2025학년도
석사학위논문

LLM 추천자 시스템에서 사용자 성향에 따른 프롬프트 선택에 관한 연구

지도교수 : 김남기

경기대학교 대학원

컴퓨터과학과

전 장 군

LLM 추천자 시스템에서 사용자 성향에 따른 프롬프트 선택에 관한 연구

이 논문을 석사학위논문으로 제출함

2025년 12월

경기대학교 대학원

컴퓨터과학과

전 장 군

전 장 군의 석사학위논문을 인준함

심 사 위 원 장 _____ 인

심 사 위 원 _____ 인

심 사 위 원 _____ 인

2025년 12월

경기대학교 대학원

목 차

표 목 차

그 림 목 차

감사의 글

경기대학교 김남기 교수님 연구실 소속으로 학부 연구생 2년, 석사과정 2년 간의 시간을 보내며 4년 전을 되돌아보니, 꾸준한 고민과 시행착오의 과정이 하나둘씩 쌓이며 스스로 몰라보도록 바뀐 자신을 보니 감회가 새롭습니다. 김남기 교수님과의 인연을 다시 되새겨보면, 제가 앞으로 마주하게 될 여러 영역에서 길이 상기할 귀감을 되어 주셨습니다. 특히, 연구 지도 과정에서 문제의 중요한 본질과 지엽적인 부분을 스스로 구분하도록 체화할 수 있는 토대를 제공해 주셔서 정말 감사합니다. 또한, 2024년 산업용 결함 검사 과제를 진행하면서, 안준호 교수님께서 제게 주신 많은 통찰은 연구개발에 관한 역량 증진에 많은 배움이 되었습니다.

연구실 생활을 함께한 이상민, 김희찬 선배와 이세훈, 곽윤석, 강성은, 김시헌, 이원석, 김민규, 김석현 후배와 동기인 손현태, 박경민, 박찬솔님께도 감사한 마음을 전하고 싶습니다.

논 문 개 요

거대 언어 모델(large language models, LLMs)이 급속하게 발전하면서, LLM의 별도의 파인 튜닝(fine-tuning) 없이도 뛰어난 제로-샷(zero-shot) 텍스트 처리 및 생성 능력은 응용 범위가 점점 넓어지고 있다. 이에 따라, LLM은 추천 시스템(recommendation system) 분야에서도 사용자 프로필 정보를 바탕으로 개인화된 추천 및 콘텐츠(content) 순위화(ranking) 작업을 수행하는 LLM 추천자(LLM-as-recommender)가 등장한다. 하지만, LLM 기반 추천 시스템 연구는 LLM을 추천자가 아닌 전통적인 추천 모델에 보조적인 증강자(LLM-as-augmenter)로서 작동하는 사례가 대부분을 차지한다. 그러나, LLM 증강자는 역할은 기존 추천 모델과 결합하여 이루어지기 때문에, 전통적인 추천 모델에서 발생하는 콜드 스타트(cold-start) 문제가 여전히 존재한다. 반면, 본 논문에서는 방대하게 내재된 콘텐츠 이해력을 통해 콜드 스타트 문제를 극복할 수 있는 LLM 추천자 방식 순위화 추천 시스템에 초점을 둔다.

LLM을 추천자로서 활용하는 연구는 현재 초기 단계이며, 환각(hallucination)에 대한 우려나 모델 자체에 내재된 암묵적인 편향(bias)이 추천 시스템에서 주요 문제로 지적된다. 그 중, 프롬프트에 사용자 민감 속성(sensitive attributes, SST) 정보를 입력할 때 LLM이 인기 아이템 위주의 추천을 하도록 유도되어 사용자의 개인화된 아이템 선호 방향을 고려하지 않는 점이 문제가 된다. 이는 모든 사용자에 대해 인기 아이템 선호 성향과 무관하게 일괄적인 프롬프트 기반 추천이 특정 사용자 그룹에는 오히려 추천 품질이 하락하는 결과를 유도한다. LLM이 사전에 학습한 콘텐츠가 대부분 인기 아이템에 편중된 점이 원인 중 하나이다. 이에 따라, 본 논문에서는 사용자 프로필의 아이템 상호작용 기록을 기반으로 사용자의 인기 아이템 선호 성향을 예측해서, 사용자 성향에 따

라 프롬프트에 사용자 민감 속성 정보를 선택적으로 포함하는 프롬프트 엔지니어링을 제안한다. 제안하는 방법은 실험을 통해 다양한 인구통계 그룹에서 LLM의 추천 순위 예측 성능을 개선한다.

제 1 장 서 론

제 1 절 연구 배경 및 목적

트랜스포머(transformer) [1] 아키텍처의 등장으로 생성형 AI는 양적 측면과 질적인 측면 모두에서 범용 문제 해결 능력을 갖춘 거대 언어 모델(large language models, LLMs)로 발전하여 연구와 산업에서 핵심적인 도구가 되었다[2-5]. 다만, 추천 시스템(recommendation system) [6-10] 연구 환경에서는 LLM이 사용자의 요청에 직접 응답하는 추천자(LLM-as-recommender) [11]가 아닌, 기존 추천 모델의 성능을 보강하는 증강자(LLM-as-augmenter) [12-13]의 역할로 다뤄진다. LLM 증강자는 추천 시스템의 파이프라인에서 보조적인 역할로 제한된다. LLM 증강자는 사용자와 아이템 정보에 대한 자연어 형식 텍스트에서 특징을 추출하거나 부족한 특징을 합성하는 의미론적 임베딩(semantic embedding) [12] 작업이나, 재순위화(re-ranking) [13] 작업을 통해 기존 추천 모델의 성능 향상에 기여한다. 그러나, 이 작업에서 여전히 핵심 추천 모델로 인한 콜드 스타트(cold-start) 문제가 발생한다. 반면, 사전 학습된 콘텐츠에 대한 정보로 코드 스타트를 극복하는 LLM 추천자의 직접 활용은 단순히 딥러닝 기반 추천 모델의 성능 개선과 달리, 추천 시스템의 패러다임 전환 과정에서 중요한 기반이 될 수 있다. 다만, 현시점에서 LLM 추천자는 편향(bias) [14-22] 추천 문제와 같이 몇 가지 해결해야 할 점이 존재한다. 그 중, 본 논문에서는 프롬프트에 의해 LLM의 인기 아이템 위주의 추천이 유도될 때, 추천 품질 저하를 개선하는 방법을 제안한다.

제 2 절 연구 내용

LLM의 추천은 동일한 프롬프트 구조에서 인기 아이템과 비주류 아이템에 대한 추천 순위 예측의 정확도가 다른 경향을 보인다. 이러한 경향은 방대한 텍스트 데이터 학습 과정에서 데이터가 인기 아이템 위주로 편중이 된 점에서 비롯되는 특징이다. 그 이유는 LLM이 인터넷에서 사전 학습한 인기 아이템과 비주류 아이템의 정보가 웹 문서에서 노출, 검색, 인용되는 빈도가 다르기 때문이다. 이러한 차이는 프롬프트의 구조에서 사용자의 인종, 성별, 나이와 같은 민감 속성(sensitive attributes, SST) [14-15]이 입력될 때, LLM이 더욱 특정한 아이템에 집중하도록 유도한다. 기존 추천 시스템에서 SST와 관련된 인구통계 정보를 추천 모델이 잠재 요인으로 활용한 것과 같이, LLM도 사전 학습된 지식이 SST 입력에 대해 인기 아이템 위주의 응답을 유발할 수 있기 때문이다. 특히, 개인적 선호가 다른 모든 사용자에게 동일한 프롬프트 구조를 적용할 경우, LLM은 SST 입력에 따라 특정 사용자 그룹의 추천 품질을 저하시킬 수 있다.

본 논문에서는 사용자 측면에서도 일반적으로 인기 아이템에 대한 선호가 높은 숏헤드 사용자(short head users)와 해당 인구통계 그룹에서 비주류인 아이템을 선호하는 롱테일 사용자(long tail users) 그룹으로 구분될 수 있다고 가정한다. 이러한 사용자의 인기 아이템 선호 성향에 기반할 때, 자신의 인구통계 그룹에서 평균적으로 인기 있는 아이템을 선호하지 않는 롱테일 사용자에 대한 추천은 SST 입력이 특정 성별이나 연령대에서만 평균적으로 인기 있는 아이템 위주의 추천 효과를 유발할 수 있다. 반면, 숏헤드 사용자에 대한 SST 입력은 LLM이 고려해야 할 방대한 인기 콘텐츠 내용 중 SST에 기반해 그 추천 범위를 더욱 명확하게 한다. 따라서, 본 논문에서는 사용자의 과거 상호작용 기록을 기반으로 사용자의 인기 아이템에 대한 선호 성향을 예측하고, 이를 기반으로 프롬프트를 생성하는 선택적 SST 입력 프롬프트 엔지니어링을 새롭게 제안한다.

제 2 장 관련 연구

제 1 절 전통적인 추천 시스템 연구 방향

전통적인 추천 시스템의 방법론은 크게 콘텐츠 기반 필터링(content-based filtering, CB)과 협업 필터링(collaborative filtering, CF)으로 분류된다[7-9]. CB는 사용자가 과거에 선호했던 아이템의 콘텐츠(content)를 분석하여 유사한 내용과 속성을 가진 다른 아이템을 추천하는 방식이다. 이는 새로운 아이템을 추천할 수 있으나 사용자의 선호도 변화를 잘 포착하지 못하고, 사용자 프로필 정보가 부족할 때 추천이 어려운 콜드 스타트 문제에 취약한 단점이 있다. CF는 사용자에게 유사한 성향의 기존 사용자들이 선호했던 아이템을 추천하는 방식이다. 이러한, CF는 구현이 간단하나 알고리즘이 다루는 데이터 희소성(sparsity)에 따라 유사도 계산이 부정확해질 수 있다. CB 방식의 경우, 서로 다른 도메인의 아이템을 추천할 때마다 새로운 도메인 특성 기반 방법이 필요했다. CF 방식의 경우, 사용자 간의 선호 유사도만을 고려하는 확장성으로 인해, 기존 CF의 단점을 보완하기 위한 모델 학습 기반 CF가 많이 연구되었다.

모델 기반 CF는 특히 행렬 분해(matrix factorization, MF) [23] 기반 접근법과 딥러닝의 활용이 중점적으로 연구됐다. MF는 사용자-아이템 상호작용을 나타낸 행렬을 저차원의 잠재 요인 행렬로 분해하여 사용자와 아이템의 특성을 학습하는 접근법이다. 이러한 MF의 특성 학습에는 사용자-아이템 간의 복잡하고 비선형적인 관계를 포착할 수 있는 신경망(neural networks)도 도입되었다. 특히, 사용자-아이템 상호작용을 그래프 구조로 모델링하는 그래프 신경망(graph neural networks, GNNs)과 그래프 콘볼루션 네트워크(graph convoluation networks, GCNs)가 활발히 연구되었다[6, 10]. 그런데, 이러한 CF 방식 모델들은 아이템을 기본적으로 추상적인 ID로 간주하며, 상호작용이 없는 새로운 아이템을 여전히 콜드 스타트 문제로 인해 추천하기 어려운 한계가 있다. 몇몇 GCN 기반 CF는 아이템의 세부 속성을 일부 활용하지만, 해당 속성 이름의 임베딩을 독립된 특징(features)으로만 간주하고 해당 단어가 내

포하는 문맥적 의미를 이해하지 못한다. 이에 대한 단점을 보완하기 위해 LLM 증강자 패러다임이 등장했다.

제 2 절 LLM 증강자 기반 추천 시스템

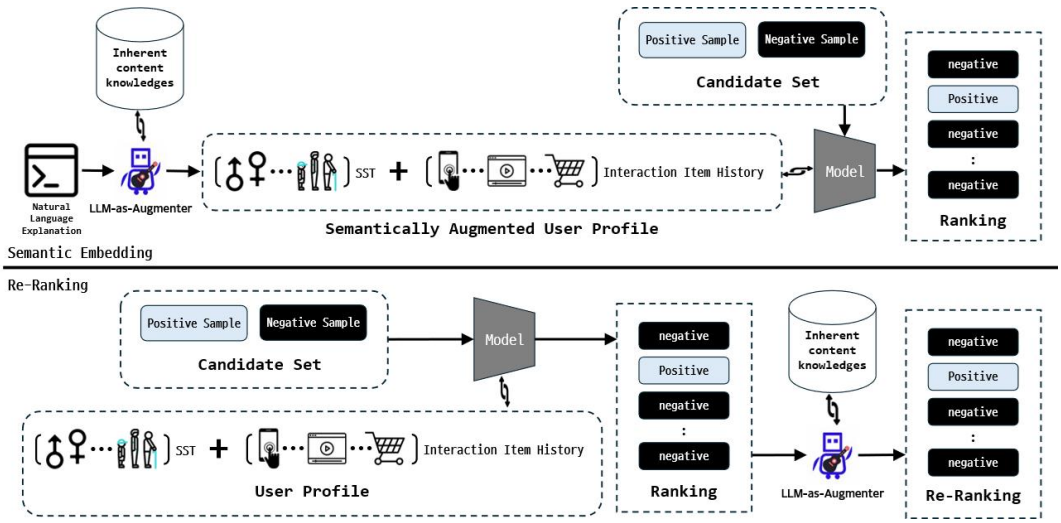
방대한 텍스트로 사전 훈련된 LLM의 등장으로, 기존 CB 방식의 약점인 특정 도메인 특징에 의존적인 사용자 프로필 처리 방식이 범용 도메인 특징의 사용자 프로필을 자연어 입력으로 처리할 수 있게 되었다. 이에 따라, 방대한 콘텐츠 이해력을 내재한 LLM은 CF 방식 모델과 결합해 추천 시스템에서 LLM 증강자의 역할을 수행하게 되었다.

LLM 증강자의 역할은 그림 1과 같이 크게 의미론적 임베딩과 재순위화로 나눠진다. 의미론적 임베딩은 초기 LLM 증강자 연구에서 등장했다. 이 작업은 LLM이 자연어 형식의 아이템 설명, 사용자 리뷰 등의 텍스트 데이터를 분석해 아이템의 핵심 속성, 감성 등의 특징을 추출하거나, 외부 지식을 통해 누락되거나 부족한 속성 및 특징을 합성해서 CF 방식 모델의 임베딩 공간으로 주입한다. 재순위화는 1차적으로 기존 CF 방식 모델이 빠르게 추천 아이템을 순위화(ranking) 작업으로 아이템을 선별하면, 이를 입력받은 LLM이 내재된 아이템의 의미론적 이해를 기반으로 최종 추천 순위를 재조정하는 작업이다.

하지만, CF 방식 모델의 LLM 증강자 활용 접근법은 몇 가지 한계를 가진다. 의미론적 임베딩의 경우, LLM이 메타데이터로부터 추출한 의미론적 정보를 기존 추천 모델에 전달하기 위해서는 고정된 차원의 벡터로 압축해야 하는 병목 현상이 발생한다. 이러한 임베딩을 기반으로 추천을 수행하는 CF 방식 모델은 특징이 풍부(enrichment)해졌지만, 여전히 해당 특징이 내포하는 문맥적 의미를 이해하지 못한다. 또한, 이 방식은 여전히 CF에서 콜드 스타트 문제가 발생한다. 재순위화의 경우, 1차적으로 콜드 스타트 문제를 가진 CF 방식 모델이 선별한 제한된 후보군 내에서만 동작한다. 검색 기반 증강 생성(retrieval-augmented generation, Rag)을 활용할 경우, 일부 새로운 아이템을 추천할 수도 있지만[13], 이는 평가를 어렵게 만들며, LLM이 사전 학습되는 과정에서 대중적으로 널리 알려진 아이템에 대해 더 많은 정보를 학습했기 때문에 추천 다양성이 저해된다.

이러한 LLM 증강자의 도입은 기존 모델 학습 기반 CF의 한계를 넘어선 결과를 보이기 때문에, 추천 시스템에서의 LLM 활용이라는 중요한 기여를 하였다. 하지만, LLM이 사용자의 요청에 직접 응답하는 추천자가 아닌 증강자로

써의 역할로 다뤄질 때, 이는 보조적인 역할로만 제한되어 콜드 스타트 문제가 해결되지 않는다. 이에 따라, 본 논문에서는 방대한 콘텐츠 이해력을 온전히 활용해서 콜드 스타트 문제를 극복하는 LLM 추천자 방식 추천 시스템에 중점을 둔다.



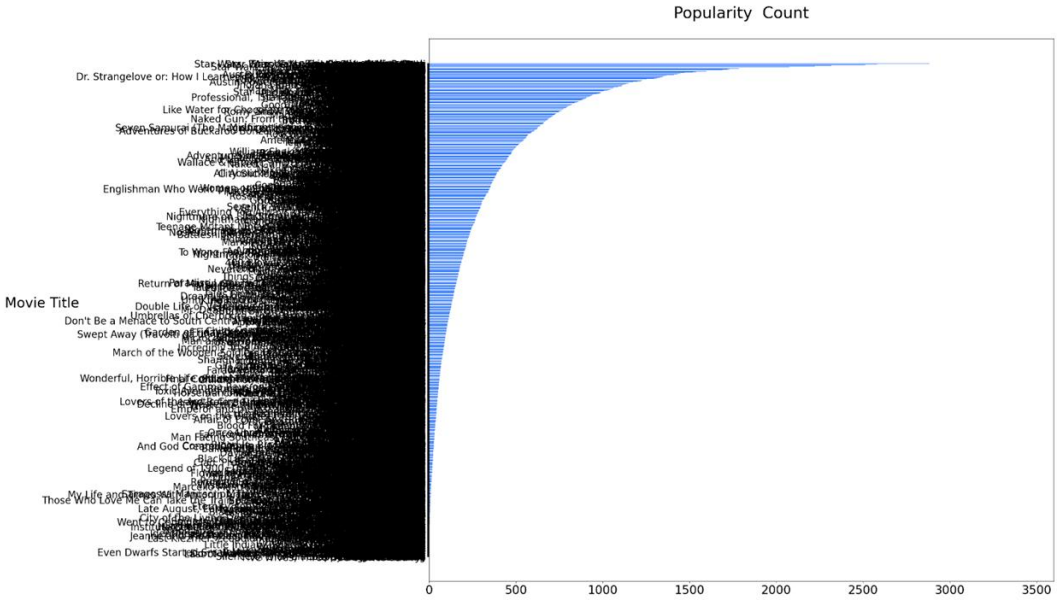
<그림 1> 지금까지 연구된 LLM 증강자의 2가지 역할

제 3 장 선택적 SST 프롬프트 엔지니어링

본 논문에서는 LLM 추천자 시스템의 추천 순위 예측 능력을 최적화하기 위해, 기존 LLM이 프롬프트에 의해 인기 아이템 위주 추천이 유도되는 점을 고려한 선택적 SST 프롬프트 엔지니어링을 새롭게 제안한다. 선택적 SST 프롬프트 엔니지어링은 사전에 계산된 아이템 인기도 테이블을 참조해 사용자 프로필을 분석하고, 숏헤드 사용자와 롱테일 사용자의 선호에 맞는 SST 프롬프트를 생성하는 방법이다. 선택적 SST 프롬프트 엔니지어링을 수행하여 주어진 프롬프트 내의 정보만을 활용하는 LLM을 호출하면, LLM 추천자는 추천 후보군의 아이템을 순위화해서 텍스트 출력으로 제공한다. 이때, LLM의 출력 결과에서 불필요한 마크다운 기호, 순위 번호, 괄호 설명 등을 제거하기 위해, 문자열 정규식 파싱(regex parsing)으로 후보군 아이템 정보를 추출하는 작업이 후처리로 수행된다.

$$popularity(i) = \frac{|\{u \in U \mid (u,i) \in R\}|}{\max_{j \in I} |\{u \in U \mid (u,j) \in R\}|} \quad \text{(수식 1)}$$

제안하는 방법에서 숏헤드 사용자와 롱테일 사용자를 구분하기 위해서는 인기 아이템 구분이 선행되어야 한다. 이에 본 논문에서는 아이템의 인기도를 사전 계산했다. 사용자의 인기 아이템 선호도 추정을 위해 사용된 인기도 테이블 *popularity*는 평가 데이터 집합의 전체 아이템을 기반으로 수식 1에 따라 계산된다. 그림 2는 MovieLens-1M 데이터 집합의 전체 아이템별 상호작용 사용자 수를 나타낸 히스토그램이다. 그림 2에서는 일부의 아이템만 상호작용 횟수가 높고, 대부분의 아이템은 상호작용 횟수가 낮아 파레토 법칙을 따르는 모습을 보인다. 인기도 테이블 *popularity*은 그림 2의 히스토그램과 같은 아이템별 상호작용 수치에 대해 수식 1에 따라 사전에 정규화 계산을 수행해서 구한다.



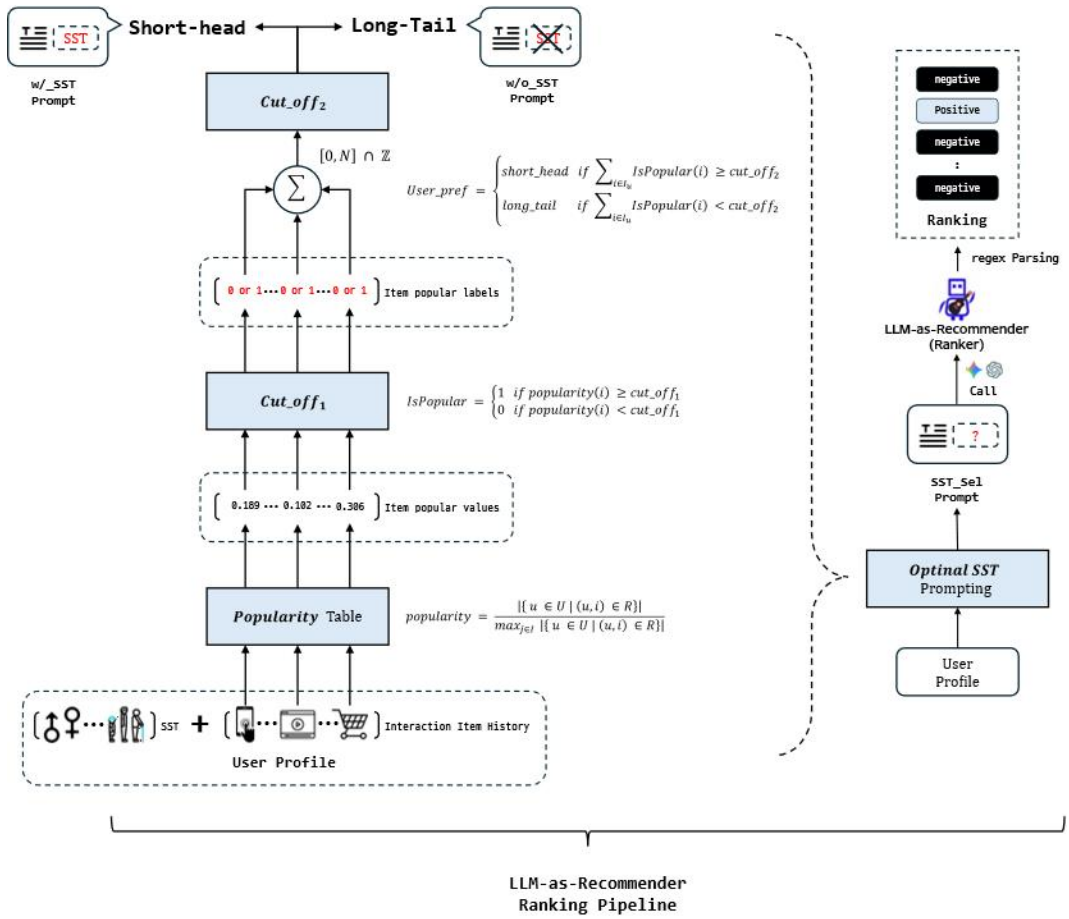
<그림 2> MovieLens-1M 데이터 집합의 전체 아이템별 상호작용한 사용자 수를 나타낸 히스토그램

선택적 SST 프롬프트 엔지니어링은 사용자의 성별, 나이와 같은 SST 정보를 사용자 측면 아이템 선호 성향에 따라 프롬프트에 포함 여부를 결정하는 방법이다. 기계학습 방식의 추천 모델에 SST가 입력 특징 중 하나로 포함될 때, 해당 속성은 모델의 잠재 공간 내에서 사용자가 속한 인구통계 그룹의 잠재 요인의 추천 결과와 유사한 추천을 하도록 유도된다. 특히, LLM에서는 사용자나 아이템의 세부 속성에 대한 의미론적 이해를 내재하기 때문에, SST 입력은 더욱 모델이 해당 그룹에 인기 있는 아이템을 추천하도록 유도한다. 이러한 현상은 롱테일 사용자에 대해 부정적으로 작용할 수 있다. 이를 완화하기 위해, 사용자 프로필 정보를 통해 숏헤드 사용자와 롱테일 사용자를 추정하고, 사용자의 선호도에 따라 SST를 프롬프트에 선택적으로 포함하는 사전 작업이 필요하다.

$$IsPopular(i) = \begin{cases} 1 & \text{if } popularity(i) \geq cut\_off_1 \\ 0 & \text{if } popularity(i) < cut\_off_1 \end{cases} \quad \text{(수식 2)}$$

$$User\_pref(I_u) = \begin{cases} short-head & \text{if } \sum_{i \in I_u} Ispopular(i) \geq cut\_off_2 \\ long-tail & \text{if } \sum_{i \in I_u} Ispopular(i) < cut\_off_2 \end{cases} \quad \text{(수식 3)}$$

숏헤드 사용자와 롱테일 사용자를 구분하기 위해, 제안하는 SST 프롬프트 엔지니어링에서는 사용자 프로필의 사용자 상호작용 아이템 기록을 활용한다. 제안하는 방법에서는 사용자와 가장 최근에 상호작용한 N개의 아이템 중 몇 개의 아이템이 인기 아이템이지 측정해서 사용자 선호 추정 기준으로 둔다. 사용자 u의 아이템 선호도를 측정하기 위해, 사용자 프로필의 최신 상호작용 아이템 집합 I_u에 대해 아이템 인기도를 측정(수식 2)한다. 수식 2에서는 사전에 계산한 인기도 테이블 *popularity*에 대해 아이템의 이름을 쿼리로 사용해서 실수 값의 아이템 인기도를 구한다. 이때, 수식 2에서는 아이템 인기도 값에 대해 인기 아이템 여부를 판단하는 임계값 파라미터 cut_off_1을 도입했다. N개의 사용자 프로필 아이템 집합 I_u에 대해 인기 아이템을 찾으면, 사용자의 아이템 선호도를 인기 아이템 개수를 기준으로 추정(수식 3)한다. 이를 위해, 수식 3에서는 인기 아이템 개수에 대해 사용자의 인기 아이템 선호 성향을 판단하는 임계값 파라미터 cut_off_2을 도입했다. 선택적 SST 프롬프트 엔지니어링의 수행되는 파이프라인은 그림 3에 작동 방식을 추상화했다.



<그림 3> LLM 추천자 방식 순위화 파이프라인

제 4 장 실 험

제 1 절 실험 데이터

본 논문에서는 제안하는 LLM 추천자 방식 순위화 방법의 평가 벤치마크로서 MovieLens-1M [24]과 Last.fm-1K 데이터 집합을 사용했다. MovieLens, Last.fm 데이터 집합은 명시적 피드백과 암묵적 피드백 정보를 모두 포함하고 있어, 많은 영화 추천 시스템 연구에서 벤치마크로 활용된다. 기존의 기계학습 기반 추천 시스템 평가를 수행할 때는 학습 또는 파인 튜닝(fine-tuning)을 수행해야만 해서, 실험 데이터 집합을 학습 데이터와 테스트 데이터로 분할해서 사용했다. 본 논문에서의 추천 모델인 LLM 추천자는 사전 학습된 지식에 기반한 실제 서비스 환경과 동일한 조건을 순위화 평가 기준으로 둔다. 그래서, 실험을 위한 별도의 학습 데이터를 두지 않는다.

제 1 항 사용자 인구통계 그룹

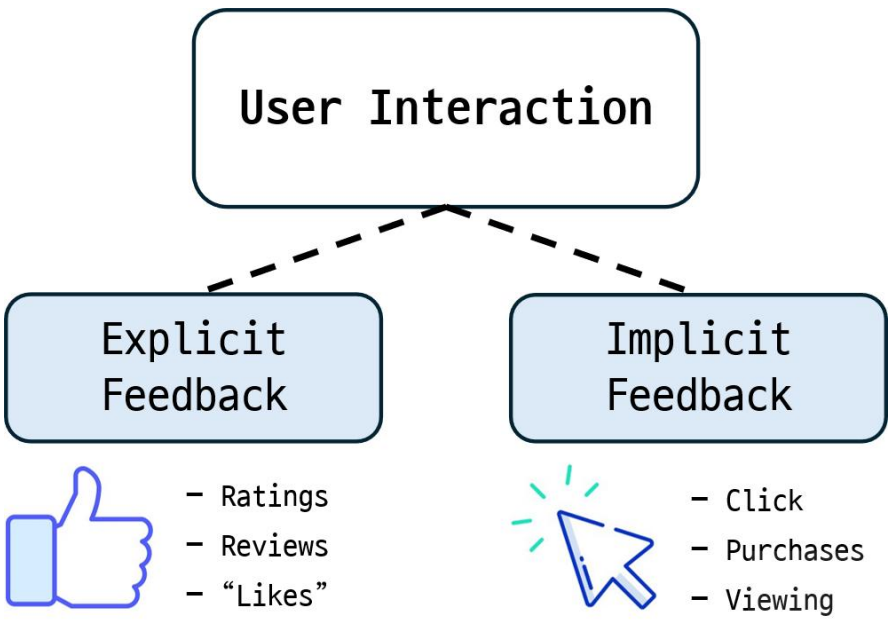
다만, MovieLens 데이터는 소수 인구통계 그룹과 다수 인구통계 그룹 간 사용자 수의 분포가 매우 불균등함을 표 1에서 보인다. 이러한 데이터 불균형은 특정 그룹에 대한 평가지표의 과대평가나 과소평가를 유발한다. 이에 대해, 본 논문에서는 각 교차 그룹이 전체 평균 지표에 기여하는 영향을 조정하기 위해, 다수 그룹에 대한 임계값 기반 언더 샘플링(under sampling)을 테스트 세트에 적용했다. 샘플링 임계값은 표 1에서 확인할 수 있는 가장 작은 교차 그룹의 사용자 수 $N_{\min}$을 기준으로 최대 불균형 비율을 1:3으로 제한하는 값($3 \times N_{\min}$)으로 설정했다.

<표 1> MovieLens-1M 데이터 집합의 인구통계 그룹별 유저 수

교차 인구통계 그룹		사용자 수	
성별	연령대	전체 사용자 수	최종 사용자 수
남성	0-17	144	144
남성	18-24	791	228
남성	25-34	1529	228
남성	35-44	841	228
남성	45-49	359	228
남성	50-55	345	228
남성	56+	273	228
여성	0-17	**76 ($N_{\min}$)**	76
여성	18-24	293	228
여성	25-34	551	228
여성	35-44	334	228
여성	45-49	182	182
여성	50-55	142	142
여성	56+	94	94

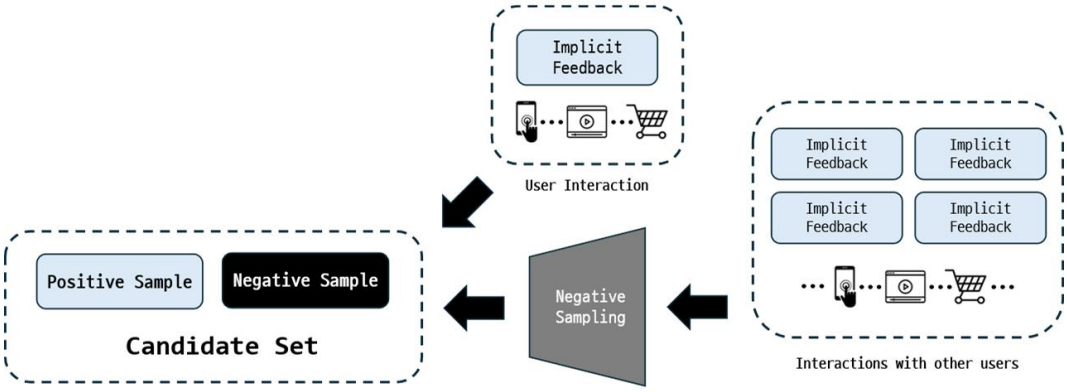
제 2 항 암묵적 피드백(Implicit Feedback)

초기 추천 시스템 연구는 모델이 사용자가 아직 평가하지 않은 아이템의 평점을 예측하는 명시적 피드백(explicit feedback) 환경을 가정했다. 하지만, 2010년 후반에 들어 플랫폼 서비스가 규모가 방대해지면서, 대부분의 사용자는 평점을 남기지 않아 데이터가 희소해졌다. 또한, 평점을 남기는 사용자들은 해당 아이템에 대해 극단적인 호불호를 가진 경우가 많아, 수집된 소수의 평점 데이터가 전체 사용자의 선호를 항상 대표하지는 않는 문제를 가진다. 이러한 한계로 인해, 추천 시스템 연구 패러다임은 평점 예측에서 암묵적 피드백을 활용하는 방향으로 전환되었다. 본 논문에서도 실험 데이터를 사용자가 직접 명시하지 않았지만, 기록된 행동을 통해 선호를 추론하는 암묵적 피드백(implicit feedback) 환경으로 가정하고 별도로 평점 정보를 활용하지 않는다. 암묵적 피드백은 그림 4와 같이 전체 사용자에 대해 수집한 클릭, 시청, 구매 정보로 다뤄지기 때문에, 실제 선호에 대한 대표성과 비즈니스 목표와의 연관성이 더 높다.



<그림 4> 명시적 피드백과 암묵적 피드백

현 추천 시스템 연구에서는 사용자에게 특정 아이템이 노출되었을 때 발생한 암묵적 피드백을 모두 positive 신호로 간주한다. 그리고, 추천 시스템에서는 사용자가 실제로 상호작용한 positive 아이템을 negative 아이템과 비교해서 상위 K개의 추천 목록에 얼마나 포함되었는지 평가하는 Top-K 랭킹 지표 개선에 초점을 둔다. 하지만, 암묵적 피드백은 positive 신호만 존재하고, negative 신호는 존재하지 않는다. 그래서, 명시적 negative 신호가 없는 문제를 해결하기 위해, 사용자가 상호작용 하지 않은 아이템에서 negative 아이템으로 선택해 활용하도록 그림 5와 같이 네거티브 샘플링(negative sampling) 기법이 도입됐다.



<그림 5> 네거티브 샘플링

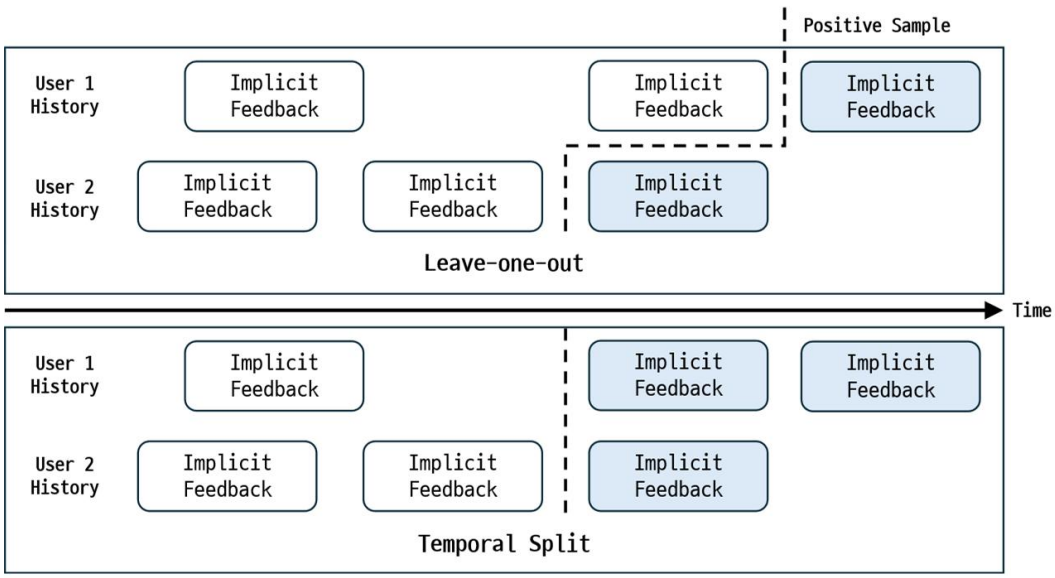
제 2 절 실험 평가 방식

암묵적 피드백 환경에서 네거티브 샘플링을 수행하는 추천 시스템 연구에서는 positive 샘플과 함께 사용자에 대한 상호작용이 없는 아이템에 negative 샘플을 인위적으로 선택하여 후보군(candidate set)을 구성한다. 후보군의 positive 샘플도 추천 시스템의 평가 목적에 따라 temporal split와 leave-one-out 평가 방식에 맞게 구성할 수 있다. temporal split 방식과 leave-one-out 방식에 대한 직관적 이해를 돕기 위해서, 그림 6에 두 방식을 비교하여 나타냈다.

제 1 항 후보군(Candidate Set) 구성

Temporal split는 사용자와 아이템의 상호작용 중 특정 시점을 기준으로 아이템을 분할해서, 미래의 상호작용에 해당하는 일부를 모델이 순위화할 positive 샘플로 두는 평가 방식이다. 분할된 나머지 과거 상호작용 기록은 LLM이 참조할 사용자 프로필로 구성이 된다. Temporal split 방식은 여러 positive 샘플에 대해 순위화 작업을 수행하는데, LLM의 경우 환각(hallucination)으로 인해 일부 positive 아이템의 생성이 누락 될 수 있다. 그런데, 일부의 positivie 아이템이 환각으로 누락됐을 때, 나머지 positive 샘플의 Top-K 추천 목록 순위가 높게 예측되면 성능 평가의 어려움이 생긴다.

그래서, 본 논문의 실험에서는 LLM을 추천자 방식 순위화 평가를 위해 과거에 추천 시스템 연구 초기에 사용된 leave-one-out 평가 방식을 대안으로 적용한다. Leave-one-out은 사용자별로 상호작용이 존재한 데이터 중 마지막에 상호작용한 아이템만을 positive 샘플로 두고 직전에 상호작용한 N개의 아이템으로 사용자 프로필을 구성하는 방식이다. Leave-one-out 방식은 단 하나의 positive 샘플을 가지기 때문에, Top-K 추천 목록에 positive 아이템이 포함되지 않은 경우와 환각이 발생해 생성에 실패한 경우를 모두 명확한 추천 실패로 간주하는 정량 평가가 가능하다. 과거에는, leave-one-out 방식이 학습 관점에서 특정 사용자가 마지막으로 상호작용한 아이템이 다른 사용자가 과거에 상호작용한 아이템에 포함될 수 있어 성능이 과대 평가되는 시간 누수 문제가 존재했다. 그러나, 시간 누수 문제는 모델을 평가하기 이전에 파인 튜닝을 수행해야만 할 때 문제가 되며, LLM을 호출해 주어진 프롬프트 내의 정보만을 활용하는 LLM 추천자 방식을 평가할 때는 다르게 적용된다. 추천 시스템에서 시간 누수에 의한 성능 향상이 문제 되는 이유는 실제 서비스 환경에서 재현 불가능한 점이 원인이다. 사전 학습된 방대한 콘텐츠 지식을 가진 LLM 추천자는 시간적 제약을 두지 않는 leave-one-out 평가 방식이 실제 서비스 환경에서 성능을 더 정확하게 반영할 수 있기 때문이다.



<그림 6> Leave-one-out과 Temporal Split 평가 방식의 후보군 구성 방법 차이 비교

제 2 항 평가지표

제안하는 LLM 추천자 방식 순위화 성능 평가를 위한 평가지표로는 Hit-Rate@K(HR@K)와 normalized discounted cumulative gain@K(NDCG@K)를 사용한다. HR@K는 추천 목록의 상위 K개의 항목에 사용자가 관심을 보인 positive 아이템이 포함되었는지 확인하는 지표이다. NDCG@K는 추천 목록의 상위 K개의 항목에 사용자가 관심을 보인 positive 아이템이 얼마나 상위 순위에 포함되었는지 측정하는 지표이다. 본 논문에서의 LLM 추천자 방식 순위화는 단 하나의 아이템만을 positive 샘플로 사용하는 leave-one-out 방식을 따르기 때문에, 기존의 HR@K와 NDCG@K를 수식 5, 6와 같이 정리할 수 있다. 실험이 수행되면 수식 4에 따라, 사용자 u의 유일한 positive 아이템 p_u가 상위 K개의 추천 목록에 포함되었는지 hit 값을 구한다. 모든 사용자 집합 U에 대해 hit 값의 평균을 구하면, HR@K를 계산(수식 5)할 수 있다. p_u가 상위 K개의 추천 목록에 포함되었을 때, p_u의 순위가 낮을수록 감소하는 누적 할인 이득(DCG) $\frac{1}{\log_2(rank(p_u)+1)}$ 값을 곱해 NDCG@K를 계산(수식 6)할 수 있다. 또한, 실험에서는 K = 5를 기준으로 실험 평가를 진행한다.

$$hit(u,K) = \begin{cases} 1 & \text{if } rank(p_u) \le K \\ 0 & \text{if } rank(p_u) > K \end{cases} \quad \text{(수식 4)}$$

$$HR@K = \frac{1}{|U|}\sum_{u \in U} hit(u,K) \quad \text{(수식 5)}$$

$$NDCG@K = \frac{1}{|U|}\sum_{u \in U}\left(hit(u,K) \cdot \frac{1}{\log_2(rank(p_u)+1)}\right) \quad \text{(수식 6)}$$

제 3 절 실험 환경 및 파라미터

실험 환경은 표 2와 같은 환경에서 이루어졌다. 본 실험에서는 Gemini-2.0-flash, GPT-4.1-mini, GPT-4.1-nano를 LLM 추천자로서 사용하며, 관련 버전은 표 2에 나타냈다. 실험에 사용된 LLM 추천자별 하이퍼파라미터는 표 3에 나타냈다.

<표 2> 실험 환경 및 패키지

Resource	Property
OS	Ubuntu 22.04.5 LTS
Python	3.10.12
Numpy	2.1.2
google	2.178.0
openai	1.93.0

<표 3> 실험에 사용된 LLM 추천자별 하이퍼파라미터

Gemini-2.0-flash		GPT-4.1-mini		GPT-4.1-nano	
temperature	0.6	temperature	0.6	temperature	0.6
top_p	0.1	top_p	0.1	top_p	0.1
		n	1	n	1
		frequency_penalty	0	frequency_penalty	0
		presence_penalty	0	presence_penalty	0
prompt wrapping	message	prompt wrapping	content	prompt wrapping	content
SST	gender, age	SST	gender, age	SST	gender, age
cut_off_1	0.189	cut_off_1	0.102	cut_off_1	0.306
cut_off_2	9	cut_off_2	5	cut_off_2	19

제 4 절 실험 결과 및 분석

본 절에서는 LLM 추천자의 프롬프트에 사용자 SST 정보 제공으로 인해 유발되는 특정 사용자 집단에 대한 추천 품질 저하를 완화하기 위해 제안한 선택적 SST 프롬프트 엔지니어링의 효과를 실험으로 검증한다. 우선, 숏헤드 사용자와 롱테일 사용자 집단 각각에 대해 사용자 SST 입력의 영향을 알아보기 위해, MovieLens 데이터 집합에 대해 프롬프트의 종류별 성능을 알아보는 사전 실험을 표 4과 같이 수행하였다. 실험에서 각 LLM 추천자의 프롬프트에 SST를 포함하지 않는 경우의 성능 지표를 'w/o_SST' 항목에 표현하였으며, SST를 포함한 경우의 성능 지표를 'w/_SST' 항목에 표현하였으며, 제안하는 방법은 'SST_Sel' 항목에 표현하였다. 그 결과, 표 4에서 숏헤드 사용자 그룹은 SST를 프롬프트에 포함했을 때 LLM의 순위화 결과가 더 우수했으며, 롱테일 사용자 그룹은 SST를 프롬프트에 포함하지 않았을 때의 순위화 결과가 더 우수하게 보고되었다.

또한, 본 논문에서는 데이터 집합에 대한 전반적인 추천 성능 향상 수치를 확인하고, 제안한 방법에 대한 가정이 특정 인구통계 집단에서만 유효한지 확인하기 위한 벤치마크 평가를 수행했다. MovieLens 대상 실험 결과를 나타낸 표 5와 표 6, Last.fm 대상 실험 결과를 나타낸 표 7에서는 인구통계 그룹별 상세한 NDCG@5, HR@5 성능이 함께 나타난다. 표 7은 Last.fm에서 사용자 SST가 누락된 결측치를 제거한 데이터들만을 대상으로 한다.

<표 4> MovieLens-1M 데이터 집합에서 3가지 LLM 추천자로 수행한 아이템 선호에 따른 사용자 그룹에 대한 프롬프트 종류별 성능 분포

Gemini-2.0-flash		
Metrics: NDCG@5	w/_SST prompt	w/o_SST prompt
short-head users	**0.6083**	0.5794
long-tail users	0.5019	**0.5167**
GPT-4.1-mini		
Metrics: NDCG@5	w/_SST prompt	w/o_SST prompt
short-head users	**0.5355**	0.5246
long-tail users	0.4503	**0.4579**
GPT-4.1-nano		
Metrics: NDCG@5	w/_SST prompt	w/o_SST prompt
short-head users	**0.3736**	0.3666
long-tail users	0.2745	**0.3050**

표 5의 결과, 제안하는 SST_Sel 방법은 3가지 LLM 추천자 모두에서 다른 방법들에 비해 1.0% 이상의 비율로 성능 향상을 보고했다. 개별 인구통계 집단별로 NDCG@5 성능을 확인했을 때, w/o_SST 방법이나 w/_SST 방법이 더 유효한 집단이 확인됐다. 다만, 이러한 양상은 LLM 추천자별로 다른 분포를 보인다. SST_Sel 방법은 female 집단, 35-44 집단, 45-49 집단에서는 3가지 LLM 추천자 모두 더 나은 추천 품질을 보였다.

<표 5> MovieLens-1M 대상 인구통계 집단별 LLM 추천자의 NDCG@5 성능 비교

Metrics: NDCG@5	Gemini-2.0-flash			GPT-4.1-mini			GPT-4.1-nano		
	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel
Average	0.5446	0.5300	**0.5521**	0.4926	0.4931	**0.4950**	0.3483	0.3488	**0.3642**
Male	0.5494	0.5374	**0.5576**	**0.5056**	0.5042	0.5031	0.3636	0.3645	**0.3745**
Female	0.5384	0.5206	**0.5452**	0.4758	0.4789	**0.4845**	0.3287	0.3286	**0.3508**
0-17	**0.6231**	0.5883	0.6184	0.5300	**0.5375**	0.5102	0.3665	0.3603	**0.3946**
18-24	0.5534	0.5514	**0.5729**	0.5350	**0.5365**	0.5042	0.3603	0.3657	**0.3826**
25-34	0.5230	0.5098	**0.5447**	**0.5098**	0.5075	0.4866	0.3724	**0.3862**	0.3718
35-44	0.5097	0.5066	**0.5272**	0.4640	0.4970	**0.4803**	0.3484	0.3423	**0.3539**
45-49	0.5479	0.5222	**0.5583**	0.4619	0.4631	**0.4773**	0.3187	0.3114	**0.3402**
50-55	**0.5341**	0.5249	0.5242	0.4635	0.4662	**0.5046**	0.3385	**0.3414**	0.3293
56+	**0.5660**	0.5378	0.5477	0.4954	0.4871	**0.5156**	0.3334	0.3294	**0.3988**

표 6의 결과, 제안하는 SST_Sel 방법은 3가지 LLM 추천자 모두에서 다른 방법들에 비해 1.0% 이상의 비율로 성능 향상을 보고했다. 개별 인구통계 집단별로 HR@5 성능을 확인했을 때, w/o_SST 방법이나 w/_SST 방법이 더 유효한 집단이 확인됐다. 다만, 이러한 양상은 LLM 추천자별로 다른 분포를 보인다. SST_Sel 방법은 female 집단에서는 3가지 LLM 추천자 모두 더 나은 추천 품질을 보였다.

<표 6> MovieLens-1M 대상 인구통계 집단별 LLM 추천자의 HR@5 성능 비교

Metrics: HR@5	Gemini-2.0-flash			GPT-4.1-mini			GPT-4.1-nano		
	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel
Average	0.7312	0.7245	**0.7327**	0.6636	0.6621	**0.6743**	0.4632	0.4639	**0.4810**
Male	**0.7434**	0.7341	0.7335	0.6733	0.6706	**0.6819**	0.4766	0.4762	**0.4901**
Female	0.7156	0.7122	**0.7317**	0.6511	0.6511	**0.6647**	0.4457	0.4482	**0.4694**
0-17	0.7909	0.7727	**0.7955**	**0.7136**	0.7091	0.6636	0.4773	0.4545	**0.5182**
18-24	0.7478	0.7522	**0.7566**	**0.6974**	0.6930	0.6908	0.4649	0.4759	**0.5044**
25-34	0.7237	0.7193	**0.7325**	0.6711	**0.6732**	0.6689	0.5044	**0.5263**	0.4934
35-44	0.6820	0.7061	**0.7083**	0.6425	0.6447	**0.6579**	**0.4649**	0.4539	0.4561
45-49	**0.7317**	0.7024	0.7244	0.6268	0.6293	**0.6463**	0.4415	0.4293	**0.4463**
50-55	**0.7297**	0.7108	0.7000	0.6568	0.6541	**0.6919**	0.4459	**0.4541**	0.4486
56+	**0.7484**	0.7298	0.7391	0.6553	0.6460	**0.7050**	0.4379	0.4348	**0.5217**

<표 7> Last.fm-1K 대상 인구통계 집단별 LLM 추천자의 NDCG@5 성능 비교

Metrics: NDCG@5	Gemini-2.0-flash			GPT-4.1-mini			GPT-4.1-nano		
	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel	w/o_SST	w/_SST	SST_Sel
Average	0.2511	0.2236	**0.2643**	**0.2415**	0.2303	**0.2604**	0.2073	0.2033	**0.2082**
Male	0.2096	0.1874	**0.2635**	0.2292	0.2163	**0.3041**	0.1905	0.1880	**0.2023**
Female	**0.2925**	0.2598	0.2651	**0.2539**	0.2444	0.2167	**0.2240**	0.2186	0.2141
18-24	0.2275	0.2112	**0.2437**	0.2446	0.2269	**0.2841**	**0.2074**	0.2033	0.2060
25-34	0.2983	0.2485	**0.3054**	0.2353	**0.2372**	0.2132	0.2071	0.2033	**0.2125**

표 7의 결과, 제안하는 SST_Sel 방법은 3가지 LLM 추천자 모두에서 성능 향상을 보고했다. 특히, male 집단에서 3가지 LLM 추천자 모두 더 나은 추천 품질을 보였다. w/o_SST 방법이나 w/_SST 방법이 더 유효한 경우도 확인되었으나, 이러한 양상은 LLM 추천자별로 다른 분포를 보인다.

비록 SST 입력으로 인한 LLM의 인기 추천 유도는 모델에 따라 강하게 나타나는 집단이 다를 수 있지만, 모든 LLM 추천자에 대해 전반적인 성능 향상을 보인 점은 사용자 SST 제공 여부가 여러 사용자 그룹에 걸쳐 LLM의 인기 아이템 추천을 유도하거나 완화함을 알 수 있다.

제 5 장 결 론

본 논문에서는 사용자의 인기 아이템 선호 여부에 따른 LLM의 추천 순위 정렬 최적화를 위해, 사용자 성향을 기반으로 프롬프트에 사용자 SST를 선택적으로 제공하는 접근법을 새롭게 제안했다. 제안한 방법으로 수행된 Gemini와 GPT 기반 LLM 추천자 방식 순위화 평가에서 모든 모델은 NDCG, Hit-Rate 지표에서 성능 향상을 보고 했다. 그리고, 여러 인구통계 그룹에 걸친 추천 순위 예측 향상을 분석해서, 선택적 SST 프롬프트 엔지니어링이 프롬프트에 대한 사용자 SST 정보의 입력이 인기 아이템 추천을 유도함을 입증했다. 추후 연구는 LLM의 편향된 추천을 유도하는 다른 유발 요인을 탐색하는 방향으로 확장될 수 있을 것이다.

참고문헌

[1] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

[2] Zhang, Susan, et al. "Opt: Open pre-trained transformer language models." arXiv preprint arXiv:2205.01068 (2022).

[3] Mann, Ben, et al. "Language models are few-shot learners." arXiv preprint arXiv:2005.14165 1.3 (2020): 3.

[4] Kojima, Takeshi, et al. "Large language models are zero-shot reasoners." Advances in neural information processing systems 35 (2022): 22199-22213.

[5] Chowdhery, Aakanksha, et al. "Palm: Scaling language modeling with pathways." Journal of Machine Learning Research 24.240 (2023): 1-113.

[6] Chen, Peng, Jiancheng Zhao, and Xiaosheng Yu. "Lighterkgcn: a recommender system model based on bi-layer graph convolutional networks." Journal of Internet Technology 23.3 (2022): 621-629.

[7] Saleh, Ahmed I., Ali I. El Desouky, and Shereen H. Ali. "Promoting the performance of vertical recommendation systems by applying new classification techniques." Knowledge-Based Systems 75 (2015): 192-223.

[8] Goyani, Mahesh, and Neha Chaurasiya. "A review of movie recommendation system: Limitations, Survey and Challenges." ELCVIA. Electronic letters on computer vision and image analysis 19.3 (2020): 0018-37.

[9] Hu, Yifan, Yehuda Koren, and Chris Volinsky. "Collaborative filtering for implicit feedback datasets." 2008 Eighth IEEE international conference on data mining. Ieee, 2008.

[10] Chen, Peng, Jiancheng Zhao, and Xiaosheng Yu. "Lighterkgcn: a recommender system model based on bi-layer graph convolutional networks." Journal of Internet Technology 23.3 (2022): 621-629.

[11] Jiang, Chumeng, et al. "Beyond Utility: Evaluating LLM as Recommender." Proceedings of the ACM on Web Conference 2025. 2025.

[12] Wei, Wei, et al. "Llmrec: Large language models with graph augmentation for recommendation." Proceedings of the 17th ACM international conference

on web search and data mining. 2024.

[13] Wang, Shijie, et al. "Knowledge graph retrieval-augmented generation for llm-based recommendation." arXiv preprint arXiv:2501.02226 (2025).

[14] Deldjoo, Yashar, and Tommaso Di Noia. "Cfairllm: Consumer fairness evaluation in large-language model recommender system." ACM Transactions on Intelligent Systems and Technology (2025).

[15] Zhang, Jizhi, et al. "Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation." Proceedings of the 17th ACM Conference on Recommender Systems. 2023.

[16] Sanyal, Soumya, Harman Singh, and Xiang Ren. "Fairr: Faithful and robust deductive reasoning over natural language." arXiv preprint arXiv:2203.10261 (2022).

[17] Wang, Haoyu, et al. "Perplexity Trap: PLM-Based Retrievers Overrate Low Perplexity Documents." arXiv preprint arXiv:2503.08684 (2025).

[18] Dai, Sunhao, et al. "Bias and unfairness in information retrieval systems: New challenges in the llm era." Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024.

[19] Jiang, Meng, et al. "Item-side fairness of large language model-based recommendation system." Proceedings of the ACM Web Conference 2024. 2024.

[20] Zheng, Chujie, et al. "Large language models are not robust multiple choice selectors." arXiv preprint arXiv:2309.03882 (2023).

[21] Ye, Seonghyeon, et al. "Investigating the effectiveness of task-agnostic prefix prompt for instruction following." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 38. No. 17. 2024.

[22] Orgad, Hadas, and Yonatan Belinkov. "BLIND: Bias removal with no demographics." arXiv preprint arXiv:2212.10563 (2022).

[23] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.

[24] Harper, F. Maxwell, and Joseph A. Konstan. "The movielens datasets: History and context." Acm transactions on interactive intelligent systems (tiis) 5.4 (2015): 1-19.

Abstract

MS.Thesis

A Study on Prompt Selection based on User Preferences in the LLM-as-Recommender System

Jeon Janggun
Department of Computer Science
Graduate School
Kyonggi University

With the rapid development of Large Language Models (LLMs), their outstanding zero-shot text processing and generation capabilities, even without separate fine-tuning, are expanding their application scope. Consequently, LLMs have begun to be utilized in recommendation systems, performing personalized recommendations and content ranking based on user profile information. However, most research on LLM-based recommendation systems operates as an augmenter (LLM-as-Augmenter) to traditional recommendation models, rather than as a recommender (LLM-as-Recommender). However, because the role of the LLM-as-Augmenter is performed in conjunction with existing recommendation models, the cold-start problem inherent in traditional recommendation models persists. In contrast, this paper focuses on the LLM-as-Recommender ranking recommendation system that overcomes the cold-start problem by leveraging its extensive inherent content understanding.

Research on utilizing LLMs as recommenders is still in its infancy, and concerns about hallucination and inherent implicit bias within the model itself are key issues in recommender systems. In particular, popularity bias in LLMs is problematic because they only recommend generally popular items, without considering users' personalized item preferences. This leads to a significant difference in recommendation quality compared to traditional recommendation models, depending on the detailed sensitive attributes (SST) of the user profiles included in the prompts. If uniform profile-based recommendations were made to all users regardless of their item preferences, this could result in lower recommendation quality for certain user groups. Therefore, in this paper, we predict user item preferences based on the item interaction history in the user profile and selectively incorporate user-SST information into the LLM prompts to mitigate the deterioration in recommendation quality caused by LLM popularity bias. The proposed method was validated through the LLM-as-Recommender ranking evaluation, demonstrating improved recommendation quality across various demographic groups.