Locate-Then-Edit 지식 편집 방법에서의 지식 망각 문제 분석

염시형°[1], 이성희[2], 박성식[1], 김학수[1]
건국대학교 인공지능학과[1], 건국대학교 컴퓨터공학과[2]
{stv10121[1],nlpshlee[2],a163912[1],nlpdrkim[1]}@konkuk.ac.kr

Analysis of knowledge forgetting problem in Locate-Then-Edit knowledge editing method

SiHyeong Yeom°[1], SeongHee Lee[2], SeongSik Park[1], Hark-Soo Kim[1]
Konkuk University, Department of Artificial Intelligence[1],
Konkuk University, Department of Computer Science and Engineering[2]

요 약

지식 편집은 거대 언어 모델에서 잘못되었거나 오래된 지식을 수정하고 새로운 지식을 주입하기 위한 기술이다. 지식 편집의 목적은 기존 모델의 성능을 유지하면서 특정 지식만을 효율적으로 변경하는 것이다. 그 중 Locate-Then-Edit 방법은 인과 매개 분석을 수행하여 모델에서 지식이 저장된 특정 위치를 탐색하고 일부 매개변수만을 편집함으로써 효율적인 지식 편집의 가능성을 보여준다. 하지만, 이와 같은 편집 방법은 동일한 subject에 대한 연속적인 지식 편집을 수행했을 때, 이전에 편집된 지식이 망각된다는 문제가 발생한다. 본 논문에서는 이러한 동일 subject에 대한 동시 및 연속 지식 편집에서 발생하는 문제점을 구체적으로 탐구한다.

주제어: 지식 편집(Knowledge Editing), 지식 망각(Knowledge Forgetting)

1. 서론

최근 GPT(Generative Pre-trained Transformer), LLaMA(Large Language Model Meta AI)와 같은 대형 언어 모델(Large Language Model, LLM)은 다양한 자연어 처리 분야에서 뛰어난 성능을 보여주고 있다. LLM은 대량의 말뭉치로부터 지식을 축적하고 이를 바탕으로 답변을 생성하기 때문에 추상적인 지식 베이스(Knowledge Base, KB) [1]로도 여겨진다. 예를 들어, "대한민국의 수도는"이라는 입력이 주어지면 LLM은 "서울이다"라는 출력을 자연스럽게 생성할 수 있다. 이는 사전 학습 단계에서 문장으로 표현된 다양한 사실적 지식들을 트리플(Subject, Relation, Object) 형태로 LLM이 학습했기 때문이다. 그러나 LLM의 지식은 사전 학습을 완료한 시점에 고정되어 있기 때문에 끊임없이 변화하는 현실 세계의 지식에 대응하기 위해선 LLM을 주기적으로 재학습할 필요가 있다. 하지만, 소규모 지식 업데이트를 위해 모델을 재학습하는 과정은 지속적으로 높은 비용이 소모되기 때문에 비효율적이다.

이러한 상황에서 지식 편집(Knowledge Editing)은 모델의 재학습 또는 미세 조정(Fine-Tuning) 없이 모델에 내재된 지식을 변경하거나 새로운 지식을 추가할 수 있는 기술로서 주목을 받고 있다. [2–8] 특히 ROME [6](Rank-One Model Editing), MEMIT [7](Mass-Editing Memory In a Transformer)과 같은 Locate-Then-Edit 방법은 지식 편집 중에서도 효율적인 방법으로 평가되고 있다. Locate-Then-Edit 방법은 인과 매개 분석(Causal Mediation Analysis) [9]을 통해 사실적 지식이 subject로부터 연상된다고 가정한다. 따라서 subject를 중심으로 특정 계층을 편집함으로써 최소한의 비용으로 효과적인 지식 편집이 가능하다. 그러나 이러한 편집 방법은 동일한 subject에 대한 여러 지식들을 동시에 편집하거나 연속적으로 편집할 경우 문제가 발생할 가능성이 존재한다. 본 논문에서는 subject에 대한 동시적, 연속적 편집 시나리오를 가정한 실험을 통해 Locate-Then-Edit 방법에서 드러나는 지식 편집의 한계점에 대하여 분석한다.

2. 관련 연구

지식 편집은 방법에 따라 크게 메모리 기반 방법 [4,10], 추가 파라미터 기반 방법 [11], Locate-Then-Edit 방법 [6–8], Meta-Learning 기반 방법 [5]으로 분류 할 수 있다 [12]. 이 중 Locate-Then-Edit 방법은 모델의 지식을 상기시키는 부분을 탐색하고 이를 직접적으로 수정하여 근본적인 지식 편집이 가능한 방법이다. ROME [6]은 Locate-Then-Edit 방법을 제시한 가장 대표적인 연구로서 인과 매개 분석을 통해 모델에서 지식이 상기되는 부분을 탐색한다. 결과적으로 subject의 마지막 token에 대한 언어 모델 중간 transformer 계층의 MLP(Multi-Layer Perceptron) 출력이 object 생성에 가장 큰 영향을 미친다는 것을 파악한다. 예를 들어, "대한민국의 수도는"이라는 문장이 입력되면 언어 모델에서 "대한민국"의 가장 마지막 token의 중간 계층 MLP 출력이 "서울"을 생성하는데 가장 큰 영향을 미친다는 것이다. ROME은 이 사실을 기반으로 해당 MLP를 수정하여 원하는 방향으로 모델의 생성 결과를 수정 할 수 있다는 것을 보였다. 이에 영감을 받아 이후의 Locate-Then-Edit 방법들은 동일한 위치의 MLP를 편집하는 것을 중점적으로 다룬다. MEMIT [7]도 마찬가지로 subject 중심의 편집 방법을

제안한다. 그러나 한 번의 하나의 지식만 편집이 가능한 ROME과는 다르게 많은 양의 지식을 한 번에 편집이 가능하도록 편집 방법을 향상시켰다. PMET [8]은 지식을 상기하는데 attention의 역할도 중요하다는 것을 주장하며 attention과 MLP를 함께 편집에 활용하여 MEMIT보다 높은 편집 성능을 보인다. 최근에는 Locate-Then-Edit 방법들이 단기적 편집에서는 뛰어난 성능을 보이지만 장기적, 지속적 편집에서는 치명적인 문제가 발생한다는 분석이 제기되고 있다. 특히 동일한 모델에 대해 지식 편집이 연속적으로 이루어질 경우, 이전에 편집된 내용이 소실되는 지식 망각 현상이 발생한다. 편집이 반복될 수록 이 현상은 점진적으로 심화되며, 특정 횟수에 도달하면 모든 편집된 지식이 사라지는 파괴적 망각 현상 [13]이 확인된다.

본 논문은 subject 중심의 편집 방식을 사용하는 Locate-Then-Edit 방법론의 한계점을 분석한다. 특히, 장기적, 지속적인 편집 환경이 아니더라도, 동일한 subject에 대한 여러 지식을 동시에 또는 연속적으로 편집할 경우 심각한 문제가 발생할 수 있음을 실험을 통해 입증하고자 한다.

3. 본론

3.1 Locate-Then-Edit의 편집 과정

Locate-Then-Edit 방법은 LLM에 내재된 지식을 편집하기 위해서 subject의 마지막 토큰에 해당하는 특정 MLP를 업데이트한다. 여기서 MLP는 그림 1과 같이 'Two-Layer Key-Value Memory'로서 작동하며 subject에 대응하는 key와 object와 연결된 value 사이의 매개변수를 업데이트하여 사실적 관계를 수정한다. 예를 들어, "대한민국의 수도는"이라는 질문에 대해 모델의 답변을 "서울"에서 "부산"으로 편집한다면, subject에 해당하는 "대한민국"으로부터 key를 생성하고 "부산"이라는 object와 연결된 value를 맵핑하는 W_{out}만을 수정한다. 이 때 subject로부터 key를 생성하기 위한 W_{in}은 변경되지 않는다.

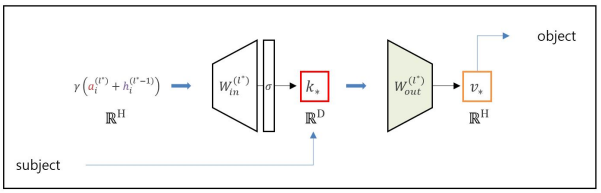

그림 1. subject에 대한 지식 편집 과정

3.2 동일 subject 반복 편집으로 인한 부작용 탐구

본 논문에서는 subject 중심의 편집 방식에서 발생하는 부작용을 탐구하기 위해 두 가지 주요 실험을 진행한다. 첫 번째 실험은 다중 편집 환경에서 여러 지식을 동시에 수정할 때, 단일 배치 내에서 subject가 중복된 데이터의 비율을 점진적으로

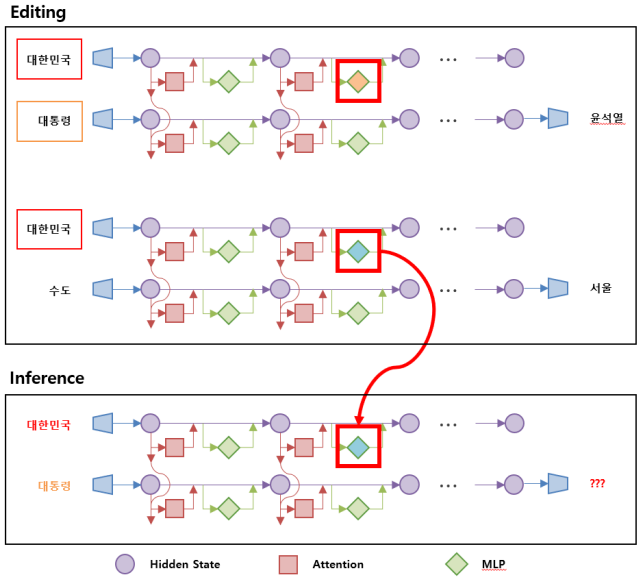

그림 2. 동일 subject에 대한 지식 편집 및 추론 과정

증가시키며 그에 따른 편집 성능 변화를 분석한다. 두 번째 실험은 배치 단위의 연속적인 편집이 이루어지는 환경에서, 각 배치 내에서는 subject가 중복되지 않도록 데이터를 구성하고, 배치 간에는 subject가 중복되도록 설정하여 편집 성능을 평가한다. 예를 들어, 첫 번째 배치가 ("대한민국의 대통령은 윤석열이다", "이순신은 조선의 장군이다", "박지성의 직업은 축구선수이다")로 구성되었다면, 두 번째 배치는 ("대한민국의 수도는 서울이다", "이순신이 만든 배는 거북선이다", "박지성의 직업은 축구 해설가이다")와 같이 구성한다.

그림 2는 입력 문장에 대해 LLM의 답변 생성 과정을 간략하게 시각화한 것이다. 여기서 보라색 원은 은닉 상태(Hidden State), 붉은색 정사각형은 attention 메커니즘, 연녹색 마름모는 MLP를 각각 나타낸다. 주황색 마름모는 "대한민국의 대통령은 윤석열이다"라는 사실적 지식이 편집된 경우이며, 청록색 마름모는 "대한민국의 수도는 서울이다"라는 지식에 대하여 편집된 경우를 나타낸다. 그림에서 볼 수 있듯이, 동일한 subject를 가지는 경우 relation과 object가 다르더라도 지식을 수정하는 데 사용되는 매개변수의 위치가 동일하게 관찰된다. 이를 통해 동일 subject에 대한 지식 편집이 편집 성능에 미치는 영향을 명확히 규명하고 장기적인 지식 편집 환경에서 발생할 수 있는 문제점을 해결하기 위한 중요한 통찰을 제공한다.

4. 실험

4.1 데이터셋

MEMIT에서 사용된 MultiCounterFact(MCF) 데이터셋 [7]을 사용한다. 이 데이터셋은 지식 편집 연구를 위해 설계되었으며, Locate-Then-Edit과 같은 방식에서 전체 모델을 재학

습하지 않고도 특정 사실을 수정할 수 있도록 한다. 일반적으로 (Subject, Relation, Object) 형식의 구조화된 사실로 구성되어 있다. 예를 들어, "프랑스의 수도는 파리이다"와 같은 사실적 정보를 "프랑스의 수도는 로마이다"와 같은 반사실적 (Counterfactual) 정보로 수정할 수 있는지 테스트하는데 사용된다.

본 실험에서 사용된 MCF 데이터셋은 총 20,877개이며, 동일한 subject를 가지는 여러 지식에 대한 편집 성능을 확인하기 위해 subject 별 relation의 수를 분석한다. 표 1은 subject 별 relation 수를 보여준다. MCF에서 단 하나의 relation만 갖고 있는 고유한 subject의 수는 19,366 개로 총 데이터의 약 92.8%를 차지한다. 2개 이상의 relation을 갖고 있는 subject의 수는 총 733 개로 총 데이터의 약 7.2%인 1,1511 개의 데이터가 여기에 포함된다.

relation의 수	subject의 수	총 데이터의 수
1	19,366	19,366 (92.8%)
2	693	1,386 (6.6%)
3	35	105 (0.5%)
4	5	20 (0.1%)

표 1. relation의 수에 따른 데이터 규모

4.2 실험 환경 및 평가 지표

지식 편집에 사용한 모델은 GPT2-XL(1.5B)이며, 편집 알고리즘은 MEMIT을 사용했다. 편집 대상 계층은 [13, 14, 15, 16, 17]이며, 학습률은 5e-1, 시드 값은 7로 MEMIT에서 제공하는 하이퍼 파라미터와 동일하다. 기존 MEMIT에서는 Efficacy, Paraphrase, Specificity와 이들에 대한 조화 평균을 편집에 대한 성능 지표를 사용한다. 하지만 이는 모델의 답변 생성에 대한 성능 평가가 아닌, 단순히 편집 전후의 타겟 object 간의 확률 비교로만 성능을 평가하는 문제가 있다.

본 논문에서는 지식 편집 성능을 보다 명확하게 비교하기 위해, 모델이 생성한 답변이 편집된 object와 일치하는지를 평가 지표로 사용했다. 수식 1은 본 논문에서 사용된 평가 방법을 나타낸다. 예를 들어, "대한민국의 수도는 한양이다"라는 지식을 "대한민국의 수도는 서울이다"로 수정한 경우, 기존 object인 "한양"은 $object_{true}$가 되고, 수정된 object인 "서울"은 $object_{new}$가 된다. 이때 단순히 $object_{new}$의 확률이 $object_{true}$보다 높은지 여부를 평가하는 것이 아니라, $object_{new}$의 확률이 전체 단어 확률 분포에서 가장 높은 값을 갖는지를 기준으로 평가한다. 이러한 평가 방식을 통해 object에 대한 단순 확률 비교가 아닌 모델이 편집된 지식을 올바르게 반영하는지를 검증한다.

$$Acc_{ours} = \begin{cases} 1 & \text{if } \arg\max_i P(token_i) = \text{index of } object_{new} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

4.3 실험 결과

4.3.1 동일 subject의 동시 편집 실험 결과

그림 3은 단일 배치에서 고유한 subject로만 구성된 100개의 샘플 데이터를 기준으로 중복된 subject로 이루어진 데이터의 비율을 0%에서 100%까지 10%씩 증가하며 성능을 비교한 결과이다. subject가 중복된 데이터의 비율이 늘어남에 따라 80%에서 40%까지 다중 편집 성능이 떨어지는 것을 확인할 수 있다. MEMIT은 사실적 지식을 저장하는 데 중요한 역할을 하는 여러 MLP 레이어를 식별하고, 이러한 레이어들에 매개변수 변화량을 분산하여 동시 편집을 가능하게 만든다. 그러나 동일한 subject에 대한 지식을 동시에 편집할 경우, 해당 subject와 관련된 매개변수의 중복 수정이 발생한다. 이로 인해 동일한 subject에 대한 새로운 사실을 추가하거나 수정할 때 기존 편집 내용을 망각한다.

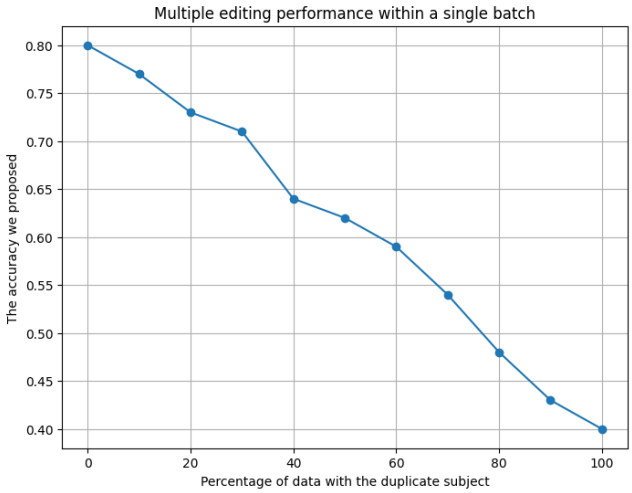

그림 3. 단일 배치 내에서 동시 편집 성능 비교

4.3.2 동일 subject의 연속 편집 실험 결과

그림 4는 한 배치 내에서는 중복된 subject가 존재하지 않도록 설정하고, 다음 배치에서 같은 subject를 편집하도록 실험 데이터를 구성 했을 때의 편집 성능 추이를 보여준다. 비교 결과, 동일한 subject를 동시에 편집하지 않더라도, 매개변수를 복원하지 않고 연속적으로 지식 편집을 수행할 경우, 4.3.1에서 언급한 바와 같이 동일한 subject에 대한 매개변수가 중복으로 업데이트되어 성능 저하를 초래한다. 배치 단위로 실험한 결과

지식 편집 성능이 63%에서 41%까지 하락하는 것을 확인할 수 있다.

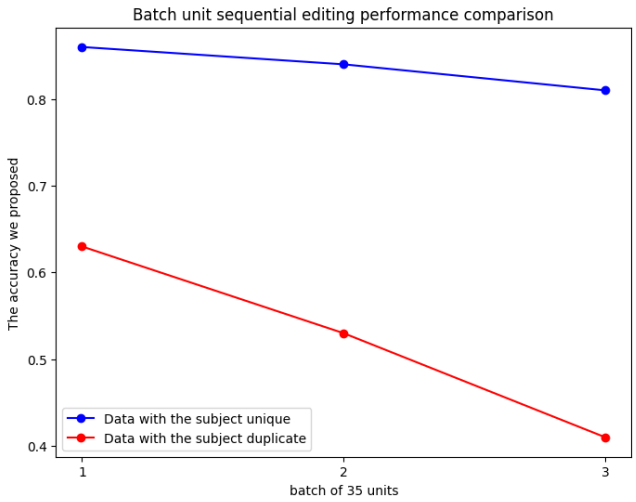

그림 4. 배치 단위에서 연속 편집 성능 비교

5. 결론

본 논문에서는 Locate-Then-Edit 방법의 한계점을 분석하기 위해 여러 지식을 동시에 편집하거나 연속적으로 편집하는 환경에서 동일한 subject가 포함된 경우를 중심으로 연구를 진행하였다. 이러한 환경에서 subject가 중복될 때 발생하는 문제를 실험적으로 검증하고, 지식 편집 과정에서 성능 저하와 지식 망각 현상이 어떻게 발생하는지를 분석하였다. 분석 결과, 동일한 subject에 대한 지식을 편집할 때, 동시 편집과 연속 편집 모두에서 지식 편집 성능이 크게 저하됨을 확인하였다. 이는 기존의 편집 방법이 subject만을 고려하기 때문에 발생하는 문제로, 지식 편집 시 단순히 subject에 국한되지 않고 relation도 함께 반영해야 한다는 점을 시사한다.

즉, 효과적인 지식 편집을 위해서는 subject와 관련된 relation을 함께 고려하고, 매개변수 업데이트 과정에서 과적합을 피할 수 있는 새로운 편집 방식을 도입해야 한다. 이러한 방식은 기존 지식과 새롭게 편집된 지식 간의 충돌을 최소화하면서, 이전에 편집된 정보를 안정적으로 유지하는 것을 목표로 해야 한다. 이를 통해 동일한 subject에 대해 동시 또는 연속적으로 편집하는 상황에서도 성능 저하와 지식 망각 현상을 방지할 수 있다. 이러한 편집 방식은 거대 언어 모델의 장기적인 지식 유지와 지식 편집 안정성을 보장하는 데 기여할 것이다.

감사의 글

이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (RS-2024-00398115, 생성AI가 생성한 결과물의 진실성과 일관성 확보를 위한 기술 연구).

참고문헌

[1] Benjamin Heinzerling and Kentaro Inui. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1772–1791, Online, April 2021. Association for Computational Linguistics.

[2] Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

[3] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

[4] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR, 2022.

[5] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale, 2022.

[6] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022. arXiv:2202.05262.

[7] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass editing memory in a transformer. *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[8] Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572, 2024.

[9] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401, 2020.

[10] Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-

hop questions. *arXiv preprint arXiv:2305.14795*, 2023.

[11] Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron, 2023.

[12] Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore, December 2023. Association for Computational Linguistics.

[13] Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics ACL 2024*, pages 15202–15232, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics.