

AI기반 데이터 분석 및 AI Agent 개발 과정

『2과목 :』 AI Agent 개발

2025.09.22-10.02(9일, 62시간)

Prepared by Daekyeong Kim

Ph.D.

1. 과정 소개
 2. Langchain 전 프롬프트 엔지니어링
 3. Langchain 기본
 4. AI 에이전트 개발
 5. AI 에이전트 프로토콜 : MCP와 A2A
- 성취도 평가
 - 미니프로젝트

『2-4』 AI 에이전트 개발

- 오픈AI의 에이전트 SDK
- 구글의 ADK
- 랭그래프





학습목표

- 이 워크샵에서는 AI 에이전트 개발과 오픈AI의 에이전트 SDK에 대해 알 수 있다

눈높이 체크

- Agents 를 알고 계시나요?



1. Agents?

Agents의 기본 개념

- 랭체인(LangChain)의 Agents는 다양한 작업을 수행하기 위해 능동적으로 의사결정을 내리고 행동할 수 있는 시스템을 의미합니다. Agents는 주어진 목표를 달성하기 위해 여러 도구와 체인을 결합하여 다양한 작업을 처리할 수 있는 강력한 기능을 제공합니다. 이를 통해 더욱 복잡한 문제를 해결하고, 다양한 데이터 소스나 API와 상호작용할 수 있는 유연한 워크플로우를 만들 수 있습니다.
- LLM(대형 언어 모델): 자연어 처리와 의사결정의 기본 엔진으로 사용됩니다.
- Chain(체인): 여러 작업을 조합하여 복잡한 워크플로우를 구성할 수 있는 체인입니다.



1. Agents?

Agents의 주요 구성 요소

- Tools(도구): 계산, 검색과 같이 "언어 모델만으로는 할 수 없는 일"을 할 수 있게 하는 모듈로, 검색, 계산, API 호출, 데이터 변환 등의 기능을 수행합니다.
- 랭체인이 제공하는 Tool로는 LLMMath(계산), Requests(URL 호출), File System(로컬 파일 접근), SerpApi(검색) 등이 있다.
- 랭체인이 제공하는 것 외에도, 직접 만든 Tool을 사용할 수 있다.
- Agent의 역할: Agents는 사용자가 지정한 목표를 달성하기 위해 스스로 결정을 내리고, 적절한 도구나 체인을 선택하여 작업을 수행하는 엔티티입니다. Tool을 선택하고 다음 단계의 처리를 수행하는 주체로, 단순히 Tool을 조작하는 것이 아닌, 스스로 어떤 Tool을 어떻게 사용하면 좋을지 고민하고 실행, 결과 검증까지 진행한다.
- Agent에서 가장 많이 사용하는 'ReAct' 기법이 있다.



1. Agents?

Agents의 동작 원리

- 입력: 사용자가 에이전트에게 작업 요청을 전달합니다.
- 처리: 에이전트는 LLM을 사용하여 입력을 분석하고, 어떤 도구나 체인을 사용할지 결정합니다.
- 실행: 필요한 도구나 체인을 실행하여 결과를 생성합니다.
- 출력: 최종 결과를 사용자에게 반환합니다.



1. Agents?

Agents의 유형

- 랭체인에서는 다양한 유형의 Agents를 제공하여 사용자가 요구하는 다양한 작업을 수행할 수 있도록 합니다.
- Zero-Shot React Description
 - 설명: 주어진 작업에 대해 직접적이고 간단한 응답을 생성하는 에이전트입니다. 외부 도구를 사용하지 않고 언어 모델의 응답만으로 결과를 생성합니다.
 - 사용 사례: 간단한 질의응답, 빠른 텍스트 생성.
- ReAct
 - 설명: ReAct는 주어진 목표를 달성하기 위해 복합적인 작업을 수행하는 에이전트입니다. 이 에이전트는 상황에 따라 필요할 때마다 도구를 호출하거나 체인을 실행할 수 있습니다.
 - 사용 사례: 질문에 대한 답변을 위해 외부 정보를 검색하거나, 계산을 수행해야 하는 복잡한 시나리오.



1. Agents?

Agents의 유형

● Self-Ask with Search

- 설명: 질문을 처리하는 과정에서 스스로 추가적인 질문을 생성하여 검색을 수행하는 에이전트입니다.
- 사용 사례: 정확한 정보 검색이 필요한 상황에서 유용하며, 추가적인 세부사항을 알아야 할 때 활용됩니다.

● MRKL (Modular Reasoning, Knowledge and Language)

- 설명: MRKL 에이전트는 복잡한 의사결정과정을 수행하기 위해 다양한 모듈을 조합하여 동작하는 구조를 가지고 있습니다.
- 사용 사례: 특정 도메인 지식이 필요하고, 여러 개의 도구와 데이터 베이스를 활용해야 하는 상황에서 사용.

● Conversational Agent

- 설명: 대화형 에이전트로, 대화의 맥락을 이해하고, 필요한 경우 이전의 발화를 참고하여 답변을 생성합니다.
- 사용 사례: 챗봇, 고객 지원 시스템 등에서 사용자와의 상호작용을 위해 사용.



1. Agents?

Agents의 활용 예시

- 질의응답 시스템: 사용자로부터 질문을 받고, 관련 정보를 검색하고, 필요에 따라 데이터를 분석하거나 요약한 뒤 응답을 생성하는 시스템.
- 자동화 도구: 다양한 API를 호출하거나 데이터베이스와 상호작용하여 데이터를 가져오고, 이를 가공하여 사용자에게 유용한 정보를 제공하는 도구.
- 복잡한 문제 해결: 여러 단계로 이루어진 복잡한 문제를 해결하기 위해, 각 단계마다 필요한 작업을 정의하고, 이를 자동으로 수행하는 에이전트.



1. 실습

| 실습 17: 인사하는 에이전트

`hello_agent_async.py`

```
pip install openai-agents==0.2.5
```

2. Tool을 추가해 Agents 구현

Tool

- Agent가 할 수 있는 것은 전달하는 Tool에 따라 달라진다
- File System(로컬 파일 접근): WriteFileTool은 파일 쓰기를 할 수 있는 Tool이다.
- SerpApi(검색): 구글 등 검색 엔진의 검색 결과를 API로 가져올 수 있는 서비스
- <https://serpapi.com/manage-api-key> 에서 API키를 발급 받을 수 있다.
- AgentType를
STRUCTURED_CHAT_ZERO_SHOT_REACT_DESCRIPTION으로 변경하였는데, CHAT_ZERO_SHOT_REACT_DESCRIPTION와의 차이점은, 여러 개의 입력이 필요한 상황에서 작동한다는 점이다 (CHAT_ZERO_SHOT_REACT_DESCRIPTION은 단일 입력).



2. Tool을 추가해 Agents 구현

SerpApi

- SerpApi는 검색 엔진 결과 페이지(SERP: Search Engine Results Page)의 데이터를 수집하고 분석할 수 있는 API 서비스입니다. SerpApi는 Google, Bing, Yahoo, Yandex, Baidu 등 여러 검색 엔진에서 검색 결과를 수집하여 JSON 형식으로 반환해주는 API를 제공합니다. 이를 통해 개발자는 웹 스크래핑 없이 손쉽게 검색 엔진의 결과를 사용할 수 있습니다.

2. Tool을 추가해 Agents 구현

SerpApi의 주요 기능 및 활용

- 다양한 검색 엔진 지원: Google, Bing, Yahoo 등 주요 검색 엔진 외에도 지역별 검색 엔진, 이미지 검색, 뉴스 검색 등을 지원합니다.
- 세밀한 결과 추출: 검색 결과 페이지의 제목, 설명, URL, 이미지, 광고 등 다양한 정보를 정확하게 추출할 수 있습니다.
- 개발 편의성: 간단한 API 호출을 통해 원하는 정보를 얻을 수 있으며, 다양한 프로그래밍 언어에서 사용할 수 있는 SDK를 제공합니다.
- 활용 분야:
 - SEO 분석: 키워드 순위 추적, 경쟁 분석, 검색 엔진 알고리즘 연구 등
 - 데이터 수집: 특정 주제에 대한 방대한 양의 데이터를 수집하여 분석
 - 챗봇 개발: 사용자의 질문에 대한 답변을 검색 엔진에서 찾아 제공
 - 시장 조사: 제품, 브랜드, 트렌드 등에 대한 정보를 수집하여 시장 분석
 - 학술 연구: 특정 주제에 대한 연구 자료를 수집하고 분석

2. Tool을 추가해 Agents 구현

SerpApi Google Search API 발급 받기

- <https://serpapi.com/>에서 Register에서 회원가입 후 사용

The screenshot displays the SerpApi website interface. At the top, there's a navigation bar with links for Documentation, Integrations, Features, Pricing, Use cases, Team, FAQ, and Contact us. A 'Sign In' button and a 'Register' button are also present. The main heading is 'Google Search API', followed by the tagline 'Scrape Google and other search engines from our fast, easy, and complete API.' Below this, there's a search interface with a 'Search Query' field containing 'Coffee' and a 'Location' dropdown menu set to 'Austin, Texas, United States'. A green 'TEST SEARCH' button is to the right. The search results are shown in a preview window, displaying a Google search for 'Coffee' with a map of coffee shops in Austin, Texas, and a detailed description of coffee. To the right of the preview, a JSON response is shown, detailing the search metadata and parameters used.

```
{  "search_metadata": {    "id": "6165916694c6c7025deef5ab",    "status": "Success",    "json_endpoint": "https://serpapi.com/searches/87fa874d05a7fcc5/6165916694c6c7025deef5ab.json",    "created_at": "2021-10-12 13:45:10 UTC",    "processed_at": "2021-10-12 13:45:11 UTC",    "google_url": "https://www.google.com/search?q=Coffee&oeq=Coffee&uile=wCAIQCiaQXVzdGluFR1eGFz",    "raw_html_file": "https://serpapi.com/searches/87fa874d05a7fcc5/6165916694c6c7025deef5ab.html",    "total_time_taken": 1.85  },  "search_parameters": {    "engine": "google",    "q": "Coffee",    "location_requested": "Austin, Texas, United States",    "location_used": "Austin, Texas, United States",    "google_domain": "google.com",    "hl": "en",    "gl": "us",    "device": "desktop"  },}
```



2. Tool을 추가해 Agents 구현

특정 숫자 이상의 임의의 숫자를 생성하는 Tool

- Tool을 직접 만들어서 할 수 있는 일의 폭을 더욱 넓힌다
- 랭체인에서 제공하는 기본적인 Tool뿐만 아니라, 사용자가 직접 Tool을 제작할 수 있다.
- Tool은 기본적으로 '이름', '기능 설명', '실행 함수' 3가지 요소로 구성되어 있으며, 사용자의 Custom Tool도 같은 형태로 작성되어야 한다.

2. Tool을 추가해 Agents 구현

Retrievers를 사용해 문장 검색 Tool 만들기

- Retrievers를 Tool로 변환해 Agent에서 사용하도록 할 수 있다(위키피디아, 벡터DB에서 검색).
- WikipediaRetriever를 Tool로 지정(직접 만든 것은 아니고, create_retriever_tool 함수에서 미리 정의해둔 코드가 있다)

```
retriever = WikipediaRetriever(           ←WikipediaRetriever를 초기화
    lang="ko",           ←언어를 한국어로 설정
    doc_content_chars_max=500,           ←글의 최대 글자 수를 500자로 설정
    top_k_results=1      ←검색 결과 중 상위 1건을 가져옴
)
```



2. Tool을 추가해 Agents 구현

Retrievers를 사용해 문장 검색 Tool 만들기

- 명령문 입력

```
result = agent.run("스카치 위스키에 대해 Wikipedia에서 찾아보고 그 개요를 한국어로  
result.txt 파일에 저장하세요.")
```

- 여러개의 Document를 가져와서

➤ Entering new AgentExecutor chain...

Thought: I will use the WikipediaRetriever tool to search for information about Scotch whisky and then write a summary in Korean to a file named "result.txt".

Action:

```
...  
[  
  "action": "WikipediaRetriever",  
  "action_input": "Scotch whisky"  
]  
...
```

Observation: [Document(page_content='위스키(영어: whisky 또는 whiskey, 문화어: 위스끼)는 맥아 및 기타 곡류를 당화 발효시킨 발효주를 증류하여 만든 술이다. 주로 보리, 옥수수, 호밀, 밀 등의 곡물이

2. Tool을 추가해 Agents 구현

Retrievers를 사용해 문장 검색 Tool 만들기

- Retrieval task를 진행

```
Thought:Action:
...
{
  "action": "write_file",
  "action_input": {
    "file_path": "result.txt",
    "text": "위스키(영어: whisky 또는 whiskey, 문화어: 위스끼)는 맥아 및 기타 곡류를 당화 발효시킨 발효주를 증류하여 만든 술이다. 주로 보리, 옥수수, 호밀, 밀 등의 곡물이 원료가 된다. 증류 후에는 L
    "append": false
  }
}
...
Observation: File written successfully to result.txt.
...

> Finished chain.
실행 결과: 작업이 성공적으로 완료되었습니다.
```

- 결과를 result.txt에 저장

6장.ipynb U ● result.txt X

langchain > 06_agent > result.txt

1 위스키(영어: whisky 또는 whiskey, 문화어: 위스끼)는 맥아 및 기타 곡류를 당화 발효시킨 발효주를 증류하여 만든 술



2. Tool을 추가해 Agents 구현

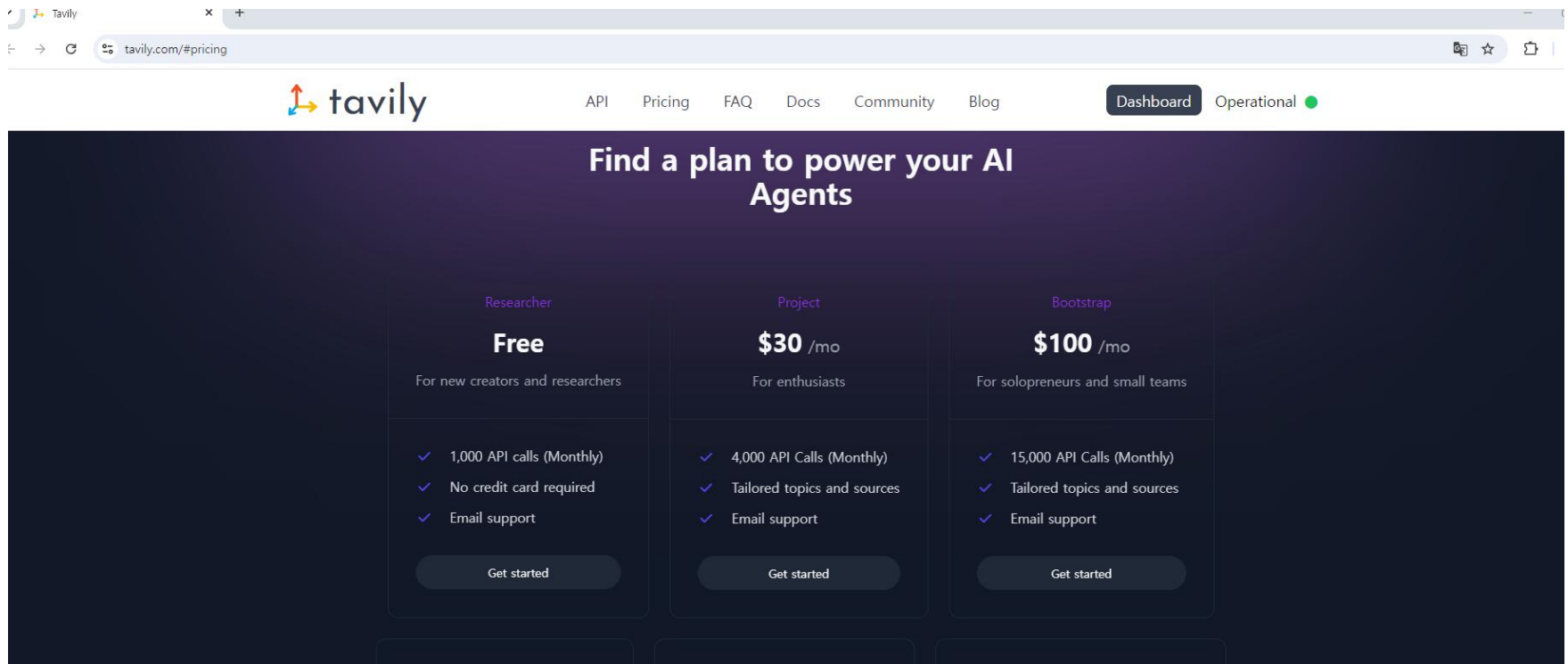
PDF 기반 문서 검색 도구: Retriever

- 우리의 데이터에 대해 조회를 수행할 retriever도 생성합니다.
- 이 코드는 웹 기반 문서 로더, 문서 분할기, 벡터 저장소, 그리고 OpenAI 임베딩을 사용하여 문서 검색 시스템을 구축합니다.

2. Tool을 추가해 Agents 구현

도구를 활용한 토론 에이전트(Agent Debates with Tools)

- <https://tavily.com/#pricing>



The screenshot shows the Tavily website's pricing page. The browser address bar displays 'tavily.com/#pricing'. The website header includes the Tavily logo, navigation links (API, Pricing, FAQ, Docs, Community, Blog), a 'Dashboard' button, and a status indicator 'Operational' with a green dot. The main heading reads 'Find a plan to power your AI Agents'. Below this, three pricing plans are presented in a grid:

Researcher	Project	Bootstrap
Free For new creators and researchers	\$30 /mo For enthusiasts	\$100 /mo For solopreneurs and small teams
<ul style="list-style-type: none">✓ 1,000 API calls (Monthly)✓ No credit card required✓ Email support	<ul style="list-style-type: none">✓ 4,000 API Calls (Monthly)✓ Tailored topics and sources✓ Email support	<ul style="list-style-type: none">✓ 15,000 API Calls (Monthly)✓ Tailored topics and sources✓ Email support
Get started	Get started	Get started



2. 실습

| 실습 18: 뉴스 에이전트 만들기

- `news_search_agent.py`



2. 실습

| 실습 19: 가드레일 사용하기

- `input_output_guardrail_test.py`



2. 실습

| 실습 20: 핸드오프 활용

- `simple_multi_agent_by_handoff.py`

『2-4』 AI 에이전트 개발

- 오픈AI의 에이전트 SDK
- 구글의 ADK
- 랭그래프



학습목표

- 이 워크샵에서는 Google의 ADK(Agent Development Kit)의 구조와 주요 구성 요소를 파악합니다.

눈높이 체크

- ADK의 주요 구성 요소(LLM, Tool, Agent)와 역할



1. AI 에이전트, 새로운 패러다임의 시작

왜 지금 AI 에이전트인가?

- 단순한 챗봇을 넘어: 사용자의 요청을 이해하고, 계획을 세우고, 스스로 행동하는 AI
 - AI 에이전트의 산업별 활용 사례
 - 고객 지원 자동화: 24시간 고객 문의 응대 및 문제 해결
 - 데이터 분석 자동화: 복잡한 데이터 조회 및 리포트 생성
 - 개인 비서: 이메일 정리, 일정 관리, 정보 검색 등

1. AI 에이전트, 새로운 패러다임의 시작

AI 에이전트의 핵심 원리: ReAct 프레임워크

- ReAct: Reason(추론) + **Act(행동)**의 결합
- LLM이 단순히 답변을 생성하는 것을 넘어, '생각'하고 '행동'하게 만드는 방법
- 작동 방식 (순환 구조)
 - Thought (생각): 사용자의 목표를 달성하기 위해 어떤 행동이 필요한지 추론한다.
 - Action (행동): 가장 적절한 도구(Tool)를 선택하고 실행한다.
 - Observation (관찰): 도구 실행 결과를 관찰(입력)받는다.
 - Thought (생각): 관찰된 결과를 바탕으로 다음 행동을 다시 추론하거나, 최종 답변을 생성한다.

1. AI 에이전트, 새로운 패러다임의 시작

| Google ADK 소개

- ADK란?: Gemini 모델을 기반으로 강력하고 확장 가능한 AI 에이전트를 쉽게 구축할 수 있도록 돕는 개발 프레임워크
- ADK의 장점:
 - Tool 사용의 단순화: 외부 API나 함수를 '도구'로 쉽게 등록하고 LLM이 사용하도록 할 수 있음.
 - 추론 능력 강화: ReAct 패턴을 내장하여 복잡한 문제 해결 능력을 극대화.
 - Google 생태계 연동: Google 검색, Google Workspace 등 다양한 구글 서비스와 손쉽게 연동 가능.

2. ADK 핵심 구성 요소 파헤치기

| LLM: 에이전트의 '두뇌'

- 에이전트의 모든 추론과 계획을 담당하는 핵심 엔진 (예: Gemini)
- 사용자의 의도를 파악하고, 어떤 도구를 사용해야 할지 결정하는 역할



2. ADK 핵심 구성 요소 파헤치기

Tools: 에이전트의 '손과 발'

- 에이전트가 세상과 상호작용할 수 있게 해주는 모든 수단.
- 예시:
 - 특정 웹사이트의 정보를 가져오는 함수
 - 데이터베이스에서 데이터를 조회하는 API
 - 사용자에게 이메일을 보내는 기능

2. ADK 핵심 구성 요소 파헤치기

Tools: 에이전트의 '손과 발'

- Tool 정의 방법: Python 함수에 @tool 데코레이터만 추가하면 간단하게 정의 가능.

예시: 날씨 정보를 가져오는 Tool
from adk.tools import tool

```
@tool
def get_weather(city: str) -> str:
    """주어진 도시의 현재 날씨 정보를 알려줍니다."""
    # ... (실제 날씨 API 호출 코드) ...
    return f"{city}의 현재 날씨는 맑고, 기온은 25도입니다."
```




2. ADK 핵심 구성 요소 파헤치기

Agent: '두뇌'와 '손과 발'을 연결하는 실행자

- 사용자의 입력을 받아 LLM에게 전달하고, LLM의 계획에 따라 Tool을 실행시키는 컨트롤 타워.
- ADK에서는 Agent 클래스를 사용하여 간단하게 생성 가능.

예시: Agent 생성 및 실행

```
from adk.agent import Agent
from adk.llm import Gemini
```

1. 에이전트의 두뇌(LLM) 정의

```
llm = Gemini()
```

2. 에이전트가 사용할 도구들 정의

```
tools = [get_weather]
```

3. 에이전트 생성

```
agent = Agent(llm=llm, tools=tools)
```

4. 에이전트 실행

```
result = agent.run("오늘 서울 날씨 어때?")
print(result)
```

'오늘의 브리핑' 에이전트 만들기

● 목표

- 오늘 날씨와 주요 뉴스 헤드라인을 요약해서 알려주는 에이전트를 직접 만들어 봅니다.

1. 프로젝트 환경 설정

- Python 가상 환경 설정 및 라이브러리 설치 (pip install google-adk google-generativeai requests)
 - # Windows CMD:
 - python -m venv .venv
 - .venv\Scripts\activate.bat
 - ADK 설치
 - pip install google-adk
 - (선택 사항) 설치 확인:
 - pip show google-adk
- 날씨 API 및 뉴스 API 키 발급 (무료 API 활용)

'오늘의 브리핑' 에이전트 만들기

2. Tools 구현하기

- `get_weather(city: str)` Tool:
 - `requests` 라이브러리를 사용하여 OpenWeatherMap 같은 날씨 API를 호출하고, 결과를 가공하여 문자열로 반환하는 함수 작성
- `get_top_news(category: str)` Tool:
 - NewsAPI 같은 뉴스 API를 호출하여 특정 카테고리(예: 'technology')의 상위 3개 뉴스 기사 제목을 가져오는 함수 작성

'오늘의 브리핑' 에이전트 만들기

3. Agent 로직 구현하기

- Gemini 모델을 LLM으로 설정합니다.
- 앞에서 만든 `get_weather`, `get_top_news` 두 가지 Tool을 리스트로 묶어 Agent를 생성합니다.
- 사용자의 질문을 받아 Agent를 실행하고, 최종 결과를 출력하는 코드를 완성합니다.

4. 에이전트 테스트 및 디버깅

- 질문 예시:
 - "서울 날씨 알려줘" (날씨 Tool만 사용)
 - "오늘의 IT 분야 주요 뉴스 3가지만 요약해줘" (뉴스 Tool만 사용)
 - "오늘 부산 날씨랑 경제 뉴스 헤드라인 같이 알려줄래?" (두 가지 Tool을 모두 사용하여 복합적인 답변 생성)
- 에이전트의 중간 추론 과정(Thought)을 출력하여 어떻게 문제를 해결하는지 관찰하고 디버깅합니다.

『2-4』 AI 에이전트 개발

- 오픈AI의 에이전트 SDK
- 구글의 ADK
- 랭그래프





학습목표

- 이 워크샵에서는 LangGraph 에 확인할 수 있다

눈높이 체크

- LangGraph 를 알고 계시나요?



1. LangGraph?

LangGraph?


- Agent 와 도구(Tool)를 이용하면 LLM 의 추론능력으로 자동화된 작업 처리능력을 얻을 수 있습니다. 사용자의 문제를 해결하기 위하여 Agent는 자신에게 주어진 도구들을 이용해서 일련을 작업들을 구성해냅니다.
- 이런 강력한 Agent 도 만능은 아닙니다. 하나의 Agent 가 많은 도구를 사용하여 사이즈가 큰 작업을 처리해야하는 경우 Agent 의 성능이 저하됩니다. 따라서 Agent 를 기능/작업 단위로 구분하고 여러 Agent 가 상호 소통하면서 문제를 처리해야 할 필요가 있습니다. 이를 Multi-Agent 구조라고 합니다.
- LLM 을 이용한 어플리케이션 구성에서 Multi-Agent 구조를 활용하는 이론은 아래 논문에서 제시되었으며, LangChain 라이브러리에서는 LangGraph 로 구현되었습니다.




1. LangGraph?

LangGraph?

- <https://arxiv.org/abs/2308.08155>



 > cs > arXiv:2308.08155

Search... All fields Search

Help | Advanced Search

Computer Science > Artificial Intelligence

[Submitted on 16 Aug 2023 (v1), last revised 3 Oct 2023 (this version, v2)]

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

[Qingyun Wu](#), [Gagan Bansal](#), [Jieyu Zhang](#), [Yiran Wu](#), [Beibin Li](#), [Erkang Zhu](#), [Li Jiang](#), [Xiaoyun Zhang](#), [Shaokun Zhang](#), [Jiale Liu](#), [Ahmed Hassan Awadallah](#), [Ryen W White](#), [Doug Burger](#), [Chi Wang](#)

AutoGen is an open-source framework that allows developers to build LLM applications via multiple agents that can converse with each other to accomplish tasks. AutoGen agents are customizable, conversable, and can operate in various modes that employ combinations of LLMs, human inputs, and tools. Using AutoGen, developers can also flexibly define agent interaction behaviors. Both natural language and computer code can be used to program flexible conversation patterns for different applications. AutoGen serves as a generic infrastructure to build diverse applications of various complexities and LLM capacities. Empirical studies demonstrate the effectiveness of the framework in many example applications, with domains ranging from mathematics, coding, question answering, operations research, online decision-making, entertainment, etc.

Comments: 43 pages (10 pages for the main text, 3 pages for references, and 30 pages for appendices)

Subjects: **Artificial Intelligence (cs.AI)**; Computation and Language (cs.CL)


Cite as: [arXiv:2308.08155 \[cs.AI\]](#)
(or [arXiv:2308.08155v2 \[cs.AI\]](#) for this version)
<https://doi.org/10.48550/arXiv.2308.08155>

Submission history

From: Qingyun Wu [[view email](#)]
[v1] Wed, 16 Aug 2023 05:57:52 UTC (2,420 KB)
[v2] Tue, 3 Oct 2023 20:47:10 UTC (6,271 KB)

Access Paper:

- [View PDF](#)
- [TeX Source](#)
- [Other Formats](#)

 [view license](#)

Current browse context:

cs.AI

< prev | next >

[new](#) | [recent](#) | [2023-08](#)

Change to browse by:

[cs](#)


[cs.CL](#)

References & Citations

- [NASA ADS](#)
- [Google Scholar](#)
- [Semantic Scholar](#)

Export BibTeX Citation

Bookmark





1. LangGraph?

LangGraph?

- LangGraph 가 Multi-Agent 구현만을 위한 도구는 아닙니다. LLM 을 이용한 코딩 작업을 모듈 단위로 나누어서 연동하는데 사용할 수도 있고, 비교적 큰 LLM 개발 프로젝트를 할 때 구조를 효율적으로 만들기 위한 도구로 사용할 수도 있습니다. 이를 위해 LangGraph 에서는 노드, 엣지, 워크플로우 등의 개념을 사용하여 작업을 모듈화 할 수 있도록 도와줍니다. 또한 사용자가 만든 구조를 시각화 하기 위한 도구도 제공됩니다



1. LangGraph?

LangGraph 의 기능

- 시각화
 - 에이전트와 도구, 데이터 흐름 등을 노드와 에지로 표현하여 시스템의 전체적인 구조를 시각적으로 보여줍니다.
- 관리
 - 에이전트와 도구를 생성, 수정, 삭제하고, 이들 간의 연결 관계를 관리할 수 있습니다.
- 디버깅
 - 시스템에서 발생하는 문제를 찾아내고 해결하는 데 도움을 줍니다.
- 저장 및 로딩
 - 생성된 그래프를 저장하고 다시 로딩하여 재사용할 수 있습니다.



1. LangGraph?

LangGraph 사용 이점

- 복잡성 관리
 - 복잡한 에이전트 시스템을 간단한 그래프로 표현하여 시스템의 이해를 돕습니다.
- 오류 감지
 - 그래프를 통해 데이터 흐름의 문제점이나 불필요한 연결을 쉽게 찾아낼 수 있습니다.
- 시스템 설계
 - 새로운 에이전트를 추가하거나 기존 에이전트를 수정할 때 그래프를 기반으로 시스템 설계를 변경할 수 있습니다.
- 팀 협업
 - 팀원들과 시스템 구조를 공유하고 함께 시스템을 개발할 수 있습니다.

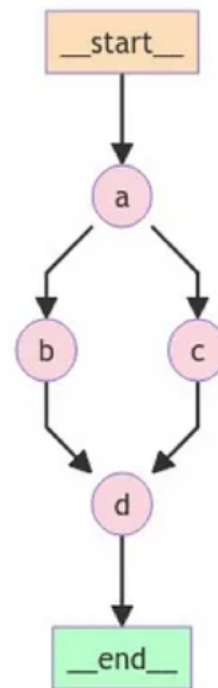


1. LangGraph?

LangGraph 개념

- LangGraph 가 사용하는 노드, 엣지, 빌더(워크플로우), 그래프. LangGraph 의 역할을 단순화하면, 작업을 모듈 단위로 나누어서 관리하고 서로간에 유기적으로 연동해서 문제를 해결할 수 있도록 하는 것

```
builder = StateGraph(State)
builder.add_node("a", ReturnNodeValue("I'm A"))
builder.set_entry_point("a")
builder.add_node("b", ReturnNodeValue("I'm B"))
builder.add_node("c", ReturnNodeValue("I'm C"))
builder.add_node("d", ReturnNodeValue("I'm D"))
builder.add_edge("a", "b")
builder.add_edge("a", "c")
builder.add_edge("b", "d")
builder.add_edge("c", "d")
builder.set_finish_point("d")
graph = builder.compile()
```





1. LangGraph?

LangGraph 개념

- 노드 (Node)
 - 특정 작업을 모듈화 한 구현체로 Agent, 함수 또는 클래스를 말합니다.
 - 따라서 작업이 실행되는 단위이기도 합니다.
 - 위 이미지에서 a, b, c, d 에 해당
- 엣지 (Edge)
 - 각 노드들을 이어주는 라인에 해당하며 노드간 작업의 전환을 의미합니다.
 - 노드가 작업을 마치면 실행해야할 다음 작업을 나타내는데 사용됩니다
 - 정적인 실행(add_edge), 조건부 실행(add_conditional_edges)을 사용할 수 있습니다



1. LangGraph?

LangGraph 개념

- 빌더, 워크플로우 (Builder, Workflow)
 - 위 이미지 좌측 코드에서 빌더로 표시되는 변수
 - 실제로 사용할 그래프의 타입에 따라 만들어지는 그래프 builder 입니다.
 - 노드나 엣지가 빌더에 등록됨으로써 서로 결합하여 실행될 준비가 됩니다
- 그래프 (Graph)
 - 빌더를 컴파일하면 생성되는 인스턴스입니다.
 - `graph.invoke()` 를 통해 그래프를 실행하면 설정한 노드와 엣지의 구조대로 작업을 수행합니다.
- 라우터 (Router)
 - `add_conditional_edges()` 를 통해 조건부로 노드간 연결을 할 때, 어떤 조건에서 어떤 노드가 연결되어야 할지에 대한 코드를 담고있는 함수가 라우터입니다.
 - 위 이미지에는 표시되지 않았지만, 실습 코드에서는 라우터 함수를 정의해서 사용합니다.



1. LangGraph?

LangGraph 주요 유형

- StateGraph
 - 상태 기반 워크플로우를 모델링하는 데 사용
 - 복잡한 의사결정 로직과 조건부 흐름을 구현
- MessageGraph
 - 메시지 기반 시스템을 모델링하는 데 사용
 - 메시지의 흐름과 변환을 추적하는 데 유용
- DAG (Directed Acyclic Graph)
 - 비순환 방향 그래프
 - 작업의 순서와 의존성을 모델링
- Channel
 - 비동기 통신을 위한 단순한 그래프 구조
- Pydantic Graph
 - Pydantic 모델을 사용하여 그래프의 노드와 엣지를 정의
 - 데이터 유효성 검사와 직렬화를 그래프 구조에 통합
- TypedGraph
 - 강력한 타입 검사를 제공하는 그래프 구조
 - 복잡한 시스템에서 타입 안전성을 보장



2. 실습

| 실습 21: 헬로 랭그래프 만들기

```
pip install langgraph==0.6.2 grandalf==0.8
```

```
hello_langgraph.py
```




2. 실습

| 실습 22: 감정 분석 챗봇에 적용해 보기

`conditional_routing.py`

THANK YOU.

앞으로의 엔지니어는 단순한 '코더'나 '기계 조작자'가 아니라 뇌-기계 인터페이스를 통해 지식과 능력을 즉각 확장하는 존재(뉴로-인터페이스: Neuro Interface)가 될 수 있습니다.

- 🎯 목표 달성을 위한 여정이 시작됩니다.
- 🌟 궁금한 점이 있으시면 언제든지 문의해주세요!
- 🚀 함께 코더와 프롬프트 전문가로 성장해 나갑시다!

