□□□□□□□□□□□□□□□ □□□□□□
□□□□□□□□□□ □□□□□□:
□□□□□□□□□□□□□□□□ □□□□□□□□□□□□

Contents

생명과학을 읽는 새로운 언어: 인공지능과 에이전트 3

서문. 다른 공부의 입구에 서서 3

1장. 생명과학의 작업대가 바뀐다 5
- 한 화면에 열린 생명과학 . . . 5
- 생물학은 데이터로 돌아온다 . . . 6
- 에이전트가 들어오는 자리 . . . 6
- 그래서 의도를 표현해야 한다 . . . 7
- 빠른 손보다 느린 판단 . . . 7

2장. 에이전트와 함께 일한다는 것 8
- 질문에서 작업으로 . . . 8
- 빠른 루프와 느린 실험 . . . 9
- 경계를 정하는 연구자 . . . 10
- 기록으로 남는 협업 . . . 10

3장. 의생명과학 학생에게 남는 질문 12
- 답이 쉬워질 때 남는 공부 . . . 12
- 생명 데이터와 모델의 만남 . . . 12
- 큰 모델을 의심하는 법 . . . 13
- 개입을 묻는 과학 . . . 13
- 자기 질문을 만드는 대학 . . . 15

4장. ChatGPT는 무엇을 하고 있는가 16
- 화면의 문장과 안쪽의 계산 . . . 16
- 다음 조각을 고르는 모델 . . . 16
- 베이스 모델과 어시스턴트 . . . 17
- 답변을 실험처럼 읽기 . . . 18
- 작은 실험으로 익히기 . . . 18

5장. 텍스트가 토큰과 확률이 되는 과정 19
- 사람이 읽는 문장과 모델이 보는 조각 . . . 19
- 토큰은 단어가 아니다 . . . 20
- 확률로 이어지는 문장 . . . 20
- 문맥 속에서 바뀌는 의미 . . . 21
- 정확한 표기를 지키는 습관 . . . 22

6장. 인터넷 문서에서 베이스 모델까지 23

웹의 문서가 훈련 데이터가 되기까지 . . . 23
다음 토큰을 맞히는 긴 훈련 . . . 23
베이스 모델이라는 첫 결과 . . . 24
데이터의 그림자와 규모의 부담 . . . 25
좋은 데이터에서 좋은 모델로 . . . 26

7장. 대화 데이터로 어시스턴트가 된다 **27**
많이 읽은 모델은 아직 조교가 아니다 . . . 27
예시로 행동을 가르치기 . . . 27
대화도 토큰의 줄이 된다 . . . 28
친절함을 믿기 전에 . . . 28
AI 튜터와 자기 문장 . . . 29

8장. LLM의 기억, 착각, 환각 **30**
기억처럼 보이는 압축 . . . 30
자연스럽게 틀리는 문제 . . . 30
모른다고 말하게 하기 . . . 31
눈앞의 자료로 답하게 하기 . . . 31
검증하는 글쓰기 . . . 33

9장. 문맥 창과 도구 사용 **34**
기억과 책상 사이 . . . 34
문맥 창을 정리하는 일 . . . 34
도구마다 맡길 일이 다르다 . . . 35
RAG와 LLM-Wiki의 차이 . . . 35
에이전트에게 맡길 때 남길 것 . . . 36

10장. 모델도 생각할 시간이 필요하다 **38**
머릿속 계산의 한계 . . . 38
모델에게도 칠판이 필요하다 . . . 38
계산은 도구로 확인한다 . . . 39
내부 생각보다 외부 기록 . . . 40
좋은 과정이 빠른 답보다 낫다 . . . 40

11장. 강화학습과 추론 모델 **41**
예제를 따라 하는 공부 다음 . . . 41
정답을 확인할 수 있는 문제 . . . 41
생물학의 느린 채점 . . . 42
추론 모델을 어떻게 쓸까 . . . 43
더 똑똑하게 묻는 학생 . . . 44

12장. 의생명과학 학생을 위한 LLM 사용 원칙 **44**
도구는 권위가 아니다 . . . 44
자료와 계산을 따로 확인하기 . . . 46
원본과 기록을 지키기 . . . 46
점수보다 실제 질문을 보기 . . . 47
자기 말로 돌아오기 . . . 47

용어 지도와 표기 약속 **49**

참고와 인용 **50**
더 읽을거리 . . . 51

생명과학을 읽는 새로운 언어: 인공지능과 에이전트

안준용 고려대학교 보건과학대학 바이오시스템의과학부

이 책은 고려대학교 보건과학대학 바이오시스템의과학부 1학년 세미나에서 학생들과 함께 읽기 위해 준비한 글입니다. 갓 대학에 들어온 학생들이 ChatGPT와 LLM을 단순한 검색창이나 과제 도구로만 보지 않고, 앞으로의 공부와 연구 환경을 바꾸는 중요한 기술로 이해할 수 있기를 바라는 마음에서 시작했습니다.

첫 독자는 대학 신입생이지만, 이 책은 중학교 과학 교실, 과학책 독서 모임, 과학 다큐멘터리 기획 회의에서도 함께 읽을 수 있습니다. 전문 연구자가 아니어도 괜찮습니다. 중요한 것은 모델 이름을 많이 아는 일이 아니라, AI가 만든 문장을 한 문장씩 붙들고 "무엇을 보았기에 이렇게 답했나", "이 말은 어디서 확인할 수 있나", "내가 모르는 용어는 무엇인가"를 묻는 태도입니다.

본문은 안드레이 카파시의 공개 강의와 인터뷰를 중요한 출발점으로 삼습니다. 다만 강의록을 그대로 옮긴 번역본은 아닙니다. 카파시가 설명한 LLM의 큰 흐름과 직관을 바탕으로, 의생명과학을 처음 배우는 한국어 독자가 강의자료와 실습 데이터에서 출발해 나중에는 논문, 코드, AI 도구까지 함께 생각할 수 있도록 다시 풀어 쓴 해설서입니다.

이 책은 먼저 생명과학을 배우는 화면과 작업대가 AI와 데이터, 에이전트로 어떻게 바뀌고 있는지 살펴봅니다. 그다음 ChatGPT가 답을 만들어내는 원리, 이 모델이 잘하는 일과 자주 틀리는 일, 자료와 도구를 함께 주었을 때 달라지는 점, 그리고 학생이 책임 있게 사용하는 방법을 차례로 다룹니다. 자세한 문제의식은 다음 서문에 이어서 적었습니다.

서문. 다른 공부의 입구에 서서

대학에 들어오기 전에도 여러분은 이미 ChatGPT를 써보았을지 모릅니다. 어려운 영어 문장을 풀어달라고 했거나, 수행평가 글을 어떻게 시작하면 좋을지 물어보았거나, 생물학 용어를 쉬운 말로 설명해달라고 했을 수 있습니다. 답변이 너무 빨리 나오면 마음이 놓이기도 하지만, 동시에 이상한 불안도 생깁니다. 이 설명을 어디까지 믿어도 될까요. 과제를 빨리 끝내는 데는 도움이 되는데, 정말 내가 이해한 것일까요. 이 책은 바로 그 불안에서 출발합니다. 의생명과학을 공부하다 보면 세포, 유전자, 단백질, 질병 기전처럼 오래전부터 생명과학을 이루어온 말들과 함께, 강의자료실에 올라온 슬라이드, 실습용 표, 낯선 그래프, 간단한 Python 예제 같은 새 물건도 만나게 됩니다. 지금 이 단어와 도구를 모두 알고 있어야 한다는 뜻은 아닙니다. Python은 과학 데이터와 표를 다룰 때 자주 쓰는 프로그래밍 언어이고, RNA-seq은 세포나 조직 안에서 어떤 유전자가 얼마나 읽히는지 살펴보는 실험 방법이며, single-cell atlas는 세포 하나하나의 정보를 모아 만든 큰 지도에 가깝다는 정도만 붙잡아도 충분합니다. 처음에는 생물학과 인공지능이 따로 있는 것처럼 보입니다. 그러나 수업 자료와 실습 파일을 따라가다 보면 둘은 이미 같은 화면에 놓입니다. 슬라이드에는 유전자 이름이 나오고, 엑셀 파일에는 측정값이 있고, 검색창에는 모르는 용어가 입력되며, 옆에는 ChatGPT의 대화창이 열립니다. 이 책은 바로 그 화면 앞에 처음 앉는 학생들을 위해 쓴 글입니다.

저는 이 책을 고려대학교 보건과학대학 바이오시스템의과학부 1학년 세미나에서 지도 학생들과 함께 읽기 위해 준비하고 있습니다. 그러니 이 책의 첫 독자는 의생명과학을 막 배우기 시작한 대학 1학년입니다. 옆 학과 자연계열 학생이나 의생명 분야를 꿈꾸는 고등학생도 많은 장을 따라올 수 있지만, 본문에는 앞으로 연구와 논문에서 실제로 만나게 될 전문 용어가 남아 있습니다. 낯선 단어가 나온다고 해서 그 자리에서 모두 외우려 하지 않아도 됩니다. 중요한 말은 처음 등장할 때 풀어 쓰고, 당장 몰라도 되는 이름은 그렇게 말해둘 것입니다. 우리 학부가 가르치는 의생명과학은 한 층위에 머물지 않습니다. 세포와 분자 수준에서 생명현상을 분석적으로 이해하는 일에서 출발하지만, 그 지식은 질병의 예방과 진단, 치료 전략, 바이오마커 발굴, 바이오헬스 산업으로 이어집니다. 유전학, 분자세포생물학, 면역학, 종양학, 신경과학, 마이크로바이옴, 줄기세포 같은 분야는 서로 떨어져 있지 않습니다. 현대의 의생명 연구에서는 이 분야들이 데이터와 모델을 사이에 두고 계속 만납니다. 그래서 이제 의생명과학 학생에게 데이터 과학과 인공지능은 부가적인 기술을 넘어, 생명현상을 읽는 또 하나의 언어가 되고 있습니다.

이 책을 꼭 한 가지 속도로만 읽을 필요는 없습니다. 고등학생 독자라면 1부와 8장, 12장을 먼저 읽어도 좋습니다. AI가 왜 단순한 검색창이 아닌지, 왜 매끄러운 답변을 그대로 믿으면 안 되는지, 수행평가와 탐구 활동에서 어떤 습관을 가져야 하는지 먼저 잡을 수 있기 때문입니다. 의생명과학을 전공하는 1학년이라면 3장, 9장, 12장에서 앞으로 논문과 데이터와 AI가 어떻게 연결되는지 조금 더 오래 머물러도 좋습니다. 비전공 1학년 독자라면 낯선 유전자 이름이나 실험 용어 앞에서 너무 오래 멈추지 않아도 됩니다. 그 이름들은 예시의 옷을 입고 있을 뿐이고, 더 중요한 질문은 "AI가 무엇을 보고 답했는가", "이 답은 어디서 확인할 수 있는가", "나는 이 설명을 내 말로 다시 할 수 있는가"입니다. 책을 읽다가 모르는 단어를 만났을 때 곧바로 포기하지 말고, 그 단어가 지금 반드시 필요한 개념인지, 아니면 뒤에서 다시 만나도 되는 이름인지

먼저 가늠해보십시오. 대학 공부는 모든 것을 한 번에 이해하는 일이 아니라, 여러 번 돌아오며 점점 더 선명하게 보는 일에 가깝습니다.

이 말이 곧 모든 학생이 컴퓨터공학자가 되어야 한다는 뜻은 아닙니다. 의생명과학 학생에게 더 중요한 것은 생물학의 질문과 데이터·AI의 절차를 번갈아 놓고 확인하는 일입니다. 수업에서 받은 작은 표 하나도 그냥 숫자의 모음이 아닙니다. 어떤 조건에서 측정했는지, 단위가 무엇인지, 빠진 값은 없는지, 서로 비교해도 되는 값인지 먼저 보아야 합니다. ChatGPT의 답변도 마찬가지입니다. 그것은 사람 연구자가 직접 근거를 확인해 쓴 문장이 아니라, 모델이 학습한 패턴과 현재 주어진 자료를 바탕으로 생성한 문장입니다. 데이터와 AI를 무조건 의심하자는 말이 아닙니다. 오히려 잘 쓰기 위해서입니다. 실험 결과를 해석하려면 실험의 조건을 알아야 하듯, LLM의 답변을 해석하려면 모델이 어떤 방식으로 배웠고 어떤 조건에서 답하고 있는지 알아야 합니다. 이 책에서 LLM의 원리를 배우려는 이유도 바로 여기에 있습니다. 원리는 시험을 위한 지식이 아니라, 강력한 도구를 책임 있게 쓰기 위한 최소한의 지도입니다.

이 책의 가장 중요한 출발점은 안드레이 카파시의 공개 강의 Deep Dive into LLMs like ChatGPT입니다 (링크). 카파시는 딥러닝과 컴퓨터 비전 분야에서 널리 알려진 연구자이자 교육자입니다. Stanford에서 박사과정 동안 CS231n, 곧 컴퓨터 비전을 위한 딥러닝 강의를 설계하고 가르쳤고, OpenAI의 창립 멤버로 연구했으며, Tesla에서는 Autopilot 컴퓨터 비전 팀을 이끌었습니다 (링크). 그러나 이 책에서 카파시가 중요한 이유는 단지 이력 때문만은 아닙니다. 그는 복잡한 AI 시스템을 설명할 때 독자를 수식의 숲으로 바로 밀어 넣지 않습니다. 먼저 무엇이 실제로 일어나고 있는지, 우리가 화면에서 보는 답변 뒤에 어떤 과정이 숨어 있는지, 왜 그 과정이 때로는 놀랍고 때로는 위험한지를 차례로 보여줍니다. LLM을 처음 배우는 학생에게는 바로 그런 설명의 태도가 필요합니다. 그래서 이 책은 카파시의 강의와 인터뷰를 중요한 자료로 삼되, 의생명과학 1학년 학생이 읽을 수 있는 한국어 산문으로 다시 옮겨오려 합니다.

이 책은 LLM을 처음부터 만드는 법을 가르치는 책도 아니고, 최신 모델의 성능표를 줄 세우는 책도 아니며, 특정 회사의 도구를 홍보하는 책도 아닙니다. 목표는 더 작고, 그래서 더 실제적입니다. ChatGPT를 이미 쓰고 있거나 곧 쓰게 될 의생명과학 학생이, 이 도구를 단순한 검색창이나 과제 지름길로 오해하지 않고 자신의 공부와 연구의 조건 안에서 읽을 수 있게 돕는 것입니다. 1부에서는 생명과학을 배우는 작업대가 데이터와 AI, 에이전트로 어떻게 바뀌는지에서 출발해, 코딩이 의도 표현과 감독의 문제로 넓어지는 변화를 봅니다. 2부에서는 텍스트가 토큰과 확률이 되어 모델 안으로 들어가는 과정을 천천히 따라갑니다. 3부에서는 왜 모델이 자연스럽게 틀릴 수 있는지, 왜 자료와 도구가 필요한지 살펴봅니다. 4부에서는 생각하는 모델과 에이전트를 실제 공부와 연구 보조에 쓸 때 어떤 기준을 가져야 하는지로 돌아옵니다. 길은 기술 안쪽으로 들어가지만, 목적지는 언제나 학생의 판단입니다.

카파시가 최근 인터뷰에서 말하는 변화 중 특히 중요한 것은, AI를 단순히 더 똑똑한 검색창으로 보지 않는다는 점입니다. 그는 이제 "code"라는 동사 자체가 예전만큼 정확하지 않을 수 있다고 말합니다. 사람이 하루 종일 코드를 직접 치기보다, 자신이 만들고 싶은 것을 자연어로 설명하고, 여러 에이전트(agent)가 그 의도를 받아 작업하며, 사람은 결과를 검토하고 방향을 다시 잡는 일이 커지고 있기 때문입니다 (링크). 어떤 사람들은 이 흐름을 vibe coding이라고 부릅니다. 직접 문법을 한 줄씩 치기보다 원하는 결과를 말하고, AI가 만든 코드나 화면의 초안을 보며 다시 지시하는 작업 방식이라는 뜻입니다. 카파시는 의도 구현(manifesting)이라는 표현도 씁니다. 유행어처럼 들릴 수 있지만, 이름보다 중요한 변화는 작업의 중심이 문법에서 의도로, 손끝의 구현에서 결과의 설계와 감독으로 조금씩 이동하고 있다는 사실입니다. 예전에는 파일을 읽는 첫 줄의 코드에서 막혀 질문까지 가지 못하는 학생이 많았습니다. 이제는 AI가 그 문턱의 일부를 낮춰줄 수 있습니다. 그러면 학생은 더 빨리 "나는 이 데이터에서 무엇을 보고 싶은가"라는 질문 앞에 서게 됩니다.

그렇다고 공부가 사라지는 것은 아닙니다. 오히려 공부의 책임이 다른 곳으로 이동합니다. AI가 논문을 요약해준다고 해서 논문 읽기가 끝나는 것은 아닙니다. AI가 코드를 만들어준다고 해서 분석을 이해한 것도 아닙니다. AI가 그럴듯한 질병 기전 설명을 써준다고 해서 그 설명이 실제 근거를 가진 것도 아닙니다. 도구가 강력해질수록, 사용자는 그 도구가 무엇을 잘하고 무엇을 못하는지 더 정확히 알아야 합니다. 의생명과학에서는 이 문제가 더 무겁습니다. 우리가 다루는 것은 단순한 문장이 아니라 유전자 이름, 변이 표기, 환자 정보, 질병 설명, 통계적 결론일 때가 많습니다. 자연스러운 문장과 믿을 만한 문장은 다릅니다. AI가 말을 잘할수록, 학생은 더 차분하게 확인하는 습관을 가져야 합니다.

이 책이 처음부터 Transformer 구조나 attention 수식으로 들어가지 않는 이유도 여기에 있습니다. 물론 그 원리들은 중요합니다. LLM은 문장을 통째로 읽는 것이 아니라 토큰(token)이라고 부르는 작은 글자 조각으로 나누어 읽고, 지금까지의 조각들을 바탕으로 다음 조각이 무엇일지 예측합니다. 그 예측이 반복되면서 우리가 보는 답변이 만들어집니다. Transformer는 요즘 LLM의 뼈대가 되는 신경망 구조이고, attention은 그 구조 안에서 문장의 어떤 부분을 더 참고할지 계산하는 장치입니다. 그러나 갓 고등학교를 졸업한 학생에게 처음 필요한 질문은 "attention의 수식은 무엇인가"보다 "앞으로 나는 AI와 어떤 방식으로 공부하고 연구하게 될 것인가"일 수 있습니다. 그래서 이 책은 먼저 패러다임의 변화를 붙잡습니다. 생명과학을 배우는 작업대가 강의자료와 실습 파일에서 데이터, 코드, 에이전트로 넓어지고, 그 안에서 코딩이 의도 표현과 감독의 문제로 바뀌는 흐름을 먼저 살펴봅니다. 여기서 에이전트는 단순히 답만 말하는 챗봇이 아니라, 파일을 읽고 코드를 실행하고 결과를 고치는 여러 단계를 이어가려는 AI 시스템을 뜻합니다. 그다음에야 기술의 안쪽으로 천천히 내려갑니다. 원리를 배우되, 원리를 외우기 위해서가 아니라 AI와 함께 공부하는 사람이 어떤 판단을 해야 하는지 알기

위해서입니다.

수학이 약하다고 느끼는 학생도 이 책을 읽을 수 있기를 바랍니다. 어려운 수식을 모두 피하겠다는 뜻은 아닙니다. 대신 수식이 나오기 전에 그 수식이 왜 필요한지 말과 비유로 먼저 설명하겠다는 뜻입니다. 토큰화는 문장을 모델이 읽을 수 있는 작은 조각으로 바꾸는 과정입니다. 처음에는 긴 문장을 여러 장의 낱말 카드로 나누어 책상 위에 놓는 장면을 떠올리면 됩니다. 사전학습(pre-training)은 모델이 많은 글을 먼저 읽으며 언어와 지식의 배경 패턴을 배우는 단계입니다. 사람이 여러 논문과 교과서를 읽으며 배경지식을 쌓는 일에 비유해볼 수 있습니다. 지도 미세조정(supervised fine-tuning)은 좋은 질문과 답변 예시를 보여주며 어시스턴트다운 말투와 행동을 가르치는 단계입니다. 실험실 선배가 후배에게 좋은 설명의 형식을 보여주는 일과 닮았습니다. 강화학습(reinforcement learning)은 여러 시도를 해본 뒤 좋은 결과로 이어진 행동을 더 자주 하도록 훈련하는 방법입니다. 학생이 연습문제를 풀고 채점하면서 풀이 습관을 조금씩 고치는 일과 비교할 수 있습니다. 이런 비유는 완벽하지 않습니다. 다만 처음 배우는 사람에게는 발판이 필요합니다. 발판 위에 올라선 뒤에야 그 비유가 어디까지 맞고 어디서부터 조심해야 하는지도 보이기 시작합니다.

이 책은 카파시의 설명을 숨은 참고자료로만 두지 않겠습니다. 강의의 흐름이나 특정 예시를 따라가는 곳에는 가능한 한 가까운 자리에 영상 링크를 남깁니다. 다만 본문 안에서 시간표와 인용표기를 길게 늘어놓지는 않겠습니다. 독자의 읽기 흐름을 해치지 않도록 본문에는 짧은 링크만 남기고, 자세한 자료 목록은 마지막 참고와 인용에 모아두었습니다. 이 책은 번역서가 아닙니다. 카파시의 강의와 인터뷰에서 출발하되, 문장과 예시는 우리 수업의 자리로 옮겨옵니다. 강의자료 끝의 참고문헌을 처음 따라가 보는 학생, 유전자 목록을 처음 만지는 학생, Python 오류 앞에서 멈춘 학생, ChatGPT의 매끄러운 답변을 보며 어디까지 믿어야 할지 고민하는 학생을 떠올리며 다시 쓴 글입니다.

저는 이 책의 목표를 LLM 전문가 양성에 두지 않습니다. 이 책을 읽은 학생이 곧바로 모델을 처음부터 훈련하거나 복잡한 논문을 모두 이해하게 되리라 기대하지도 않습니다. 목표는 더 소박하지만 더 중요합니다. AI 도구를 두려워하지 않되, 무작정 믿지도 않는 태도를 배우는 것. 논문 요약을 받을 수는 있지만 원문으로 돌아가 확인해야 한다는 것을 아는 것. 코드 초안을 만들 수는 있지만 실행 결과와 분석 가정을 검토해야 한다는 것을 아는 것. 모델이 그럴듯한 설명을 만들 수 있지만, 생명과 질병에 관한 주장은 반드시 근거와 연결되어야 한다는 것을 아는 것. 이런 기준을 갖춘 학생은 AI를 금지된 지름길이 아니라 공부의 동반자로 사용할 수 있습니다. 그리고 언젠가 연구자가 되었을 때, 더 빠른 도구 앞에서도 더 느리고 정확한 판단을 잃지 않을 수 있습니다.

카파시가 말하는 변화는 사람이 사라진다는 이야기로 좁혀지지 않습니다. 사람이 직접 손으로 하던 많은 일이 모델과 에이전트와 도구로 이동할 때, 사람은 무엇을 더 잘해야 하는가라는 질문이 남습니다. 의도를 분명히 말하는 능력, 좋은 경계를 세우는 능력, 결과를 의심하고 확인하는 능력, 그리고 끝내 자기 말로 설명하는 능력입니다. 앞으로 AI 모델의 이름과 성능은 계속 바뀔 것입니다. 오늘의 ChatGPT가 내일의 표준이 아닐 수 있고, 에이전트의 형태도 빠르게 달라질 것입니다. 그러나 좋은 질문을 세우고, 근거를 확인하며, 책임 있게 판단하는 태도는 쉽게 낡지 않습니다. 이 책은 그 태도를 배우기 위한 작은 입구입니다. 이제 그 입구를 지나, 먼저 무엇이 바뀌고 있는지부터 천천히 살펴보겠습니다.

1장. 생명과학의 작업대가 바뀐다

한 화면에 열린 생명과학

대학에 들어와 처음 의생명과학 세미나에 앉은 학생을 떠올려봅니다. 책상 위에는 전공 교재 한 권만 놓여 있지 않습니다. 노트북 화면에는 강의자료실에서 내려받은 슬라이드, 실습용 엑셀 파일, 조별 과제 안내문이 나란히 열려 있습니다. 슬라이드 한쪽에는 낯선 유전자와 단백질 이름이 나오고, 다른 쪽에는 세포 사진과 그래프가 있습니다. 실습 파일을 열어 보니 표의 열 이름에는 샘플, 조건, 측정값, 평균 같은 말이 섞여 있습니다. 누군가가 공유한 예제 분석 파일에는 Python notebook, 곧 코드와 설명을 함께 남기는 분석 노트가 있고, 다른 탭에는 ChatGPT나 Claude 같은 대화창이 떠 있습니다. 아직 학생은 세포생물학도, 통계도, 코딩도 충분히 배우지 않았습니다. 그런데 현대 생명과학의 작업대는 이미 이렇게 생겼습니다. 교재, 강의자료, 데이터, 코드, 검색, 그림, 그리고 AI가 한 화면 안에 함께 올라옵니다.

이 장면은 공학 전공 학생에게만 해당하지 않습니다. 암세포가 약물에 어떻게 반응하는지 읽는 학생도, 면역세포가 감염 뒤에 어떤 유전자를 더 많이 읽는지 궁금한 학생도, 뇌 발달 논문에서 단일세포 RNA-seq 그림을 만난 학생도 비슷한 화면 앞에 앉습니다. 예전에는 생물학을 배운다는 말이 현미경, 해부도, 실험 프로토콜, 교과서 문장을 먼저 떠올리게 했습니다. 그 모든 것은 여전히 중요합니다. 다만 지금은 그 위에 다른 층이 얹혔습니다. 관찰은 파일이 되고, 파일은 표가 되며, 표는 코드와 모델을 지나 그림과 주장으로 바뀝니다. 생물학은 사라진 것이 아니라, 더 많은 매개를 거쳐 우리에게 돌아옵니다.

그래서 이 책의 첫 질문은 “코딩을 배워야 하는가”보다 조금 넓어야 합니다. 물론 코딩은 중요합니다. 그러나 더 앞에 놓이는 질문은 “생명과학의 작업대가 왜 이렇게 바뀌었는가”입니다. 학생이 앞으로 마주할 과학은 종이 교과서에 정리된 사실의 목록만이 아닙니다. 강의자료와 참고문헌에서 만난 문장, 대규모 실험에서 나온 숫자, 분석 코드가 만든 그림,

모델이 제안한 설명, 그리고 그 모든 것을 사람이 다시 확인하는 과정이 한데 얽힌 과학입니다. AI와 에이전트는 바로 이 얽힌 작업대 위로 들어옵니다.

생물학은 데이터로 돌아온다

생명과학에서 빅데이터라는 말은 단순히 파일이 크다는 뜻만은 아닙니다. 파일이 큰 것도 사실입니다. 유전체를 읽으면 염기서열 파일이 생기고, RNA를 읽으면 발현량 표가 생기며, 현미경으로 조직을 찍으면 이미지가 쌓입니다. 단일세포 RNA-seq은 세포 하나하나에서 어떤 유전자가 얼마나 읽혔는지 보려고 합니다. 그러면 행에는 세포가, 열에는 유전자가 놓인 거대한 표가 생깁니다. 세포가 수만 개이고 유전자가 수만 개라면, 그 표는 사람의 눈으로 한 칸씩 읽을 수 있는 물건이 아닙니다. 우리는 그 표를 줄이고, 묶고, 색칠하고, 비교하고, 다시 생물학적 의미로 번역해야 합니다.

용어 메모

단일세포 RNA-seq: 세포를 한 덩어리로 평균 내지 않고, 세포 하나하나에서 RNA 정보를 읽는 방법입니다.

유전자 발현: 유전자의 정보가 RNA로 읽혀 세포 안에서 실제로 쓰이는 정도를 말합니다.

빅데이터: 여기서는 단순히 큰 파일이 아니라, 여러 샘플과 조건과 측정값이 연결되어 사람 혼자 눈으로 읽기 어려운 자료를 뜻합니다.

이때 데이터는 생물학 바깥에서 온 장식이 아닙니다. 데이터는 생물학이 자기 모습을 드러내는 방식이 되었습니다. 병리 슬라이드의 이미지, 환자군의 유전 변이 표, 약물 처리 뒤의 세포 반응, 단백질 구조 예측, 공공 데이터베이스에 쌓인 논문과 실험 기록은 모두 질문의 일부가 됩니다. "이 유전자는 어떤 기능을 하는가"라는 질문은 이제 자주 "어떤 세포에서, 어떤 조건에서, 어떤 자료를 근거로 그렇게 말할 수 있는가"라는 질문을 데리고 옵니다. 생물학의 언어가 분자와 세포의 언어에서 데이터와 모델의 언어로 확장되는 것입니다.

이 변화는 학생에게 조금 불친절하게 느껴질 수 있습니다. 생물학을 좋아해서 들어왔는데 왜 표와 코드와 통계를 봐야 하는지 묻게 됩니다. 그러나 조금만 학년이 올라가도 이런 풍경을 만나게 됩니다. 논문 속 figure 하나를 제대로 읽으려 해도, 그 그림이 어떤 자료에서 나왔는지 알아야 합니다. 처리군과 대조군이 무엇인지, 표본 수가 충분한지, 배치 효과가 있을 수 있는지, 색이 다른 점들이 정말 다른 세포 유형인지, 아니면 분석 방법이 그렇게 나누었는지 물어야 합니다. 데이터는 답을 주기도 하지만, 동시에 새로운 의심을 요구합니다. 숫자가 많아졌다고 해서 판단이 자동으로 정확해지지는 않습니다.

여기에 AI 모델이 들어오면 풍경이 한 번 더 바뀝니다. 언어 모델은 영어 문단이나 수업 자료를 요약하고, 낯선 용어를 풀어주고, 분석 코드의 초안을 만들 수 있습니다. 생물학 자료를 학습한 모델은 유전자, 단백질, 세포 상태, 약물 반응처럼 생물학 자료 자체의 패턴을 배우려 합니다. 어떤 모델은 DNA 서열을 읽고 조절 요소를 예측하려 하고, 어떤 모델은 많은 단일세포 자료에서 세포 상태의 관계를 배우려 하며, 어떤 모델은 아직 해보지 않은 유전자 개입 뒤의 변화를 짐작하려 합니다. 이 시도들은 모두 완성된 답이 아니라 진행 중인 연구입니다. 하지만 방향은 분명합니다. 생명과학은 더 이상 실험실의 손작업과 컴퓨터 앞의 계산 작업으로 깔끔하게 나뉘지 않습니다. 두 일은 계속 서로를 부릅니다.

에이전트가 들어오는 자리

ChatGPT를 처음 쓰면 우리는 질문과 답변의 모양을 먼저 떠올립니다. "이 개념을 설명해줘." "이 문장을 번역해줘." "이 긴 영어 문단을 요약해줘." 이것만으로도 학생에게는 꽤 큰 변화입니다. 그런데 에이전트라는 말이 붙으면 장면은 더 움직입니다. 에이전트는 단지 한 번 답하는 도구가 아니라, 사용자의 목표를 받아 여러 단계를 이어서 수행하려는 시스템입니다. 파일을 찾고, 코드를 쓰고, 실행하고, 오류를 보고, 다시 고치고, 결과를 정리하려 합니다. 카파시가 말하듯, 지금의 에이전트는 아직 완성된 직원이라기보다 계속 감독해야 하는 초기 형태의 작업자에 가깝습니다. 그래서 그는 "에이전트의 해"보다 "에이전트의 10년"이라는 표현이 더 어울린다고 말합니다 (링크).

용어 메모

LLM: large language model의 줄임말입니다. 많은 텍스트를 학습해 다음에 올 말을 예측하고, 그 과정을 바탕으로 대화와 글쓰기, 요약, 코드 작성 등을 수행하는 모델입니다.

에이전트(agent): 사용자의 목표를 받아 여러 단계를 이어서 수행하려는 AI 시스템입니다. 이 책에서는 사람의 감독 아래 일하는 작업자로 이해하면 충분합니다.

의생명과학의 작업대에서 에이전트는 여러 자리에 들어올 수 있습니다. 한 에이전트는 수업 주제와 관련된 자료를 찾아 목록으로 묶을 수 있습니다. 다른 에이전트는 공개 데이터셋의 설명 파일을 읽고 샘플 이름과 조건을 정리할 수 있습니다.

또 다른 에이전트는 작은 표를 불러와 빠진 값이 있는지 확인하고, 처리군과 대조군의 평균을 비교하는 코드 초안을 만들 수 있습니다. 이 일들은 각각 작아 보이지만, 실제 공부와 연구에서는 많은 시간을 잡아먹습니다. 에이전트는 그런 반복 작업의 속도를 높이고, 학생이 조금 더 빨리 질문의 본체에 닿게 해줄 수 있습니다.

용어 메모

처리군: 약물, 조건, 자극처럼 어떤 처리를 받은 대상입니다.

대조군: 처리군과 비교하기 위해 두는 기준 대상입니다.

빠진 값(결측값): 표 안에서 측정되지 않았거나 기록되지 않은 빈칸입니다. 분석 전에 어떻게 다룰지 확인해야 합니다.

하지만 속도가 빨라진다는 말은 위험도 빨라진다는 뜻입니다. 에이전트가 논문 제목을 잘못 읽으면 잘못된 문헌 목록이 아주 그럴듯하게 정리됩니다. 데이터 파일의 열 이름을 착각하면 그래프는 예쁘지만 비교가 틀릴 수 있습니다. 특정 유전자의 기능을 모델이 매끄럽게 설명해도, 그 설명이 다른 유전자와 섞였을 수 있습니다. 생명과 질병을 다루는 공부에서는 이런 오류가 작지 않습니다. 그래서 에이전트는 혼자 달리게 두는 존재가 아니라, 사람이 목표와 경계와 확인 기준을 정해주어야 하는 존재입니다. 카파시가 코드 에이전트와 AutoResearch를 이야기할 때도, 여러 작업을 더 큰 단위로 맡기는 능력은 사람이 지시와 결과를 계속 검토하는 구조 안에서 의미를 갖습니다 (링크, 링크).

생물학에서는 이 감독이 특히 중요합니다. 소프트웨어에서는 코드를 고치고 곧바로 테스트를 돌려 성공과 실패를 비교할 수 있는 경우가 많습니다. 반면 생물학 실험은 느리고 비싸며, 실패의 원인도 자주 흐립니다. 세포 상태가 나빴는지, 시약이 달랐는지, 샘플 준비가 흔들렸는지, 분석 파이프라인이 잘못되었는지 한 번에 알기 어렵습니다. 그러므로 생물학에서 AI는 "실험을 대신하는 기계"라기보다, 질문을 정리하고 자료를 찾고 분석을 반복하며 다음 확인을 준비하는 동료에 더 가깝습니다. 젖은 실험의 판단까지 자동으로 사라지는 것은 아닙니다.

그래서 의도를 표현해야 한다

이제 "코딩이 아니라 의도를 표현한다"는 말이 제자리를 찾습니다. 이 말은 여전히 초반에 들어올 만큼 중요합니다. 다만 생물학의 작업대가 어떻게 바뀌었는지를 먼저 본 뒤에야, 그 말의 무게가 제대로 보입니다. 카파시는 최근 인터뷰에서 "code"라는 동사 자체가 예전만큼 정확하지 않을 수 있다고 말합니다. 사람이 하루 종일 코드를 직접 치는 대신, 자신이 원하는 바를 에이전트에게 설명하고, 에이전트가 만든 결과를 검토하고, 다시 지시하는 일이 커지고 있기 때문입니다. 그는 이 변화를 의도 구현(manifesting)이라는 말로도 표현합니다 (링크).

의도 구현은 소원을 말하면 결과가 생긴다는 뜻이 아닙니다. 오히려 반대입니다. 머릿속의 흐릿한 바람을 에이전트가 실행할 수 있을 만큼 분명한 조건과 기준으로 바깥에 드러내는 일입니다. "이 표를 분석해줘"는 출발점일 수 있지만, 그 자체로는 너무 넓습니다. "원본 파일은 수정하지 말고, `group` 열을 기준으로 처리군과 대조군을 나눈 뒤, 먼저 결측값과 표본 수를 확인하고, 평균과 표준편차를 표로 정리한 다음 막대그래프를 그려줘. 마지막에는 이 비교를 해석할 때 조심해야 할 점을 적어줘." 이 문장은 길어졌지만 단순히 친절해진 것이 아닙니다. 자료, 경계, 절차, 산출물, 검토 기준이 들어갔습니다. 생물학 실험 프로토콜이 "세포를 키운다"로 끝나지 않고 배지, 온도, 시간, 농도, 세척 조건을 적는 것과 비슷합니다.

처음 배우는 학생이 이런 문장을 한 번에 쓸 필요는 없습니다. 좋은 지시는 대화 속에서 만들어집니다. 처음에는 "이 표가 무엇을 담고 있는지 설명해줘"라고 물을 수 있습니다. 다음에는 "처리군과 대조군을 나눌 수 있는 열이 무엇인지 찾아줘"라고 물을 수 있습니다. 다시 "이 비교를 하려면 어떤 정보를 더 확인해야 하지?"라고 물을 수 있습니다. 질문을 좁히는 동안 학생은 에이전트에게 일을 시키는 법만 배우는 것이 아닙니다. 자기 질문이 어디에서 흐릿한지 알아차립니다. 그 흐릿함을 조금씩 말로 바꾸는 과정이 의도 표현입니다.

여기서 코딩은 사라지지 않습니다. 오히려 위치가 바뀝니다. Python의 `for` 문을 완벽하게 외우지 못해도 반복 작업을 부탁할 수는 있습니다. 그러나 반복이 무엇인지 모르면 에이전트가 만든 코드를 읽을 수 없습니다. `groupby` 문법을 손으로 모두 기억하지 못해도 그룹별 평균을 구하라고 말할 수는 있습니다. 하지만 그룹별 평균이 무엇을 비교하는지 모르면 결과를 해석할 수 없습니다. 기초 문법은 이제 혼자 모든 것을 만들기 위한 도구이면서, AI가 만든 것을 읽고 판단하기 위한 언어가 됩니다. 학생은 프로그래머가 되기 위해서만 코딩을 배우는 것이 아니라, AI와 함께 일할 때 자신의 판단을 잃지 않기 위해 코딩을 배웁니다.

빠른 손보다 느린 판단

AI 시대의 공부는 빠릅니다. 모르는 개념을 여러 수준으로 풀어달라고 할 수 있고, 영어 문단이나 강의자료의 어려운 문장을 한국어로 바꿔볼 수 있으며, 작은 데이터 표로 그림을 그려볼 수 있습니다. 이 속도는 학생에게 좋은 기회를 줍니다. 문법의

벽에서 너무 오래 멈추지 않고, 더 일찍 자기 질문으로 들어갈 수 있기 때문입니다. 그러나 빠른 도구는 학생의 생각을 대신 자라게 해주지 않습니다. 질문이 흐릿하면 에이전트는 흐릿한 질문을 빠르게 실행합니다. 비교 기준이 잘못되면 잘못된 비교가 더 빨리 예쁘게 나옵니다. 출처를 확인하지 않으면 틀린 설명도 매끄러운 문장으로 남습니다.

그래서 첫 장에서 우리가 붙잡아야 할 태도는 단순한 낙관도, 단순한 경계도 아닙니다. AI는 의생명과학 학생에게 실제로 큰 도움을 줄 것입니다. 어려운 자료 읽기의 첫 문턱을 낮추고, 데이터 정리의 반복을 줄이며, 코드 초안을 만들어주고, 낯선 개념을 여러 번 다른 말로 설명해줄 수 있습니다. 동시에 AI는 잘못된 확신을 빠르게 만들어낼 수 있습니다. 모델이 만든 문장이 유창할수록, 학생은 자신이 이해했다고 착각하기 쉽습니다. 에이전트가 여러 단계를 수행할수록, 중간에 무엇을 잘못했는지 놓치기 쉽습니다.

생명과학에서 책임은 조금 더 무겁습니다. 우리가 다루는 것은 때로 세포의 상태이고, 환자의 질병이며, 약물 반응이고, 유전자의 기능입니다. 틀린 요약 하나가 당장 사람을 해치지는 않더라도, 틀린 이해가 쌓이면 잘못된 질문과 잘못된 실험으로 이어질 수 있습니다. 그러므로 학생은 AI를 금지된 지름길로 볼 필요도 없고, 모든 것을 맡겨도 되는 자동 연구자로 볼 필요도 없습니다. 조금 더 가까운 표현은 감독받는 작업자입니다. 일을 나누어 맡길 수 있지만, 목적과 기준과 책임은 사람이 붙잡아야 합니다.

다음 장에서 우리는 이 작업자를 조금 더 자세히 보게 됩니다. 에이전트가 답변기와 어떻게 다른지, 여러 단계를 이어가는 일이 왜 힘이 되면서도 위험한지, 생물학 연구에서는 어떤 루프가 빨리 돌고 어떤 루프가 느리게 닫히는지 살펴볼 것입니다. 그전에 첫 장에서 분명히 해두고 싶은 점이 있습니다. AI는 생명과학 바깥에서 들어온 유행어가 아닙니다. 이미 강의자료, 데이터, 코드, 그림, 질문이 모이는 작업대 안으로 들어왔습니다. 의생명과학을 처음 배우는 학생은 그 작업대 앞에서 문법만 배우는 사람이 아니라, 무엇을 묻고 무엇을 맡기며 무엇을 확인할지 배우는 사람이 되어야 합니다. 그때 코딩은 낯선 문법 암기에서 조금씩 벗어나, 생명현상을 읽고 자기 의도를 실행 가능한 형태로 바꾸는 공부가 됩니다.

2장. 에이전트와 함께 일한다는 것

질문에서 작업으로

ChatGPT를 처음 쓸 때 우리는 대개 질문과 답변의 형식으로 생각합니다. 내가 묻고, 모델이 답합니다. 이것만으로도 충분히 놀랍습니다. 낯선 개념을 설명해달라고 하면 곧바로 문단이 나오고, 영어 문단이나 짧은 초록을 붙여넣으면 한국어 요약이 만들어지고, 오류 메시지를 보여주면 가능한 원인을 짚어줍니다. 그러나 에이전트(agent)라는 말이 등장하면 장면이 조금 달라집니다. 에이전트는 단순히 답을 말하는 데서 멈추지 않고, 어떤 목표를 향해 여러 단계를 수행하려는 시스템을 가리킵니다. 파일을 읽고, 코드를 고치고, 실행하고, 오류를 보고, 다시 수정하고, 결과를 정리하는 식입니다.

용어 메모

에이전트: 사용자의 목표를 받아 여러 단계를 이어서 수행하려는 AI 시스템입니다.

코드 실행: 말로만 답하는 것이 아니라 실제 프로그램을 돌려 결과를 확인하는 일입니다.

오류 메시지: 컴퓨터가 "여기서 문제가 생겼다"고 알려주는 짧은 보고서입니다.

학생의 입장에서 이 차이는 처음에는 잘 보이지 않을 수 있습니다. 둘 다 화면 속에서 대화하니까요. 하지만 실제 작업을 맡겨보면 금방 다릅니다. 질문형 AI는 "이 오류는 아마 이런 이유일 수 있습니다"라고 말합니다. 에이전트는 저장소를 열고, 파일을 찾고, 코드를 고치고, 테스트를 돌리고, 실패하면 다시 수정합니다. 물론 항상 잘하는 것은 아닙니다. **대화에서 실행으로 넘어가는 순간, 우리는 AI를 단순한 설명자에서 작업자로 보기 시작합니다.**

조금 더 구체적으로 말해봅시다. 일반 챗봇에게 "이 표를 분석하려면 어떻게 해야 해?"라고 물으면, 챗봇은 결측값을 확인하고, 그룹별 평균을 내고, 그래프를 그리라는 절차를 설명할 수 있습니다. 에이전트에게 같은 일을 맡기면, 한 번의 지시 안에서 파일을 열고, 열 이름을 확인하고, 결측값 개수를 계산하고, 그래프 코드를 만들고, 실행하다가 오류가 나면 다시 고치려 할 수 있습니다. 학생이 매번 직접 "파일 열기 → 코드 쓰기 → 오류 보기 → 수정하기 → 결과 정리하기"를 반복하던 일을, 에이전트는 하나의 작업 흐름으로 이어가려 합니다. 이 차이가 에이전트형 작업 흐름의 출발점입니다. 물론 이어간다는 말이 곧 믿어도 된다는 뜻은 아닙니다. 오히려 여러 단계를 이어가므로, 중간에 무엇을 잘못 읽었는지 사람이 더 잘 볼 수 있어야 합니다.

카파시는 에이전트를 아직 완성된 직원이라기보다, 함께 일할 수는 있지만 계속 감독해야 하는 초기 형태의 동료에 가깝게 설명합니다. 그는 "에이전트의 해"라는 표현보다 "에이전트의 10년"이라는 표현이 더 맞다고 말합니다 (링크). 이미 Claude Code나 Codex 같은 도구는 인상적입니다. 둘 다 프로그래머가 저장소 안에서 코드 작성과 수정, 실행을 맡길 수

있는 대표적인 코드 작성 보조 에이전트입니다. 하지만 사람을 그대로 대체하기에는 아직 부족한 것이 많습니다. 기억은 제한적이고, 멀티모달 이해는 완전하지 않으며, 긴 작업을 안정적으로 이어가는 능력도 아직 거칠습니다.

이 신중함은 중요합니다. AI를 둘러싼 말들은 자주 너무 빨리 달려갑니다. 곧 모든 연구자가 필요 없어질 것처럼 말하기도 하고, 반대로 아무것도 믿을 수 없는 장난감처럼 말하기도 합니다. 실제 경험은 그 사이에 있습니다. 에이전트는 놀라울 만큼 많은 일을 해낼 수 있지만, 동시에 아주 기본적인 지시를 오해할 수 있습니다. 긴 코드를 고치다가 핵심 가정을 바꾸어버릴 수도 있고, 멋진 분석을 만든 뒤 실제 파일 이름을 잘못 읽었을 수도 있습니다.

빠른 루프와 느린 실험

그렇다면 에이전트와 함께 일한다는 것은 어떤 모습일까요. 카파시는 한 줄의 코드나 함수 하나를 넘어서 더 큰 단위의 행동을 생각합니다. 이 기능을 구현해보라, 이 오류를 조사하라, 이 결과를 비교하라, 이 실험을 여러 조건으로 돌려보라 같은 식의 큰 작업 단위입니다 (링크). 사람은 모든 줄을 직접 쓰는 대신, 여러 에이전트에게 서로 다른 일을 맡기고, 각 결과를 검토하고, 충돌을 조정하고, 최종 판단을 내립니다.

의생명 연구에서도 비슷한 상상을 해볼 수 있습니다. 한 에이전트는 논문 목록을 정리하고, 다른 에이전트는 공개 데이터셋의 메타데이터를 읽고, 또 다른 에이전트는 분석 코드를 초안으로 만들 수 있습니다. 사람 연구자는 각 에이전트의 결과를 모아 질문을 다시 좁히고, 생물학적으로 말이 되는지 확인하고, 통계적으로 타당한지 검토합니다. 여기서 중요한 것은 에이전트가 연구자를 대신한다는 말이 아닙니다. **연구자의 일이 손작업에서 설계와 감독으로 조금씩 이동한다는 말입니다.**

카파시가 AutoResearch를 이야기할 때도 같은 선이 보입니다. 목표와 지표와 경계를 정해두고, 에이전트가 여러 실험을 반복하며 개선을 찾도록 하는 흐름입니다 (링크). 연구자가 매번 다음 실험 버튼을 누르는 대신, 실험의 틀을 정하고 에이전트가 그 안에서 많은 시도를 하게 합니다. 이것은 연구를 완전히 자동화한다는 거창한 선언이라기보다, 반복 가능한 부분과 사람이 판단해야 하는 부분을 새로 나누는 일에 가깝습니다.

용어 메모

AutoResearch: 연구의 일부 과정을 에이전트가 반복적으로 시도하고 비교하게 만드는 흐름입니다.

지표: 결과가 좋아졌는지 나빠졌는지 판단하기 위해 미리 정해두는 숫자나 기준입니다.

경계: 에이전트가 해도 되는 일과 하면 안 되는 일을 나누어 둔 선입니다.

소프트웨어에서는 이 변화가 비교적 빨리 일어납니다. 이유는 단순합니다. 실행과 판정의 루프가 짧기 때문입니다. 코드를 고치면 곧바로 테스트를 돌릴 수 있습니다. 웹페이지가 깨졌는지 눈으로 볼 수 있습니다. 함수가 원하는 값을 반환하는지 확인할 수 있습니다. 틀리면 다시 고치면 됩니다. 물론 소프트웨어도 어렵지만, 적어도 많은 경우에는 성공과 실패를 빠르게 확인할 수 있습니다.

생물학 연구는 다릅니다. 실험은 느리고 비쌉니다. 세포를 키우는 데 시간이 걸리고, 샘플 준비에는 숙련된 손놀림이 필요하며, 실험이 실패해도 실패의 원인이 항상 명확하지 않습니다. 측정값은 잡음을 품고 있고, 판정 기준은 흐릴 때가 많습니다. 어떤 유전자를 건드렸을 때 세포 상태가 달라졌다고 해도, 그것이 진짜 인과적 변화인지, 배치 효과인지, 세포 조성의 변화인지, 분석 파이프라인의 산물인지 구분해야 합니다. 그래서 소프트웨어에서 가능한 에이전트 루프가 생물학에서는 쉽게 닫히지 않습니다.

용어 메모

배치 효과: 실제 생물학 차이가 아니라 실험 날짜, 장비, 시약 차이 때문에 생기는 차이입니다.

파이프라인: 데이터를 넣으면 여러 분석 단계를 지나 결과가 나오도록 이어 둔 절차입니다.

인과적 변화: 단순히 함께 보이는 것이 아니라, 한 변화가 다른 변화를 일으켰다는 뜻입니다.

이 차이를 이해하면 AI가 바이오 연구에 들어오는 방식도 더 차분하게 볼 수 있습니다. 에이전트가 논문을 찾고, 코드를 쓰고, 표를 만들고, 후보 유전자를 정리하는 일은 빠르게 좋아질 것입니다. 그러나 세포를 실제로 배양하고, 개입 실험(perturbation)을 하고, 표현형을 측정하고, 그 결과가 생물학적으로 무엇을 뜻하는지 판단하는 일은 훨씬 느리게 바뀝니다. 그러므로 생물학에서 에이전트의 능력을 이야기할 때는 항상 루프를 보아야 합니다. 어디까지는 디지털 루프 안에서 빠르게 반복할 수 있고, 어디서부터는 실험실의 시간과 몸을 통과해야 하는지 나누어 보아야 합니다.

용어 메모

개입 실험(perturbation): 세포나 시스템에 일부러 변화를 주어 반응을 보는 일입니다.

표현형: 유전자나 환경의 영향이 실제 세포, 조직, 몸의 모습이나 기능으로 드러난 결과입니다.

assay: 어떤 반응이나 상태를 측정하기 위해 정해 둔 실험 방법입니다.

디지털 루프: 컴퓨터 안에서 빠르게 반복할 수 있는 분석과 검토의 순환입니다.

학생에게는 이 구분이 아주 중요합니다. AI가 논문 열 편을 요약해주었다고 해서 문헌 조사가 끝난 것은 아닙니다. AI가 표를 정리하고 그래프를 만들어주었다고 해서 분석이 끝난 것도 아닙니다. AI가 그럴듯한 설명을 제안했다고 해서 생물학적 결론이 생긴 것도 아닙니다. 각각의 단계에는 확인해야 할 것이 있습니다. 논문 요약에는 원문 확인이 필요하고, 표와 그래프에는 입력 파일과 비교 기준 확인이 필요하며, 생물학적 설명에는 독립적인 근거와 반대 가능성에 대한 검토가 필요합니다.

경계를 정하는 연구자

여기서 에이전트와 함께 일하는 능력은 "많이 시키는 능력"과 다릅니다. 오히려 무엇을 시키지 말아야 하는지 아는 능력에 가깝습니다. 원본 데이터를 수정하지 말라고 잠그는 일, 환자 개인정보가 포함된 파일을 외부 도구에 넣지 않도록 막는 일, 발견용 분석과 확인용 분석을 분리하는 일, 여러 번 시도한 분석 경로를 출처와 절차 기록으로 남기는 일, 성공 기준과 실패 기준을 미리 정하는 일이 중요합니다. **빨리 돌리는 것보다 먼저 해야 할 일은 경계를 정하는 것입니다.**

왜 이렇게 조심해야 할까요. 에이전트는 편향도 자동화할 수 있기 때문입니다. 예전에도 사람은 자신이 원하는 결과가 나올 때까지 비교 대상을 바꾸거나, 특정 값을 빼거나, 그래프 모양을 여러 번 바꿔볼 수 있었습니다. 문제는 에이전트가 이런 일을 훨씬 더 빠르게, 더 많이 해볼 수 있다는 데 있습니다. 잘못된 목표를 주면 에이전트는 그 목표를 향해 열심히 움직입니다. "진짜 차이가 있는지 확인해줘"가 아니라 "내가 기대한 결과가 잘 보이게 만들어줘"라는 식의 목표가 들어가면, AI는 확인편향을 생산성으로 포장할 수 있습니다.

그래서 에이전트 시대의 연구자에게 필요한 것은 분석 능력의 총량만이 아닙니다. 증거를 다루는 규율입니다. 질문을 미리 적어두는 습관, 어떤 비교를 할지 정하는 습관, 어떤 결과가 나오면 내 가설을 버릴지 생각하는 습관, 여러 분석 경로를 숨기지 않고 남기는 습관, 마지막에는 독립 데이터나 다른 실험으로 확인하는 습관이 필요합니다. AI가 많이 도와줄수록 이런 규율은 더 중요해집니다.

기록으로 남는 협업

의생명과학에서 앞으로 먼저 달라질 연구는 아마 루프 안으로 잘 들어오는 연구일 것입니다. 표준화 가능한 개입 실험이 있고, 측정 가능한 표현형이 있고, 자동화 가능한 분석법(assay)이 있고, 결과를 비교할 수 있는 지표가 있는 분야는 에이전트와 모델의 도움을 빨리 받을 수 있습니다. 반대로 판정이 모호하고, 샘플 준비의 암묵지가 크고, 실패가 데이터로 남지 않는 분야는 훨씬 천천히 움직일 것입니다. 이 차이는 연구 주제의 중요성과 별개입니다. 중요한 문제라도 루프로 번역하기 어렵다면 AI의 도움을 받는 속도가 느릴 수 있습니다.

이때 "루프"라는 말은 단순한 자동화를 뜻하지 않습니다. 가설을 세우고, 실험이나 분석을 수행하고, 결과를 측정하고, 그 측정이 다음 가설로 돌아가는 구조를 뜻합니다. 좋은 에이전트형 작업 흐름(agentic workflow)은 이 순환을 분명하게 만듭니다. 어디서 시작했고, 무엇을 바꾸었고, 어떤 결과를 얻었고, 왜 다음 단계로 넘어갔는지 남깁니다. 연구 노트와 데이터 관리가 귀찮은 행정 업무가 아니라, 에이전트와 함께 일하기 위한 기반이 되는 이유도 여기에 있습니다.

용어 메모

에이전트형 작업 흐름(agentic workflow): 에이전트가 읽기, 실행, 수정, 보고 같은 단계를 이어가도록 짠 작업 흐름입니다.

연구 노트: 무엇을 했고 무엇을 보았는지 나중에 되짚을 수 있게 남기는 기록입니다.

이 점은 1학년 학생에게도 멀리 있는 이야기가 아닙니다. 여러분이 처음 쓰는 작은 분석 노트북도 하나의 루프가 될 수 있습니다. 데이터를 불러오고, 이상한 값을 발견하고, 왜 그런지 찾아보고, 결측값 처리 기준을 정하고, 다시 그림을 그리는 과정이 그렇습니다. 처음에는 단순한 과제처럼 보이지만, 사실은 연구의 축소판입니다. 에이전트가 옆에 있으면 이 루프는 더 빨리 돌 수 있습니다. 그러나 빨리 도는 루프가 좋은 루프라는 뜻은 아닙니다. 무엇을 바꾸었는지 기록하지 않으면, 좋은 결과가 나와도 왜 나왔는지 알 수 없습니다.

이런 기록 습관은 처음에는 지나치게 엄격해 보일 수 있습니다. 과제 하나 하는데 왜 원본 파일을 따로 두고, 어떤 명령을 했는지 적고, 결과가 어디에 저장되었는지 남겨야 할까요. 그러나 작은 과제에서 배운 습관은 나중에 큰 연구를 만났을 때 학생을 지켜줍니다. 파일 이름이 비슷한 두 표를 헷갈리지 않는 일, 어제 그린 그림과 오늘 그린 그림이 왜 다른지 설명할

수 있는 일, 에이전트가 어떤 중간 결과를 보고 다음 단계로 넘어갔는지 되짚을 수 있는 일은 모두 기록에서 나옵니다. 연구에서 좋은 기억력은 머릿속에만 있지 않습니다. 좋은 폴더 구조, 읽을 수 있는 노트, 되돌릴 수 있는 코드, 그리고 왜 그 선택을 했는지 적어둔 짧은 문장에서 생깁니다. AI와 함께 일할수록 이런 기록은 더 중요해집니다. 에이전트는 많은 일을 빠르게 만들어내지만, 사람이 그 흔적을 붙잡아두지 않으면 결과만 남고 과정은 사라집니다.

그래서 에이전트와 함께 일할 때는 부탁의 문장보다 기록의 구조가 더 중요해질 때가 많습니다. "다시 해봐"라는 말만 반복하면, 에이전트는 여러 시도를 하겠지만 나중에 어떤 시도가 왜 버려졌는지 알 수 없습니다. 반대로 "이번에는 빈칸 처리 기준만 바꾸고, 나머지는 그대로 둔 뒤 결과 차이를 표로 남겨줘"라고 말하면 작업의 흔적이 남습니다. "이전 결과와 다른 점을 세 문장으로 적어줘"라고 요구하면, 다음 판단을 위한 메모가 생깁니다. 이런 작은 습관이 쌓이면 에이전트는 단순한 작업자가 아니라 연구 노트의 일부가 됩니다.

생물학의 루프가 어려운 이유는 실패가 자주 말없이 사라지기 때문이기도 합니다. 배양이 잘 안 된 세포, 품질이 낮은 라이브러리, 기대한 marker가 보이지 않은 figure, 논문에는 들어가지 못한 조건들이 연구실 어딘가에서 흩어집니다. 그러나 AI가 학습하고 에이전트가 다음 실험을 제안하려면 실패도 데이터가 되어야 합니다. 무엇을 시도했고, 왜 안 되었고, 어느 지점까지는 괜찮았는지 남아야 다음 루프가 배울 수 있습니다. 성공한 그림만 남기는 연구 문화에서는 에이전트가 배울 수 있는 세계가 좁아집니다.

이 말은 연구가 기계처럼 차갑게 바뀐다는 뜻이 아닙니다. 오히려 반대입니다. 생물학 연구에는 여전히 사람의 판단이 깊게 들어갑니다. 세포 상태가 이상해 보이는지 알아차리는 눈, 논문 결과가 너무 깔끔할 때 드는 의심, 통계적으로는 작지만 생물학적으로 중요한 신호를 알아보는 눈은 쉽게 자동화되지 않습니다. 다만 그 판단이 혼자 머릿속에만 남아 있으면 에이전트와 나눌 수 없습니다. 앞으로의 연구자는 자신의 암묵지를 조금씩 말과 기록으로 바깥에 꺼내는 사람이어야 합니다.

카파시가 말하는 에이전트 시대는 결국 인간의 역할을 더 높은 곳으로 옮깁니다. 그러나 높은 곳으로 옮긴다는 말은 편한 자리로 간다는 뜻이 아닙니다. 더 많은 판단을 해야 한다는 뜻입니다. 에이전트가 작업의 속도를 높이면, 사람은 작업의 방향과 의미를 더 깊이 책임져야 합니다. 좋은 학생은 에이전트가 만들어준 결과를 그대로 제출하는 사람이 아니라, 에이전트가 어디까지 잘했고 어디서 위험해졌는지 설명할 수 있는 사람입니다.

그래서 에이전트를 공부한다는 것은 최신 유행을 따라가는 일이 아닙니다. 앞으로의 공부와 연구 환경에서 자신이 어떤 역할을 맡게 될지 미리 생각하는 일입니다. 사람은 더 이상 모든 일을 혼자 손으로 하는 존재가 아닐 수 있습니다. 그러나 무엇이 중요한지 알아보고, 무엇이 틀렸는지 감지하고, 왜 그 결과를 믿을 수 있는지 설명하는 책임은 여전히 사람에게 남습니다. 의생명과학 학생에게 이 책임은 작지 않습니다. 우리가 다루는 것은 코드의 성공 여부만이 아니라, 생명현상과 질병에 대한 주장일 때가 많기 때문입니다.

처음 에이전트를 쓰는 학생은 대개 두 가지 실수를 번갈아 합니다. 하나는 너무 적게 맡기는 것입니다. 모델에게 단어 뜻만 묻고, 실제로는 혼자 모든 일을 하다가 금세 지칩니다. 다른 하나는 너무 많이 맡기는 것입니다. 데이터 파일을 던져주고 "분석해줘"라고 말한 뒤, 결과가 그럴듯하면 그대로 믿습니다. 좋은 사용법은 그 사이에 있습니다. 에이전트에게 반복적이고 명시적인 작업은 맡기되, 질문의 방향과 검토 기준은 사람이 붙잡아야 합니다. 예를 들어 문헌 목록을 정리하게 할 수는 있지만, 어떤 논문을 정말 읽어야 하는지는 연구 질문에 비추어 판단해야 합니다. 코드를 작성하게 할 수는 있지만, 그 코드가 어떤 열을 읽고 어떤 행을 버렸는지는 확인해야 합니다. 그림을 만들게 할 수는 있지만, 그림이 보여주는 차이가 생물학적으로 의미 있는지는 따로 생각해야 합니다. 이 균형을 배우는 일이 앞으로의 연구 훈련에서 중요한 자리를 차지하게 될 것입니다.

에이전트와 함께 일하는 시대에는 실패를 다루는 방식도 달라집니다. 사람이 혼자 작업할 때는 실패가 자기 머릿속에 남는 경우가 많습니다. 어떤 분석을 해봤는데 잘 안 되었고, 어떤 조건을 바꿨더니 이상한 그림이 나왔고, 어떤 논문을 읽어보니 처음 생각이 틀렸다는 사실이 조용히 지나갑니다. 그러나 에이전트와 함께 일할 때는 이 실패들을 기록으로 남길 수 있습니다. "이 접근은 왜 버렸는가", "이 결과가 왜 믿기 어려운가", "다음 시도에서는 무엇을 바꿀 것인가"를 문장으로 남기면, 에이전트는 다음 작업에서 그 기록을 읽을 수 있습니다. 사람도 자신의 시행착오를 더 잘 돌아볼 수 있습니다. **연구의 속도는 성공한 작업만으로 빨라지지 않습니다. 실패가 다음 판단에 연결될 때 빨라집니다.** AI는 그 연결을 돕는 도구가 될 수 있지만, 실패를 정직하게 기록하려는 태도는 사람에게서 시작됩니다.

1학년 학생에게 이 말은 아직 멀게 느껴질 수 있습니다. 하지만 작은 과제에서도 실패를 기록하는 습관은 바로 시작할 수 있습니다. 그래프가 이상하게 나왔을 때 그냥 지우고 새로 만들지 말고, 왜 이상했는지 한 줄로 남겨보십시오. 파일 이름을 잘못 골랐는지, 열 이름을 착각했는지, 비교할 두 그룹을 잘못 나누었는지 적어두면 다음번 실수가 줄어듭니다. AI가 도와주는 시대의 실력은 실패가 없는 사람이 되는 것이 아니라, 실패를 다음 질문으로 바꾸는 사람이 되는 데서 자랍니다.

3장. 의생명과학 학생에게 남는 질문

답이 쉬워질 때 남는 공부

AI가 많은 일을 도와줄 수 있다면, 우리는 무엇을 배워야 할까요. 이 질문은 대학에 막 들어온 학생에게 꽤 현실적으로 다가옵니다. 이미 ChatGPT는 생물학 개념을 설명하고, 영어 문단이나 강의자료의 어려운 문장을 풀어주고, Python 코드를 써주고, 수행평가 글이나 발표문 초안을 만들어줍니다. 그렇다면 세포생물학을 왜 배워야 할까요. 통계를 왜 배워야 할까요. 코딩은 정말 필요한 걸까요.

이 질문을 너무 쉽게 닫아버리면 안 됩니다. "그래도 기초는 중요하다"라는 말은 맞지만, 그것만으로는 충분하지 않습니다. 학생은 이미 AI가 답을 만들어주는 장면을 보았습니다. 예전과 똑같이 공부하라고 말하기 전에, 무엇이 정말 달라졌는지 정직하게 보아야 합니다. AI는 분명히 공부의 표면을 바꿉니다. 낯선 개념을 처음 만나는 순간의 두려움을 줄이고, 막힌 코드 앞에서 보내는 시간을 줄이며, 논문을 읽기 전 배경지식을 빠르게 잡아줄 수 있습니다.

하지만 바로 그 이유 때문에 기초가 더 중요해지는 순간이 옵니다. **LLM의 답변은 매끄럽습니다. 틀린 설명도 매끄럽고, 존재하지 않는 논문 인용도 그럴듯합니다.** 잘못된 유전자 기능 설명도 논문 초록처럼 보일 수 있습니다. 생명과 질병을 다루는 공부에서는 이 차이가 작지 않습니다. 틀린 철자 하나가 다른 유전자를 뜻할 수 있고, 부정확한 약물 설명이 위험한 오해로 이어질 수 있으며, 통계 결과를 잘못 읽으면 실험의 결론이 바뀔 수 있습니다.

생명 데이터와 모델의 만남

앞에서 보았듯, 의생명과학에서 AI가 중요한 이유는 ChatGPT가 편리해서만은 아닙니다. 생명과학 자체가 점점 더 큰 데이터의 언어로 말해지고 있기 때문입니다. 여기서는 그 흐름을 한 걸음 더 깊게 들여다보겠습니다. 현미경으로 세포 모양을 보던 일은 이미지 파일이 되고, 실험실에서 얻은 측정값은 표가 되며, 논문 속 결론은 그래프와 통계와 데이터베이스 링크를 지나 우리에게 옵니다. 학생은 처음에 이 흐름을 모두 이해하지 못해도 괜찮습니다. 다만 앞으로 생명현상을 공부할 때 "세포 안에서 무슨 일이 일어났는가"라는 질문이 자주 "어떤 데이터가 그 일을 보여주는가"라는 질문과 함께 온다는 사실은 일찍 알아두는 편이 좋습니다. AI는 이 데이터들을 표현하고, 비교하고, 때로는 다음 실험을 예측하는 도구로 등장합니다.

단일세포 RNA-seq도 그 흐름 안에 있습니다. 이름은 길지만 출발점은 단순합니다. 예전의 유전자 발현 분석은 여러 세포가 섞인 조직을 한꺼번에 보고 평균적인 신호를 얻는 경우가 많았습니다. 반면 단일세포 RNA-seq은 세포 하나하나를 따로 떼어, 각 세포 안에서 어떤 유전자가 얼마나 읽혀 있었는지 살펴보려는 방법입니다. 여기서 "유전자가 발현된다"는 말은 유전자의 정보가 RNA로 읽혀 세포 안에서 실제 작업에 쓰이는 정도를 뜻합니다. 모든 유전자가 모든 세포에서 똑같이 켜져 있지는 않습니다. 간세포와 면역세포는 같은 DNA를 가지고 있어도 다른 유전자들을 더 많이 읽고, 같은 면역세포라도 감염이나 약물 처리 뒤에는 읽히는 유전자의 모습이 달라질 수 있습니다. 그래서 단일세포 데이터는 결국 큰 표가 됩니다. 행에는 세포가 놓이고, 열에는 유전자가 놓이며, 각 칸에는 "이 세포에서 이 유전자가 얼마나 읽혔는가"에 가까운 숫자가 들어갑니다.

단계	학생이 떠올리면 좋은 그림
조직이나 배양 세포를 준비한다	여러 종류의 세포가 섞인 교실을 떠올립니다.
세포를 하나씩 나누어 본다	반 전체 평균만 보지 않고 학생 한 명씩 보는 일과 비슷합니다.
각 세포의 RNA를 읽는다	각 학생이 지금 어떤 노트를 펼쳐 읽고 있는지 보는 일에 가깝습니다.
세포 x 유전자 표를 만든다	행은 세포, 열은 유전자, 칸은 발현량인 큰 표가 됩니다.

용어 메모

단일세포 RNA-seq: 세포를 한 덩어리로 평균 내지 않고, 세포 하나하나에서 RNA 정보를 읽는 방법입니다.

전사체: 한 세포나 조직에서 읽혀 나온 RNA들의 전체 모습입니다.

유전자 발현: 유전자의 정보가 RNA로 읽혀 세포 안에서 실제로 쓰이는 정도를 말합니다.

이런 표는 사람의 눈으로 직접 읽기 어렵습니다. 세포가 수만 개, 유전자가 수만 개라면 표의 칸은 너무 많아집니다. 연구자는 이 표를 더 보기 쉬운 그림으로 줄이고, 비슷한 세포끼리 묶고, 각 묶음이 어떤 세포인지 추정하고, 조건 사이의

차이를 봅니다. 여기까지도 이미 많은 계산이 필요합니다. 최근에는 한 걸음 더 나아가, 이런 대규모 생물학 데이터를 바탕으로 기반 모델(foundation model)을 만들려는 흐름도 커지고 있습니다. 이 이름은 지금 낯설어도 됩니다. 여기서는 "아주 많은 자료를 먼저 학습해 여러 과제의 출발점으로 쓰는 큰 모델" 정도로만 이해하면 충분합니다. 언어 모델이 많은 문장을 읽고 단어와 문맥의 반복을 배우듯, 생물학의 큰 모델도 많은 세포와 유전자 자료에서 반복되는 관계를 배우려 합니다.

용어 메모

기반 모델(foundation model): 아주 많은 자료를 먼저 학습해 여러 과제의 출발점으로 쓰는 큰 모델입니다.

이 분야의 규모는 빠르게 커지고 있습니다. 실제 논문을 읽다 보면 여러 모델 이름을 만나게 되겠지만, 1학년 학생이 지금 그 이름을 외울 필요는 없습니다. 모델 이름보다 중요한 것은 질문의 모양입니다. 연구자들은 "많은 세포 자료를 먼저 읽은 모델이 새로운 세포 상태를 더 잘 이해할 수 있을까", "한 실험에서 배운 관계가 다른 조직이나 다른 질병에서도 통할까"를 묻고 있습니다. 어떤 모델은 세포를 하나의 문장처럼 보고 유전자를 토큰처럼 다루려 하고, 어떤 모델은 유전자 발현량과 유전자 정체성을 함께 넣으며, 어떤 모델은 유전자들 사이의 연결 구조를 활용합니다. 겉으로 보면 LLM과 비슷해 보입니다. 많은 데이터를 읽고, 그 안의 반복되는 관계를 압축해, 새로운 과제에 옮겨 쓰려는 시도이기 때문입니다.

그러나 생물학 데이터는 자연어와 다릅니다. 문장에는 단어 순서가 있습니다. "세포가 신호를 받았다"와 "신호가 세포를 받았다"는 다릅니다. 하지만 단일세포 발현 행렬에서 유전자들은 자연스러운 문장 순서로 놓여 있지 않습니다. 유전자 A가 먼저 오고 유전자 B가 뒤에 온다고 해서 생물학적 시간이 흐르는 것은 아닙니다. 그래서 단일세포 기반 모델에서는 토큰화와 입력 구조가 어렵습니다. 유전자를 어떤 순서로 넣을 것인지, 발현량을 어떻게 표현할 것인지, 세포 유형이나 배치 정보를 어떻게 다룰 것인지가 모델의 성격을 바꿉니다.

이 대목에서 1학년 학생이 모든 기술 세부사항을 이해할 필요는 없습니다. 중요한 것은 생물학 데이터도 모델이 배울 수 있는 패턴을 담고 있지만, 자연어 문장과 같은 방식으로 놓여 있지는 않다는 점입니다. 문장은 앞뒤 순서가 의미를 많이 정하지만, 세포 데이터에서는 실험 조건, 세포 상태, 측정 방법, 샘플의 출처가 함께 의미를 만듭니다. 그래서 모델을 크게 만들었다는 말만으로는 충분하지 않습니다. 어떤 자료를 배웠는지, 그 자료가 어떤 실험에서 왔는지, 모델이 새 조건에서도 잘 작동하는지 물어야 합니다. 의생명과학 학생은 AI 모델을 볼 때도 생물학자의 질문을 잃지 않아야 합니다. 이 모델은 무엇을 실제로 보았는가. 어떤 조건에서는 잘하고, 어떤 조건에서는 흔들리는가. 이 질문이 있어야 큰 숫자에 압도되지 않고 모델을 과학적으로 읽을 수 있습니다.

큰 모델을 의심하는 법

더 중요한 질문은 따로 있습니다. 모델이 크게 학습했다고 해서 정말 생물학을 이해한 것일까요. 최근 단일세포 기반 모델 분야에서는 스케일이 커지는 흐름과 동시에, 그 스케일이 실제로 가치를 만드는지를 묻는 회의적 평가도 함께 나오고 있습니다. 어떤 모델은 간단한 비교 기준보다 못한 결과를 보이기도 하고, 특정 과제에 맞춘 추가 훈련이 있어야 비로소 성능이 나아지는 경우도 있습니다. 여기서 비교 기준이나 추가 훈련의 세부 방법을 모두 알 필요는 없습니다. 1학년 학생에게 더 중요한 것은 큰 모델이라는 말 앞에서 한 번 멈추는 태도입니다. "얼마나 큰가"보다 "무엇을 잘 설명하고, 어디서 틀리는가"를 물어야 합니다.

용어 메모

비교 기준(baseline): 새 모델이 정말 나은지 비교하기 위해 두는 기본 방법입니다.

미세조정(fine-tuning): 이미 학습된 모델을 특정 과제나 데이터에 맞게 조금 더 훈련하는 일입니다.

AI를 배울 때 우리는 자주 크기와 성능에 압도됩니다. 모델 안의 숫자 몇 억 개, 세포 몇 억 개, 글자 조각 몇 조 개 같은 표현은 놀랍습니다. 그러나 과학에서 중요한 질문은 "큰가"에 머물지 않습니다. 크기가 커졌는데도 새로운 조건에서 일반화하지 못한다면, 우리는 그 모델을 조심스럽게 써야 합니다. 반대로 작아 보이는 모델이라도 특정 실험 조건에서 검증이 잘 되어 있다면 더 유용할 수 있습니다. 생명과학에서 좋은 모델은 숫자로 위압감을 주는 모델이 아니라, 낯선 조건 앞에서도 자신이 어디까지 맞고 어디서 흔들리는지 드러내는 모델입니다.

개입을 묻는 과학

특히 의생명과학에서 중요한 시험대는 개입 실험(perturbation) 예측입니다. 이 단어는 책에서 여러 번 나오므로 처음에 조금 천천히 잡아두겠습니다. 개입 실험은 어떤 시스템에 일부러 변화를 주고, 그 뒤에 무엇이 달라지는지 보는 일입니다. 라디오에서 베이스 음만 낮추면 노래가 어떻게 달라지는지 듣는 장면을 떠올려도 좋습니다. 노래 전체를 듣는 것만으로는 베이스가 어떤 역할을 하는지 알기 어렵지만, 베이스만 줄여보면 그 빈자리가 드러납니다. 생물학에서는 유전자의 기능을

줄이거나, 약물을 처리하거나, 특정 신호를 막아 세포가 어떻게 반응하는지 봅니다. 유전자를 녹아웃(knock-out)한다는 말은 유전자가 물리적인 스위치처럼 딸깍 꺼진다는 뜻은 아닙니다. CRISPR, RNA 간섭, 약물 처리처럼 여러 방법으로 특정 유전자의 기능을 줄이거나 없애고, 그 결과를 관찰한다는 뜻에 가깝습니다. 방법의 자세한 차이는 뒤에 배워도 됩니다. 지금은 "일부러 한 가지를 바꾸어 반응을 본다"는 정도로 이해하면 충분합니다.

용어 메모

개입 실험(perturbation): 세포나 생물학적 시스템에 일부러 변화를 주어 반응을 보는 일입니다.

녹아웃(knock-out): 특정 유전자의 기능을 없애거나 크게 줄여 어떤 변화가 생기는지 보는 실험 방법입니다.

인과: 두 일이 함께 보이는 수준을 넘어, 한 일이 다른 일을 일으켰다는 관계입니다.

단순히 "이 세포는 어떤 세포 유형인가"를 맞히는 것과 "이 유전자의 기능을 줄이면 세포 상태가 어떻게 달라지는가"를 예측하는 것은 전혀 다릅니다. 앞의 질문은 분류에 가깝고, 뒤의 질문은 인과에 가까워집니다. 생물학이 의학과 치료로 이어지려면 결국 이런 질문을 피할 수 없습니다. 어떤 유전자가 질병과 함께 보인다는 사실만으로는 충분하지 않습니다. 그 유전자를 바꾸면 세포가 달라지는지, 어떤 경로가 움직이는지, 그 변화가 질병의 원인인지 결과인지 물어야 합니다. 상관관계는 출발점이지만, 치료는 개입의 언어를 요구합니다. 세상이 어떻게 생겼는가를 보는 것과, 세상에 손을 대면 무엇이 달라지는가를 묻는 것은 다릅니다.

이 차이를 조금 더 쉽게 말해보겠습니다. 어떤 질병 조직에서 유전자 A와 유전자 B가 함께 높게 발현된다고 합시다. 이것은 관찰입니다. 둘이 같은 세포 유형에서 높아서 그럴 수도 있고, A가 B를 조절해서 그럴 수도 있고, B가 A를 조절해서 그럴 수도 있고, 둘 다 다른 원인 C의 영향을 받았을 수도 있습니다. 공발현 분석은 이런 관계를 보여주는 데 유용하지만, 그 자체로 인과 방향을 말해주지는 않습니다. 반면 개입 실험은 "A를 일부러 바꾸면 B가 어떻게 되는가"를 묻습니다. 이 질문은 훨씬 어렵지만, 생물학적으로 더 결정적인 질문입니다. 그래서 개입 실험 예측은 생물학 모델의 좋은 시험대가 됩니다. 이미 보이는 세포를 분류하는 일은 사진 속 물체 이름을 맞히는 일에 가깝지만, 유전자의 기능을 바꾸었을 때 세포가 어떻게 달라질지 예측하는 일은 아직 찍지 않은 사진의 결과를 말하는 일에 가깝습니다. 모델이 진짜로 유용해지려면, 과거 데이터의 모양을 흉내 내는 데서 그치지 않고 "이 조건을 바꾸면 무엇이 달라질까"라는 질문에 조금씩 답할 수 있어야 합니다.

AI가 바이오 연구에 줄 수 있는 큰 가능성은 바로 여기 있습니다. 모델이 충분히 좋은 표현을 배우고, 충분히 좋은 개입 실험 데이터를 학습하고, 실험적 검증과 연결된다면, 우리는 실제로 모든 실험을 하기 전에 어떤 개입이 유망한지 좁혀볼 수 있습니다. 이것은 실험을 대체한다기보다, 실험의 방향을 정하는 데 도움을 주는 일입니다. 후보를 넓게 훑고, 가능성이 낮은 길을 줄이고, 더 결정적인 검증으로 나아가게 하는 일입니다.

하지만 이 가능성은 데이터 생산 방식과 붙어 있습니다. 모델만 커져서는 안 됩니다. 어떤 질문에 답하려면 그 질문에 맞는 데이터가 필요합니다. 약물 반응을 예측하려면 잘 설계된 약물 개입 실험 데이터가 필요하고, 발달 과정을 예측하려면 시간축과 공간축을 가진 데이터가 필요하며, 질병을 이해하려면 유전 정보와 임상 정보가 통제된 데이터가 필요합니다. 실패한 실험도 구조화된 데이터로 남아야 합니다. 그래야 모델이 성공의 모양만이 아니라 실패의 경계도 배울 수 있습니다.

이 점에서 앞으로의 바이오 연구는 "분석을 잘하는 연구실"을 넘어 "데이터를 잘 생산하는 연구실"의 문제로 이동합니다. 좋은 데이터 생산은 단순히 많이 측정하는 일이 아닙니다. 어떤 조건을 비교할지, 어떤 변수를 통제할지, 어떤 결과를 성공으로 볼지, 어떤 실패를 기록할지 설계하는 일입니다. AI가 더 강해질수록, 데이터는 그냥 모델에 넣는 재료가 아니라 연구 철학의 표현이 됩니다. 무엇을 측정할 것인가가 곧 무엇을 믿을 수 있는가를 결정합니다.

카파시는 교육에 대해서도 비슷한 긴장을 이야기합니다. AI tutor가 있다면 사람은 훨씬 멀리 갈 수 있지만, 좋은 tutor는 단지 답을 주는 존재가 아닙니다. 학생이 어디까지 알고 어디서 막히는지 알아보고, 너무 쉽지도 너무 어렵지도 않은 문제를 건네야 합니다 (링크). 카파시는 교육을 지식으로 올라가는 경사로를 만드는 일이라고도 말합니다 (링크). 지금의 LLM은 이미 훌륭한 학습 도구가 될 수 있지만, 완벽한 tutor라고 부르기에는 아직 부족합니다.

이 말은 오히려 우리에게 좋은 출발점을 줍니다. AI를 완벽한 선생님으로 믿지 말고, 함께 공부하는 보조자로 두는 것입니다. 모르는 개념을 여러 수준으로 설명하게 하고, 어려운 문단을 쉬운 말로 바꾸게 하고, 내 설명의 빈틈을 물어보게 할 수 있습니다. 하지만 마지막에는 내가 다시 읽고, 내가 다시 확인하고, 내가 내 말로 설명할 수 있어야 합니다. 설명하지 못하는 지식은 아직 내 것이 아닙니다.

의생명과학은 본래 여러 층위를 오가는 학문입니다. DNA 염기 하나에서 시작해 단백질, 세포, 조직, 환자, 인구집단, 보건의료 시스템으로 올라갑니다. 여기에 데이터 과학과 AI가 더해지면 층위는 더 많아집니다. 실험실의 관찰이 데이터 파일이 되고, 데이터 파일이 통계 모델을 지나 그림이 되고, 그 그림이 논문의 주장으로 바뀝니다. AI는 이 과정의 여러 지점에 들어올 수 있습니다. 그래서 학생은 생물학도 알아야 하고, 데이터도 알아야 하며, AI의 한계도 알아야 합니다.

우리 학부가 의생명과학을 가르치는 방식도 이 지점과 맞닿아 있습니다. 세포와 분자 수준의 생명현상을 배우는 일은 여전히 기초입니다. 그러나 그 지식은 질병의 예방과 진단, 치료 전략, 바이오헬스 산업, 데이터 기반 의과학으로 이어집니다. 단백질 하나의 기능을 아는 것과 환자군의 데이터를 해석하는 것은 다른 기술처럼 보이지만, 실제 연구에서는 둘이 계속 만납니다. 어떤 유전자가 세포 안에서 무엇을 하는지 이해하지 못하면 큰 데이터에서 나온 후보를 해석하기 어렵고, 데이터를 다룰 줄 모르면 현대 생명과학이 만들어내는 증거를 충분히 읽기 어렵습니다.

자기 질문을 만드는 대학

이 때문에 1학년 학생에게 필요한 공부는 둘 중 하나를 고르는 일이 아닙니다. "나는 생물학만 할래" 또는 "나는 데이터만 할래"로 나누기에는 연구의 현실이 이미 섞여 있습니다. 면역세포의 분화, 암세포의 약물 저항성, 신경발달의 시간표, 장내미생물과 숙주의 상호작용 같은 질문들은 모두 생물학적 직관과 데이터 해석을 함께 요구합니다. AI는 이 두 세계 사이에 놓이는 번역기처럼 보일 수 있지만, 번역기가 있다고 해서 두 언어를 몰라도 되는 것은 아닙니다. 오히려 번역이 맞는지 확인하려면 두 언어를 어느 정도 읽을 줄 알아야 합니다.

학생에게 남는 또 하나의 질문은 속도와 깊이의 균형입니다. AI는 공부의 속도를 올려줍니다. 모르는 개념을 빠르게 설명받고, 어려운 문단을 몇 단계 난이도로 풀어보고, 코드를 고치고, 발표문 초안을 만들 수 있습니다. 그러나 속도가 빨라질수록 놓치는 것도 생깁니다. 잘 모르는 문장을 대충 이해한 것처럼 지나가고, 모델이 만들어준 설명을 내 생각으로 착각하고, 내가 실제로 무엇을 모르는지 확인하지 않은 채 다음 단계로 넘어갈 수 있습니다. 빠르게 배우는 도구가 생겼기 때문에, 천천히 확인하는 습관도 함께 배워야 합니다.

저는 여기서 대학 교육의 의미가 새롭게 생긴다고 생각합니다. AI가 답을 많이 줄수록 대학은 단순히 정답을 전달하는 곳이기 어려워집니다. 학생이 자기 질문을 만들고, 그 질문을 자료와 연결하고, 자신의 언어로 설명하는 훈련이 더 중요해집니다. 어떤 학생은 암의 면역치료가 궁금할 수 있고, 어떤 학생은 뇌 발달이 궁금할 수 있으며, 어떤 학생은 희귀질환 데이터 분석이 궁금할 수 있습니다. AI는 각각의 길에서 도움을 줄 수 있지만, 어느 길을 걸을지 선택하는 욕구는 학생에게서 나와야 합니다.

이 책은 여러분을 LLM 전문가로 만들려는 책이 아닙니다. **대신 앞으로 계속 만나게 될 질문을 피하지 않게 하려는 책입니다.** AI가 답을 잘 만들어줄 때 나는 무엇을 더 배워야 할까. AI가 코드를 대신 써줄 때 나는 코드를 어디까지 이해해야 할까. AI가 자료를 요약해줄 때 나는 원문을 어떻게 확인해야 할까. AI가 연구의 일부를 자동화할 때 사람 연구자의 책임은 어디에 남을까.

이 질문들에 답하려면 결국 원리를 조금 알아야 합니다. 그래서 이제 우리는 기술의 안쪽으로 천천히 들어갑니다. 다만 처음부터 수식과 구조의 이름을 외우기 위해서가 아닙니다. AI와 함께 공부하고 연구하는 사람이 어떤 판단을 해야 하는지 알기 위해서입니다. 원리는 책임 있는 사용을 위한 최소한의 지도입니다.

앞으로의 장들을 읽을 때도 같은 마음을 유지하면 좋겠습니다. 토큰, 매개변수(parameter), 문맥 창(context window), 강화학습(reinforcement learning) 같은 말은 처음에는 낯설고 딱딱하게 느껴질 수 있습니다. 그러나 이 단어들은 시험에 쓰기 위한 용어가 아니라, 모델의 답을 더 잘 읽기 위한 손잡이입니다. 토큰을 알면 왜 모델이 글자 세기에서 실수하는지 보이고, 문맥 창을 알면 왜 자료를 함께 주어야 하는지 보이며, 환각(hallucination)을 알면 왜 출처 확인이 필요한지 보입니다. 그러면 기술 설명은 더 이상 공학 전공자만의 이야기가 아닙니다. 의생명과학 학생이 논문을 읽고 데이터를 다루고 AI와 대화할 때 매일 만나는 판단의 언어가 됩니다. 이 책의 목표도 바로 그 정도의 이해입니다. 모든 수식을 증명하지 않아도 괜찮습니다. 대신 모델이 무엇을 잘하고, 무엇을 흐릿하게 말하며, 어느 지점에서 사람이 다시 확인해야 하는지를 알아차릴 수 있어야 합니다. 그 알아차림이 AI 시대의 첫 번째 연구 윤리입니다.

> **작은 실습**
>
> 같은 주제를 세 가지 방식으로 물어보십시오. 첫째, "단일세포 RNA-seq을 설명해줘." 둘째, "단일세포 RNA-seq을 고등학교 생명과학을 마친 학생에게, 조직 평균과 비교해서 설명해줘." 셋째, "단일세포 RNA-seq이 질병 연구에서 왜 유용한지 설명하되, 아직 확실히 말할 수 없는 한계도 함께 적어줘." 세 답변에서 설명의 범위, 조심스러움, 생물학적 예시가 어떻게 달라지는지 한 문단으로 적어보면, 프롬프트가 단순한 질문문이 아니라 생각의 틀이라는 사실이 보이기 시작합니다.

4장. ChatGPT는 무엇을 하고 있는가

화면의 문장과 안쪽의 계산

ChatGPT에게 질문을 던지면 화면에는 답변이 문장처럼 흘러나옵니다. 그래서 우리는 자연스럽게 그 안쪽에 어떤 사람이 앉아 있는 것처럼 느낍니다. 누군가가 질문을 읽고, 생각하고, 문장을 골라서 적어주는 장면을 상상하게 됩니다. 그러나 카파시는 이 익숙한 느낌을 잠시 내려놓고, 입력창 뒤에서 실제로 벌어지는 일을 훨씬 낮은 층위에서 보라고 말합니다. 사용자의 문장이 들어오면 그것은 먼저 작은 조각으로 잘리고, 모델은 지금까지 놓인 조각들을 보고 다음에 올 조각을 하나 고릅니다. 그 조각이 붙으면 다시 같은 일이 반복됩니다. 긴 답변도 한 번에 완성된 생각으로 튀어나오지 않습니다. 다음 조각을 예측하고 붙이는 과정이 매우 빠르게 이어진 결과입니다 (링크). 이 말을 처음 들으면 조금 허무할 수 있습니다. 우리가 방금 읽은 친절한 설명이 사실은 거대한 자동완성이라는 뜻처럼 들리기 때문입니다. 하지만 바로 그 지점에서 LLM을 이해하는 첫 번째 문이 열립니다.

용어 메모

LLM: Large Language Model의 줄임말입니다. 많은 글을 학습해 다음에 올 말을 예측하는 큰 언어 모델입니다.

토큰: 모델이 문장을 한 번에 통째로 읽지 못하므로 잘라서 보는 작은 글자 조각입니다.

다음 토큰 예측: 지금까지의 글 뒤에 어떤 조각이 올지 맞히는 훈련 목표입니다.

처음 이 설명을 접할 때 학생들이 가장 자주 하는 반응은 두 가지입니다. 하나는 “그렇다면 ChatGPT는 정말 생각하지 않는 건가요?”라는 질문이고, 다른 하나는 “그런데 왜 이렇게 똑똑해 보이나요?”라는 질문입니다. 두 질문은 모두 중요합니다. 다음 토큰을 예측한다는 설명은 모델을 단순한 계산 절차로 내려놓지만, 그 절차가 아무 의미 없는 장난이라는 뜻은 아닙니다. 다음 말을 잘 맞히려면 앞 문장의 문법만 보는 것으로는 부족합니다. 질문의 의도, 글의 장르, 생물학 용어의 쓰임, 코드의 구조, 논문 초록의 전개까지 어느 정도 배워야 합니다. 그래서 낮은 층위의 목표는 단순해 보여도, 그 목표를 거대한 규모로 훈련하면 높은 층위의 행동이 나타납니다. 과학에서는 이런 일이 낯설지 않습니다. DNA 염기 네 종류만으로도 세포와 몸의 복잡한 현상이 만들어지듯, 단순한 구성 요소가 층층이 쌓이면 놀라운 복잡성이 생길 수 있습니다. 다만 LLM의 복잡성은 생명의 복잡성과 같은 것은 아니며, 그 차이를 기억해야 합니다.

다음 조각을 고르는 모델

여기서 중요한 것은 “자동완성”이라는 말이 이 도구를 깎아내리는 표현이 아니라는 점입니다. 스마트폰 키보드가 다음 단어를 추천하는 자동완성도 같은 방향의 작은 예입니다. 다만 LLM은 그 일을 상상하기 어려울 만큼 큰 규모에서, 훨씬 복잡한 문맥 위에서, 수많은 패턴을 배운 신경망으로 수행합니다. 신경망이라는 말이 어렵게 느껴지면, 처음에는 수많은 작은 계산 단위가 층층이 연결된 거대한 계산 사슬을 떠올리면 됩니다. 각 단위는 들어온 숫자를 조금 바꾸어 다음 단위로 넘기는 단순한 일을 합니다. 한두 개만 있으면 별일을 못하지만, 그런 단위가 아주 많이 쌓이고 학습을 통해 연결의 세기가 조정되면 복잡한 패턴을 표현할 수 있습니다. 이것은 생물학적 신경세포를 그대로 흉내 낸 뇌가 아닙니다. 이름은 비슷하지만, 여기서의 신경망은 입력된 토큰의 줄을 다음 토큰의 확률로 바꾸는 수학적 장치에 가깝습니다. 모델은 바로 앞 단어 하나만 보지 않습니다. 입력창 안의 긴 대화와 자료를 토큰의 긴 줄로 함께 받아들입니다. 그 줄 안에는 사용자의 질문, 이전 답변, 보이지 않는 시스템 메시지(system message), 도구 사용 결과, 붙여넣은 수업 자료나 짧은 초록이 함께 들어갈 수 있습니다. 모델은 그 전체를 보고 다음 토큰의 확률을 계산합니다. 여러 후보 중 어떤 토큰이 나올지 확률적으로 선택하고, 그렇게 선택된 토큰이 다음 선택의 조건이 됩니다. 그래서 같은 질문을 두 번 던져도 답이 조금씩 달라질 수 있습니다. 이것은 모델이 변덕스럽기 때문이라기보다, 확률분포에서 토큰을 샘플링하는 방식으로 문장이 만들어지기 때문입니다.

용어 메모

신경망: 많은 계산 단위가 층층이 연결되어 입력을 출력으로 바꾸는 수학적 구조입니다.

확률분포: 여러 후보가 각각 얼마나 나올 법한지 숫자로 나누어 둔 것입니다.

샘플링: 가장 그럴듯한 후보들 중에서 실제로 하나를 뽑는 과정입니다.

확률분포도 처음에는 어렵게 들리지만, 아주 단순한 예에서 출발할 수 있습니다. 동전을 던지면 앞면과 뒷면에 각각 0.5의 확률을 줄 수 있습니다. LLM의 다음 토큰 선택은 동전 두 면보다 훨씬 큽니다. 모델은 수만 개, 때로는 10만 개가 넘는 후보 토큰마다 “다음에 올 법한 정도”를 매깁니다. 어떤 후보는 매우 높고, 어떤 후보는 거의 0에 가깝습니다. 가장 높은 후보가 자주 선택되지만, 설정에 따라 두 번째나 세 번째로 그럴듯한 후보가 뽑힐 수도 있습니다. 그래서 같은 질문을 다시

던졌을 때 문장이 조금 달라지고, 그 작은 차이가 뒤쪽 문장의 방향을 바꿀 수 있습니다. 이 차이를 알면 "모델이 왜 매번 똑같이 답하지 않는가"라는 질문이 조금 덜 신비롭게 느껴집니다.

작은 문장을 떠올려봅시다. "오늘 생명과학 수업에서 세포를"이라고 쓰면, 다음에는 "관찰했다", "배웠다", "염색했다" 같은 말이 이어질 가능성이 높습니다. 반대로 "삼겹살을"이나 "비행기를" 같은 말이 바로 이어질 가능성은 낮습니다. 사람은 이 차이를 상식과 문맥으로 느낍니다. LLM은 그 차이를 후보 토큰들의 확률로 계산합니다. 물론 실제 모델은 이렇게 단순한 세 단어만 놓고 고르지 않습니다. 앞뒤 문장, 질문의 목적, 사용자가 붙여넣은 자료, 이전 대화까지 긴 토큰열을 함께 봅니다. 처음에는 여기까지만 이해해도 충분합니다. 모델은 "정답 문장"을 창고에서 꺼내는 것이 아니라, 지금까지의 문맥에서 다음 조각으로 가장 그럴듯한 것들을 고르고 이어 붙입니다.

대학교 1학년 학생에게 이 설명이 필요한 이유는 분명합니다. **ChatGPT의 답변은 사람의 설명처럼 보이지만, 그 내부는 사람의 사고와 다릅니다.** 사람은 질문을 읽고 잠시 멈추어 계획을 세우고, 머릿속에서 여러 개념을 붙잡고, 때로는 잘 모르면 검색을 합니다. LLM은 기본적으로 입력된 토큰열과 이미 학습된 매개변수를 거쳐 다음 토큰을 예측합니다. 물론 오늘날의 ChatGPT는 검색, 코드 실행, 이미지 이해, 파일 읽기 같은 도구와 결합되어 훨씬 복잡하게 작동합니다. 그래도 가장 바닥에는 "지금까지의 토큰 다음에 무엇이 올 법한가"라는 일이 있습니다. 이 바닥을 알면 답변을 다르게 읽게 됩니다. 모델이 자신 있게 말한다고 해서 실제로 어딘가의 데이터베이스에서 값을 꺼내온 것은 아닐 수 있습니다. 모델이 "제가 보기에는"이라고 말한다고 해서 사람처럼 자기 의견을 가진 것도 아닙니다. 화면에 나타난 문장은 모델이 학습한 패턴과 현재 문맥이 만난 자리에서 생성된 결과입니다.

용어 메모

매개변수(parameter): 모델 안에 저장된 수많은 숫자입니다. 학습을 거치며 조금씩 조정됩니다.

문맥(context): 모델이 지금 답변을 만들 때 눈앞에 놓고 참고하는 입력 자료입니다.

시스템 메시지(system message): 사용자에게는 보이지 않지만 모델의 말투와 규칙을 정하는 지시문입니다.

베이스 모델과 어시스턴트

카파시는 이 과정을 설명할 때 베이스 모델(base model)과 어시스턴트(assistant)를 구분합니다. 베이스 모델은 인터넷 문서의 흐름을 흉내 내는 토큰 시뮬레이터에 가깝습니다. "2 더하기 2는?"이라고 물으면 반드시 "4입니다"라고 친절하게 답하는 것이 아니라, 인터넷 어딘가에 있을 법한 문장을 이어가려 합니다. 때로는 답을 하기도 하고, 때로는 철학적 문장으로 흘러가기도 하고, 때로는 질문과 답변이 섞인 웹페이지처럼 이어가기도 합니다. 우리가 익숙한 ChatGPT는 여기에 대화 데이터와 추가 훈련이 얹힌 어시스턴트입니다. 그래서 질문에 답하고, 사용자의 의도를 따라가고, 위험한 요청을 거절하고, 모르는 내용은 조심스럽게 말하도록 훈련됩니다. **하지만 어시스턴트가 되었다고 해서 바닥의 원리가 사라지는 것은 아닙니다.** 여전히 대화도 토큰열이고, 답변도 다음 토큰 예측의 반복입니다.

용어 메모

베이스 모델(base model): 질문에 친절히 답하도록 길들여지기 전, 글의 흐름을 이어 쓰는 능력을 먼저 배운 모델입니다.

어시스턴트(assistant): 사용자의 질문에 도움이 되는 답변을 하도록 추가 훈련된 모델의 사용 형태입니다.

추가 훈련: 이미 배운 모델을 특정 행동 방식에 맞게 더 가르치는 단계입니다.

의생명과학 공부에서는 이 구분이 특히 중요합니다. "이 유전자의 기능을 설명해줘"라는 질문에 모델이 답할 때, 그 답은 논문 데이터베이스에서 해당 유전자의 최신 기능 주석(annotation)을 정확히 조회한 결과가 아닐 수 있습니다. 모델은 훈련 데이터에서 자주 본 유전자와 경로와 질병의 문맥을 바탕으로 그럴듯한 설명을 만들 수 있습니다. BRCA1, TP53, EGFR처럼 많이 등장하는 이름에 대해서는 꽤 안정적으로 말할 수 있지만, 드문 유전자나 새로 보고된 변이에 대해서는 흐릿한 기억에 기대기 쉽습니다. 따라서 ChatGPT를 사용할 때 우리는 두 층위를 나누어야 합니다. 하나는 설명의 초안을 얻는 층위입니다. 다른 하나는 그 설명이 실제 근거와 맞는지 확인하는 층위입니다. 모델이 무엇을 하고 있는지 안다는 것은 이 두 층위를 섞지 않는다는 뜻입니다. 학생이 AI를 잘 쓴다는 말은 더 멋진 프롬프트(prompt)를 외운다는 뜻이 아니라, 생성된 문장을 원리와 한계 속에서 읽을 수 있다는 뜻입니다.

용어 메모

기능 주석(annotation): 유전자나 단백질에 대해 알려진 기능, 위치, 관련 질병 같은 설명을 붙여 둔 정보입니다.

프롬프트(prompt): 사용자가 모델에게 주는 질문이나 지시문입니다.

이 장면을 실험실에 비유하면 조금 더 선명해집니다. 현미경 사진을 본다고 해서 세포의 모든 생화학적 상태를 직접 보는 것은 아닙니다. 우리는 염색된 신호와 해상도와 노출 조건을 통해 세포의 일부를 읽습니다. 수업 실험에서 얻은 숫자도 마찬가지입니다. 세포 수, 흡광도, 반응 시간 같은 값은 그냥 하늘에서 떨어진 숫자가 아니라, 어떤 기구로 어떤 조건에서 어떻게 잰 결과입니다. ChatGPT의 답변도 비슷합니다. 답변은 모델 내부의 모든 것을 투명하게 보여주는 창이 아니라, 특정 입력과 학습된 매개변수가 만들어낸 관측값입니다. 그래서 좋은 사용자는 답변을 그대로 믿기보다, 그 답변이 어떤 조건에서 나왔는지 생각합니다. 어떤 자료를 넣었는지, 검색을 했는지, 코드 실행을 했는지, 모델이 기억에만 의존했는지 살핍니다. 이 태도가 있어야 LLM을 공부와 연구의 도구로 안전하게 사용할 수 있습니다.

답변을 실험처럼 읽기

이 관점에서 보면 ChatGPT의 대화창은 단순한 질문 상자가 아닙니다. 사용자가 입력한 문장만 들어가는 곳도 아닙니다. 실제 서비스에서는 보이지 않는 지시문이 앞에 붙고, 이전 대화가 이어지고, 사용자가 올린 파일의 일부가 들어가고, 검색이나 코드 실행 결과가 다시 대화 안으로 들어올 수 있습니다. 카파시는 사용자가 보고 있는 말풍선 뒤에, 모델이 읽는 더 긴 토큰열이 존재한다는 점을 계속 강조합니다 (링크). 우리가 "이 질문에 답했다"고 느끼는 순간에도, 모델 입장에서는 "이 긴 토큰열 다음에 어떤 토큰이 올 가능성이 높은가"를 계산하고 있는 것입니다. 이 차이를 이해하면 프롬프트를 쓰는 태도도 달라집니다. 질문을 짧고 멋있게 던지는 것보다, 모델이 읽어야 할 조건과 자료와 원하는 답변의 형태를 차분히 제공하는 편이 더 중요해집니다. 사람에게도 "그 논문 좀 봐줘"라고만 말하면 애매하지만, "이 논문의 연구 질문, 데이터, 통계 방법, 결론의 강도를 따로 봐줘"라고 말하면 훨씬 좋은 대화를 할 수 있습니다. LLM도 마찬가지로, 좋은 답변은 좋은 문맥에서 시작합니다.

또 하나 눈여겨볼 점은 모델이 문장을 만들 때마다 세계를 다시 계산하는 것이 아니라는 사실입니다. 모델은 이미 학습을 마친 매개변수를 들고 있고, 대화가 들어오면 그 매개변수를 사용해 다음 토큰의 확률을 계산합니다. 이때 매개변수는 모델이 지금까지 읽은 텍스트의 압축된 흔적입니다. 대학생이 한 학기 동안 생물학을 공부하고 시험장에서 기억을 꺼내 쓰는 것과 비슷해 보이지만, 완전히 같지는 않습니다. 사람은 "내가 어디까지 아는가"를 어느 정도 느끼고, 모르면 책을 찾아보거나 교수님에게 질문할 수 있습니다. LLM은 그런 자기 인식을 기본으로 갖고 있지 않습니다. 다만 후속훈련(post-training)과 도구 사용을 통해 모를 때 조심하는 행동, 검색을 요청하는 행동, 계산을 코드로 넘기는 행동을 배울 수 있습니다. 그래서 ChatGPT를 볼 때는 모델 자체의 능력과, 그 모델을 둘러싼 서비스의 장치를 함께 보아야 합니다. 같은 LLM이라도 어떤 시스템 메시지가 붙었는지, 어떤 도구(tool)가 허용되었는지, 어떤 안전 규칙이 걸려 있는지에 따라 사용 경험이 달라질 수 있습니다.

의생명과학 학생에게 이 점은 작은 세부사항이 아닙니다. 예를 들어 "이 실험 결과를 설명해줘"라고 물었을 때, 모델은 꽤 그럴듯한 해석을 만들어낼 수 있습니다. 그러나 모델이 실제 표를 보고 말한 것인지, 사용자가 적어준 몇 문장만 보고 추측한 것인지, 아니면 일반적으로 교과서에 자주 나오는 설명을 떠올린 것인지는 서로 다릅니다. 첫 번째 경우에는 눈앞의 자료가 근거가 되고, 두 번째 경우에는 제한된 정보 안에서의 추론이 되며, 세 번째 경우에는 모델의 흐릿한 사전지식이 주로 작동합니다. 겉으로는 세 경우 모두 자연스러운 한국어 문장으로 보일 수 있습니다. 하지만 과학적으로는 전혀 다른 신뢰도를 가집니다. 그러므로 답변을 평가할 때는 문장의 매끄러움보다 생성 조건을 먼저 보아야 합니다. 어떤 입력을 주었고, 어떤 자료를 확인했고, 어떤 계산을 실행했는지 묻는 습관이 필요합니다. 이 습관은 AI 시대의 문해력입니다. 글을 읽는 능력만이 아니라, 글이 어떤 기계적 과정에서 나왔는지 읽는 능력입니다.

고등학교를 갓 졸업한 학생에게 LLM은 처음에는 마술처럼 보일 수 있습니다. 질문을 넣으면 몇 초 뒤에 설명이 나오고, 영어 문단도 풀어주고, 코드도 만들어주기 때문입니다. 그런데 과학 공부는 마술을 해체하는 일에 가깝습니다. 현상이 놀랍다고 해서 그대로 숭배하지 않고, 원리를 낮은 층위로 내려가 살펴봅니다. LLM도 그렇게 보아야 합니다. 다음 토큰 예측이라는 설명은 이 도구를 시시하게 만들기 위한 말이 아닙니다. 오히려 그 단순한 목표가 거대한 데이터와 모델 규모와 훈련 방법을 만나면 얼마나 풍부한 행동을 만들 수 있는지 보여주는 말입니다. 생명도 DNA, RNA, 단백질, 세포, 조직, 환경이 여러 층에서 만나 나타나는 복잡한 현상입니다. LLM 역시 토큰, 매개변수, 데이터, 문맥, 후속훈련, 도구가 겹쳐 만들어지는 복잡한 현상입니다. 낮은 원리를 안다고 해서 높은 수준의 유용성이 사라지는 것은 아닙니다. 다만 우리는 그 유용성을 더 정확한 자리에서 이해하게 됩니다.

작은 실험으로 익히기

이 장의 관점을 실제 대화에 적용해보면, **ChatGPT 답변은 하나의 결과물이라기보다 하나의 실험 결과처럼 보이기 시작합니다.** 같은 질문이라도 입력한 자료, 지시의 구체성, 모델 설정, 도구 사용 여부에 따라 답이 달라집니다. 실험에서 조건이 바뀌면 결과가 바뀌듯, LLM에서도 문맥이 바뀌면 생성되는 문장이 달라집니다. 그래서 좋은 사용자는 답변 하나를 보고 바로 결론으로 가지 않습니다. "이 답은 어떤 조건에서 나왔는가"를 먼저 묻습니다. 내가 충분한 자료를 주었는지, 모델이 최신 정보를 확인했는지, 계산이 필요한 일을 말로 처리하지는 않았는지, 답변 안의 확실성과 추측이 구분되어

있는지 살핍니다. 이렇게 읽으면 LLM은 더 이상 신탁처럼 보이지 않습니다. 강력하지만 조건에 민감한 도구로 보입니다. 과학을 공부하는 학생에게는 바로 이 시선이 필요합니다.

처음에는 이 시선이 AI의 매력을 줄이는 것처럼 느껴질 수 있습니다. 모델이 사람처럼 생각하는 것이 아니라고 말하면, 방금 전까지 느꼈던 놀라움이 조금 사라지는 듯합니다. 그러나 실제로는 반대입니다. 원리를 알고 나면 놀라움은 더 깊어집니다. 단순한 다음 토큰 예측이 어떻게 번역, 요약, 코드 작성, 개념 설명, 대화, 계획 세우기처럼 다양한 행동으로 이어질 수 있는지 묻게 되기 때문입니다. 생명과학에서도 ATP 합성효소의 회전, DNA 복제의 효소 반응, 신경세포의 전기적 신호를 알게 되면 생명이 덜 신비로워지는 것이 아니라 더 정교하게 놀라워집니다. LLM도 마찬가지입니다. 원리를 안다는 것은 감탄을 버리는 일이 아니라, 감탄을 더 정확하게 만드는 일입니다. 학생은 AI를 막연히 신기해하는 단계에서 벗어나, 왜 신기한지 설명할 수 있는 단계로 가야 합니다. 그때부터 AI는 유행이 아니라 공부의 대상이 됩니다.

이런 공부의 첫 연습은 아주 작게 시작할 수 있습니다. 같은 질문을 두 번 던져보고 답이 어떻게 달라지는지 비교해보는 것입니다. 이번에는 아무 자료 없이 묻고, 다음에는 짧은 초록이나 수업 자료를 넣고 묻고, 그다음에는 "모르는 것은 모른다고 말하라"는 조건을 붙여 묻는 식입니다. 그러면 모델이 하나의 고정된 지식 창고처럼 답하는 것이 아니라, 주어진 문맥과 지시에 따라 다른 방식으로 문장을 만든다는 사실이 눈에 보입니다. 또 답변의 문체와 구조도 바뀝니다. 어떤 답은 자신감 있게 넓은 설명을 하고, 어떤 답은 주어진 자료 안에서 조심스럽게 움직입니다. 학생은 이 차이를 관찰하면서 LLM의 작동 방식을 몸으로 익힐 수 있습니다. 이때 중요한 것은 모델을 시험해 망신 주는 것이 아닙니다. 과학 실험에서 조건을 바꾸어 현상을 이해하듯, 프롬프트와 문맥을 바꾸어 생성의 조건을 이해하는 것입니다. 그렇게 보면 ChatGPT와의 대화도 작은 실험실이 됩니다. 그리고 그 실험실에서 가장 중요한 장비는 사용자의 질문입니다.

이 연습을 반복하다 보면, 학생은 답변을 읽는 속도도 달라집니다. 처음에는 내용만 봅니다. 시간이 지나면 답변의 근거, 조건, 생략된 가정이 보이기 시작합니다. 모델이 "일반적으로"라고 말할 때 정말 일반적인지, "최근 연구에서는"이라고 말할 때 실제로 최신 자료를 본 것인지, "가능성이 있습니다"라고 말할 때 어느 정도의 불확실성을 뜻하는지 묻게 됩니다. 이런 질문은 까다롭게 굴기 위한 것이 아닙니다. 과학적 문장을 읽는 기본 자세입니다. LLM은 그 자세를 매일 연습하게 해줍니다. 답변이 빨리 나오기 때문에, 우리는 더 자주 확인하고 더 자주 고칠 수 있습니다. 그러므로 ChatGPT가 무엇을 하고 있는지 이해한다는 것은 내부 구조를 암기하는 일에서 끝나지 않습니다. 생성된 문장을 과학적으로 읽는 눈을 기르는 일로 이어집니다.

처음에는 이런 읽기가 느리게 느껴질 수 있습니다. 친구들은 AI 답변을 그대로 받아 빠르게 과제를 끝내는 것처럼 보이는데, 나는 왜 자꾸 근거를 찾고 조건을 따지는지 답답할 수도 있습니다. 그러나 과학 공부에서 느린 확인은 낭비가 아닙니다. 오히려 나중에 더 큰 오류를 막아주는 시간입니다. 모델이 무엇을 하고 있는지 아는 학생은 답변을 버리는 법도 배우고, 필요한 부분만 가져오는 법도 배우며, 부족한 자료를 다시 넣어 더 나은 답을 얻는 법도 배웁니다. 이것이 ChatGPT를 단순한 답변기가 아니라 공부 도구로 바꾸는 차이입니다.

5장. 텍스트가 토큰과 확률이 되는 과정

사람이 읽는 문장과 모델이 보는 조각

사람은 문장을 볼 때 글자와 단어와 의미를 거의 동시에 느낍니다. "안녕하세요"라는 말을 보면 인사라는 느낌이 먼저 오고, "세포가 신호를 받았다"라는 문장을 읽으면 세포와 신호와 반응이라는 장면이 떠오릅니다. 하지만 컴퓨터는 처음부터 그런 방식으로 문장을 보지 않습니다. 컴퓨터 안에서 모든 것은 결국 숫자와 기호의 배열로 바뀌어야 합니다. LLM도 마찬가지입니다. 모델은 화면에 보이는 한글 문장이나 영어 문장을 그대로 보는 것이 아니라, 먼저 그 문장을 토큰이라는 조각들의 줄로 바꿔서 봅니다. 카파시는 텍스트가 신경망으로 들어가기 전에 먼저 "1차원의 기호열"이 되어야 한다고 설명합니다 (링크). 처음에는 긴 문장을 낱말 카드와 글자 조각 카드로 잘라 책상 위에 한 줄로 놓는 장면을 떠올리면 됩니다. 모델은 그 카드들을 차례로 보며 다음에 올 카드를 예측합니다.

생명과학 학생이라면 DNA 염기서열을 A, T, C, G의 문자열로 표현하는 일을 떠올릴 수도 있습니다. 그러나 이 비유는 조심해서 써야 합니다. DNA 염기는 정해진 네 종류의 생물학적 분자이고, 토큰은 모델이 텍스트를 계산하기 위해 만든 기호 단위입니다. 토큰의 종류는 훨씬 많고, 길이도 들쭉날쭉하며, 세 글자씩 끊는 코돈 같은 규칙을 따르지도 않습니다. 그러니 "토큰은 DNA 염기와 같다"고 외우면 오히려 위험합니다. 두 경우가 닮은 지점은 더 좁습니다. 복잡한 대상을 컴퓨터가 다룰 수 있는 긴 줄로 바꾸어야 한다는 점입니다. 처음 읽을 때는 여기까지만 붙잡고 넘어가도 됩니다. 모델은 우리가 문장을 읽듯이 의미를 바로 붙잡지 않고, 먼저 문장을 계산 가능한 조각들로 바꿉니다. 이 조각이 어떻게 나뉘느냐에 따라 모델이 잘하는 일과 실수하는 일이 달라집니다.

토큰은 단어가 아니다

가장 단순하게 생각하면 글자를 하나씩 기호로 삼을 수도 있습니다. 더 아래로 내려가면 컴퓨터의 byte나 bit를 기호로 삼을 수도 있습니다. 그러나 그렇게 하면 토큰열이 너무 길어집니다. 신경망에서 문맥 길이는 귀한 자원입니다. 같은 의미를 너무 긴 기호열로 표현하면, 모델은 제한된 문맥 안에 더 적은 정보를 담게 됩니다. 그래서 현대 LLM은 보통 글자보다 크고 단어보다 작은 조각들을 사용합니다. 자주 함께 등장하는 문자 조합이나 단어 조각을 하나의 토큰으로 묶습니다. "hello world"가 두 개의 토큰이 될 수도 있고, 공백이 붙은 단어 조각이 하나의 토큰이 될 수도 있습니다. 예를 들어 영어에서 `ing`나 `tion`처럼 자주 반복되는 조각은 하나의 덩어리로 다루는 편이 효율적일 수 있습니다. 카파시는 byte-pair encoding 같은 알고리즘을 통해 자주 붙어 다니는 기호 쌍을 새 토큰으로 만들어 토큰열을 줄이는 과정을 보여줍니다 (링크). 알고리즘 이름을 지금 외울 필요는 없습니다. 중요한 것은 모델이 글을 사람이 보는 단어 그대로 읽지 않고, 자주 쓰이는 조각을 학습된 규칙에 따라 묶고 자른다는 점입니다.

용어 메모

토큰화(tokenization): 문장을 모델이 다룰 수 있는 작은 조각들로 나누는 과정입니다.

토큰화 도구(tokenizer): 문장을 어떤 토큰들로 나눌지 정하는 도구입니다.

byte / bit: 컴퓨터가 문자를 숫자로 다룰 때 쓰는 아주 작은 정보 단위입니다.

토큰열: 토큰들이 앞에서 뒤로 이어진 줄입니다.

토큰은 사람이 생각하는 단어와 정확히 일치하지 않습니다. 영어에서는 공백 때문에 단어 경계가 비교적 잘 보이지만, 그래도 토큰은 단어와 다르게 잘릴 수 있습니다. 한글이나 유전자 이름, 특수기호가 섞인 문장에서는 더 복잡합니다. 모델은 "BRCA1"을 사람이 보는 것처럼 하나의 유전자 이름으로만 보는 것이 아니라, 토큰화 도구가 정한 몇 개의 조각으로 볼 수 있습니다. 어떤 희귀한 유전자명은 여러 조각으로 갈라질 수 있고, 하이픈이나 숫자나 그리스 문자가 섞인 단백질 이름도 예상과 다르게 나뉠 수 있습니다. 그래서 LLM이 철자 세기, 특정 글자 찾기, 문자열 조작에 약한 경우가 생깁니다. 사람은 글자를 눈으로 보지만, 모델은 토큰을 봅니다. 우리가 "strawberry에 r이 몇 개 있나" 같은 질문을 쉽게 느끼는 이유는 글자를 직접 볼 수 있기 때문입니다. 모델에게는 그 단어가 몇 개의 토큰으로 보일 수 있고, 그 토큰 안의 글자 구조를 정확히 다루는 일은 생각보다 어려울 수 있습니다.

이 점은 의생명과학 데이터에서도 곧장 문제가 됩니다. 유전자 symbol, 변이 표기, 단백질 isoform, chemical ID는 문자 하나가 중요합니다. `TP53`과 `TP63`은 다르고, `BRCA1`과 `BRCA2`는 다르며, 변이 표기에서 숫자 하나가 바뀌면 전혀 다른 위치가 됩니다. LLM이 이런 문자열을 설명하는 데 도움을 줄 수는 있지만, 정확한 표기와 계산은 반드시 원문이나 데이터베이스나 코드로 확인해야 합니다. 모델에게 "이 유전자 목록에서 중복을 제거해줘"라고만 맡기면, 눈에 보이는 문자열을 사람이 하듯이 다루리라고 기대하기 쉽습니다. 그러나 안전한 방법은 다릅니다. 목록을 파일로 주고, 코드로 중복을 확인하게 하고, 실행 결과를 보게 해야 합니다. 토큰화의 한계를 이해하면 이런 사용 습관이 자연스럽게 생깁니다. 모델이 언어를 잘 다룬다고 해서 모든 문자열 작업을 정확히 한다는 뜻은 아니기 때문입니다.

확률로 이어지는 문장

토큰화가 끝나면 모델은 이제 다음 토큰을 예측하는 문제를 풀기 시작합니다. 어떤 문맥이 주어졌을 때, 가능한 모든 토큰에 대해 "다음에 올 확률"을 계산합니다. vocabulary가 10만 개라면 모델은 10만 개 후보 각각에 대한 점수를 내놓습니다. 그중 확률이 높은 토큰이 더 자주 선택되지만, 항상 가장 높은 것만 선택되는 것은 아닙니다. 샘플링 방식에 따라 조금 다른 토큰이 선택될 수 있고, 그 작은 차이가 뒤의 문장을 완전히 다른 방향으로 보낼 수 있습니다. 그래서 LLM은 deterministic한 계산기와 다르게 느껴집니다. 같은 질문에 비슷하지만 조금 다른 답을 주는 이유가 여기에 있습니다. 모델이 사람처럼 그때그때 기분이 바뀌는 것이 아니라, 확률분포에서 토큰을 뽑아 이어가는 생성 방식이 그런 변화를 만듭니다.

용어 메모

vocabulary: 모델이 고를 수 있는 전체 토큰 목록입니다.

샘플링: 확률이 매겨진 후보들 중 실제 다음 토큰 하나를 뽑는 과정입니다.

deterministic: 같은 입력이면 언제나 같은 결과가 나오는 성질입니다.

학생이 이 과정을 알아야 하는 이유는 수식 때문이 아닙니다. 토큰과 확률을 이해하면 LLM의 장점과 약점을 동시에 볼 수 있습니다. 모델은 긴 문맥 속에서 자연스러운 다음 문장을 이어가는 데 뛰어납니다. 논문 초록의 구조, 생물학 설명의 문체, 코드의 일반적인 패턴, 보고서 문장의 흐름을 잘 배웠기 때문입니다. 하지만 모델은 언제나 확률적으로 그럴듯한

다음 조각을 만들고 있습니다. **그럴듯함은 사실성과 다릅니다.** 특히 의생명과학에서는 그럴듯한 설명이 가장 위험할 수 있습니다. 문장이 자연스럽게 이어진다는 사실과, 그 문장이 실제 유전자 기능이나 실험 결과를 정확히 반영한다는 사실은 따로 확인해야 합니다. 토큰화와 다음 토큰 예측은 LLM을 신비한 지능에서 내려오게 합니다. 그리고 바로 그 덕분에 우리는 이 도구를 더 차분하고 더 실용적으로 사용할 수 있습니다.

문맥 속에서 바뀌는 의미

토큰이 모델 안으로 들어가면 곧바로 의미가 되는 것은 아닙니다. 먼저 각 토큰은 여러 숫자의 묶음으로 바뀝니다. 이것을 embedding이라고 부릅니다. 벡터라는 말이 어렵게 느껴지면, 당장은 "토큰마다 붙는 긴 점수표"라고 생각해도 됩니다. 우리가 책을 분류할 때 소설인지 시인지 한 가지 기준만 보지 않고, 문장 길이, 분위기, 주제, 독자층, 시대 배경 같은 여러 기준을 함께 보듯이, 모델도 토큰을 여러 숫자로 표현합니다. 이 숫자들은 사람이 미리 "생물학성 8점", "동사성 2점"처럼 이름을 붙여 준 점수가 아닙니다. 다음 토큰을 더 잘 맞히도록 훈련되는 동안 모델 안에서 만들어진 표현입니다. 출발점은 간단합니다. 생물학 문장에서 `cell`, `gene`, `protein`은 자주 함께 등장하므로 서로 관련된 방식으로 표현되는 편이 다음 단어를 예측하는 데 도움이 됩니다. 반대로 `cell`이 엑셀 문서에서 칸을 뜻할 때와 생물학 논문에서 세포를 뜻할 때는 주변 문맥이 달라집니다. 실제 모델의 숫자 묶음은 우리가 눈으로 그릴 수 없을 만큼 길지만, 굳이 수백 개의 축을 머릿속에 그리려 애쓸 필요는 없습니다. 중요한 것은 모델이 토큰을 낱말 뜻 하나로 저장하지 않고, 여러 기준의 숫자 표현으로 바꾼 뒤 문맥 속에서 계속 조정한다는 점입니다. 그래서 같은 `cell`도 생물학 논문에서는 세포를 뜻하고, 엑셀 문서에서는 칸을 뜻하고, 감옥 이야기에서는 독방을 뜻할 수 있습니다. LLM이 문맥을 읽는다는 말은 바로 이런 변화를 계산한다는 뜻입니다.

용어 메모

embedding: 토큰을 여러 숫자의 묶음으로 바꾼 표현입니다.

벡터: 여러 숫자를 한 줄로 모아 둔 값입니다. 여기서는 토큰마다 붙는 긴 점수표처럼 생각해도 됩니다.

문맥: 모델이 지금 답을 만들 때 참고하는 앞뒤 문장과 자료입니다.

Transformer에서 attention이 중요한 이유도 여기에 있습니다. 문장 안의 어떤 토큰이 다른 토큰을 얼마나 참고해야 하는지 계산할 수 있기 때문입니다. "그 유전자는 종양 억제 기능을 가진다"라는 문장에서 "그"가 무엇을 가리키는지 이해하려면 앞 문장의 유전자 이름을 봐야 합니다. 조금 더 생물학적으로 써보면 이렇습니다. "BRCA1은 DNA 복구에 관여한다. 이 유전자는 변이가 생기면 유방암 위험과 관련될 수 있다." 두 번째 문장의 "이 유전자"를 처리할 때, 모델은 첫 문장의 `BRCA1` 쪽을 강하게 참고해야 합니다. 사람은 자연스럽게 연결하지만, 모델 안에서는 attention이 이런 연결을 계산합니다. 논문 초록에서는 첫 문장에 질병 이름이 나오고, 뒤 문장에서는 "this disorder"라고만 말할 수 있습니다. 사람은 자연스럽게 연결하지만, 모델도 이런 연결을 수많은 예시 속에서 배웁니다. 카파시는 Transformer의 내부를 토큰들이 여러 층을 지나며 서로 정보를 주고받는 계산으로 설명합니다 (링크). 여기서 attention은 마치 사람이 중요한 단어에 밑줄을 긋는 것처럼 보일 수 있지만, 실제로는 더 복잡한 수치 계산입니다. 그래도 비유는 도움이 됩니다. 한 문장의 의미는 단어 하나하나에 고립되어 있지 않고, 서로의 관계 안에서 만들어집니다. LLM은 이 관계를 매우 큰 규모의 통계적 패턴으로 학습합니다.

용어 메모

attention: 문장 안에서 어떤 토큰을 더 참고할지 계산하는 장치입니다.

Transformer: attention을 여러 층으로 쌓아 문맥 관계를 계산하는 LLM의 대표 구조입니다.

확률이라는 말도 조금 더 천천히 생각할 필요가 있습니다. 모델이 다음 토큰의 확률을 계산한다는 것은, 모델이 "참인 문장"과 "거짓인 문장"을 직접 구분한다는 뜻이 아닙니다. 훈련 목표는 주어진 문맥에서 실제 데이터에 등장했던 다음 토큰에 높은 확률을 주는 것입니다 (링크). 이 목표는 놀라울 만큼 강력합니다. 왜냐하면 다음 토큰을 잘 맞히려면 문법, 상식, 코드 구조, 논리 흐름, 사실의 일부를 배워야 하기 때문입니다. 하지만 목표 자체는 여전히 예측입니다. 의학적으로 올바른 설명을 보상하는 규칙이 처음부터 들어 있는 것이 아닙니다. 잘못된 설명이 인터넷에 많이 있고, 특정 표현이 자주 반복되면, 모델은 그 패턴도 배울 수 있습니다. 그래서 확률이 높다는 것은 "많이 본 흐름에 잘 맞는다"에 가깝고, "실험적으로 검증되었다"와는 다릅니다. 학생은 이 차이를 붙잡아야 합니다. LLM이 생물학을 말할 때도, 그 말은 먼저 언어적 확률의 산물이고, 과학적 사실성은 그다음 단계에서 확인해야 합니다.

정확한 표기를 지키는 습관

한글을 쓰는 우리에게 토큰화는 또 다른 문제를 남깁니다. 많은 LLM과 토큰화 도구는 영어 중심의 데이터와 사용 환경에서 발전했습니다. 물론 오늘날 모델은 한국어도 매우 잘 다루지만, 한국어의 조사, 어미, 띄어쓰기, 한자어, 영어 약어가 섞인 문장은 영어와 다른 방식으로 잘릴 수 있습니다. 먼저 아주 쉬운 예를 떠올려봅시다. 사람에게 "안녕하세요"는 하나의 인사말처럼 보이지만, 모델 안에서는 , 처럼 나뉠 수도 있고, 더 어색한 작은 조각으로 갈라질 수도 있습니다. 공백을 하나 넣거나 빼도 토큰열이 달라질 수 있습니다. 여기에 전문 용어가 들어오면 더 복잡해집니다. "세포분화가 억제되었다"와 "세포 분화가 억제되었다"는 사람에게 거의 같은 뜻으로 읽히지만 토큰열은 달라질 수 있고, "TP53 변이가 관찰되었다" 같은 문장에서는 한글, 숫자, 영어 대문자가 함께 들어갑니다. 생물학 글에서는 유전자 약어, 단백질 이름, 생물학 경로 이름, 그리스 문자, 숫자, 괄호가 계속 섞입니다. 이런 표기가 토큰 단위에서 어떻게 쪼개지는지는 모델의 작은 성능 차이를 만들 수 있습니다. 그래서 학생이 LLM으로 한글 과학 글을 쓸 때는, 중요한 고유명사를 원문 그대로 유지하고, 표기법을 일관되게 쓰고, 모델이 바꾼 용어를 반드시 확인해야 합니다. 부드러운 번역이 항상 좋은 번역은 아닙니다. 특히 유전자 약어나 질병명처럼 표기가 지식의 일부인 경우에는, 자연스러운 문장보다 정확한 표기가 먼저입니다.

이 모든 과정을 지나면 우리는 LLM을 조금 더 편안하게 다룰 수 있습니다. 토큰은 모델이 보는 기본 단위이고, embedding은 그 토큰을 계산 가능한 형태로 바꾼 표현이며, attention과 여러 층의 계산은 문맥 속 관계를 반영하고, 마지막에는 다음 토큰의 확률분포가 나옵니다. 복잡하지만 붙잡을 줄기는 분명합니다. 모델은 글을 사람처럼 눈으로 읽는 것이 아니라, 토큰의 줄을 수학적으로 처리합니다. 그러므로 모델이 잘하는 일은 이 구조와 잘 맞는 일입니다. 긴 설명을 이어가고, 문체를 바꾸고, 낯선 개념을 비슷한 표현으로 풀고, 코드의 일반 패턴을 제안하는 일은 매우 잘합니다. 반대로 문자 하나하나의 정확한 조작, 최신 사실 조회, 검증되지 않은 생물학적 결론, 임상적 판단은 별도의 장치가 필요합니다. LLM을 잘 쓰는 학생은 이 구분을 외우는 것이 아니라, 몸에 익힙니다. 질문을 던지기 전에 "이 일은 언어 패턴의 문제인가, 정확한 계산의 문제인가, 최신 근거 확인의 문제인가"를 잠시 생각합니다. 그 짧은 멈춤이 AI 사용의 질을 크게 바꿉니다.

토큰과 확률을 이해하는 일은 글쓰기에도 도움이 됩니다. 모델이 문장을 만들 때는 이전 문맥에 어울리는 다음 조각을 이어가므로, 사용자가 제공한 문체와 구조를 매우 민감하게 따라갑니다. 딱딱한 bullet과 짧은 명령만 주면 답변도 그 형식을 닮기 쉽고, 긴 설명문과 구체적인 독자를 알려주면 더 부드러운 글이 나올 가능성이 커집니다. 이것은 모델이 인간 작가처럼 취향을 이해한다는 뜻은 아닙니다. 문맥 안에 들어간 패턴을 보고 다음 토큰을 고르는 과정이 그런 효과를 만드는 것입니다. 그래서 한글로 교재를 쓸 때도 입력 자료가 중요합니다. "쉽게 써줘"라고만 하면 흔한 AI식 문단이 나오기 쉽지만, "대학교 1학년 의생명과학 학생에게, 수학을 겁내지 않도록, 실험실 비유를 섞어 긴 설명문으로 써줘"라고 말하면 훨씬 나은 출발점이 생깁니다. 그래도 마지막 윤문은 사람이 해야 합니다. 모델은 그럴듯한 흐름을 만들 수 있지만, 어떤 문장이 우리 학생들에게 실제로 와닿을지는 수업을 알고 독자를 아는 사람이 더 잘 판단할 수 있습니다. 토큰의 원리를 아는 일은 결국 더 나은 질문과 더 나은 글쓰기의 기초가 됩니다.

이 원리는 번역에서도 드러납니다. 영어 강의를 한국어로 옮길 때, 단어를 하나씩 대응시키는 것만으로는 좋은 설명이 되지 않습니다. "매개변수", "문맥", "attention", "loss" 같은 단어는 이미 한국어 과학 글 안에서 영어와 번역어가 섞여 쓰입니다. 어떤 말은 그대로 두는 편이 낫고, 어떤 말은 먼저 쉬운 설명을 붙여야 합니다. LLM은 이런 혼합 문체를 잘 흉내 낼 수 있지만, 때로는 너무 자연스럽게 보이도록 전문 용어의 날카로움을 무디게 만들기도 합니다. 예를 들어 "loss"를 단순히 "손실"이라고만 옮기면, 학생은 무엇을 잃는다는 뜻인지 헷갈릴 수 있습니다. 실제로는 모델의 예측이 정답 토큰과 얼마나 다른지를 나타내는 훈련 신호라고 풀어주어야 합니다. 좋은 과학 번역은 매끄럽기만 해서는 안 됩니다. 낯선 개념이 어디서 낯선지 보여주고, 필요한 만큼 천천히 풀어주어야 합니다. LLM을 글쓰기 도구로 쓸 때도 이 기준을 잊지 않아야 합니다.

용어 메모

매개변수: 모델 안에 저장되어 학습 중 조금씩 바뀌는 수많은 숫자입니다.

loss: 모델의 예측이 정답에서 얼마나 빗나갔는지 나타내는 훈련 신호입니다.

학생이 직접 해볼 수 있는 작은 연습도 있습니다. 짧은 한글 문장, 영어 문장, 유전자 이름이 섞인 문장을 토큰화 도구에 넣어보고 어떻게 잘리는지 관찰해보는 것입니다. 처음에는 별것 아닌 장난처럼 보이지만, 곧 모델이 우리가 보는 글자와 다른 단위로 세상을 본다는 사실이 실감납니다. 한 단어처럼 보이는 것이 여러 토큰으로 갈라지고, 공백 하나가 토큰 구성을 바꾸고, 숫자와 기호가 예상보다 낯선 방식으로 처리될 수 있습니다. 이 경험은 LLM을 더 현실적인 도구로 보게 해줍니다. 모델이 어떤 일을 잘하고 어떤 일에서 실수할지, 추상적인 설명보다 훨씬 빨리 이해됩니다. 특히 의생명과학에서는 표기 하나가 의미를 바꾸기 때문에 이런 차이를 아는 일이 중요합니다. 유전자명과 변이 표기를 모델에게 맡길 때는, 문장의 자연스러움보다 토큰 이전의 원문 표기를 먼저 지켜야 합니다. 토큰화는 작아 보이지만, 정확성의 출발점입니다.

이 연습은 어려운 프로그램을 설치하지 않아도 시작할 수 있습니다. 온라인 토큰화 도구에 짧은 문장을 넣고, 공백을 하나 바꾸거나 영어 약어를 붙여보면 됩니다. 학생은 곧 자신이 보는 단어와 모델이 보는 조각이 다르다는 사실을 눈으로

확인하게 됩니다. 이 작은 경험은 나중에 모델의 실수를 만났을 때 큰 도움이 됩니다. "왜 이렇게 쉬운 글자 세기를 틀리지?"라고 화내기보다, 모델이 글자를 사람처럼 직접 보지 않는다는 사실을 떠올릴 수 있기 때문입니다. 원리를 조금 아는 일은 모델을 용서하자는 뜻이 아니라, 모델의 실수를 더 정확히 다루자는 뜻입니다.

6장. 인터넷 문서에서 베이스 모델까지

웹의 문서가 훈련 데이터가 되기까지

LLM을 만든다는 말은 처음에는 거창하게 들립니다. 마치 연구자가 지식을 하나하나 입력하고, 규칙을 적고, 질문에 대한 답을 데이터베이스처럼 넣어두는 장면을 떠올리기 쉽습니다. 그러나 카파시가 보여주는 첫 단계는 훨씬 투박하고 거대합니다. 먼저 인터넷에서 공개적으로 접근 가능한 텍스트를 엄청나게 모읍니다. Common Crawl처럼 오랫동안 웹을 수집해 온 자료가 출발점이 될 수 있고, 그 원자료에서 광고, 메뉴, HTML 코드, 스팸, 중복 문서, 개인정보, 품질이 낮은 페이지를 걸러냅니다. 카파시는 FineWeb을 예로 들며, 거대한 인터넷이 실제 훈련 데이터가 되기까지 여러 단계의 필터링과 정제가 필요하다고 설명합니다 (링크). "많이 모으면 된다"가 아닙니다. 무엇을 남기고 무엇을 버리는지가 모델의 성격을 바꿉니다. 의생명과학에서도 원자료가 그대로 지식이 되지 않는 것처럼, LLM에서도 웹 전체가 그대로 모델의 배움이 되지는 않습니다.

용어 메모

Common Crawl: 웹페이지를 오랫동안 대규모로 모아 온 공개 웹 자료 모음입니다.

FineWeb: 웹 자료에서 중복과 낮은 품질의 글을 걸러 훈련에 쓰기 좋게 만든 데이터셋입니다.

HTML: 웹페이지의 구조를 표시하는 언어입니다. 메뉴와 광고 같은 표시도 함께 섞일 수 있습니다.

이 과정을 생물학 데이터로 비유하면 이해하기 쉽습니다. 수업 실험에서 나온 숫자도 바로 결론이 되지는 않습니다. 실험 기록을 다시 보고, 단위를 확인하고, 빈칸이나 이상한 값을 살피고, 같은 조건끼리 묶어보아야 비로소 해석할 수 있는 표가 됩니다. 현미경 사진도 마찬가지입니다. 초점이 맞았는지, 염색이 잘 되었는지, 어떤 조건의 세포를 찍었는지 확인하지 않으면 사진만 보고 결론을 내리기 어렵습니다. 인터넷 문서도 비슷합니다. 웹에는 좋은 설명, 논문 초록, 교과서적 글, 코드 예제가 있지만, 동시에 광고 문구, 복붙 문서, 오류가 많은 글, 악성 사이트, 개인정보가 섞여 있습니다. **모델은 자신이 먹은 데이터의 세계를 닮습니다.** 그래서 사전학습 데이터 구축은 단순한 수집이 아니라, 모델이 어떤 문화를 배우고 어떤 문체를 따라 하며 어떤 지식을 자주 기억하게 될지를 정하는 일입니다.

용어 메모

사전학습(pre-training): 모델이 어시스턴트가 되기 전에 많은 글을 읽으며 언어와 지식의 패턴을 배우는 단계입니다.

훈련 데이터: 모델이 배우는 데 사용되는 글, 표, 이미지 같은 자료입니다.

다음 토큰을 맞히는 긴 훈련

텍스트가 정제되면 토큰화 도구를 통해 토큰의 긴 줄로 바뀝니다. 카파시는 FineWeb 같은 데이터가 저장 용량으로는 수십 테라바이트이고, 토큰으로는 수조에서 수십조 개 규모가 될 수 있음을 보여줍니다 (링크). 이 숫자들은 지금 외울 필요가 없습니다. 그냥 "한 학기 강의노트"가 아니라 "도서관 여러 층을 가득 채운 문서"에 가까운 규모라고 생각하면 됩니다. 그다음 모델은 이 긴 토큰열에서 작은 구간을 잘라 다음 토큰을 예측하는 훈련을 반복합니다. 서문에서 말한 비유로 돌아가면, 이것이 모델의 배경지식 쌓기에 해당합니다. 학생이 교과서와 논문을 많이 읽으며 생물학 문장의 흐름을 익히듯, 모델은 수많은 문서의 다음 토큰을 맞히며 언어와 지식의 패턴을 익힙니다. 물론 사람처럼 뜻을 곱씹는 것은 아니지만, 많은 예시를 통해 어떤 표현 뒤에 어떤 말이 이어지는지 배우는 셈입니다. 처음의 신경망은 거의 아무것도 모릅니다. 매개변수가 무작위로 놓여 있기 때문에 출력도 엉망입니다. 그러나 훈련 자료에서는 정답 토큰이 무엇인지 알고 있으므로, 모델의 예측이 그 토큰에 더 가까워지도록 매개변수를 조금씩 조정할 수 있습니다. 이 일이 수많은 토큰과 수많은 수정 단계에서 반복됩니다. 연구자는 loss라는 숫자를 보며 모델이 조금씩 더 나은 예측을 하게 되는지 확인합니다. loss가 내려간다는 것은 모델이 훈련 데이터의 통계적 패턴을 더 잘 맞추고 있다는 뜻입니다.

loss와 매개변수 조정의 연결이 처음에는 비어 있는 것처럼 느껴질 수 있습니다. 지금은 수식을 몰라도 됩니다. 시험에서 답을 맞힌 뒤 채점표를 보고 "어느 부분에서 많이 틀렸는지" 확인하는 장면을 떠올려봅시다. 채점표는 공부한 내용을 직접 고쳐주지는 않지만, 다음에 어디를 고쳐야 할지 방향을 줍니다. 모델의 loss도 그런 신호에 가깝습니다. 예측이 정답에서 많이 빗나가면 loss가 커지고, 훈련 알고리즘은 그 숫자가 조금이라도 작아지는 방향으로 매개변수를 조정합니다. 실제로는

미분과 최적화라는 수학이 들어가지만, 1학년 독자는 우선 “loss는 모델이 얼마나 틀렸는지 알려주는 점수이고, 훈련은 그 점수를 낮추는 방향으로 숫자들을 고치는 반복”이라고 이해해도 충분합니다.

용어 메모

토큰화 도구(tokenizer): 글을 모델이 읽을 수 있는 작은 조각으로 나누는 도구입니다.

테라바이트: 아주 큰 저장 용량 단위입니다. 보통 노트북 저장 공간보다 훨씬 큰 규모를 말할 때 씁니다.

매개변수: 모델 안에 저장되어 학습 중 조정되는 숫자들입니다.

수정 단계(update): 모델의 예측이 조금 더 나아지도록 매개변수를 한 번 고치는 단계입니다.

loss: 예측이 정답 토큰에서 얼마나 빗나갔는지 나타내는 숫자입니다.

여기서 Transformer가 등장합니다. Transformer는 토큰열을 입력으로 받아, 그 안의 토큰들이 서로 어떤 관계를 가지는지 계산하고, 다음 토큰의 확률을 내놓는 신경망 구조입니다. 카파시는 Transformer 내부를 거대한 수학식으로 보라고 설명합니다 (링크). 수많은 매개변수가 있고, 토큰은 embedding으로 바뀌고, attention block과 MLP block을 지나며 여러 중간값이 만들어집니다. block이라는 말은 여기서 복잡한 계산을 묶어 부르는 이름입니다. 지금은 내부 회로를 다 외우지 않아도 됩니다. 더 중요한 것은 오해를 피하는 일입니다. 이때 “neuron”이라는 말을 쓸 수는 있지만, 생물학적 neuron과 같은 것은 아닙니다. 우리 뇌의 neuron은 전기적, 화학적, 시간적 동역학을 가진 매우 복잡한 세포입니다. 축삭, 시냅스, 신경전달물질, 발화 시간 같은 생물학적 요소가 얽혀 있습니다. Transformer 안의 neuron은 그런 세포가 아니라 계산 중간에 생기는 수학적 값에 가깝습니다. 비유는 도움이 되지만, 비유를 그대로 믿으면 오해가 생깁니다. LLM은 생물학적 뇌를 복제한 것이 아니라, 토큰열에서 다음 토큰을 예측하도록 최적화된 거대한 함수에 가깝습니다.

용어 메모

Transformer: 토큰들 사이의 관계를 여러 층에서 계산하는 LLM의 대표 구조입니다.

embedding: 토큰을 계산 가능한 숫자 묶음으로 바꾼 표현입니다.

attention block: 어떤 토큰을 더 참고할지 계산하는 부분입니다.

MLP block: attention 뒤에서 숫자 표현을 한 번 더 바꾸는 계산 부분입니다.

베이스 모델이라는 첫 결과

사전학습이 끝나면 베이스 모델(base model)이 생깁니다. 베이스 모델은 질문에 친절하게 답하는 어시스턴트가 아닙니다. 카파시는 이것을 인터넷 텍스트의 토큰 시뮬레이터라고 부릅니다. 사용자가 어떤 문장을 앞부분(prefix)으로 넣으면, 베이스 모델은 그 뒤에 이어질 법한 인터넷 문서를 생성합니다. 그래서 “2 더하기 2는?”이라고 물었을 때도 반드시 선생님처럼 답하지 않습니다. 질문과 답변 형식의 웹페이지처럼 이어갈 수도 있고, 철학적 문장으로 흘러갈 수도 있으며, 같은 앞부분에서도 매번 다른 이어 쓰기(continuation)를 만들 수 있습니다. 하지만 이 베이스 모델은 이미 엄청난 것을 배웠습니다. 다음 토큰을 맞히는 과정에서 언어의 문법, 사실의 일부, 코드의 패턴, 논문의 문체, 세상의 상식, 사람들이 질문하고 답하는 방식을 매개변수 안에 압축해두었습니다. 카파시는 이 매개변수를 인터넷의 lossy compression, 곧 손실 압축처럼 생각할 수 있다고 말합니다 (링크). 처음 듣는 학생은 손실 압축이라는 말보다, 한 학기 강의를 자기 노트 한 권에 옮겨 적는 장면을 먼저 떠올려도 좋습니다. 노트에는 중요한 흐름과 자주 반복된 설명은 남지만, 교수자의 모든 말과 칠판의 모든 흔적이 그대로 들어가지는 않습니다.

용어 메모

베이스 모델(base model): 질문에 답하도록 길들여지기 전, 글을 이어 쓰는 능력을 먼저 배운 모델입니다.

앞부분(prefix): 모델에게 먼저 주어진 글의 앞부분입니다.

이어 쓰기(continuation): 앞부분 뒤에 모델이 이어 쓰는 글입니다.

손실 압축: 원본을 완벽히 보존하지 않고, 중요한 패턴만 남겨 줄여 담는 방식입니다.

학습 cutoff: 모델 훈련에 들어간 자료가 어느 시점까지였는지를 가리키는 경계입니다.

손실 압축이라는 말은 아주 중요합니다. 베이스 모델은 인터넷의 모든 문장을 정확히 저장한 데이터베이스가 아닙니다. 어떤 내용은 자주 보았기 때문에 꽤 잘 기억하고, 어떤 내용은 흐릿하게만 남아 있으며, 어떤 내용은 아예 모를 수 있습니다. 유명한 유전자나 질병 이름은 많이 등장했기 때문에 안정적으로 설명할 수 있지만, 희귀한 변이나 최근 논문 결과는

부정확할 수 있습니다. 많이 반복된 문서는 때때로 긴 구절을 거의 외워서 말할 수도 있지만, 그것이 모델이 지식을 올바르게 이해했다는 뜻은 아닙니다. 외운다는 것은 이해한다는 것과 다릅니다. 모델이 같은 문장을 정확히 반복할 수 있다고 해서, 그 문장 안의 개념을 새로운 실험 조건이나 낯선 데이터에 바르게 적용할 수 있다는 뜻은 아닙니다. 반대로 드문 사실에 대해서는 그럴듯한 평행우주를 만들어낼 수 있습니다. 카파시가 2024년 선거처럼 학습 cutoff 이후의 일을 베이스 모델에게 이어 쓰게 했을 때, 모델은 가능한 과거 패턴을 섞어 여러 가짜 이어 쓰기를 만들어냅니다 (링크). 이것이 베이스 모델을 이해할 때 가장 조심해야 할 점입니다.

의생명과학 학생에게 베이스 모델의 의미는 이렇게 정리할 수 있습니다. 모델은 많은 텍스트를 읽고, 그 텍스트의 패턴을 매개변수 안에 압축해두었습니다. 그래서 우리에게 배경 설명을 주고, 논문 문체를 흉내 내고, 코드 초안을 만들고, 낯선 개념을 여러 수준으로 풀어줄 수 있습니다. 그러나 그 지식은 도서관 서가의 책처럼 제목과 위치가 정확히 붙어 보관된 것이 아닙니다. 오래전에 읽은 논문을 흐릿하게 기억하는 사람의 머릿속에 더 가깝습니다. 그러므로 베이스 모델에서 어시스턴트(assistant)로 넘어가는 다음 단계가 필요합니다. 우리는 인터넷 문서를 이어 쓰는 모델이 아니라, 질문을 이해하고, 적절히 답하고, 모르면 조심하며, 필요한 경우 도구를 사용하는 어시스턴트를 원합니다. 그 전환이 바로 다음 장의 이야기입니다.

데이터의 그림자와 규모의 부담

데이터를 모으고 거르는 일에는 기술만이 아니라 판단도 들어갑니다. 어떤 언어의 문서가 많이 들어가는지, 어떤 분야의 글이 적게 들어가는지, 어떤 커뮤니티의 표현이 반복되는지에 따라 모델이 쉽게 말하는 세계가 달라집니다. 영어 논문과 개발자 문서는 인터넷에 풍부하므로 모델이 비교적 잘 다룹니다. 반대로 한국어 학부 강의노트, 지역 의료 현장의 기록, 공개되지 않은 실험실 프로토콜, 최신 프리프린트(preprint)의 세부 논쟁은 상대적으로 덜 들어갔을 수 있습니다. 생물학에서도 많이 연구된 유전자와 질병은 설명이 풍부하지만, 연구가 적은 현상은 문헌 자체가 빈약합니다. 모델은 세계를 직접 경험한 것이 아니라 텍스트를 통해 배웠으므로, 텍스트가 많은 곳과 적은 곳의 불균형을 그대로 안고 갑니다. 따라서 베이스 모델의 유창함을 볼 때 우리는 데이터의 그림자도 함께 보아야 합니다. 모델이 잘 말하는 영역은 실제로 더 참인 영역이 아니라, 더 많이 쓰이고 더 많이 반복된 영역일 수 있습니다. **많이 말해지는 것과 잘 검증된 것은 다를 수 있습니다.**

사전학습의 규모는 개인이 쉽게 재현할 수 있는 수준을 넘어섭니다. 카파시는 GPT-2와 현대 모델의 차이를 설명하면서, 매개변수 수, 문맥 길이, 훈련 토큰 수, 계산 비용이 어떻게 커졌는지 보여줍니다 (링크). 이런 모델은 한 연구실의 노트북 몇 대로 처음부터 만들기 어렵습니다. 개인 노트북을 며칠 켜둔다고 끝나는 일이 아니라, 수많은 계산 장치를 오랫동안 안정적으로 돌리고, 중간에 장애가 나도 훈련이 이어지도록 관리해야 하는 일입니다. 대규모 그래픽 처리 장치 묶음(GPU cluster), 긴 시간, 잘 관리된 데이터 파이프라인, 훈련 중 장애를 견디는 시스템 운영(engineering)이 필요합니다. 이 단어들도 지금 모두 외우려 하지 않아도 됩니다. 학생이 LLM을 공부한다는 것은 거대한 모델을 직접 사전학습한다는 뜻이 아닙니다. 이미 만들어진 기반 모델(foundation model)의 작동 원리를 이해하고, 그 모델을 어떤 문제에 어떻게 안전하게 적용할지 배우는 것이 더 현실적입니다. 모든 학생이 현미경 렌즈를 직접 만들지는 않지만, 초점이 맞지 않은 사진을 조심해야 한다는 사실은 알아야 합니다. LLM도 마찬가지입니다. 모델을 직접 훈련하지 않더라도, 사전학습이 어떤 데이터를 어떤 목표로 압축하는 과정인지 알아야 답변을 제대로 해석할 수 있습니다.

> **작은 실습**
>
> 온라인 tokenizer 도구에 세 문장을 넣어보십시오. “cell differentiation was inhibited”, “세포분화가 억제되었다”, “TP53 변이가 관찰되었다.” 공백 하나를 넣거나 빼고, 유전자 이름을 다른 이름으로 바꾸어보면 토큰 수와 조각이 달라질 수 있습니다. 결과를 보며 “사람이 보는 단어”와 “모델이 보는 조각”이 어디서 달라지는지 표시해보는 것만으로도, 토큰화가 추상적인 용어가 아니라 실제 사용의 조건이라는 점이 분명해집니다.

용어 메모

그래픽 처리 장치 묶음(GPU cluster): 큰 계산을 빠르게 하기 위해 여러 그래픽 처리 장치를 묶어 둔 컴퓨터 묶음입니다.

문맥 길이: 모델이 한 번에 읽을 수 있는 토큰의 길이입니다.

훈련 토큰: 훈련에 실제로 사용된 토큰입니다.

기반 모델(foundation model): 많은 자료를 먼저 학습해 여러 과제의 출발점으로 쓰는 큰 모델입니다.

sequencing: DNA나 RNA의 순서를 읽어 데이터로 만드는 실험 기술입니다.

시스템 운영(engineering): 큰 시스템이 오래 안정적으로 돌아가게 만드는 설계와 운영 작업입니다.

이 생각은 생물학의 AI 모델로도 자연스럽게 이어집니다. 언어 모델이 인터넷 텍스트에서 반복되는 패턴을 배운다면, 생물학 모델은 단백질 서열, 유전체 서열, 현미경 이미지, 여러 실험에서 나온 큰 표에서 반복되는 패턴을 배울 수 있습니다. 여기서도 원리는 비슷합니다. 많은 데이터를 모으고, 품질을 관리하고, 모델이 풀 수 있는 연습문제를 만들고, 그 연습을 반복하게 하며 표현을 학습합니다. 그러면 모델은 특정 실험 하나의 답만 배우는 것이 아니라, 여러 조건에서 자주 나타나는 일반적인 패턴을 포착할 수 있습니다. 물론 텍스트와 생물학 데이터는 같지 않습니다. 생물학 데이터는 실험 조건, 장비, 샘플 상태, 사람의 해석에 강하게 묶여 있습니다. 그래도 "많은 데이터를 보고 일반적인 표현을 배운다"는 흐름은 공통적입니다. 그래서 LLM을 이해하는 일은 앞으로 의생명 데이터 과학을 이해하는 데도 도움이 됩니다. 학생은 언어 모델을 배울 때, 동시에 현대 생명과학이 데이터 기반 모델로 이동하는 큰 흐름을 함께 보게 됩니다.

좋은 데이터에서 좋은 모델로

베이스 모델은 때때로 너무 사람처럼 보입니다. 시를 쓰고, 논문 문체를 흉내 내고, 코드 오류를 고치고, 어려운 개념을 초등학생에게 설명하듯 풀어냅니다. 그래서 우리는 모델 안에 지식의 도서관이 정리되어 있다고 상상하기 쉽습니다. 하지만 손실 압축이라는 관점은 이 상상을 바로잡아줍니다. 모델은 문장과 사실과 양식을 매개변수에 통째로 저장한 것이 아니라, 다음 토큰을 잘 예측하도록 수많은 수치를 조정한 결과입니다. 이 과정에서 어떤 능력은 놀랍게 나타납니다. 문법을 배운 적 없는 것처럼 보이지만 문법을 지키고, 코드를 직접 실행하지 않았어도 코드 패턴을 말하고, 낯선 질문에 대해 관련 개념을 연결합니다. 그러나 이것은 "이해"라는 말을 어디까지 쓸 수 있는지 조심스럽게 묻게 만듭니다. 모델이 어떤 문장을 이어갈 수 있다는 사실과, 그 문장이 가리키는 실험적 세계를 경험했다는 사실은 다릅니다. 생물학자는 이 차이에 민감해야 합니다. 텍스트로 배운 세계와 실험으로 확인한 세계 사이에는 늘 간격이 있습니다.

결국 사전학습은 LLM의 첫 번째 탄생이라고 할 수 있습니다. 인터넷의 거대한 말뭉치가 토큰으로 바뀌고, Transformer가 그 흐름을 예측하고, 매개변수가 조금씩 조정되며, 베이스 모델이라는 이상한 존재가 만들어집니다. 이 존재는 아직 우리에게 친절하지 않을 수 있지만, 이미 수많은 언어와 지식의 흔적을 품고 있습니다. 다음 단계의 후속훈련(post-training)은 이 존재를 대화 가능한 어시스턴트로 길들이는 과정입니다. 그러나 후속훈련을 이해하려면 먼저 베이스 모델의 성격을 알아야 합니다. 어시스턴트의 말투가 아무리 공손해도, 그 안쪽에는 인터넷 텍스트를 압축한 모델이 남아 있기 때문입니다. 그러므로 우리는 LLM을 볼 때 두 얼굴을 함께 보아야 합니다. 하나는 거대한 문서 세계의 시뮬레이터이고, 다른 하나는 사용자를 돕도록 훈련된 대화 상대입니다. 이 두 얼굴을 구분할 수 있을 때, 모델의 유용함과 위험을 함께 이해할 수 있습니다.

이 두 얼굴은 학생이 실제로 답변을 읽을 때 계속 나타납니다. 모델은 인터넷에서 본 생물학 설명의 문체를 잘 따라 하면서도, 어시스턴트로서 사용자의 질문에 맞추어 친절하게 정리합니다. 그래서 답변은 마치 전문가가 학생을 위해 직접 쓴 설명처럼 보입니다. 그러나 그 안에는 베이스 모델의 흔적이 남아 있습니다. 많이 반복된 지식은 안정적으로 나오고, 드문 지식은 흐릿해지며, 오래된 지식이 최신인 것처럼 나타날 수 있습니다. 후속훈련은 이 문제를 줄여주지만 없애지는 않습니다. 그러므로 좋은 사용자는 어시스턴트의 표면과 베이스 모델의 바닥을 함께 봅니다. 공손한 말투에 안심하지 않고, 그 말투 뒤에 어떤 데이터와 훈련 목표가 있는지 기억합니다. 의생명과학에서 이 태도는 특히 중요합니다. 질병, 변이, 약물, 유전자 기능을 다루는 문장은 자연스럽게 보이는 것만으로 충분하지 않습니다.

베이스 모델을 이해하는 일은 AI에 대한 윤리적 질문으로도 이어집니다. 인터넷 문서를 대규모로 모아 학습한다는 것은, 사람들이 쓴 수많은 글과 지식의 흔적이 모델 안에 들어간다는 뜻입니다. 공개 웹에 있었다고 해서 모든 문서가 같은 의미로 사용될 수 있는지, 저작권과 개인정보와 데이터 편향을 어떻게 다루어야 하는지, 특정 언어와 지역의 지식이 덜 반영되면 어떤 문제가 생기는지는 여전히 중요한 논쟁입니다. 이 책이 기술 교과서라고 해서 이런 질문을 완전히 피할 수는 없습니다. 의생명과학도 마찬가지입니다. 공개 데이터라고 해서 아무 맥락 없이 써도 되는 것은 아니고, 환자 데이터는 더 엄격한 보호가 필요합니다. AI 모델은 데이터로 만들어지기 때문에, 데이터의 출처와 사용 조건을 따지는 일이 필요합니다. 학생은 모델을 쓰는 소비자이면서 동시에 앞으로 데이터를 만들고 공유할 연구자가 될 수 있습니다. 그러므로 사전학습을 단순한 기술 단계로만 보지 말고, 어떤 지식이 어떤 방식으로 모델 안에 들어가는지 묻는 출발점으로 삼아야 합니다.

이 질문은 앞으로 학생이 만드는 작은 데이터셋에도 적용됩니다. 수업 과제로 정리한 논문 표, 연구실에서 만든 실험 노트, 공개 데이터에서 내려받은 메타데이터(metadata)도 언젠가는 다른 분석과 모델의 입력이 될 수 있습니다. 그때 데이터가 지저분하면 다음 사람은 잘못된 결론에 가까워집니다. 열 이름이 모호하고, 단위가 빠져 있고, 제외 기준이 기록되지 않고, 실패한 조건이 사라진 데이터는 겉으로는 표처럼 보이지만 지식으로 쓰기 어렵습니다. LLM의 사전학습을 배우는 일은 그래서 거대한 회사의 GPU 이야기에만 머물지 않습니다. **좋은 모델은 좋은 데이터에서 출발하고, 좋은 데이터는 작은 기록 습관에서 출발합니다.** 학생이 오늘 파일 이름을 분명히 쓰고, 조건을 남기고, 원본을 보존하는 일도 같은 흐름 안에 있습니다. 거대한 베이스 모델과 작은 연구 노트는 멀어 보이지만, 둘 다 "무엇을 남기고 무엇을 버릴 것인가"라는 질문을 공유합니다. 데이터 과학의 윤리는 바로 그 질문에서 시작됩니다.

이 장을 읽은 뒤 학생이 기억할 것은 모델을 처음부터 만드는 기술적 세부사항이 전부가 아닙니다. 더 중요한 것은 LLM이

"어딘가에서 지식을 꺼내 말하는 상자"가 아니라, 많은 텍스트를 보고 다음 조각을 맞히도록 훈련된 압축 시스템이라는 점입니다. 이 점을 알면 모델의 답변을 더 현실적으로 읽게 됩니다. 유명한 사실을 잘 말한다고 해서 모든 드문 사실도 정확히 말할 것이라고 기대하지 않게 됩니다. 영어 웹에 많은 자료가 있다고 해서 한국어 수업 맥락까지 똑같이 잘 이해할 것이라고 생각하지 않게 됩니다. 생물학에서도 마찬가지입니다. 많이 측정된 현상과 아직 잘 측정되지 않은 현상 사이에는 모델의 언어가 다르게 흔들릴 수 있습니다. 결국 학생에게 필요한 태도는 크고 멋진 모델을 무조건 믿는 것이 아니라, 그 모델이 어떤 자료를 통해 어떤 목표로 배웠는지 묻는 것입니다. 그 질문이 있어야 다음 장에서 어시스턴트의 친절한 말투를 만날 때도, 그 말투 뒤에 있는 베이스 모델의 바닥을 잊지 않을 수 있습니다.

7장. 대화 데이터로 어시스턴트가 된다

많이 읽은 모델은 아직 조교가 아니다

베이스 모델(base model)은 인터넷 문서를 이어 쓰는 데 익숙합니다. 그래서 그대로 두면 우리가 기대하는 어시스턴트처럼 행동하지 않습니다. "이 개념을 설명해줘"라고 물었을 때 친절하게 답하고, "이 코드를 고쳐줘"라고 하면 문제를 짚어주고, 위험한 요청에는 조심스럽게 거절하는 태도는 저절로 생기는 것이 아닙니다. 카파시는 이 지점을 분명히 나눕니다. 사전학습(pre-training)으로 만들어진 베이스 모델은 많은 지식을 품은 토큰 시뮬레이터입니다. 우리가 매일 만나는 ChatGPT 같은 도구는 그 위에 후속훈련(post-training)이 얹힌 어시스턴트입니다 (링크). 후속훈련은 모델을 조금 더 똑똑하게 만드는 장식이 아니라, 모델의 행동 양식과 말투와 거절 방식까지 바꾸는 과정입니다. 모를 때 어떻게 말할지, 어디까지 도울지, 어떤 요청 앞에서 멈출지도 여기에서 배웁니다. 사람으로 비유하면, 아주 많은 책을 읽은 사람이 아직 좋은 조교가 된 것은 아니라는 뜻입니다. 책을 많이 읽은 사람도 학생의 질문에 어떻게 답해야 하는지, 어디까지 도와줘야 하는지, 어떤 말은 조심해야 하는지 따로 배워야 합니다.

용어 메모

베이스 모델(base model): 많은 글을 읽고 이어 쓰는 능력을 먼저 배운 모델입니다.

어시스턴트(assistant): 사용자의 질문에 도움 되는 답을 하도록 추가 훈련된 모델의 사용 형태입니다.

후속훈련(post-training): 베이스 모델을 대화와 도움의 형식에 맞게 다시 훈련하는 단계입니다.

토큰 시뮬레이터: 다음 토큰을 이어가며 글의 흐름을 흉내 내는 모델이라는 뜻입니다.

이때 사용하는 핵심 재료가 대화 데이터입니다. 인터넷 문서 대신, 사람과 어시스턴트가 주고받는 대화 예시를 많이 만듭니다. 사용자가 "2 더하기 2는?"이라고 묻고 어시스턴트가 "4입니다"라고 답하는 예시가 있을 수 있습니다. 사용자가 앞 질문을 바꾸어 "곱하기라면?"이라고 물으면 어시스턴트가 이전 문맥을 이어 받아 답하는 예시도 있을 수 있습니다. 어떤 요청에는 도와주지 않는 답변이 들어갑니다. 답변은 도움이 되고(helpful), 진실하며(truthful), 해롭지 않아야(harmless) 한다는 기준을 따라야 합니다. 이 말들은 짧지만, 실제로는 아주 많은 판단을 포함합니다. 도움이 된다는 것은 사용자의 의도를 이해한다는 뜻이고, 진실하다는 것은 근거 없는 말을 꾸며내지 않는다는 뜻이며, 해롭지 않다는 것은 위험한 행동을 부추기지 않는다는 뜻입니다. 이 기준을 데이터로 만들면 모델은 그 대화의 통계를 배웁니다.

예시로 행동을 가르치기

카파시는 이 과정을 "예시로 프로그래밍하기(programming by example)"로 설명합니다 (링크). 우리가 전통적인 프로그램을 만들 때는 규칙을 코드로 씁니다. "이런 입력이 오면 이렇게 처리하라"는 절차를 명시합니다. 그러나 LLM 어시스턴트를 만들 때는 수많은 예시 대화를 통해 행동을 가르칩니다. 사람 라벨러(labeler)가 프롬프트(prompt)를 만들고, 이상적인 어시스턴트 답변을 씁니다. 이 라벨러들은 아무렇게나 답하지 않습니다. 회사가 만든 긴 라벨링 지침(labeling instruction)을 읽고, 어떤 답변이 좋은지에 대한 기준을 배운 뒤 예시를 작성합니다. 이런 지침에는 보통 "사용자의 질문에 직접 답하라", "근거가 없으면 추측을 사실처럼 말하지 말라", "위험한 행동을 구체적으로 돕지 말라", "여러 답변이 가능하면 더 도움이 되는 구조를 고르라" 같은 항목이 들어갑니다. 실제 공개 논문과 기술 보고서에서도 이런 사람 평가와 지침이 후속훈련의 중요한 재료였음을 볼 수 있습니다. 그러면 베이스 모델은 인터넷 문서의 통계에서 조금 벗어나, "사람이 질문하면 어시스턴트는 이런 식으로 답한다"는 패턴을 배우게 됩니다. 오늘날에는 다른 LLM이 이런 데이터 생성에 많이 참여하고, 사람은 감수하거나 수정하거나 기준을 잡는 역할을 하기도 합니다. **어시스턴트의 친절함과 조심성은 하늘에서 떨어진 성격이 아니라, 데이터와 기준을 통해 훈련된 행동입니다.**

용어 메모

예시로 프로그래밍하기(programming by example): 규칙을 하나하나 쓰기보다 좋은 예시를 많이 보여주어 행동을 배우게 하는 방식입니다.

프롬프트(prompt): 사용자가 모델에게 주는 질문이나 지시문입니다.

라벨러(labeler): 모델 훈련에 쓸 좋은 질문과 답변 예시를 만드는 사람입니다.

라벨링 지침(labeling instruction): 라벨러가 어떤 답을 좋은 답으로 볼지 참고하는 자세한 기준 문서입니다.

대화도 토큰의 줄이 된다

대화도 모델 안으로 들어갈 때는 결국 토큰열이 됩니다. 사람에게는 화면에 말풍선처럼 보이는 사용자 질문과 어시스턴트 답변도, 서버 안에서는 특수 토큰과 텍스트 토큰이 섞인 1차원의 긴 줄로 바뀝니다. "여기서부터 사용자의 말이다", "여기서부터 어시스턴트의 말이다", "이 대화는 여기서 끝난다" 같은 표지가 토큰으로 들어갑니다. 카파시는 대화 규약(conversation protocol)을 보여주며, 우리가 구조화된 대화라고 느끼는 것이 실제로는 모델이 읽을 수 있는 토큰열로 부호화(encoding)된다고 설명합니다 (링크). 이 사실은 조금 건조하지만 중요합니다. 어시스턴트가 대화의 역할을 이해하는 것은 어떤 영혼이 생겼기 때문이 아니라, 대화 구조가 반복적으로 토큰열 안에 들어가고, 모델이 그 패턴을 배웠기 때문입니다. 시스템 메시지도 같은 원리입니다. 사용자에게 보이지 않는 지시가 문맥 맨 앞에 들어가면, 모델은 그것을 보고 자신의 말투와 경계를 조정합니다. 그래서 "너는 친절한 생물학 튜터다" 같은 지시가 답변의 분위기를 바꿀 수 있습니다.

용어 메모

특수 토큰: 대화의 시작, 사용자 말, 어시스턴트 말처럼 역할을 표시하는 특별한 토큰입니다.

대화 규약(conversation protocol): 대화를 모델이 읽을 수 있는 일정한 형식으로 바꾸는 약속입니다.

시스템 메시지(system message): 사용자에게 보이지 않지만 모델의 말투와 규칙을 정하는 앞쪽 지시문입니다.

이 설명은 ChatGPT의 답변을 조금 다르게 보게 만듭니다. 카파시는 후속훈련된 어시스턴트의 답변을, 라벨링 지침을 배운 인간 라벨러가 시간을 들여 썼을 법한 답변의 신경망적 시뮬레이션으로 생각할 수 있다고 말합니다 (링크). 물론 실제로 매번 사람이 뒤에서 답을 쓰는 것은 아닙니다. 우리는 몇 초 만에 답을 받습니다. 하지만 통계적으로 보면 모델은 사전학습에서 얻은 배경지식과 후속훈련에서 배운 대화 양식을 결합해, 이상적인 답변 예시에 가까운 문장을 생성합니다. 이 관점은 LLM을 과소평가하지도 과대평가하지도 않게 해줍니다. 모델은 단순한 검색기가 아니며, 아주 많은 지식과 패턴을 압축해둔 신경망입니다. 동시에 모델은 전지전능한 연구자가 아니라, 특정 방식으로 만들어진 어시스턴트 행동을 빠르게 시뮬레이션하는 시스템입니다. 그래서 답변이 훌륭해 보여도 "이 답은 어떤 종류의 예시와 기준을 닮은 것일까"라고 묻는 태도가 필요합니다.

이 질문은 1학년 학생에게도 실용적입니다. 모델의 답변이 친절하게 보일 때, 우리는 쉽게 "이 설명은 나를 위해 생각해서 쓴 것"이라고 느낍니다. 어느 정도는 맞습니다. 지금 입력된 질문에 맞추어 생성되었기 때문입니다. 그러나 동시에 그 친절함은 많은 예시 답변을 닮은 결과이기도 합니다. 그래서 모델은 때때로 너무 일반적인 조언을 하거나, 실제 자료보다 보기 좋은 구조를 먼저 만들 수 있습니다. 학생이 "이 답이 내 자료를 실제로 읽은 결과인가, 아니면 좋은 답변의 형식을 흉내 낸 것인가"를 물어보면 사용 방식이 달라집니다. 논문 초록을 넣었을 때는 근거 문장을 표시하게 하고, 표를 넣었을 때는 어떤 열을 사용했는지 묻게 되며, 모르는 내용을 설명받을 때는 "내가 이해했는지 확인할 질문을 해달라"고 말할 수 있습니다. 어시스턴트의 말투를 이해하면, 그 말투에 끌려가기보다 그 말투를 공부에 맞게 사용할 수 있습니다.

친절함을 믿기 전에

의생명과학 학생에게 지도 미세조정(SFT)의 의미는 실용적입니다. ChatGPT가 논문 초록을 요약해줄 때, 그것은 논문을 이해한 연구자가 직접 쓴 리뷰가 아닙니다. 많은 대화 예시와 논문 요약 패턴을 배운 모델이, 지금 입력된 초록을 바탕으로 어시스턴트다운 요약을 생성하는 것입니다. 이 도구는 초보자가 논문에 들어가는 문턱을 낮춰줍니다. 낯선 단어를 풀어주고, 연구 질문과 방법과 결과를 나누어주고, 코드 오류를 설명해줄 수 있습니다. 그러나 학생이 해야 할 일까지 사라지는 것은 아닙니다. **좋은 어시스턴트 답변은 공부의 끝이 아니라 시작입니다.** 모델이 나눈 "연구 질문, 방법, 결과, 한계"가 실제 논문에 맞는지 다시 읽어야 합니다. 모델이 안전하게 거절하거나 조심스럽게 말하는 이유도 이해해야 합니다. 어시스턴트가 된다는 것은 모델이 더 인간처럼 보인다는 뜻이지만, 그 인간다움은 훈련된 대화 양식입니다. 바로 그 사실을 알 때 우리는 모델을 더 잘 활용할 수 있습니다.

어시스턴트의 말투가 만들어지는 과정에는 사회적 선택도 들어갑니다. 어떤 답변을 친절하다고 볼 것인지, 어느 정도의 확신을 허용할 것인지, 어떤 주제에서 조심해야 할 것인지, 어떤 요청을 거절해야 할 것인지는 모두 사람이 정한 기준에

영향을 받습니다. 라벨링 지침은 단순한 기술 문서가 아니라, 모델이 사용자와 어떤 관계를 맺어야 하는지에 대한 규범을 담습니다. 카파시가 InstructGPT와 라벨러의 역할을 설명할 때 보여주는 핵심도 여기에 있습니다 (링크). 모델은 인터넷을 읽으며 얻은 수많은 말투 중에서, 어시스턴트에게 어울리는 말투를 따로 배우게 됩니다. 그래서 ChatGPT의 답변은 "중립적인 기계의 목소리"라기보다, 특정 서비스가 좋은 답변이라고 판단한 예시들의 평균적인 목소리에 가깝습니다. 이 말은 불편하게 들릴 수 있지만, 오히려 우리에게 중요한 자유를 줍니다. 우리는 어시스턴트의 답변을 절대적 권위로 받아들이지 않고, 어떤 기준과 훈련을 거쳐 나온 말인지 물을 수 있습니다. 특히 대학 공부에서는 이 태도가 필요합니다. 공손한 문장이라고 해서 늘 깊은 이해를 담고 있는 것은 아니며, 조심스러운 거절이라고 해서 늘 최선의 학문적 판단인 것도 아니기 때문입니다.

대화 데이터가 모델을 바꾼다는 사실은 교육적으로도 큰 의미가 있습니다. 좋은 조교는 정답만 말하지 않습니다. 학생이 어디에서 막혔는지 보고, 너무 앞서가지 않고, 때로는 질문으로 되돌려주고, 학생 스스로 생각할 수 있는 발판을 놓습니다. LLM 어시스턴트도 이런 역할을 하도록 요청할 수 있습니다. "답만 알려줘"라고 말하면 모델은 답을 줍니다. 그러나 "내가 1학년 학생이라고 생각하고, 먼저 내가 이해한 부분을 점검한 뒤, 필요한 개념을 한 단계씩 설명해줘"라고 말하면 전혀 다른 대화가 열립니다. 후속훈련은 모델에게 어시스턴트다운 기본 행동을 가르치지만, 사용자의 프롬프트는 그 행동을 어느 방향으로 꺼낼지 정합니다. 학생은 AI를 정답 자판기로 쓰기보다, 자기 생각을 드러내고 되돌려받는 튜터로 쓸 수 있습니다. 이때 중요한 것은 부끄러워하지 않는 것입니다. 사람에게 묻기에는 너무 기초적인 질문도 모델에게는 여러 번 물을 수 있습니다. 다만 모델이 답한 내용을 수업 자료, 교과서, 논문과 다시 맞춰보는 과정은 남겨두어야 합니다. AI 튜터는 문턱을 낮춰주지만, 학문적 책임을 대신 져주지는 않습니다.

의생명과학에서는 이 어시스턴트의 성격을 더 섬세하게 써야 합니다. 예를 들어 논문을 읽을 때 모델에게 "이 논문을 요약해줘"라고만 묻는 것은 편하지만, 좋은 공부가 되지 않을 수 있습니다. 먼저 "이 논문이 해결하려는 생물학적 질문을 한 문단으로 풀어줘"라고 묻고, 다음에는 "사용한 데이터와 실험 방법을 초보자가 이해할 수 있게 설명해줘"라고 묻고, 이어서 "저자들의 결론이 데이터에서 직접 나온 것인지, 해석이 더 들어간 것인지 구분해줘"라고 물을 수 있습니다. 이렇게 대화를 나누면 어시스턴트는 단순 요약기가 아니라 읽기 동반자가 됩니다. 모델이 틀릴 수 있다는 사실은 여전히 중요하지만, 틀릴 수 있기 때문에 아예 쓰지 말자는 결론으로 갈 필요는 없습니다. 오히려 틀릴 수 있음을 아는 상태에서, 질문을 잘게 나누고, 원문으로 확인하고, 자신의 이해를 다시 쓰는 방식으로 사용하면 됩니다. 좋은 어시스턴트 사용법은 완벽한 답을 받는 기술이 아니라, 더 나은 학습 경로를 만드는 기술입니다. 이것은 대학 1학년 학생에게 특히 중요합니다. 처음부터 모든 논문을 혼자 읽을 수는 없지만, AI와 함께 읽는 법을 배우면 낯선 지식의 문턱을 훨씬 낮출 수 있습니다.

어시스턴트가 된 모델을 볼 때 또 하나 기억해야 할 점은, 친절함이 정확성을 보장하지 않는다는 사실입니다. 후속훈련은 모델이 사용자의 의도에 잘 맞추도록 돕지만, 모델의 매개변수 안에 없는 사실을 새로 만들어 넣지는 않습니다. 모르는 내용을 모른다고 말하게 훈련할 수는 있지만, 모든 불확실성을 완벽하게 감지하게 만드는 것은 어렵습니다. 그래서 어시스턴트는 때때로 매우 공손하게 틀립니다. "좋은 질문입니다"라고 시작하고, "일반적으로"라는 말로 부드럽게 이어가며, 마지막에는 자신감 있는 결론을 제시할 수 있습니다. 문장은 친절하지만 근거는 약할 수 있습니다. 이럴 때 학생이 해야 할 일은 모델을 혼내는 것이 아니라, 사용 조건을 바꾸는 것입니다. 원문을 붙여넣고, 답변의 근거 문장을 표시하게 하고, 불확실한 부분을 따로 말하게 하고, 필요한 경우 검색이나 코드 실행을 쓰게 해야 합니다. 어시스턴트의 장점은 대화할 수 있다는 데 있습니다. 첫 답변이 끝이 아니라, 그 답변을 다시 점검하도록 요구할 수 있다는 점이 중요합니다.

결국 후속훈련은 LLM을 우리 곁에 앉힐 수 있게 만드는 과정입니다. 베이스 모델은 인터넷 문서의 흐름을 이어 쓰는 존재에 가까웠지만, 어시스턴트는 사용자의 질문을 대화의 형식으로 받아들이고, 적절한 어조로 응답하고, 때로는 거절하고, 때로는 설명을 단계화합니다. 이 변화 덕분에 LLM은 전문가만 다루는 연구 도구가 아니라, 학부 신입생도 사용할 수 있는 공부 도구가 되었습니다. 그러나 그 친근함 때문에 더 조심해야 합니다. 도구가 사람처럼 말할수록, 우리는 그것이 사람처럼 이해한다고 믿기 쉽습니다. 카파시가 보여주는 후속훈련의 이야기는 바로 이 착각을 덜어줍니다. 모델은 대화의 예시를 통해 어시스턴트다운 행동을 배웠고, 우리는 그 행동을 이용해 공부할 수 있습니다. 다만 **공부의 마지막 문장은 여전히 학생이 써야 합니다.** AI가 만들어준 설명을 읽고, 원문과 대조하고, 자기 언어로 다시 정리하는 순간에 비로소 지식은 자기 것이 됩니다.

AI 튜터와 자기 문장

이 지점에서 학생은 어시스턴트를 대하는 자신의 태도도 점검해볼 수 있습니다. 어떤 사람은 모델이 너무 사람처럼 말하기 때문에 과하게 신뢰합니다. 또 어떤 사람은 모델이 결국 통계적 생성기라는 말을 듣고 과하게 무시합니다. 둘 다 공부에는 도움이 되지 않습니다. 어시스턴트는 분명히 훈련된 시뮬레이션이지만, 그 시뮬레이션은 실제로 학습과 연구를 도울 만큼 강력합니다. 중요한 것은 "진짜 사람인가"라는 질문에 매달리는 것이 아니라, "어떤 일을 어느 조건에서 맡길 수 있는가"를 묻는 것입니다. 개념의 첫 설명, 글의 구조 잡기, 코드 오류의 원인 추정, 논문 읽기의 발판 만들기에는 매우 유용할 수 있습니다. 반대로 최신 사실의 최종 확인, 임상적 판단, 데이터의 정확한 계산, 연구 윤리와 관련된 결정은 반드시 별도의

확인을 거쳐야 합니다. 이런 구분은 처음에는 번거롭지만, 곧 자연스럽게 익힐 수 있습니다. 좋은 조교에게도 모든 일을 맡기지 않고, 맡길 일과 직접 확인할 일을 나누는 것과 같습니다.

어시스턴트의 등장은 교수자와 학생의 관계도 조금 바꿉니다. 예전에는 학생이 기초적인 질문을 하기 어려워하는 경우가 많았습니다. 너무 쉬운 질문처럼 보일까 봐, 수업 시간에 흐름을 끊을까 봐, 이미 배운 내용을 다시 묻는 것이 부끄러워서 넘어가곤 했습니다. LLM은 그런 질문을 받아줄 수 있습니다. 같은 설명을 세 번, 네 번 다른 비유로 다시 들을 수도 있고, 영어와 한국어를 오가며 물을 수도 있고, 내가 이해한 내용을 검사받을 수도 있습니다. 그러나 이 편리함이 수업과 교수자의 역할을 없애지는 않습니다. 오히려 수업에서는 더 깊은 질문을 나누고, 모델이 놓친 맥락을 짚고, 학생이 만든 이해를 함께 검토하는 시간이 중요해질 수 있습니다. AI가 낮은 문턱을 맡아주면, 사람은 더 높은 대화로 갈 수 있습니다. 대학 공부는 답을 받는 시간이 아니라, 자기 질문을 점점 더 나은 질문으로 바꾸는 시간이 되어야 합니다.

따라서 어시스턴트를 잘 쓰는 학생은 대화의 마지막에 자기 문장을 남깁니다. 모델이 설명한 내용을 그대로 저장하지 않고, "내가 이해한 것은 이렇다"고 다시 적어봅니다. 그 문장을 모델에게 보여주고 틀린 부분을 찾아달라고 할 수도 있고, 친구나 교수자에게 설명해볼 수도 있습니다. 이 과정은 느리지만 결정적입니다. AI가 만든 설명을 읽는 순간에는 이해한 것처럼 느껴지지만, 자기 말로 다시 쓰려고 하면 빈칸이 드러납니다. 그 빈칸이 공부할 자리입니다. 어시스턴트는 그 빈칸을 부끄럽지 않게 발견하게 해줍니다. 그리고 학생은 그 빈칸을 채우면서 모델의 답을 자기 지식으로 바꿉니다. 결국 후속훈련된 어시스턴트의 가치는 답을 대신 써주는 데서 끝나지 않습니다. 학생이 자신의 이해를 더 자주, 더 안전하게 시험해볼 수 있게 하는 데 있습니다.

이때 자기 문장은 길 필요가 없습니다. 처음에는 세 문장도 충분합니다. "이 논문은 무엇을 물었다. 저자들은 어떤 방법으로 답하려 했다. 내가 아직 헷갈리는 부분은 이것이다." 이렇게 적어두면 AI가 만든 긴 설명보다 훨씬 좋은 공부 흔적이 됩니다. 모델은 그 문장을 보고 더 구체적으로 도와줄 수 있고, 학생은 자신이 어디까지 이해했는지 볼 수 있습니다. **어시스턴트와의 좋은 대화는 모델의 마지막 답변으로 끝나지 않습니다. 학생이 남긴 마지막 문장에서 비로소 공부가 자기 쪽으로 돌아옵니다.**

8장. LLM의 기억, 착각, 환각

기억처럼 보이는 압축

LLM의 지식은 데이터베이스의 지식과 다릅니다. 데이터베이스는 어떤 항목을 정해진 자리에 저장하고, 필요할 때 그 값을 꺼냅니다. 유전자 데이터베이스라면 유전자 공식 약어(gene symbol), 염색체(chromosome) 위치, 전사체 식별자(transcript ID), 기능 주석이 각각의 칸에 들어 있습니다. 물론 데이터베이스도 오류가 있을 수 있지만, 적어도 구조는 분명합니다. 반면 모델의 매개변수 안에 들어 있는 지식은 그렇게 정리된 표가 아닙니다. 카파시는 이것을 오래전에 읽은 내용을 어렴풋이 기억하는 것에 비유합니다 (링크). 자주 본 내용은 비교적 안정적으로 떠올리고, 드문 내용은 흐릿하게 떠올립니다. 우리는 사람에게도 비슷한 일을 경험합니다. 유명한 논문이나 자주 쓰는 개념은 바로 설명할 수 있지만, 몇 년 전에 스쳐 지나간 결과는 자신 있게 말하기 어렵습니다. LLM도 많은 경우 그 흐릿한 기억에서 문장을 만듭니다.

용어 메모

유전자 공식 약어(gene symbol): 유전자를 짧게 부르는 공식 표기입니다. TP53, BRCA1 같은 표기가 여기에 해당합니다.

전사체 식별자(transcript ID): 한 유전자에서 만들어지는 RNA 형태를 구분하기 위해 붙인 식별자입니다.

기능 주석: 유전자나 단백질이 어떤 일을 하는지 설명해 둔 기록입니다.

매개변수: 모델 안에 저장된 수많은 숫자입니다. 학습된 흔적이 여기에 남습니다.

자연스럽게 틀리는 문제

문제는 모델이 흐릿함을 항상 흐릿하게 표현하지 않는다는 데 있습니다. 질문을 받았을 때 대화 데이터 안의 어시스턴트들은 대개 자신감 있게 답합니다. 사람이 라벨러(labeler)로서 좋은 답변을 쓸 때도, 모르는 질문을 받으면 검색을 해서 매끄러운 답을 작성합니다. 모델은 그런 답변 양식까지 배웁니다. 그래서 잘 모르는 이름이나 드문 사실을 물어도, "모릅니다"라고 멈추기보다 답변처럼 보이는 문장을 이어갈 수 있습니다. 카파시는 "Orson Kovac"처럼 모델이 알지 못하는 이름을 물었을 때, 오래된 모델이 여러 번 서로 다른 가짜 설명을 만들어내는 장면을 보여줍니다 (링크). 한 번은 작가라고 하고, 다른 번에는 TV 프로그램의 인물이라고 하고, 또 다른 번에는 야구 선수라고 말합니다. 이 답들은 문체만 자연스러울

뿐, 모두 사실이 아닙니다. 우리가 환각(hallucination)이라고 부르는 것은 바로 이런 현상입니다. 모델은 답변의 형식을 배웠지만, 그 형식 안에 들어갈 사실을 충분히 갖고 있지 않을 때도 말을 계속할 수 있습니다.

용어 메모

환각(hallucination): 모델이 사실처럼 보이는 틀린 내용을 만들어내는 현상입니다. 한국어로는 착각, 환각, 지어내기처럼 풀 수 있지만, AI 분야에서는 hallucination이라는 영어도 널리 쓰입니다.

라벨러(labeler): 모델 훈련에 쓸 좋은 답변 예시를 만드는 사람입니다.

의생명과학에서는 이 문제가 특히 위험합니다. 유전자 기능 설명이 논문 초록처럼 보인다고 해서 참인 것은 아닙니다. 질병과 약물의 관계를 그럴듯하게 말한다고 해서 실제 임상 근거가 있는 것도 아닙니다. "이 변이가 병적일 가능성이 높다"는 문장은 환자와 가족에게 큰 의미를 가질 수 있지만, LLM의 매끄러운 문장만으로는 그런 판단을 할 수 없습니다. 희귀질환 이름, 변이 표기, 약물 용량, pathway 해석은 반드시 원자료와 전문 데이터베이스로 돌아가야 합니다. 모델이 자주 본 정보는 잘 말할 수 있지만, 자주 보았다는 사실은 곧 사실이라는 뜻이 아닙니다. 인터넷에 많이 반복된 오류도 함께 학습될 수 있고, 오래된 지식이 최신 지식처럼 나타날 수 있습니다. **따라서 LLM의 답변을 볼 때 첫 번째 질문은 "문장이 자연스러운가"가 아니라 "어디서 확인할 수 있는가"여야 합니다.** 특히 생명과 질병을 다루는 글에서는 그 순서가 바뀌면 안 됩니다.

> **짧은 사례**
>
> 어떤 학생이 "TP53의 특정 변이가 폐선암에서 흔한지 알려줘"라고 물었다고 합시다. TP53은 암 연구에서 자주 등장하는 유전자 이름입니다. 변이 표기는 보통 `c.  > 나 p.  ` 처럼 생겼고, 숫자와 글자 하나가 달라져도 다른 변이를 가리킬 수 있습니다. 모델은 논문 초록 같은 문체로 "이 변이는 폐선암에서 반복적으로 보고되며 예후와 관련될 수 있습니다"라고 답할 수 있습니다. 하지만 실제로는 변이 표기가 틀렸거나, 다른 암종에서 보고된 변이를 모델이 섞었거나, TP53이라는 유명한 이름 때문에 일반적인 암 설명을 가져왔을 수 있습니다. 이런 답은 문장만 보면 전문적으로 보입니다. 그러나 ClinVar 같은 전문 데이터베이스와 논문 원문에서 해당 변이와 암종을 다시 확인하기 전까지는 과학적 주장으로 쓸 수 없습니다. 지금 ClinVar의 사용법을 외울 필요는 없습니다. 학생이 배워야 할 것은 "TP53은 중요하다"는 일반 지식이 아니라, 특정 표기와 특정 맥락을 확인하는 습관입니다.

모른다고 말하게 하기

환각을 줄이려면 모델이 모르는 것을 모른다고 말할 수 있게 해야 합니다. 카파시는 Meta의 Llama 3 계열에서 사용된 접근을 예로 들어, 모델이 어떤 질문을 아는지 모르는지 경험적으로 조사하고, 모르는 질문에 대해서는 "기억하지 못한다"거나 "모른다"고 답하는 예시를 훈련 예시 묶음(training set)에 추가하는 과정을 설명합니다 (링크). 이 설명이 흥미로운 이유는, 모델 안쪽 어딘가에는 불확실성을 나타내는 신호가 있을 수 있지만, 그 신호가 말로 표현되도록 연결되어 있지 않을 수 있다는 점입니다. 사람이 속으로는 자신이 모른다는 느낌을 갖고도 체면 때문에 답을 지어내는 장면과 조금 닮았습니다. 모델에게도 "이럴 때는 모른다고 말해도 된다"는 행동을 데이터로 가르쳐야 합니다. 이것은 작은 변화처럼 보이지만, 실제 사용에서는 매우 큽니다. **모르는 것을 모른다고 말하는 어시스턴트는 연구와 공부에서 훨씬 안전합니다.** 틀린 답을 자신 있게 말하는 어시스턴트보다, 부족함을 표시하는 어시스턴트가 더 믿을 만합니다.

용어 메모

훈련 예시 묶음(training set): 모델을 가르치기 위해 모아 둔 예시 자료입니다.

불확실성: 답이 맞는지 모델이나 사람이 충분히 확신하기 어려운 상태입니다.

눈앞의 자료로 답하게 하기

두 번째 해결책은 도구를 쓰게 하는 것입니다. 사람이 모르는 사실을 물으면 검색을 하듯이, 모델도 자신의 매개변수 안의 흐릿한 기억에만 의존하지 않고 웹 검색이나 데이터베이스 조회를 할 수 있습니다. 카파시는 웹 검색 도구(web search tool)가 어떻게 작동하는지 설명하면서, 모델이 특수 토큰과 정해진 형식으로 검색을 요청하고, 검색 결과의 텍스트가 문맥 창 안으로 들어오면 그 자료를 바탕으로 답변을 생성한다고 말합니다 (링크). 이때 매개변수 안의 지식은 오래된 기억이고, 문맥 창 안의 텍스트는 방금 눈앞에 놓인 자료에 가깝습니다. 사람이 논문을 다시 펴놓고 설명하면 더 정확해지는 것처럼, 모델도 관련 문서를 문맥에 넣어주면 더 안정적으로 답할 수 있습니다. 물론 검색 결과 자체도 평가해야 합니다. 웹에서 찾았다고 해서 모두 믿을 수 있는 것은 아니기 때문입니다. 하지만 적어도 모델이 머릿속에서 지어낸 문장보다, 출처가 있는 자료를 보고 답하게 하는 편이 훨씬 낫습니다.

용어 메모

웹 검색 도구(web search tool): 모델이 웹에서 자료를 찾아 문맥에 가져오도록 돕는 도구입니다.

문맥 창: 모델이 지금 답할 때 눈앞에 놓고 읽을 수 있는 입력 공간입니다.

특수 토큰: 모델에게 "검색을 시작한다" 같은 특별한 역할을 알려주는 표시입니다.

LLM의 자기소개도 조심해서 읽어야 합니다. 모델에게 "너는 어떤 모델이니?"라고 물으면 그 답이 실제 내부 상태의 고백처럼 느껴질 수 있습니다. 그러나 카파시는 그런 질문 자체가 조금 어색하다고 설명합니다. 모델은 사람처럼 지속적인 자아를 가진 존재가 아니라, 대화가 시작될 때마다 문맥과 매개변수를 바탕으로 토큰을 생성하는 시스템입니다. 어떤 모델은 자신이 OpenAI가 만들었다고 잘못 말할 수 있고, 어떤 모델은 시스템 메시지(system message)나 별도의 훈련 데이터(training data)를 통해 자기 이름과 개발자를 말하도록 조정될 수 있습니다 (링크). 그러므로 모델의 자기 설명도 다른 답변과 마찬가지로 생성된 문장입니다. "나는 무엇이다"라는 답변이 나왔다고 해서 그것이 기계의 내면 고백은 아닙니다. LLM을 사람처럼 대화할 수는 있지만, 사람처럼 믿어서는 안 됩니다.

환각을 이해한다는 것은 AI를 두려워하자는 뜻이 아닙니다. 오히려 더 잘 쓰기 위한 조건입니다. 모델은 낯선 개념의 첫 설명을 제공하고, 어려운 문장을 여러 수준으로 풀고, 논문 구조를 잡아주는 데 큰 도움을 줍니다. 그러나 그 도움은 언제나 확인과 함께 가야 합니다. 모르는 질문에는 "모른다"고 말하게 하고, 사실 질문에는 출처를 찾게 하고, 계산 문제에는 코드를 쓰게 하고, 의생명 정보에는 원문과 데이터베이스로 돌아가게 해야 합니다. 이 태도를 익히면 LLM은 위험한 자동답변기가 아니라, 공부의 좋은 동반자가 될 수 있습니다. 매끄러운 문장을 의심하는 일은 냉소가 아닙니다. 그것은 과학을 공부하는 사람이 가져야 할 기본적인 정직함입니다.

문맥 창은 이 문제를 완화하는 중요한 장치입니다. 카파시는 매개변수 안의 지식을 오래된 기억에, 문맥 창 안의 텍스트를 지금 눈앞에 펼쳐놓은 자료에 비유합니다 (링크). 사람도 기억만으로 논문을 설명할 때보다, 논문 원문을 옆에 펼쳐놓고 설명할 때 더 정확해집니다. LLM도 비슷합니다. 모델에게 논문 초록, 표, 그림 설명(figure legend), 방법(methods) 일부를 직접 넣어주면, 모델은 그 텍스트를 보고 답할 수 있습니다. 이때 답변의 근거는 모델의 흐릿한 매개변수 기억보다 훨씬 가까운 곳에 있습니다. 그러나 문맥에 넣었다고 해서 자동으로 안전해지는 것은 아닙니다. 모델이 긴 문서의 중요한 부분을 놓칠 수 있고, 서로 모순되는 문장을 잘못 조합할 수 있으며, 사용자가 넣은 자료 자체가 틀렸을 수도 있습니다. 그래도 **문맥을 활용하는 것은 환각을 줄이는 가장 실용적인 방법 중 하나입니다.** 논문을 읽을 때는 "이 논문에 대해 알려줘"라고 묻기보다, 실제 초록과 주요 문단을 넣고 "이 문단 안에서만 근거를 찾아 설명해줘"라고 요청하는 편이 훨씬 낫습니다.

여기서 RAG라는 말을 만날 수 있습니다. 검색 보강 생성(Retrieval-augmented generation)은 모델이 답변을 만들기 전에 관련 문서를 검색해 문맥에 넣고, 그 자료를 바탕으로 답하게 하는 방식입니다. 이 장에서는 환각을 줄이는 방법으로 짧게 맛보고, 다음 장에서 문맥 창과 도구 사용의 관점에서 다시 다룰 것입니다. 원리는 어렵지 않습니다. 사람도 보고서를 쓸 때 기억만으로 쓰지 않고, 관련 논문과 데이터베이스를 찾아 책상 위에 펼쳐놓습니다. LLM도 검색된 문서를 보고 답하면 더 근거 있는 문장을 만들 수 있습니다.

하지만 RAG를 마법처럼 생각해서는 안 됩니다. 어떤 문서를 검색했는지, 검색어가 적절했는지, 검색된 문서가 믿을 만한지, 모델이 그 문서의 어느 부분을 근거로 삼았는지 확인해야 합니다. 특히 의생명 분야에서는 학술 데이터베이스의 초록, 데이터베이스 항목(database entry), 진료·연구 지침(guideline), 종설(review article)의 성격이 서로 다릅니다. 단순 블로그 글과 동료 심사를 거친 논문(peer-reviewed paper)을 같은 무게로 놓으면 안 됩니다. RAG는 모델에게 기억을 보충해주는 기술이지만, 출처를 평가하는 일은 여전히 사람의 몫입니다. 좋은 AI 사용자는 "검색했으니 됐다"에서 멈추지 않고, "무엇을 검색했고 무엇을 근거로 답했는가"를 따집니다.

용어 메모

RAG: 관련 자료를 먼저 찾아 문맥에 넣고, 그 자료를 바탕으로 답하게 하는 방식입니다.

데이터베이스 항목(database entry): 데이터베이스 안의 한 항목입니다. 유전자 하나나 논문 하나의 기록처럼 생각하면 됩니다.

지침(guideline): 전문가 집단이 진료나 연구 판단에 참고하도록 정리한 권고문입니다.

동료 심사 논문(peer-reviewed paper): 동료 연구자의 검토를 거쳐 학술지에 실린 논문입니다.

환각은 단지 모델의 결함만이 아니라, 사용자의 질문 방식과도 연결됩니다. 질문이 너무 넓고, 자료가 없고, 최신 사실을 요구하고, 답변 형식만 강하게 요청하면 모델은 빈 공간을 그럴듯한 문장으로 채우기 쉽습니다. "최근 연구를 정리해줘"라는 말은 편하지만 위험합니다. 최근이 언제까지인지, 어떤 데이터베이스를 볼 것인지, 종설과 원 논문을 어떻게 구분할 것인지, 어떤 종과 조직과 질병 맥락으로 제한할 것인지가 빠져 있기 때문입니다. 반대로 "아래 초록 세

개만 근거로, 공통된 주장과 서로 다른 한계를 구분해줘"라고 물으면 훨씬 안전합니다. 모델에게도 일할 재료와 경계가 필요합니다. 학생이 연구실에서 실험을 배울 때도 마찬가지입니다. "세포를 잘 키워봐"라는 지시는 좋은 실험 지시가 아닙니다. 어떤 배지를 쓸지, 세포를 처음 얼마나 담을지, 몇 번 키운 세포인지, 어떤 온도와 시간 조건에서 배양할지, 언제 관찰할지 같은 조건이 있어야 합니다. AI에게 던지는 질문도 실험 조건처럼 다루면 환각이 줄어듭니다.

검증하는 글쓰기

또 하나 중요한 습관은 모델에게 근거의 위치를 요구하는 것입니다. 단순히 "출처를 달아줘"라고 하면 모델이 존재하지 않는 논문 제목이나 그럴듯한 DOI를 만들어낼 수 있습니다. 더 좋은 방법은 사용자가 제공한 텍스트 안에서 근거 문장을 찾아달라고 하거나, 검색 도구를 사용한 경우 링크와 함께 어떤 문장이 어떤 결론을 뒷받침하는지 설명하게 하는 것입니다. 그래도 마지막 확인은 사람이 해야 합니다. 링크가 실제로 열리는지, 제목과 저자가 맞는지, 논문이 주장하는 내용과 모델의 해석이 일치하는지 확인해야 합니다. 이 과정이 번거롭게 느껴질 수 있습니다. 그러나 **과학에서 근거 확인은 번거로운 절차가 아니라 지식을 만드는 핵심입니다.** LLM은 그 과정을 빠르게 도와줄 수 있지만, 없애지는 않습니다. 오히려 답변이 빨리 나오기 때문에, 근거 확인의 습관이 더 중요해집니다. 빠른 생성은 빠른 검증과 함께 있을 때 공부가 됩니다.

환각을 줄이는 마지막 방법은 모델을 과하게 사람처럼 대하지 않는 것입니다. 대화가 자연스럽기 때문에 우리는 모델이 우리를 이해하고, 자기 생각을 가지고, 기억을 계속 유지한다고 느끼기 쉽습니다. 그러나 많은 경우 모델은 현재 문맥 안에서 다음 토큰을 생성하고 있을 뿐입니다. 물론 서비스에 따라 메모리 기능이나 개인화 기능이 있을 수 있지만, 그것도 별도의 시스템으로 관리되는 정보입니다. 모델 자체가 사람처럼 지난 학기 대화를 마음속에 간직하고 있는 것은 아닙니다. 이 차이를 이해하면 실망도 줄어듭니다. 모델이 어제와 다른 말을 한다고 해서 배신한 것이 아닙니다. 주어진 문맥과 샘플링 방식과 도구 환경이 달라졌을 수 있습니다. 학생은 AI를 친구처럼 편하게 사용할 수 있지만, 과학적 판단에서는 기계로 다루어야 합니다. 편안함과 검증은 함께 갈 수 있습니다. 오히려 그 둘을 함께 붙잡을 때, LLM은 가장 좋은 공부 도구가 됩니다.

환각을 다루는 태도는 학생의 글쓰기에도 영향을 줍니다. AI가 써준 문장을 그대로 가져오면, 그 문장 안에 숨어 있는 불확실성까지 함께 가져오게 됩니다. 문장이 매끄럽기 때문에 읽는 사람은 근거가 탄탄하다고 착각할 수 있습니다. 그래서 AI를 사용해 글을 쓸 때는 먼저 주장과 근거를 분리해야 합니다. "이 문장은 어떤 자료에서 나온 것인가", "이 문장은 자료의 직접 요약인가, 모델의 해석인가", "이 문장은 내가 실제로 이해해서 다시 쓸 수 있는가"를 확인해야 합니다. 특히 한글 교재나 보고서에서는 부드러운 설명이 중요하지만, 부드러움이 근거의 자리를 대신해서는 안 됩니다. 좋은 문장은 친절하면서도 정직해야 합니다. 모르는 부분은 모른다고 남기고, 추측은 추측으로 표시하고, 확인한 내용은 출처와 연결해야 합니다. AI 시대의 글쓰기에서 가장 위험한 것은 틀린 문장이 아니라, 틀렸는지조차 알아차리기 어려운 자연스러운 문장입니다. 그러므로 윤문은 마지막 단계이고, 그 전에 검증이 있어야 합니다.

학생에게 권하고 싶은 가장 단순한 방법은 답변을 세 가지 색으로 읽는 것입니다. 실제로 색연필을 쓰지 않아도 됩니다. 마음속으로라도, 원문에서 확인된 내용, 모델이 합리적으로 해석한 내용, 아직 근거를 찾지 못한 내용을 나누어보면 됩니다. 처음에는 한 문단의 대부분이 회색, 곧 확인되지 않은 영역처럼 보일 수 있습니다. 그것이 나쁜 일은 아닙니다. 오히려 공부가 시작되는 지점입니다. 회색 문장을 원문으로 확인하고, 필요한 경우 검색하고, 그래도 모르면 "불확실하다"고 남기는 과정에서 이해가 깊어집니다. LLM은 회색 영역을 빠르게 만들어줍니다. 사람은 그 회색을 검은 근거와 흰 빈칸으로 나누어야 합니다. 이 작업을 반복하면 환각은 막연한 공포가 아니라 관리해야 할 연구 위험으로 보이기 시작합니다. 과학은 위험을 없애는 일이 아니라, 위험을 알고 통제하는 일에 가깝습니다.

이 색 나누기는 보고서나 발표문을 쓸 때도 도움이 됩니다. AI가 만들어준 설명을 그대로 붙여 넣으면 문장이 매끄럽기 때문에 이미 완성된 것처럼 느껴집니다. 그러나 한 문장씩 따져보면 출처가 분명한 문장, 원문에서 직접 나오지는 않았지만 타당해 보이는 해석, 아직 확인하지 못한 추측이 섞여 있습니다. 학생은 이 세 층위를 구분해야 합니다. 원문에서 나온 내용은 근거와 함께 남기고, 해석은 해석이라고 표시하며, 확인하지 못한 추측은 과감히 지우거나 "가능성"으로 낮추어 써야 합니다. 좋은 과학 글은 자신감 있는 문장만으로 만들어지지 않습니다. 자신이 어디까지 알고 어디서 멈추어야 하는지 아는 문장으로 만들어집니다. LLM을 쓰는 시대에는 이 태도가 더 중요해집니다. 모델이 문장을 쉽게 만들어줄수록, 학생은 더 정직하게 문장의 신분을 확인해야 합니다.

이렇게 보면 환각은 AI 시대에 새로 생긴 완전히 낯선 문제가 아닙니다. 사람도 모르는 것을 아는 것처럼 말할 때가 있고, 기억을 잘못 합치고, 읽지 않은 논문을 읽은 듯이 착각하고, 권위 있는 말투로 틀릴 수 있습니다. 다만 LLM은 그 일을 매우 빠르고 유창하게 할 수 있습니다. 그래서 위험이 커집니다. 그러나 대응 원칙은 과학의 오래된 원칙과 닮았습니다. 근거를 묻고, 재현을 요구하고, 출처를 확인하고, 불확실성을 표시하는 것입니다. AI가 등장했다고 해서 과학의 기준이 사라지는 것이 아니라, 그 기준을 더 자주 적용해야 하는 상황이 온 것입니다. 학생은 이 점을 부담으로만 볼 필요가 없습니다. 오히려 LLM은 검증의 중요성을 매일 연습하게 해주는 도구가 될 수 있습니다. 매끄러운 답변 앞에서 잠시 멈추는 습관, 그 습관이 연구자의 기본 자세를 길러줍니다.

학생이 처음부터 모든 환각을 잡아낼 수는 없습니다. 그것은 교수자나 연구자에게도 쉽지 않은 일입니다. 그래서 **중요한 것은 완벽한 감별 능력보다 좋은 절차입니다.** 원문을 넣고, 근거 문장을 요구하고, 모르는 부분은 모른다고 말하게 하고, 숫자는 코드로 확인하고, 마지막에는 자신의 말로 다시 요약하는 절차입니다. 절차가 있으면 학생은 모델보다 지식이 적어도 위험을 줄일 수 있습니다. 과학은 천재적인 직감만으로 움직이지 않습니다. 누구나 따라 할 수 있는 확인 절차가 있기 때문에 조금씩 단단해집니다. AI와 함께 공부할 때도 같은 원칙이 필요합니다.

9장. 문맥 창과 도구 사용

기억과 책상 사이

오래전에 읽은 논문을 기억만으로 설명해보라고 하면 누구나 조금 불안해집니다. 제목은 떠오르지만, 표본 수가 몇 명이었는지 흐릿합니다. 결론은 기억나지만, 그 결론이 어떤 실험 조건에서 나온 것인지 헷갈립니다. figure의 모양은 대충 떠오르는데, 막상 축 이름을 묻는 순간 말이 막히기도 합니다. 그런데 논문을 다시 펴놓고 설명하라고 하면 사정이 달라집니다. 눈앞에 초록과 그림 설명과 표가 있기 때문입니다.

LLM을 사용할 때도 이 차이가 중요합니다. 모델의 매개변수 안에 들어 있는 지식은 사람의 오래된 기억과 비슷합니다. 엄밀히 말하면 사람의 기억과 같은 방식은 아니지만, 사용자 입장에서는 그렇게 느껴집니다. 무엇인가를 배웠고, 그 배움의 흔적을 바탕으로 답을 만들어냅니다. 반면 문맥 창 안에 넣어준 텍스트는 눈앞에 펼쳐놓은 자료에 가깝습니다. 논문 초록, 그림 설명(figure legend), 실험 조건, 표, 코드, 학술 검색 결과, 강의 노트 일부를 프롬프트(prompt) 안에 넣어주면 모델은 그것을 보면서 답을 만들 수 있습니다.

용어 메모

매개변수: 모델 안에 저장되어 있는 수많은 숫자입니다. 오래된 학습의 흔적처럼 작동합니다.

문맥 창: 모델이 지금 답변을 만들 때 한 번에 읽을 수 있는 입력 공간입니다.

그림 설명(figure legend): 논문 그림 옆이나 아래에 붙는 설명문입니다.

프롬프트(prompt): 사용자가 모델에게 주는 질문, 자료, 지시문을 묶어 부르는 말입니다.

이 차이는 실제 공부에서 매우 큽니다. “이 논문을 요약해줘”라고만 묻는 것과, 논문 초록과 방법 일부와 주요 표를 함께 넣고 “이 자료 안에서만 연구 질문, 실험 설계, 핵심 결과, 한계를 나누어 설명해줘”라고 묻는 것은 전혀 다른 일입니다. 앞의 질문은 모델의 일반 지식과 논문 요약 문체에 기대는 일입니다. 뒤의 질문은 눈앞의 자료를 읽고 정리하게 하는 일에 가깝습니다. 사람도 원문 없이 기억만으로 설명할 때보다, 원문을 펴놓고 설명할 때 훨씬 정확합니다.

문맥 창을 정리하는 일

카파시는 LLM을 설명하면서 문맥이 모델에게 일종의 작업 공간이 된다는 점을 여러 번 강조합니다. 모델은 긴 답을 만들 때도 한 번에 완성된 생각을 꺼내는 것이 아니라, 지금까지 나온 토큰과 입력된 문맥을 보며 다음 토큰을 이어갑니다. 그래서 어떤 자료를 문맥 안에 넣어주느냐가 답의 질을 바꿉니다. 그는 추론 모델(reasoning model)을 설명하면서도 모델이 생각할 시간을 토큰의 형태로 써야 한다는 말을 합니다 (링크). 학생 입장에서는 이 말을 이렇게 받아들여도 좋습니다. 모델에게도 읽을 자료와 메모할 공간과 차근차근 생각할 시간이 필요합니다.

의생명과학에서는 이 차이가 더 중요합니다. 유전자 이름 하나가 비슷하게 생긴 다른 유전자와 헷갈릴 수 있고, 약물 이름 하나가 다른 계열의 약물처럼 보일 수 있습니다. 질병명도 마찬가지입니다. 모델에게 “BRCA에 대해 설명해줘”라고 묻는 것과 “이 논문에서 BRCA1 변이를 어떻게 정의했는지, 이 문단 안의 표현만 근거로 설명해줘”라고 묻는 것은 다릅니다. 앞의 질문은 넓은 설명을 불러오고, 뒤의 질문은 특정 자료 안에서 답을 찾게 합니다.

문맥 창은 그래서 단순히 길이가 긴 입력칸이 아닙니다. 무엇을 믿고 답할 것인지 정하는 자리입니다. 학생이 논문을 읽을 때 형광펜으로 밑줄을 긋고, 여백에 질문을 적고, 중요한 표를 따로 표시하는 것과 비슷합니다. 아무 자료나 많이 넣는다고 좋은 답이 나오는 것은 아닙니다. 관련 없는 자료가 섞이면 모델은 엉뚱한 연결을 만들 수 있습니다. 오래된 리뷰와 최신 논문이 함께 들어가 있으면 어느 쪽을 우선해야 하는지 혼란스러울 수 있습니다. 서로 다른 종, 서로 다른 조직, 서로 다른 실험 조건의 데이터가 한 프롬프트 안에 섞이면, 모델은 차이를 충분히 분리하지 못한 채 매끄러운 설명을 만들 수도 있습니다.

그래서 좋은 사용자는 자료를 넣기 전에 작은 편집자가 됩니다. 지금 답에 필요한 자료가 무엇인지 고릅니다. 논문 전체를 무작정 넣기보다, 연구 질문과 직접 관련된 초록, 방법, 결과, 그림 설명, 표를 나눠 넣습니다. “이 자료만 근거로 답하라”고

말하기도 하고, “확실하지 않은 내용은 추측이라고 표시하라”고 말하기도 합니다. 모델이 답을 만든 뒤에는 원문으로 돌아가 숫자와 문장을 확인합니다. 이 과정은 번거로워 보이지만, 의생명과학에서는 이 번거로움이 바로 안전장치입니다.

도구마다 맡길 일이 다르다

도구 사용은 이 원리의 연장선에 있습니다. 모델에게 모든 것을 머릿속으로 처리하게 하지 않고, 필요한 일을 외부 도구에 맡기는 것입니다. 검색은 최신 논문이나 드문 사실을 문맥 안으로 가져오는 도구입니다. 코드 실행은 숫자 계산과 파일 처리를 정확한 절차로 확인하는 도구입니다. 데이터베이스 조회는 유전자명, 변이 표기, 단백질 기능, pathway 정보를 원자료에 연결하는 도구입니다. LLM이 설명을 잘한다고 해서 표의 행 개수를 머릿속으로 세게 할 필요는 없습니다. 유전자 리스트의 교집합, 결측값 개수, 평균과 표준편차, p-value 계산은 코드로 확인하는 편이 낫습니다.

용어 메모

도구 사용: 모델이 말로만 답하지 않고 검색, 코드 실행, 데이터베이스 조회 같은 외부 기능을 쓰는 일입니다.

pathway: 세포 안에서 여러 분자들이 이어져 신호나 반응을 만드는 길입니다.

p-value: 관찰한 차이가 우연만으로도 나올 수 있는지 따져볼 때 쓰는 통계값입니다.

학생들이 자주 하는 실수는 모델의 언어 능력을 계산 능력으로 착각하는 것입니다. 모델은 표를 보고 “대체로 이런 경향이 있다”고 말할 수 있습니다. 하지만 실제로 몇 행이 있는지, 특정 조건을 만족하는 샘플이 몇 개인지, 중복된 gene symbol이 몇 개인지는 실행으로 확인해야 합니다. “이 파일을 읽고 group별 샘플 수를 코드로 계산해줘. 실행 결과도 보여줘.” 이렇게 묻는 것이 훨씬 낫습니다. 설명은 말로 받을 수 있지만, 계산은 실행으로 받는 습관이 필요합니다.

처음에는 어떤 일을 어떤 도구에 맡겨야 하는지 헷갈릴 수 있습니다. 이때는 일을 네 칸으로 나누어 보면 좋습니다.

하고 싶은 일	더 어울리는 도구	이유
최신 논문이나 드문 사실 찾기	검색, PubMed, 학술 데이터베이스	모델의 기억보다 방금 찾은 자료가 더 가깝습니다.
표의 행 개수, 평균, 교집합 계산	코드 실행	말로 세기보다 실행 결과가 검증하기 쉽습니다.
유전자 이름, 변이 표기, 질병 연결 확인	ClinVar 같은 전문 데이터베이스	표준 이름과 출처가 필요합니다.
어려운 문단을 쉬운 말로 풀기	LLM의 설명 능력	문장을 낮은 계단으로 내려주는 데 강합니다.

이 표의 목적은 도구를 복잡하게 늘리는 데 있지 않습니다. 오히려 반대입니다. 모델에게 모든 일을 한꺼번에 맡기지 말고, 말로 풀 일과 실행으로 확인할 일과 원자료로 돌아갈 일을 나누자는 뜻입니다. 고등학생이나 1학년 학생이 처음부터 모든 데이터베이스를 능숙하게 쓸 필요는 없습니다. 다만 “이 질문은 기억으로 답하게 해도 되는가, 아니면 찾아보고 계산해야 하는가”를 묻기 시작하면 AI 사용의 질이 달라집니다.

RAG와 LLM-Wiki의 차이

이 지점에서 RAG라는 말이 등장합니다. RAG는 검색으로 가져온 자료를 바탕으로 답을 생성하는 방식입니다. 논문, 노트, 데이터 설명서, 프로토콜을 잘게 나누어 저장해두고, 질문이 들어오면 관련 있는 조각을 찾아 문맥에 넣은 뒤 답을 만들게 합니다. 연구자에게는 매우 매력적인 방법입니다. PubMed 논문, 실험 노트, 코드 설명, 데이터베이스 문서를 연결해두면 모델이 아무 근거 없이 답하는 대신, 내가 가진 자료를 바탕으로 답할 수 있기 때문입니다.

용어 메모

RAG: 관련 자료를 검색해 문맥에 넣은 뒤, 그 자료를 보고 답하게 하는 방식입니다.

프로토콜: 실험이나 분석을 어떤 순서와 조건으로 할지 적은 절차서입니다.

그러나 RAG가 모든 것을 해결하지는 않습니다. 검색은 관련 있어 보이는 자료를 찾아올 수 있지만, 그 자료들이 서로 어떤 관계에 있는지까지 자동으로 이해해주지는 않습니다. 어떤 논문은 오래되었지만 여전히 기본 개념을 잘 설명하고, 어떤 논문은 최신이지만 특정 조건에만 맞습니다. 어떤 결과는 내 질문과 직접 관련이 있고, 어떤 결과는 비슷해 보이지만

다른 조직이나 다른 종에서 나온 것입니다. 연구에서는 단순히 정보를 찾는 일보다, 정보들 사이의 관계를 세우는 일이 더 어렵습니다.

이 책에서 LLM-Wiki라고 부르는 접근이 흥미로운 이유도 여기에 있습니다. 여기서 LLM-Wiki는 아직 학생들이 외워야 할 표준 학술 용어가 아닙니다. 이 책에서는 "LLM이 읽고 활용할 수 있도록 개념 노트를 서로 연결해두는 공부 방식"을 가리키는 이름으로 쓰겠습니다. 자료를 그냥 저장하고 검색하는 데서 멈추지 않고, 개념별 문서로 풀고, 서로 연결하고, 점점 하나의 위키처럼 엮어가는 접근입니다. 연구자에게 필요한 것은 검색 시스템만이 아니라, 지식을 만들어가는 시스템에 가깝습니다. 학생이 논문을 읽고 요약한 내용이 흩어진 메모로 끝나지 않고, "이 논문은 무엇을 물었나", "어떤 방법을 썼나", "내가 아직 모르는 용어는 무엇인가" 같은 작은 문서로 이어진다면 그 노트는 다음 질문을 위한 발판이 됩니다.

용어 메모

LLM-Wiki: 이 책에서 쓰는 말로, LLM이 읽고 활용할 수 있도록 개념 문서들을 연결해 지식 구조를 만드는 방식입니다.

이 차이를 공부에 적용해봅시다. 시험 전날에 "세포분화에 대해 알려줘"라고 묻는 것은 검색형 공부에 가깝습니다. 당장의 답을 얻을 수는 있지만, 내 머릿속 구조가 크게 바뀌지는 않을 수 있습니다. 조금 더 익숙한 예로 말하면, 수학 오답노트를 문제 번호 순서로만 쌓아두는 것과, "분수 계산 실수", "조건을 잘못 읽은 문제", "그래프 해석 문제"처럼 원인을 나누어 다시 묶는 것은 다릅니다. 생명과학 노트도 마찬가지입니다. 한 학기 동안 읽은 논문과 강의 내용을 바탕으로 "세포분화", "전사인자", "대조군", "측정값" 같은 작은 문서를 만들고, 각 문서가 서로 어떻게 이어지는지 적어둔다면 이야기가 달라집니다. 그때 AI는 단순히 답을 주는 도구가 아니라, 내가 만든 지식 구조를 다시 비추는 도구가 됩니다.

에이전트에게 맡길 때 남길 것

에이전트는 문맥과 도구 사용이 길게 이어진 형태로 볼 수 있습니다. 한 번 검색하고 끝나는 것이 아니라, 검색하고, 읽고, 코드를 만들고, 실행하고, 실패하면 다시 고치고, 결과를 보고하는 흐름이 생깁니다. 카파시도 에이전트를 긴 시간에 걸쳐 작업을 수행하고, 사람에게 진행 상황을 보고하며, 사람의 감독을 필요로 하는 시스템으로 설명합니다 (링크). 그래서 에이전트와 함께 일할 때는 문맥과 도구가 더 중요해집니다. 무엇을 읽게 할지, 어떤 파일을 건드릴 수 있게 할지, 어떤 결과를 반드시 확인하게 할지 정해야 합니다. 에이전트는 한 번 답하고 사라지는 말풍선이 아니라 여러 단계를 이어가는 작업자에 가깝기 때문에, 초반의 잘못된 가정이 뒤쪽 결과에 계속 누적될 수 있습니다. 처음 파일 이름을 잘못 읽으면 그다음 그래프와 보고서가 모두 그 오류 위에 세워집니다. 처음 논문 초록을 과장해서 해석하면 이후 문헌 정리도 그 과장에 맞춰질 수 있습니다. 그래서 에이전트에게는 더 좋은 문맥이 필요하고, 사람에게는 더 꼼꼼한 중간 확인이 필요합니다.

예를 들어 에이전트에게 수업에서 받은 작은 데이터 파일을 살펴보게 한다고 합시다. 좋은 지시는 "이 데이터를 분석해줘"가 아닙니다. 먼저 어떤 열이 무엇을 뜻하는지 표로 만들게 하고, 샘플 수와 조건을 확인하게 하고, 원본 파일은 수정하지 못하게 하고, 그래프를 만들 때마다 어떤 열을 사용했는지 기록하게 해야 합니다. 결과 그림만 보는 것이 아니라, 중간에 어떤 값이 빠졌는지, 어떤 값을 서로 비교했는지, 그림의 축이 무엇인지 보고하게 해야 합니다. 에이전트가 유능해질수록 이런 지시는 더 중요해집니다.

왜냐하면 에이전트는 작업을 빠르게 진행하면서도 자신이 어디서 미끄러졌는지 모를 수 있기 때문입니다. 파일 이름을 잘못 읽었는데도 그럴듯한 표를 만들 수 있고, 훈련 데이터와 검증 데이터를 섞어버릴 수 있으며, 논문 초록의 한 문장을 과장해 결론처럼 쓸 수 있습니다. LLM의 답이 매끄러울수록 사람은 더 쉽게 안심합니다. 그러나 연구에서 안심은 증거를 확인한 뒤에야 와야 합니다.

문맥 창과 도구 사용을 배운다는 것은 결국 믿음의 위치를 조정하는 일입니다. 모델의 말투를 믿는 것이 아니라, 모델이 어떤 자료를 보았는지, 어떤 도구를 썼는지, 어떤 결과를 근거로 답했는지 확인하는 것입니다. 좋은 답변은 예쁜 문장만으로 만들어지지 않습니다. 좋은 입력 자료, 정확한 도구 실행, 남겨진 출처, 사람이 다시 확인한 판단이 함께 있어야 합니다.

이 원칙은 작은 과제에서도 그대로 적용됩니다. 예를 들어 어떤 학생이 "TP53과 관련된 질병을 알려줘"라고 물었다고 합시다. 모델은 곧바로 그럴듯한 질병 목록을 만들 수 있습니다. 하지만 더 좋은 흐름은 조금 다릅니다. 먼저 TP53이라는 표기가 정확한지 확인하고, 사람 유전자인지 생쥐 유전자인지처럼 생물종을 분명히 한 뒤, ClinVar 같은 전문 데이터베이스나 논문 원문에서 근거를 확인하게 하는 것입니다. 지금 데이터베이스 이름을 많이 외울 필요는 없습니다. 한 가지 믿을 만한 출처에서 공식 표기와 근거를 확인한다는 습관이 먼저입니다. 모델의 흐릿한 기억보다 이런 출처가 훨씬 구체적이지만, 출처마다 목적과 범위가 다르므로 서로 같은 무게로 읽어서는 안 됩니다. 모델이 답을 빨리 만드는 능력보다, 답이 어떤 근거 위에 있는지 드러내게 하는 능력이 더 중요합니다.

용어 메모

유전자 공식 약어(gene symbol): 유전자를 짧게 부르는 공식 약어입니다. TP53 같은 표기가 여기에 해당합니다.

생물종(species): 사람, 생쥐, 초파리처럼 어떤 생물종인지 가리키는 말입니다.

ClinVar: 변이와 질병 관련 정보를 확인할 때 쓰는 대표적인 전문 데이터베이스 중 하나입니다.

데이터베이스 주석: 데이터베이스가 유전자나 단백질에 붙여 둔 설명 정보입니다.

논문 읽기에서도 마찬가지입니다. 초록만 넣고 요약을 받으면 간단하지만, 연구의 위험한 지점은 보통 방법과 보충자료에 숨어 있습니다. 어떤 연구 대상 집단을 썼는지, 제외 기준은 무엇인지, 실험 날짜나 장비 차이에서 온 흔들림을 어떻게 줄였는지, 통계 검정은 어떤 가정을 두었는지, 그림 설명이 실제로 무엇을 보여주는지 확인해야 합니다. 그래서 AI에게 논문을 읽힐 때는 "좋게 요약해줘"보다 "이 연구가 틀릴 수 있는 지점을 찾아줘"라는 질문이 더 유익할 때가 많습니다. 좋은 독자는 논문을 칭찬하는 사람만이 아니라, 논문이 기대고 있는 다리를 확인하는 사람입니다.

용어 메모

보충자료: 논문 본문에 다 넣지 못한 추가 표, 그림, 방법 설명입니다.

통계 검정: 관찰한 차이가 우연인지 아닌지 따져보기 위한 계산 절차입니다.

LLM-Wiki식 노트는 이런 읽기를 오래 남기는 데 도움을 줍니다. 오늘 읽은 논문의 요약이 내일 사라지지 않고, "세포분화", "전사인자", "실험 조건", "대조군", "측정값" 같은 개념 문서에 연결되면 다음 공부가 쉬워집니다. 처음에는 작은 메모라도 괜찮습니다. 이 논문은 무엇을 물었는지, 어떤 방법이 낯설었는지, 어떤 한계가 있었는지, 내 질문과 어떻게 이어지는지 적어두면 됩니다. 나중에 에이전트는 이 노트들을 읽고, 내가 어떤 맥락에서 질문하는지 더 잘 이해할 수 있습니다.

여기서 중요한 것은 노트의 양보다 구조입니다. 논문을 많이 저장해두었다고 해서 지식이 쌓이는 것은 아닙니다. PDF가 가득한 폴더는 기억을 대신해주지 않습니다. 어떤 논문이 어떤 개념과 연결되는지, 어떤 결과가 서로 충돌하는지, 어떤 용어가 분야마다 다르게 쓰이는지 표시해야 합니다. AI는 그 구조를 만드는 일을 도와줄 수 있지만, 구조의 기준은 연구자의 질문에서 나옵니다. 그래서 문맥 창의 문제는 결국 공부의 문제로 이어집니다. 무엇을 가까이에 두고 생각할 것인가. 무엇을 서로 연결해 기억할 것인가.

의생명과학 학생에게 이 태도는 처음부터 몸에 익히는 편이 좋습니다. 논문을 읽을 때는 원문을 옆에 두고, 데이터를 볼 때는 코드 실행 결과를 옆에 두고, AI가 설명한 내용을 들을 때는 출처와 조건을 옆에 두십시오. AI는 여러분의 기억을 넓히고, 손을 빠르게 하고, 초안을 만들어줄 수 있습니다. 그러나 무엇을 눈앞에 놓고 생각할지는 여러분이 정해야 합니다. **문맥 창은 모델의 눈앞에 놓인 책상입니다.** 그 책상 위에 무엇을 올릴지 고르는 일이, 앞으로의 공부와 연구에서 점점 더 중요한 능력이 됩니다.

이 책상 비유를 조금 더 밀고 가보면, 좋은 연구자는 책상 정리를 잘하는 사람입니다. 필요한 논문은 가까이에 두고, 오래된 자료와 최신 자료를 구분하고, 서로 다른 실험 조건의 데이터를 섞어놓지 않으며, 방금 계산한 결과와 아직 확인하지 않은 추측을 한곳에 던져두지 않습니다. 문맥 창도 그렇게 써야 합니다. 모델에게 너무 많은 것을 한꺼번에 넣으면, 마치 지저분한 책상 위에서 중요한 논문 한 장을 찾는 일처럼 어려워질 수 있습니다. 반대로 너무 적게 넣으면, 모델은 빈칸을 자기 기억으로 채우려 합니다. 그래서 좋은 프롬프트는 길이가 아니라 정리가 중요합니다. 필요한 자료를 고르고, 자료의 성격을 알려주고, 답변의 경계를 정하고, 확인할 지점을 지정해야 합니다. 이것은 귀찮은 형식이 아니라 사고의 위생입니다. 실험실에서 오염을 막기 위해 작업대를 정리하듯, AI와 공부할 때도 문맥을 정리해야 합니다. 정리된 문맥은 더 좋은 답변을 만들 뿐 아니라, 사용자가 자신의 질문을 더 분명하게 보게 합니다.

도구 사용도 결국 같은 원칙을 따릅니다. 좋은 실험실에서는 장비마다 쓰임이 다르고, 현미경으로 할 일과 피펫으로 할 일을 섞지 않습니다. 실험실이 아직 낯선 학생이라면 자와 계산기와 사전을 한꺼번에 같은 도구로 쓰지 않는다고 생각해도 됩니다. LLM과 함께 일할 때도 검색, 코드 실행, 데이터베이스 조회, 문장 생성의 역할을 나누어야 합니다. 최신 논문은 검색으로 찾고, 숫자는 코드로 계산하고, 유전자와 변이의 표준 이름은 데이터베이스로 확인하고, 설명문은 모델의 언어 능력을 빌려 다듬는 식입니다. 하나의 모델에게 모든 일을 말로 처리하게 하면 편하지만, 편한 만큼 오류가 숨어들 자리가 생깁니다. 반대로 각 도구의 역할을 분명히 하면 결과를 확인하기가 쉬워집니다. 학생은 AI에게 무엇을 물을지만 배우는 것이 아니라, 어떤 도구로 확인할지도 함께 배워야 합니다. 그것이 문맥 창과 도구 사용을 공부하는 실제 이유입니다. 좋은 답은 좋은 도구 배치에서 나옵니다.

작은 실습

공개 초록 하나나 강의자료의 짧은 문단을 고른 뒤 같은 질문을 두 번 해보십시오. 먼저 자료 없이 주제만 말하고 "이 내용이 무엇을 보였는지 설명해줘"라고 묻습니다. 다음에는 그 문단을 붙여넣고 "아래 문단 안에서만 연구 질문, 방법, 결과, 한계를 나누어 설명해줘. 문단에 없는 내용은 추측이라고 표시해줘"라고 묻습니다. 두 답변에서 근거의 위치가 어떻게 달라지는지 표시해보면, 문맥 창이 단순한 긴 입력칸이 아니라 모델 앞에 놓인 책상이라는 비유가 훨씬 선명해집니다.

이 장의 이야기는 결국 “무엇을 모델 앞에 놓을 것인가”라는 질문으로 돌아옵니다. 학생이 공부할 때 책상 위에 교과서, 강의노트, 빈 종이, 계산기를 어떻게 올려두느냐에 따라 공부의 흐름이 달라지듯, 모델의 문맥에도 무엇을 넣고 무엇을 빼는지가 중요합니다. 너무 적게 주면 모델은 기억으로 빈칸을 채우려 하고, 너무 많이 주면 중요한 자료가 묻힙니다. 도구도 마찬가지입니다. 검색이 필요한 일을 기억으로 처리하게 하면 낡은 답이 나올 수 있고, 계산이 필요한 일을 말로 처리하게 하면 그럴듯한 실수가 나올 수 있습니다. 좋은 AI 사용자는 모델이 똑똑한지 아닌지만 묻지 않습니다. 모델 앞의 책상이 잘 정리되어 있는지, 필요한 도구가 제대로 놓여 있는지, 결과를 다시 확인할 길이 남아 있는지를 봅니다. 이것이 앞으로 의생명과학 학생에게 필요한 새로운 공부 습관입니다.

10장. 모델도 생각할 시간이 필요하다

머릿속 계산의 한계

사람도 복잡한 문제를 머릿속으로만 풀려고 하면 자주 틀립니다. 간단한 덧셈은 암산으로 할 수 있지만, 숫자가 커지고 조건이 늘어나면 종이에 중간 과정을 적어야 합니다. 실험 데이터도 마찬가지입니다. 샘플 수가 몇 개 되지 않을 때는 눈으로 대강 볼 수 있지만, 수천 개의 세포와 수만 개의 유전자가 들어 있는 표를 머릿속으로 처리할 수는 없습니다. 우리는 표를 만들고, 중간 계산을 남기고, 코드를 쓰고, 그림을 그리며 생각을 밖으로 꺼냅니다. 카파시가 “models need tokens to think”라고 말할 때도 비슷한 이야기를 합니다 (링크). **모델에게도 생각할 자리가 필요합니다.** 다만 여기서 말하는 생각은 신비로운 내면의 독백이 아닙니다. LLM은 토큰을 왼쪽에서 오른쪽으로 생성하므로, 중간 과정을 토큰으로 써 내려갈 때 더 많은 계산을 단계별로 나누어 수행할 수 있습니다. 답만 곧장 요구하면, 모델은 많은 일을 한순간에 압축해야 합니다. 그 압축이 쉬운 문제에서는 성공하지만, 조금만 어려워져도 실패할 수 있습니다.

용어 메모

토큰: 모델이 글을 읽고 쓸 때 사용하는 작은 글자 조각입니다.

중간 과정: 바로 답으로 가지 않고 조건, 계산, 판단을 순서대로 남기는 과정입니다.

이 말은 학생에게도 꽤 위로가 됩니다. 어려운 문제를 한 번에 이해하지 못한다고 해서 머리가 나쁜 것이 아닙니다. 사람은 원래 생각을 밖으로 꺼내면서 배웁니다. 수학 문제를 풀 때 식을 적고, 생물학 그림을 그릴 때 화살표를 긋고, 실험 계획을 세울 때 조건을 표로 나누는 이유가 여기에 있습니다. LLM도 비슷하게, 답을 바로 내기보다 조건과 중간 결과를 글로 남길 때 더 안정적으로 문제를 다룰 수 있습니다. 물론 모델의 “생각”은 사람의 내면 경험과 같지 않습니다. 하지만 외부에 적힌 중간 과정이 다음 답변의 재료가 된다는 점에서는, 칠판과 노트가 사람의 사고를 돕는 방식과 닮아 있습니다. 그러므로 **“생각할 토큰”이라는 표현은 모델의 신비한 마음을 말하는 것이 아니라, 복잡한 문제를 작은 단계로 나누어 처리하게 하는 실용적인 방법으로 이해하면 됩니다.**

모델에게도 칠판이 필요하다

카파시가 드는 산수 예시는 이 원리를 잘 보여줍니다. 어떤 학생이 사과와 오렌지를 샀고, 전체 가격과 오렌지 가격이 주어졌을 때 사과 하나의 가격을 묻는 문제가 있습니다. 답만 먼저 말하고 뒤에 풀이를 붙이는 방식은 겉보기에는 효율적입니다. 그러나 모델 입장에서는 “정답 숫자”를 내는 바로 그 토큰에서 거의 모든 계산을 끝내야 합니다. 그 뒤에 이어지는 설명은 이미 나온 답을 정당화하는 문장이 되기 쉽습니다. 반대로 오렌지 총액을 먼저 계산하고, 전체 금액에서 빼고, 남은 금액을 사과 개수로 나누는 식으로 중간 단계를 먼저 쓰게 하면, 모델은 계산을 여러 토큰에 나누어 수행할 수 있습니다. 예를 들어 “오렌지 2개가 각각 500원이고, 사과 4개와 오렌지 2개의 전체 가격이 5,000원이라면 사과 하나는 얼마인가”라는 문제를 생각해봅시다. 답만 요구하면 모델은 바로 “1,000원”을 맞힐 수도 있지만, 틀렸을 때 어디서 틀렸는지 보기 어렵습니다. 단계로 쓰면 “오렌지 총액은 2 x 500 = 1,000원, 전체에서 오렌지 값을 빼면 사과 4개의 값은 4,000원, 따라서 사과 하나는 4,000 / 4 = 1,000원”이 됩니다. 각각의 작은 단계는 한 토큰이나 몇 토큰 안에서 처리하기 쉬운 문제입니다. 앞에서 계산한 중간 결과는 뒤의 문맥 안에 남아 다음 토큰을 고르는 데 도움을 줍니다. 그래서 풀이를 쓰게 하는 것은 사람에게 보여주기 위한 친절한 설명만이 아닙니다. 모델 자신이 문제를 더 잘 풀기 위한 작업 공간이기도 합니다.

이 원리는 LLM이 어떻게 계산하는지와 연결됩니다. 모델은 매 토큰을 생성할 때 고정된 신경망 계산을 한 번 통과합니다. 그 계산은 매우 크고 복잡하지만 무한하지 않습니다. 카파시는 한 토큰에서 일어나는 계산량이 제한되어 있으므로, 어려운 문제를 한 토큰에 몰아넣지 말고 여러 토큰에 분산해야 한다고 설명합니다 (링크). 사람도 한 줄짜리 답안보다 풀이 과정이 있는 답안에서 실수를 찾기 쉽습니다. 모델도 중간 단계가 있으면 다음 단계가 더 쉬워지고, 사용자도 어디서 틀렸는지 검토할 수 있습니다. 이것은 “길게 말하면 더 똑똑해 보인다”는 문제가 아닙니다. 긴 문장이 항상 좋은 것도

아닙니다. 중요한 것은 필요한 계산과 판단을 적절한 중간 단위로 나누는 것입니다. 어려운 문제에서는 모델에게 바로 답을 요구하기보다, 먼저 변수와 조건을 정리하고, 가능한 풀이 경로를 나누고, 계산을 단계별로 수행하게 해야 합니다.

계산은 도구로 확인한다

하지만 여기서 한 가지를 조심해야 합니다. 중간 과정을 쓰게 한다고 해서 모든 문제가 해결되는 것은 아닙니다. 모델이 중간 단계 자체를 틀릴 수 있기 때문입니다. 특히 숫자 계산, 문자열 처리, 표의 행 개수 세기처럼 정확한 절차가 필요한 일은 토큰으로 생각하게 하는 것보다 코드나 계산 도구로 실행하게 하는 편이 훨씬 안전합니다. 카파시도 산수 문제가 조금만 복잡해지면, 모델에게 머릿속 계산을 시키기보다 Python 같은 도구를 쓰게 하는 편이 낫다고 말합니다 (링크). 이것은 의생명과학 학생에게 매우 중요한 습관입니다. 표에 몇 행이 있는지, 각 조에 샘플이 몇 개 있는지, 평균과 표준편차가 얼마인지는 말로 추정할 일이 아닙니다. 계산 도구로 확인하고, 가능하면 그 과정까지 남겨야 합니다. **모델은 코드를 작성하는 데 도움을 줄 수 있지만, 계산의 믿음은 실행 가능한 절차에서 와야 합니다.**

용어 메모

문자열 처리: 글자, 숫자, 기호로 된 줄을 정확히 세거나 바꾸거나 비교하는 일입니다.

표준편차: 값들이 평균 주변에 얼마나 퍼져 있는지 나타내는 숫자입니다.

Python: 데이터 계산과 자동화에 자주 쓰는 프로그래밍 언어입니다.

Counting과 spelling 문제도 같은 원리를 보여줍니다. 사람에게는 점의 개수를 세거나 단어 속 특정 글자를 찾는 일이 쉬워 보입니다. 그러나 모델은 글자를 눈으로 보는 것이 아니라 토큰을 봅니다. 카파시는 많은 점의 개수를 묻는 예시와 “ubiquitous” 같은 단어의 글자 위치를 묻는 예시를 통해, 모델이 이런 일을 왜 예상보다 못할 수 있는지 설명합니다 (링크). 모델에게는 점의 긴 문자열이 몇 개의 토큰으로 압축되어 보일 수 있고, 단어도 사람이 보는 글자 단위가 아니라 tokenizer가 정한 조각으로 보일 수 있습니다. 그러면 “몇 개냐”, “세 번째 글자마다 봅아라” 같은 문제는 사람에게 쉬워도 모델에게는 이상하게 어렵습니다. 이런 일을 해결하려면 모델에게 직접 세라고 하기보다, 문자열을 코드로 넘기고 프로그램이 세게 하는 편이 낫습니다. 사람도 현미경 사진의 세포 수를 대강 눈으로 세기보다 image analysis 프로그램을 쓰는 편이 정확합니다. LLM도 비슷합니다. 언어적 설명과 정확한 계산을 분리해야 합니다.

의생명과학에서 “생각할 토큰”은 단순히 풀이를 길게 쓰는 기술이 아닙니다. 질문을 단계로 나누는 태도입니다. 예를 들어 “이 논문이 맞는지 평가해줘”라고 묻는 것은 너무 큽니다. 먼저 이 논문이 무엇을 물었는지 정리하고, 어떤 자료를 사용했는지 확인하고, 비교한 두 대상이 적절한지 보고, 결과가 결론을 뒷받침하는지 나누어 물어야 합니다. 수업에서 받은 작은 표를 볼 때도 마찬가지입니다. 어떤 열이 조 이름인지, 어떤 열이 측정값인지, 빠진 값은 없는지, 어떤 그림이 가장 이해하기 쉬운지 단계가 필요합니다. 이 단계들은 사람의 사고를 돕고, 모델의 생성도 돕고, 나중의 검토도 돕습니다. **AI를 잘 쓴다는 것은 한 번에 큰 답을 받는 일이 아니라, 큰 질문을 검증 가능한 작은 질문으로 나누는 일입니다.** 모델에게 생각할 시간을 주는 일은 결국 우리 자신에게도 생각할 구조를 주는 일입니다.

학생이 실제로 사용할 때는 몇 가지 문장을 습관처럼 붙여도 좋습니다. “바로 답하지 말고, 먼저 주어진 조건을 정리해줘.” “계산이 필요한 부분은 코드로 확인해줘.” “중간 결과를 표로 남기고, 마지막에 결론을 한 문단으로 써줘.” “확실하지 않은 부분은 추측이라고 표시해줘.” 이런 지시는 모델을 더 느리게 만드는 것처럼 보이지만, 실제로는 더 안전하게 만듭니다. 빠른 답은 편하지만, 의생명과학에서는 편한 답보다 확인 가능한 답이 중요합니다. 모델의 토큰은 작업 공간이고, 도구는 계산기이며, 사용자의 질문은 실험 설계서에 가깝습니다. 좋은 실험이 조건을 나누고 대조군을 세우듯, 좋은 AI 사용도 사고의 단계를 나누고 검증의 자리를 둡니다.

생각할 토큰을 준다는 말은 모델에게 장황한 말을 허락한다는 뜻이 아닙니다. 장황함과 사고는 다릅니다. 어떤 답변은 길지만 핵심이 없고, 어떤 답변은 짧지만 중간 판단이 잘 드러납니다. 학생이 요청해야 하는 것은 길이가 아니라 구조입니다. “먼저 내가 준 정보만 정리하고, 그다음에 필요한 추가 정보를 따로 말하고, 마지막에 가능한 결론과 불확실성을 구분해줘”라고 요청하면 모델의 답변은 훨씬 검토하기 쉬워집니다. 특히 생명과학에서는 정보의 출처와 확실성 수준이 섞이면 위험합니다. 논문에서 직접 나온 결과, 저자의 해석, 모델의 추측, 사용자의 가정이 한 문단 안에 뒤섞이면 읽는 사람은 어디까지 믿어야 할지 알기 어렵습니다. 좋은 답변은 이 층위를 나눕니다. 좋은 사용자는 모델에게 이 층위를 나누도록 요구합니다. 이것이 “생각할 시간을 준다”는 말의 실제 의미입니다. 모델이 더 많은 토큰을 쓰게 하되, 그 토큰들이 검증 가능한 역할을 갖도록 만드는 것입니다.

카파시가 보여주는 사례 중 흥미로운 것은, 모델이 처음에는 틀린 답을 말하다가 중간 과정을 쓰게 하면 맞는 답에 가까워지는 장면입니다. 이것은 사람의 풀이와 닮았습니다. 수학 문제를 풀 때 답만 적으면 실수해도 어디서 잘못됐는지 알 수 없습니다. 그러나 중간 줄이 있으면 부호를 잘못 붙였는지, 나눗셈을 잘못했는지, 조건을 빠뜨렸는지 볼 수 있습니다. LLM도 생성한 중간 결과가 문맥에 남으면, 뒤의 토큰이 그 결과를 참고합니다. 이때 문맥은 칠판처럼 작동합니다. 칠판에

아무것도 쓰지 않고 머릿속으로만 복잡한 계산을 하면 실수하기 쉽지만, 칠판에 조건과 중간값을 적으면 다음 사람이 이어서 검토할 수 있습니다. 모델에게 풀이 과정을 쓰게 하는 것은 모델의 내부를 완전히 들여다보는 일은 아니지만, 최소한 외부에 남은 흔적을 통해 답을 점검할 수 있게 합니다. 사용자는 그 흔적을 읽고, 틀린 부분을 다시 질문하고, 필요한 경우 계산을 코드로 돌릴 수 있습니다. 이 반복이 AI와 함께 생각하는 기본 리듬입니다.

내부 생각보다 외부 기록

다만 최근의 추론 모델(reasoning model)에서는 내부 사고 과정이 사용자에게 그대로 보이지 않는 경우가 많습니다. 모델은 내부적으로 더 긴 추론을 할 수 있지만, 최종 답변에는 그 요약만 보여줄 수 있습니다. 이것은 보안, 안전, 품질 관리 등 여러 이유와 관련이 있습니다. 학생 입장에서는 모델의 모든 생각을 보지 못한다고 해서 완전히 손을 놓을 필요는 없습니다. 대신 외부적으로 검토 가능한 산출물을 요구하면 됩니다. "계산에 사용한 식을 보여줘", "최종 표를 만들어줘", "근거 문장을 원문에서 찾아줘", "실행한 코드와 결과를 함께 보여줘" 같은 요청은 여전히 유효합니다. **내부 사고 과정(chain of thought)을 보는 것보다 중요한 것은, 최종 결론을 뒷받침하는 공개된 근거와 절차가 있는가입니다.** 과학 논문도 연구자의 머릿속 모든 생각을 보여주지는 않습니다. 대신 방법(methods), 결과(results), 그림(figure), 부록 자료를 통해 검토 가능한 흔적을 남깁니다. AI와의 작업에서도 같은 기준을 적용하면 됩니다. 모델의 내면을 궁금해하기보다, 검증 가능한 외부 기록을 남기게 하는 편이 더 실용적입니다.

용어 메모

추론 모델(reasoning model): 어려운 문제를 더 오래 붙잡고 단계적으로 풀도록 훈련된 모델입니다.

사고 과정(chain of thought): 모델의 내부 풀이 과정을 가리키는 말입니다. 실제 서비스에서는 그대로 보이지 않을 수 있습니다.

methods / results: 논문에서 실험 방법과 결과를 나누어 적는 부분입니다.

부록 자료: 논문 본문에 다 싣지 못한 추가 자료입니다.

의생명 데이터 분석에서 이 원리는 아주 구체적으로 나타납니다. 처음에는 복잡한 연구 데이터가 아니라 작은 실습 표를 떠올려도 충분합니다. 모델이 "처리군이 더 높아 보입니다"라고 말하는 것만으로는 충분하지 않습니다. 어떤 열을 비교했는지, 빈칸이나 이상한 값은 어떻게 처리했는지, 평균을 냈는지 중앙값을 봤는지, 그림이 실제 표와 맞는지 확인해야 합니다. 모델이 코드를 제안했다면 그 코드는 실제 데이터에서 실행되어야 하고, 오류가 나면 수정되어야 하며, 결과 표가 확인되어야 합니다. 그다음에야 생물학적 해석을 붙일 수 있습니다. 이 순서를 거꾸로 하면 위험합니다. 먼저 그럴듯한 이야기를 만들고, 나중에 데이터를 맞추려 하면 과학이 아니라 이야기 꾸미기가 됩니다. LLM은 이야기 꾸미기에 매우 능하기 때문에, 사용자는 더더욱 절차를 앞세워야 합니다. 생각할 토큰은 절차를 드러내는 토큰이어야 합니다. 계산은 계산으로, 해석은 해석으로, 추측은 추측으로 남겨야 합니다.

좋은 과정이 빠른 답보다 낫다

학생이 이 습관을 갖추면 AI 사용은 단순한 편법이 아니라 공부 방법이 됩니다. 질문을 단계로 나누는 과정에서 자기 이해의 빈틈이 드러납니다. 모델에게 설명을 요청하다 보면, 자신이 사실은 어떤 용어를 모르는지 알게 됩니다. 모델의 답을 검토하다 보면, 근거와 해석을 구분하는 눈이 생깁니다. 코드 실행을 요구하다 보면, 말로만 아는 통계와 실제 계산의 차이를 느끼게 됩니다. 이 모든 과정은 대학 공부의 핵심입니다. AI가 대신 공부해주는 것이 아니라, AI와 대화하면서 공부의 구조가 더 선명해지는 것입니다. 카파시의 "models need tokens to think"라는 말은 결국 우리에게도 돌아옵니다. 사람도 생각을 밖으로 꺼내야 잘 배웁니다. 메모하고, 표를 만들고, 그림을 그리고, 중간 결론을 적고, 다시 고치는 과정에서 지식은 단단해집니다. 모델에게 생각할 자리를 주는 일은, 학생 자신에게도 생각할 자리를 마련하는 일입니다.

이 장의 이야기를 한 문장으로 줄이면, 빠른 답보다 좋은 과정이 중요하다는 말입니다. AI는 빠른 답을 너무 쉽게 줍니다. 그래서 학생은 자신이 이해했다고 느끼기 쉽고, 과제는 금방 끝난 것처럼 보입니다. 그러나 진짜 공부는 답이 나온 뒤에 시작될 때가 많습니다. 왜 그 답이 나왔는지, 다른 답은 가능하지 않은지, 어떤 조건이 바뀌면 결론이 달라지는지, 계산으로 확인할 부분은 무엇인지 따져보아야 합니다. 모델에게 생각할 토큰을 주는 일은 결국 이런 질문을 위한 공간을 만드는 일입니다. 의생명과학에서 좋은 결론은 대개 한 번의 번뜩임이 아니라, 여러 작은 확인이 쌓여 만들어집니다. AI와 함께 공부할 때도 그 리듬을 잃지 않아야 합니다. 빠르게 시작하되, 천천히 확인하는 사람이 되어야 합니다.

실제로 좋은 프롬프트는 작은 연구 계획서와 닮아 있습니다. 문제를 정의하고, 사용할 자료를 정하고, 중간 산출물을 요청하고, 검증 방법을 붙입니다. "답만 말해줘"가 아니라 "조건을 정리하고, 필요한 계산은 코드로 확인하고, 불확실한 가정은 따로 표시해줘"라고 말하면 모델은 더 좋은 작업 공간을 갖게 됩니다. 이 지시는 모델을 위한 것이면서 동시에 학생 자신을 위한 것입니다. 무엇을 모르는지, 무엇을 계산해야 하는지, 어디서 판단이 필요한지 스스로 보게 만들기 때문입니다.

AI가 잘 작동하는 순간에도 학생은 구경꾼이 아니라 설계자여야 합니다. 생각할 토큰을 주는 일은 설계를 밖으로 꺼내는 일입니다. 그 설계가 분명할수록 모델도 덜 헤매고, 사용자도 답을 더 잘 검토할 수 있습니다.

그러므로 10장의 실천은 아주 단순한 문장으로 시작할 수 있습니다. "한 번에 답하지 말고, 먼저 조건을 정리해줘." "계산은 코드로 확인하고, 해석은 따로 써줘." "내가 준 자료에서 직접 확인되는 내용과 네가 추측한 내용을 나누어줘." 이런 문장들은 화려하지 않지만, 모델에게 작업 공간을 만들어줍니다. 동시에 학생에게도 생각의 순서를 만들어줍니다. AI 사용이 위험해지는 순간은 모델이 너무 빨리 답할 때만이 아닙니다. 사용자가 자신의 질문을 너무 빨리 끝냈다고 느낄 때도 위험해집니다. **좋은 공부는 답을 얻은 뒤에도 한 번 더 멈추어, 그 답이 어떤 길로 왔는지 살피는 일입니다.**

11장. 강화학습과 추론 모델

예제를 따라 하는 공부 다음

공부에는 여러 종류가 있습니다. 설명을 읽는 공부가 있고, 잘 풀린 예제를 따라 하는 공부가 있습니다. 그러나 그것만으로는 충분하지 않을 때가 많습니다. 결국 스스로 문제를 풀어보아야 합니다. 틀려보고, 다시 풀어보고, 어떤 풀이가 맞는 답으로 이어졌는지 몸에 익혀야 합니다. 카파시는 LLM의 훈련 과정을 학교 공부에 비유합니다. 사전학습(pre-training)은 교과서와 인터넷 문서를 많이 읽으며 배경지식을 쌓는 과정에 가깝고, 지도 미세조정(supervised fine-tuning, SFT)은 잘 쓴 답변 예시를 보고 어시스턴트다운 말투와 행동을 배우는 과정에 가깝습니다. SFT가 모범답안을 베껴 쓰며 답변의 형식을 익히는 공부라면, 강화학습(reinforcement learning, RL)은 직접 문제를 풀어보고 채점 결과에 따라 풀이 습관을 고치는 공부에 가깝습니다. 이름은 어렵지만, 출발점은 익숙합니다. 어떤 시도가 좋은 결과로 이어졌는지 보고, 그런 방향의 행동을 더 자주 하도록 훈련하는 것입니다 (링크).

용어 메모

사전학습(pre-training): 모델이 많은 글을 먼저 읽으며 배경 패턴을 배우는 단계입니다.

지도 미세조정(SFT): 좋은 질문과 답변 예시를 보고 어시스턴트다운 답변 방식을 배우는 단계입니다.

강화학습(RL): 좋은 결과로 이어진 행동을 더 자주 하도록 훈련하는 방법입니다.

강화학습이 SFT와 다른 점은 "정답 풀이를 사람이 모두 써주지 않아도 된다"는 데 있습니다. SFT에서는 사람이 이상적인 답변을 써주고, 모델은 그것을 흉내 냅니다. 하지만 수학 문제나 코드 문제처럼 답이 맞았는지 비교적 명확하게 확인되는 영역에서는, 모델이 여러 풀이를 시도하고 정답에 도달한 경로를 강화할 수 있습니다. 사람 라벨러(labeler)가 모든 풀이를 미리 작성하지 않아도, 최종 답이 맞는지 확인하는 방식으로 학습 신호를 줄 수 있습니다. 이것이 추론 모델(reasoning model)의 중요한 배경입니다. 모델은 단순히 예쁜 답변을 흉내 내는 데서 조금 벗어나, 문제를 풀기 위한 중간 전략을 더 많이 탐색하게 됩니다. 카파시는 DeepSeek R1과 OpenAI의 추론 모델을 예로 들며, 이런 모델들이 더 긴 사고 과정을 사용하도록 훈련된다고 설명합니다 (링크). DeepSeek R1 같은 이름을 지금 외울 필요는 없습니다. 여기서는 2025년 무렵부터 추론과 강화학습을 강하게 내세운 모델들이 등장했고, 그 흐름이 LLM 사용법을 바꾸고 있다는 정도를 붙잡으면 됩니다. 물론 그 내부의 모든 사고 과정(chain of thought)이 사용자에게 그대로 보이는 것은 아닙니다. 우리가 보는 것은 요약된 답변일 수 있지만, 모델 안에서는 더 긴 탐색과 선택이 일어날 수 있습니다.

용어 메모

SFT: supervised fine-tuning의 줄임말입니다. 이 책에서는 지도 미세조정이라고도 부릅니다.

추론 모델(reasoning model): 문제를 바로 답하지 않고 더 긴 풀이와 검토를 사용하도록 훈련된 모델입니다.

사고 과정(chain of thought): 모델의 내부 풀이 과정을 가리키는 말입니다. 사용자가 항상 볼 수 있는 것은 아닙니다.

정답을 확인할 수 있는 문제

이 과정이 잘 작동하려면 평가가 가능해야 합니다. 수학 문제는 답이 맞는지 틀리는지 비교적 분명합니다. 코드도 테스트를 통과하는지 확인할 수 있습니다. 퍼즐이나 형식 논리 문제도 어느 정도 검증할 수 있습니다. 카파시는 이런 영역을 답을 확인할 수 있는 영역(verifiable domain)으로 설명합니다. 답을 확인할 수 있는 문제에서는 모델이 많은 시도를 해보고, 맞는 시도에서 배울 수 있습니다. 반대로 좋은 에세이, 좋은 농담, 좋은 연구 질문, 좋은 생물학적 해석처럼 평가가 애매한 문제는 훨씬 어렵습니다. 어떤 답이 더 좋은지 사람이 판단할 수는 있지만, 그 판단은 하나의 정답처럼 명확하지 않습니다.

RLHF는 이 애매한 영역을 다루기 위해 사람의 선호를 이용합니다. 여러 답변 중 사람이 더 낫다고 고른 쪽을 바탕으로, 어떤 답변이 좋아 보이는지 점수를 주는 점수표 모델(reward model)을 만듭니다. 여기에는 두 번의 간접화가 들어갑니다.

먼저 사람의 복잡한 판단을 “A가 B보다 낫다”는 선호 자료로 줄입니다. 그다음 그 선호를 다시 점수표 모델이 흉내 냅니다. 그래서 모델이 점수표 모델의 점수를 높이는 데만 지나치게 맞춰지면, 실제로 좋은 답이 아니라 점수표 모델이 좋아하는 모양의 답을 만들 수도 있습니다. 예를 들어 정답을 더 정확히 말하기보다, 점수표 모델이 좋아하는 길고 단정한 형식으로 답을 늘이는 식입니다. 이것을 점수 기준 속이기(reward hacking)라고 부르기도 합니다. 학생이 기억할 점은 간단합니다. 정답이 분명한 문제에서의 강화학습과, 사람이 “이 답이 더 좋아 보인다”고 평가하는 RLHF는 같은 이름 아래 있어도 신뢰의 성격이 다릅니다. 그래서 추론 모델의 힘을 보면서도, 그 힘이 어디에서 잘 발휘되고 어디서 흐려지는지 함께 보아야 합니다.

용어 메모

답을 확인할 수 있는 영역(verifiable domain): 답이 맞는지 비교적 분명하게 확인할 수 있는 문제 영역입니다.

RLHF: 사람의 선호를 이용해 모델 답변을 더 낫게 조정하는 훈련 방식입니다.

점수표 모델(reward model): 어떤 답변이 더 좋은지 점수로 흉내 내는 모델입니다.

점수 기준 속이기(reward hacking): 실제 목표보다 점수 기준의 허점을 맞추는 방향으로 행동이 바뀌는 현상입니다.

생물학의 느린 채점

이 구분은 의생명과학에서 매우 중요합니다. 생물학에는 답을 비교적 빨리 확인할 수 있는 부분과 그렇지 않은 부분이 섞여 있습니다. 유전자 리스트의 길이를 세는 일, 샘플 수를 확인하는 일, 코드가 실행되는지 보는 일은 비교적 검증 가능합니다. 시퀀싱 자료를 다룰 때 나오는 alignment rate도 계산 자체는 확인할 수 있는 숫자입니다. 아직 시퀀싱을 배우지 않은 학생은 “읽어낸 DNA나 RNA 조각이 기준 유전체에 얼마나 잘 맞았는지 보는 비율” 정도로만 알고 지나가도 됩니다. 그러나 어떤 pathway가 질병의 원인인지, 어떤 세포 상태 변화가 치료 반응을 설명하는지, 어떤 후보 유전자가 후속 실험의 우선순위가 되어야 하는지는 훨씬 어렵습니다. 최종 답이 바로 확인되지 않기 때문입니다. 실험을 해야 하고, 독립 데이터에서 봐야 하고, 때로는 몇 달 뒤에야 결과가 나옵니다. 그러므로 추론 모델이 생물학 연구에 도움이 되더라도, 수학 문제를 풀 때와 같은 방식으로 모든 것을 맡길 수는 없습니다. 모델은 후보를 만들고, 논리를 정리하고, 가능한 반례를 제안할 수 있습니다. 하지만 생물학적 주장은 실험과 데이터와 문헌 검증을 통과해야 합니다. 답을 확인할 수 있는 영역에서 배운 reasoning이 느리게 검증되는 영역으로 얼마나 잘 옮겨지는지는 아직 조심스럽게 보아야 합니다.

영역	답을 바로 채점할 수 있는가	추론 모델의 현재 강점	의생명과학에서의 사용 기준
수학 문제	네. 정답과 비교할 수 있습니다.	높음	통계 개념을 풀어보는 데 도움을 받되, 실제 계산은 코드로 확인합니다.
코드 작성	네. 테스트와 실행 결과를 볼 수 있습니다.	높음	분석 코드 초안, 오류 원인 탐색, 반복 작업 자동화에 유용합니다.
퍼즐과 형식 논리	대체로 가능합니다.	높음	가설의 논리 구조를 연습하는 데 쓸 수 있습니다.
논문 해석	부분적으로만 가능합니다.	중간	주장, 근거, 한계를 나누는 보조자로 쓰고 원문으로 확인합니다.
질병 기전 추론	느리거나 어렵습니다.	제한적	후보 설명을 넓히는 데 쓰되, 결론은 문헌과 실험으로 검증합니다.
약물 효과 예측	바로 채점하기 어렵습니다.	매우 제한적	임상적 판단이나 치료 결정에 직접 사용해서는 안 됩니다.

용어 메모

alignment rate: 시퀀싱에서 읽어낸 DNA나 RNA 조각이 기준 유전체에 얼마나 잘 맞았는지 나타내는 비율입니다.

pathway: 세포 안에서 여러 분자들이 이어져 신호나 반응을 만드는 길입니다.

강화학습을 학생 공부에 비유하면, 답안지가 있는 연습문제와 답안지가 없는 탐구 과제의 차이도 보입니다. 고등학교 수학 문제를 풀 때는 정답이 있습니다. 틀렸는지 바로 알 수 있고, 풀이를 고칠 수 있습니다. 그러나 "이 논문은 왜 중요한가" 또는 "이 데이터에서 가장 흥미로운 생물학적 질문은 무엇인가"라는 질문에는 하나의 정답지가 없습니다. 좋은 답이 있을 수는 있지만, 그것은 근거와 맥락과 목적에 따라 달라집니다. AI 시대의 공부가 어려운 이유도 여기에 있습니다. 모델은 정답이 있는 문제에서 점점 강해질 것입니다. 그러나 대학에서 중요한 많은 질문은 정답을 찾는 문제라기보다, 질문 자체를 만들고 근거를 세우는 문제입니다. 따라서 학생은 추론 모델을 쓰더라도, 그것을 "최종 판단 기계"가 아니라 "가능한 사고 경로를 넓혀주는 도구"로 보아야 합니다.

이 차이는 대학 공부의 성격을 잘 보여줍니다. 고등학교까지는 정해진 답을 빨리 찾는 훈련이 많았을 수 있습니다. 물론 대학에서도 정확한 지식과 계산은 필요합니다. 그러나 연구에 가까워질수록 더 중요한 질문은 "무엇이 답인가"만이 아니라 "이 질문을 어떻게 물어야 하는가"가 됩니다. 어떤 논문을 읽을지, 어떤 비교가 공정한지, 어떤 결과가 나오면 가설을 바꾸어야 하는지, 어떤 설명이 아직 근거가 부족한지 판단해야 합니다. 추론 모델은 이런 판단의 후보를 많이 만들어줄 수 있습니다. 하지만 후보가 많아질수록 고르는 힘이 더 필요합니다. 학생은 모델이 내놓은 사고 경로를 정답처럼 받아들이기보다, 그 경로가 어떤 근거와 어떤 검증 가능성 위에 있는지 살펴야 합니다. 이것이 추론 모델을 공부 도구로 사용할 때의 출발점입니다.

추론 모델을 어떻게 쓸까

카파시는 thinking model이 흥미로운 이유를, 단순히 인간 labeler를 흉내 내는 수준을 넘어 새로운 문제 풀이 전략이 강화학습 과정에서 생길 수 있기 때문이라고 설명합니다 (링크). 이 점은 정말 흥미롭습니다. 모델이 많은 문제를 풀며 자신만의 전략을 발견할 수 있다면, 어떤 영역에서는 사람이 바로 떠올리지 못한 풀이를 제안할 수도 있습니다. 바둑에서 AlphaGo의 move 37이 사람들에게 충격을 주었던 것처럼, 추론 모델도 언젠가 특정 문제에서 새로운 길을 보여줄 수 있습니다. 2016년 이세돌 9단과의 대국에서 AlphaGo가 둔 37번째 수는 당시 많은 인간 프로기사의 직관에서 벗어난 수였지만, 결과적으로 매우 강력한 수로 평가되었습니다. 그러나 카파시는 동시에 이것이 아직 초기 단계이며, 특히 수학과 코드처럼 검증 가능한 영역에서 먼저 빛난다고 조심스럽게 말합니다. 이 균형이 중요합니다. 새로운 가능성을 열어두되, 모든 분야에서 이미 같은 수준으로 작동한다고 믿지 않는 태도입니다. 과학에서 흥분과 검증은 함께 가야 합니다. AI가 새로운 가설을 만들 수 있다는 가능성은 매력적이지만, 그 가설이 실제 세계를 설명하는지는 따로 확인해야 합니다.

의생명과학 학생이 추론 모델을 사용할 때는 역할을 분명히 정하는 것이 좋습니다. 어려운 논문을 읽을 때 모델에게 "주장의 논리 구조를 단계별로 정리해줘"라고 물을 수 있습니다. 실험 계획을 세울 때 "이 설계에서 통제해야 할 변수를 찾아줘"라고 할 수 있습니다. 데이터 분석을 할 때 "이 결론을 약하게 만드는 대안 설명을 세 가지 제안해줘"라고 할 수 있습니다. 이런 요청은 추론 모델의 장점을 잘 살립니다. 모델은 중간 단계와 반례와 검토 기준을 만들어줄 수 있습니다. 그러나 p-value를 실제로 계산하는 일은 코드로 확인해야 하고, 논문의 핵심 문장은 원문으로 돌아가야 하며, 생물학적 결론은 실험과 독립 데이터로 닫아야 합니다. 추론 모델을 신뢰한다는 말은 아무 검토 없이 믿는다는 뜻이 아닙니다. 더 좋은 검토를 할 수 있도록 모델을 사용하는 것입니다.

결국 강화학습과 추론 모델은 AI가 "말 잘하는 모델"에서 "문제를 더 오래 붙잡는 모델"로 이동하고 있음을 보여줍니다. 이 변화는 앞으로 공부와 연구에 큰 영향을 줄 것입니다. 학생은 모델에게 단순한 요약뿐 아니라, 풀이 계획, 반례, 검증 기준, 코드 실행 전략을 요청하게 될 것입니다. 연구자는 모델에게 후보 가설을 만들게 하고, 가능한 실패 원인을 묻게 하고, 실험 설계의 빈틈을 찾게 할 수 있습니다. 그러나 이 모든 과정에서 사람의 역할은 사라지지 않습니다. 오히려 어떤 문제가 검증 가능한지, 어떤 부분은 아직 판단이 필요한지 구분하는 능력이 더 중요해집니다. 강화학습으로 훈련된 추론 모델은 강력한 도구입니다. 하지만 도구가 강해질수록, 그 도구가 무엇을 잘하고 무엇을 아직 못하는지 아는 사람이 더 필요합니다.

강화학습을 조금 더 직관적으로 느끼려면, 정답을 맞힌 경험이 행동을 바꾸는 장면을 떠올리면 됩니다. 학생이 문제집을 풀 때도 비슷합니다. 처음에는 풀이를 읽고 따라 합니다. 그다음에는 혼자 풀어보고 채점합니다. 틀렸으면 해설을 보고, 다음에는 비슷한 문제에서 다른 방법을 시도합니다. 시간이 지나면 어떤 풀이 습관이 자주 정답으로 이어지는지 몸에 남습니다. 모델의 RL도 물론 사람의 공부와 같지는 않지만, 좋은 결과를 낸 행동을 더 강화한다는 점은 비슷합니다. 중요한 것은 결과를 판단할 수 있는 기준입니다. 채점할 수 없는 문제집으로는 이런 연습을 하기가 어렵습니다. 그래서 추론 모델은 먼저 수학, 코드, 퍼즐처럼 평가가 비교적 명확한 영역에서 빠르게 발전합니다. 학생은 이 사실을 기억해야 합니다. 모델이 어떤 영역에서 강해졌다는 말이 곧 모든 지적 작업에서 같은 방식으로 강해졌다는 뜻은 아닙니다.

생물학의 많은 문제는 답이 늦게 옵니다. 어떤 유전자가 질병의 핵심 원인인지 판단하려면 문헌을 읽고, 데이터를 분석하고, 실험을 설계하고, 세포나 동물 모델에서 검증하고, 때로는 임상 자료까지 보아야 합니다. 모델이 오늘 그럴듯한 가설을 제안할 수는 있지만, 그 가설이 맞는지 확인하는 데는 시간이 걸립니다. 이 지연은 강화학습의 관점에서 큰 어려움입니다.

수학 문제는 바로 채점할 수 있지만, 생물학 가설은 바로 채점하기 어렵습니다. 따라서 AI가 생명과학 연구를 돕는 방식은, 정답을 즉시 내는 모델이라기보다 연구자의 탐색을 넓혀주는 모델에 가까울 가능성이 큽니다. 후보 설명을 만들고, 빠진 대조군을 찾고, 가능한 교란변수(confounder)를 지적하고, 문헌에서 서로 충돌하는 주장들을 모아주는 역할입니다. 이것만으로도 큰 변화입니다. 그러나 최종 판단은 여전히 데이터와 실험의 시간 속에서 이루어집니다. 학생은 AI의 빠른 언어와 생물학의 느린 검증 사이의 속도 차이를 이해해야 합니다.

더 똑똑하게 묻는 학생

추론 모델이 강력해질수록, 사용자에게 필요한 질문도 달라집니다. 예전에는 "이게 뭐야?"라고 물어도 충분히 놀라운 답을 얻을 수 있었습니다. 이제는 "이 주장에 반대되는 설명을 세 가지 만들어줘", "각 설명을 검증하려면 어떤 데이터가 필요한지 말해줘", "가장 먼저 실패할 가능성이 큰 가정을 찾아줘"처럼 더 높은 수준의 질문을 던질 수 있습니다. 모델이 더 오래 생각할 수 있다면, 우리는 더 어려운 역할을 맡길 수 있습니다. 하지만 어려운 역할을 맡길수록 평가 기준도 함께 세워야 합니다. 반례를 만들라고 했으면 그 반례가 실제로 가능한지 확인해야 하고, 실험 설계를 제안하게 했으면 비용과 시간과 윤리적 조건을 따져야 합니다. 모델이 만든 계획이 멋있어 보인다고 해서 좋은 계획은 아닙니다. 좋은 계획은 실행 가능하고, 검증 가능하고, 실패했을 때 무엇을 배울 수 있는지 분명해야 합니다. AI가 계획을 잘 만들수록, 사람은 계획을 평가하는 능력을 더 키워야 합니다. 이것이 추론 모델 시대의 역설입니다. 모델이 똑똑해질수록 학생도 더 똑똑하게 물어야 합니다.

카파시는 에이전트와 사람의 감독을 함께 이야기합니다. 모델이 한 번 답하고 끝나는 것이 아니라, 도구를 쓰고, 중간 결과를 보고, 다시 계획을 고치고, 사람에게 확인을 받는 흐름이 중요해집니다 (링크). 추론 모델은 이런 에이전트형 작업 흐름의 두뇌 역할을 일부 맡을 수 있습니다. 예를 들어 문헌 검색 에이전트가 관련 논문을 찾고, 코드 에이전트가 데이터를 정리하고, 추론 모델이 결과의 논리적 빈틈을 점검하고, 사람이 마지막 판단을 내리는 식입니다. 이 그림은 흥미롭지만, 동시에 위험도 있습니다. 여러 에이전트가 서로의 오류를 이어받으면, 겉으로는 정교해 보이는 잘못된 결론이 만들어질 수 있습니다. 그래서 사람의 감독은 형식적인 승인 버튼이 아니라 실제 검토여야 합니다. 어떤 자료를 읽었는지, 어떤 코드가 실행되었는지, 어떤 가정이 들어갔는지 볼 수 있어야 합니다. 에이전트형 작업 흐름은 사람을 빼는 기술이 아니라, 사람이 더 높은 수준에서 감독하도록 만드는 기술이어야 합니다. 그렇지 않으면 빠른 자동화는 빠른 오류 증폭이 될 수 있습니다.

여기서 말하는 강화학습과 추론 모델은 학생에게 겁을 주기 위한 이야기가 아닙니다. 오히려 앞으로 공부가 더 흥미로워질 수 있다는 신호입니다. 이제 학생은 혼자 막막하게 문제 앞에 앉아 있지 않아도 됩니다. 모델에게 풀이의 첫 발판을 요청하고, 다른 접근법을 비교하고, 자신의 설명을 비판하게 할 수 있습니다. 그러나 그 자유는 책임과 함께 옵니다. AI가 제안한 사고 경로를 그대로 따라가는 사람은 쉽게 끌려갑니다. 반대로 AI가 만든 여러 경로를 놓고, 근거와 검증 가능성과 목적에 따라 고르는 사람은 더 넓게 생각할 수 있습니다. 대학에서 배워야 할 능력은 바로 이것입니다. 정답을 빨리 받는 능력이 아니라, 좋은 질문을 만들고, 가능한 답을 비교하고, 확인 가능한 근거로 자기 판단을 세우는 능력입니다. 추론 모델은 그 능력을 대신하지 않습니다. 다만 그 능력을 연습할 수 있는 더 넓은 장을 열어줍니다.

앞으로 추론 모델은 더 강해질 것입니다. 더 긴 문제를 붙잡고, 더 많은 도구를 쓰고, 더 복잡한 계획을 세울 수 있게 될 것입니다. 그러나 그 발전이 우리에게 요구하는 것은 손을 놓는 태도가 아니라, 더 좋은 감독의 태도입니다. 모델이 제안한 풀이를 읽을 수 있어야 하고, 모델이 사용한 자료를 확인할 수 있어야 하며, 모델이 놓친 가정을 물을 수 있어야 합니다. 의생명과학에서는 이 감독이 특히 중요합니다. 잘못된 수학 풀이 하나는 점수를 잃게 할 수 있지만, 잘못된 질병 해석이나 약물 설명은 훨씬 더 큰 오해를 만들 수 있습니다. 그래서 추론 모델을 배우는 일은 기술 감탄으로 끝나지 않습니다. 어떤 질문이 검증 가능한지, 어떤 결론은 아직 기다려야 하는지, 어떤 답은 사람의 판단을 통과해야 하는지 구분하는 훈련으로 이어져야 합니다. AI가 더 오래 생각할수록, 사람도 더 깊게 읽어야 합니다. 그것이 추론 모델 시대의 공부입니다.

1학년 학생에게 이 장의 목표는 강화학습의 수식을 이해하는 것이 아닙니다. 더 중요한 것은 모델이 문제를 더 오래 붙잡을 수 있게 되었을 때, 사용자도 더 좋은 질문을 던질 수 있어야 한다는 점입니다. "답을 알려줘"에서 "이 답이 틀릴 수 있는 지점을 찾아줘"로, "요약해줘"에서 "이 주장을 지탱하는 근거와 약한 부분을 나눠줘"로 질문이 바뀌어야 합니다. 이런 질문을 던질 수 있으면 AI는 정답 자판기가 아니라 사고를 넓히는 도구가 됩니다. 그리고 그 넓어진 사고를 다시 좁혀 근거 있는 판단으로 만드는 일은 여전히 학생의 몫입니다.

12장. 의생명과학 학생을 위한 LLM 사용 원칙

도구는 권위가 아니다

LLM은 의생명과학 학생에게 아주 강력한 도구가 될 수 있습니다. 낯선 개념을 처음 접할 때, 영어 자료나 논문을 읽기 전에 배경을 잡을 때, 코드 오류 앞에서 멈췄을 때, 실험 아이디어를 정리할 때 큰 도움을 줍니다. 에이전트까지 결합되면

할 수 있는 일은 더 넓어집니다. 파일을 읽고, 코드를 실행하고, 오류를 고치고, 결과를 요약하는 흐름이 한 번의 대화 안에서 이어질 수 있습니다.

용어 메모

LLM: 많은 글을 학습해 문장을 만들고 질문에 답하는 큰 언어 모델입니다.

에이전트(agent): 목표를 받아 파일 읽기, 코드 실행, 수정, 보고 같은 작업을 이어가려는 AI 시스템입니다.

하지만 이 도구는 권위가 아닙니다. 이 문장은 평범해 보이지만, 의생명과학에서는 아주 무겁습니다. 우리가 다루는 것은 유전자 이름, 질병명, 약물명, 변이 표기, 환자 데이터, 통계 결과, 실험 조건입니다. 이런 정보는 한 글자 차이로 의미가 달라질 수 있습니다. 매끄러운 설명과 믿을 만한 설명은 다릅니다. LLM은 문장을 자연스럽게 만드는 데 강하지만, 자연스러운 문장이 언제나 사실을 보증하지는 않습니다.

원칙을 하나씩 세어보면 많아 보이지만, 실제로는 세 가지 일로 모입니다. 자료를 지키고, 검증을 분리하고, 마지막 설명을 자기 자리로 되돌리는 일입니다.

묶음	포함되는 원칙	기억할 문장
자료 지키기	원문 확인, 자료 함께 주기, 원본 보존하기, 출처 기록(provenance) 남기기	무엇을 넣었고 어디서 왔는지 남깁니다.
검증 분리하기	질문 잘게 나누기, 계산은 실행으로 확인하기, 발견과 확인 나누기, 상관과 개입 구분하기	말로 만든 답과 확인된 결과를 섞지 않습니다.
자기 자리 지키기	평가 문제(benchmark)보다 실제 질문 보기, 자기 말로 다시 설명하기	마지막 판단과 설명은 학생에게 돌아옵니다.

학생들이 가장 자주 묻는 것은 “그러면 언제 멈춰야 하나요”입니다. AI를 쓰지 말아야 할 순간을 모두 외울 수는 없지만, 멈춤 신호는 기억할 수 있습니다.

멈춤 신호	왜 멈추는가	다음 행동
친구 이름, 학번, 연락처, 건강 정보가 들어 있다	개인정보가 외부 서비스로 나갈 수 있습니다.	익명화하거나 사용하지 않습니다.
환자, 질병, 약물, 변이에 관한 판단이다	잘못된 정보가 실제 불안이나 피해로 이어질 수 있습니다.	원자료, 전문가, 공식 데이터베이스로 확인합니다.
모델이 논문 제목이나 DOI를 자신 있게 제시한다	존재하지 않는 출처를 만들 수 있습니다.	링크가 열리는지, 본문 내용과 맞는지 직접 봅니다.
숫자 계산이나 표 처리 결과가 필요하다	언어 모델은 계산을 말로 그럴듯하게 틀릴 수 있습니다.	코드 실행 결과와 입력 파일을 확인합니다.
“수업 내부용”, “연구실 외부 공유 금지”라고 들은 자료다	편리함보다 자료 보안과 약속이 우선입니다.	담당자에게 사용 가능 여부를 먼저 묻습니다.

이 표는 겁을 주기 위한 금지 목록이 아닙니다. 오히려 AI를 오래 쓰기 위한 브레이크입니다. 자전거를 잘 타려면 페달을 밟는 힘만큼 멈출 줄도 알아야 합니다. AI 사용도 마찬가지입니다. 멈출 줄 아는 학생은 더 멀리 갈 수 있습니다.

그러므로 첫 번째 원칙은 원문으로 돌아가는 습관입니다. AI가 어떤 논문을 요약해주었다면, 그 논문이 실제로 존재하는지 확인해야 합니다. DOI가 맞는지, 저널과 연도가 맞는지, 모델이 말한 결과가 논문 본문에 정말 있는지 보아야 합니다. PubMed, Google Scholar, 학술지 페이지, 데이터베이스 원문을 확인하는 과정은 귀찮은 뒷정리가 아닙니다. AI 시대의 기본 문해력입니다. **특히 생명과 질병에 관한 주장은 반드시 원자료와 연결되어야 합니다.**

두 번째 원칙은 질문을 잘게 나누는 것입니다. “이 논문을 설명해줘”라는 질문은 너무 큽니다. 처음에는 편하지만, 답이 막연해지기 쉽습니다. “이 초록에서 연구 질문, 사용한 데이터, 핵심 결과, 한계를 나누어 설명해줘”라고 물으면 훨씬 낫습니다. “이 데이터 분석해줘”보다 “이 파일에서 group 열을 기준으로 value의 분포를 비교하고, 결측값 개수와 샘플 수를 먼저 보고해줘”가 낫습니다. 질문을 잘게 나누면 모델의 답도 확인하기 쉬워집니다. 어디서 틀렸는지 찾을 수 있기 때문입니다.

자료와 계산을 따로 확인하기

세 번째 원칙은 자료를 함께 주는 것입니다. 모델의 오래된 기억에 기대기보다, 논문 초록, 방법, 그림 설명(figure legend), 표, 분석 코드, 데이터 설명서를 문맥 창 안에 넣어주는 편이 안전합니다. 다만 자료를 넣을 때는 두 가지를 동시에 생각해야 합니다. 하나는 충분한 근거를 주는 일이고, 다른 하나는 넣으면 안 되는 자료를 지키는 일입니다. 환자 식별 정보, 공개하면 안 되는 연구 데이터, IRB나 공동연구 계약상 외부 서비스에 올리면 안 되는 파일은 넣지 않아야 합니다. 1학년 학생에게는 IRB가 아직 멀게 느껴질 수 있지만, 원칙은 지금도 가깝습니다. 친구의 이름과 학번이 들어간 표, 수업에서 내부용으로 받은 자료, 연구실에서 "밖에 올리지 말라"고 들은 파일은 편하다는 이유로 외부 서비스에 넣지 않아야 합니다. AI 사용 능력은 편리함을 얻는 기술이면서 동시에 정보 보호와 연구 윤리를 지키는 기술입니다.

용어 메모

그림 설명(figure legend): 논문 그림이 무엇을 보여주는지 설명하는 글입니다.

문맥 창: 모델이 지금 답변을 만들 때 한 번에 읽을 수 있는 입력 공간입니다.

IRB: 사람 대상 연구가 윤리적으로 설계되었는지 심의하는 위원회입니다.

네 번째 원칙은 **계산을 말이 아니라 실행으로 확인하는 것입니다.** LLM은 설명을 잘하지만, 숫자 세기나 복잡한 문자열 처리나 통계 계산에서 실수할 수 있습니다. 유전자 리스트의 교집합, 표의 행 개수, 평균과 표준편차, p-value 계산, 다중검정 보정처럼 확인 가능한 일은 코드로 실행하게 해야 합니다. "계산해줘"라고만 묻기보다 "코드를 보여주고 실행 결과를 확인해줘"라고 요구하는 습관이 필요합니다. 실행 로그, 입력 파일, 출력 파일이 남아야 나중에 다시 볼 수 있습니다.

용어 메모

p-value: 관찰한 차이가 우연만으로도 나올 수 있는지 따져볼 때 쓰는 통계값입니다.

다중검정 보정: 많은 비교를 한꺼번에 할 때 우연한 발견이 늘어나는 문제를 줄이는 절차입니다.

실행 로그: 어떤 코드가 어떤 순서로 돌았고 어떤 결과가 나왔는지 남긴 기록입니다.

원본과 기록을 지키기

다섯 번째 원칙은 원본을 보존하는 것입니다. 에이전트에게 파일을 맡길 때는 원본 데이터를 수정하지 말라고 분명히 말해야 합니다. 새 파일은 별도 폴더에 만들게 하고, 전처리 단계마다 무엇을 했는지 기록하게 하고, 삭제나 덮어쓰기는 하지 못하게 해야 합니다. 연구 데이터는 한 번 망가지면 복구하기 어렵습니다. 특히 여러 단계의 분석이 이어질 때는 어떤 파일이 원본이고 어떤 파일이 가공본인지 분명히 남겨야 합니다.

여섯 번째 원칙은 발견과 확인을 나누는 것입니다. 데이터를 처음 볼 때는 탐색이 필요합니다. 여러 그림을 그려보고, 의외의 패턴을 찾고, 후보 유전자를 넓게 살펴볼 수 있습니다. 이것은 발견용 분석입니다. 그러나 발견용 분석에서 눈에 띈 결과를 바로 결론으로 쓰면 위험합니다. 확인용 분석은 미리 정한 비교, 미리 정한 기준, 가능하면 독립 데이터나 다른 실험으로 다시 보는 과정입니다. 에이전트가 수십 가지 분석을 빠르게 돌릴 수 있는 시대에는 이 구분이 더 중요해집니다. 많이 돌릴수록 우연히 그럴듯한 결과를 만날 가능성도 커지기 때문입니다.

이 문제는 카파시가 말한 의도 구현(manifesting)과도 이어집니다. 사람이 에이전트에게 의도를 표현하고, 에이전트가 여러 작업을 수행하는 시대가 오면, 의도 자체가 작업의 방향을 정합니다 (링크). 좋은 의도는 좋은 루프를 만들 수 있지만, 나쁜 의도도 빠르게 실행될 수 있습니다. "진짜 차이가 있는지 확인하라"는 지시와 "내가 기대한 유전자가 나오게 하라"는 지시는 전혀 다릅니다. 후자는 확인편향을 자동화합니다.

생물학 연구에서 이 위험은 작지 않습니다. 비교 대상을 바꾸고, 어떤 값을 제외할지 바꾸고, 그래프 모양을 바꾸다 보면 언젠가 원하는 그림이 나올 수 있습니다. 예전에도 이런 유혹은 있었습니다. 그러나 에이전트는 그 유혹을 훨씬 빠르고 세련되게 실행할 수 있습니다. 그래서 에이전트 시대의 연구자에게 필요한 것은 단순히 분석을 많이 돌리는 능력이 아닙니다. 무엇을 돌려도 되고 무엇은 돌리면 안 되는지 먼저 정하는 규율입니다.

일곱 번째 원칙은 출처 기록(provenance)을 남기는 것입니다. 출처 기록은 어떤 결과가 어디서 왔는지에 대한 기록입니다. 어떤 데이터 파일을 썼는지, 어떤 코드가 실행되었는지, 어떤 패키지 버전(package version)을 사용했는지, 어떤 프롬프트를 주었는지, 어떤 중간 결과를 보고 다음 단계로 넘어갔는지 남겨야 합니다. 연구실 노트가 실험의 기억이라면, 에이전트 작업 로그는 디지털 연구의 기억입니다. 나중에 결과가 이상해 보일 때, 이 기록이 있어야 되돌아갈 수 있습니다.

용어 메모

출처 기록(provenance): 결과가 어떤 자료와 절차에서 나왔는지 남기는 기록입니다.

패키지 버전(package version): 코드가 사용한 프로그램 묶음의 버전입니다. 버전이 달라지면 결과가 달라질 수 있습니다.

이 원칙은 연구실에 들어간 뒤에야 필요한 것이 아닙니다. 1학년 수업 과제에서도 원본 파일과 가공 파일을 나누어 두는 습관은 중요합니다. 처음 받은 표를 그대로 두고, 정리한 표는 새 이름으로 저장하고, 어떤 열을 지웠는지 짧게 적어두는 것만으로도 나중에 훨씬 덜 헤맵니다. AI를 사용하면 이런 구분은 더 필요해집니다. 에이전트가 파일을 열고 정리하고 새 표를 만들 때, 사람이 중간 과정을 놓치기 쉽기 때문입니다. "결과가 잘 나왔으니 됐다"라고 생각하는 순간, 그 결과가 어떤 선택의 연속에서 나왔는지 사라질 수 있습니다. 과학에서 결과만 남고 과정이 사라지면, 다른 사람이 확인할 수도 없고 자신도 다시 설명할 수 없습니다. 좋은 AI 사용은 편리한 자동화가 아니라, 되짚을 수 있는 자동화여야 합니다.

점수보다 실제 질문을 보기

여덟 번째 원칙은 모델 평가를 겉점수만으로 보지 않는 것입니다. AI 모델은 평가 문제(benchmark)에서 높은 점수를 받을 수 있습니다. 그러나 의생명 연구의 실제 작업 흐름(workflow)은 시험 문제보다 훨씬 지저분합니다. 데이터 설명서가 불완전하고, 샘플 이름이 이상하고, 실험 조건이 애매하고, 논문 간 결론이 서로 다를 수 있습니다. 어떤 모델이 특정 평가 문제에서 잘했다고 해서, 여러분의 데이터와 질문에서도 잘한다는 뜻은 아닙니다. 점수표를 볼 때도 결국 물어야 할 것은 이 모델이 내 자료, 내 질문, 내 검증 조건에서 어디까지 버티는가입니다.

용어 메모

평가 문제(benchmark): 여러 모델의 성능을 비교하기 위해 정해 둔 시험 문제나 평가 기준입니다.

작업 흐름(workflow): 자료를 준비하고, 분석하고, 검토하고, 결과를 남기는 전체 흐름입니다.

예를 들어 어떤 바이오 AI 모델이 아주 많은 세포나 유전체 자료로 학습되었다고 해도, 그것만으로 충분하지 않습니다. 그 모델이 실제로 어떤 질문에 도움이 되는지, 새 자료에서도 비슷하게 작동하는지, 간단한 방법보다 정말 나은지 따로 물어야 합니다. 학생 수준에서는 이렇게 생각해도 좋습니다. AI 회사가 발표한 점수표만 보고 도구를 고르지 말고, 내가 읽는 논문 초록, 내가 받은 수업 표, 내가 작성해야 하는 발표문에서 작은 시험을 해보아야 합니다. "이 모델이 세상에서 제일 좋은가"보다 "내 과제의 이 단계에서 믿고 쓸 수 있는가"가 더 가까운 질문입니다. 최근 바이오 AI 분야에서는 큰 모델이 항상 좋은 결과를 내는 것은 아니라는 논의도 계속됩니다. 크기는 출발점일 뿐이고, 검증은 별개의 일입니다.

아홉 번째 원칙은 상관관계와 개입을 구분하는 것입니다. 어떤 유전자들이 함께 발현된다는 사실은 중요할 수 있지만, 그것만으로 하나가 다른 하나를 조절한다고 말할 수는 없습니다. 질병군에서 특정 경로가 높게 보인다고 해서, 그 경로가 질병의 원인이라고 바로 말할 수도 없습니다. 생명과학에서 치료와 기전으로 가려면 "함께 보인다"를 넘어 "바꾸면 무엇이 달라지는가"를 물어야 합니다. 이 질문이 개입 실험(perturbation)이고, 더 나아가 반사실적 사고(counterfactual thinking)의 출발점입니다. AI가 제안한 후보도 이 구분을 통과해야 합니다.

용어 메모

상관관계: 두 일이 함께 움직이는 것처럼 보이는 관계입니다.

개입 실험(perturbation): 시스템에 일부러 변화를 주어 반응을 보는 일입니다.

반사실적 사고(counterfactual thinking): "만약 이 조건만 바꾸었다면 결과가 달라졌을까"를 묻는 사고방식입니다.

자기 말로 돌아오기

열 번째 원칙은 자기 말로 다시 설명하는 것입니다. AI가 만든 설명을 읽고 고개를 끄덕였다고 해서 배운 것은 아닙니다. 모델이 만든 코드를 실행했다고 해서 분석을 이해한 것도 아닙니다. 친구에게 말로 설명할 수 있어야 합니다. "이 비교가 왜 필요한지", "이 그림의 x축과 y축이 무엇인지", "이 p-value가 무엇을 뜻하지 않는지", "이 분석의 가장 큰 한계가 무엇인지"를 자신의 문장으로 말할 수 있어야 합니다. 설명하지 못하는 결과는 아직 자기 것이 아닙니다.

이 태도는 두려움과도 다르고 맹신과도 다릅니다. AI를 멀리하면 앞으로의 연구 환경을 놓칠 수 있습니다. 반대로 AI를 권위처럼 받아들이면 생명과학에서 가장 중요한 검증 절차를 잃을 수 있습니다. 의생명과학 학생에게 필요한 것은 그 사이의 길입니다. AI를 곁에 두되, 질문과 기준과 책임은 스스로 갖는 길입니다.

카파시는 좋은 tutor가 학생에게 지식으로 올라가는 경사로를 만들어준다고 말합니다 (링크). LLM은 그런 경사로의 일부가 될 수 있습니다. 어려운 논문을 처음 읽게 해주고, 복잡한 개념을 낮은 곳에서부터 다시 설명해주고, 코드 앞에서

포기하지 않게 도와줄 수 있습니다. 그러나 경사로를 올라가는 일은 여전히 학생의 일입니다. AI가 손을 잡아줄 수는 있지만, 어느 방향으로 가고 있는지 묻는 사람은 학생 자신이어야 합니다.

이런 원칙들은 처음에는 많아 보일 수 있습니다. 그러나 실제로는 하나의 태도로 모입니다. AI가 만든 결과를 연구 결과처럼 대하지 말고, 연구 과정의 초안으로 대하는 것입니다. 초안은 고칠 수 있고, 의심할 수 있고, 더 좋은 근거를 붙일 수 있습니다. 초안을 빨리 얻는 것은 큰 장점입니다. 다만 **초안이 빨라졌다고 해서 검토가 짧아져서는 안 됩니다.** 오히려 초안이 많아질수록 무엇을 버릴지 판단하는 시간이 중요해집니다.

학생이 이 태도를 연습하는 가장 좋은 방법은 작은 작업에서부터입니다. 논문 한 편을 읽을 때 AI에게 먼저 요약을 부탁하고, 그다음 원문을 보며 맞는지 표시해보십시오. 유전자 리스트 하나를 분석할 때 AI에게 코드를 만들게 하고, 그 코드가 어떤 열을 읽고 어떤 기준으로 필터링하는지 한 줄씩 설명해보십시오. 발표문을 만들 때 AI가 쓴 문장을 그대로 쓰지 말고, 자신이 이해한 만큼 다시 고쳐보십시오. 이 작은 연습들이 쌓이면, AI와 함께 일하면서도 자기 판단을 잃지 않게 됩니다.

의생명 연구에서 이 태도는 결국 사람을 향합니다. 데이터는 추상적인 숫자처럼 보이지만, 많은 경우 그 뒤에는 환자, 기증자, 실험동물, 세포주, 연구자의 시간이 있습니다. 환자 데이터를 다룰 때는 개인정보와 동의의 문제가 있고, 실험 데이터를 다룰 때는 재현성과 정직성의 문제가 있으며, 질병에 대해 설명할 때는 잘못된 정보가 누군가에게 실제 불안을 줄 수 있습니다. AI가 말을 쉽게 만들어줄수록, 그 말이 도달할 사람을 생각해야 합니다. 좋은 과학 문장은 빠른 문장이 아니라 책임 있는 문장입니다.

또 하나 기억할 것은 AI가 모든 학생을 같은 방식으로 돕지 않는다는 점입니다. 어떤 학생에게 AI는 낯선 영어 자료나 논문을 처음 열게 해주는 문이 될 수 있고, 어떤 학생에게는 코딩 공포를 낮춰주는 친구가 될 수 있습니다. 반대로 어떤 학생에게는 너무 쉽게 답을 주기 때문에 스스로 생각할 시간을 빼앗는 도구가 될 수도 있습니다. 그래서 AI 사용법에는 정답이 하나만 있지 않습니다. 중요한 것은 내가 어디서 도움을 받고, 어디서 스스로 멈추어 생각해야 하는지 알아가는 일입니다.

이 책의 마지막 사용 원칙은 그래서 간단합니다. **AI에게 맡기되, 믿기 전에 확인하십시오. 빠르게 만들되, 천천히 읽으십시오. 많은 답을 얻되, 자기 질문을 잃지 마십시오.** 의생명과학에서 AI는 강력한 도구가 될 것입니다. 그러나 도구가 강해질수록, 그것을 쓰는 사람의 기준도 더 단단해야 합니다.

이 기준은 하루아침에 생기지 않습니다. 처음에는 모델이 써준 답을 읽고 어디가 틀렸는지 잘 보이지 않을 수 있습니다. 그것은 당연합니다. 그래서 작은 단위에서 연습해야 합니다. 한 문단을 읽고 근거를 찾고, 한 줄의 코드를 읽고 무슨 일을 하는지 설명하고, 작은 표 하나를 보고 어떤 계산이 필요한지 말해보는 식입니다. 이런 훈련은 느려 보이지만, 나중에 큰 분석과 복잡한 논문을 다룰 때 학생을 지켜줍니다. AI가 만들어준 결과를 검토하는 눈은 결국 기초 개념, 데이터를 읽는 연습, 글 읽기 습관에서 나옵니다. 그러므로 AI를 잘 쓰고 싶다면 AI만 배워서는 안 됩니다. 생물학도 배워야 하고, 통계도 배워야 하며, 글도 읽어야 합니다. 도구가 강해질수록 기초가 덜 필요한 것이 아니라, 기초가 다른 방식으로 더 중요해집니다.

마지막으로, AI와 함께 공부하는 일은 혼자만의 기술이 아닙니다. 수업, 연구실, 회사, 기관마다 함께 기준을 만들어야 합니다. 어떤 자료를 AI에 넣어도 되는지, 어떤 과제와 업무에서 AI 사용을 허용할 것인지, AI가 만든 문장을 어떻게 표시할 것인지, 코드와 결과를 어떻게 검토할 것인지 함께 이야기해야 합니다. 학생이나 실무자가 몰래 쓰고, 교수자나 책임자가 막연히 금지하는 방식으로는 좋은 문화를 만들기 어렵습니다. AI는 이미 공부와 일의 환경 안으로 들어왔습니다. 그렇다면 필요한 것은 숨기는 문화가 아니라, 책임 있게 드러내고 점검하는 문화입니다. 이 책이 바라는 것도 그 방향입니다. 학생이 AI를 두려워하지 않고, 동시에 가볍게 여기지도 않으며, 자기 질문과 근거와 책임을 가지고 사용할 수 있기를 바랍니다. 그렇게 사용할 때 LLM은 지름길이 아니라 더 넓은 공부와 일의 길이 됩니다.

앞으로 여러분이 연구자가 되든, 바이오 기업이나 병원, 공공기관, 교육·콘텐츠 현장으로 가든, 어느 직종에서나 AI는 거의 늘 곁에 있을 것입니다. 논문을 읽고, 데이터를 다루고, 발표문을 만들고, 동료에게 설명하고, 새로운 자료를 검토하는 일은 직함이 달라도 계속 남습니다. 어떤 날에는 AI가 막힌 코드를 풀어주고, 어떤 날에는 어려운 문장을 낮은 곳으로 내려주고, 어떤 날에는 여러분이 놓친 반례를 보여줄 것입니다. 그러나 AI가 곁에 있다는 사실이 여러분의 고유한 시선을 대신하지는 못합니다. 어떤 문제를 오래 붙잡고 싶은지, 어떤 생명현상이 이상하게 느껴지는지, 어떤 데이터가 마음에 걸리는지, 어떤 설명이 충분하지 않은지 알아차리는 일은 여러분 안에서 자라야 합니다. 그래야 AI가 만든 수많은 답 중에서 중요한 것을 고를 수 있습니다. 의생명과학은 결국 살아 있는 세계를 이해하려는 학문입니다. 그 세계는 텍스트보다 복잡하고, 모델보다 느리고, 때로는 우리가 만든 설명을 거부합니다. 그래서 AI와 함께하더라도 겸손해야 합니다. 좋은 도구를 들었으니 더 빨리 단정하는 것이 아니라, 더 깊이 확인할 수 있어야 합니다. 그것이 이 책이 마지막에 남기고 싶은 태도입니다.

이 원칙들은 한 번 읽고 끝나는 규칙표가 아닙니다. 수업에서 작은 표를 다룰 때, 논문 초록을 처음 읽을 때, 발표문을 만들 때, 일터에서 첫 데이터 파일을 받을 때마다 다시 돌아와야 하는 기준입니다. AI는 계속 바뀔 것이고, 오늘 쓰는 모델 이름도 몇 년 뒤에는 낡아 있을 수 있습니다. 이 책의 본문에서 언급한 모델 이름들도 2025–2026년 무렵의 사례로 읽으면

충분합니다. 하지만 원문으로 돌아가기, 계산을 실행으로 확인하기, 원본을 지키기, 기록을 남기기, 자기 말로 설명하기 같은 습관은 쉽게 낡지 않습니다. 도구가 바뀌어도 이 습관은 학생을 지켜줍니다.

그래서 이 장의 마지막 말은 금지가 아니라 초대에 가깝습니다. AI를 쓰는 일은 과제를 더 빨리 끝내는 기술을 익히는 데서 멈추지 않습니다. 지금 우리 사회가 마주한 건강, 돌봄, 환경, 격차, 교육의 문제를 더 정확히 읽고 더 책임 있게 풀어보라는 초대이기도 합니다. "실천하는 지성, 생각하는 리더, 사회에 힘이 되는 대학"이라는 말은 이런 자리에서 구체적인 의미를 얻습니다. 개척하는 지성은 낯선 도구를 먼저 써보는 호기심에 머물지 않습니다. 그 도구를 누구를 위해, 어떤 근거 위에서, 어떤 책임으로 쓸 것인지 묻는 태도입니다. 그것이 고려대학교가 학생들에게 기대하는 힘이고, 앞으로의 교육이 향해야 할 방향입니다. **AI를 쓰십시오. 다만 더 정직하게 읽고, 더 분명하게 묻고, 더 책임 있게 남기십시오. 그리고 그 힘을 우리 사회의 어려운 문제를 푸는 데 쓰십시오.**

용어 지도와 표기 약속

이 책에는 영어 약어와 한국어 풀이가 함께 나옵니다. 의생명과학 학생은 앞으로 논문과 도구 설명에서 영어 용어를 계속 만나게 되므로 영어를 완전히 없애지는 않겠습니다. 대신 처음 등장할 때는 가능한 한 한국어 풀이와 영어를 함께 적고, 그 뒤에는 한국어 표기를 주로 쓰겠습니다. 예를 들어 에이전트(agent), 토큰(token), 사전학습(pre-training), 지도 미세조정(supervised fine-tuning)은 첫 등장 뒤에는 에이전트, 토큰, 사전학습, 지도 미세조정으로 씁니다. LLM, RAG, RLHF, SFT처럼 약어 자체로 널리 쓰이는 말은 영어 약어를 유지합니다. Transformer, Common Crawl, FineWeb 같은 고유명사도 영어로 남깁니다. BRCA1, TP53 같은 유전자와 단백질 이름은 표기 자체가 지식의 일부이므로 바꾸지 않습니다.

이 약속은 독자를 영어에서 떼어놓기 위한 것이 아니라, 처음 만나는 용어 앞에서 숨을 고르게 하기 위한 장치입니다. 영어 이름을 알아야 나중에 논문과 도구 문서를 찾을 수 있고, 한국어 풀이가 있어야 그 이름이 지금 무엇을 가리키는지 놓치지 않습니다. 한 용어가 여러 장에서 다시 나와도 매번 길게 정의하지는 않겠습니다. 대신 이 페이지를 돌아올 수 있는 작은 지도처럼 두겠습니다.

용어를 읽을 때도 힘을 나누어 쓰면 좋습니다. 토큰, 문맥 창, 프롬프트처럼 책 전체에서 자주 만나는 말은 처음에 짧은 풀이만 붙잡아도 됩니다. 신경망, attention, embedding, 강화학습처럼 나중에 전공 수업에서 깊게 배울 말은 여기서 완전히 익히려 하기보다 "대략 어떤 역할을 하는 장치인가"를 기억하면 됩니다. Transformer 같은 이름은 분야의 표지판에 가깝습니다. 표지판을 본다고 그 도시 전체를 알아야 하는 것은 아닙니다. 낯선 영어가 많이 보일 때는 지금 붙잡을 말, 나중에 깊게 배울 말, 이름만 알고 지나가도 되는 말을 구분하는 것만으로도 읽기가 한결 가벼워집니다.

문장 안에서 모르는 말이 나오면 세 가지를 물어보십시오. 이 말이 지금 문장을 이해하는 데 꼭 필요한가. 한국어로 바꾸면 어떤 일인가. 나중에 원문이나 검색에서 다시 찾으려면 어떤 영어 이름을 알아두어야 하는가. 이 세 질문은 중학생을 가르치는 선생님에게도, 과학책을 읽는 일반 독자에게도, 다큐멘터리 원고를 쓰는 작가에게도 유용합니다. 모르는 말을 없애는 것이 목표가 아닙니다. 모르는 말 앞에서 멈추고, 뜻과 역할과 확인 방법을 나누어 보는 것이 목표입니다.

용어	이 책에서의 뜻
LLM	많은 글을 학습해 다음 토큰을 예측하고 문장을 생성하는 큰 언어 모델입니다.
ChatGPT	LLM을 대화창 형태로 만나는 대표적인 서비스입니다.
토큰(token)	모델이 글을 읽고 쓸 때 사용하는 작은 글자 조각입니다. 단어와 정확히 같지 않습니다.
토큰화(tokenization)	문장을 토큰의 줄로 바꾸는 과정입니다.
매개변수(parameter)	모델 안에 저장되어 학습 중 조정되는 수많은 숫자입니다.
문맥(context)	모델이 지금 답할 때 눈앞에 놓고 참고하는 입력 자료입니다.
문맥 창(context window)	모델이 한 번에 읽을 수 있는 문맥의 길이와 공간입니다.
프롬프트(prompt)	사용자가 모델에게 주는 질문, 자료, 지시문입니다.
신경망(neural network)	많은 작은 계산 단위가 층층이 연결되어 입력을 출력으로 바꾸는 수학적 구조입니다. 생물학적 신경세포와 같지는 않습니다.
Transformer	토큰들 사이의 관계를 attention으로 계산하는 현대 LLM의 대표 구조입니다.

용어	이 책에서의 뜻
attention	문장 안에서 어떤 토큰을 더 참고할지 계산하는 장치입니다.
embedding	토큰을 여러 숫자의 묶음으로 바꾼 표현입니다.
사전학습(pre-training)	모델이 많은 글을 먼저 읽으며 언어와 지식의 배경 패턴을 배우는 단계입니다.
베이스 모델(base model)	질문에 친절히 답하기 전, 글의 흐름을 이어 쓰는 능력을 먼저 배운 모델입니다.
어시스턴트(assistant)	사용자의 질문에 도움이 되는 답을 하도록 추가 훈련된 모델의 사용 형태입니다.
지도 미세조정(SFT)	좋은 질문과 답변 예시를 보여주며 어시스턴트다운 행동을 배우게 하는 단계입니다.
강화학습(RL)	좋은 결과로 이어진 행동을 더 자주 하도록 훈련하는 방법입니다.
RLHF	사람의 선호를 이용해 모델 답변을 더 낫게 조정하는 훈련 방식입니다.
환각(hallucination)	모델이 사실처럼 보이는 틀린 내용을 만들어내는 현상입니다. 이 책에서는 분야에서 널리 쓰이는 영어도 함께 남깁니다.
RAG	관련 자료를 먼저 찾아 문맥에 넣고, 그 자료를 바탕으로 답하게 하는 방식입니다.
에이전트(agent)	목표를 받아 파일 읽기, 코드 실행, 수정, 보고 같은 여러 단계를 이어가려는 AI 시스템입니다.
추론 모델(reasoning model)	어려운 문제를 더 오래 붙잡고 단계적으로 풀도록 훈련된 모델입니다.
도구 사용(tool use)	검색, 코드 실행, 데이터베이스 조회 같은 외부 기능을 모델과 함께 쓰는 일입니다.
출처 기록(provenance)	결과가 어떤 자료와 절차에서 나왔는지 남기는 기록입니다.
개입 실험(perturbation)	세포나 시스템에 일부러 변화를 주어 반응을 보는 일입니다.
비교 기준(baseline)	새 방법이 정말 나은지 비교하기 위해 두는 기본 방법입니다.

참고와 인용

이 책은 안드레이 카파시의 공개 강의와 인터뷰를 중요한 출발점으로 삼습니다. 그렇지만 이 책은 강의록이나 인터뷰를 그대로 옮긴 번역본이 아닙니다. 카파시가 LLM과 에이전트를 설명할 때 사용하는 큰 흐름, 직관, 몇몇 예시를 바탕으로 삼되, 고려대학교 보건과학대학 바이오시스템의과학부 1학년 학생들이 읽을 수 있도록 한국어 산문으로 다시 구성한 해설서입니다.

본문에서 카파시의 설명 순서나 특정 예시를 직접 따라가는 곳에는 가까운 자리에 출처를 표시합니다. 본문 인용 링크는 가능한 한 참고문헌 페이지가 아니라 영상의 해당 시점으로 바로 연결합니다. 직접 인용은 필요한 경우에만 짧게 사용하고, 대부분의 설명은 의생명과학 학생을 위한 맥락에 맞추어 다시 씁니다.

본문의 영상 인용은 독자의 읽기 흐름을 끊지 않기 위해 `([ ](...))`처럼 앞말과 한 칸 띄어 표시합니다. 시간 정보는 문장 안에 쓰지 않고, YouTube URL의 `t=` 값 안에만 남깁니다.

Deep Dive into LLMs like ChatGPT

Karpathy, A. (2025, February 5). Deep Dive into LLMs like ChatGPT [Video lecture]. YouTube.

https://www.youtube.com/watch?v=7xTGNNLPyMI

이 책에서 LLM의 전체 제작 과정, 토큰화, 사전학습, Transformer, assistant로의 후속 훈련, hallucination, 문맥 창, 도구 사용, reasoning model, reinforcement learning을 설명할 때 가장 자주 참고하는 자료입니다.

주요 연결 지점:

- 00:00:00: 전체 강의의 문제의식 (링크)
- 00:01:47: FineWeb과 Common Crawl (링크)
- 01:56:21: “models need tokens to think” (링크)
- 02:11:10: 강화학습을 학교 공부에 비유하는 대목 (링크)
- 02:33:12: reasoning model (링크)
- 03:09:26: 에이전트와 사람의 감독 (링크)
- 03:21:24: ChatGPT 입력창 뒤에서 일어나는 일의 요약 (링크)

Andrej Karpathy – “We’re summoning ghosts, not building animals”

Patel, D. (2025, October 17). Andrej Karpathy – “We’re summoning ghosts, not building animals” [Interview with Andrej Karpathy]. YouTube.

https://www.youtube.com/watch?v=lXUZvyajciY

이 책에서 “에이전트의 해”가 아니라 “에이전트의 10년”이라는 시간 전망, AI가 생물학적 지능과 다른 방식으로 만들어진다는 관점, AI 시대의 교육과 tutor, 지식으로 가는 ramp를 설명할 때 참고하는 인터뷰입니다.

주요 연결 지점:

- 00:00:00: “에이전트의 해”가 아니라 “에이전트의 10년”이라는 시간 전망 (링크)
- 02:00:49: 좋은 tutor의 역할 (링크)
- 02:02:34: 지식으로 올라가는 ramp (링크)

Skill Issue: Andrej Karpathy on Code Agents, AutoResearch, and the Loopy Era of AI

No Priors. (2026, March 20). Skill Issue: Andrej Karpathy on Code Agents, AutoResearch, and the Loopy Era of AI [Interview with Andrej Karpathy]. YouTube.

https://www.youtube.com/watch?v=kwSVtQ7dziU&t=69s

이 책에서 vibe coding, 의도 구현(manifesting), AI 에이전트, AutoResearch, 연구 자동화, 에이전트형 작업 흐름을 생각할 때 참고하는 인터뷰입니다.

주요 연결 지점:

- 00:00:00: code라는 동사와 의도 구현(manifesting) (링크)
- 00:03:40: 여러 에이전트를 통한 macro action (링크)
- 00:16:11: AutoResearch의 동기와 목표 (링크)

Andrej Karpathy

Karpathy, A. (n.d.). Andrej Karpathy.

https://karpathy.ai/

카파시의 이력과 교육 활동을 소개할 때 참고한 개인 홈페이지입니다.

더 읽을거리

Training language models to follow instructions with human feedback

Ouyang, L., Wu, J., Jiang, X., et al. (2022). Training language models to follow instructions with human feedback. arXiv.

https://arxiv.org/abs/2203.02155

7장에서 사람 labeler, labeling instruction, RLHF가 assistant의 말투와 행동을 바꾸는 과정을 더 깊게 보고 싶을 때 읽을 수 있는 논문입니다.

The Llama 3 Herd of Models

Dubey, A., Jauhri, A., Pandey, A., et al. (2024). The Llama 3 Herd of Models. arXiv.

https://arxiv.org/abs/2407.21783

8장에서 Llama 계열 모델과 hallucination 완화, post-training, 안전성 평가를 더 자세히 살펴보고 싶을 때 참고할 수 있는 기술 보고서입니다.