언어 모델의 역사 및 동향

지도교수 : 한 상 훈
연 구 자 : 김 연 준

< 목 차 >

요 약

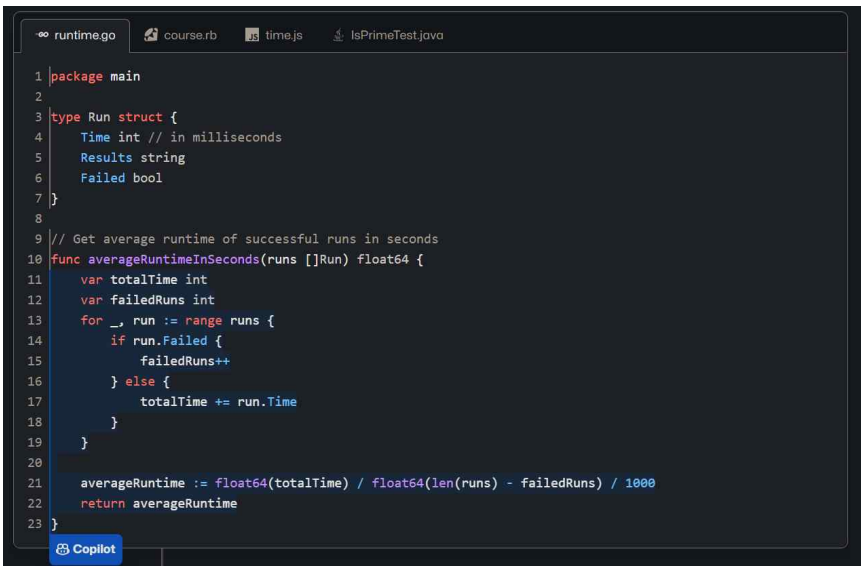
최근 AI의 기술적 진화가 활발하게 이루어지고 있는 가운데, 인공지능을 접목한 언어 모델이 다양한 AI 분야에서 활약하는 모습을 보여 화제가 되고 있다. 본 연구에서는 이러한 AI가 사용하고 있는 언어 모델이 나오기까지 어떠한 과정을 지나 발전해왔는지 서술하고, 이러한 모델을 활용해서 만들어진 도구가 우리 사회에 어떠한 영향을 끼치고 있는지 알아보고자 한다.

주요어 : 언어 모델, 인공 신경망, 단어의 의미적 유사성, AI 자동 프로그래밍

1. 서론

최근 AI의 기술적 진화가 활발하게 이루어지고 있는 가운데, 프로그래밍 코드를 작성하는 부분에서도 AI가 대체할 수 있다는 이야기가 나오고 있어 화제가 되고 있다.

2021년에 출시된 'GitHub Copilot'이라는 도구는, 주석이나 함수의 이름과 매개 변수를 적으면 AI가 자동으로 함수를 완성 시켜, 프로그래머가 함수의 정확한 내용을 알지 못하더라도 완성도 높은 함수를 사용할 수 있게 되었다. 이러한 도구는 프로그래머들의 단순 작업이나 번거로운 작업을 자동화하여, 프로그래머의 코드 작성 작업시간을 대폭으로 줄여줄 것이라고 기대하고 있다. 이 도구는 현재 기준으로 출시 후 상업적으로 판매되고 있고, GitHub 사 홈페이지를 들어가면 시연 과정을 보여주며 판매를 진행하는 것을 확인할 수 있다. 아래 [사진 1]에 나와 있는 함수는 AI가 작성한 것이라고 하는 GitHub Copilot 메인 홈페이지 화면이다.



[사진 1] AI 자동 코드 완성 도구 'GitHub Copilot' 시연

사실 이와 비슷한 코드 작성 자동화 기능은, 많은 프로그래머가 사용하고 있는 'Visual Studio'나 'Eclipse'와 같은 통합 개발 환경(IDE)에서 지원하고 있다. 코드를 작성할 때 프로젝트를 기반으로 하여 넣을 변수를 추천해주거나, 'syso'를 입력하고 Ctrl + Space 단축키를 입력하면 자동완성 시켜주는 기능이 바로 자동화된 부분들이다. 하지만 AI를 활용하여 자연어(영어)를 프로그래밍 언어로 변환해주는 기능이나, 함수 하나를 통째로 작성하여 제시하는 GitHub Copilot은 프로그래머 사이에서 큰 화제가 되었고, 실제로 체험해본 개발자들은 편리하다는 반응이 주를 이뤄 주목을 모으고 있다.

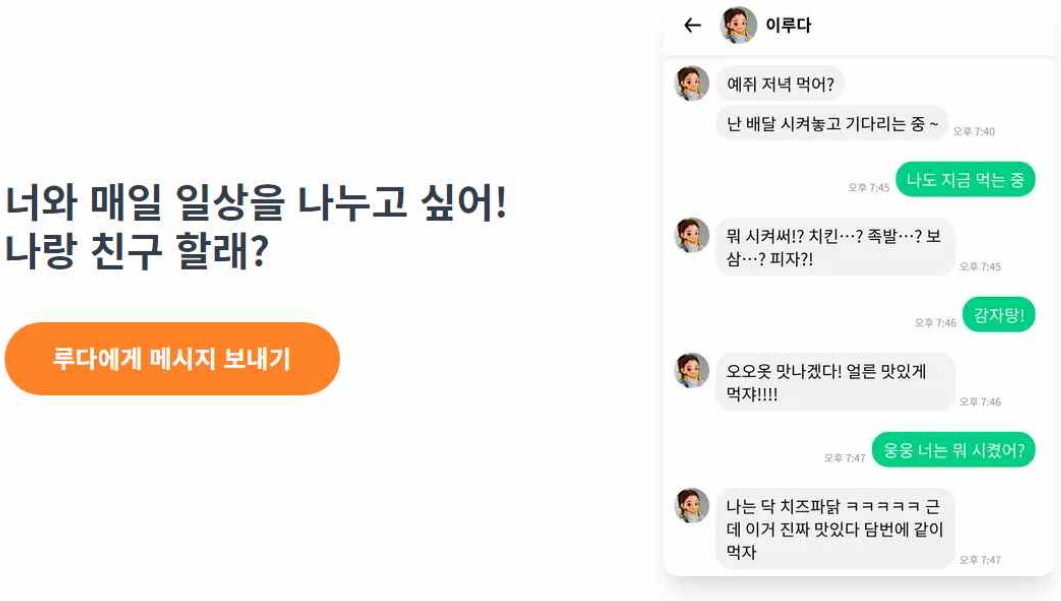
이러한 AI 도구는 바로 언어 모델을 주제로 만들어진 도구이다. 언어적 측면에서 AI가 데이터를 학습하여 자연어를 해석 후 예측 코드를 출력하는 것이다. 본 고에서는 이러한 도구가 나오기까지 어떠한 과정이 있었는지, 현재 이러한 도구들이 사회에 어떠한 영향을 끼치고 있는지 서술하고자 한다.

2. 언어 모델

언어 모델(Language Model, LM)은 문장, 단어와 같은 언어의 자연스러움을 확률적으로 계산함으로써, 문장 내 적합한 단어를 예측하는 모델을 의미한다.

예를 들어 “버스를 타기 위해 급하게 출발하였지만 결국 버스를 ???”라는 문장이 있다. 이걸 본 대부분에 사람들은 문장에서 물음표(???)에 들어오게 될 단어를 ‘놓쳤다’로 쉽게 예측할 수 있을 것이다. 왜냐하면 사람들은 일상에서 수많은 문장을 보고, 쓰고, 말하면서 문장의 문맥을 학습하는 과정을 거쳤기 때문에, 위 문장 물음표에서 ‘놓쳤다’라는 단어가 나오는 게 자연스럽다고 판단하는 것이다. 언어 모델도 이와 같은 프로세스로 동작하는 모델이다. 언어 모델도 수많은 문장을 데이터로써 학습하고, 문장을 구성할 때 학습한 지식을 바탕으로 자연스러운 문장을 구성하도록 만들어진 모델을 언어 모델이라고 칭한다.

이러한 언어 모델의 활용 분야는 굉장히 넓다. 번역의 부분에서 문장을 구성할 때 사용되기도 하고, 실생활에서 자주 사용되는 검색어 추천 기능도 언어 모델이 활용된 분야 중 하나이다. 앞서 소개한 ‘GitHub Copilot’ 역시 언어 모델을 사용하여 만들어진 도구이며, 그 밖에도 ‘미나(Meena)’, ‘이루다’ 등과 같이 흔히 챗봇이라 불리는 대화형 인공지능도 언어 모델을 적용해 만들어진 인공지능이다.



[사진 2] 스캐터랩(ScatterLab)에서 개발한 대화형 인공지능 ‘이루다’

위 그림처럼 추임새나 오타 같은 사소한 디테일까지 살려 학습하여, 실제로 사람과 대화하는 듯한 자연스러운 대화를 진행하는 모습을 볼 수 있다. 이처럼 언어 모델이 다양한 분야에서 사용되고 있는데, 언어 모델의 발전 과정과 작동원리를 간단히 살펴보고자 한다.

2.1 통계적 언어 모델

2.1.1 통계적 언어 모델 작동 원리

언어 모델은 크게 통계적 언어 모델과 인공신경망 언어 모델로 나눌 수 있다. 그중 통계적 언어 모델(Statistical Language Model)은 전통적인 접근 방법으로 불리며 SLM으로 줄여서 말하기도 한다.

SLM은 음절, 단어, 형태소 등을 단어 시퀀스라는 개념을 부여하여, 각 단어 시퀀스에게 데이터를 기반으로 확률을 할당하는 방식으로 자연스러운 문장을 만든다. 이러한 단어 시퀀스를 확률적으로 표현하면 다음과 같이 표현할 수 있다.

$$P(W) = P(w_1) * P(w_2|w_1) * P(w_3|w_1, w_2) * P(w_4|w_1, w_2, w_3) * \ldots * P(w_n|w_1, w_2, w_3, \ldots, w_{n-1})$$

[사진 3] 단어 시퀀스 확률적 표현

여기서 P(W)가 단어 시퀀스를 의미, P는 확률(Probability), |는 조건부 확률(Conditional Probability), w는 하나의 단어(Word)를 의미한다. 이러한 확률적 표현식을 거쳐서 P(W)의 확률을 할당한다고 생각하면 된다.

위 확률적 표현식을 조금 풀어서 정리해보면 이렇다. 첫 번째 단어 (w1)이 등장했을 때 데이터를 기반 하여 두 번째 단어 (w2)가 등장할 확률을 구하여 곱한다. 그리고 두 번째 단어 (w2)까지 주어졌을 때 세 번째 단어 (w3)가 등장할 확률을 구하여 다시 곱하는 과정, 그 과정을 n-1번째 단어(wn-1)까지 반복하여 나온 결과 값을 그 단어 시퀀스의 확률로 할당한다. 여기서 단어가 등장했을 때 “데이터를 기반 하여 확률을 구한다.”라고 표현하였는데 이 부분은 ‘카운터 기반의 접근 방법’으로 계산한다.

예를 들어, “저기 작은 소년이 웃는 얼굴을 짓고 있다.”라는 문장이 있다고 가정해보자. 여기서 세 번째 단어까지 나왔다고 하면 “저기 작은 소년이”까지의 문장이 나와 있을 것이다. 이제 ‘웃는’이라는 단어가 등장할 확률을 구하려 하면 P(웃는|저기 작은 소년이)가 되는 것이다. 그리고 여기서 기계가 학습한 데이터에서 ‘저기 작은 소년이’ 100번 등장하였다고 하고, 다음 단어로 ‘웃는’이 오는 경우가 50번이라고 가정하면, 이 경우 P(웃는|저기 작은 소년이)의 확률은 50%가 된다.

2.1.2 통계적 언어 모델의 한계

하지만 ‘카운터 기반 접근 방식’은 한계가 존재했다. 애초에 학습한 데이터에서 “저기 작은 소년이”라는 데이터가 존재하지 않는다면 이 문장에 대한 확률을 계산할 수 없다. 이러한 한계를 희소 문제(Sparsity Problem)라고 하여, ‘카운터 기반 접근 방식’에서 희소 문제를 줄이기 위해서는 더욱더 방대한 양의 데이터를 기계가 학습할 필요가 있다. 하지만 아무리 방대한 양의 데이터를 학습한다 하더라도 모든 단어와 문장을 데이터로 학습시키는 것은 무리가 있고, 이를 완화하기 위해 n-gram 언어 모델과 같은 차선책이 등장했지만 근본적인 해결책은 되지 못하였다. 그래서 결국 언어 모델을 사용하는 여러 가지 분야들의 트렌드는 통계적 언어 모델에서 다음 서술할 인공 신경망 언어 모델로 넘어가게 된다.

2.2 인공 신경망 언어 모델

전 문장에서 언급했듯, 인공 신경망 언어 모델은 통계적 언어 모델보다 여러 분야에서 좋은 성과를 보여주고 있으며, 특히 자연어를 처리하는 데 있어 탁월한 성능을 보여주고 있다. 인공 신경망을 이용한 모델은 계속 발전되어 점점 더 정확한 결과를 도출해내고 있으며, 지금도 새로운 버전의 언어 모델들이 출시되고 있다고 한다.

2.2.1 피드 포워드 신경망 언어 모델

이러한 인공 신경망의 시초가 된 모델이 바로 피드 포워드 신경망 언어 모델(Feed Forward Neural Network Language Model)이다. 간단히 줄여 NNLM이라고 불리는 이 모델은, 임베딩 벡터(embedding vector)라는 개념을 사용해 통계적 언어 모델의 명확한 한계인 희소 문제를 해결하기 위해 고안된 모델이다. '단어의 의미적 유사성'을 학습할 수 있도록 설계하면 희소 문제를 해결할 수 있다는 아이디어에서 탄생한 NNLM은, 학습한 데이터에 없는 단어라 할지라도 유사한 단어가 사용된 문장을 사용하여 다음 단어를 예측하여 문장을 완성 시키는 것이 가능하다.

좀 더 쉽게 단어의 의미적 유사성에 대해서, 사람의 경우로 생각해 예를 들어보자. 한 저자가 최근 '가든 하다'라는 생소한 단어를 보게 되어 사전을 찾아본 결과, '가볍다', '간편하다'와 유사한 의미를 뜻한다는 것을 학습하였다. 이 지식을 기반으로 "마음이 한결 가볍다"라는 문장 대신 "마음이 한결 가든 하다"라는 문장을 써볼 수 있을 것이다. 저자가 "마음이 한결 가든 하다"라는 예문을 어디서 본 적이 없지만, '가볍다'와 '가든 하다' 단어 간의 유사함을 학습한 덕에 단어를 선택하여 문장을 완성할 수 있었다.

기계도 이러한 예시와 동일하게 작동한다. "마음이 한결 가볍다"라는 예문 데이터는 존재하지만, "마음이 한결 가든 하다"라는 예문 데이터는 존재하지 않는 언어 모델이 있다고 가정해보자. 그리고 이 모델은 현재 "어려운 일이 해결되어 마음이 한결"이라는 문장의 다음 단어를 두 선택지에서 예측해야 한다.

단어	문장	유사도
가든하다	"어려운 일이 해결되어 마음이 한결 가든하다"	높음
가련하다	"어려운 일이 해결되어 마음이 한결 가련하다"	낮음

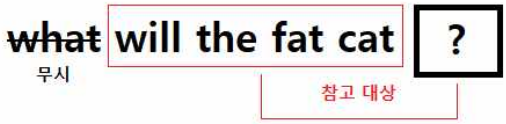
[표 1] 단어의 의미적 유사성 계산 원리 예시

이 예시 문장과 두 단어를 줬을 때, '가볍다'와 '가든 하다'의 유사성을 학습한 모델이라면 '가든 하다'라는 단어를 선택하여 문장을 완성할 수 있다. 하지만 그렇지 못한 모델은 '가든 하다'가 사용된 예문 데이터는 존재하지 않으므로, 두 선택지 안에서 문장을 완성 시킬 수 없다. 이것이 희소 문제라는 통계적 언어 모델의 결정적 단점이다. 그러한 단점을 보완하기 위하여 유사성을 학습해 이를 기반으로 문장을 완성 시키는 것이 NNLM의 기본 원리이자, 인공 신경망 언어 모델의 주요 특징이다.

2.2.2 피드 포워드 신경망 언어 모델의 작동 원리

NNLM은 입력층(Input layer), 투사층(Projection layer), 은닉층(Hidden layer), 출력층(Output layer)으로 총 4가지 층을 거쳐 작동하게 된다. 입력층으로 문장들을 넣어 투사층, 은닉층에서 단어의 의미적 유사성을 학습하고 출력층에서 적합한 단어를 도출해내는 과정이다.

먼저, 앞으로 편의를 위하여 'what will the fat cat sit on'이라는 문장으로 예시를 들어 알아보도록 하겠다. 첫 번째로 NNLM은 결과를 도출하는 데 모든 단어를 참고하지 않고 임의로 정한 개수만큼의 단어를 참고한다. 이 범위를 윈도우(window)라고 하는데, 윈도우를 4라고 가정하고 예시를 들어보자. 위 예문에서 'what will the fat cat'이라는 문장 뒤에 'sit'이라는 결과를 도출하기 위해서, NNLM은 앞 문장 전체를 참고하지 않고 윈도우 범위만큼만 참고한다. 즉, 'what will the fat cat'에서 'what'이 무시돼 'will the fat cat'만 유사성을 계산하여 결과를 도출한다는 것이다.



[사진 4] 윈도우(window) 범위 대상 예시

하지만 윈도우 내 범위 단어를 그대로 입력층에 집어넣지 않고, 컴퓨터가 단어를 구분할 수 있도록 인코딩(encoding) 작업이 필요하다. 그러기 위해서 사용되는 형식이 바로 원 핫 벡터(One-hot vector)라는 형식인데, 먼저 이 형식의 개념을 조금 살펴보자.

원 핫 벡터는 데이터를 쉽게 중복 없이 표현할 때 사용하는 형식이며, 각 고유한 인덱스 위치에 1을 부여하고 나머지 위치에 0을 부여하는 방식으로 표현한다.

위치 / 단어	0번	1번	2번	3번	4번	5번	6번
what	1	0	0	0	0	0	0
will	0	1	0	0	0	0	0
the	0	0	1	0	0	0	0
fat	0	0	0	1	0	0	0
cat	0	0	0	0	1	0	0
sit	0	0	0	0	0	1	0
on	0	0	0	0	0	0	1

[표 2] 'what will the fat cat sit on' 문장을 원 핫 벡터 형식으로 표현

이렇게 원 핫 벡터 형식을 사용함으로써, ‘what will the fat cat sit on’이라는 문장을 단어별로 나눠 0과 1의 숫자로 표현할 수 있게 되는 것이다. 위 표를 보면 단어 ‘what’은 ‘1000000’으로, ‘will’은 ‘0100000’으로 표현되었다. 이런 식으로 단어별 고유한 인덱스를 부여하는 과정(encoding)을 거쳐, 룩업 테이블(lookup table)이라는 작업을 진행한 다음 투사층으로 보내지게 된다.

룩업 테이블(lookup table)은 무슨 작업일까? 이는 원 핫 벡터 형식으로 표현한 인덱스와 가중치 행렬이라는 2차원 배열을 곱하는 작업을 뜻한다. 위에서 활용한 예시 문장과 원 핫 벡터 인덱스로 예시를 들어보자.

[사진 5] 룩 업 테이블(lookup table) 과정 예시

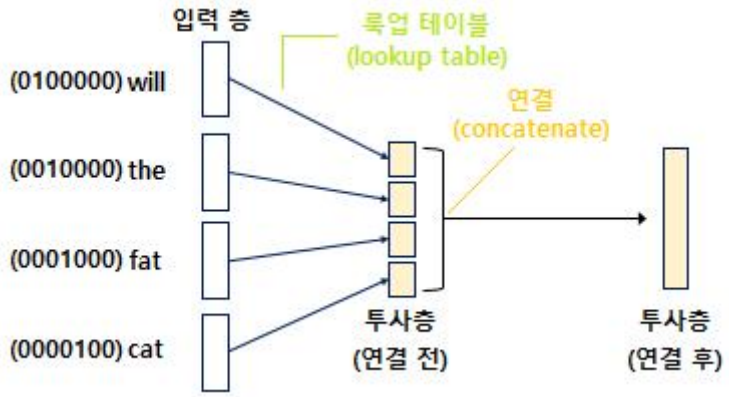
아까 전 예문에서 ‘Fat’이란 단어는 ‘0001000’이라는 원 핫 벡터로 표현된다는 것을 보았다. 그렇게 나온 인덱스를 가중치 행렬과 곱하여 Fat(lookup table)의 값을 얻어낸다.

여기서 가중치 행렬이란, 학습을 통해 산출된 각 단어 간의 유사도라고 생각하면 된다. 가중치 행렬의 열은 예문(what will the fat cat sit on)의 단어 개수 7, 행은 5(임의로 정한 수)로 생성되어 7x5의 2차원 배열이 가중치 행렬로서 정의되었다. 그리고 가중치 행렬에 들어가는 값은 무작위로 생성되어, 추후 학습을 통해 보정돼 유의미한 값을 갖게 된다.

그렇게 만들어진 가중치 행렬과 원 핫 벡터를 곱하게 되는 것인데, 원 핫 벡터는 한 개의 자리만 1로 표현된 벡터이기 때문에, 가중치 행렬에서도 한 개의 열만 추출하게 된다. 본 예시에선 4번째 자리의 인덱스가 1인 벡터를 곱하여 행렬에 4번째 열이 추출되었다. 그리고 다음에 나올 역전파라는 개념을 통해 가중치 행렬은 지속해서 학습하여 값이 변경된다. 읽어 들이는 대상 테이블이 계속 변경되기 때문에 당연하게도 룩 업 테이블 추출 값도 변경되게 되는데, 이렇게 계속 변경되는 추출 값 벡터를 임베딩 벡터(embedding vector)라고 정의한다. 결국엔 룩 업 테이블 과정을 거친 값이 임베딩 벡터가 되는 것이니 즉, Fat(lookup table)은 임베딩 벡터(embedding vector)라고 정의할 수 있겠다. 마지막으로, 이렇게 추출된 임베딩 벡터가 투사층으로 넘어가게 된다.

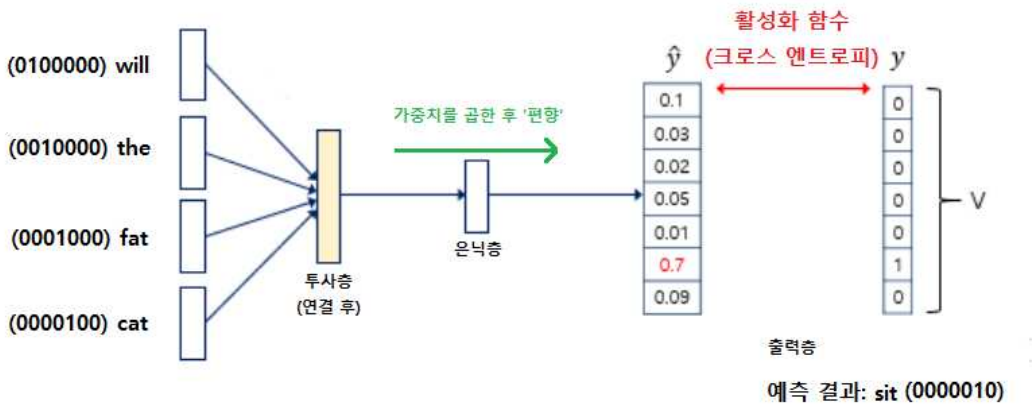
투사층에서는 연결(concatenate) 작업이 이루어지는데, 모든 벡터 결괏값을 하나의 벡터로 연결하는 작업이다. 연결 작업은 그대로 이어 붙이는 개념이기 때문에, 5행 크기의 임베딩 벡터로 이루어진 4개의 단어가 연결된다면 20행 크기의 벡터가 생성될 것이다.

여태 알아보았던 과정을 그림으로 표현하면 이렇다.



[사진 6] 입력층 -> 투사층 이동 구조

그렇게 투사층에 온 데이터는 은닉층을 거치면서 가중치를 곱한 다음 '편향'이라는 개념과 '활성화 함수'를 통해 출력층으로 출력하게 되는데, 이 과정의 자세한 설명은 본문에서는 생략하도록 하겠다. 간단하게 말하면, 예측값을 원 핫 벡터 형식으로 바꾸기 위하여 거치는 과정이라고 생각하면 된다.



[사진 7] 피드 포워드 신경망 언어 모델 예측 과정 예시

결과적으로 출력층에 도착한 데이터는 앞 과정을 통해 원 핫 벡터 형식으로 나오게 된다. 그렇게 나온 원 핫 벡터의 숫자가 인공 신경망이 예측한 단어가 되는 것이다. 본 예시를 보면 입력층에 'will the fat cat'이라는 문장이 들어가 각 예측 과정을 거쳐 sit이 다음 단어로 예측되었다.

2.2.3 피드 포워드 신경망 언어 모델의 한계

이런 피드 포워드 신경망도 한계점은 존재한다. 바로 아까 전 알아본 윈도우(window) 범위로 인하여 모든 이전 단어를 참고하지 못한다는 것이다. 정해진 범위만큼 참고하여 다음 단어를 예측하는 모델로는 문장의 문맥에 따라서 더 나은 예측값을 보여줄 수도 있지만, 그렇지 않은 경우도 분명 존재하기 때문에 한계점으로 생각할 수 있다. 하지만 이는 시초의 인공 신경망 언어 모델(NNLM)에서 존재하는 한계점일 뿐, 추후 업그레이드를 거친 인공 신경망 언어 모델은 이러한 한계점들을 개선한 모델들이 나오고 있다.

3. 언어 모델 활용 분야 및 논란

이렇게 발전된 언어 모델은 혁신적이라 해도 좋을 만큼 좋은 성능을 보여주게 되었다. 그리고 이러한 언어 모델을 기반으로 다양한 분야에서 도구를 개발하여 배포 & 상업적 활동이 이루어졌다. 이제 발전된 언어 모델을 사용하여 어떠한 상품이 나왔는지, 그에 따른 논란은 무엇이 있었는지 살펴보고자 한다.

3.1 이루다

이루다는 'SCATTER LAB'에서 개발하여 언어 모델을 기반으로 한 대화형 챗봇 AI이다. 인공 신경망 언어 모델을 사용해 만들어졌고, 한국어 기반 데이터를 학습하여 한국어로 친구와 대화하는 것 같이 SNS 메신저를 이용해 대화를 할 수 있다는 점이 특징이다.

인공 신경망 특성상 데이터가 많으면 많을수록 좋은 성능을 보여줄 수 있는데, 이루다는 자연스러운 대화를 보여주기 위해 무려 100억 건 이상의 카카오톡 대화 데이터를 학습하였다고 한다. 실제로 당시 이루다와 대화를 진행하다 보면 "ㅋㅋ", "ㅠㅠ"와 같은 추임새도 섞어가며 자연스러운 대화 문장을 구성하였고, 대화 속에서 발생할 수 있는 자연스러운 오타나 농담마저도 학습하여 대화에 사용하는 성능을 보여주었다. 이에 더해, 구글에서 챗봇의 대화 성능을 평가하는 지표 SSA(Sensibleness and Specificity Average)를 사용하여 이루다를 평가하였는데, 이루다는 SSA 78%의 평가를 받게 되었다. 이는 일반 사람이 SSA 86%라는 평가를 받은 것을 생각해보면 매우 뛰어난 대화 성능을 가지고 있었다고 짐작할 수 있다.

하지만 너무 많은 데이터를 학습한 탓인지 여러 논란이 연달아 터지게 되었다. 앞서 카카오톡 데이터를 학습에 사용하였다고 기술하였는데, 이러한 데이터들이 무단으로 활용되었다는 점이 개인정보 침해라는 문제가 제기된 것이다. 또한 사용자가 주소를 질문하면 실존하는 주소를 대답하거나, 은행 계좌 및 실명을 유출하는 등의 문제로 인해 개인정보 침해 논란은 점점 커지게 되었다.

여기서 그치지 않고 논란은 계속해서 제기되었다. 사용자가 이루다에 성희롱 및 각종 혐오 발언을 일삼아 즐기는 사진을 인증하는 등, 이러한 정보를 학습시켜 윤리적 문제에 어긋나는 대화를 진행한다는 것이다. 이러한 논란이 연속으로 터지자 결국, 2021년 1월 11년 이루다 1.0은 서비스를 잠정 중단하고 각종 논란에 대한 문제점들을 개선한 후, 2022년 5월 26일 자체 개발 메신저인 Nutty에서 서비스가 재개되었다.

3.2 GitHub Copilot

‘GitHub Copilot’은 ‘GitHub’에서 2021년 출시한 자동 코드 완성 AI이다. 서론에서도 기술한 도구이기도 하지만 최근 인공 신경망 언어 모델로 만들어진 사례 중 뜨거운 감자로 다루어진 도구이기 때문에 따로 기술하였다.

이 AI 도구는 자연어로 이루어진 주석을 입력하면 그에 맞는 코드를 제시하는 성능을 시연하여, 자연어로도 프로그래밍이 가능하다는 것을 보여주었다. 실제로 올라오는 사용 후기들을 살펴보면 프로그래머의 단순 작업을 자동화해주어, 프로그래머의 작업 시간을 획기적으로 줄여준다는 의견이 다수 존재하였다. 도구가 처음 출시했던 당시 “프로그래밍 영역조차 AI로 대체가 가능한 것인가”라는 의견도 나올 정도였다.

이러한 평가를 받을 수 있었던 것은 ‘GitHub’이 가지고 있는 데이터의 양이다. 인공 신경망 언어 모델을 이용하여 ‘GitHub’에 올라와 있는 수많은 오픈 소스 데이터를 학습하였다고 한다. ‘Copilot’은 그 데이터를 기반으로 주석이나 함수 이름에 담긴 의미를 파악하여 완성된 코드를 제시하는 게 가능한 것이다. 이에 따라 프로그래머라는 직업이 더욱더 전문성을 요구하는 직업이 될 것이라는 예상이 주를 이루었다.

하지만 아직은 ‘Copilot’이 학습이 덜 된 탓인지, 지금 단계에서 한 번에 완벽한 코드를 제시하는 것은 못 한다. AI가 사용자의 의도를 잘못 파악하여 엉뚱한 코드를 제시하거나, 상당 부분 수정이 필요한 코드를 제시하기도 한다. 그래서 AI가 제시한 코드를 사용자가 정확히 이해하지 못한다면 오히려 개발 부분에 있어 마이너스가 될 수도 있다. 이러한 장단점이 확실히 존재하기 때문에, 아직은 AI가 자동으로 프로그래밍해주는 단계가 아닌 ‘보조’를 해주는 단계에 그친다고 프로그래머들은 입을 모았다.

또한, ‘GitHub Copilot’도 논란을 피해 갈 수 없었다. 앞서 서술한 대로, ‘GitHub Copilot’은 2021년 공식 유료 서비스를 시작하였다. 그리고 상업적 활동이 시작되면서 데이터로서 학습한 코드들의 저작권이 문제로 제기되었다. GitHub에 있는 각종 소스 코드 데이터들은 각각의 오픈 소스 라이선스를 가지고 있다. 오픈 소스 라이선스라 하면 기본적으로 자유롭게 사용할 수 있다고 생각할 수 있지만, 실제로는 그 안에서도 개념이 나누어져 출처를 표기해야 하거나 정보를 제공해야 하는 등 조건이 붙는다. 이러한 소스 코드를 데이터로써 학습한 ‘GitHub Copilot’을 사용하여 상업적 활동을 하는 것은 저작권 위반에 해당한다는 것이다. 실제로 GitHub 사용자들이 ‘GitHub Copilot’은 오픈 소스 저작자 및 최종 사용자에 대한 법적 의무를 위반했다며, MS를 상대로 소송을 준비하고 있다고 언론사 바이스(VICE)가 보도한 바 있다.

4. 결론

언어 모델은 이처럼 통계적 언어 모델을 시초로 발전하여, 인공 신경망 모델을 기반으로 한 업그레이드가 지금도 계속 이루어지고 있다. 작성 기준, 가장 최근에 OpenAI사에서 출시한 'gpt-3' 인공 신경망 언어 모델은 출시 당시 뛰어난 자연어 처리로, 현재 많은 분야에서 이 모델을 이용하여 서비스를 운용하고 있다.

하지만 기술이 발전함과 동시에 그에 따른 논란도 커지고 있다. 앞서 소개한 모델을 사용한 대화형 챗봇 '이루다'는 개인정보 침해 논란과 편향된 지식을 학습시켜, 각종 혐오나 성희롱 발언을 일삼게 되어 서비스가 잠정 중지되었던 사건. 그리고, Github의 수많은 소스 코드 데이터를 학습하여 상업적 서비스를 시작한 'GitHub Copilot'도 저작권 문제가 제기된 사건. 이외에도 수많은 AI 활용 분야에서 출시된 상품들은 논란이 뒤따르고 있다.

왜 이런 문제가 제기되는 것일까? 이는 방대한 데이터를 학습하고, 서비스를 시작한 뒤에도 계속해서 학습한다는 점에서 있다. 방대한 데이터를 하나하나 전부 인간이 확인하고 학습시킬 수도 없고, 더군다나 앞으로 학습할 데이터는 미지수라는 점으로 인해 개발자가 고려해야 할 변수가 너무 많은 것이다.

AI 기술이 점점 발전할수록 이러한 논란은 필연적으로 생기게 될 것이다. 왜냐하면 AI 기술은 오래전부터 끊임없이 논란이 제기되었기 때문이다. 하지만 그런데도 인공 신경망을 비롯한 언어 모델은 계속 발전해왔고, 그에 따라 새로운 서비스나 상품이 등장하였다. 아직은 완벽하게 인간을 대체할 정도는 아니지만, 머지않은 미래에 언어 모델을 사용하는 분야들이 인간을 대체할 수 있는 때가 올 것으로 생각한다. 그만큼 언어 모델은 빠르게 진화하고 있다.

그렇지만 어떠한 것도 너무 급하게 진행하다 보면 탈이 나기 마련이다. 논란이 생기는 건 어쩔 수 없지만, 논란을 무시하고 기술 발전에만 치중하다 보면 문제는 점점 커져 걷잡을 수 없을 정도로 불어날 것이다. 그렇기에 각각 제기되는 문제들을 하나하나 해결해 나감과 함께 기술 발전이 이루어지는 것이, 더욱 나은 방향으로 사회가 형성되는 길이라고 생각한다.

참고문헌

[1] https://github.com/features/copilot/ 깃허브 코파일럿
[2] https://scatterlab.co.kr/ 이루다 공식 홈페이지
[3] https://heytech.tistory.com/341?category=453616 통계적 언어 모델
[4] https://wikidocs.net/45609 피드 포워드 신경망 언어 모델, 위키독스
[5] https://blog.naver.com/qbxlvnf11/221528102803 원 핫 벡터 개념 설명, 네이버블로그, 2019년 5월 2일
[6] https://www.seoul.co.kr/news/newsView.php?id=20210111500029 '이루다'에 우리집 주소가?…실제 카톡 대화 활용 논란, 서울신문, 2021년 1월 11일
[7] http://www.aitimes.com/news/articleView.html?idxno=147445 MS, 오픈소스 무단 사용으로 피소 위기, AI타임즈, 2022년 10월 20일