

숫자의 대소관계 파악을 위한 Explicit Feature Extraction(EFE) Reasoner 모델

안지수¹, 민태원², 권가진¹

¹ 서울대학교 융합과학기술대학원

² 서울대학교 자연과학대학 통계학과
{ajs7270, taywonmin, ggweon}@snu.ac.kr

Explicit Feature Extraction(EFE) Reasoner: A model for Understanding the Relationship between Numbers by Size

Jisu An¹, Taywon Min², Gahgene Gweon¹

¹Graduate School of Convergence Science and Technology, Seoul National University

²Dept. of Statistics, Seoul National University

요 약

본 논문에서는 서술형 수학 문제 풀이 모델의 숫자 대소관계 파악을 위한 명시적 자질추출방식 Explicit Feature Extraction(EFE) Reasoner 모델을 제안한다. 서술형 수학 문제는 자연현상이나 일상에서 벌어지는 사건을 수학적으로 기술한 문제이다. 서술형 수학 문제 풀이를 위해서는 인공지능 모델이 문장에 함축된 논리를 파악하여 수식 또는 답을 도출해야 한다. 때문에 서술형 수학 문제 데이터셋은 인공지능 모델의 언어 이해 및 추론 능력을 평가하는 지표로 활용되고 있다. 기존 연구에서는 문제를 이해할 때 숫자의 대소관계를 파악하지 않고 문제에 등장하는 변수의 논리적인 관계만을 사용하여 수식을 도출한다는 한계점이 존재했다. 본 논문에서는 자연어 이해계열 모델 중 SVAMP 데이터셋에서 가장 높은 성능을 내고 있는 Deductive-Reasoner 모델에 숫자의 대소관계를 파악할 수 있는 방법론인 EFE를 적용했을 때 RoBERTa-base에서 1.1%, RoBERTa-large에서 2.8%의 성능 향상을 얻었다. 이 결과를 통해 자연어 이해 모델이 숫자의 대소관계를 이해하는 것이 정답률 향상에 기여할 수 있음을 확인한다.

1. 서론

서술형 수학 문제는 일상에서 접할 수 있는 상황을 묘사한 문자열이 주어질 때, 주어진 정보를 이용하여 문제에서 요구하는 수식 혹은 정답을 도출하는 문제이다. 표 1은 서술형 수학 문제의 예시로 문제와 정답 수식을 나타낸다. 인공지능 모델이 서술형 수학 문제를 풀기 위해서는 자연어로 기술된 수학 문제를 이해하고, 문제 상황에서 필요한 정보를 추출한 뒤, 추출한 정보를 기반으로 적절한 수식 혹은 정답을 생

성하는 과정을 거치게 된다. 이러한 서술형 수학 문제 풀이를 통해 자연어 추론 능력을 평가하고 모델의 성능을 측정할 수 있다.

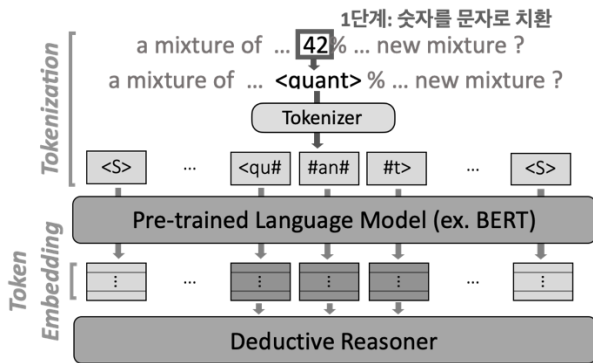
본 논문에서는 서술형 수학 문제 데이터셋인 SVAMP에서 자연어 이해 모델 중 최고 성능(SoTA)을 기록한 Deductive-Reasoner에서 숫자의 대소관계를 인지하지 않고 정답 수식을 도출하는 과정에서 숫자의 대소관계를 구분할 수 있도록 정보를 추가할 수 있도록 Explicit Feature Extraction(EFE) Reasoner 모델을 제

안한다. 이를 통해 문제의 논리적인 정보를 이해하여 문제를 해결할 때, 숫자의 대소관계 정보가 정답률 개선에 도움을 줄 수 있음을 확인한다.

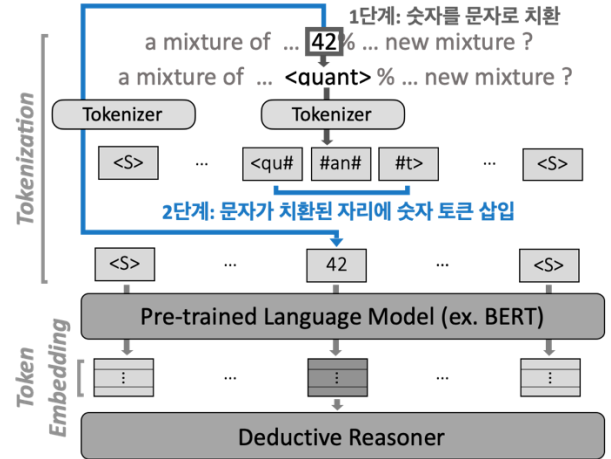
<표 1> 문장형 수학 문제의 예시 (SVAMP)

수학문제	Bryan took a look at his books as well . If Bryan has 56 books in each of his 3 bookshelves , how many books does he have in total ?
------	--

수학문제 (템플릿)	Bryan took a look at his books as well . If Bryan has number0 books in each of his number1 bookshelves , how many books does he have in total ?
정답 수식	Multiply(number0, number1) or Multiply(56, 3)
정답	168



(a) Deductive Reasoner



(b) Explicit Feature Extraction Reasoner

그림 1. (a) Deductive Reasoner (b) Explicit Feature Extraction(EFE) Reasoner 모델 구조

2. 관련 연구

서술형 수학 문제 자동풀이는 1960년대부터[1] 지금까지 활발히 연구되고 있는 분야이다. 규칙 기반 패턴 매칭 알고리즘[2, 3]에서 통계 기반 방법론[4, 5], 트리 기반 방법론[6] 그리고 최근에는 딥러닝 기반 방법론[7, 8]으로 이어지고 있다.

서술형 수학 문제 자동풀이를 위해서는 문제 자체에 명시되어 있는 정보뿐 아니라 사회적으로 약속되어 있거나 유추할 수 있는 내용까지 포함하여 문제를 이해해야 한다. 예를 들어 1시간은 60분이라는 약속을 알고 있어야 시간에 관한 문제를 이해할 수 있다.

최근 인공지능 수학 문제 풀이 모델은 세계 지식(global knowledge)을 고려하여 문제를 풀기 위해 트랜스포머 구조를 활용한 사전학습 언어모델을 사용하여 연구를 진행하고 있다[9, 10].

2.1. 자연어이해 모델을 사용한 수학문제 풀이

선행 연구들에서는 BERT 같이 사전학습된 자연어 이해 모델을 사용하여 수학 문제의 자질 임베딩을 추출하고 추출된 자질을 이용하여 수학 문제의 수식이나 정답을 도출해 왔다[7, 8, 9, 10]. 자연어 이해 모델 중 Deductive Reasoner는 SVAMP 데이터 셋에서 최고 수준의 결과(State-of-the-Art)를 보이고 있다.

Deductive Reasoner 모델

자연어 이해 모델 중 Deductive Reasoner(DR)는 SVAMP 데이터 셋에서 최고 수준의 결과(State-of-the-Art)를 보이고 있다. Deductive reasoner는 (1) Reasoner와 (2) Rationalizer로 이루어진다. Reasoner의 목적은 가능한 피연산자에 대한 모든 순서쌍에 대한 표현 임베딩(representation)을 얻는 것으로, 문제에 등장하는 숫자와 문제에 등장하지 않지만, 문제를 풀 때 활용될 수 있는 숫자에 대해 모든 가능한 순서쌍을 구하고 문제를 풀 때 사용되는 모든 연산자에 대하여 임베딩 벡터를 구한다. 이후 임베딩 벡터에 대해 스코어를 계산하여 가장 높은 스코어의 순서쌍을 이번 step의 수식으로 결정한다. Rationalizer는 이번 step의

수식이 결정되었을 때 기존의 수량 표현을 업데이트 하는 기능을 수행하며, 피연산자로 가능한 모든 임베딩을 Rationalizer 에 통과시켜 업데이트를 수행한다. Deductive Reasoner 는 위 두 과정을 반복하여 정답 수식을 완성한다. 또한 수식의 생성을 멈출 수 있도록 Reasoning 단계에서 임베딩 벡터에 대한 스코어를 계산할 때, 각 벡터에 대한 종료 스코어도 계산하여 종료 스코어가 선택되었을 때 수식의 생성을 멈춘다[10]. 한편, Deductive Reasoner 는 문제에 등장하는 숫자로부터 임베딩을 추출할 때 숫자를 <quant>로 바꾼 뒤 임베딩을 추출하는 방식을 사용하고 있다. 이는 토큰라이저가 생성하는 토큰들은 띄어쓰기 단위가 아닌 바이트 페어 인코딩(BPE) 같이 언어모델을 학습시킬 때 같이 등장했던 바이트들의 묶음 중 높은 빈도로 등장한 토큰이 되기 때문에 숫자와 문자가 겹쳐서 하나의 토큰이 되는 문제를 방지하기 위함이다[8]. 그러나 이로 인해 Deductive Reasoner 는 답을 도출할 때 숫자의 크기에 대한 정보 없이 문제를 풀게 되며, 문장의 논리적 구조만으로 올바른 수식을 계산해야 한다.

3. 방법론

3.1. 실험 데이터셋 및 구현 세부 사항

본 논문에서는 서술형 수학 문제 자동 풀이 모델의 성능을 평가하기 위해 Simple Variations on Arithmetic Math word Problems(SVAMP) 데이터 셋을 사용한다. SVAMP 는 모델의 강건함을 테스트하기 위한 1,000 개의 문제로 이루어져 있다. 문제들은 2,373 문항으로 이루어진 MAWPS 데이터셋과 1,218 문항으로 이루어진 ASDiv-a 데이터셋에서 문제를 선정한 후 9 가지 변형 기법을 적용하여 만들어 놓은 데이터셋이다. 이때 사용한 변형 기법은 문장의 순서 바꾸기, 불필요한 숫자를 문제에 추가하기 등 문제의 난이도는 유지되는 상태에서 정답을 도출하기 위해 필요한 수식을 변경한다. 이 데이터셋을 푸는 모델은 MAWPS 와 ASDiv-a 데이터셋의 모음을 학습 집합으로 사용하고, SVAMP 를 테스트 집합으로 사용하여 성능을 측정한다. 따라서 학습데이터와 난이도는 유사하지만, 문제를 정확하게 이해해야 정답 수식을 추출할 수 있다

[11]. 본 실험에서 모델의 성능은 선행연구와 같이 정답을 정확하게 생성하는지 확인하여 측정한다[10].

3.2. 명시적 자질 추출 방식(Explicit feature extraction)의 제안과 활용 방법

그림 1 은 문제에 해당하는 숫자에 대한 자질을 명시적으로 추출하는 과정으로, 그림 1a 은 Deductive Reasoner 이고, 그림 1b 은 EFE Reasoner 이다. 두 종류의 deductive reasoner 를 활용하기 전, 입력은 크게 tokenization 과 token embedding 두 단계를 거치게 된다. Tokenization 에서는 문제에 등장하는 숫자의 자질 임베딩을 추출하기 위해 서술형 수학 문제는 사전학습 언어 모델의 토큰라이저를 통해 각각의 토큰(Token)으로 분할된다. 두번째로, token embedding 단계에서는 사전학습 언어 모델(Pre-trained Language Model)에 분할된 토큰들을 통과시켜 각 토큰의 임베딩(Embedding) 벡터가 얻어진다. 그림 1a 에 소개된 Deductive Reasoner 에서는 숫자와 문자가 겹쳐서 토큰이 되는 문제를 해결하기 위해 1 단계와 같이 토큰을 유일한 형태로 나타나는 특수한 문자열(ex. <quant>)로 숫자로 대체한 뒤, 토큰화한다. 해당 방식으로 얻어진 임베딩을 활용하면 추론 모델이 숫자의 대소관계를 파악하기 쉽지 않다. 서술형 수학 문제는 여러 숫자들의 대소 관계를 비교하고, 이를 토대로 정답을 도출하는 경우가 많다. 따라서 비교 대상인 숫자들의 대소 관계를 포함한 임베딩을 제공하는 것은 문제를 푸는데 도움이 될 수 있다.

그림 1b 에 소개된 EFE Reasoner 는 본 논문에서 제안하는 모델이며, 2 단계와 같이 치환된 토큰자리에 문제에 등장하는 숫자를 토큰라이저에 따로 통과시켜 대체한다. 그렇게 얻은 문장 전체에 대한 토큰들을 사전학습모델에 통과시키면 숫자의 크기에 대한 정보가 포함된 임베딩 벡터를 얻을 수 있다. 만약 숫자가 여러 개의 토큰으로 토큰나이징되면 숫자 토큰의 첫 번째 토큰의 임베딩과 마지막 토큰의 임베딩을 이어 붙이고(Concatenate) 이를 선형 레이어로 투사(Projection)하여 임베딩을 얻는다. 상수나 연산자의 경우에도 여러 개의 토큰으로 토큰나이징 될 시 첫 번째 토큰의 임베딩과 마지막 토큰의 임베딩을 이어 붙이고 선형 레이어로 투사하여 임베딩을 얻는다.

수학 문제풀이 모델에서 자연어이해 모델을 통해

추출할 수 있는 자질은 피연산자와 연산자에 해당하는 자질 임베딩이다. 피연산자에 해당하는 자질 임베딩은 문제에 등장하는 숫자와 문제에는 등장하지 않지만, 문제를 풀기 위해 필요한 상수(ex. 3.14(원주율), 24(시간))가 있고, 연산자에 해당하는 자질 임베딩은 문제를 풀기 위해 필요한 연산(+, -, x, /등)이 있다.

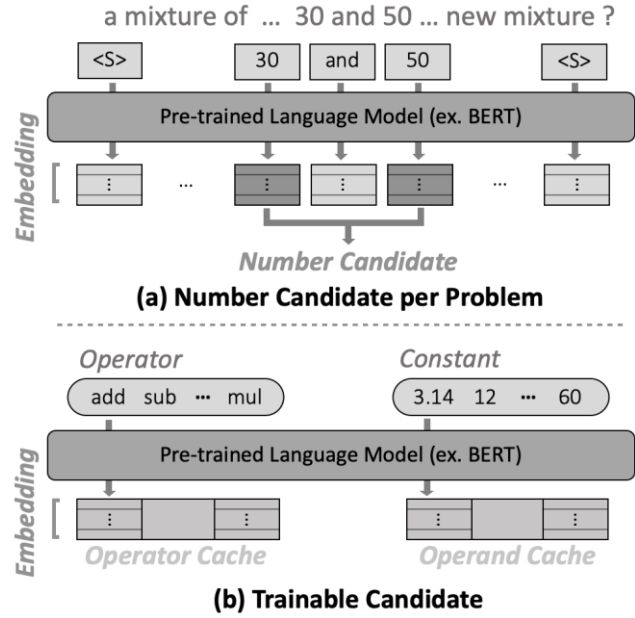


그림 2. (a) Number Candidate per Problem (b) Trainable Candidate

그림 2 는 명시적 자질 추출 방식을 사용하여 피연산자와 연산자에 해당하는 후보군 임베딩(Embedding)을 초기화하는 과정이다. 위의 그림 문제에서 등장하는 숫자의 임베딩을 후보군(Number Candidate)추출하는 과정(Number Candidate per Problem)을 나타내고, 아래의 그림은 연산자(Operator)와 상수(Constant)에 대한 임베딩을 추출하여 연산자/피연산자 캐시(Operator/Operand Cache)에 저장하는 과정을 나타낸다. 먼저, 숫자 후보군은 그림 1 에서 설명한 방식을 활용하여 임베딩을 추출한다. 연산자/상수에 해당하는 임베딩의 경우 문제에 등장하는 숫자와 달리 항상 같은 수를 사용하기 때문에 임베딩 값이 학습될 수 있도록 캐시에 저장하여 학습가능한 연산자 후보군(Trainable Candidate)으로 문제풀이에 활용된다. 이렇게 추출된 연산자 피연산자 임베딩들은 정답 수식 생성 시 스코어를 계산할 때 사용한다.

<표 2> SVAMP 데이터셋의 정답률

Model	Val Acc.
-------	----------

S2S	GroupAttn [12]	21.5
	BERT-BERT[13]	24.8
	Roberta-Roberta[13]	30.3
S2T/G2T	GTS[14]	30.8
	Graph2Tree[15]	36.5
	BERT-Tree[16]	32.4
	Roberta-GTS[11]	41.0
	Roberta-Graph2Tree[11]	43.8
DEDUCTIVE REASONER		
DR	- Roberta-base	47.3
	- Roberta-large	48.9
Explicit Feature Extract Reasoner		
EFE	- Roberta-base	48.4
	- Roberta-large	51.7

4. 실험 결과 방법론

표 2 는 자연어 이해 계열 모델에서 SVAMP 데이터셋의 정답률을 나타낸다. 이때 사용한 베이스라인 모델은 Sequence-to-Sequence(S2S), Sequence-to-Tree(S2T) and Graph-to-Tree(G2T)기반 모델들이다. 우리는 현재 자연어 이해 계열 모델 중 SVAMP 데이터셋에서 최고성능을 보 모델인 Deductive Reasoner(DR)모델에 명시적 자질추출 방식의 적용 유무에 따른 정답률을 비교한다. 본 실험의 결과로 RoBERTa-base 를 사용했을 때는 1.1%의 성능향상을, RoBERTa-large 를 사용한 Deductive Reasoner 모델에 명시적 자질 추출 방식을 도입했을 때 2.8%의 성능향상을 얻어 기존 자연어 이해모델의 최고성능을 갱신할 수 있었다. 이는 자연어 이해 계열의 모델을 사용할 때 숫자토큰을 사용하는 것이 숫자의 대소관계 파악할 수 있는 가능성을 주고, 이러한 가능성은 모델의 정답률 상승에 도움을 줄 수 있었음을 의미한다.

5. 결론 및 한계점

위 실험의 결과로 숫자의 대소 관계 이해에 도움을 줄 수 있는 정보를 추가해 주었을 때, 인공지능 모델의 수학 문제 풀이에서 성능 향상의 가능성을 확인할 수 있었다. Deductive Reasoner 모델에서 임의의 문자열 대신 숫자 토큰을 넣어주었을 때 성능 향상을 확인할 수 있었다. 이를 통해 숫자 토큰을 사용하여 숫자의 대소관계를 파악할 수 있도록 해주는 것이 인공지능 모델이 수학 문제를 해결할 때 도움을 줄 수 있다는 것을 확인할 수 있다. 본 논문에서는 최고 성능 모델인 Deductive Reasoner 를 기반으로 명시적 자질추출 방식을 도입한 EFE Reasoner 를 제시하였다. 향

후 명시적 자질 추출 방식을 다른 자연어 이해계열 모델에 적용했을 때 정답률이 올라가는지 확인하는 추가 연구가 필요할 것이다.

사사

이 성과는 2023 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2020R1C1C1010162)

참고문헌

- [1] D. Bobrow et al., "Natural language input for a computer problem solving system," 1964.
- [2] C. R. Fletcher, "Understanding and solving arithmetic word problems: A computer simulation," Behavior Research Methods, Instruments, & Computers, vol. 17, no. 5, pp. 565–571, 1985.
- [3] M. Yuhui, Z. Ying, C. Guangzuo, R. Yun, and H. Ronghuai, "Frame-based calculus of solving arithmetic multi-step addition and subtraction word problems," in 2010 Second International Workshop on Education Technology and Computer Science, vol. 2, pp. 476–479, IEEE, 2010
- [4] M. J. Hosseini, H. Hajishirzi, O. Etzioni, and N. Kushman, "Learning to solve arithmetic word problems with verb categorization.," in EMNLP, pp. 523–533, 2014
- [5] S. S. Sundaram and D. Khemani, "Natural language processing for solving simple word problems," in Proceedings of the 12th international conference on natural language processing, pp. 394–402, 2015.
- [6] L. Wang, D. Zhang, L. Gao, J. Song, L. Guo, and H. T. Shen, "Mathdqn: Solving arithmetic word problems via deep reinforcement learning," in Proceedings of the AAAI conference on artificial intelligence, vol. 32, 2018
- [7] B. Kim, K. S. Ki, D. Lee, and G. Gweon, "Point to the expression: Solving algebraic word problems using the expression pointer transformer model," in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3768–3779, 2020
- [8] K. S. Ki, D. Lee, B. Kim, and G. Gweon, "Generating equation by utilizing operators: Geo model," in Proceedings of the 28th International Conference on Computational Linguistics, pp. 426–436, 2020.
- [9] J. Zhang and Y. Moshfeghi, "Elastic: numerical reasoning with adaptive symbolic compiler," arXiv preprint arXiv:2210.10105, 2022
- [10] Z. Jie, J. Li, and W. Lu, "Learning to reason deductively: Math word problem solving as complex relation extraction," in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 5944–5955, 2022
- [11] A. Patel, S. Bhattamishra, and N. Goyal, "Are nlp models really able to solve simple math word problems?," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2080–2094, 2021
- [12] J. Li, L. Wang, J. Zhang, Y. Wang, B. T. Dai, and D. Zhang, "Modeling intra-relation in math word problems with different functional multi-head attentions," in Proceedings of the ACL, 2019.
- [13] Y. Lan, L. Wang, Q. Zhang, Y. Lan, B. T. Dai, Y. Wang, D. Zhang, and E.-P. Lim, "Mwptoolkit: An open-source framework for deep learning-based math word problem solvers," arXiv preprint arXiv:2109.00799, 2021
- [14] Z. Xie and S. Sun, "A goal-driven tree-structured neural model for math word problems.," in Proceedings of IJCAI, 2019.
- [15] J. Zhang, L. Wang, R. K.-W. Lee, Y. Bin, Y. Wang, J. Shao, and E.-P. Lim, "Graph-to-tree learning for solving math word problems," in Proceedings of ACL, 2020.
- [16] Z. Li, W. Zhang, C. Yan, Q. Zhou, C. Li, H. Liu, and Y. Cao, "Seeking patterns, not just memorizing procedures: Contrastive learning for solving math word problems," arXiv preprint arXiv:2110.08464, 2021.