LLM의 '중간 실종(Lost-in-the-Middle)' 문제와 주의 편향 해결책 요약

1. 서론: LLM은 긴 문맥을 정말 잘 이해하고 있을까?

1.1. 일반적인 통념과 문제 제기

일반적으로 거대 언어 모델(LLM)은 입력된 문맥의 길이가 아무리 길어져도, 정보의 위치와 상관없이 모든 토큰을 균일하게 처리할 것이라고 가정합니다. 100번째 토큰이든 10,000번째 토큰이든 동일한 신뢰도로 다룰 것이라는 믿음입니다.하지만 이러한 가정은 실제와 다릅니다. 이 문제는 "Lost in the Middle" (Liu et al., 2023) 논문을 통해 처음으로 명확히 규명되었습니다. 연구진은 LLM의 성능이 입력 문맥의 길이에 따라 크게 변동하며, 특히 모델이 처리해야 할 핵심 정보가 문맥의 중간 에 위치할 때 성능이 급격히 저하되는 **'중간 실종(Lost-in-the-Middle)'** 현상을 발견했습니다. 이는 GPT-4나 Claude와 같이 긴 문맥 처리에 특화된 최신 모델에서도 공통으로 나타나는 문제입니다.이후 "Found in the Middle" (Hsieh et al.) 논문은 이 문제의 근본 원인이 모델의 '주의(Attention)' 메커니즘에 내재된 편향 때문임을 진단하고, 이를 직접 보정하는 해결책을 제시하며 문제 해결의 실마리를 제공했습니다.

1.2. '중간 실종' 현상의 핵심

'중간 실종' 현상이란, **LLM**이 긴 문맥 속에서 정보의 위치에 따라 정보를 처리하는 능력이 달라지며**,** 특히 문맥의 중간에 위치한 정보를 효과적으로 활용하지 못해 성능이 저하되는 문제 를 의미합니다.이 현상의 핵심적인 관찰 결과는 다음과 같습니다.

- 최고 성능 구간 : LLM은 관련 정보가 입력 문맥의 시작 부분**(**초두 효과**, Primacy Bias)** 또는 **끝 부분(최신 효과, Recency Bias)**에 있을 때 가장 높은 정확도를 보입니다.
- 성능 저하 구간 : 반면, 모델이 문맥의 중간 에 있는 정보에 접근해야 할 때 성능은 눈에 띄게 저하됩니다. 이는 마치 U자 모양의 성능 곡선을 그리는 것과 같습니다.이러한 성능 저하는 모델이 정보를 무작위로 놓치는 것이 아니라, 모델의 핵심 작동 방식인 '주의(Attention)' 메커니즘에 내재된 구조적인 편향에서 비롯됩니다.

2. 근본 원인: U자형 주의 편향 (U-Shaped Attention Bias)

2.1. 주의 편향의 개념

'중간 실종' 현상은 LLM의 내재적인 **'U자형 주의 편향(U-Shaped Attention Bias)'**과 직접적으로 연결됩니다. 이는 모델이 내용의 실제 중요도나 관련성과는 무관하게, 입력 문맥의 시작과 끝 부분에 위치한 토큰에 본능적으로 더 높은 주의(attention) 점수를 할당하는 경향을 의미합니다. 이러한 편향은 트랜스포머 아키텍처 자체의 특성, 사전 학습 데이터의 분포, 그리고 명령어 튜닝 과정의 부산물일 가능성이 높습니다.이는 심리학에서 인간이 목록의 처음과 마지막 항목을 더 잘 기억하는 **'서열 위치 효과(serial-position effect)'**와 매우 유사한 패턴으로, LLM 역시 비슷한 정보 처리 편향을 가지고 있음을 시사합니다.

2.2. U자형 편향의 시각적 이해

"Found in the Middle" 논문에서 제시된 것처럼, RAG(검색 증강 생성) 작업에서의 성능 곡선과 모델의 실제 주의 분포는 거의 동일한 U자형 패턴을 보입니다. 이는 성능 저하가 주의 편향에서 비롯된다는 강력한 증거입니다.| 현상 구분 | 설명 || ------ | ------ || **U**자형 **RAG**

성능 | 정답 문서(Gold doc)가 프롬프트의 시작이나 끝에 있을 때 정확도(Acc.)가 높고, 중간에 있을 때 가장 낮아지는 U자형 곡선을 보입니다. || **U**자형 모델 주의 | 모델의 주의(Model attention) 역시 실제 내용과 관계없이 프롬프트의 시작과 끝 부분에 높게 집중되고 중간 부분에서는 낮아지는 U자형 패턴을 보입니다. |
이처럼 고질적인 주의 편향을 인위적으로 '보정'함으로써 '중간 실종' 문제를 해결할 수 있다는 아이디어에서 'Found-in-the-middle' 방법론이 출발합니다.

3. 해결책: 'Found-in-the-Middle'을 통한 주의 보정

3.1. 핵심 아이디어

'Found-in-the-middle'은 모델의 주의 메커니즘을 직접 수정하여 고질적인 위치 편향을 완화하는 혁신적인 접근법입니다. 이 접근법은 마치 결함이 있는 저울의 무게를 보정하는 것과 같습니다. 먼저, 측정된 전체 무게가 '실제 무게'와 저울 자체의 '오차'의 합이라고 가정합니다. 다음으로, 빈 그릇을 올려 저울의 순수한 '오차' 값만을 측정합니다. 마지막으로, 원래 측정값에서 이 오차를 빼서 진짜 무게를 알아내는 원리입니다.이 방법론의 핵심 작동 원리는 다음과 같은 3단계로 이루어집니다.

1. 주의 분해**:** 모델이 특정 위치에 할당하는 전체 주의 값(Attn)을 내용의 **'실제 관련도(relevance)'**와 위치에 따른 고유한 **'편향(bias)'**의 합으로 간주합니다. (주의(Attn) ≈ 실제 관련도(relevance) + 위치 편향(bias))
2. 편향 측정**:** 내용이 없는 '더미 문서(dummy document)'를 문맥의 여러 위치에 넣어봅니다. 이때 측정되는 주의 값은 내용의 관련도가 배제된 순수한 **'위치 편향'** 값이 됩니다. 이는 저울의 오차를 측정하는 것과 같습니다.
3. 주의 보정**:** 원래 문서의 주의 값에서 해당 위치에서 측정된 '위치 편향' 값을 빼줍니다. 이를 통해 위치의 영향을 제거하고, 내용의 실제 관련도만을 충실히 반영하는 **'보정된 주의(calibrated attention)'** 값을 얻게 됩니다.

3.2. 보정의 효과

이러한 보정 과정을 거치면, 기존에는 강력한 U자형 위치 편향에 의해 그 중요도가 압도당했던 중간 위치의 관련 정보(Gold doc)가 다시 부상하여 주목받을 수 있게 됩니다. 보정 없이는 중간 문서의 주의 값이 양 끝단의 높은 기준 편향 값에 묻혀버리지만, 이 기준 편향을 빼내고 나면 중간에 위치한 진짜 관련 정보의 상대적 주의 값이 두드러지게 나타나는 것입니다. 즉, 위치 편향이라는 '노이즈'를 제거함으로써 모델이 진정으로 내용의 중요도에 집중할 수 있도록 만듭니다.이제 이러한 주의 보정이 실제 모델의 RAG 성능에 어떤 긍정적인 영향을 미치는지 구체적인 결과로 확인할 수 있습니다.

4. 결과: RAG 성능의 극적인 향상

'Found-in-the-middle' 방법론을 적용했을 때, RAG 작업 성능이 다음과 같이 극적으로 향상되었습니다.

- 성능 향상 : 정답 문서가 문맥 중간에 위치하는 가장 어려운 시나리오에서 RAG 작업의 정확도를 최대 **15%** 포인트 까지 향상시킵니다.
- 안정성 확보 : 기존의 불안정한 U자형 성능 곡선을 완화하여, 정보의 위치에 관계없이 모델이 일관되고 안정적인 성능을 발휘하도록 돕습니다.
- 상호 보완성 : 문서를 재정렬(re-ordering)하는 기존의 해결책들과 함께 사용될 경우, 성능을 더욱 향상시키는 상호 보완적인 효과를 제공합니다. 재정렬이 입력 순서를 바꾸는 임시방편적 해결책이라면, 주의 보정은 모델의 내재적 편향을 직접 다루는 근본적인 해결책이기 때문입니다.이 결과들은 주의 보정이 긴 문맥을 다루는 LLM의 근본적인 한계를 극복하는 효과적인 전략임을 명확히 보여줍니다.

5. 최종 요약: 학습자를 위한 핵심 정리

지금까지의 논의를 통해 LLM의 '중간 실종' 문제에 대해 학습자가 반드시 기억해야 할 핵심 사항은 다음과 같습니다.

1. 문제의 본질은 **'위치 편향'**입니다**.** LLM은 긴 문맥을 처리할 때 내용의 실제 중요도와 무관하게 입력의 시작과 끝 부분에 본능적으로 더 많은 주의를 할당하는 **'U**자형 주의 편향'을 내재하고 있습니다. 이로 인해 문맥 중간에 위치한 핵심 정보를 효과적으로 활용하지 못하는 '중간 실종' 현상이 발생합니다.
2. 해결의 핵심은 **'**주의 보정**'**입니다**.** 'Found-in-the-middle'은 이러한 위치 편향을 수학적으로 측정하고 원래의 주의 값에서 제거하는 '주의 보정' 메커니즘을 사용합니다. 이를 통해 모델이 정보의 위치 가 아닌 실제 관련성 에 충실히 집중하도록 만들어 RAG 성능을 획기적으로 향상시킵니다.