KoEPT: Transformer 기반 생성 모델을 사용한 한국어 수학 문장제 문제 자동 풀이

임상규*, 기경서*, 김부근*, 권가진*
*서울대학교 융합과학기술대학원
e-mail: {sk.rhim, kskee88, cd4209, ggweon}@snu.ac.kr

KoEPT: Automatically Solving Korean Math Word Problems using Generative Transformer

Sang-kyu Rhim, Kyung Seo Ki, Bugeun Kim, Gahgene Gweon
Graduate School of Convergence Science and Technology,
Seoul National University

요 약

이 논문에서는 자연어로 구성된 수학 문장제 문제를 자동으로 풀이하기 위한 Transformer 기반의 생성 모델인 KoEPT를 제안한다. 수학 문장제 문제는 일상 상황을 수학적 형식으로 표현한 자연어 문제로, 문장제 문제 풀이 기술은 실생활에 응용 가능성이 많아 국내외에서 다양하게 연구된 바 있다. 한국어의 경우 지금까지의 연구는 문제를 유형으로 분류하여 풀이하는 기법들이 주로 시도되었으나, 이러한 기법은 다양한 수식을 포괄하여 분류 난도가 높은 데이터셋에 적용하기 어렵다는 한계가 있다. 본 논문은 이를 해결하기 위해 우선 현존하는 한국어 수학 문장제 문제 데이터셋인 CC, IL, ALG514의 분류 난도를 측정한 후 5겹 교차 검증 기법을 사용하여 KoEPT의 성능을 평가하였다. 평가에 사용된 한국어 데이터셋들에 대하여, KoEPT는 CC에서는 기존 최고 성능과 대등한 99.1%, IL과 ALG514에서 각각 89.3%, 80.5%로 새로운 최고 성능을 얻었다. 뿐만 아니라 평가 결과 KoEPT는 분류 난도가 높은 데이터셋에 대해 상대적으로 개선된 성능을 보였다.

1. 서론

수학 문장제 문제(math word problems)는 금융, 교육, 서비스 등 일상적으로 접할 수 있는 상황을 수학적으로 기술한 자연어 문제이다. 수학 문장제 문제 자동 풀이는 일상에서의 문제 해결에 활용될 수 있어 최근까지 꾸준히 연구된 바 있다. 이 문제들은 피연산자-연산자로 구성된 방정식 형태로 나타낼 수 있다. 각 문제를 방정식으로 표현하면, 연산자의 종류와 그 연산자들이 배치되는 위치는 동일하고 문제에 따라 그 문제에 대응되는 숫자만 서로 다르게 나타나는 형태의 방정식을 도출하게 되는 문제들이 존재할 것이다. 이 경우, 이 방정식들에서 나타나는 숫자를 추상화하여 하나의 '템플릿'으로 묶을 수 있다. 수학 문장제 문제와 방정식, 템플릿에 대한 예시는 표 1을 통해 확인할 수 있다.

<표 1> 수학 문장제 문제의 예시

문제	어느 미용사는 7%의 과산화수소 용액과 4%의 과산화수소 용액을 가지고 있다. 미용사는 5% 과산화수소 용액 300ml가 필요하다. 각 과산화수소 용액을 몇 ml씩 섞어야 하는지 구하여라.
방정식	$0.07 * x + 0.04 * y = 0.05 * 300$, $x + y = 300$
템플릿	$a * x + b * y = c * d$, $x + y = c$
정답	$x = 100$, $y = 200$

표 1과 같이 자연어로 구성된 수학 문장제 문제를 템플릿 형태로 표현한다고 할 때, 언어권을 막론하고 템플릿을 통해 기계 학습 모델을 사용하여 수학 문제를 풀기 위해서는 기계 학습 모델이 다음과 같은 조건을 충족해야 한다. 1) 문제의 구조를 파악해야 하며, 2) 문제 내의 자연어 정보들을 취합하여 템플릿에 문제에 알맞은 숫자 및 미지수를 삽입할 수 있어야 한다.

현재까지 한국어에서 이 두 조건을 충족하는 알고리즘을 개발한 연구 사례는 [1]과 [2]이다. 이 두 연구 사례는 분류(classification)모델에 속하는 알고리즘이다. 분류 모델은 데이터셋의 각 데이터들이 어떤 유형으로 분류될지를 학습하는 모델로, 수학 문제 풀이의 경우에는 각 문제들이 어떤 템플릿에 속할지에 대한 결정경계(decision boundary)를 학습하게 된다. 그러나 어떤 데이터셋들은 유형이 너무 많거나, 유형별 분포가 너무 유사하거나, 유형들 간의 분포가 균일하지 않아 결정경계를 설정하기 어려운 경우들이 있다. 이런 데이터셋을 분류 난도가 높다고 지칭할 수 있다([3]). 이렇게 분류 난도가 높은 데이터의 경우, 각 템플릿들의 결정경계를 학습하는 분류 방식보다는 데이터의 전체 분포를 모델이 학습함으로써 방정식을 직접 세울 수 있는 생성(generation) 방식을 쓰는 것이 더 적절하다. 나아가 분류 모델은 학습 데이터에 등장하는 템플릿만

출력할 수 있지만 생성 모델은 학습 데이터에 등장하지 않은 수식도 생성할 수 있다는 추가적인 강점을 지닌다.

본 논문은 단순 분류 모델을 쓰는 기존의 한국어 수학 문장제 문제 모델들의 한계점을 극복하는 모델인 KoEPT를 제시한다. 이에 따른 본 논문의 기여점은 다음과 같다.

1) 수식을 분류하는 대신 수식을 생성하여 분류 난도가 높은 데이터셋도 잘 학습하는 KoEPT를 개발했다.
2) KoEPT로 기존 한국어 데이터셋인 CC, IL, ALG514를 학습하여 CC에서는 최고 성능에 대등한 99.1%, IL에서는 85.2%에서 89.3%, ALG514에서는 78.6%(±0.5)에서 80.5%(±1.2)로 최고 성능을 얻었다.

2. 선행 연구

본 절에서는 (1) 최근에 시도되었던 수학 문장제 문제 풀이 연구의 맥락에 대해서 간단하게 소개하고, (2) 대표적인 연구 사례에 해당하는 EPT를 개괄한다.

먼저 최근에 시도된 연구 사례들은 크게 초기 연구와 후기 연구로 유형화할 수 있다. 초기의 연구들은 주로 문제 풀이에 도움이 될 자질을 사람이 직접 추출하여 모델을 학습시키는 방법을 활용했다([1],[6],[7],[8],[9]). 대표적으로 한국어의 경우, 사람이 직접 추출한 자질로 학습을 시도했던 모델은 [1]에서 제시됐다. [1]의 연구에서는 의존 구문 분석 및 언어 유형적 특성을 고려한 문맥 자질을 인위적으로 미리 추출하고자 하였으며, 대표적인 벤치마크 데이터셋인 ALG514[6]를 한국어로 번역하여 Log-linear 기반의 통계 모델로 학습을 수행하였다. 그러나 이런 방법을 채택한 모델들의 경우, 사람이 어떤 자질을 추출할 것인지를 정해줘야 학습이 가능하다는 단점을 가지고 있다. 이런 단점을 해결하고자 순수 신경망을 사용하는 후기 연구들이 제안되었다.

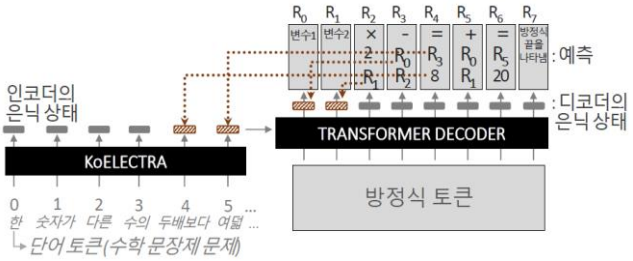
후기의 연구에서는 순수 신경망을 이용한 모델을 통해서 자질을 자동적으로 추출할 수 있도록 설계했다. 대표적으로 연구 [2]에서는 BERT[10]라는 사전 학습 언어 모델 (Pre-Trained Language Model)을 이용하고 방정식 템플릿으로 분류하는 순수 신경망 모델인 KoTAB을 만들었다. 분류 모델 방법 (Classification model) 기반인 KoTAB은 사람이 직접 정하는 언어적 자질 추출이 필요가 없어지도록 BERT를 사용하여 한국어에 대한 세계 지식(world knowledge)을 사용할 수 있었다. 하지만, 분류 모델 방법 기반의 모델들은 문장제 문제에 대해서 학습할 때 한개의 템플릿으로만 문제를 분류한다. 분류 모델을 사용하는 방법은 분류(Class)의 개수가 많고 각 분류 별로 데이터의 수가 적을 때, 분류 간의 결정경계를 학습할 데이터의 수가 충분하지 않아서 효과적이지 않다. 또한 분류 모델은 학습 데이터에 등장하는 템플릿만 출력할 수 있다. 학습 데이터에 등장하지 않는 새로운 유형의 문제를 보게 되면 기존에 학습한 템플릿 중에서만 출력을 선택할 뿐, 그 문제에 적절한 새로운 템플릿을 출력할 수 없다는 단점을 가지고 있다. 분류 모델의 단점을 보완할 수 있는 방안은 생성 모델 방법 (Generation model)이다. 생성 모델은 데이터셋의 분포 자체를 학습하기 때문에 분류의 영향을 상대적으로 적게 받는다. 또한 생성 모델은 처음 접하는 유형의 데이터를 접하더라도 새로운 출력을 생성할 수 있다는 점에서 분류 모델의 단점을 보완할 수 있다.

분류 모델이 아닌 생성 모델 기반의 순수 신경망 모델에 대한 영어 수학 문장제 문제 연구의 보고된 state-of-the-art는 Expression Pointer Transformer (EPT)[8]이다. EPT는 Transformer[11] 모델의 인코더-디코더(Encoder-Decoder) 구조를 가지고 있다. EPT의 인코더 부분에는 ALBERT[12]라는 사전 학습된 언어 모델이 적용되어 수학 문장제 문제의 언어적 특징을 자동으로 학습한다. 또한 디코더에서는 문제에 적절한 수식을 문장제 문제의 언어적 정보를 바탕으로 생성한다. 이렇게 현재 수학 문장제 문제 자동 풀이 연구에서는 생성 모델 기반의 접근이 새롭게 제기되는 실정이다. 본 연구에서는 영어에서의 최신 기술인 EPT 모델을 한국어에 적용할 수 있는 KoEPT를 제안했다.

3. 방법론

본 연구에서 제안하는 KoEPT는 한국어 수학 문장제 문제를 자동으로 풀기 위한 모델로, [13]에서 제안된 EPT의 구조에 기반하여 한국어에 맞게 개선된 생성 기반 모델이다. KoEPT는 Transformer처럼 인코더와 디코더 구조를 가지고 있다. KoEPT를 나타낸 그림 1에서 이러한 구조를 확인할 수 있다. 그림 1은 한국어 수학 문장제 문제를 자동 풀이하기 위해서 수식을 생성하는 과정을 나타낸 그림이다. 그림 1에는 “한 숫자가 다른 수의 두배보다 여덟이 더 크고 이 두 수의 합은 20이다. 두 수를 구하시오.”라는 문제의 일부를 적용한 예시를 보여줬다. KoEPT의 구조를 설명하기 위해서 인코더, 디코더, 그리고 출력단으로 나누어서 각 부분을 순차적으로 다루겠다.

(그림 1) KoEPT 모형도



인코더는 주어진 문제의 단어들을 입력으로 받아 KoELECTRA[14]라는 사전 학습 언어 모델에서 해당 문장제 문제 단어들의 토큰들을 얻는다. 이렇게 함으로써 KoEPT는 사람이 직접 자질을 선별할 필요가 없다. 대신, 인코더층에서 받은 정보를 바탕으로 적절한 연산자와 피연산자를 선택하여 수식을 생성할 수 있다. 여기서 KoELECTRA는 ELECTRA[15]의 구조를 이용한 한국어 언어 모델이다. EPT에서는 ALBERT를

썼지만 본 연구에서 ELECTRA 계열의 언어 모델을 썼다. 그 이유는 [15]에서 ELECTRA가 ALBERT를 포함한 다른 BERT 계열의 사전 학습 언어 모델보다 각종 downstream task에 좋은 성능을 낸다고 보고됐기 때문이다.

한편 디코더에서는 인코더의 토큰들을 받아 '식'(Expression) 토큰을 순차적으로 생성한다. 이때 '식' 토큰은 주어진 문제를 수식의 형태로 나타낸 것이다. 일례로 표 1의 경우, '식' 토큰은 데이터셋에 포함된 '0.07 * x + 0.04 * y = 0.05 * 300'와 'x + y = 300'의 정보를 연산자와 피연산자들의 묶음으로 나타내어 한 단위씩 나누어서 담고 있다. KoEPT는 다음에 올 것으로 예상되는 연산자와 피연산자를 묶어 동시에 출력한다. 또한 디코더에서는 피연산자와 관련된 문맥적 정보를 반영하기 위해 [16]의 '포인터 네트워크(Pointer Network)'를 응용한 '피연산자-문맥 포인터'를 사용한다. 이를 사용하여 KoEPT는 피연산자의 출처에 따라 피연산자와 직접 대응할 것으로 예상되는 후보 벡터를 선택하게 된다. 그림 1에서 이런 '포인팅'을 문제의 단어와 이전에 출력된 토큰들을 가리키는 점선 화살표들로 나타냈다.

마지막으로 디코더에서 받은 정보를 바탕으로 출력단에서 해당 문제에 대한 수식을 출력한다. KoEPT 출력단에서는 디코더에서 얻은 수식을 가지고 SymPy[17]를 통해서 수학 문장제 문제에 대한 답을 자동으로 출력한다.

4. 측정 지표 및 데이터셋

KoEPT을 분석하기에 앞서 본 섹션에서는 KoEPT로 실험한 데이터셋의 특성을 알아보기 위해 사용한 측정 지표에 대해서 설명하고 데이터셋에 대해서 소개를 한다. 먼저 본 논문은 측정 지표로'분류 난도'를 사용한다. 수학 문장제 문제 자동 풀이에서 앞서 언급한 분류와 생성 모델들 간의 성능 차이가 발생하는데, 그 이유는 학습 데이터셋의 분류 난도가 영향을 주었을 것으로 추정한다. 이를 보이기 위해서 분류 난도에 대한 객관적인 측정을 위한 척도가 필요하다. 분류 난도 척도에 관한 연구로는 [3]이 있다. [3]은 데이터셋의 분류 난도를 결정하는 대표적인 요소들을 제시하고 이를 종합적으로 측정하기 위한 척도를 제안했다. 이 척도는 크게 Shannon 다양성 인덱스[4], Hellinger 유사도[5], Distinct 단어 빈도 비율, 그리고 Imbalance 인덱스[3]를 종합하여 도출된다. 각 수치는 독립적으로 다음과 같은 의미를 가진다:

1) Shannon 다양성: 데이터셋에 존재하는 분류의 다양성
2) Hellinger 유사도: 분류 간 유사성
3) Distinct 단어의 빈도 비율: 전체 단어 수 대비 고유 단어의 수
4) Imbalance 인덱스: 분류 간 분포의 균일도

[3]은 이 네 가지 수치를 합산하여 데이터셋의 분류 난도를 평가할 수 있는 값을 얻을 수 있다고 보고한 바 있는데, 수학 문장제 문제 데이터셋에도 이 척도를 적용할 수 있을 것으로 판단된다.

KoEPT의 성능 측정 및 선행 연구 결과와의 비교를 위해, [1]과 [2]에서 번역한 한국어 데이터셋 CC, IL, ALG514를 실험에 사용하였다. CC는 [18]에서 commoncoresheets.com으로부터 600개의 문제를 수집하여 구축한 데이터셋이다. 이 데이터셋은 사칙연산으로 이루어진 일차방정식으로 구성되어 있다. CC의 템플릿 종류는 12가지가 있으며 각 종류마다 문제 수는 동일하다. IL은 [19]에서 제작한 데이터셋으로 562개의 사칙연산으로 이루어진 일차방정식 문제를 포함하고 있으며, CC와 마찬가지로 템플릿 종류는 12가지가 있다. 다만 CC와 달리, IL의 템플릿 별 문제의 개수는 균등하지 않다. ALG514는 [6]에서 제작한 데이터셋으로, algebra.com에서 수집한 514개의 문제로 구성되어 있다. 이 문제들은 모두 사칙연산으로 이루어져 있으며, 일차방정식 혹은 이원일차연립방정식에 해당한다. 하지만 CC나 IL과 달리, ALG514는 템플릿이 25개로 세 데이터셋들 중 가장 많은 템플릿을 가지고 있다.

각 데이터셋 별로 템플릿의 개수나 템플릿 별 문제의 개수 차이가 난도의 차이를 유발할 것으로 예상되기 때문에, [3]의 측정 기준을 CC, IL, ALG514에 도입하였다. 이 종합 수치를 데이터셋들에 대해 계산하면 데이터셋 간의 상대적인 분류 난도를 파악할 수 있다. 해당 측정 기준에 따른 본 연구의 데이터셋들의 난도는 표 2와 같다.

<표 2> 한국어 데이터셋 난도 비교

CC	IL	ALG514
3.89	4.09	4.84

표 2에 의하면 CC, IL, ALG514 한국어 수학 문장제 문제 데이터셋의 분류 난도를 비교했을 때, ALG514가 한국어 수학 문장제 문제 데이터셋 중에서 분류 난도가 가장 높은 데이터셋이라는 것을 알 수 있다.

5. 실험 및 분석

<표 3> KoEPT 성능 측정 결과 비교 (괄호: 표준편차)

	CC	IL	ALG514
Log-linear[1] (기존 모델)	-	-	78.6 (0.5)
KoTAB[2] (기존 모델)	99.3	85.2	-
KoEPT (제안 모델)	**99.1 (0.3)**	**89.3 (0.7)**	**80.5 (1.2)**

본 연구에서는 KoEPT가 출력한 식을 통해 도출한 답이 실제 답과 같은지를 비교하여 산출한 정답율을 평가 척도로 제시한다. 또한 객관적인 평가를 위해 데이터셋마다 5겹 교차 검증(5 fold cross-

[1] [1]에서 CC와 IL로 실험하지 않았다.

[2] [2]에서 KoTAB의 표준오차를 공개하지 않았다.

validation)을 이용하였다. 선행 연구에서 보고한 성능과 본 논문에서 진행한 실험 결과를 표 3을 통해 확인할 수 있다.

성능 결과와 데이터셋 별 난도를 고려하였을 때, 분류 난도가 가장 높은 데이터셋인 ALG514에서 KoEPT, KoTAB, Log-linear 세 모델 중 KoEPT가 가장 좋은 성능을 보였다. 뿐만 아니라 KoEPT는 CC와 IL처럼 분류 난도가 낮은 데이터셋조차 선행 연구보다 더 낫거나 비슷한 성능을 낸다는 것을 확인할 수 있었다. 따라서 실험 결과에 의하면 KoEPT의 성능은 분류 난도에 적게 영향을 받는 것으로 보인다.

실험 결과에 대해 우리는 다음과 같은 해석을 제시한다. KoEPT가 기존 모델들에 비해서 나은 성능을 나타내는 이유는 생성 모델이 데이터셋의 전체 분포를 학습하여 데이터셋을 구성하는 분류들의 경계만을 학습하는 분류 모델에 비해 더 많은 정보를 학습하는 모델이기 때문이다. 분류 모델은 데이터셋 분포의 결정경계만 탐색하면 된다는 이점이 있지만 탐색해야 하는 결정경계가 너무 많을 경우, 분포 자체를 학습하는 것이 더 정확도가 높게 나올 수 있다. 이는 표 2에서 분류 난도가 높다고 파악이 된 ALG514를 KoEPT가 기존 분류 모델보다 높은 정확도로 학습했다는 것을 통해 확인할 수 있다. 또한 생성 모델은 데이터의 분포 자체를 학습했기 때문에 분류 난도가 낮아 분포의 결정경계만 학습해도 높은 성능이 나오는 데이터셋에서도 좋은 성능을 낼 수 있다. 이는 CC와 IL 같은 데이터셋에 KoEPT가 좋은 성능을 냈다는 점을 통해 확인할 수 있다.

6. 결론 및 한계점

본 논문은 한국어 수학 문장제 문제 해결을 위해서, 기존 연구들과 달리 분류 난도가 높은 데이터셋도 학습할 수 있는 KoEPT를 제안했다. KoEPT는 인코더-디코더 구조를 가지고 있는 생성 기반의 모델이다. KoEPT의 성능을 평가하기 위해 우리는 현존하는 한국어 데이터셋 CC, IL, ALG514를 기준으로 KoEPT의 성능을 평가했다. 우리는 각 데이터셋의 분류 난도에 따른 KoEPT의 성능 변화도 함께 평가했다. 평가 결과, KoEPT는 CC에서 최고 성능에 대등한 성능을 냈고, IL과 ALG514에서 최고 성능 냈다. 즉, KoEPT는 분류 난도에 영향을 상대적으로 적게 받는다는 것을 확인했다. 이는 KoEPT가 데이터셋의 전체 분포를 학습하는 생성 모델을 사용하기 때문이라고 추정된다. 그러나 본 논문에서 실험한 데이터셋들은 모두 사칙연산 데이터셋이다. 따라서 KoEPT가 더 복잡한 문제에 대한 학습할 수 있는지는 본 연구로 확인하기 어렵다. 하지만 KoEPT는 생성 기반의 모델이므로 출력할 수 있는 수식의 종류가 다양하도록 설계할 수 있기 때문에 더 다채로운 수식의 형태를 포괄하고 있는 사칙연산 이상의 수준의 수학 문장제 문제에도 활용할 수 있을 것이라고 기대한다.

사사

이 성과는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2020R1C1C1010162).

참고문헌

[1] 우창협, 권가진, “한국어 수학 문장제 문제 자동 풀이”, 제30회 한글 및 한국어 정보처리 학술대회, 2018, pp. 310-315.

[2] K. Ki, D. Lee & G. Gweon, “KoTAB: Korean Template-Based Arithmetic Solver with BERT”, 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), 2020, pp. 279-282.

[3] E. Collins, N. Rozanov & B. Zhang, “Evolutionary Data Measures: Understanding the Difficulty of Text Classification Tasks.”, Proceedings of the 22nd Conference on Computational Natural Language Learning, 2018, pp. 380-391.

[4] C. E. Shannon, “A mathematical theory of communication.”, ACM SIGMOBILE Mobile Computing and Communications Review, Volume 5(1), pp. 3–55, 2001.

[5] L. Le Cam & G. L. Yang, “Asymptotics in statistics: some basic concepts.”, Springer Science & Business Media, 2012.

[6] N. Kushman, Y. Artzi, L. Zettlemoyer & R. Barzilay, “Learning to automatically solve algebra word problems.”, Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers, pp. 271-281, 2014.

[7] D. Zhang, L. Wang, L. Zhang, B. T. Dai & H. T. Shen, “The gap of semantic parsing: A survey on automatic math word problem solvers.”, IEEE transactions on pattern analysis and machine intelligence, Volume 42 no. 9, pp. 2287-2305, 2019.

[8] S. Roy & D. Roth, “Mapping to Declarative Knowledge for Word Problem Solving”, Transactions of the Association for Computational Linguistics, Volume 6, pp.159-172, 2018.

[9] L. Zhou, S. Dai & L. Chen, “Learn to solve algebra word problems using quadratic programming”, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 817-822.

[10] J. D. Kenton, M. W. Chang & L. K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.”, Proceedings of NAACL-HLT, pp. 4171-4186. 2019.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser & I. Polosukhin. “Attention is all you need.”, Advances in neural information processing systems, pp. 5998–6008, 2017.

[12] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, & R. Soricut. “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.”, International Conference on Learning Representations, 2019.

[13] B. Kim, K. Ki, D. Lee & G. Gweon, “Point to the Expression: Solving Algebraic Word Problems Using the Expression-Pointer Transformer Model.”, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020, pp. 3768-3779.

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

[15] K. Clark, M. Luong, Q. V. Le & C. D. Manning. “ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators.”, International Conference on Learning Representations. 2019.

[16] O. Vinyals, M. Fortunato & N. Jaitly, “Pointer networks.”, Advances in Neural Information Processing Systems 28, pp. 2692–2700, 2015.

[17] A. Meurer et al. “SymPy: Symbolic computing in Python.”, PeerJ Computer Science, Volume 3, 2017.

[18] S. Roy & D. Roth, “Solving General Arithmetic Word Problems.”, In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 1743-1752.

[19] S. Roy, T. Vieira & D. Roth, “Reasoning about quantities in natural language.”, Transactions of the Association for Computational Linguistics, Volume 3, pp. 1-13, 2015.