 추상적인 개념을 표현한 단어로 문장에서 실질적인 요소로 많이 사용된다. 명사는 문장에서 행동의 주체나, 대상이 되는 단어의 기초적인 성

[13] http://nlplab.ulsan.ac.kr/

질을 파악할 수 있으며, 해당 명사가 가진 특성을 통해 문장의 분석에 도움을 줄 수 있다. 이러한 명사들은 의미에 따라 다른 명사들을 포괄하는 추상적인 개념이 될 수 있으며, 이를 통해 명사 간의 상하 관계를 정립할 수도 있다. 언어 지식 정보를 추출하기 위해 사용한 UWordMap에는 명사의 뜻풀이를 이용하여 명사들 간의 상하 관계성이 구축되어 있다.

[그림 30] UWordMap에서 명사 '칫솔'의 상하 관계 정보

[그림 30]는 UWordMap에 구축된 '칫솔'(이를 닦는데 쓰는 솔)의 상하 관계 정보를 보여주는 그림이다. '칫솔'(이를 닦는 데 쓰는 솔)은 '솔'이라는 상위 개념과 연결이 되어 있으며, '칫솔'의 하위어에는 '칫솔'의 의미를 포함하고 있는 '이솔', '잇솔', '음파전동칫솔', '음파칫솔'과 연결되어 있다. 단어가 가진 의미를 사용해 UWordMap에는 다의어 수준으로 총 439,248개의 단어에 대해 상하 관계를 구축했다.

명사들이 가지는 상하 관계를 통해 개념적으로 유사한 단어들에 대해 군집화를 이룰 수 있다. 기존의 문맥 정보만 이용하여 학습한 경우 동일한 개념에 속한 단어라고 해도 비슷하게 사용된 문장을 학습하지 않으면, 언어 모델이 동일한 개념이라고 판단하지 않을 수 있다. 반면, 상위어를 학습에 사용하면 해당 단어가 사용된 문맥이 적어도 단어들이 동일한 상위어를 학습하여 비슷한 의미를 담을 수 있다. 예를 들어 '중학교', '고등학교'는 '학교'라는 상위 개념에 속한 경우 '고등학교'만 사용된 문장에 대해 '중학교'로 치환되어도 비슷한 문장 해석이 가능하게 된다.

명사의 상하 관계 정보를 학습하기 위해 UWordMap에 구축된 다의어 기반의 상하 관계 정보를 동형이의어 수준으로 변환하여 사용했다. 단어가 여러 다의어를 가진 경우 각

각의 다의어가 가지고 있는 모든 상위어를 묶어서 사용했다. 상위어로 등록된 단어는 품사와 의미 번호를 제거한 표제어만 사용했으며, 해당 단어가 하나의 토큰으로 변환되는 경우만 사용했다. 다음의 표는 본 논문에서 사용한 상위어 관계 정보를 동형이의어 수준으로 수집하여 구축한 예시를 보여준다.

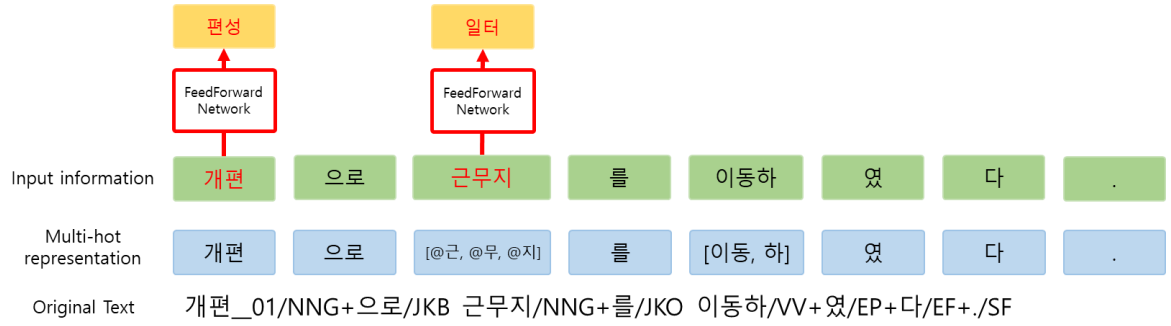
[표 4] 표제어 '일가'에 대해 동형이의어 별 상위어 관계 정보 수집 예시

사전 등재 어휘	상위어	학습용 대상 어휘	상위어
일가__010000/NNG	시렁	일가__01/NNG	시렁
일가__020001/NNG	가족	일가__02/NNG	가족, 경지, 현상
일가__020002/NNG	겨레붙이		
일가__020003/NNG	경지		
일가__020004/NNG	현상		
일가__030001/NNG	값	일가__02/NNG	값
일가__030002/NNG	원자가		
일가__030003/NNG	히드록시기		
일가__040000/NNG	여유	일가__02/NNG	여유

[표 4]은 '일가'라는 표제어의 다양한 다의어에 대해 상위어 데이터를 구축하는 방식을 보여준다. 먼저 '일가'가 가진 다의어 중에 상위어를 가진 대상들을 선정한다. '일가__010000/NNG'(시렁 하나.)처럼 다의어가 없는 경우 '시렁'이 상위어 학습의 대상으로 구축된다. 반면 '일가__02/NNG'는 4개의 다의어가 가지며, 각각의 다의어가 가진 상위어 중에서 하나의 토큰으로 표현 가능한 '가족', '경지', '현상'을 대상으로 선정했다. '일가__020002/NNG'의 상위어인 '겨레붙이'는 토큰 목록에 존재하지 않기 때문에 제외했다. 어휘망에 구축된 상위어를 가지는 다의어들에 대해 [표 4]과 같이 구축 규칙을 적용하여, 동형이의어 수준으로 총 194,513개의 상위어 관계 정보 자료를 구축했다.

어휘망을 이용해 구축한 명사의 상위어 데이터는 언어 모델의 학습에 사용한 문장에서

등장한 명사들 중에 상위어를 가지는 대상을 선정하여 상위어를 추론하도록 학습을 구성했다. 다중-핫 표현 입력 생성기를 통해 생성된 각 형태소에 대한 입력 벡터를 상위어 추론 학습에 사용했다. 각각의 형태소가 가진 상위어의 개념을 해당 형태소를 표현한 벡터가 이해하기 위함이다. 동일한 상위어를 가진 단어들이 비슷한 입력 벡터를 가지게 유도함으로 인해 해당 단어가 학습되지 않은 문장에서도 동일한 상위어를 가진 다른 단어들이 학습한 정보로 인해 분석할 수 있다.



[그림 31] 명사의 상위어 추론 학습

[그림 31]은 입력 문장으로 "개편__01/NNG+으로/JKB 근무지/NNG+를/JKO 이동하/VV+였/EP+다/EF+./SF"가 사용된 경우 상위어 추론 학습이 진행되는 예시이다. 입력 문장에서 사용된 '개편__01/NNG'과 '근무지/NNG'의 경우 상위어 학습 데이터를 가지고 있으므로 학습의 대상으로 선정된다. 상위어 학습을 위해 '개편__01/NNG'에 대해 다중-핫 표현 입력 생성기로 생성된 입력 계층의 값을 사용했다. 다중-핫 입력 생성기에 의해 생성된 '개편__01/NNG'의 벡터 값을 순방향 네트워크(FeedForward Network)를 통해 토큰 목록의 수로 벡터의 크기를 변환했다. 변환된 벡터의 확률 값이 '개편__01'의 상위어인 '편성'을 추론하도록 설계했다.

$$x_{hyper} \approx Softmax(FNN(input(x_i))) \qquad \text{[수식 10]}$$

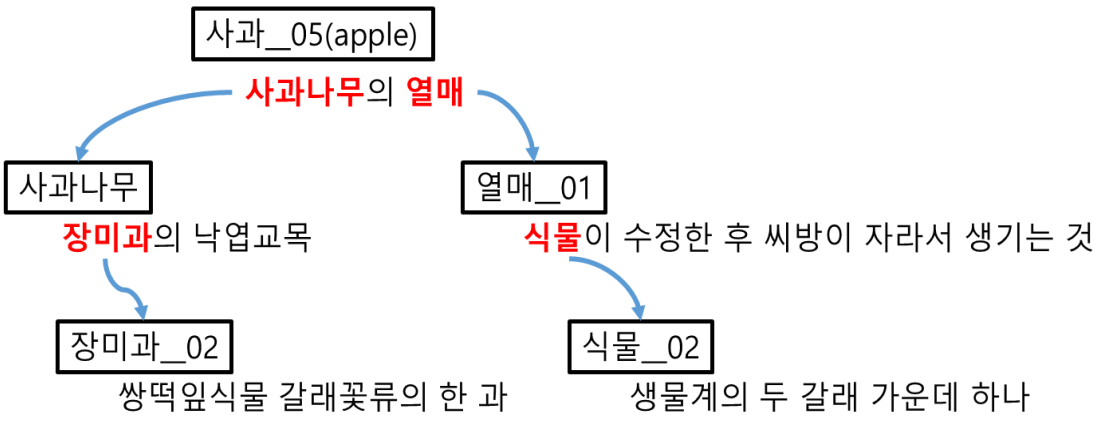
[수식 10]은 [그림 31]에서 명사의 상위어를 추론하는 학습을 표현한 수식이다. 수식에서 사용된 x_{hyper}는 명사 x_i의 상위어이다. 명사의 상위어 추론은 [수식 4]를 통해 각 형태소의 모델 입력 값인 $input(x_i)$을 순방향 신경망 통과시켜 상위어인 x_{hyper}를 추론할 수 있도록 설계했다. 실제 학습 과정에서는 입력 문장에서 상위어를 가지는 최대 20개의 단어를 임의로 추출하여 학습의 대상으로 선정했다. 그리고 하나의 단어가 여러 상위어

를 가지는 경우를 대비해 다중-핫 표현 방식을 학습하기 위해 사용한 목표 확률을 조정하는 소프트맥스 크로스 엔트로피 손실 함수를 이용했다.

4.2 어휘 뜻풀이 기반 벡터를 사용한 어휘의 의미 학습

단어에 대한 정의는 인간이 오랜 시간 걸쳐 정립하고 구축한 정보로 해당 단어를 이해하기 위한 가장 기초적인 정보로 볼 수 있다. 인간은 단어를 이해하는 하나의 방법으로 사전을 통해 뜻풀이를 보고 이해할 수 있으며, 뜻풀이에 표현된 다양한 단어들에 대해서도 다시 사전적 정의를 통해 이해할 수 있다. 사전에 정의된 뜻풀이들의 연쇄성을 통해 인간은 다양한 자연어의 의미를 이해함과 동시에 단어 간의 관계성을 파악하는 것이 가능하다. 단어를 정의한 뜻풀이는 유사한 개념을 가진 단어끼리 비슷한 의미의 뜻풀이로 정의될 수 있다. 예를 들어 '공감하다'(남의 감정, 의견, 주장 따위에 대하여 자기도 그렇다고 느끼다.)와 '동감하다'(어떤 견해나 의견에 같은 생각을 가지다.)는 유사한 뜻풀이를 가지고 있다. 그렇기 때문에 단어가 가진 뜻풀이만으로도 단어들 간의 유의, 반의 관계를 판단할 수 있는 근거가 된다.

본 논문에서는 단어가 가진 사전적 뜻풀이를 사용해 구축한 어휘 의미 벡터를 언어 모델에 학습했다. 단어의 뜻풀이를 벡터로 표현하기 위해 USenseVector[49]를 사용했다. USenseVector는 단어가 가진 뜻풀이를 사용하여 단어를 벡터로 표현한 모델이며, 뜻풀이에 등장한 다른 단어가 가진 개념이 해당 단어에 영향을 주도록 설계한 모델이다.

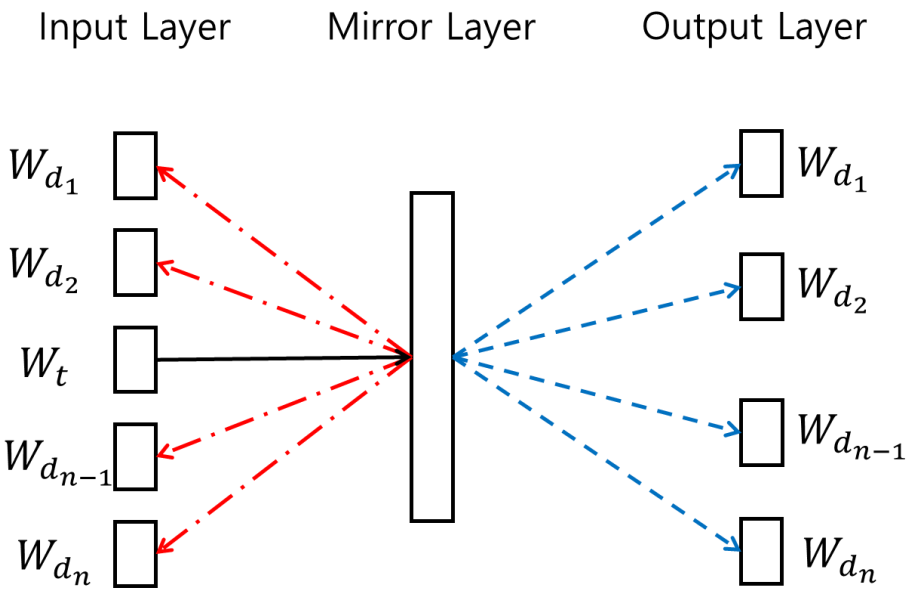


[그림 32] 각 자연어의 뜻풀이의 연관성

(출처: 단어 의미와 자질 거울 모델을 이용한 단어 임베딩[49])

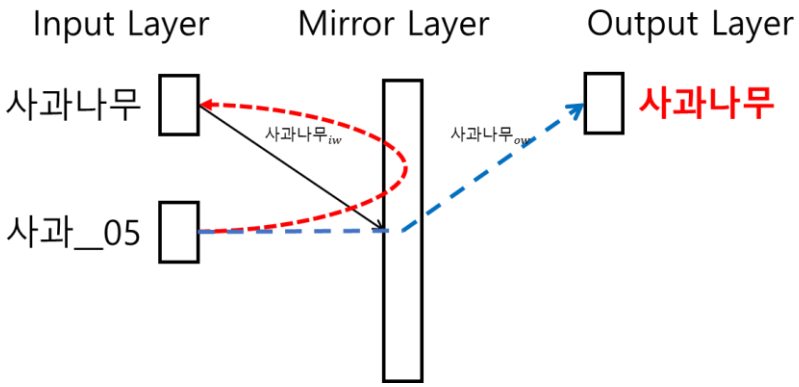
[그림 32]은 사전에 정의된 뜻풀이의 연관성을 표현한 그림이다. 그림에서 '사과'(사과나무의 열매)를 학습하는 경우 '사과'의 뜻풀이에 등장한 '사과나무'와 '열매'의 정의를 통해 '사과'라는 개념을 이해할 수 있다. 각 단어가 가진 뜻풀이가 가진 연관성을 학습하여 하나의 벡터로 표현하는 USenseVector는 '사과'의 뜻풀이 학습에서 '사과나무'나 '열매'가 가진 뜻풀이가 '사과'라는 단어의 벡터 구성에 영향을 주도록 설계된 모델이다.

USenseVector의 모델 구조는 Word2Vec[1]에서 사용한 Skip-gram의 학습 방식과 유사하다. 하지만 단어가 가진 뜻풀이의 연쇄성을 학습하기 위해 자질 거울(Feature Mirror) 방식을 사용했다.



[그림 33] USenseVector 학습 모델 구조

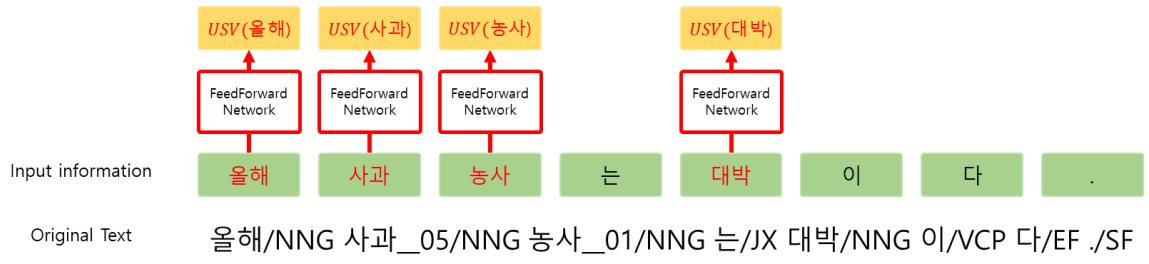
[그림 33]은 USenseVector가 사용한 자질 거울 모델을 표현한 그림이다. USenseVector는 하나의 단어가 가진 뜻풀이를 정답으로 사용했으며, [그림 33]에서 W_t는 학습의 대상이 되는 단어를 뜻하며, W_{d_1}~W_{d_n}은 단어 W_t의 뜻풀이에 있는 실질 형태소의 집합이다. 자질 거울 모델은 기존의 Skip-gram처럼 점선으로 표현된 가중치를 사용하는 것이 아닌, 입력 층과 거울 층 사이에 있는 각 자연어의 실제 가중치를 학습에 사용한 모델이다. 이 과정에서 오로지 학습의 대상이 되는 W_t와 연결된 가중치만 손실 값에 의해 갱신이 된다. [그림 32]처럼 뜻풀이에 사용된 자연어가 정의된 의미의 벡터 정보가 학습의 대상이 되는 단어에만 영향을 주며, 뜻풀이에 등장한 자연어는 각각의 뜻풀이에만 영향을 받게 된다.



[그림 34] USenseVector 모델이 '사과_05'를 학습하는 과정 예시

[그림 34]은 USenseVector의 학습 예시이다. '사과_05'를 학습하는 경우 뜻풀이에 등장한 실질 형태소인 '사과나무'와 '열매_01'가 정답이 된다. '사과_05'가 '사과나무'를 정답으로 추론하기 위해서는 파란색 점선인 사과_05$_{iw}$ × 사과나무$_{ow}$의 값이 증가해야 한다. USenseVector는 '사과나무'의 뜻풀이를 학습한 벡터 값이 '사과_05'의 학습에 영향을 주기 위해 빨간색 점선과 같이 '사과나무'의 입력 층과 거울 층 사이의 가중치인 사과나무$_{iw}$를 사용하여 추론한다. 사과나무$_{iw}$는 '사과나무'의 뜻풀이를 가지고 있는 가중치로 이를 사용함으로 인해 '사과나무'의 뜻풀이 정보가 '사과_05'에 내포되도록 학습한 모델이다. 이 과정에서 가중치 사과나무$_{iw}$를 갱신하지 않는다. USenseVector를 사용해 UWordMap에 등재된 총 792,367개의 동형이의어를 50차원의 벡터로 표현했다.

어휘 뜻풀이 기반 의미 벡터 학습은 대상으로 선정된 단어의 입력 벡터가 USenseVector로 표현한 벡터와 가까워지도록 학습했다. 뜻풀이를 학습한 벡터와 유사하도록 학습하여 자연어가 가진 뜻풀이 간의 연관성을 언어 모델이 이해하도록 설계한 학습 방식이다. 뜻풀이 기반의 의미 벡터 학습은 뜻풀이 기반 벡터를 가지는 형태소들 중에 임의로 최대 20개를 선정하여 학습에 사용했다.



[그림 35] 어휘 뜻풀이 기반 벡터의 언어 모델 학습 예시

[그림 35]는 어휘 뜻풀이 기반 의미 벡터를 언어 모델이 학습하는 예시이다. 문장에서 USenseVector로 표현한 의미 벡터를 가지는 단어들인 '올해/NNG', '사과__05/NNG', '농사__01/NNG', '대박/NNG'를 학습의 대상으로 선정되며, 명사의 상위어 학습과 마찬가지로 다중-핫 표현 입력 생성기를 통해 생성된 입력 값이 USenseVector로 표현된 벡터와 유사하도록 학습한다.

$$\mathcal{L}_{wme} = MSE(USV(x_i), FNN(input(x_i))) \quad \text{[수식 11]}$$

[수식 11]는 어휘 의미 학습에 선정된 형태소의 손실 값을 추론하는 수식이다. 여기서 $USV(x_i)$는 i 번째 형태소에 대해 USenseVector와 뜻풀이로 학습한 어휘 의미 벡터이다. $\mathrm{input}(\mathrm{x_i})$는 i번째 형태소에 대한 입력 벡터이며, 해당 입력 벡터를 순방향 네트워크(FeedForward Network)를 통과시켜 어휘의 뜻풀이 벡터와 동일한 차원으로 변환했으며 어휘 뜻풀이 벡터와 유사하도록 학습했다. 실제 학습 과정에서는 두 벡터에 대해 평균 제곱 오차(Mean Squared Error)를 사용하여 손실 값을 구했다.

4.3 용언의 제약정보를 활용한 어휘 간의 관계성 학습

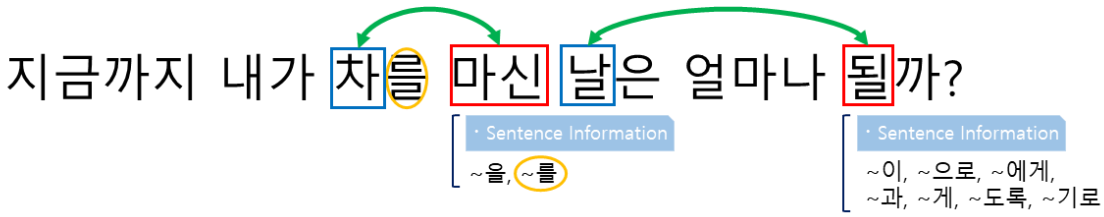
문장에서 단어 간의 관계성을 나타내는 정보로는 의존관계, 의미제약 등이 있다. 하지만, 이러한 정보를 대용량 말뭉치에 구축하기 위해서는 막대한 자원이 필요하다. 모두의 말뭉치 의존관계 말뭉치의 경우 신문기사 말뭉치 중에 300만 어절만 구축되어 있기 때문에 언어 모델의 사전 학습(pre-training)에 사용하기 어렵다. 그렇기 때문에 본 논문에서는 의미제약 정보를 사용하여 용언과 명사 간의 관계 정보를 언어 모델의 학습에 사용했다. 용언의 의미제약정보는 용언에 올 수 있는 문형정보와 문형정보 앞에 등장한 명사와의 관계성을 표현한 정보이다. 의미제약정보를 통해 문장의 중요 요소인 용언과 용언에 연관성이 있는 논항을 찾는데 도움이 되며, 문장에 중요한 요소들을 통해 문장을 파악하고 정확한 이해에 도움이 된다.

[그림 36] 용언 '막다_010101'에 대한 UWordMap의 의미 제약 정보

[그림 36]은 UWordMap에서 '막다'라는 용언에 실제로 구축된 의미제약정보를 보여준다. '막다'(길, 통로 따위가 통하지 못하게 하다.)의 문형 정보인 '을'에 올 수 있는 여러 명사(길, 정위, 공규, 구멍, 구멍노리, 공구, 공혈)들을 보여준다.

용언이 가지는 의미제약정보는 문장에서 사용된 용언과 다른 단어 사이의 관계성을 언어 모델이 이해하도록 학습했다. 용언의 의미제약정보를 통해 자연어 간의 의미적인 관계성을 어텐션 계층(attention layer)들이 습득하며, 문장에서 관계성이 높은 자연어들 간의 영향력이 커지도록 유도했다. 본 논문에서는 용언에 관련성이 높은 명사의 위치를 모델이 유추하는 학습 방법을 사용했다. 해당 용언의 앞에 등장한 단어를 대상으로 문형 정보를 이용해 대상이 되는 명사의 위치를 선정했다. 학습 데이터의 정답이 될 명사의 위치에 대한 선정은 다음의 규칙을 사용했다.

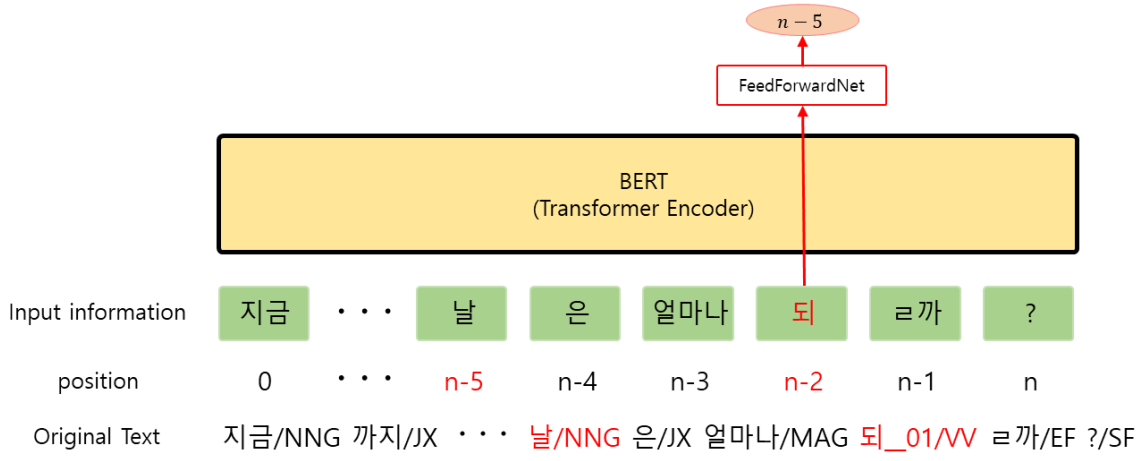
- 문장에서 해당 용언의 앞에 등장한 자연어만 탐색
- 다른 용언을 만나는 경우 탐색을 중지함
- 용언이 가진 문형 정보를 사용한 명사가 있으면 해당 명사를 정답으로 지정
- 문형 정보를 사용한 명사가 없는 경우 가장 가까운 명사를 지정
- 복합 명사일 경우 가장 마지막에 사용된 명사를 대상으로 선정
- 대상이 존재하지 않는 경우 자기 자신을 지정



[그림 37] 용언의 의미제약정보를 활용한 어휘 간의 관계 데이터 생성 방법

[그림 37]에서는 "지금까지 내가 차를 마신 날은 얼마나 될까?"라는 문장에서 의미제약 정보를 이용한 용언과 명사의 관계성 학습 데이터를 생성하는 방식을 보여준다. 문장에서 사용된 두 용언 중에 먼저 '마시다'의 경우 2가지 문형 정보인 '을'과 '를'을 가지고 있다. 관계를 가진 명사를 선정하는 규칙에 의해 '마시다'의 앞에 사용된 단어들 중에 조사 '를'이 존재하기 때문에 '를'의 앞에 등장한 명사 '차'가 관계성을 가진 명사로 선정되며, 해당 단어의 위치 정보가 학습 데이터로 구성된다. 반면 마지막에 사용된 용언인 '되다'의 경우 문형 정보가 문장에서 등장하지 않는다. 이러한 경우 가장 가까운 위치에 있는 명사인 '날'을 정답으로 선정했다.

용언과 의미제약을 이용해 구축한 데이터는 앞의 4.1, 4.2에서 소개한 상위어, 뜻풀이 기반의 학습과 달리 마스킹 학습과 동일하게 트랜스포머 모델의 출력을 사용해 관계를 가지는 명사의 위치를 추론하는 방식을 사용했다.



[그림 38] 용언의 관계성 학습 예시

[그림 38]은 용언의 관계성을 학습하는 예시를 보여준다. 앞서 [그림 37]의 예시 문장인 "지금__03/NNG+까지/JX 내__04/NP+가/JKS 차__09/NNG+를/JKO 마시/VV+ㄴ/ETM 날

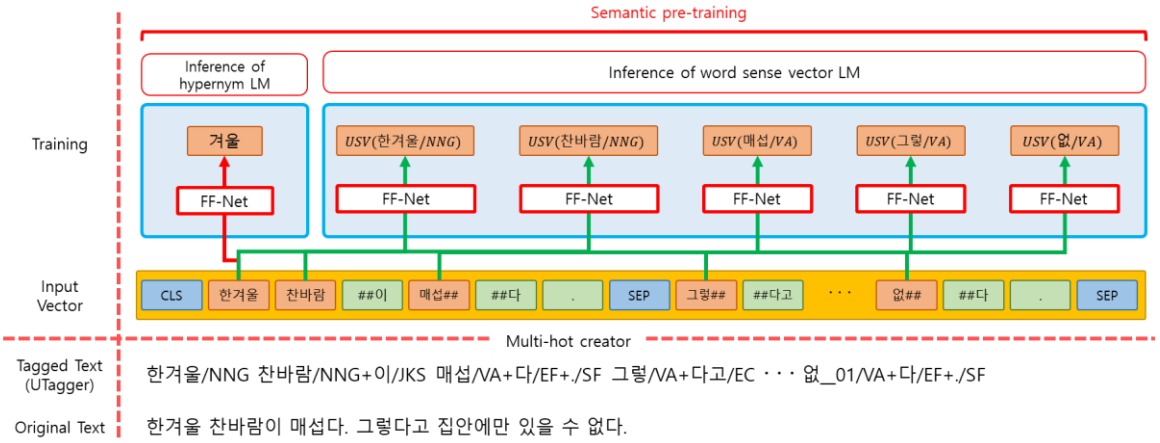
__01/NNG+은/JX 얼마나/MAG 되__01/VV+ㄹ까/EF+?/SF"라는 문장에서 뒤에 등장한 '되__01/VV'와 관련된 명사의 위치를 학습하는 방식을 보여준다. '되__01/VV'가 모델에서 입력된 위치 $n-2$의 결과 값을 사용해 규칙을 통해 관계를 가지는 '날/NNG'의 위치 값인 $n-5$의 위치를 맞추도록 설계했다.

$$\mathcal{L}_{verb\ relation} = SCE(VR_i, O_i) \quad \text{[수식 12]}$$

[수식 12]은 용언과 명사의 관계성 학습에 사용하는 손실 값을 구하는 수식이다. 용언의 관계성 학습을 위해 소프트맥스 크로스 엔트로피를 사용하여 손실 값을 구했다. 여기서 VR_i은 i번째 위치에 있는 용언과 관계성을 가지는 명사의 위치정보이며, O_i 용언이 사용된 위치의 모델의 출력 값이다.

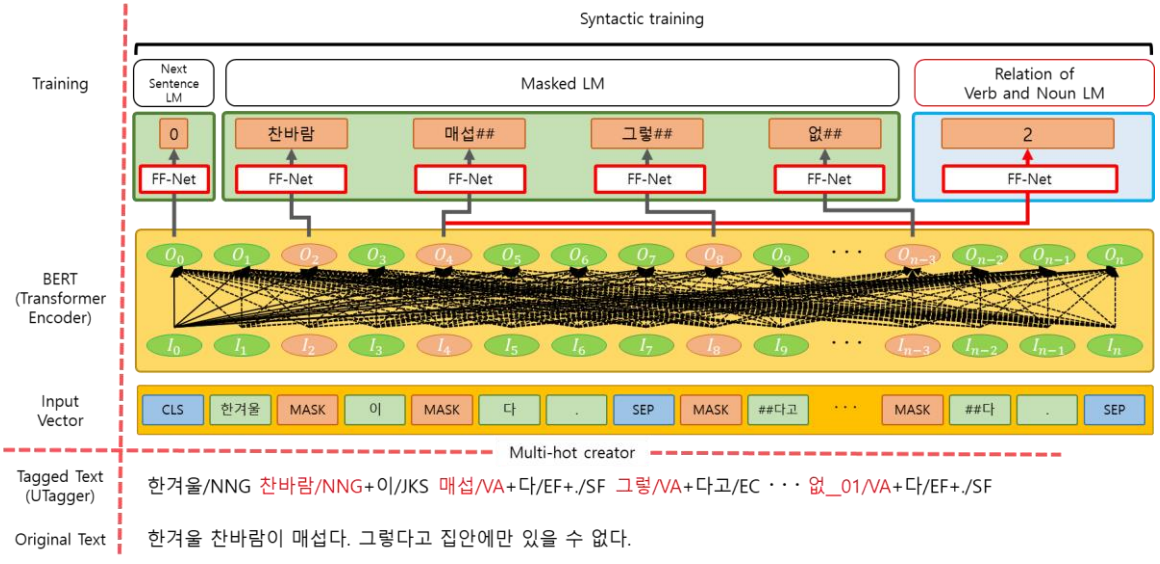
4.4 언어 지식을 융합한 언어 모델

언어 지식을 융합한 언어 모델을 위해 4.1, 4.2, 4.3에서 소개한 다양한 언어 지식을 기존 모델의 학습에 추가한 UKnowBERT(Ulsan Knowledge ensemble BERT)를 설계했다. 여러 언어 지식의 사전 학습을 통해 단어들이 가진 다양한 의미와 구조적인 정보를 언어 모델이 학습하도록 설계했다. 다양한 언어 지식은 UKnowBERT의 사전 학습에만 사용되며, 실제로 해당 모델을 응용 영역에서 사용할 경우 기존의 BERT와 동일하게 트랜스포머 인코더의 출력만 이용한다. UKnowBERT는 모델의 입력으로 형태소 단위를 사용하며, 다중-핫 표현 방식에 의해 형태소 분석된 문장을 토큰으로 변환했다. 토큰으로 표현된 각 형태소들은 다중-핫 입력 생성기에 의해 모델에 사용할 수 있는 입력 형태로 생성했다. 언어 지식의 성격에 맞춰 각기 다른 학습 방식을 사용했으며, 상위어 추론 학습과 뜻풀이 기반의 벡터(USenseVector)의 학습은 다중-핫 입력 생성기로 인해 생성된 값을 활용했다.



[그림 39] 상위어 추론과 뜻풀이 기반의 벡터 학습 예시

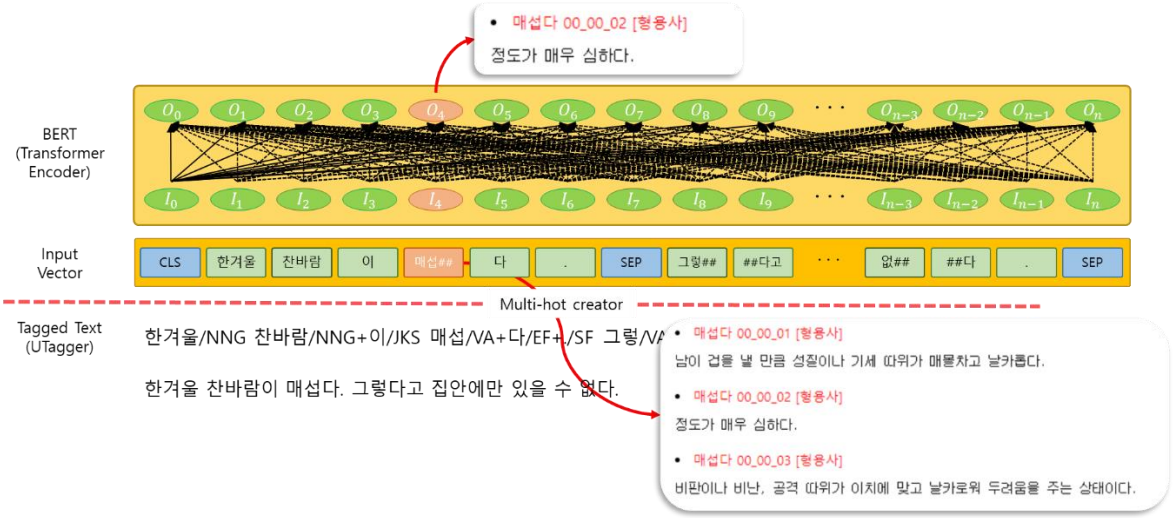
[그림 39]은 실제로 상위어 추론과 뜻풀이 기반의 벡터를 언어 모델의 입력 계층이 학습하는 방식을 보여준다. 예시 문장에서 등장한 토큰들 중에 상위어를 가진 '한겨울'이라는 토큰의 입력 벡터가 상위어인 '겨울'에 유사하도록 학습하며, '찬바람'의 경우 USenseVector로 표현한 '찬바람/NNG'의 벡터와 유사하도록 학습했다. 두 학습 모두 언어 모델의 출력을 이용하지 않고 입력 값을 이용한 이유로는 단어가 가진 의미를 입력 벡터에 녹여서 인간처럼 이미 습득한 지식을 통한 1차적으로 해당 단어의 의미를 판단하도록 하기 위함이다. 이러한 과정은 마스킹 학습과 유사하지만, 문맥 정보 없이 해당 형태소가 가진 다양한 의미를 임베딩 벡터에 포함하도록 했다. 다음은 언어 모델의 출력을 활용하여 용언의 의미제약과 기존의 두 학습에 대한 예시 그림이다.



[그림 40] 용언의 의미제약을 이용한 학습과 마스킹, 다음 문장 예측 학습의 예시

[그림 40]에서 용언의 관계 정보의 학습의 대상이 된 '매섭/VA'의 모델 출력 값이 관계를 가지는 '찬바람/NNG'의 위치 번호인 '2'를 유추하도록 학습했다. 이를 통해 용언과 관계를 가지는 명사의 위치를 추론함으로 인해, 문장에서 중요한 역할을 하는 명사와 용언의 다양한 관계를 언어 모델이 이해하도록 했다.

UKnowBERT의 경우 [그림 39], [그림 40]의 학습 방식을 모두 사용하여 사전 학습을 진행했다. UKnowBERT가 추구하는 문맥에 대한 해석의 방식은 다음과 같다.



[그림 41] UKnowBERT의 문장에서 단어의 의미 해석

[그림 41]은 실제 UKnowBERT가 추구하는 '매섭다'라는 단어를 표현하는 방식을 보여준다. 먼저 '매섭다'의 입력 임베딩 값에는 상위어와 뜻풀이 추론 학습을 통해 형태소가 가진 다양한 의미를 1차적으로 토큰의 벡터가 가지고 있으며, 언어 모델을 통해 문맥에 따라 '매섭다'에 대해 알맞은 의미를 출력하도록 설계한 모델이다.

5 실험

5.1 실험의 환경 구성

5.1.1 언어 지식을 사용하기 위한 토큰의 구성

실험에 사용할 토큰 목록을 구축하기 위해 형태소 분석을 완료한 신문 기사 말뭉치와 BPE(Byte-pair Encoding)를 사용해 토큰 목록을 생성했다. BPE 방식과 형태소 분석된 말뭉치를 통해 생성한 토큰들은 품사를 함께 가질 수 있다. 하지만, 다중-핫 표현 방식에서는 품사가 하나의 정보로 활용되기 때문에 토큰에서 품사의 분리가 필요하다. 하지만, 단순히 품사가 결합된 토큰에서 형태소를 분리하여 사용하는 경우 중복되는 토큰이 발생할 수 있다. 또한 형태소가 다양한 품사를 가지며, 그 형태소들 모두가 중요한 경우 하나의 토큰에 모든 정보가 담기는 문제가 발생한다. 예를 들어 '먹/VV'이라는 단어에 대해 형태소와 품사로 분리한 경우 '먹'이 토큰으로 등록된다. 하지만 '먹/NNG'과 동일한 토큰을 사용하기 때문에 하나의 토큰에 두 정보가 섞이는 문제점이 발생하게 된다.

본 논문에서 사용한 토큰 목록은 학습 말뭉치에 등장한 품사의 비율과 BPE로 생성한 토큰들에 대해 변환 규칙을 제정하고 적용해 구축했다. 빈도에 따라 해당 품사의 중요도를 판단해 다른 표현법을 적용했다. 먼저 토큰을 생성하기 위해 사용한 신문 기사 말뭉치의 품사별 등장 빈도를 측정했다.

[표 5] 모두의 말뭉치에서 품사별 빈도 통계

품사	세종태그	빈도	비율
일반명사	NNG	594,524,670	26.4708%
고유명사	NNP	81,423,660	3.6253%
의존명사	NNB	88,177,246	3.9260%
대명사	NP	12,026,218	0.5355%
수사	NR	9,075,704	0.4041%
동사	VV	196,841,657	8.7642%
형용사	VA	49,358,172	2.1976%
보조동사	VX	31,952,000	1.4226%
긍정지정사	VCP	29,440,638	1.3108%
부정지정사	VCN	2,207,655	0.0983%
수관형사	MMN	6,376,521	0.2839%
지시관형사	MMD	8,957,564	0.3988%
성상관형사	MMA	5,570,288	0.2480%
일반부사	MAG	37,921,503	1.6884%
접속부사	MAJ	2,712,546	0.1208%
감탄사	IC	340,899	0.0152%
조사	JKS, JKC, JKG, JKO, JKB, JKV, JKQ, JX, JC	359,996,940	16.0286%
선어말어미	EP	46,534,918	2.0719%
종결어미	EF	78,384,610	3.4900%
연결어미	EC	113,045,530	5.0333%
명사형전성어미	ETN	9,127,909	0.4064%
관형형전성어미	ETM	113,635,674	5.0595%
접두사	XPN	6,731,501	0.2997%
명사파생접미사	XSN	58,684,036	2.6129%
동사파생접미사	XSV	1,015,152	0.0452%
형용사파생접미사	XSA	105,898	0.0047%
기호	SE, SF, SN, SO, SP, SS, SW	202,772,482	9.0283%
외래어, 한자	SL, SH	18,732,688	0.8341%
숫자	SN	65,420,385	2.9128%
합계		2,245,967,942	

[표 5]의 모두의 말뭉치 신문 기사 말뭉치에서 품사별 등장 비율을 보여준다. 품사의 등장 빈도가 높은 체언류와 용언, 조사, 어미류에 대해서 형태소를 보존하기 위해 특수한 규칙을 적용했다. 품사에 따른 토큰 구성 규칙을 보여준다.

- 용언과 접두사, 동사파생접미사, 형용사파생접미사의 경우 원본 토큰 뒤에 '##' 기호를 추가한다.
- 어미, 조사, 명사파생접미사는 원본 토큰 앞에 '##' 기호를 부착한다.
- 체언과 이외의 품사가 등장한 토큰의 경우 품사만 제거하여 사용한다.
- BPE로 인해 품사 없이 등록된 토큰들은 그대로 사용한다.

[표 6] 각 형태소 토큰에 규칙을 적용한 예시

원본 토큰	규칙을 적용한 후 생성된 토큰
사과/NNG	사과
잠재적/NNG	잠재적
잠재적/MMA	
마시/VV	마시##
좋/VA	좋##
답/XSA	답##
시키/XSV	시키##
상/XSN	##상
의/JKG	##의
액화	액화
서면	서면

[표 6]는 형태소 품사에 따른 규칙을 사용하여 BPE로 구축한 토큰을 변경한 예제이다. BPE로 구성한 토큰 목록에 대해 해당 토큰이 가지고 있는 품사에 따라 본 논문에서 사용할 토큰들로 변환했다. 하지만 변환 과정에서 중복된 토큰에 대해 하나의 토큰만 사용했다. 만약 위의 규칙에 포함되지 않은 빈도가 낮은 품사들을 사용한 형태소가 동일한 토큰으로 표현되어도 입력 단계에서 품사 정보를 사용하기 때문에 구분이 가능하다.

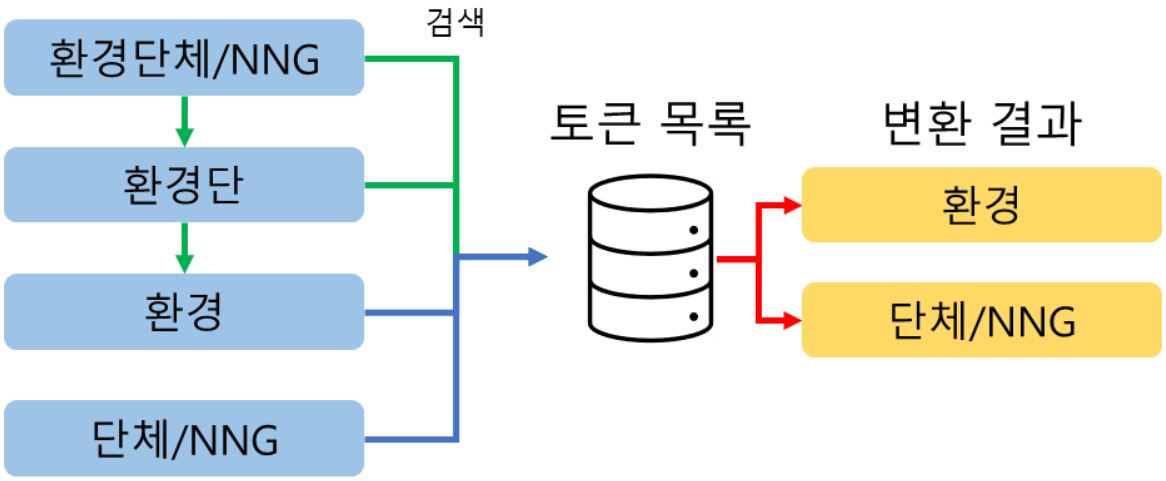
BPE와 품사의 빈도에 따른 규칙을 통해 수집한 토큰들 이외에도 하나의 토큰으로 형태소를 표현하기 위해 음절 토큰을 사용했다. 음절 토큰의 목록을 구축하기 위해 말뭉치에서 등장한 형태소가 하나의 토큰으로 표현되지 않는 모든 형태소를 수집했다. 수집한 형태소들에 대해 음절 단위로 분리를 실시했으며, 각 음절에 해당하는 빈도수를 구했다.

수집한 정보를 바탕으로 50회 이상 등장한 음절들을 모아 음절 토큰으로 사용했다. 만약 현재 형태소가 음절 토큰 목록에 존재하지 않은 음절을 사용한 경우 해당 형태소를 미등록 토큰(unknown 토큰)으로 변환했다. 음절 토큰은 1음절 형태소와의 차별을 위해 음절 앞에 '@' 기호를 부착했다. 예를 들어 '소크라테스'라는 단어를 음절 토큰으로 표현하면 '@소', '@크', '@라', '@테', '@스'가 된다. 그리고 '강'(대표적인 뜻: 넓고 길게 흐르는 큰 물줄기.)이라는 단어가 토큰으로 등록된 경우 음절 토큰 목록에 '강'이 있으면 구분을 위해 '@강'으로 표시했다.

숫자의 경우 자릿수에 따라 표현 방식을 분할했다. 한 자릿수 숫자의 경우 그대로 0부터 9라는 토큰을 사용했으며, 두 자릿수 이상 숫자와 구분을 위해 뒤에 각 숫자 뒤에 '##'을 부착하여 0에서 9까지의 토큰으로 구성했다. 예를 들어 '20'의 경우 '2##'과 '0##'으로 표현된다. 마지막으로 영어의 경우 알파벳 대소문자를 모두 토큰으로 등록했으며, 한자의 경우 특수 토큰인 '[CHC]'를 사용했으며, 나머지 언어의 경우 '[OTL]'를 사용했다.

5.1.2 형태소의 의미 유지를 위한 토큰 분리 방식

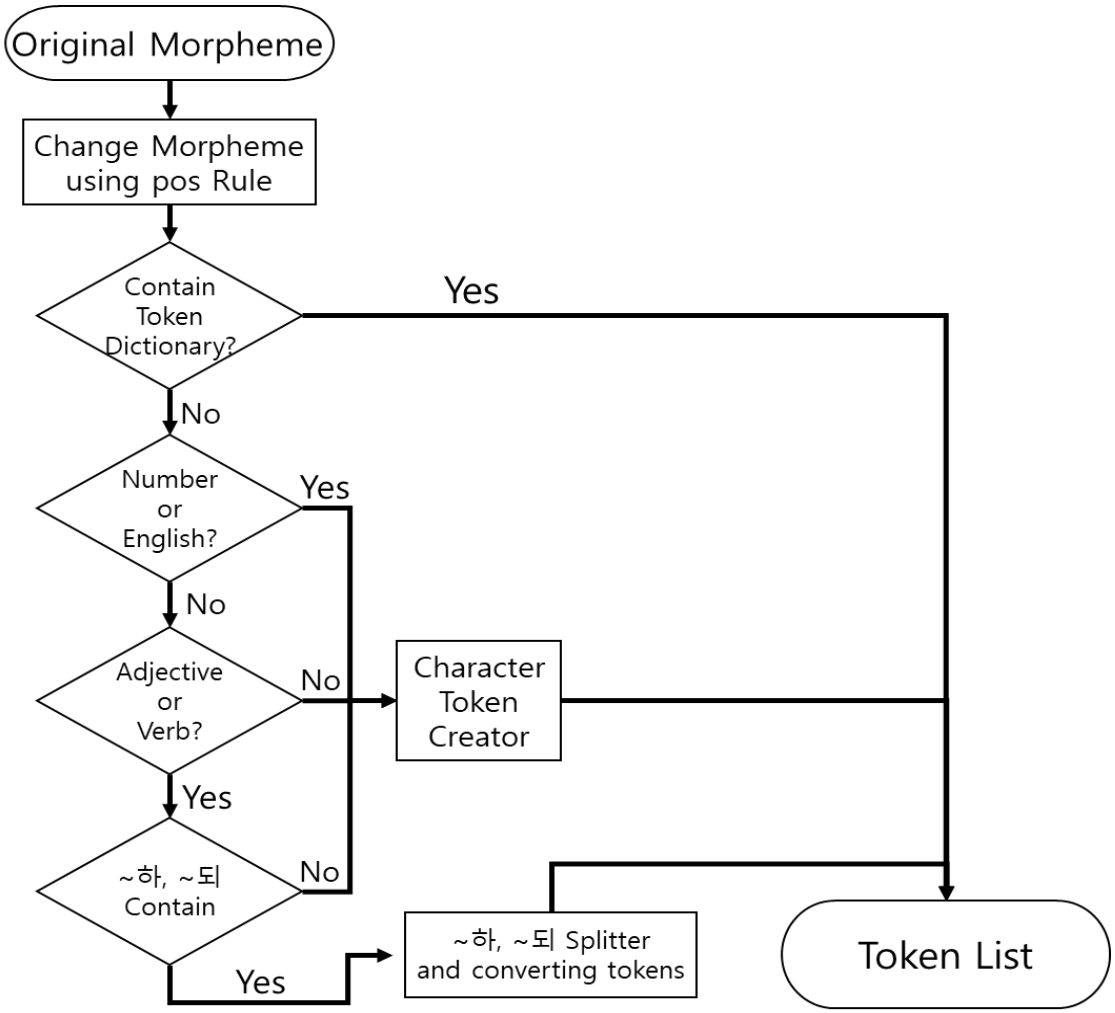
기존의 언어 모델이 사용한 단어의 토큰 변환 및 분리 방식은 하나의 어절이나, 형태소에 대해 최장 일치를 기준으로 토큰으로 변환하는 방법을 사용했다. 토큰 표현 대상인 단어에 대해 대응하는 토큰이 존재하지 않는 경우 해당 단어의 마지막 한 글자를 제거하여 다시 검색하여 구성 목록에 존재할 때까지 반복한다.



[그림 42] ' 환경단체/NNG'라는 형태소에 대해 기존의 언어 모델의 토큰 분리 방식

[그림 42]는 '환경단체/NNG'라는 형태소가 기존의 언어 모델에서 토큰으로 변환하는 과정을 표현한 그림이다. '환경단체/NNG'가 토큰 사전에 등재되지 않은 경우 마지막에 한 음절인 '체/NNG'를 제외한 '환경단'을 토큰 목록에서 검색한다, 만약 '환경단'도 등재되지 않은 경우 '단'을 제거한 '환경'이 토큰으로 등록되었는지 찾는 방식이다. 기존의 토큰 분리 방식은 간단한 방법으로 효율적으로 단어를 분리하여 표현하는 방법에 속한다. 하지만 [그림 42]의 토큰 분리 방식도 모든 단어를 표현할 수 없다. 그리고 마지막 글자를 하나씩 제거하는 분리 방법의 경우 영어에 적합한 방식이다. 예를 들어 영어의 경우 ing, ly, est처럼 접미사가 사용된 단어를 분리하기에 유리하다. 하지만, 한국어나 한자와 같은 언어의 경우 분리되는 위치에 따라 의미의 소실이나 변질이 일어날 수 있다.

본 논문에서는 다중-핫 표현 방식과 형태소 단위의 유지를 위해 형태소가 가지는 품사에 따라 분리하는 방법을 다르게 했다.



[그림 43] 다중-핫 표현 모델에서 토큰 분리의 순서도

[그림 43]은 본 논문에서 사용한 형태소의 품사에 따른 토큰 분리 방법의 순서도이다. 입력된 형태소가 가진 품사에 따라 다른 변환 규칙을 사용했다. 먼저 입력 받은 형태소와 품사를 [표 6]의 규칙을 통해 분리했다. 예를 들어 '광학/NNG'라는 단어는 '광학'과 'NNG'로 분리했다. 형태소가 토큰 목록에 존재하는지 여부를 확인하며, 존재하면 해당 토큰으로 바로 변환했다. 만약 형태소가 토큰 목록에 존재하지 않는 경우 품사 정보를 활용해 숫자, 영어인지 판단했다. 숫자나 영어인 경우 음절 단위로 분리하여 토큰으로 변환했다. 마지막으로 품사가 용언인 경우 '~하다', ~'되다'로 구성된 용언인지 확인했으며, 해당 형태소가 '~하다', '~되다'의 형태가 아니면 숫자, 영어와 마찬가지로 음절 단위로 분리했다. 하지만 '~하다'와 '~되다'의 형태인 경우 '하'와 '되'를 분리해 처리했다. '~하다'와 '~되다'가 분리된 단어에 대해 토큰 목록에서의 검색을 진행하며, '하'와 '되'가 분리되고 남은 부분이 토큰 목록에 존재하지 않는 경우 음절 토큰으로 변환했다. '~하다'와 '~되다'로 인해 분리된 토큰의 경우 음절 토큰과 동일하게 토큰 집합으로 표현되어 반환된다.

[표 7] 입력된 형태소에 따른 토큰 분리 결과 예제

형태소	품사 변환	토큰화
서울/NNP	서울	[서울]
공원/NNG	공원	[공원]
신기루/NNG	신기루	[@신, @기, @루]
당하/VV	당하##	[당하##]
놀랍/VA	놀랍##	[놀랍##]
동의하/VV	동의하##	[동의, 하##]
투입되/VV	투입되##	[투입, 되##]
삭히/VV	삭히##	[@삭, @히]
꼼꼼히/MAG	꼼꼼히	[꼼꼼히]
1/SN	1	[1]
700/SN	700	[7##, 0##, 0##]
3.14/SN	3.14	[3##, .##, 1##, 4##]
2,000/SN	2,000	[2##, ,##, 0##, 0##]
Seed/SL	Seed	[S, e, e, d]

[표 7]은 [그림 43]의 과정을 통해 하나의 형태소가 토큰으로 표현된 예시를 보여준다. 먼저 토큰 목록에 등재된 '서울/NNP'와 '공원/NNG'의 경우 품사를 제거한 '서울'과 '공원'

이 토큰으로 변환된다. 하지만 토큰 목록에 없는 형태소인 '신기루/NNG'의 경우 음절 토큰인 '@신', '@기', '@루'로 변환되며 3개의 토큰을 묶어서 반환했다. 용언인 '동의하다'의 경우 '하다'를 가진 형태이기 때문에 '하'를 먼저 분리하고, 나머지 '동의'가 토큰 목록 존재 여부를 확인했다. 이 경우 존재하기 때문에 '동의'와 '하' 두 토큰의 집합으로 표현된다. 숫자의 경우 사용된 한자리 이상인 경우 '##'이 부착된 숫자 토큰의 집합으로 표현된다. 만약 숫자에 자릿수 구분을 위한 ','나 소수점인 '.'이 사용된 경우 기존의 기호와 구분하기 위해 ',##', '.##'으로 표현했다. 마지막으로 영어 단어는 대소문자를 구분하여 음절 토큰의 집합으로 표현했다.

본 논문에서 언어 지식을 위해 구성한 토큰 목록과 [그림 43]의 토큰 분리 방식이 실제 사전 학습(pre-training) 말뭉치에 적용된 결과의 통계를 측정했다. 통계 결과를 통해 다중-핫 표현 방식과 기존의 토큰 표현 방식이 형태소에 어떠한 영향을 주는지 확인했다. 통계를 측정하기 위한 대상 말뭉치로는 모두의 말뭉치 신문 기사 말뭉치를 사용했다.

[표 8] 사전 학습 말뭉치에서 사용된 토큰 방식에 따른 통계

종류	BPE기반	다중-핫 표현 방식 기반
사용한 토큰 목록	37,027	20,892
전체 토큰의 수	2,365,791,718	2,245,984,111
unknown 토큰의 수	20,335,168 (0.85%)	1,945,383 (0.0008%)
토큰 집합으로 표현된 형태소의 수	x	502,440,172 (21.23%)

[표 8]은 학습 말뭉치에서 각각의 토큰 목록과 토큰 분리 방식에 의한 통계 수치를 보여준다. 기존의 BPE 방식의 경우 미등록 토큰으로 변환되는 토큰의 수가 전체 토큰의 약 0.85%를 차지했다. 반면 다중-핫 표현 방식의 경우 0.0008%가 미등록 토큰으로 변환되었다. 미등록 토큰의 예로 BPE 기반에서는 '멱살', '바삐'와 같은 단어들이 등록되어 있지 않았으며, 다중-핫 표현 방식에서는 '끙끙', '뭥미'와 같은 단어들이 미등록 토큰으로 표현되었다. [표 8]의 결과를 통해 다중-핫 표현 방식을 통해 더 적은 토큰의 수로 더 많은 토큰들을 표현 가능함을 확인할 수 있었다. 그리고 다중-핫 표현 방식의 경우 전체 형태소의 21.23%가 토큰들의 집합 형식으로 형태소를 표현했다.

5.1.3 모델의 성능 비교를 위한 실험 영역 설정

다중-핫 표현 방식의 적합성과 언어 지식을 융합한 언어 모델의 성능 평가 실험을 위해 다양한 자연어 처리 영역을 선정했다. 다양한 자연어 처리 응용 영역에서의 평가 실험을 통해 언어 지식을 융합한 언어 모델이 어떠한 영역에서 강점과 약점을 가지는지 분석했다.

[표 9] 언어 모델의 평가를 위한 데이터의 상세 정보

평가 영역	상세 영역	데이터 구성 (문장 단위, 개)		데이터 출처
기계 독해	KorQuAD v1.0[50]	학습	60,407	https://korquad.github.io/KorQuad%201.0/
		테스트	5,774	
문장 분석	개체명 인식	학습	120,066	모두의 말뭉치(개체명 인식 데이터)
		테스트	30,018	
	의미역 인식	학습	108,509	모두의 말뭉치(의미역 인식 데이터)
		테스트	27,501	
감성 분석	네이버 영화 감성 분석 데이터 (NSMC)	학습	149,996	https://github.com/e9t/nsmc
		테스트	49,997	
의미 분별	동형이의어 의미 분별	학습	722,913	세종 말뭉치
		테스트	80,321	
	다의어 의미 분별	학습	121,563	모두의 말뭉치 (어휘 의미분석 말뭉치)
		테스트	13,500	

[표 9]은 본 논문에서 제안한 언어 모델의 평가를 위해 사용한 학습 및 실험 데이터의 자료이다. 평가 영역은 크게 4가지로 구성되어 있으며 총 5개의 실험 영역으로 구성되어 있다. 실험에 선정된 영역들은 각 영역이 요구하는 능력이 다르도록 구성되어 있으며, 제안 모델을 다각도로 기존 모델과 비교하기 위해 선정했다.

기계 독해 영역은 자연어 표현 모델이 문장을 넘어선 문단이나, 문서 단위에서의 이해력을 평가하기 위해 사용했다. 언어 모델의 기계 독해 능력을 평가하기 위해 질의응답 학습 데이터인 LG CNS에서 구축한 KorQuAD(Korean Question Answering Dataset) v1.0[50]

을 사용했다. 위키백과의 정보를 사용해 한국어 기반의 질의응답 데이터로 스탠포드에서 구축한 SQuAD(Stanford Question Answering Dataset)의 구축 기준을 사용했다.

문장 분석 평가 영역에서는 문장에서 사용된 단어 중에 특별한 개념을 가지는 단어들을 찾고, 해당 단어에 대한 영역 분류를 언어 모델이 수행 가능한지 판단했다. 문장 분석 평가는 문장에 존재하는 특별한 요소에 대해 언어 모델의 이해력을 평가하기 위해 사용했다. 문장 분석 평가를 위해 개체명과 의미역 인식을 사용했다. 학습 말뭉치는 국립국어원에서 구축하고 공개한 모두의 말뭉치를 사용했다. 개체명 인식은 총 15개의 태그 셋으로 구축되어 있으며, 의미역은 18개의 태그 셋으로 구성되어 있다.

개체명 인식

②/SW 두__01/MMN 손__01/NNG+으로/JKB 아랫배/NNG+를/JKO 두드리/VV+ㄴ다/EF+./SF

의미역 인식

부상자/NNG+도/JX 24.7/SN+%/SW 감소하/VV+았/EP+다/EF+./SF [SEP] 감소하/VV+았/EP+다/EF+./SF

[그림 44] 개체명 인식과 의미역 인식의 실험을 위해 사용한 입력의 방식

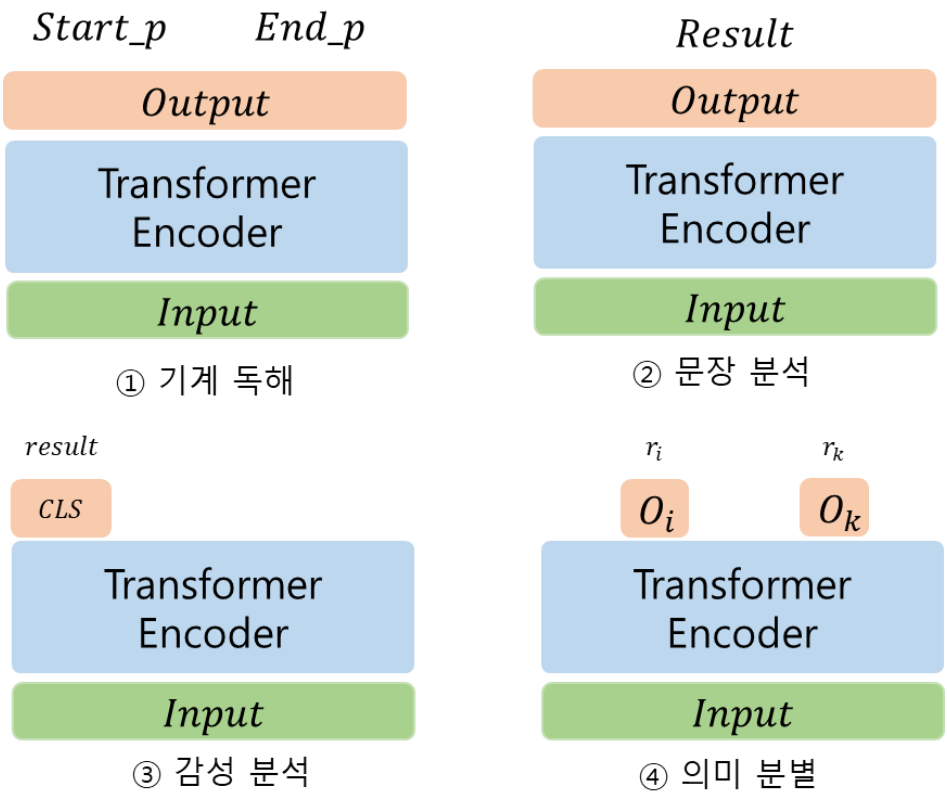
[그림 44]은 개체명 인식과 의미역 인식의 실험에서 사용한 입력의 구조이다. 개체명의 경우 문장을 그대로 사용하지만, 의미역의 경우 대상으로 선정된 용언이 속한 어절 전체를 문장 구분자인 [SEP]와 함께 입력 문장의 뒤에 위치시켰다. 두 영역 모두 BIO(Begin, Inside, Outside) 태그 방식을 사용한 학습 형태를 사용했다.

감성 분석 영역의 경우 언어 모델의 사전 학습에 사용한 말뭉치와 전혀 다른 성격을 가진 구어체 말뭉치에 대해 언어 지식을 사용함에 따른 언어 모델에 가져오는 영향력을 확인하기 위해 사용했다. 학습 말뭉치로 네이버의 영화 리뷰 댓글에 대해 긍정, 부정 평가를 부착한 말뭉치를 사용했다. 네이버 영화 리뷰 감성 분석 데이터는 영어권의 영화 리뷰 사이트인 IMDB에서 리뷰를 수집하여 구축한 감성 분석 데이터 방식으로 구축된 데이터이다.[51]

마지막으로 의미 분별 영역은 문장에서 사용된 자연어가 가진 의미를 언어 모델이 이해하고 구분되는지 확인했다. 의미 분별을 위해 세종 말뭉치와 모두의 말뭉치의 어휘 의미 분석 말뭉치를 사용했다. 의미 분별의 대상이 되는 단어들은 말뭉치 상에서 어깨번호를 가지는 단어 전부를 대상으로 했다. 예를 들어 "대단히/MAG 잘/MAG+하/XSV+아/EC+주__013/VX+었/EP+다/EF+"/SS+고/JKQ 치켜세우__002/VV+었/EP+다/EF+./SF"라는

문장에서 의미 분별의 대상이 되는 단어는 '주__013/VX'와 '치켜세우__002/VV'로 선정된다. 대상으로 선정되는 단어에 대해 아무런 사전 정보 없이 현재 문장에서의 사용된 의미에 맞는 의미 번호를 추론하도록 설계했다.

언어 모델을 평가하기 위한 4가지 영역에 대해 각 영역에 맞춰 평가 모델을 구축했다. 각 영역의 실험을 위해 설계한 모델들은 각 응용 영역에서의 최고 성능을 달성하기 위한 목적이 아닌 기존 모델과의 성능 비교가 목적이기 때문에 각각의 학습 데이터에 맞춰 분별을 위한 구조만 추가했다.



[그림 45] 각 평가 영역별 사용한 모델의 구조

[그림 45]은 각 평가 영역에 사용한 모델의 구조를 보여준다. 기계 독해의 경우 입력으로 질의문과 지문이 주어지며, 지문에 정답의 시작 부분과 끝 부분을 추론하는 방식으로 구성했다. 문장 분석 영역에서는 모든 입력에 대해 특정한 분류에 속하는지 판단하며, 분류까지 구분하도록 설계했으며, 감성 분석 영역은 입력 받은 문장에 대해 첫번째 출력 값을 사용해 긍정과 부정을 판단했다. 마지막으로 의미 분별 모델은 입력으로 사용한 문장에서 의미 번호를 가지는 형태소 위치의 출력을 사용하여 의미 번호를 판단했다.

5.1.4 실험 영역의 말뭉치 분석

실험을 위해 선정한 4가지 영역에 대해 말뭉치 분석을 실시했다. 토큰 표현 방식으로 인해 형태소를 얼마나 분리하는지 파악하며, 이를 통해 실험 결과의 분석에 활용했다. 전체 형태소 대비 토큰으로 분리되는 형태소의 개수를 구했으며, 그 중에 일반명사와 고유명사의 비율을 계산했다.

[표 10] 기존의 토큰 표현 방식을 사용한 각 실험 영역 말뭉치 분석

학습 데이터	종류	형태소 수	형태소 분리 비율	명사의 분리 비율
KorQuAD v1.0	학습	15,019,855	9.47% (1,423,020)	73.35% (1,043,787)
	테스트	1,489,542	8.59% (127,920)	72.39% (92,605)
개체명	학습	3,433,515	5.56% (191,029)	76.00% (145,175)
	테스트	857,113	5.55% (47,567)	76.20% (36,244)
의미역	학습	3,285,176	5.45% (178,925)	76.29% (136,501)
	테스트	835,151	5.07% (42,358)	76.33% (32,331)
NSMC	학습	2,950,238	6.80% (200,661)	73.60% (147,696)
	테스트	987,361	6.74% (66,555)	73.59% (48,980)
세종 말뭉치	학습	19,494,545	6.25% (1,219,011)	67.56% (823,568)
	테스트	2,344,731	6.15% (144,261)	68.76% (99,196)
모두의 말뭉치	학습	4,008,364	5.28% (211,449)	83.99% (177,586)
	테스트	496,836	5.18% (25,760)	84.76% (21,835)

[표 10]은 기존의 토큰 방식을 사용해 각 실험 영역들에 대해 형태소가 여러 토큰으로 분리되는 경우와 그 경우 중에 명사의 비율을 보여준다. 기계 독해 영역의 경우 나머지 영역에 비해 형태소의 분리 비율이 높았다. 그리고 분리된 형태소의 70% 정도가 명사였다. 다음은 본 논문에서 제안한 다중-핫 표현 방식을 사용해 동일한 분석을 진행했다.

[표 11] 다중-핫 표현 방식을 사용한 경우에 대해 각 실험 영역 말뭉치 분석

학습 데이터	종류	형태소 수	토큰 집합 표현 비율	명사의 분리 비율
KorQuAD v1.0	학습	15,019,855	25.90% (3,890,043)	26.32% (1,023,772)
	테스트	1,489,542	25.14% (374,424)	24.28% (90,908)
개체명	학습	3,433,515	22.08% (757,957)	18.45% (139,814)
	테스트	857,113	22.07% (189,125)	18.47% (34,938)
의미역	학습	3,285,176	22.29% (732,317)	18.07% (132,314)
	테스트	835,151	21.75% (181,653)	17.15% (31,150)
NSMC	학습	2,950,238	26.47% (780,792)	22.89% (178,701)
	테스트	987,361	26.38% (260,435)	22.70% (59,125)
세종 말뭉치	학습	19,494,545	24.13% (4,703,830)	16.99% (799,401)
	테스트	2,344,731	24.20% (567,437)	16.93% (96,071)
모두의 말뭉치	학습	4,008,364	20.88% (836,973)	19.07% (159,637)
	테스트	496,836	20.61% (102,393)	19.09% (19,544)

[표 11]은 본 논문에서 제안한 다중-핫 표현 방식을 사용해 각 영역의 형태소가 토큰의 집합으로 표현된 경우의 비율과 그 중에 품사가 명사인 비율을 보여준다. 다중-핫 표현을 사용한 경우 전체 형태소의 20~26%가 토큰들의 집합으로 표현됐다. 명사의 비율은 달랐지만, [표 10]에서 품사가 명사인 개수는 비슷하게 나타났다.

5.2 다중-핫 표현 방식의 적합성과 손실 함수의 선정

본 논문에서 제안한 다중-핫 표현 방식은 형태소가 가진 의미의 보존과 형태소 단위로 구축한 언어 지식을 언어 모델에 쉽게 적용하기 위해 사용했다. 언어 지식을 사용한 UKnowBERT 모델 평가에 앞서 다중-핫 표현을 사용한 언어 모델과 기존 모델을 비교했다. 비교 평가를 통해 다중-핫 표현 방식이 언어 모델에 미치는 영향력을 분석했다.

먼저 다중-핫 표현 방식을 사용한 언어 모델의 학습에 적합한 손실 함수를 선정하는 실험을 진행했다. 다중-핫 표현은 마스킹 학습에서 마스킹의 대상이 되는 토큰이 음절 단위의 집합으로 표현된 토큰들이 선정될 수 있으며, 이 경우 정답이 1개 이상인 문제(multi-label classification)로 정의할 수 있다. 그렇기 때문에 이에 적합한 손실 함수를 찾는 것이 선행 실험으로 진행되었다. 실험을 위해 기존의 바이너리 크로스 엔트로피와 소

프트맥스 크로스 엔트로피를 비교 대상으로 선정했으며, 본 논문에서 제안한 정답의 수에 따른 목표 확률을 조정하는 소프트맥스 크로스 엔트로피의 실효성을 판단했다.

다중-핫 표현 방식에 적합한 손실 함수의 선정 실험을 통해 선택된 손실 함수를 사용하여 학습한 다중-핫 표현 방식을 사용한 언어 모델과 기존의 BERT를 다양한 영역에서 비교 실험을 진행했다. 실험을 통해 다중-핫 표현 방식을 통한 형태소 단위의 유지가 언어 모델에 미치는 영향력을 확인했다.

5.2.1 다중-핫 표현 방식의 적합성 실험을 위한 언어 모델의 설정

다중-핫 표현 방식의 적합성 실험은 기존의 BERT를 비교 대상으로 선정했으며, BPE를 사용해 토큰 목록을 구축했다. 다중-핫 표현을 사용한 언어 모델이 사용할 토큰 목록과 분리 방식은 5.1.1, 5.1.2에서 제안한 방식을 사용했다. 토큰 목록을 구성하기 위해 모두의 말뭉치에서 신문기사 말뭉치를 사용했다.

[표 12] 다중-핫 표현 적합성 실험에 사용한 토큰 구성

토큰 생성 기반 / 토큰 종류	Byte-pair Encoding (비교 모델 용)	Byte-pair Encoding + 음절 토큰
일반 토큰	37,027	18,898
음절 토큰	×	1,994
합계	37,027	20,892

[표 12]은 각 모델들이 사용한 토큰 목록의 구성 및 개수이다. 실험의 비교 대상인 BERT는 총 37,027개의 토큰으로 사용했다. 본 논문에서 제안한 다중-핫 표현 언어 모델의 경우 BPE로 구축한 37,027개의 토큰에 대해 5.1.1에서 정의한 방식을 사용해 총 18,898개의 토큰 목록을 생성했다. 그리고 다중-핫 표현을 위해 학습 말뭉치에서 하나의 토큰으로 표현이 불가능한 형태소들을 수집하여 음절 단위의 빈도수를 통해 음절 토큰을 구성했다. 등장 빈도가 50번 이상인 음절들과 숫자, 영어를 표현하기 위한 토큰들을 합쳐 총 1,994개의 음절 토큰을 구성했다.

다중-핫 표현 방식의 적합성 실험에 사용한 모델들은 기존의 BERT가 사용한 마스킹 학습과 다음 문장 예측 학습만을 사용했다. 모델 크기에 따른 다중-핫 표현 방식이 언어

모델에 가져오는 영향을 함께 측정하기 위해 두 가지(small, base) 크기를 선정해 사전 학습을 진행했으며, 다양한 영역에서 다중-핫 표현의 적합성을 실험했다. 아래의 표는 실험에 사용한 두 가지 크기 언어 모델에 대한 설정표이다.

[표 13] 다중-핫 표현 모델의 실험을 위한 모델 설정

설정 종류		small	base
Training Corpus		모두의 말뭉치 (신문기사 말뭉치)	
Transformer layer		4	12
Attention header		4	12
Hidden layer size		256	768
Maximum token length		256	512
Batch Size		128	128
Training Step		1,000,000	1,000,000
Learning rate		0.0001	0.0001
Model Size	one-hot	49MB	436MB
	multi-hot	33MB	389MB
Machine		TPU-v2	TPU-v3

[표 13]는 실험에 사용한 두 가지 크기(small, base)의 모델들의 설정을 보여준다. 두 가지 모델은 트랜스포머 레이어의 수, 은닉 층의 크기가 다르며, 사용한 토큰 표현 방식에 따라 토큰의 양과 모델의 크기가 다르다. 다양한 크기와 설정을 통해 다중-핫 표현 방식이 언어 모델에 미치는 영향력을 판단했다.

5.2.2 다중-핫 방식을 사용한 다양한 손실 함수 실험 결과

다중-핫 표현을 사용한 모델과 기존 BERT와의 성능 평가에 앞서 다중-핫 표현을 사용한 언어 모델의 학습에 적합한 손실 함수를 선정하기 위한 실험을 진행했다. 다중-핫 표현 방식을 위한 손실 함수 선정 실험은 마스킹 학습을 대상으로 진행했다. 실험을 위해 비교군으로 설정한 손실 함수는 BERT에서 손실 함수로 사용된 소프트맥스 크로스 엔트로피와 다중 선택 문제(multi-label classification)에서 사용하는 바이너리 크로스 엔트로피를 선정했다. 3가지 손실 함수에 대해 비교 평가를 진행했다. 사용한 모델의 크기는 [표 13]에서 small 크기이며, 음절 토큰이 포함된 20,892개의 토큰을 사용했다.

[표 14] 손실 함수에 따른 다중-핫 표현 방식의 성능

실험영역 \ 손실함수		Binary-Cross Entropy	Softmax-Cross Entropy	Softmax-Cross Entropy (multi-hot base)
KorQuAD v1.0	Exact	43.38%	69.81%	**73.62%**
	F1	55.80%	76.30%	**80.11%**
NER	F1-macro	75.49%	79.79%	**79.81%**
	F1-micro	84.45%	**87.43%**	87.39%
SRL	F1-macro	41.51%	**43.83%**	43.18%
	F1-micro	51.86%	**54.39%**	53.21%
NSMC	Accuracy	86.25%	**87.69%**	87.62%
WSD	Sejong	93.70%	95.29%	**95.46%**
	Modu	90.26%	**93.75%**	93.68%

[표 14]은 다중-핫 표현 모델이 손실 함수에 따라 실험 영역의 평가 결과이다. 먼저 실험의 결과에서 바이너리 크로스 엔트로피는 나머지 두 손실 함수보다 모든 평가 영역에서 낮은 성능을 기록했다. 기계 독해 영역에서 F1-Score 기준 24.1% 포인트 차이를 보여 가장 격차가 크게 나타났으며, 나머지 영역에서도 1%~3% 포인트 차이를 보였다. 이러한 결과를 통해 바이너리 크로스 엔트로피의 경우 다중-핫 표현을 사용하는 언어 모델에 적합하지 않다고 볼 수 있다. 반면 목표 확률을 조정한 소프트맥스 크로스 엔트로피 손실 함수는 기존의 방법에 비해 기계 독해 영역에서 3.8% 포인트 앞섰다. 나머지 영역에 대해서도 0.2% 포인트 내의 차이를 보이거나 앞서는 결과를 보였다.

소프트맥스 크로스 엔트로피 기반의 두 모델의 성능 평가에서 성능의 격차가 가장 큰 기계 독해 영역에 대해 정답의 형태에 따른 성능 여부 차이를 분석했다. 정답으로 구성된 형태소가 음절 토큰의 사용 여부에 따른 두 손실 함수의 차이점을 분석했다. 예를 들어 '한국/NNP'라는 형태소가 KorQuAD v1.0에서 정답이면서 하나의 토큰으로 표현이 가능한 경우와 '워싱턴/NNP'처럼 '@워', '@싱', '@턴'과 같이 3개의 음절 토큰 집합으로 표현된 경우에 대해서 두 손실 함수의 성능을 비교했다.

[표 15] 손실 함수에 따른 KorQuAD v1.0에서 정답의 토큰 구성에 따른 성능

정답 종류 \ 모델			Softmax-Cross Entropy	Softmax-Cross Entropy (multi-hot base)
음절 토큰 미사용	1,469개	Exact	66.37%	**70.04%**
		F1-Score	72.20%	**76.18%**
음절 토큰 사용	4,305개	Exact	70.98%	**74.84%**
		F1-Score	77.70%	**81.45%**

[표 15]은 손실 함수와 정답 토큰의 구성에 따른 성능의 차이를 보여준다. 목표 확률을 변경하는 소프트맥스 크로스 엔트로피 방식이 음절 토큰을 사용하지 않은 정답에 대해서는 3.98% 포인트 차이를 보였다. 이는 지문에 나타난 다양한 형태소들도 음절 토큰의 집합으로 표현될 수 있으며, 이러한 지문을 이해하는 능력이 기존의 소프트맥스 크로스 엔트로피를 사용해 학습한 언어 모델보다 본 논문에서 제안한 손실 함수 방식이 나음을 알 수 있다. 또한 음절 토큰으로 분리되는 영역에서도 3.75% 포인트 앞섰다. 정답의 영역에 대한 평가 결과를 통해 다중-핫 표현 방식에서 음절 토큰 집합의 형태로 표현되는 형태소의 의미를 이해와 문맥의 이해에 본 논문이 제안한 손실 함수가 더 적합함을 보였다.

[표 14], [표 15] 실험 결과를 통해 다중-핫 표현 방식을 사용한 언어 모델의 학습에는 정답의 수에 따른 목표 확률을 조절하는 소프트맥스 크로스 엔트로피 방식이 적합함을 알 수 있다.

5.2.3 언어 모델을 위한 다중-핫 표현 방식의 성능 실험

앞의 실험을 통해 다중-핫 표현 방식에 어울리는 손실 함수를 선정할 수 있었다. 실험 결과를 바탕으로 정답의 수에 따른 목표 확률을 조절하는 손실 함수를 사용한 다중-핫 표현이 기존 언어 모델과 어떠한 성능의 차이를 가져오는지 평가했다. 평가를 통해 다중-핫 표현 방식이 언어 모델에 적합한지를 평가했다. 실험을 위해 비교 모델로 BPE 기반으로 구축한 37,027개의 토큰을 사용하는 BERT 모델을 선정했으며, 2가지의 크기(small, base)에서 평가를 진행했다. 평가를 위해 기계 독해 영역, 문장 분석, 감성 분석 영역에서 진행했다. 의미 분별 영역은 의미 분별의 대상이 되는 형태소가 기존의 BERT에서는 여러 토큰으로 분리되는 경우가 발생하며, 이 경우 의미 분별의 대상을 선정하는 문제가

발생하기 때문에 평가에서 제외했다. 평가를 위해 각 실험 영역에 대해 모두 동일한 설정을 사용했다. 먼저 기계 독해 영역에서 다중-핫 표현 방식이 가져오는 영향력을 비교했다.

[표 16] 다중-핫 표현 방식과 기존 토큰 분리 방식의 성능 실험

Token 종류 (크기)	KorQuAD v1.0	NER	SRL	NSMC
	F1	F1	F1	Accuracy
BPE base (small)	**81.81%**	86.78%	**54.35%**	86.99%
multi-hot base BERT (small)	80.11%	**87.39%**	53.21%	**87.62%**
BPE (base)	**89.17%**	88.02%	57.12%	87.95%
multi-hot base BERT (base)	87.97%	**89.05%**	**58.64%**	**88.91%**

[표 16]은 기존의 토큰 분리 방식과 다중-핫 표현 방식을 사용한 모델들에 대한 각 영역에서의 실험 결과이다. 실험 결과에서 기계 독해 영역을 제외한 나머지 영역에서는 모델이 커질수록 다중-핫 표현 방식을 사용한 언어 모델의 성능이 높게 나타났다. 형태소를 표현하는 방법에 따른 각 모델들의 실험 결과를 세밀하게 분석하기 위해 각 실험 결과를 세밀히 분석했다. 먼저 기계 독해 영역에 대해 base 크기의 모델에서 정답의 형태에 따른 실험 결과를 분석했다. 정답의 형태로 사용한 것은 정답이 둘 이상의 복수개의 형태소로 되어 있는 경우와 단일인 경우를 구분했으며, 형태소가 기존의 토큰 방식에 의해 여러 토큰으로 분리되는 경우와 그렇지 않은 경우를 비교했다.

[표 17] 기계 독해 영역에서의 정답의 형태에 따른 성능 비교

정답의 형태		BERT	multi-hot base BERT
형태소 분리	단일(1,811개)	**88.00%**	86.55%
	복수(1,753개)	**89.44%**	89.12%
형태소 미분리	단일(1,345개)	**89.31%**	88.43%
	복수(865개)	**90.86%**	87.90%

[표 17]의 결과를 통해 기계 학습에서는 정답의 형태와 상관없이 기존의 토큰 표현 방식이 높은 결과를 얻었다. 정답을 구성하는 형태소의 수에 상관없이 형태소가 여러 토큰

으로 분리가 일어나는 경우 두 모델의 차이가 작게 발생했다. 이는 다중-핫 표현 방식만 사용한 경우 여러 토큰으로 분리되는 형태소에 대한 이해력이 높음을 알 수 있었다. 하지만, 다중-핫 표현 방식의 경우 음절 단위로 형태소를 표현하기 때문에 해당 형태소에 대한 정보의 양이 부족하여 기존 토큰 표현 방식보다 낮은 결과를 보였다. 다음의 표는 실제로 정답의 차이가 나는 예시를 보여준다.

[표 18] 토큰 방식에 따른 기계 독해 영역의 질의문에 대한 정답의 분석

질의	정답	BERT	multi-hot base BERT
미국에서 실적주의로 전환 개혁된 제도는?	인사 제도	인사 제도	펜틀던 법
7음 가운데 블루노트가 뜻하는 반음 내려가는 음은?	미와 시	미와 시	블루 노트
알아사드가 수니파와 쿠르드인에 대한 유화책으로 교사들에게 착용을 허가한 것은?	니카브	니카브	다시 니카브
윤정훈이 총신대학교 목회신학전문대학원에서 공부한 전공은?	구약학	설교학과	구약학

[표 18]은 실제 base 크기의 모델의 실험에서 질의에 대한 각 모델들의 정답을 보여준다. 앞의 두 문제는 여러 토큰으로 분리되지 않고 정답이 두 개 이상의 형태소로 표현된 경우이다. 이 경우 다중-핫 표현 방식의 모델에서는 문맥에 대한 해석과 해당 단어에 대한 정보가 부족해 정답을 유추하지 못했다. 하지만, 아래의 두 문제에서는 형태소가 여러 토큰으로 분리되는 문제로 '구약학'은 기존 토큰 방식에서는 '구', '약', '학' 3개의 글자로 분리되어 기존 모델이 해당 단어의 이해가 낮아 정답을 유추하지 못했지만, 다중-핫 표현 방식의 모델에서는 정답을 유추했다. 이러한 결과를 통해 다중-핫 표현 방식이 성능은 기존 모델보다 낮지만, 형태소가 완전히 분리되는 경우 문맥을 통해 단어의 이해에 장점을 가지는 것으로 볼 수 있다.

다음은 기계 독해 영역과 달리 다중-핫 표현 방식이 우수한 성능을 보인 개체명 영역에 대해 실험 결과를 비교했다. 개체명 영역에서는 small 크기의 모델에서 0.61% 포인트 차이를 보였으며, base 크기의 모델에서는 1.03% 포인트 차이를 보였다. 이러한 차이는 개체명 인식에서는 형태소의 유지가 중요하기 때문에 기존의 토큰 방식보다 다중-핫 표

현을 통한 형태소의 유지가 중요한 요소로 반영이 되었기 때문이다. 다음은 실제로 base 크기의 모델에서 개체명 인식을 진행한 결과이다.

[표 19] 토큰 방식에 의한 각 모델들의 개체명 인식 분석

모델	개체명 인식
원문	[미국__03/NNP]:LC [뉴욕시/NNP]:LC+가/JKS [보행자/NNG]:CV+와/JC [자전거/NNG]:AF [운전자/NNG]:CV ...
BERT	[미국/NNP]:LC [뉴욕 시/NNP]:OG 가/JKS [보행자/NNG]:CV 와/JC [자전거/NNG]:AF [운전자/NNG]:CV ...
multi-hot base BERT	(미국):LC ([@뉴 @욕 @시]):LC ##가 (보행자):CV ##와 (자전거):AF (운전자):CV ...

[표 19]는 토큰 표현 방식에 따라 개체명 인식 영역에서의 결과를 보여준다. 예시 문장에 대해 형태소의 분리가 일어나지 않은 단어들에 대해 두 방식 모두 정답을 맞혔다. 하지만, 원문에서 '뉴욕시/NNP'라는 형태소는 기존의 토큰 방식을 사용한 경우 '뉴욕'과 '시/NNP'로 분리되며, '뉴욕'이 학습한 이전의 정보에 의해 조직명(organization)로 개체명 태그가 부착됐다. 하지만, 다중-핫 표현 방식을 사용한 경우 '뉴욕시/NNP'라는 형태소가 유지되어 주변 문맥에 의해 지명(location)로 부착되었다. 이러한 예시를 통해 다중-핫 표현 방식을 사용한 경우 개체명 인식에서 형태소의 의미를 온전히 사용할 수 있음을 보였으며, 주변 문맥의 정보를 그대로 습득할 수 있음을 보였다.

다음으로 의미역 영역에서는 small 크기의 모델에서는 기존의 언어 모델이 1.14% 포인트 앞서 나갔다. 하지만, base 크기의 모델에서는 다중-핫 표현 방식이 1.52% 포인트 높은 결과를 보였다. 다음은 실제로 base 크기의 모델에서의 결과를 비교한 표이다.

[표 20] 토큰 방식에 의한 각 모델들의 의미역 인식 분석

모델	대상 용언	의미역 인식
원문	돌/VV	[필러__88/NNG+·/SP+보톡스/NNG 중독설/NNG]:ARG0 돌/VV+았/EP+지만/EC+…/SE 특검/NNG+이/JKS 확인하/VV+ㄴ/ETM 거__01/NNB+ㄴ/JX 3/SN+년__02/NNB+간__16/XSN 8/SN+차례__01/NNG
BERT	돌/VV	[필러/NNG ·/SP 보톡스/NNG 중독]:ARG0 [설/NNG]:ARG1 돌/VV 았/EP 지만/EC …/SE 특검/NNG 이/JKS 확인하/VV ㄴ/ETM 거/NNB ㄴ/JX 3/SN 년/NNB 간/XSN 8/SN 차례/NNG
multi-hot base BERT	돌/VV	(필러 · 보톡스 [@중 @독 @설]):ARG0 [돌##] ##았 ##지만 … 특검 ##이 [확인 하##] ##ㄴ 거 ##ㄴ [3] 년 [@간] [8] 차례 [SEP]

[표 20]은 의미역 영역에서 토큰의 표현 방식에 따른 결과의 차이를 보여주고 있다. 개체명 인식과 마찬가지로 토큰의 분리로 인해 기존의 모델에서는 '설/NNG'라는 부분이 다른 태그로 표현되었지만, 다중-핫 표현 방식에는 해당 단어가 하나로 표현되어 동일한 의미역 태그로 결과가 나타났다. 의미역 영역에서의 실험 결과를 통해 언어 모델의 파라미터가 늘어나서 담을 수 있는 정보가 많을수록 다중-핫 표현 방식이 기존 방식에 비해 문장에 대한 관계 분석을 잘함을 알 수 있었으며, 개체명 인식과 마찬가지로 다중-핫 표현 방식을 통해 형태소 단위를 유지하면, 분별의 대상을 줄일 수 있으며, 오류를 줄일 수 있었다.

마지막으로 감성 분석 영역에서의 평가를 통해 small 크기의 모델에서는 0.63% 포인트 차이를 보였으며, base 크기에서는 0.86% 포인트 차이를 보였다. 토큰 방식에 따른 성능의 차이에 대한 원인을 파악하기 위해 실제 예제에 대한 토큰 변환 결과와 감성 분석 결과를 비교했다.

[표 21] 토큰 표현 방식에 따른 감성 분석 문장의 분리와 결과

모델	토큰 표현	정답
원문	안보__02/NNG+면__05/NNG 후회__01/NNG+ㅠㅠ/NNG+./SF+./SF+./SF	긍정
BERT	안보/NNG 면/NNG 후회/NNG [UNK] ./SF ./SF ./SF	부정
multi-hot base BERT	안보 면 후회 [@ㅠ @ㅠ] . . .	긍정
원문	할일없/VA+고/EC+,/SP 지루하__01/VA+ㄴ/ETM+거__01/NNB 좋아하/VV+는/ETM+사람/NNG+한테/JKB 강추/NNG	부정
BERT	할 일 없/VA 고/EC ,/SP 지루하/VA ㄴ/ETM 거/NNB 좋아하/VV 는/ETM 사람/NNG 한테/JKB 강 추/NNG	긍정
multi-hot base BERT	[할 일 없##] ##고 , [지루 하##] ##ㄴ 거 [좋아 하##] ##는 사람 ##한테 [@강 @추]	부정

[표 21]은 실제로 감성 분석 영역에서 토큰 표현 방식에 따른 토큰 표현 결과와 감성 영역의 결과를 보여준다. 첫 번째 문장에서 사용된 'ㅠㅠ'라는 이모티콘이 토큰으로 표현할 수 없어 기존의 방식으로는 미등록 토큰으로 표현되었다. 그러나 해당 이모티콘은 감성을 판단하는 중요한 정보로 미등록 토큰으로 표현되었기 때문에 기존의 BERT 모델은 해당 문장에 대해 잘못된 분석이 발생했다. 반면, 다중-핫 표현을 사용할 경우 이러한 표현까지 토큰으로 표현되어 감성 분석에 활용되는 결과를 보였다. 또한 두 번째 예제에서는 감성 분석에 중요한 형태소인 '할일없/VA'와 신조어인 '강추/NNG'가 여러 토큰으로 분리되어 원래 형태소와 다른 의미로 해석하여 정답을 맞히지 못했다. 하지만, 다중-핫 표현을 사용한 모델에서는 맞추는 모습을 확인했다. 이를 통해 다중-핫 표현을 사용하는 경우 토큰으로 표현하는 가짓수가 많기 때문에 신조어나, 'ㅠㅠ'와 같은 감정 표현 텍스트를 토큰으로 표현할 수 있었으며, 이 정보가 감성 분석 영역에서 중요한 역할을 할 수 있음을 확인했다.

다양한 영역에서 기존의 토큰 표현 방식과 다중-핫 표현 방식의 비교 실험을 통해 다중-핫 표현 방식이 비록 기계 독해 영역을 제외한 나머지 영역에서 형태소 유지로 인해 좋은 성능을 보였다. 이를 통해 다중-핫 표현 방식을 언어 모델에 적용하게 되면, 주변 문맥 정보를 형태소에 고스란히 전달되며, 분별의 대상 수를 줄이는 효과를 통해 성능의 향상을 가져온다. 그리고 더 많은 단어들을 적은 토큰 수로 표현 가능하기 때문에 신조

어와 중요 표현에 대해 형태소 단위로 해석이 가능한 장점을 가짐을 확인했다. 하지만, 형태소가 가진 다양한 의미나, 표현들을 학습하지 못하는 문제점을 확인할 수 있었다.

5.3 언어 지식과 문맥 정보가 결합된 자연어 표현 모델 실험

5.3.1 언어 지식 정보가 결합된 자연어 표현 모델 실험

언어 지식을 융합한 UKnowBERT(Ulsan Knowledge ensemble BERT)의 성능 평가를 위해 기존 모델과의 4가지 실험 영역에 대해 비교 실험을 진행했다. UKnowBERT 모델은 언어 지식을 쉽게 활용하기 위해 5.2.3의 실험에서 증명한 다중-핫 표현 방식을 모델에 사용했다. 다양한 언어 지식과 기존의 문맥 정보를 사전 학습(pre-training)한 UKnowBERT과 기존의 BERT와의 차이점을 분석했다. 실험을 위해 5.2에서 정의한 2가지 크기(small, base)의 언어 모델을 사용했으며, 학습에 사용한 언어 지식 정보의 조합에 따라 총 8개의 모델을 구성했다.

[표 22] 언어 지식 정보를 활용한 자연어 표현 모델의 실험에 사용한 모델의 정보

모델이름	토큰 방식	Transformer layer	Hidden layer	사전 학습에 사용한 데이터 종류*
BERT-S**	one-hot	4	256	①②
mBERT-m	multi-hot	4	256	①
mBERT-S	multi-hot	4	256	①②
UKnowBERT-r	multi-hot	4	256	①②⑤
UKnowBERT-w	multi-hot	4	256	①②③④
UKnowBERT-S	multi-hot	4	256	①②③④⑤
BERT-B**	one-hot	12	768	①②
mBERT-B	multi-hot	12	768	①②
UKnowBERT-B	multi-hot	12	768	①②③④⑤

* ①: Masked LM, ②: Next Sentence LM, ③: hypernym LM, ④: USenseVector LM, ⑤: Verb relation LM

** 기존의 BERT의 학습 방식을 그대로 사용한 모델

[표 22]는 실험에서 사용한 각 언어 모델의 크기와 사용한 학습 데이터의 종류를 보여주고 있다. 먼저 BERT-S과 BERT-B는 BERT 방식을 그대로 사용하여 사전 학습을 실시한

모델이다. mBERT-m는 다중-핫 표현 기반으로 마스킹 학습만을 사용하여 사전 학습한 모델이다. mBERT-S와 mBERT-B의 경우 다중-핫 표현 방식을 기반으로 마스킹 학습과 다음 문장 예측 학습을 사용한 모델이다. UKnowBERT-r은 BERT에서 사용한 두 가지 학습과 용언의 관계 정보만을 추가하여 학습한 모델이며, UKnowBERT-w는 용언의 관계 정보 대신 명사의 상위어 정보와 뜻풀이 정보를 학습한 모델이다. 마지막으로 UKnowBERT-S와 UKnowBERT-B는 기존에 사용한 학습정보와 언어 지식에서 추출한 학습 데이터를 모두 사용하여 학습한 모델이다. 각각의 모델이 사용한 세부 설정은 [표 13]와 대부분 동일하지만, base 크기의 모델인 BERT-B, mBERT-B, UKnowBERT-B에 대해서는 사전 학습을 250,000 스텝만 진행했다. 또한 모든 모델의 최대 토큰 길이를 256으로 고정했다.

5.3.2 언어 지식을 활용한 UKnowBERT 모델과의 성능 비교

다양한 언어 지식을 융합한 언어 모델에 어떠한 영향을 가져오는지 확인하기 위해 5.1.3에서 정의한 4개의 영역에서 성능 평가를 진행했다. 모든 모델은 각 실험영역에 대해 동일한 설정을 사용했다. 먼저 small 크기를 가지는 모델들에 대해 평가했다. BERT의 방식과 다중-핫 표현을 사용하지 않은 BERT-S은 토큰 분리로 인해 형태소 단위를 유지할 수 없기 때문에 의미 분별 영역에 대한 실험을 진행하지 않았다.

[표 23] 언어 지식을 사용한 언어 모델들의 성능 비교 실험(small)

모델	KorQuAD v1.0 (F1)	NER (F1)	SRL (F1)	NSMC (Accuracy)	WSD(Accuracy)	
					Sejong	Modu
BERT-S	**81.23%**	86.54%	53.78%	87.19%	x	x
mBERT-m	65.84%	**87.24%**	53.59%	**87.89%**	**97.50%**	**93.94%**
mBERT-S	78.89%	87.20%	53.79%	87.71%	97.45%	93.73%
UKnowBERT-r	80.28%	87.08%	53.93%	87.77%	97.44%	93.55%
UKnowBERT-w	77.23%	87.17%	53.69%	87.80%	97.45%	93.80%
UKnowBERT-S	79.55%	**87.24%**	**54.37%**	87.64%	97.47%	93.55%

[표 23]의 실험의 결과에서 가장 격차가 크게 나타난 기계 독해 영역에서 BERT-S가 81.23%로 가장 높았다. 다중-핫 표현 방식을 사용한 나머지 5개의 모델 중에서는 용언의 의미 제약 정보를 통한 용언과 명사의 관계를 학습한 모델들이 높은 성능을 차지했다.

그리고 기존의 학습 방식과 다중-핫 표현을 사용한 mBERT-S가 뒤를 이었다. 기계 독해 영역에서 각 모델의 평가 결과를 통해 문맥의 이해와 독해 능력의 향상을 위해서는 용언의 관계성을 학습하는 것이 중요하다고 볼 수 있다. 그리고 단순히 단어의 의미만 추가하여 학습한 UKnowBERT-w와 UKnowBERT-S의 결과 비교를 통해 단편적인 언어 지식을 사용하는 것보다 다양한 언어 지식을 사용하는 것이 상호 보완하는 역할을 하는 것을 보였다. 다음은 실제로 기계 독해 영역에서 각 모델들이 정답을 찾은 결과를 비교해 보았다.

[표 24] 언어 지식의 활용에 따른 각 모델들의 기계 독해 출력 비교

지문	… 대표적으로 대중음악에선 해롤드 알렌의 〈Blues in the Night〉, 블루스 발라드에선 〈Since I Fell for You〉, 〈Please Send Me Someone to Love〉 등 심지어 조지 거시윈의 〈Rhapsody in Blue〉, 〈Concerto in F〉와 같은 오케스트라에게도 영향을 끼쳤다. … 블루스의 기본틀은 배트맨의 테마곡이고 당시 10대 아이돌이었던 파비안 포르테가 부른 히트곡 〈Turn Me Loose〉, 컨트리 음악의 신 지미 로저스, 그리고 기타리스트이자 보컬리스트인 트레이시 채프먼의 히트곡 〈Give Me One Reason〉에도 사용되었다.
질의	blues in the night을 부른 가수가 누구인가요?
답변 (모델)	해롤드 알렌 (BERT-S, mBERT-S, UKnowBERT-r, UKnowBERT-S) (정답)
	파비안 포르테 (mBERT-m, UKnowBERT-w)
지문	이틀 후, 알아사드는 국영 TV를 통해 방영된 대국민연설에서 국가비상사태의 해제를 약속하였다. 4월 19일, 정부는 국가비상사태를 해체하는 의안을 승인했으며, 국가비상사태는 선포된 지 43년 만에 해제되었다. …
질의	알아사드는 국영 TV를 통해 무엇을 약속했는가?
답변 (모델)	국가비상사태의 해제 (BERT-S, mBERT-S, UKnowBERT-r, UKnowBERT-S) (정답)
	국가비상사태 (mBERT-m, UKnowBERT-w)

[표 24]는 언어 지식을 사용하는 것이 기계 독해에 어떠한 영향력을 가져오는 확인하기 위해 실험의 결과의 예시를 보여준다. 언어 지식을 사용하지 않고 기존의 모든 정보를 사용한 BERT-S와 mBERT-S의 경우 두 예시에서 모두 정답을 맞혔다. 그리고 언어 지

식을 사용한 경우 용언과 명사의 관계 정보를 사용한 UKnowBERT-r과 UKnowBERT-S가 정답을 유추했다. 하지만, 마스킹 학습만 진행한 mBERT-m과 단어의 의미 정보를 추가한 UKnowBERT-w의 경우 문맥을 올바로 이해하지 못해 정답과 다른 단어가 사용되거나, 일부만 출력되었다. 이러한 결과를 통해 기계 독해 영역에서 언어가 가진 관계성을 학습하는 것이 중요하며, 언어 지식을 모두 활용해야 단어의 의미만으로 치우쳐진 모델과 달리 문맥을 유추하는 것이 가능했다.

기계 독해 영역과 달리 개체명 인식 영역의 경우 BERT-S가 가장 낮은 성능을 보였으며, 용언의 관계성만 추가로 학습한 UKnowBERT-r의 경우 다중-핫 표현 방식을 사용한 5개의 모델 중에 가장 낮은 성능을 보였다. 하지만, 모든 정보를 학습한 UKnowBERT-S 모델이 가장 높은 성능을 보여줌에 따라. 용언의 관계성 정보로 인해 손실되는 문맥의 이해가 어휘가 가진 관계나 뜻풀이 정보에 의해 보정됨을 알 수 있었다.

[표 25] 언어 지식을 학습한 언어 모델이 개체명 인식 영역에 미치는 영향

모델	개체명 인식
원문	소송__01/NNG+은/JX [지에스칼텍스/NNP 사건__01/NNG]:EV+이/JKS 사실상/NNG 유일하/VA+다/EF+./SF
BERT-S	소송/NNG 은/JX 지 [에스]:OG [칼텍스/NNP]:EV [사건/NNG]:OG 이/JKS 사실상/NNG 유일하/VA 다/EF ./SF
mBERT-m	소송 ##은 ([@지 @에 @스 @칼 @텍 @스] 사건):EV ##이 사실상 [유일 하##] ##다 .
mBERT-S	소송 ##은 ([@지 @에 @스 @칼 @텍 @스] 사건):EV ##이 사실상 [유일 하##] ##다 .
UKnowBERT-r	소송 ##은 ([@지 @에 @스 @칼 @텍 @스]):OG 사건 ##이 사실상 [유일 하##] ##다 .
UKnowBERT-w	소송 ##은 ([@지 @에 @스 @칼 @텍 @스] 사건):EV ##이 사실상 [유일 하##] ##다 .
UKnowBERT-S	소송 ##은 ([@지 @에 @스 @칼 @텍 @스] 사건):EV ##이 사실상 [유일 하##] ##다 .

[표 25]는 다양한 언어 모델들이 개체명 인식에서의 예시에 대한 결과를 보여준다. 가장 낮은 성능을 보였던 BERT-S의 경우 기존의 토큰 표현 방식의 문제점이 그대로 적용되어 '지에스칼텍스'라는 단어가 여러 토큰으로 표현되었기 때문에 올바른 개체명 태그를

부착하지 못했다. 그리고 용언의 의미제약을 사용한 UKnowBERT-r의 경우 '사건'이라는 단어의 의미를 부족하게 가지고 있어 나머지 모델과 달리 조직명으로 부착되는 오류가 발생했다. 이를 통해 언어 지식 중에 관계 정보와 단어에 대한 의미 정보를 함께 사용해야 조화를 이루어 다양한 영역에 대한 문제 해결 능력을 높일 수 있다.

의미역 인식 영역에서는 기계 독해 영역과 동일한 결과를 보였다. 문맥의 관계성을 학습하지 하나도 학습하지 않은 mBERT-S가 가장 낮은 성능을 보였다. 모든 정보를 학습한 UKnowBERT-S가 가장 높은 성능을 나타내는 것으로 보아 언어 지식을 편향된 정보를 사용하는 것보다 다양한 정보를 언어 모델이 습득할 수 있는 것이 중요하다고 볼 수 있다.

[표 26] 언어 지식을 학습한 언어 모델이 의미역 인식 영역에 미치는 영향

모델	대상 용언	의미역 인식
원문	줄/VV	(부정적/NNG+이/VCP+ㄴ/ETM 표현/NNG):ARG0+도/JX 줄/VV+었/EP+다/EF+./SF
BERT-S	줄/VV	(부정적/NNG 이/VCP ㄴ/ETM 표현/NNG):ARG1 도/JX 줄/VV 었/EP 다/EF ./SF
mBERT-m	줄/VV	(부정적 이## ##ㄴ 표현):ARG1 ##도 줄## ##었 ##다 .
mBERT-S	줄/VV	(부정적 이## ##ㄴ 표현):ARG1 ##도 줄## ##었 ##다 .
UKnowBERT-r	줄/VV	(부정적 이## ##ㄴ 표현):ARG0 ##도 줄## ##었 ##다 .
UKnowBERT-w	줄/VV	(부정적 이## ##ㄴ 표현):ARG1 ##도 줄## ##었 ##다 .
UKnowBERT-S	줄/VV	(부정적 이## ##ㄴ 표현):ARG0 ##도 줄## ##었 ##다 .

[표 26]은 의미역 영역에 대한 실험 결과 예제이다. 용언 '줄다'에 대해 의미역을 부착한 결과로 언어 지식 중에 용언의 관계성을 학습한 UKnowBERT-r과 UKnowBERT-S만 ARG0(서술어의 동작주, 행위자, 경험주) 태그로 부착되었으며, 나머지 모델은 ARG1(서술어의 피동작주, 대상)으로 부착되었다. 이러한 점을 통해 용언의 의미제약을 통한 용언과 명사의 관계정보가 의미역 인식에 효과를 보이며, 단어의 의미만 추가 학습한 UKnowBERT-w의 경우 오답이 결과로 나타났지만, 모든 정보를 사용할 경우 이러한 오류를 줄일 수 있음을 확인했다.

마지막으로 감성 분석과 의미 분별의 경우 마스킹 기법만 사용한 mBERT-m이 가장 높게 나타났다. 감성 분석과 의미 분별의 결과를 통해 언어 지식이 해당 영역의 분별에 크게 관여하지 않았다. 하지만, 토큰 표현 방식에 많은 영향을 받았음을 확인할 수 있었다.

기존의 토큰 표현 방식을 사용한 BERT-S는 다중-핫 표현 방식을 사용한 모델들 보다 감성 분석에서 낮은 성능을 기록했다. 이를 통해 다중-핫 표현 방식을 통해 성격이 다른 말뭉치에 등장한 형태소를 토큰으로 변환하는 것이 감성 분석에 도움이 됨을 확인했다.

small 크기에서의 실험 결과를 통해 용언의 관계 정보가 문맥이나, 문장에서 단어 간의 관계성에 대해 이해가 필요한 기계 독해나 의미역 영역에서 효과적임을 보였으며, 단어의 의미를 사용한 경우 개체명과 감성분석 영역에서 단어의 의미를 보조하기 때문에 좋은 역할을 했다. 하지만 각각의 정보만 쓰일 경우 해당 정보로 편향이 일어나 나머지 영역에서는 낮은 성능을 보였다. 이러한 실험 결과를 통해 본 논문에서 제안한 UKnowBERT 모델과 같이 다양한 언어 지식을 사용하여 언어 모델을 사전 학습하는 것이 특정 데이터에 편향되는 현상을 막아주는 효과를 가져왔다. 다음은 base 크기의 언어 모델들에 대해 [표 23]의 결과와 유사한지를 파악하기 위해 실험을 진행했다.

[표 27] 언어 지식을 사용한 언어 모델들의 성능 비교 실험(base)

모델	KorQuAD v1.0 (F1-Score)	NER (F1-Score)	SRL (F1-Score)	NSMC (Accuracy)	WSD(Accuracy)	
					Sejong	Modu
BERT-B	84.11%	88.45%	**58.77%**	86.98%	x	x
mBERT-B	83.89%	89.23%	58.03%	88.34%	97.72%	**95.81%**
UKnowBERT-B	**85.19%**	**89.30%**	57.86%	**88.36%**	**97.76%**	95.78%

[표 27]은 언어 지식 정보가 base 크기를 가지는 언어 모델에 가져오는 영향력을 확인하기 위해 실험한 결과이다. [표 23]의 결과와 달리 의미역 인식을 제외한 실험 영역에서 UKnowBERT-B가 최고 성능을 보였다. [표 23]의 실험 결과에서 가장 차이를 보였던 기계 독해 영역에서는 기존 모델인 BERT-B에 비해 1.08% 포인트 앞서는 차이를 보였다. 반면 small 크기의 언어 모델에서 앞서던 의미역 인식의 경우 0.91% 포인트 낮은 결과를 보였다. 감성 분석 영역에서는 small 크기의 결과와 마찬가지로 다중-핫 표현 방식을 사용한 두 모델이 높은 성능을 보였으며, 의미 분별 영역에서는 언어 지식이 성능에 큰 영향을 주지 못했다. 이러한 결과를 통해 언어 지식을 융합한 UKnowBERT의 경우 모델의 크기가 커져 많은 파라미터를 사용하는 경우 다양한 언어 지식을 더 많이 기억할 수 있으며, 이를 통해 다양한 응용 영역의 성능 향상을 가져왔다.

[표 23], [표 27]에서의 실험 결과를 통해 용언의 관계성 정보를 활용하는 경우 문맥의 이해에 도움을 주며, 상위어 정보나 뜻풀이 벡터는 어휘의 의미를 이해하는데 효과를 가

져왔다. 하지만 각각의 정보만 학습하는 경우 편향되는 현상이 발생해 용언의 관계 정보의 경우 개체명과 같은 영역에서 약점을 보였으며, 반대로 어휘의 의미 정보는 기계 독해와 같은 영역에서 낮은 성능을 보였다. 본 논문에서 제안한 UKnowBERT 모델의 경우 성격이 다른 두 정보를 모두 학습하기 때문에 특정 응용 영역에 편향되지 않고 모든 영역에서 고른 성능을 보였다. 그리고 다중-핫 표현 방식만 사용한 모델에서 부족한 정보를 언어 지식을 함께 학습함으로 인해 해결이 가능함을 확인했다.

5.3.3 제한된 환경에서 UKnowBERT 모델과 각 언어 모델의 성능 비교

5.3.2의 실험을 통해 언어 지식을 사용하게 되면 사전 학습 단계에서 형태소들이 다양한 표현과 의미를 언어 모델이 학습함을 확인했다. 실험에 사용한 각 영역은 충분한 학습 데이터를 사용해 미세 조정(fine-tuning)을 실시하였기 때문에 각 실험 영역을 언어 모델이 충분히 이해할 수 있었다. 그러나 응용 영역을 학습하기 위한 학습 데이터는 충분히 확보하기 힘들며, 새로운 데이터를 구축하기 위해서는 많은 시간과 비용이 발생한다. 그렇기 때문에 충분히 학습 데이터를 확보하지 못한 제한된 환경에 대해서 언어 모델의 성능도 중요하다. 제한된 환경에서 언어 지식을 융합한 UKnowBERT 모델이 기존 언어 모델과 어떠한 차이점을 가지는지 평가를 진행했다.

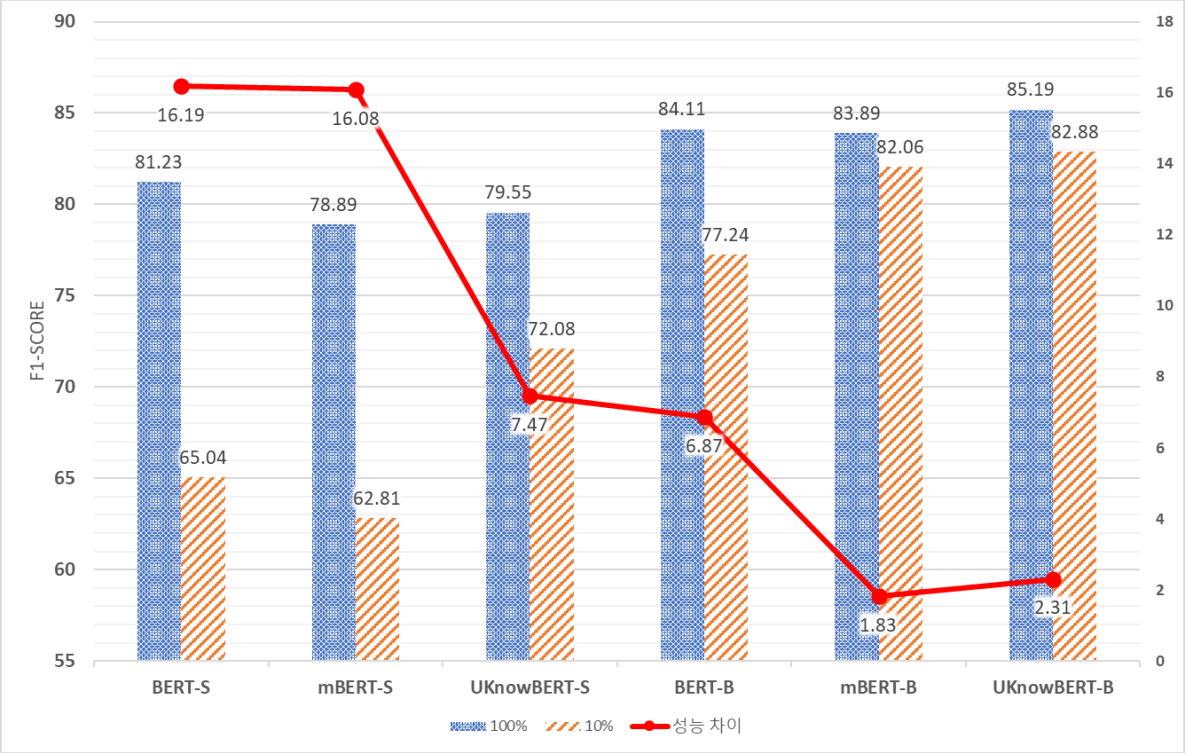
실험을 위해 사용한 언어 모델들은 기존의 BERT와 다중-핫 표현 방식을 사용한 BERT, 언어 지식을 융합한 UKnowBERT을 대상으로 선정했다. 두 가지 모델의 크기(small, base)에서 각 영역을 평가했으며, 제한된 환경을 구현하기 위해 각 평가 영역이 가진 학습 데이터의 10%만 사용하여 미세 조정(fine-tuning) 학습을 진행했다. 실험 평가에 사용하는 데이터의 크기는 동일하게 했다.

[표 28] 제한된 환경에서 언어 지식을 활용한 언어 모델의 성능 평가

모델	KorQuAD v1.0 (F1)	NER (F1)	SRL (F1)	NSMC (Accuracy)	WSD(Accuracy)	
					Sejong	Modu
BERT-S	65.04%	79.71%	45.59%	83.45%	x	x
mBERT-S	62.81%	**80.83%**	44.25%	84.08%	**94.41%**	**84.87%**
UKnowBERT-S	**72.08%**	80.60%	**45.95%**	**84.13%**	94.33%	83.99%
BERT-B	77.24%	83.20%	53.38%	79.63%	x	x
mBERT-B	82.06%	84.11%	52.78%	84.05%	95.84%	88.09%
UKnowBERT-B	**82.88%**	**84.26%**	**53.41%**	**84.29%**	**95.86%**	**88.67%**

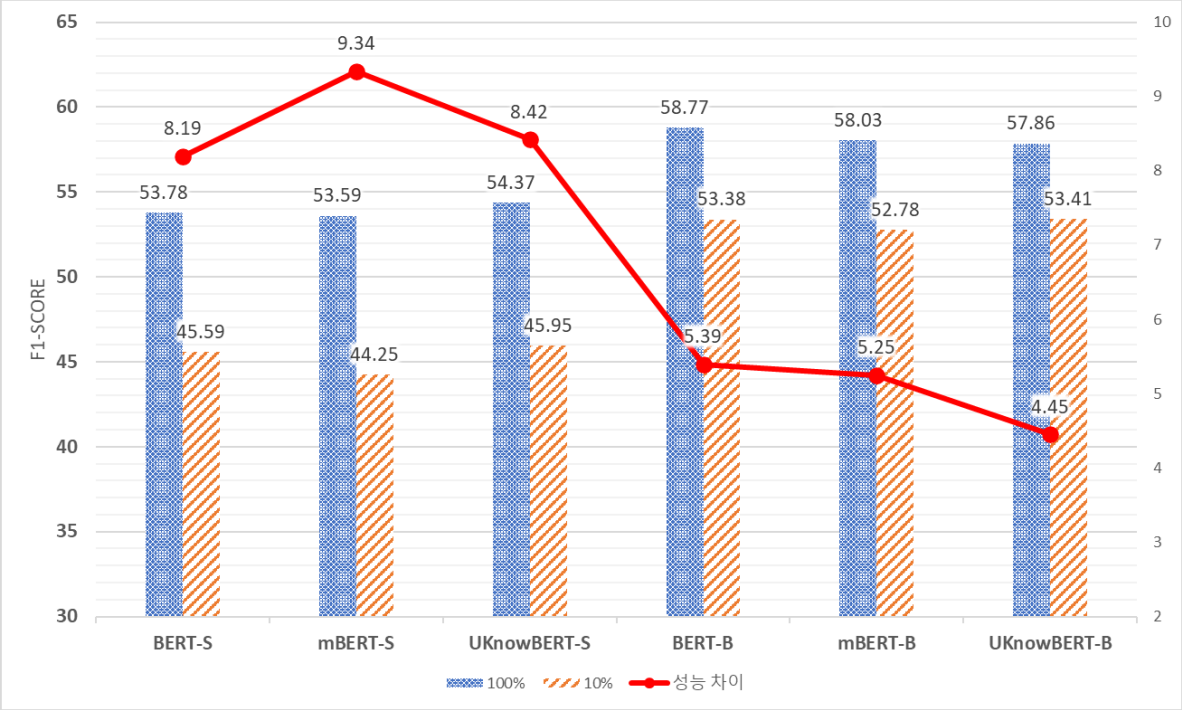
[표 28]은 제한된 환경에서 언어 지식을 융합한 언어 모델과 기존 모델의 성능을 평가한 결과이다. small 크기의 모델에서 개체명과 의미 분별 영역을 제외하고 모든 영역에서 UKnowBERT가 기존의 BERT와 mBERT 보다 높은 성능을 보였다. 다만 UKnowBERT는 기계 독해 영역과 의미역 영역에서 다중-핫 표현만 사용한 모델과 비교하여 0.5% 포인트 이상의 차이를 보였으며, 나머지 영역에서는 비슷한 성능 결과를 보였다.

[표 28]의 실험 결과를 통해 학습에 사용한 언어 지식이 제한된 환경에서 가져오는 영향력을 상세히 분석하기 기계 독해 영역과 의미역 영역에 대해 학습을 제한하지 않은 환경과 제한된 환경에서의 성능의 차이를 비교했다. 나머지 영역에 비해 난이도가 높은 영역들을 선택했다.



[그림 46] 기계 독해 영역에서 학습 환경에 따른 각 언어 모델의 성능

[그림 46]은 실험에 사용한 각 모델들이 학습 환경에 따른 기계 독해 영역에서의 성능 평가 결과와 성능 격차를 그래프로 표현했다. 실험 결과에서 기존의 BERT의 경우 UKnowBERT와 비교하여 small 크기의 모델에서는 2.16배 격차가 발생했으며, base 크기에서 2.97배 차이가 발생했다. 다중-핫 표현만 사용한 언어 모델의 경우 small 크기에서는 일반 BERT와 비슷한 성능 격차가 발생했지만, base 크기의 모델에서는 가장 작은 성능 격차를 보였다. UKnowBERT의 경우 모델의 크기와 상관없이 제한된 환경에서 모델의 크기와 상관없이 가장 높은 성능을 보였다. 실험을 통해 제한된 환경의 기계 독해 영역에서는 다양한 언어 지식 정보가 학습하지 못한 부분에 대해 보완을 해준다고 볼 수 있다.



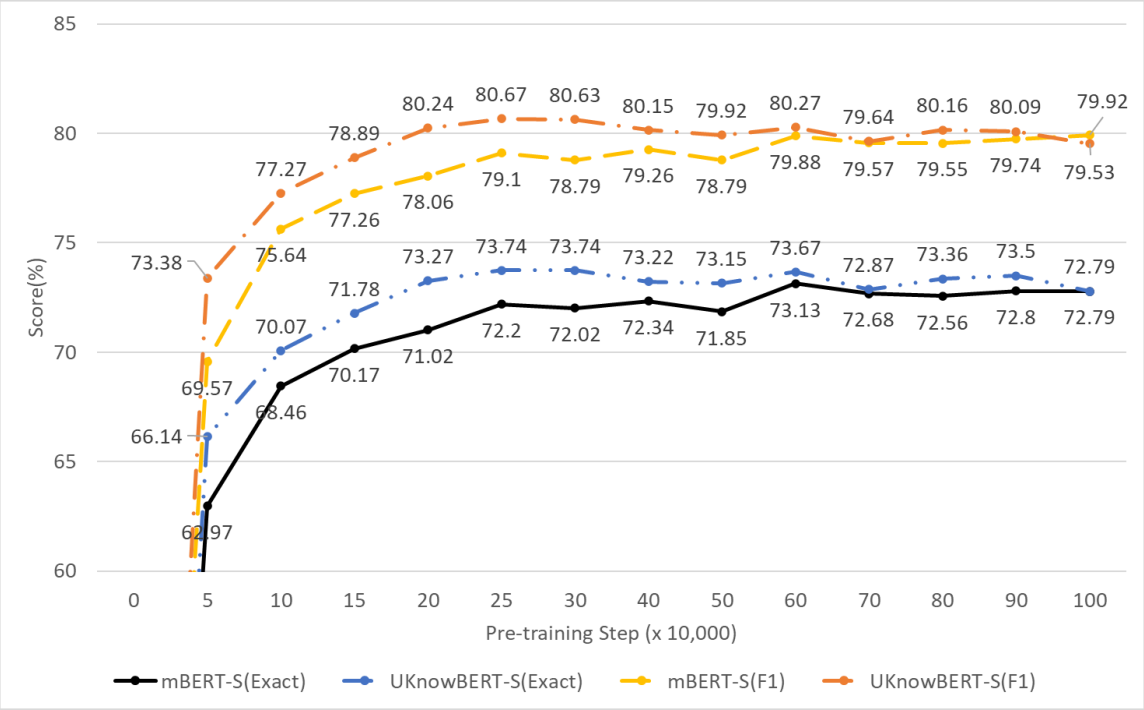
[그림 47] 의미역 인식에서 학습 환경에 따른 각 언어 모델의 성능

[그림 47]은 의미역 인식 영역에서 실험에 사용한 모델들이 미세 조정의 환경에 따른 성능과 그 차이를 비교했다. 앞서 기계 독해 영역과 달리 small 크기의 모델에서는 BERT-S의 성능 하락이 가장 낮게 나타났다. 반면 다중-핫 표현만 사용한 mBERT-S 모델은 UKnowBERT-S 비해 제한된 환경에서 성능의 하락이 0.88 포인트만큼 발생했다. base 크기에서는 UKnowBERT-B 모델이 가장 적은 성능 하락과 가장 높은 성능을 보였다. 반면 기존의 방식인 BERT-B의 경우 성능의 하락이 가장 크게 발생했다. 실험을 통해 언어 지식을 학습한 UKnowBERT-B가 의미역 영역에서도 제한된 환경에서 우수함이 확인했다.

제한된 환경에서의 실험 결과들을 통해 언어 지식을 융합한 UKnowBERT 모델이 응용 영역의 학습 데이터가 부족한 경우 언어 지식을 통해 필요한 정보를 획득할 수 있어 유사한 패턴이나 의미가 비슷한 단어에 대한 해석이 높아 충분한 데이터를 가진 환경에 비해 성능의 하락이 적어 말뭉치의 확장이 어려운 새로운 응용 영역에 대해 말뭉치 구축 비용을 절감할 수 있는 하나의 대안 언어 모델로 사용할 수 있었다.

5.3.4 사전 학습의 진행도에 따른 언어 지식의 영향력

언어 지식을 융합한 UKnowBERT 모델이 앞의 5.3.3의 실험을 통해 제한된 환경에서 기존 모델 대비 우수함을 확인했다. 사전 학습(pre-training) 단계에서 언어 지식이 가져오는 영향력을 확인하기 위한 실험을 진행했다. 실험은 다중-핫 표현 방식을 사용한 mBERT-S와 언어 지식을 사용한 UKnowBERT-S를 대상으로 진행했다. 실험을 위해 100만 스텝까지 사전 학습을 진행하는 과정 중에 5만 스텝 단위로 학습 중인 모델을 저장하여 기계 독해 영역에 대해 평가를 진행했다.

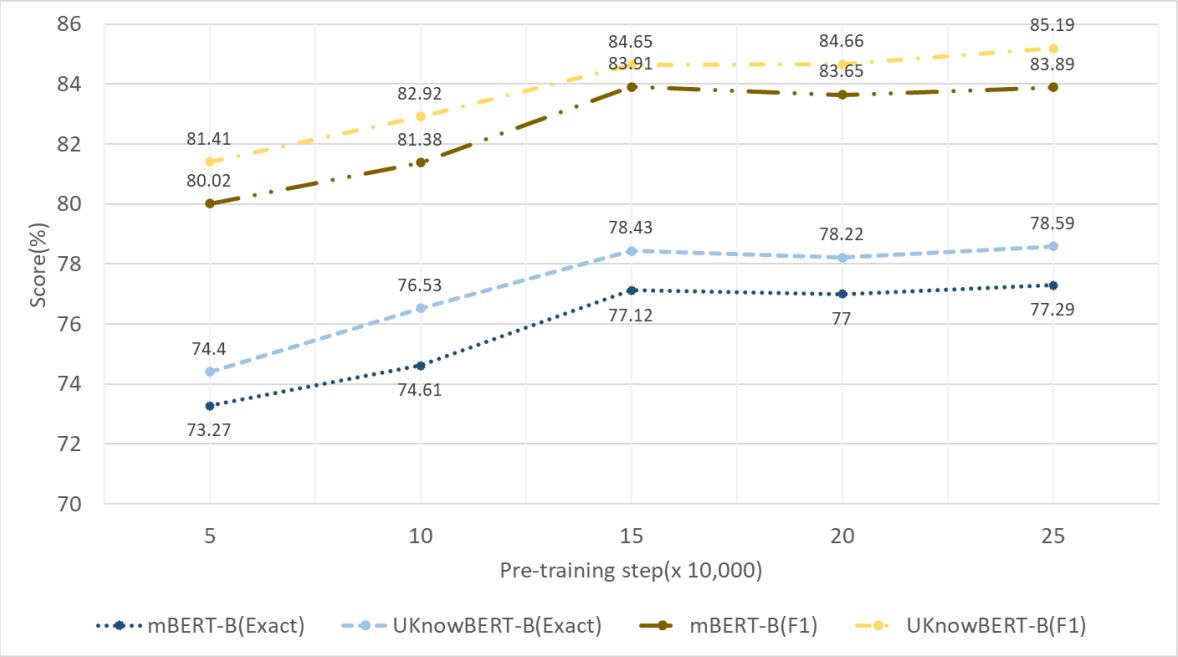


[그림 48] mBERT-S와 UKnowBERT-S의 각 사전 학습 진행 단계에 따른 KorQuAD v1.0의 결과 그래프

[그림 48]는 mBERT-S와 UKnowBERT-S에 대해 사전 학습의 진행 단계별 KorQuAD v1.0의 실험 결과를 그래프로 표현했다. 그래프는 0부터 300,000 스텝까지 50,000 단위로 표현했으며, 300,000 스텝 이후로는 100,000 단위로 표현했다. 그래프에서 초기 250,000 스텝까지 UKnowBERT-S가 mBERT-S에 비해 1% 포인트 이상의 성능을 유지했다. 이후 UKnowBERT-S는 300,000 스텝 이후 일정한 성능을 유지했다. 하지만, mBERT-S의 경우 600,000 스텝이 진행된 후에 일정하게 유지되었다. small 크기의 모델을 기준으로 언어 지식을 융합한 UKnowBERT-S가 더 빨리 일정 성능에 수렴을 하며, 성능이 더 높게 유지

되는 것을 확인할 수 있었다.

다음은 [그림 48]의 실험과 동일하게 base 크기의 두 모델인 mBERT-B와 UKnowBERT-B에 대해 사전 학습 진행 단계별 KorQuAD v1.0의 성능을 그래프로 표현했다. 앞선 실험과 마찬가지로 총 25만 스텝의 사전 학습 과정 중에 5만 스텝 단위로 모델을 저장하여 기계 독해 영역에 대한 성능을 비교했다.



[그림 49] mBERT-B와 UKnowBERT-B의 사전 학습 진행 단계에 따른 KorQuAD v1.0의 결과 그래프

[그림 49]는 base 크기의 모델에 대해 사전 학습 단계별 기계 학습 영역의 결과 그래프이다. 언어 지식을 사용한 UKnowBERT-B는 다중-핫 표현만 사용한 mBERT-B에 비해 1% 포인트 높은 성능을 유지하는 형태의 결과로 나타났다. 두 모델은 모두 150,000 스텝 전까지 일정한 증가폭을 보였으며, 이후 수렴함을 보였다.

사전 학습 단계에 따른 기계 독해 영역에서의 성능 평가의 결과를 통해 동일한 말뭉치를 사용해 사전 학습을 진행해도 언어 지식을 추가하여 여러 정보를 언어 모델에 주입하는 경우 각 정보가 서로 영향을 주어 빠른 수렴과 더 높은 성능을 가져오는 효과를 보였다. 이를 통해 언어 모델이 다양한 표현과 의미를 이해하기 위해 말뭉치를 늘리는 방법이 아닌 언어 지식을 활용하는 방법도 사용이 가능하며, 이미 구축된 다양한 언어 지식을 활용하게 되면 말뭉치 구축에 필요한 비용을 줄일 수 있는 방법이 될 수 있다.

6 결론

본 논문은 다양한 언어 지식과 문맥 정보를 융합한 형태소 단위의 언어 모델인 UKnowBERT(Ulsan Knowledge ensemble BERT)를 제안했다. 형태소 단위의 유지를 위해 토큰 목록에 존재하지 않은 형태소들에 대해 음절 토큰의 집합으로 표현했으며, 다중-핫 입력 생성기를 통해 형태소 단위를 유지하여 언어 모델에 입력으로 사용했다. 그리고 단어가 가진 다양한 의미와 표현을 학습하기 위해 언어 지식과 기존의 문맥 정보를 함께 사전 학습(pre-training)하는 언어 모델을 설계했다.

먼저 실험을 통해 다중-핫 표현 방식에 적합한 손실 함수로 본 논문에서 제안한 정답의 수에 따른 목표 확률을 조정하는 손실 함수를 채택했다. 그리고 다중-핫 표현 방식과 기존 모델의 비교 실험을 통해 형태소 단위의 유지가 중요한 응용 영역이나, 신조어나 이모티콘과 같이 기존의 토큰 표현 방식으로 표현할 수 없는 단어들을 다중-핫 표현 방식을 통해 토큰으로 표현할 수 있어서 성능의 향상을 가져왔으며, 언어 모델이 형태소 단위를 입출력으로 사용할 수 있음으로 인해 주변의 문맥 정보가 고스란히 해당 형태소에 전달됨을 보였다.

언어 지식을 융합한 UKnowBERT 모델과 다양한 비교 모델들의 비교 실험을 통해 용언과 명사의 관계 정보의 경우 문맥의 해석이나, 문장의 관계 정보를 해석하는 분야에서 효과적임을 확인했으며, 단어의 상위어나, 뜻풀이 기반의 벡터(USenseVector)의 경우 해당 단어의 의미의 해석이 중요한 영역에서 적용될 수 있음을 확인했다. 하지만, 각각의 정보를 분리해서 사용할 경우 다른 영역에서 성능의 손실이 발생했으며, 이를 줄이기 위해 다양한 언어 지식을 학습에 사용하는 것이 적합함을 확인했다. 그리고 언어 모델의 크기가 커질수록 많은 언어 지식을 파라미터에 담을 수 있어 기존 언어 모델보다 좋은 성능을 보였다. 제한된 환경에서의 실험을 통해 언어 지식이 비슷한 표현이나 단어에 대한 부족한 이해와 해석에 도움이 되어 제한하지 않은 환경에 비해서 성능의 하락이 적게 발생했다. 마지막으로 사전 학습 단계별 성능 비교 실험을 통해 언어 지식을 함께 사전 학습하는 경우 전체 모델의 빠른 학습 수렴을 가져올 수 있으며, 언어 지식을 사용하지 않은 모델에 비해 높은 성능을 보임으로, 언어 지식을 함께 학습함으로 인해 사전 학습에 필요한 비용을 줄일 수 있는 하나의 방안이 될 수 있다.

본 논문에서 제안한 UKnowBERT 모델을 통해 형태소 단위를 유지할 수 있음으로 인해 형태소 단위로 구축된 다른 언어 지식에 대한 쉬운 접근이 가능하며, 응용 영역에서도 형태소 단위로 결과를 도출할 수 있어서 쉽게 서비스를 구성할 수 있다. 그리고 언어 지식을 언어 모델이 학습함으로 인해 나타나는 다양한 효과들을 통해 언어 모델을 연구에서 말뭉치의 부족을 해결할 수 있는 하나의 방안을 제안할 수 있으며, 마지막으로 본 연구의 결과를 통해 자연어 이해 영역을 넘어서 자연어 생성 모델에도 언어 지식을 통한 확장 방법에 대한 연구가 필요하다.

참고 문헌

[1] Mikolov, T., et al., *Efficient estimation of word representations in vector space.* arXiv preprint arXiv:1301.3781, 2013.

[2] Vaswani, A., et al., *Attention Is All You Need.* Advances in Neural Information Processing Systems 30 (Nips 2017), 2017. 30.

[3] Bengio, Y., et al., *A neural probabilistic language model. journal of machine learning research, Vol. 3, No*. 2003, Feb.

[4] Pennington, J., R. Socher, and C.D. Manning. *Glove: Global vectors for word representation*. in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014.

[5] Bojanowski, P., et al., *Enriching word vectors with subword information.* Transactions of the association for computational linguistics, 2017. 5: p. 135-146.

[6] Peters, M., et al., *Deep contextualized word representations. arXiv 2018.* arXiv preprint arXiv:1802.05365, 2018. 12.

[7] Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. 9(8): p. 1735-1780.

[8] Devlin, J., et al., *Bert: Pre-training of deep bidirectional transformers for language understanding.* arXiv preprint arXiv:1810.04805, 2018.

[9] Radford, A., et al., *Improving language understanding by generative pre-training.* 2018.

[10] Joshi, M., et al., *Spanbert: Improving pre-training by representing and predicting spans.* Transactions of the Association for Computational Linguistics, 2020. 8: p. 64-77.

[11] Xiao, D., et al., *ERNIE-Gram: pre-training with explicitly N-Gram masked language modeling for natural language understanding.* arXiv preprint arXiv:2010.12148, 2020.

[12] Liu, Y., et al., *Roberta: A robustly optimized bert pretraining approach.* arXiv preprint arXiv:1907.11692, 2019.

[13] Lample, G. and A. Conneau, *Cross-lingual language model pretraining.* arXiv preprint arXiv:1901.07291, 2019.

[14] Yang, Z., et al., *Xlnet: Generalized autoregressive pretraining for language understanding.* Advances in neural information processing systems, 2019. 32.

[15] Lan, Z., et al., *Albert: A lite bert for self-supervised learning of language representations.* arXiv preprint arXiv:1909.11942, 2019.

[16] Bai, S., J.Z. Kolter, and V. Koltun, *Deep equilibrium models.* Advances in Neural Information Processing Systems, 2019. 32.

[17] Dehghani, M., et al., *Universal transformers.* arXiv preprint arXiv:1807.03819, 2018.

[18] Clark, K., et al., *Electra: Pre-training text encoders as discriminators rather than generators.* arXiv preprint arXiv:2003.10555, 2020.

[19] Goodfellow, I., et al., *Generative adversarial networks.* Communications of the ACM, 2020. 63(11): p. 139-144.

[20] Radford, A., et al., *Language models are unsupervised multitask learners.* OpenAI blog, 2019. 1(8): p. 9.

[21] Brown, T., et al., *Language models are few-shot learners.* Advances in neural information processing systems, 2020. 33: p. 1877-1901.

[22] Bai, Y., et al., *Training a helpful and harmless assistant with reinforcement learning from human feedback.* arXiv preprint arXiv:2204.05862, 2022.

[23] Kaelbling, L.P., M.L. Littman, and A.W. Moore, *Reinforcement learning: A survey.* Journal of artificial intelligence research, 1996. 4: p. 237-285.

[24] Touvron, H., et al., *Llama: Open and efficient foundation language models.* arXiv preprint arXiv:2302.13971, 2023.

[25] Thoppilan, R., et al., *Lamda: Language models for dialog applications.* arXiv preprint arXiv:2201.08239, 2022.

[26] Lewis, M., et al., *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.* arXiv preprint arXiv:1910.13461, 2019.

[27] Raffel, C., et al., *Exploring the limits of transfer learning with a unified text-to-text transformer.* The Journal of Machine Learning Research, 2020. 21(1): p. 5485-5551.

[28] Levine, Y., et al., *SenseBERT: Driving some sense into BERT.* arXiv preprint arXiv:1908.05646, 2019.

[29] Miller, G.A., *WordNet: a lexical database for English.* Communications of the ACM, 1995. 38(11): p. 39-41.

[30] Peters, M.E., et al., *Knowledge enhanced contextual word representations.* arXiv preprint arXiv:1909.04164, 2019.

[31] Yamada, I., et al., *LUKE: Deep contextualized entity representations with entity-aware self-attention.* arXiv preprint arXiv:2010.01057, 2020.

[32] Yu, W., et al., *Dict-bert: Enhancing language model pre-training with dictionary.* arXiv preprint arXiv:2110.06490, 2021.

[33] Chen, Q., et al., *Dictbert: Dictionary description knowledge enhanced language model pre-training via contrastive learning.* arXiv preprint arXiv:2208.00635, 2022.

[34] Sun, T., et al., *Colake: Contextualized language and knowledge embedding.* arXiv preprint arXiv:2010.00309, 2020.

[35] Yu, D., et al. *Jaket: Joint pre-training of knowledge graph and language understanding*. in *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022.

[36] Lee, S., et al., *A Small-Scale Korean-Specific BERT Language Model.* Journal of KIISE, 2020. 47(7): p. 682-692.

[37] 김동규, et al., *KB-BERT: 금융 특화 한국어 사전학습 언어모델과 그 응용.* 지능정보연구, 2022. 28(2): p. 191-206.

[38] 박상민, et al., *KoLegal-BERT: 법률 도메인 텍스트 마이닝을 위한 법률 언어 표상 모델.* 한국정보과학회 학술발표논문집, 2022: p. 1061-1063.

[39] 장지모, 민재옥, and 노한성, *KorPatELECTRA: 자연어처리 분야에서의 성능 향상을 위한 한*

국어 특허 문헌 사전학습 언어모델 (KorPatELECTRA). 한국컴퓨터정보학회논문지, 2022. 27(2): p. 15-23.

[40] Kim, B., et al., *What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers*. arXiv preprint arXiv:2109.04650, 2021.

[41] Schuster, M. and K. Nakajima. *Japanese and korean voice search*. in *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. 2012. IEEE.

[42] Shibata, Y., et al., *Byte Pair encoding: A text compression scheme that accelerates pattern matching*. 1999.

[43] Sennrich, R., B. Haddow, and A. Birch, *Neural machine translation of rare words with subword units.* arXiv preprint arXiv:1508.07909, 2015.

[44] Kudo, T. and J. Richardson, *Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing.* arXiv preprint arXiv:1808.06226, 2018.

[45] Ma, W., et al., *CharBERT: character-aware pre-trained language model.* arXiv preprint arXiv:2011.01513, 2020.

[46] Tay, Y., et al., *Charformer: Fast character transformers via gradient-based subword tokenization.* arXiv preprint arXiv:2106.12672, 2021.

[47] Szegedy, C., et al. *Rethinking the inception architecture for computer vision*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[48] Bae, Y.-j. and C. Ock. *Introduction to the Korean word map (UWordMap) and API*. in *Annual Conference on Human and Language Technology*. 2014. Human and Language Technology.

[49] 이주상, 신준철, and 옥철영, *단어 의미와 자질 거울 모델을 이용한 단어 임베딩*. 정보과학회 컴퓨팅의 실제 논문지, 2017. 23(4): p. 226-231.

[50] 임승영, 김명지, and 이주열, *KorQuAD: 기계독해를 위한 한국어 질의응답 데이터셋.* 한국정보과학회 학술발표논문집, 2018: p. 539-541.

[51] Maas, A., et al. *Learning word vectors for sentiment analysis*. in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 2011.

[Abstrat]

A Language Model of Morpheme Unit that Ensemble Knowledge of Language and Syntactic Information

Natural languages, as used and built by humans, have diverse and complex structures. Natural language representation has become a priority problem in natural language processing to represent natural language in a form that computers can understand. Many approaches have been used to represent natural language. Today, Language Models based on deep learning dominate natural language representation. Language models change the outcome of a natural language representation depending on the context. The various uses and evolutions of language models have led to significant advances in the field of natural language processing.

Language models use units of representation called tokens to efficiently represent natural language. A token is a unit of expression greater than or equal to a syllable and less than or equal to a word. However, tokens are generated using a frequency of occurrence, which makes it difficult to learn the meaning of words that occur infrequently in the training data. And, if a morpheme is represented by multiple tokens, it is a proper noun or neologism that has a low frequency in the training corpus. In this case, information from the surrounding context is not fully applicable to the morpheme. In addition, If the tokens that make up a morpheme comprise tokens that have been used in other senses, they may be interpreted as meaning something different from the original morpheme. This is more likely to happen the less often the morpheme is learned. To solve the problem of morpheme separation based on token representation, we need a training corpus of natural language in various forms. However, it is difficult to secure such a corpus in various forms, and it is expensive to build.

In this paper, we propose UKnowBERT (Ulsan Knowledge ensemble BERT), a morpheme

unit-based language model that fuses various language knowledge and contextual information. UKnowBERT uses a multi-hot representation to solve the problem of morpheme segregation due to token representation. Multi-hot representation represents morphemes as a set of syllable tokens when they cannot be represented by a single token and preserve morpheme units by the input generator. Since the multi-hot representation approach allows for more than one correct answer in masked LM, we used a loss function that adjusts the target probability based on the number of correct answers rather than a traditional loss function. Experiments on the loss function confirm that the loss function proposed in this paper is suitable for multi-hot representations. In addition, through a comparison experiment with the BERT model using the existing token method, the performance was 1% higher than the existing token representation method in the areas of object name entity recognition and sentiment analysis. However, when it came to understanding the relationships between words and sentences, as in machine reading comprehension and semantic role labeling, traditional models did not perform better.

We pre-trained language knowledge to solve the problem that a multi-hot representation that preserves morpheme units is not enough for a language model to understand the different meanings and expressions of a word. We used UWordMap, a Korean lexical semantic network, by extracting hypernym information, relational information using semantic constraints on verbs and adjectives, and information represented as a vector of word meanings. In the case of hypernym inference and lexicon-based vector learning, we applied it to the vectors representing the words, just as humans do, using prior acquired information. The language model is designed to interpret the context of a word to find the appropriate information among its many meanings and to proactively memorize information about a word if it has not been trained often enough.

In various experiments, we have shown that using the semantic constraints of verbs and adjectives outperforms other language models in machine reading comprehension and semantic role labeling. In addition, in the case of hypernym inference and lemma vector learning, it outperformed in areas where the meaning of words is important, such as person

name recognition. However, language models that only added semantic constraints on terms performed poorly on object name entity recognition. Similarly, hypernym inference and lemma vector learning performed poorly on machine reading comprehension and semantic role labeling. However, language models that used all language knowledge performed evenly across all domains. The small-sized model had areas of similar or lower performance than the traditional model, but the base-sized model outperformed the traditional BERT in all areas because it could remember more information for more parameters. In experiments with limited experimental environments, language knowledge helped to understand the application domain, resulting in less performance degradation than the existing BERT model and higher performance than the comparison model. Finally, we compared the performance of each pre-training phase and found that learning language knowledge led to faster convergence of the language model and outperformance by 1%.

The UKnowBERT proposed in this paper has the advantage of reducing the size of the overall language model by reducing the number of tokens, representing more words as tokens, and maintaining performance through multi-hot representation. In addition, by pre-training language knowledge together, the language model was able to acquire insufficient information that could not be obtained by corpus learning alone, and it had the effect of learning various expressions and meanings for words, resulting in higher performance than the existing BERT. The results of this study suggest that the multi-hot representation approach used by UKnowBERT may be a solution for a variety of other natural language processing applications that require morphological units. The cost or time required to build a corpus for an application domain or for pre-training a language model could be eliminated through language knowledge. Furthermore, research is needed on the applicability of linguistic knowledge to natural language generation models beyond natural language understanding.