GLI Lab Seminar

# LLM-based Recommendation Agents for Personalized Conversational Recommender Systems


**Hae-Yoon Koo (구해윤)**

Supervisor: Byungkook Oh

Graph & Language Intelligence Laboratory
Department of Computer Science and Engineering
Konkuk University


2026.08.12

# CONTENTS

1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# CONTENTS


1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Background

## Recommender System (RS)

- **추천 시스템이란?**
  - ✓ 사용자의 **과거 행동 및 선호 정보**를 기반으로
  - ✓ 사용자가 관심을 가질 가능성이 높은 아이템을 예측하고 추천하는 시스템

- **일반적인 동작 방식**
  - ✓ 사용자와 아이템 간의 상호작용 데이터를 활용
  - ✓ 각 아이템의 선호도를 예측한 뒤, 점수가 높은 아이템을 순위화하여 제공
  - ✓ Ex) 사용자 행동 이력 : "스파이더맨 시청" → "아이언맨 시청" → "어벤져스 시청"
    - 추천 결과 : "가디언즈 오브 갤럭시" 추천

- **핵심 특징**
  - ✓ One-shot recommendation
    - 사용자의 과거 정보로 한 번에 추천 결과를 생성
    - 추천 이후 사용자의 현재 의도나 세부 조건을 추가로 확인하지 않음
    - Ex) 사용자 : "영화 추천해줘." → RS : "스파이더맨 : 브랜드 뉴 데이를 추천합니다."
  - ✓ Implicit preference modeling
    - 클릭, 구매, 시청, 평점 등 과거 행동으로 사용자의 선호를 간접적으로 추론
    - 사용자가 "지금 무엇을 원하는지" 직접 표현하지 않아도 추천 가능
    - Ex) 최근 마블 영화를 여러 편 시청 → "액션 / SF 영화를 선호한다"고 추론

# Background

## Conversational Recommender System (CRS)

- **대화형 추천 시스템이란?**
  - ✓ 사용자와 **여러 차례 자연어로 상호작용**하면서 사용자의 선호와 제약을 파악
  - ✓ 적절한 아이템을 추천 및 설명하며, 사용자의 피드백에 따라 추천을 수정하는 시스템

- **핵심 특징**
  - ✓ Task-oriented
    - CRS의 목적은 단순한 자유 대화가 아니라 사용자의 의사결정을 돕는 것
    - Ex) 자유 대화 : "요즘 영화 뭐 봤어?" 는 대화 자체가 목적
    - Ex) CRS : "오늘 가족과 영화관에서 볼만한 영화 뭐 있어?" 는 영화를 고르는 것이 목적
  - ✓ Multi-turn interaction
    - 한 번의 질문과 한 번의 답변으로 끝나지 않음
    - Ex) 질문 : "오늘 가족과 영화관에서 볼만한 영화 뭐 있어?"
      - Single-turn → "스파이더맨 : 브랜드 뉴 데이를 추천합니다."
      - Multi-turn → "시리즈물도 괜찮나요?" (아니오) → "러닝타임이 길어도 괜찮나요?" (네)
      - → "오디세이를 추천합니다."

# Background

## Why Conversational Recommendation?

- 기존 추천 시스템의 한계
  - ✓ 과거 행동이 사용자의 **현재 의도**를 항상 반영하지는 않음
  - ✓ 현재 상황이나 세부 조건은 행동 이력만으로 파악하기 어려움

- 예시
  - ✓ 과거 시청 이력 : 컨저링 → 엑소시스트 → 애나벨
    - 기존 RS의 추론 : "이 사용자는 공포 영화를 좋아한다"
  - ✓ 하지만 현재 요청은
    - "오늘은 부모님과 같이 볼 영화가 필요해. 너무 무서운 건 싫고, 2시간 이내면 좋겠어."

- 무엇이 달라지는가
  - ✓ 과거 장르 선호만 사용하면 **현재 상황과 맞지 않는 공포 영화를 추천**
  - ✓ 현재 대화를 함께 사용하면 **공포 강도, 동행자, 러닝타임**을 반영하여 추천을 조정

# Background

## Personalized Conversational Recommender System

- **Personalized CRS는 왜 등장했는가?**
  - ✓ 기존 CRS는 주로 **현재 대화 세션**에서 드러난 선호를 파악
  - ✓ 현재 대화만으로는 사용자의 장기 취향과 개인적 특성을 충분히 반영하기 어려움

- **대표적인 연구 방향**
  - ✓ **Long-term Preference 활용**
    - 과거 평점, 구매, 클릭, 시청 행동에서 장기 선호를 학습
    - 현재 세션의 일시적인 요청과 함께 사용
  - ✓ **Historical Dialogue 활용**
    - 현재 대화 이전에 사용자가 나눈 과거 대화 세션을 활용
    - 사용자가 과거에 직접 표현한 선호와 피드백을 반영
  - ✓ **Multi-aspect User Modeling**
    - 현재 대화, 과거 대화, 유사 사용자 등 여러 관점에서 사용자 선호를 모델링

- **예시**
  - ✓ 과거 행동 : 컨저링 → 엑소시스트 → 애나벨 (장기적으로 공포 영화를 선호)
  - ✓ 현재 대화 : "오늘은 부모님과 볼 거라 너무 무서운 건 싫어요."
  - ✓ 개인화된 CRS : 현재 대화의 낮은 공포 강도를 우선하되, 평소 선호한 긴장감도 함께 고려

# Background

## Challenge

- **장기 선호는 항상 현재 추천에 도움이 될까?**
  - ✓ 사용자의 과거 행동과 이전 대화는 중요한 개인화 정보
  - ✓ 하지만 현재 상황과 관련이 없거나 충돌하면 **추천을 방해**할 수 있음

- **잘못된 개인화**
  - ✓ 과거 공포 영화 선호를 지나치게 반영
  - ✓ 현재의 동행자와 공포 강도 제약을 충분히 고려하지 않음
  - ✓ 현재 상황과 맞지 않는 공포 영화를 추천

- **적절한 개인화**
  - ✓ 현재 요청과 관련된 장기 선호만 선택적으로 활용
  - ✓ 현재 제약을 우선하면서 평소 취향을 보조적으로 반영
  - ✓ 공포 강도는 낮지만 긴장감 있는 미스터리 / 스릴러 후보를 추천

- **핵심 질문**
  - ✓ **어떤 과거 정보를 현재 추천에 사용할 것인가?**
  - ✓ 현재 의도와 장기 선호가 충돌하면 무엇을 우선할 것인가?
  - ✓ 오래되거나 무관한 선호는 어떻게 수정하거나 잊을 것인가?

# Background

## 대화형 추천 시스템의 발전

- 초기 Interactive CRS
  - ✓ 시스템이 정해진 속성이나 조건을 질문
  - ✓ 사용자의 답변을 통해 후보 아이템을 줄이는 방식
  - ✓ Ex) 장르, 가격대, 브랜드, 러닝타임 질문

- Neural / Open-ended CRS
  - ✓ 자유로운 자연어 추천 대화를 neural model로 처리

- Personalized / User-Centric CRS
  - ✓ 현재 세션뿐 아니라 장기 선호, 과거 대화 및 유사 사용자 활용
  - ✓ 사용자별 추천 및 응답 생성 강화

- LLM-based CRS
  - ✓ 복합적인 자연어 요구 이해
  - ✓ zero-shot / few-shot 추천 및 자연스러운 설명 생성

- Agentic CRS
  - ✓ LLM이 대화만 생성하는 것을 넘어
  - ✓ memory, planning, tool use, feedback을 이용해 추천 과정을 조율

# Background

## LLM 기반 추천 에이전트란?

- **LLM 기반 CRS**
  - ✓ 대화를 LLM에 넣으면 추천과 응답을 한 번에 생성
  - ✓ User Dialogue → LLM → Recommendation / Response
  - ✓ LLM이 하는 일은 **입력을 읽고 출력을 만드는 것** 하나

- **LLM 기반 추천 에이전트**
  - ✓ LLM이 **현재 상태를 파악하고 다음 행동을 스스로 결정**
  - ✓ 필요하면 외부 도구를 호출하고, 결과를 보고 행동을 수정
  - ✓ User Dialogue → (Memory 읽기 → Planning → Tool 호출 → 결과 확인) → Response

- **구성 요소**
  - ✓ **Memory** : 과거 대화와 행동 기록을 저장하고 필요한 것만 꺼내 씀
  - ✓ **Planning** : 무엇을 먼저 할지, 언제 물어보고 언제 추천할지 결정
  - ✓ **Tool** : 추천 모델, 지식 그래프, 검색, DB 조회를 도구로 호출
  - ✓ **Feedback** : 사용자의 반응을 받아 다음 행동과 기억을 갱신

- **핵심 차이**
  - ✓ 추천 모델이 LLM을 돕는 것이 아니라, **LLM이 추천 모델을 도구로 호출**

# Background

## 왜 LLM 기반 추천 에이전트인가?

- **개인화에 쓸 정보가 많아짐**
  - ✓ 현재 대화, 장기 선호, 과거 대화 세션, 유사 사용자
  - ✓ 정보가 늘수록 **무엇을 쓸지 고르는 일** 자체가 문제가 됨

- **한 번의 프롬프트로는 처리하기 어려움**
  - ✓ 개인화 정보를 전부 프롬프트에 넣으면 길이 한계에 걸림
  - ✓ 현재 요청과 무관한 과거 정보까지 들어가 의도 파악을 흐림
  - ✓ 무엇을 넣을지 고르려면 **현재 요청과 비교하는 판단 단계**가 필요

- **추천에 필요한 자원은 대화 밖에 있음**
  - ✓ 실제 후보 아이템, 사용자 프로필, 지식 그래프는 LLM 안에 없음
  - ✓ 필요한 시점에 **외부 도구를 호출**하고 결과를 다시 읽어야 함

- **대화는 한 번에 끝나지 않음**
  - ✓ 물어볼지 추천할지, 다시 계획할지를 턴마다 결정해야 함
  - ✓ 사용자 피드백을 받아 기억과 계획을 갱신해야 함

- **그래서 Memory, Planning, Tool, Feedback을 갖춘 에이전트 형태로 발전**

# CONTENTS


1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Related Work

## Recent Works

- 개인화된 LLM 추천 에이전트의 핵심 설계 문제 세 가지
  - ✓ 어떤 과거 정보를 기억하고 사용할 것인가
    - 과거 대화가 쌓일수록 현재 요청과 무관한 정보가 섞임
    - 전부 사용하면 개인화가 오히려 추천을 방해함
  - ✓ 언제, 어떻게 계획하고 도구를 사용할 것인가
    - 계획과 도구 호출을 늘리면 정확해지지만 응답이 느려짐
    - 단순한 질문까지 무거운 과정을 거치면 사용자가 이탈
  - ✓ 어떤 행동 경로를 좋은 추천으로 평가할 것인가
    - 여러 추론 경로를 만들 수 있지만 무엇이 좋은 경로인지 기준이 없음
    - Recall이나 BLEU가 높아도 사용자가 만족하지 않는 경우가 많음
- 오늘 살펴볼 세 논문
  - ✓ EyeCRS (WWW 2026) : 과거 대화 중 쓸 것만 골라내는 방법
  - ✓ AdaptJobRec (AAAI 2026) : 질의 복잡도에 따라 처리 경로를 나누는 방법
  - ✓ HARPO (ACL 2026) : 추론 경로를 만들고 추천 품질로 평가하는 방법

# CONTENTS


1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Related Work

[2026][WWW][EyeCRS]

## Not All Information Brings Benefits: Personalization-Driven Agent Debate for Conversational Recommendation


Pengfei Zhang*
202421080437@std.uestc.edu.cn
University of Electronic Science and Technology of China
Chengdu, China

Guojia An*
an_guojia@163.com
University of Electronic Science and Technology of China
Chengdu, China

Jin Huang
jh2642@cam.ac.uk
University of Cambridge
Cambridge, United Kingdom

Yuhan Yang
y.yuhan2000@gmail.com
University of Electronic Science and Technology of China
Chengdu, China

Yang Yang
yang.yang@uestc.edu.cn
University of Electronic Science and Technology of China
Chengdu, China

Jie Zou†
jie.zou@uestc.edu.cn
University of Electronic Science and Technology of China
Chengdu, China

# [2026][WWW][EyeCRS]

## Background

- **Personalized CRS는 과거 대화를 개인화 정보로 사용**
  - ✓ **historical dialogue session** : 현재 대화 이전에 나눈 완결된 과거 대화
  - ✓ **dialogue history** : 현재 대화 안의 이전 발화
  - ✓ EyeCRS가 다루는 것은 **과거 대화 세션**

- **기존 방식**
  - ✓ 사용자의 과거 대화 세션을 **전부 모아서** 현재 추천에 입력
  - ✓ 정보가 많을수록 개인화가 잘 된다고 가정

- **Cognitive Negative Transfer**
  - ✓ 이전에 배운 지식을 새로운 상황에 잘못 적용해 오히려 수행이 나빠지는 현상
  - ✓ CRS에서는 오래되거나 무관한 과거 선호가 현재 의도 파악을 방해
  - ✓ Ex) 과거 대화에서 "공포 영화를 좋아한다"고 말한 사용자
    - 현재 요청은 "부모님과 같이 볼 영화"
    - 과거 선호를 그대로 쓰면 공포 영화를 추천하게 됨

[2026][WWW][EyeCRS] Not All Information Brings Benefits: Personalization-Driven Agent Debate for Conversational Recommendation

# [2026][WWW][EyeCRS]

## Introduction

- 과거 정보는 정말 도움이 되는가
  - ✓ 과거 대화를 넣었을 때와 넣지 않았을 때 정답 아이템과의 유사도 변화를 측정
  - ✓ TGReDial : 증가 35.9%, 변화 없음 36.1%, **감소 27.9%**
  - ✓ ReDial : 증가 33.2%, 변화 없음 38.5%, **감소 28.3%**
  - ✓ 도움이 되는 경우는 **약 3분의 1**, 나머지는 무의미하거나 오히려 해로움

- 문제 정의
  - ✓ 과거 대화를 쓸지 말지를 시스템이 스스로 판단해야 함
  - ✓ 판단 기준을 학습할 정답 라벨이 없음

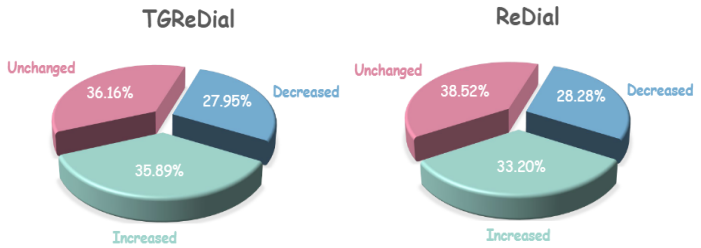



Figure 2: Distribution of cognitive negative transfer phenomena on TGReDial and ReDial datasets.

- 핵심 아이디어
  - ✓ 과거 세션마다 **찬성 에이전트와 반대 에이전트가 토론**
  - ✓ 토론 내용을 보고 판정 모델이 유용한 세션만 선택
  - ✓ 선택된 세션을 프롬프트가 아니라 **모델 파라미터에 주입**

[2026][WWW][EyeCRS] Not All Information Brings Benefits: Personalization-Driven Agent Debate for Conversational Recommendation

# [2026][WWW][EyeCRS]

## Introduction : 예시

- 상황
  - ✓ 현재 대화 : "Hi, can you recommend a comedy movie?"
  - ✓ 이 사용자에게는 과거 대화 세션 네 개가 남아 있음
  - ✓ 정답 아이템 : Four Weddings and a Funeral

- 기존 방식 (그림 위쪽)
  - ✓ 과거 세션 네 개를 **구분 없이 전부** 추천기에 입력 (Blind Usage)
  - ✓ 현재 의도와 무관한 세션까지 반영되어 **정답과 다른 영화를 추천**

- 제안 방식 (그림 아래쪽)
  - ✓ Agent Debate가 세션 하나하나를 검토
  - ✓ **세션 2와 세션 4만 유용하다고 판정**하고 나머지는 제외
  - ✓ 선별된 세션만 사용해 정답과 일치하는 추천 결과를 얻음

- 읽는 포인트
  - ✓ 과거 정보를 버리는 것이 아니라 **현재 대화와 맞는 것만 고르는 것**

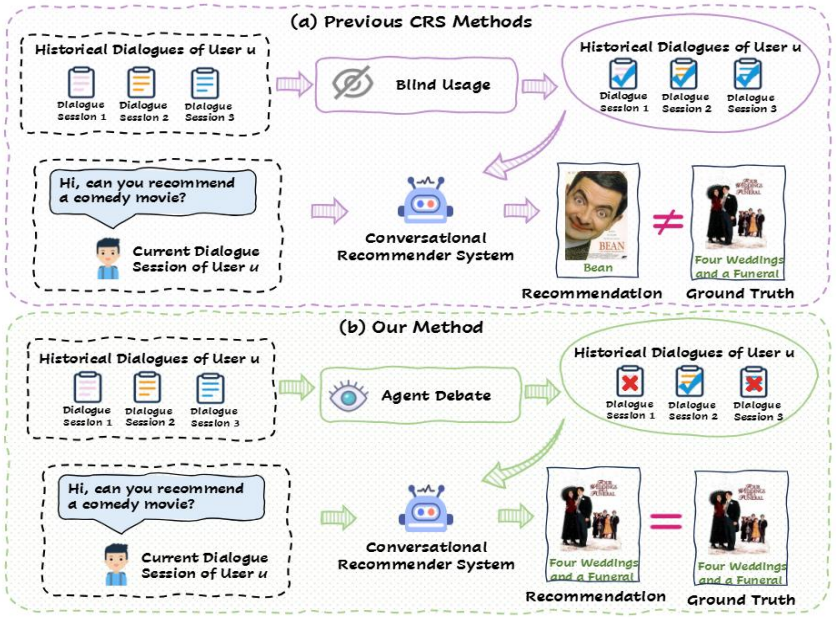



**Figure 1: Previous CRS methods vs. our method.**

# [2026][WWW][EyeCRS]

## Proposed Method

- 두 개의 모듈로 구성
  - ✓ Multi-Agent Debate Module : 과거 세션 중 쓸 것만 선별
  - ✓ Parametric Injection Module : 선별된 세션을 LoRA 파라미터로 주입

- 현재 대화 세션은 그대로 사용하고, 과거 세션만 걸러냄

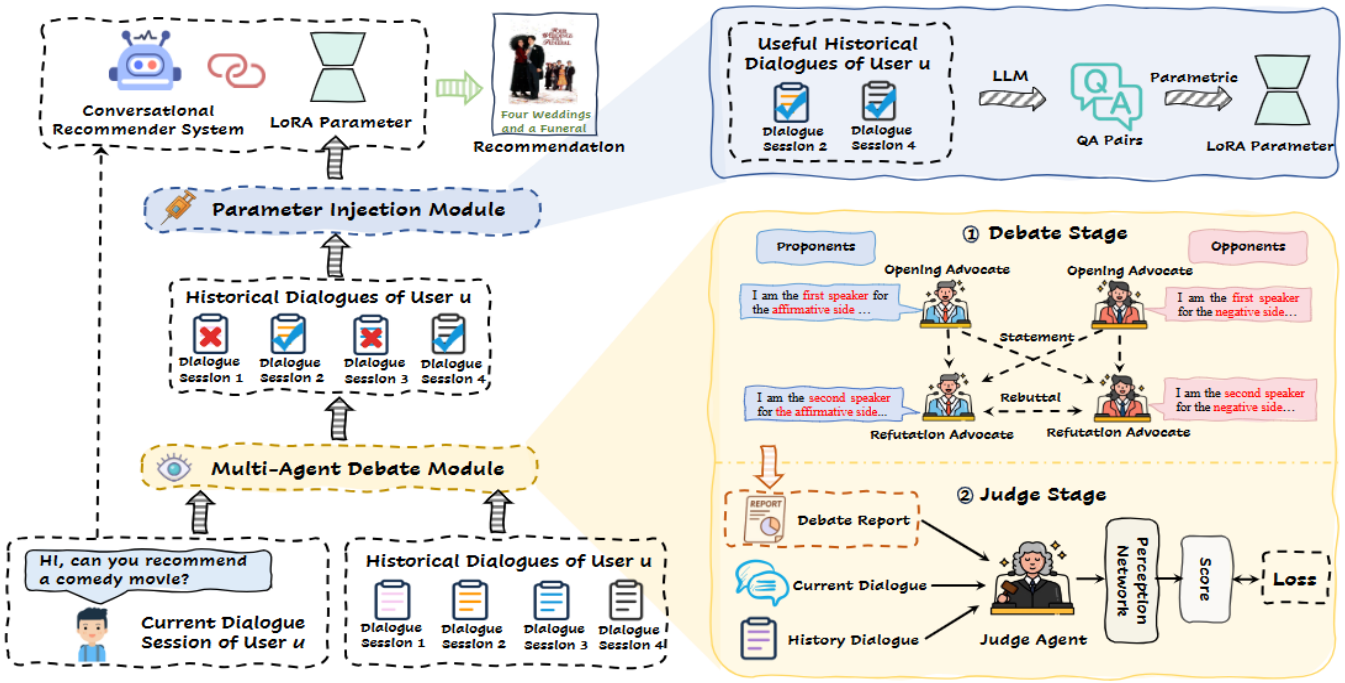
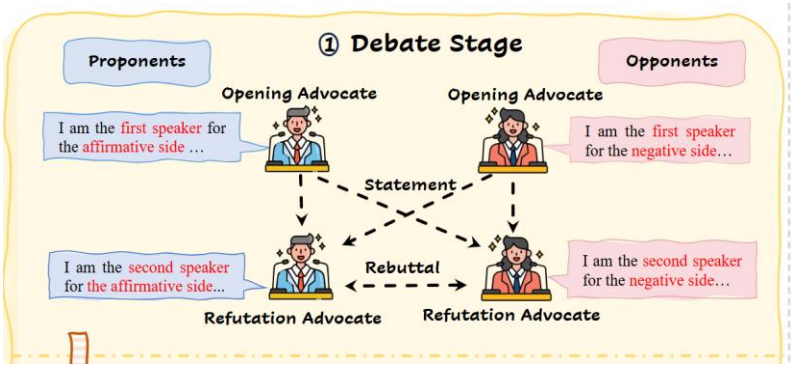



**Figure 3: Overview of EyeCRS and its two components: (i) Multi-Agent Debate Module, which filters out unhelpful historical dialogue sessions to mitigate cognitive negative transfer; and (ii) Parametric Injection Module, which injects beneficial historical knowledge into the intrinsic parameter space of the LLM for personalized reasoning.**

## Proposed Method : Multi-Agent Debate

- **과거 세션 하나마다 두 팀을 붙임**
  - ✓ **Proponent** : 이 세션은 현재 의도를 이해하는 데 도움이 된다
  - ✓ **Opponent** : 이 세션은 현재 판단을 방해한다

- **토론은 두 단계**
  - ✓ **Statement** : 각자 현재 대화와 과거 세션을 보고 처음 주장을 생성
  - ✓ **Rebuttal** : 상대 주장을 읽고 반박을 생성
  - ✓ 두 단계 기록을 합쳐 **토론 기록**으로 저장하고 다음 단계로 전달

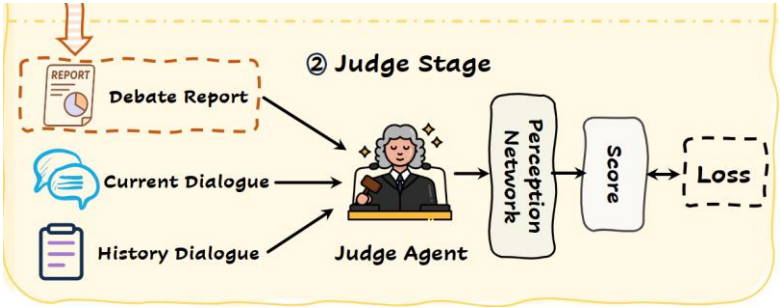



- **왜 토론인가**
  - ✓ LLM에게 한 번만 물어보면 근거 없이 판단하기 쉬움
  - ✓ **양쪽 근거를 모두 만들게 해서** 판단 신뢰도를 높임

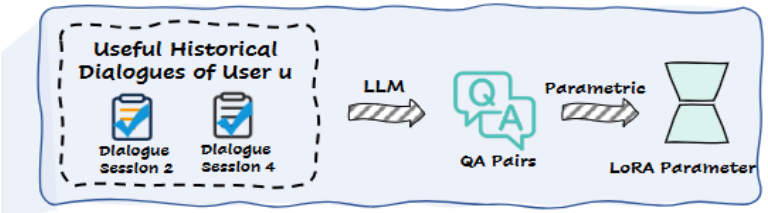
# [2026][WWW][EyeCRS]

## Proposed Method : Judge & Injection

- Judge
  - ✓ 토론 기록과 현재 대화를 하나의 입력으로 만들어 점수를 계산
  - ✓ 임계값을 넘는 세션만 사용
  - ✓ 인코더는 고정하고 **점수 계산 부분만 학습**




- **정답 라벨이 없는 문제를 푸는 방법**
  - ✓ 과거 세션을 넣었을 때 정답 아이템과의 유사도가
    - ▪ **올라가면 positive**, **내려가면 negative**
  - ✓ 이 신호로 판정 모델을 학습 (weak supervision)




- Parametric Injection
  - ✓ 선별된 세션을 질문 답변 쌍으로 바꿔 LoRA로 학습
  - ✓ 프롬프트에 넣지 않으므로 컨텍스트 길이를 소모하지 않음
  - ✓ 개인 정보가 참고 자료가 아니라 **모델 안의 기억**이 됨

# CONTENTS

1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Related Work

## [2026][AAAI][AdaptJobRec]

**AdaptJobRec: Enhancing Conversational Career Recommendation through an LLM-Powered Agentic System**

**Qixin Wang[1,2], Dawei Wang[1], Kun Chen[1], Yaowei Hu[1]**
**Puneet Girdhar[1], Ruoteng Wang[1], Aadesh Gupta[1], Chaitanya Devella[1]**
**Wenlai Guo[1], Shangwen Huang[1], Bachir Aoun[1], Greg Hayworth[1]**
**Han Li[1]*, Xintao Wu[2]***
[1]Walmart Global Tech
han.li@walmart.com
[2]University of Arkansas
xintaowu@uark.edu

# [2026][AAAI][AdaptJobRec]

## Background

- **Agentic CRS의 일반적인 동작**
  - ✓ 질의를 받으면 계획을 세우고, 도구를 호출하고, 결과를 확인해 다시 계획
  - ✓ ReAct, Plan-and-Execute, RecMind, InteRecAgent가 모두 이 형태

- **얻는 것**
  - ✓ 복잡한 요청을 여러 단계로 나눠서 처리 가능
  - ✓ 필요한 정보를 도구로 가져와 정확도 상승

- **잃는 것**
  - ✓ 단계가 늘어난 만큼 **응답 시간이 길어짐**
  - ✓ LLM 호출 횟수가 늘어 비용도 함께 증가

- **실제 서비스에서의 문제**
  - ✓ Walmart 채용 서비스는 연간 수백만 명이 사용
  - ✓ 질의의 상당수는 "내 지원 상태 알려줘" 같은 **단순 조회**
  - ✓ 단순 질의까지 계획과 메모리 단계를 거치면 **느려서 쓸 수 없음**

# [2026][AAAI][AdaptJobRec]

## Introduction

- **문제**
  - ✓ 모든 질의를 같은 파이프라인으로 처리하는 것이 지연의 원인
  - ✓ 첫 토큰을 빨리 내보내는 것만으로는 해결되지 않음
  - ✓ 사용자가 원하는 것은 **정보에 도달하는 시간**을 줄이는 것

- **핵심 아이디어**
  - ✓ 질의를 **simple과 complex로 먼저 분류**
  - ✓ simple은 추천 API와 지식 그래프로 바로 응답
  - ✓ complex만 메모리 처리와 계획 단계를 거치게 함

- **함께 해결한 것**
  - ✓ 대화 기록 중 현재 질의와 관련된 부분만 뽑아 계획 품질을 높임
  - ✓ 동시에 실행 가능한 작업을 묶어 병렬로 처리

- **결과**
  - ✓ 평균 응답 시간 최대 **53.3% 감소**, 추천 정확도는 오히려 상승

# [2026][AAAI][AdaptJobRec]

## Introduction : 예시

- **Simple query** : "Check my Job Application Status"
  - ✓ 대화 기록과 질의를 합쳐 지원 상태 API로 바로 조회
  - ✓ 메모리 처리와 계획 단계를 **거치지 않음**
  - ✓ 사용자는 몇 초 안에 원하는 정보를 받음

- **Complex query** : "Help me make a career development plan"
  - ✓ Memory Process가 대화 기록에서 관련 부분만 추출
    - Ex) "딥러닝 자격증을 땄다"는 이전 발화는 사용, 사무실 위치 질문은 제외
  - ✓ Planner가 다섯 개의 서브태스크로 분해
    - ① 커리어 패스 추천 ② 스킬 갭 식별 ③ 학습 자료 추천
    - ④ 멘토 추천 ⑤ 지원 가능한 공고 추천
  - ✓ 각 태스크를 도구로 실행한 뒤 하나의 응답으로 종합

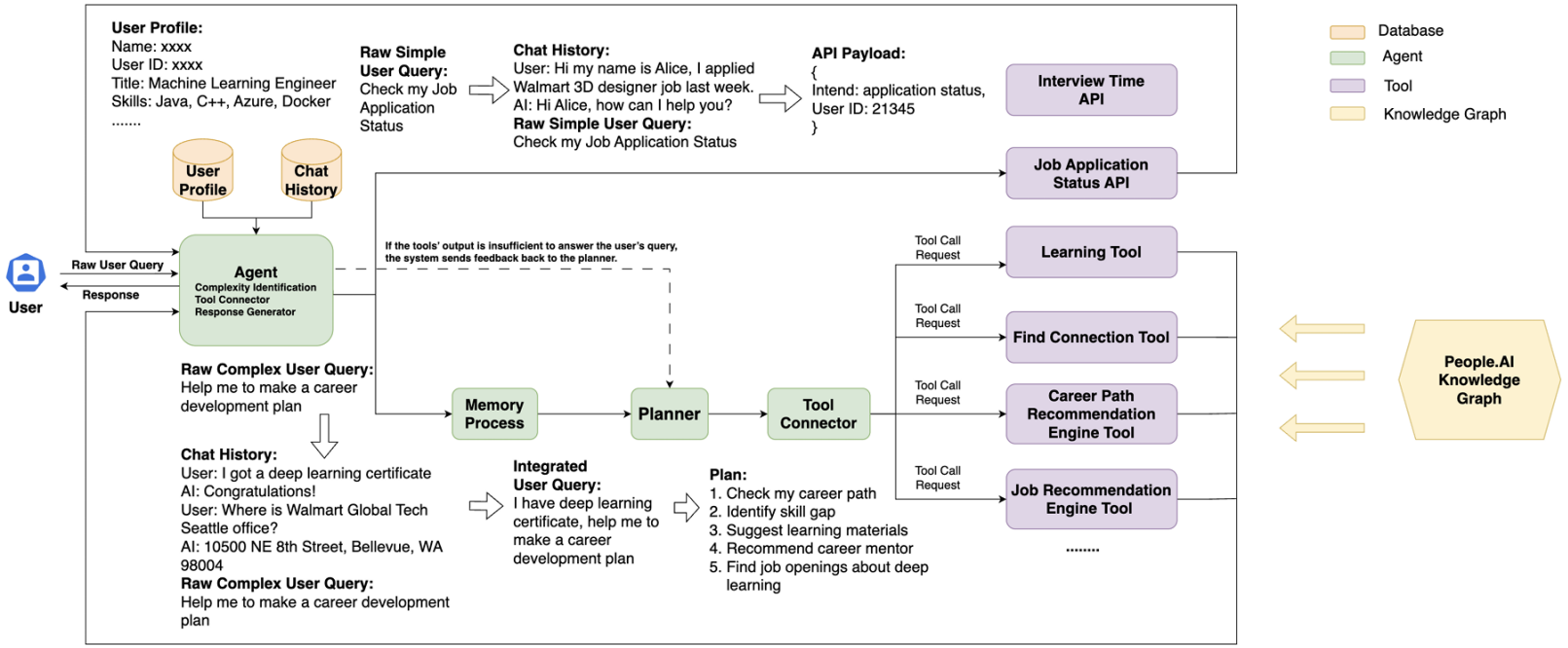
- **읽는 포인트**
  - ✓ 같은 시스템 안에서 **두 개의 처리 경로**가 갈린다는 것

# [2026][AAAI][AdaptJobRec]

## Proposed Method

- 구성
  - ✓ Agent : 질의 복잡도 판단, 도구 연결, 응답 생성
  - ✓ Memory Process, Planner, 개인화 추천 도구, People.AI 지식 그래프

- Complexity Identification
  - ✓ simple : 대화 기록과 질의를 합쳐 추천 API로 바로 처리
  - ✓ complex : Memory Process → Planner → Tools 순서로 처리



# [2026][AAAI][AdaptJobRec]

## Proposed Method : Memory & Planner

- **Memory Processing Module**
  - ✓ 기존 방식은 대화 기록을 잘라내거나 요약해서 사용
  - ✓ 필요한 내용이 빠지거나 무관한 내용이 남아 계획이 흔들림
  - ✓ few shot 예시로 **현재 질의와 관련된 부분만 추출**해 질의와 합침
  - ✓ 되묻는 횟수와 후속 질의 수가 줄어드는 효과

```
You need to re-write the user request on the last part of the text under [TEXT]
while considering the chat history if additional information is needed.
For example:

[TEXT]
User: Where is the Bentonville Walmart Fitness Center located?
Assistant: 1400 SE 5th St, Bentonville, AR 72716
User: How many people applied for the 3D Designer job last week?
Assistant: 512 candidates applied for the 3D Designer job last week.
User: how about the Graphic Designer role?
[OUTPUT]
How many people applied for the Graphic Designer job in last week?

[TEXT]
User: How many people apply 3D Designer job in last week?
Assistant: 512 candidates apply 3D designer job in last week.
User: How's the weather today?
[OUTPUT]
How's the weather today?

[TEXT]
{chat_history}
{user_query}
[OUTPUT]
```

Few-shot example 1: Chat history contains relevant information

Few-shot example 2: No relevant information in chat history

Figure 3: System Prompt of Few-shot Learning Memory Processing Module.

- **Planner**
  - ✓ 기존 플래너는 작업을 **순서대로 하나씩** 나열
  - ✓ 동시에 해도 되는 작업을 묶어 **중첩 리스트**로 출력
  - ✓ Ex) 시애틀 공고 수 조회와 서니베일 공고 수 조회는 동시에 실행한 뒤 비교

- **개인화 추천 도구**
  - ✓ People.AI 지식 그래프 사용 (노드 160만, 엣지 8300만)
  - ✓ 프로필의 스킬, 학력, 지역을 공고와 매칭해 점수를 계산
  - ✓ 클릭, 저장, 좋아요 같은 실시간 반응으로 점수를 조정

- **도구 결과가 부족하면 Planner로 되돌려 다시 계획**

```
You need to split the sentences under [TEXT] into several sub-tasks,
following the format shown in the examples. Group the subtasks that can be
executed asynchronously into a single list.
If the sentence contains only one intent, do not split it.

For example,

[TEXT]
Which city has more machine learning engineer job openings, Seattle or
Sunnyvale?
[OUTPUT]
- [Get machine learning engineer job opening number from Seattle; Get machine
learning engineer job opening number from Sunnyvale]
- [Compare job opening numbers of Sunnyvale and Seattle]

 [TEXT]
How many machine learning engineer job openings at Sunnyvale?
[OUTPUT]
- [Get machine learning engineer job opening number from Sunnyvale]

[TEXT]
{integrated_user_query}
[OUTPUT]
```

Few-shot example 1: Contain asynchronously executable sub-tasks

Few-shot example 2: Do not contain asynchronously executable sub-tasks

Figure 4: System Prompt of AdaptJobRec Planner.

# CONTENTS


1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Related Work

[2026][ACL][HARPO]

## HARPO: Hierarchical Agentic Reasoning for User-Aligned Conversational Recommendation

**Subham Raj**[1] **Aman Vaibhav Jha**[1] **Mayank Anand**[2] **Sriparna Saha**[1]
[1]Department of Computer Science and Engineering, Indian Institute of Technology Patna, India
[2]Indian Institute of Information Technology Allahabad, India
{subham_2221cs25, 2201ai54_aman, sriparna}@iitp.ac.in
anandmayank698@gmail.com

# [2026][ACL][HARPO]

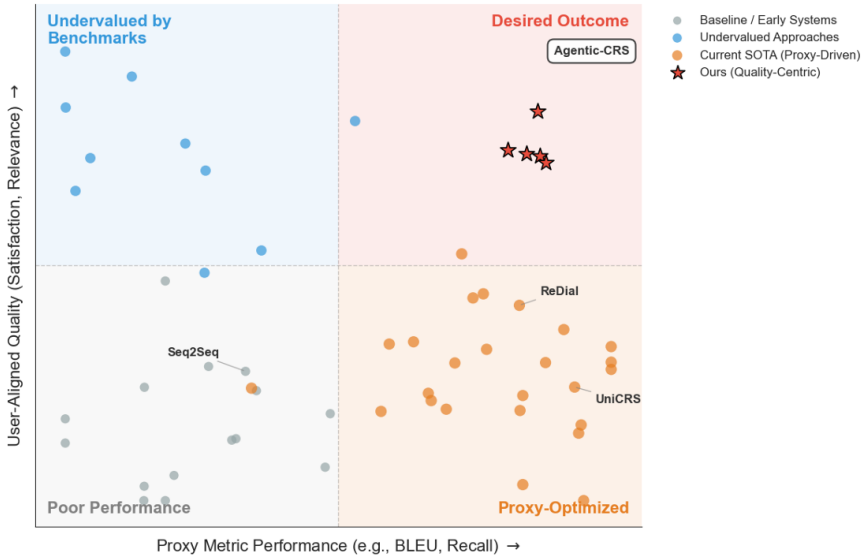
## Background

- **지금까지의 CRS는 무엇으로 평가되었는가**
  - ✓ Recall@K, BLEU 같은 **대리 지표(proxy metric)**로 성능을 측정
  - ✓ 대리 지표가 높아도 사용자가 만족하지 않는 경우가 많음

- **예시**
  - ✓ 요청 : "여름 결혼식에 입을 캐주얼한 옷"
  - ✓ 시스템이 이를 **평상복**으로 이해해도 지표 점수는 잘 나옴
  - ✓ 하지만 실제 상황에는 맞지 않는 추천 → **지표와 만족도가 어긋남**

- **원인**
  - ✓ 학습 목표가 **검색 정확도나 문장 유창성**에 맞춰져 있음
  - ✓ 추천 품질 자체를 직접 학습하지 않음

- **그래서 필요한 것**
  - ✓ 여러 후보 경로를 만들어 보고
  - ✓ **어떤 경로가 좋은 추천인지 판단하는 기준**을 함께 학습해야 함



# [2026][ACL][HARPO]

## Introduction

- **핵심 아이디어**
  - ✓ 추론 경로를 **여러 개 만들어 보고**, 품질로 평가하고, 여러 에이전트가 다듬음

- **네 개의 구성 요소**
  - ✓ **STAR** : Tree of Thought로 여러 추론 경로를 탐색하고 최적 경로를 선택
  - ✓ **CHARM** : 추천 품질을 네 가지로 분해해 보상으로 학습
  - ✓ **BRIDGE** : 다른 도메인으로 옮겨도 동작하도록 표현을 적응
  - ✓ **MAVEN** : 추천, 비판, 설명 에이전트가 합의하여 최종 응답 생성

- **예시**
  - ✓ Query : "I loved The Dark Knight but want something with less violence for family movie night."
  - ✓ 후보 생성과 평가를 거쳐 **The Incredibles**를 최종 추천
  - ✓ 응답 : 다크나이트의 영웅적 요소는 유지하면서 가족이 함께 볼 수 있는 작품이라고 설명

- **결과**
  - ✓ ReDial : Recall@10 **+18.7%**, 사용자 만족도 +21.4%
  - ✓ INSPIRED : Recall@10 **+34.3%**, 만족도 +26.9% / MUSE : Recall@10 +12.9%, 만족도 +18.0%

# [2026][ACL][HARPO]

## Proposed Method

- 하나의 LLM backbone 위에 네 개의 모듈이 연결됨
  - ✓ Query → **BRIDGE** (도메인 적응) → **STAR** (경로 탐색) → **MAVEN** (합의) → 최종 응답
  - ✓ **CHARM**이 만든 품질 보상이 STAR의 경로 선택 기준으로 사용됨

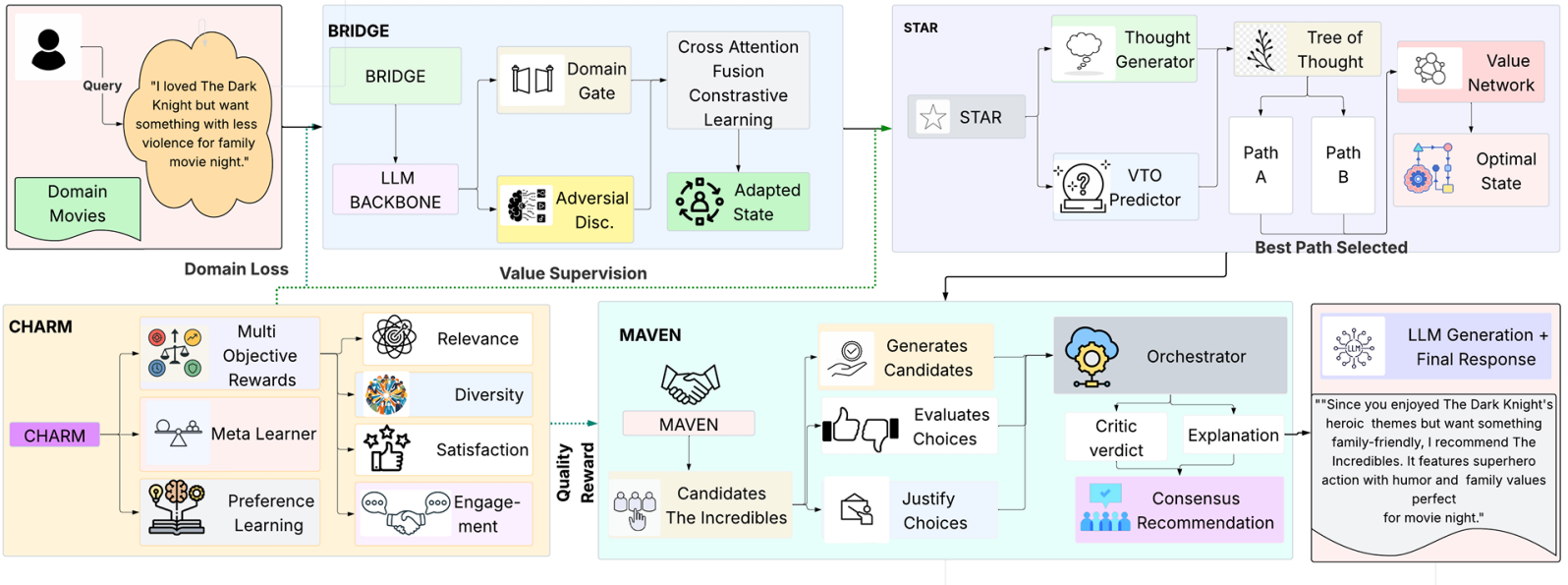



Figure 2: Overall architecture of the HARPO framework. The model integrates four components: STAR for structured agentic reasoning, CHARM for hierarchical preference optimization, BRIDGE for cross-domain transfer, and MAVEN for multi-agent refinement, all built on a shared language model backbone.

# [2026][ACL][HARPO]

## Proposed Method : STAR & CHARM

- STAR (Structured Tree-of-Thought Agentic Reasoning)
  - ✓ Thought Generator가 여러 추론 경로를 생성 (Path A, Path B)
  - ✓ Value Network가 각 경로의 **추천 품질**을 예측
  - ✓ 작업을 끝냈는지가 아니라 **품질 기준으로 최적 경로를 선택**

- CHARM (Contrastive Hierarchical Alignment with Reward Marginalization)
  - ✓ 추천 품질을 네 가지로 분해
    - Relevance 요청과 맞는가 / Diversity 비슷한 것만 주지 않는가
    - Satisfaction 사용자가 만족하는가 / Engagement 대화를 이어가고 싶은가
  - ✓ 항목마다 별도의 reward head를 두고 점수를 계산
  - ✓ Meta Learner가 **대화 맥락에 따라 항목별 가중치를 조정**

- **두 모듈의 연결**
  - ✓ CHARM이 만든 품질 신호로 **STAR의 Value Network를 학습**
  - ✓ 즉 무엇이 좋은 추천인지를 배운 뒤, 그 기준으로 경로를 고름

# [2026][ACL][HARPO]

## Proposed Method : BRIDGE & MAVEN

- BRIDGE (Cross-Domain Transfer)
  - ✓ Adversarial Discriminator로 **도메인을 구분하는 정보를 지움**
  - ✓ Domain Gate가 적응된 표현과 원래 표현을 섞는 비율을 학습
  - ✓ 영화 도메인에서 배운 것을 패션 도메인으로 옮겨도 동작

- MAVEN (Multi-Agent Refinement)
  - ✓ 세 개의 에이전트가 각자 역할을 맡음
    - 후보 생성 / 후보 평가 및 비판 / 추천 이유 설명
  - ✓ Orchestrator가 결과를 모아 **합의된 추천**을 만듦
  - ✓ 서로 어긋나지 않도록 agreement loss로 함께 학습

- 세 논문 정리
  - ✓ EyeCRS : **무엇을 기억할 것인가**
  - ✓ AdaptJobRec : **언제 도구를 부를 것인가**
  - ✓ HARPO : **어떤 경로를 좋은 추천으로 볼 것인가**

# CONTENTS

1. Background
2. Related Work
   - EyeCRS
   - AdaptJobRec
   - HARPO
3. Research Direction

# Research Direction

## 연구 방향 A : 개인화된 대화형 추천

- **다루는 문제**
  - ✓ 개인화 정보 = 현재 세션 정보에 **장기 사용자 정보를 보완적으로 결합**한 것
  - ✓ **Current-session Signals** : 현재 대화에서 직접 드러난 목적, 선호, 제약
    - Ex) "부모님과 같이 볼 영화", "너무 무서운 건 싫음", "2시간 이내"
  - ✓ **Long-term Preference / Profile** : 과거 행동이나 rating에서 추정한 장기 선호
    - Ex) 컨저링 → 엑소시스트 → 애나벨 (공포 영화 선호)
  - ✓ **Historical Dialogue Sessions** : 현재 대화 이전에 나눈 과거 대화 세션
  - ✓ **Similar / Look-alike Users** : 비슷한 취향을 가진 다른 사용자들의 정보

- **오늘 본 논문과의 연결**
  - ✓ EyeCRS가 다룬 것은 네 가지 중 **Historical Dialogue Sessions 하나**
  - ✓ 네 종류를 함께 쓸 때 무엇을 우선할지는 아직 정해진 답이 없음

- **해볼 만한 것**
  - ✓ 턴이 진행될 때마다 **쓸 개인화 정보를 다시 고르는 방법**
  - ✓ "이건 싫어" 같은 부정 피드백을 선호에 반영하는 방법

# Research Direction

## 연구 방향 B : Knowledge Graph + LLM 추천

- **다루는 문제**
  - ✓ LLM 추천기가 가진 한계
    - **Hallucination** : 존재하지 않는 아이템이나 근거를 만들어 냄
    - **최신 지식과 도메인 지식 부족** : 학습 이후에 나온 아이템 정보를 모름
    - 일반적인 RAG는 **노이즈가 섞이고 지식의 구조 관계를 놓침**
  - ✓ 해결 방향 : 지식 그래프에서 **구조 정보를 검색해** 추천 생성에 결합

- **대표 연구**
  - ✓ **K-RagRec** : Knowledge Graph Retrieval-Augmented Generation for LLM-based Recommendation
  - ✓ Wang et al., ACL 2025. KG의 구조 정보를 검색해 LLM 추천에 주입하는 프레임워크

- **남아 있는 문제**
  - ✓ 어떤 부분 그래프를 얼마나 가져올 것인가
  - ✓ 검색한 지식과 사용자 선호를 어떻게 결합할 것인가
  - ✓ 그래프가 커질수록 **검색 비용과 노이즈**가 함께 증가

- **오늘 본 논문과의 연결**
  - ✓ AdaptJobRec도 People.AI 지식 그래프를 **도구로 호출**
  - ✓ 즉 KG는 이미 에이전트의 도구 자리에 들어와 있음

[2025][ACL] Knowledge Graph Retrieval-Augmented Generation for LLM-based Recommendation (K-RagRec)

# Thank you for listening

Hae-Yoon Koo ( hykoo@konkuk.ac.kr )

Graph & Language Intelligence Laboratory
Konkuk University

2026.08.12