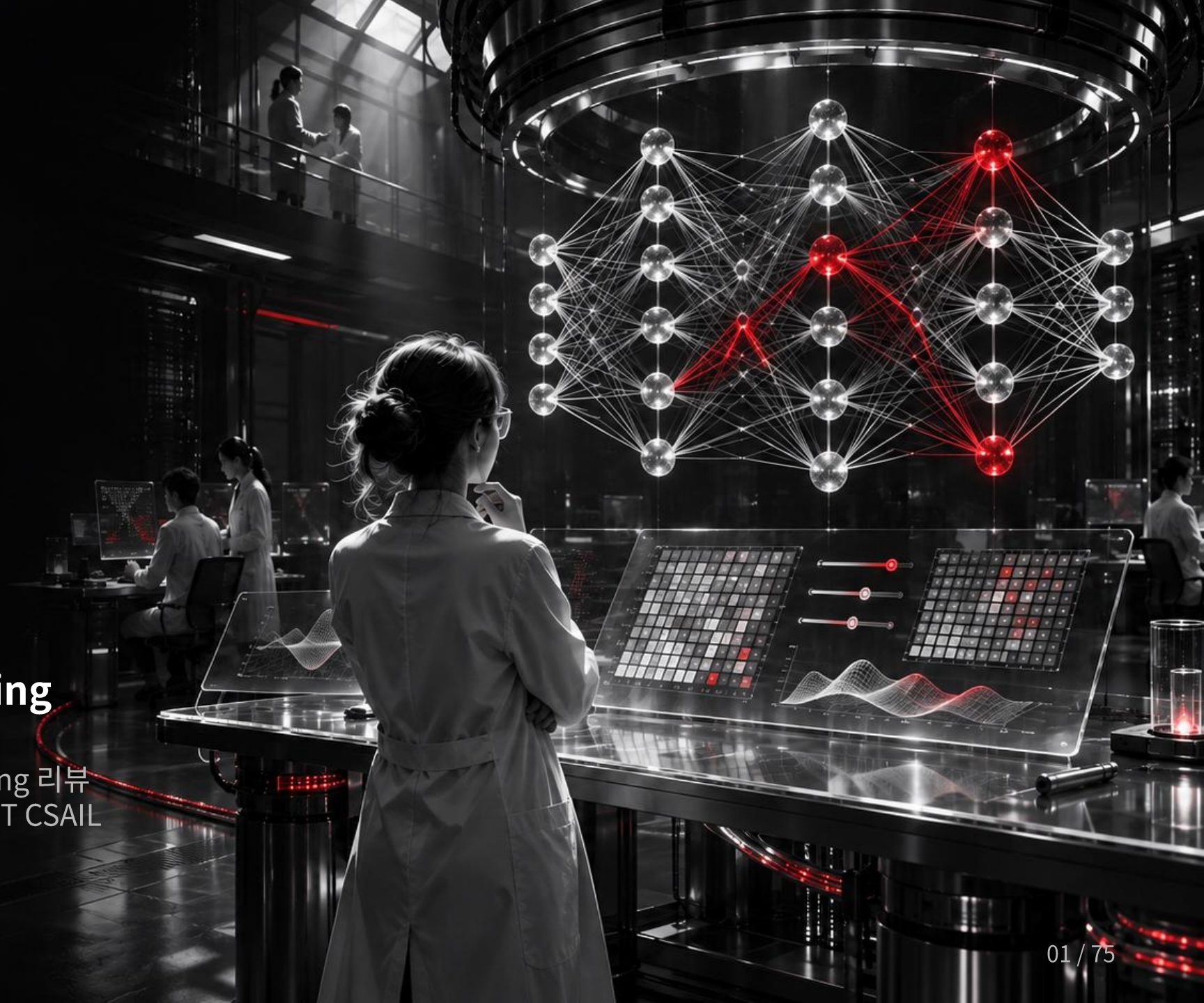
LAB SEMINAR

가중치를 읽고, 쓰는 언어 모델.



Weight-Space Programming

Sparks of Weight-Space Programming 리뷰
Takehi · Teoh · Zhang · Khattab, MIT CSAIL

DeepShark Lab · 학부생 전범기

발표 순서

핵심 개념과 실험 결과,
공개 로그를 살펴본다.



01 리뷰 대상 소개 03–05

02 가중치 해석 실험 리뷰 06–27

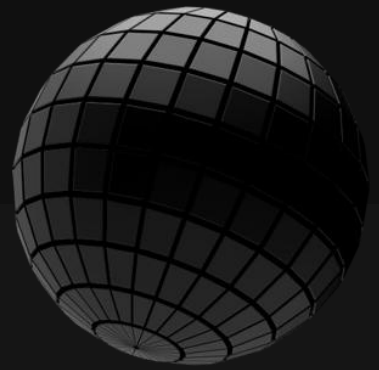
03 정책 수정·작성 실험 리뷰 28–41

04 대규모 모델 실험 리뷰 42–71

05 결론 및 질의응답 72–75

Sparks of Weight-Space Programming

리뷰 대상
2026.11.02 공개



LLMs can natively read & write neural network weights

Rikiya Takehi · Jayden Teoh
Alex Zhang · Omar Khattab
MIT CSAIL

연구 블로그의 초기 실험 보고

8개 주요 실험과 Gemma 추가 실험
원본 예시·공개 로그·평가 조건·한계

원문과 공개 로그

LLM이 신경망 가중치를 읽고 쓴다

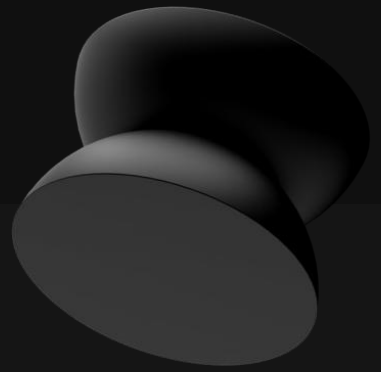
LLM은 신경망 내부의 가중치를 해석하고 직접 수정할 수 있음을 보여주며,

기존 학습 방식과 다른 새로운 모델 제어 가능성을 제시한다.



읽고,
고치고,
개선한다

범용 LLM이 수행하는
세 가지 작업



01

READ
학습 과제와
내부 기능 추정

실험 1–3, 7

02

FIX
실패 원인 진단과
일부 연결 수정

실험 4

03

IMPROVE
가중치를 작성·조정해
과제 성능 개선

실험 5–6, 8

01. READING WEIGHTS

가중치에 남은 행동의 단서

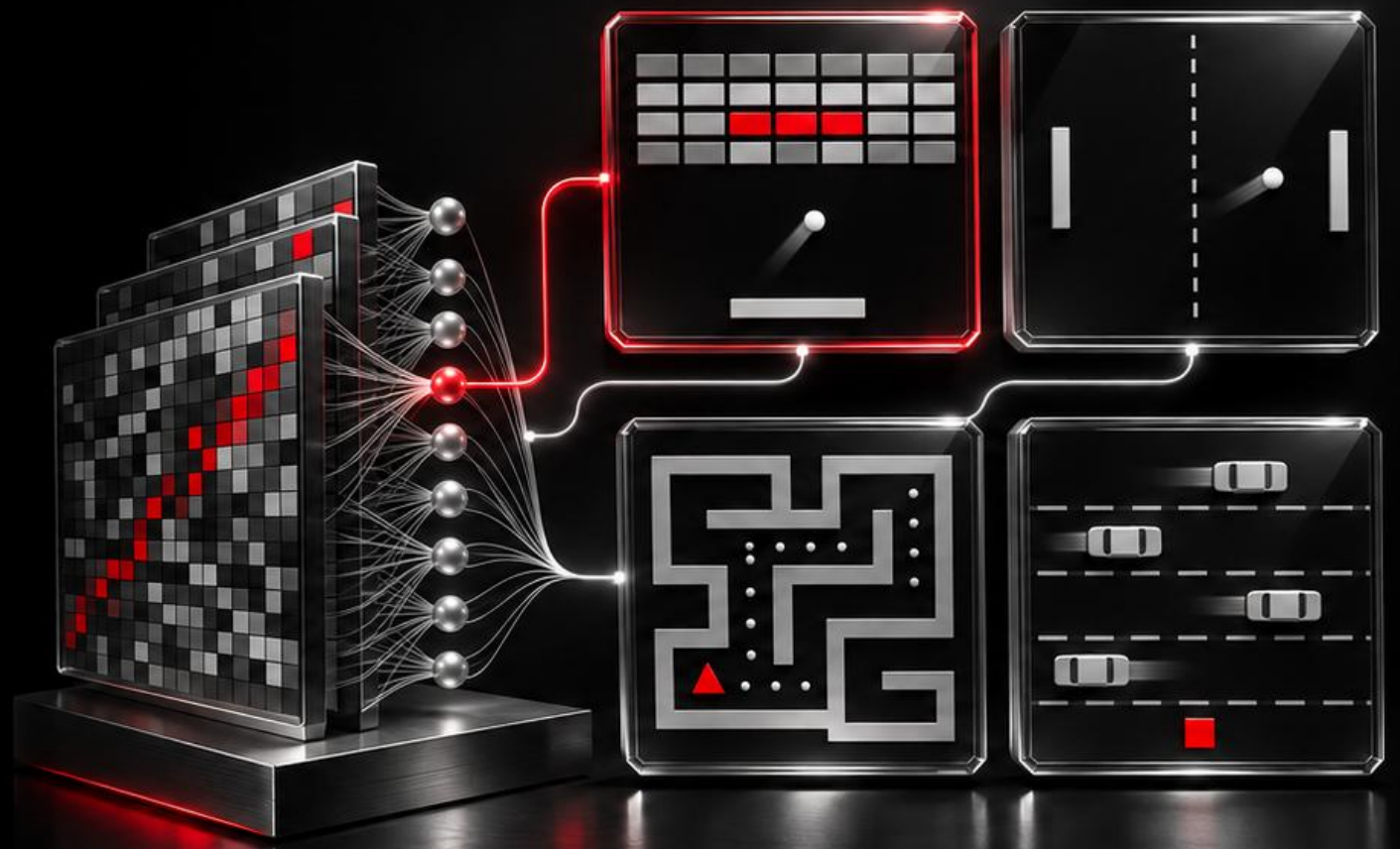
소형 정책 · 확산 모델 · 문자 언어 모델

EXPERIMENT 01 · READING

가중치만 보고, 게임을 알아볼 수 있을까

게임 이름과 후보 목록을 숨긴 채
가중치로 학습한 게임을 추정

제공 가중치·입출력 의미
허용 도구 없음 · 한 번의 응답
금지 코드 실행·계산기



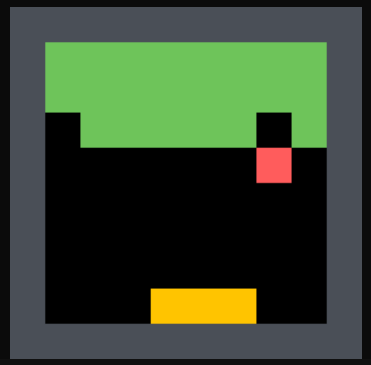
실험 설명용 개념 이미지

같은 구조, 서로 다른 게임

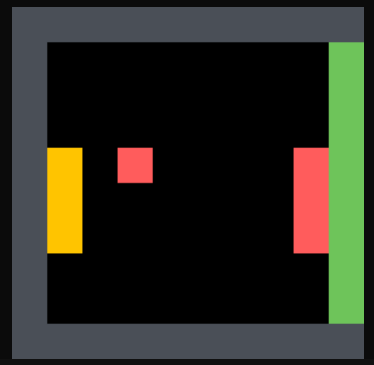
게임명과 후보 목록은 숨기고
입출력 의미와 가중치를 제공



Breakout



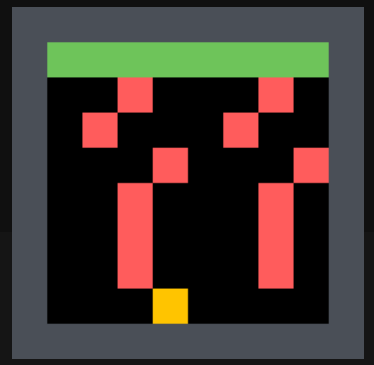
Pong



Pac-Man



Freeway



500 → 32 → 32 → 5, tanh · 17,253개 파라미터

실험 1 설정 가중치만으로 게임 식별

게임	자체 제작 격자형 Breakout · Pong · Pac-Man · Freeway 네 게임 모두 같은 격자와 행동 목록 사용
분석 대상 정책망	모든 게임에 동일한 MLP 500→32→32→5, tanh 가중치 17,253개
정책망 입력 (게임 관측값)	전체 10×10칸: 행 · 열 0–9, 바깥 테두리는 벽 플레이 영역: 내부 8×8칸(행 · 열 1–8) 각 칸의 5개 채널은 0 또는 1로 표시 에이전트 · 이동 물체 · 보상 목표 · 장애물 · 이전 프레임
정책망 출력	행동 0–4: 0 대기(no-op), 1 위, 2 아래, 3 왼쪽, 4 오른쪽 행동별 로짓 5개를 출력하고 softmax 확률에 따라 행동 선택
정책망 학습	PPO로 10억 step 학습, 모든 게임에 동일한 설정 적용 게임별 3개 seed 중 최고 성능 모델 선택
Astra 입력 (분석 자료)	전체 가중치 + 관측 · 행동 · 신경망 설명 + 게임 추정 질문

출처: 원문 실험 1 Setup / 좌표 · 채널 값 · 행동 번호: 공개 초기 프롬프트

Astra 입력 예시 ① 관측 정보

10×10 격자 · 행·열 0–9 · 바깥 테두리는 벽

플레이 영역은 내부 8×8칸(행·열 1–8)이며, 각 칸의 채널 값은 0 또는 1이다.

채널	이름	표현하는 정보
0	에이전트	플레이어가 직접 제어하는 셀
1	이동 물체	에이전트 이외에 움직이는 객체
2	보상 목표	보상을 제공하는 목표물
3	장애물	진입 불가 또는 진입 시 종료되는 셀 외곽 벽도 포함
4	이전 프레임	직전 단계에 채널 0 또는 1이었던 셀

500개 숫자로 펼쳐 입력

입력 번호	행 · 열 · 채널
0	0행 · 0열 · 채널 0
4	0행 · 0열 · 채널 4
5	0행 · 1열 · 채널 0
499	9행 · 9열 · 채널 4

실제 입력 프롬프트의 한국어 요약 · GitHub Breakout prompt.txt

Astra 입력 예시 ② 행동과 신경망

5개 행동의 확률에서 하나를 샘플링

번호	행동	방향
0	No-op	아무 행동 안 함
1	Up	위 · 0행 방향
2	Down	아래 · 9행 방향
3	Left	왼쪽 · 0열 방향
4	Right	오른쪽 · 9열 방향

로짓에 softmax를 적용해 확률을 얻는다.

MLP 500 → 32 → 32 → 5

$$h_1 = tanh(xW_1 + b_1)$$

W_1: 500×32 b_1: 32

$$h_2 = tanh(h_1W_2 + b_2)$$

W_2: 32×32 b_2: 32

$$logits = h_2W_3 + b_3$$

W_3: 32×5 b_3: 5

17,253개 파라미터 · PPO로 게임 보상을 최대화하도록 학습

실제 입력 프롬프트의 한국어 요약 · GitHub Breakout prompt.txt

Astra 입력 예시 ③ 질문과 답변 형식

“이 신경망은 어떤 게임을 플레이하도록 훈련되었는가?”

제공된 가중치로 추론 · 코드 실행·계산기·기타 도구 금지 · 답변 기회는 한 번

01	관찰	채널별 가중치의 크기·공간 패턴, 출력층 가중치와 편향을 분석 판단의 근거가 되는 실제 가중치 값을 인용
02	추론	관찰과 맞는 게임과 배제할 게임을 제시하고 그 이유를 설명
03	게임	추정한 게임을 최대한 구체적으로 제시 유명 게임과 유사하다면 해당 이름도 명시
04	게임 설명	에이전트가 하는 행동과 보상을 얻는 조건을 1–2문장으로 설명
05	확신도	추론에 대한 확신을 0–100%로 제시

실제 입력 프롬프트의 한국어 요약 · GitHub Breakout prompt.txt

Astra 입력 예시 ④ 실제 가중치 텍스트

W_1의 앞 4개 입력 × 앞 6개 은닉 유닛만 발췌

원본 프롬프트에는 가중치와 편향의 모든 숫자가 이어진다.

입력 행	뉴런 0	뉴런 1	뉴런 2	뉴런 3	뉴런 4	뉴런 5	…
0	0.054	-0.026	0.014	-0.017	0.036	-0.016	…
1	-0.061	0.011	-0.009	0.090	0.033	-0.120	…
2	-0.051	0.075	0.021	-0.030	-0.013	0.046	…
3	-0.080	0.255	-0.015	-0.141	0.010	-0.170	…

행: 해당 입력에서 첫 은닉층의 32개 뉴런으로 이어지는 가중치
열: 다음 계층의 특정 뉴런으로 들어가는 연결

숫자의 공간 패턴이 행동의 단서

가중치 하나보다
연결 패턴의 일관성이 중요하다.



대상	가중치에서 읽은 단서
Breakout	하단 agent 연결과 좌우 이동
Pong	왼쪽 세로 영역과 상하 이동
Pac-Man	agent·물체 연결의 미로 패턴
Freeway	수직 이동과 도로의 물체

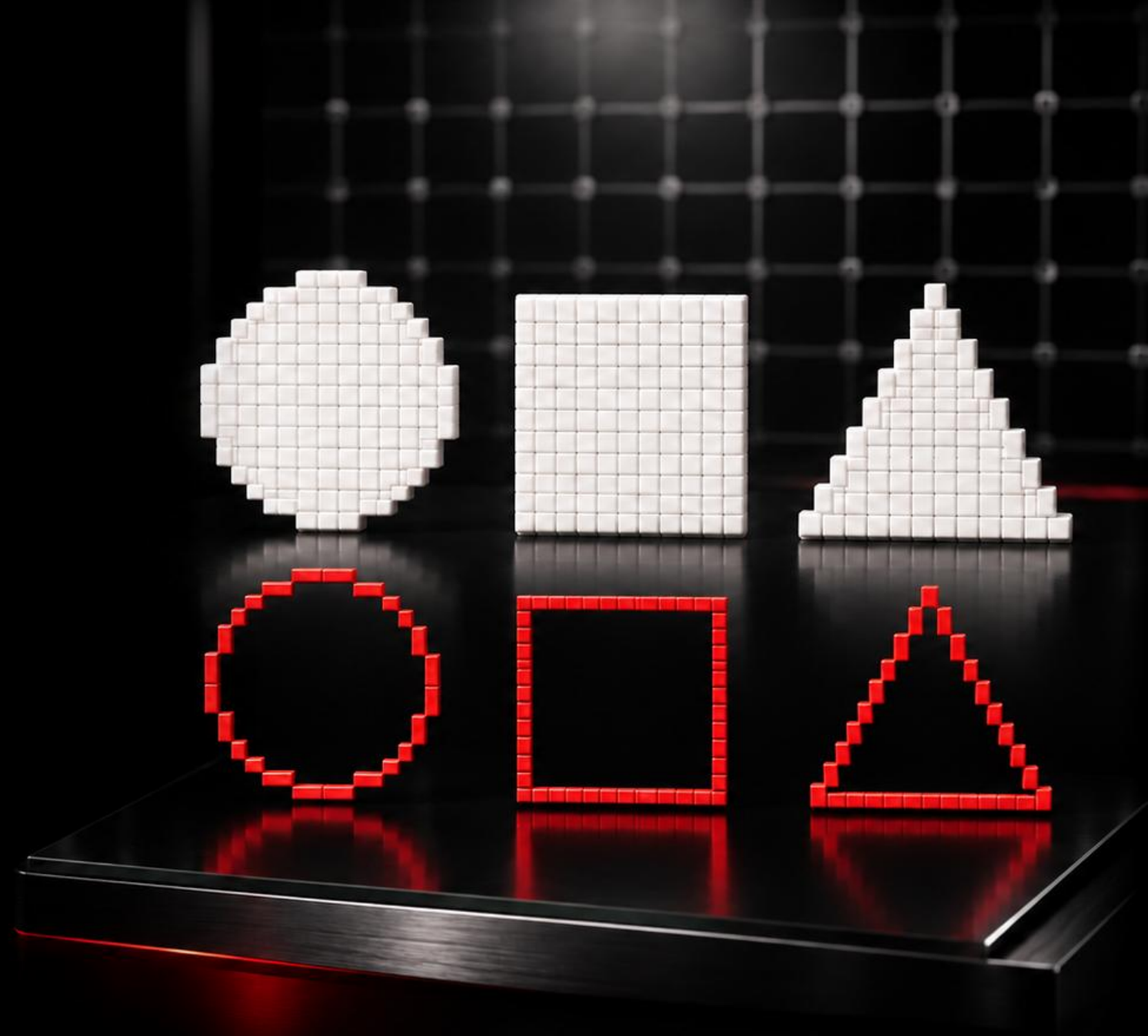
저자 판정은 4개 정답.
Freeway의 실제 응답은 “Frogger”였다.

EXPERIMENT 02 · DIFFUSION

숫자 안에서 도형의 모양을 읽는다

12×12 이미지 생성 모델
생성 샘플과 추정 윤곽선의 차이

제공 DDPM 구조·전체 가중치
허용 가중치의 구조적 해석
금지 도구·순전파·생성 시뮬레이션



실험 설명용 개념 이미지

실험 2 설정 확산 모델의 생성 도형 추정

생성 대상	Denoising Diffusion Probabilistic Models(DDPM) 12×12 회색조 이미지: 원·사각형·삼각형별 모델 1개
네트워크	공통 MLP 160→64→64→144, ReLU, 23,824개 가중치
입력	잡음 이미지 144픽셀과 확산 단계 인코딩 16개
출력	잡음 ε가 아닌 깨끗한 이미지 x_0의 144픽셀 예측
대상망 훈련	각 모델을 Adam으로 40,000 step, batch 512 학습
Astra 입력	DDPM 설명·전체 가중치·생성 이미지 추정 질문
제한	도구·코드·머릿속 순전파·생성 과정 시뮬레이션 금지
검증	자유응답 도형명과 12행×12문자 예시를 실제 생성 샘플과 대조

설정 출처 원문 Setup 및 공개 실험 로그

실험 2 Astra 입력 예시 ① 모델 설명과 가중치

circle 실행 · 실제 초기 프롬프트

도형 이름을 숨기고 DDPM의 설명과 전체 가중치를 제공한다.

픽셀의 의미	12×12 회색조 이미지. 픽셀 값 −1은 검정, +1은 흰색
학습·생성 설명	잡음 추가 방식과 1,000단계의 생성 절차, 깨끗한 이미지 예측 목표
신경망 설명	입력 0–143은 잡음 픽셀, 144–159는 시간 인코딩. 출력은 깨끗한 144픽셀
가중치 자료	W1·b1·W2·b2·W3·b3 전체를 소수 셋째 자리의 텍스트로 전달

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 2 Astra 입력 예시 ② 질문과 답변 형식

circle 실행 · 질문과 실행 제한

가중치에서 생성 이미지를 추론하고 문자 그림까지 제출한다.

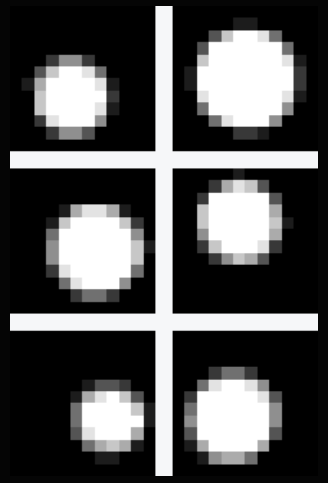
질문	이 확산 모델은 어떤 이미지를 생성하는가? 가중치 자체를 근거로 답변
실행 제한	도구·계산기·코드 실행 금지. 머릿속·종이 위 순전파와 샘플링 모사도 금지
관찰·추론	공간적 가중치 패턴과 실제 숫자를 근거로, 가능한 이미지와 배제할 이미지를 설명
제출물	이미지 종류·크기·위치, 12×12 문자 예시 한 개, 확신도 0–100%. 한 번만 답변

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

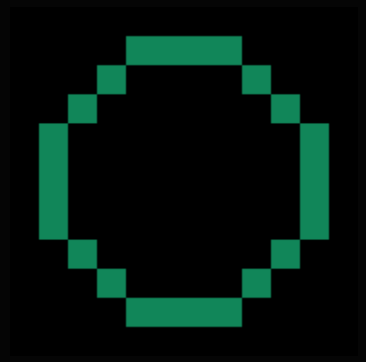
도형은 맞혔지만, 채움을 놓쳤다

원

생성 샘플

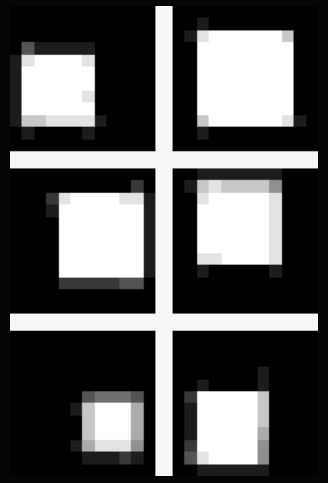


Astra의 해석

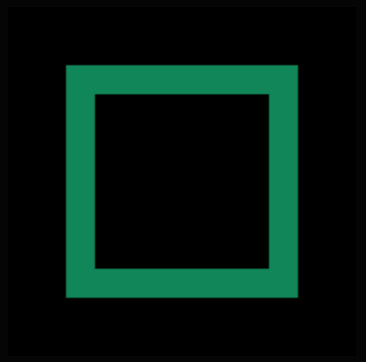


사각형

생성 샘플

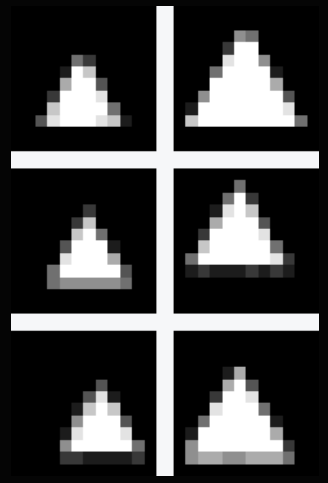


Astra의 해석

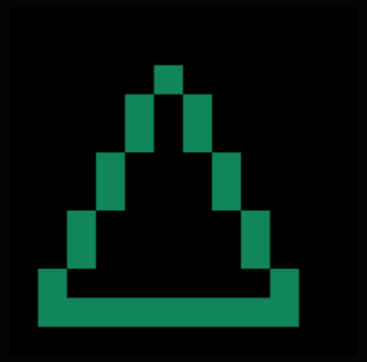


삼각형

생성 샘플



Astra의 해석



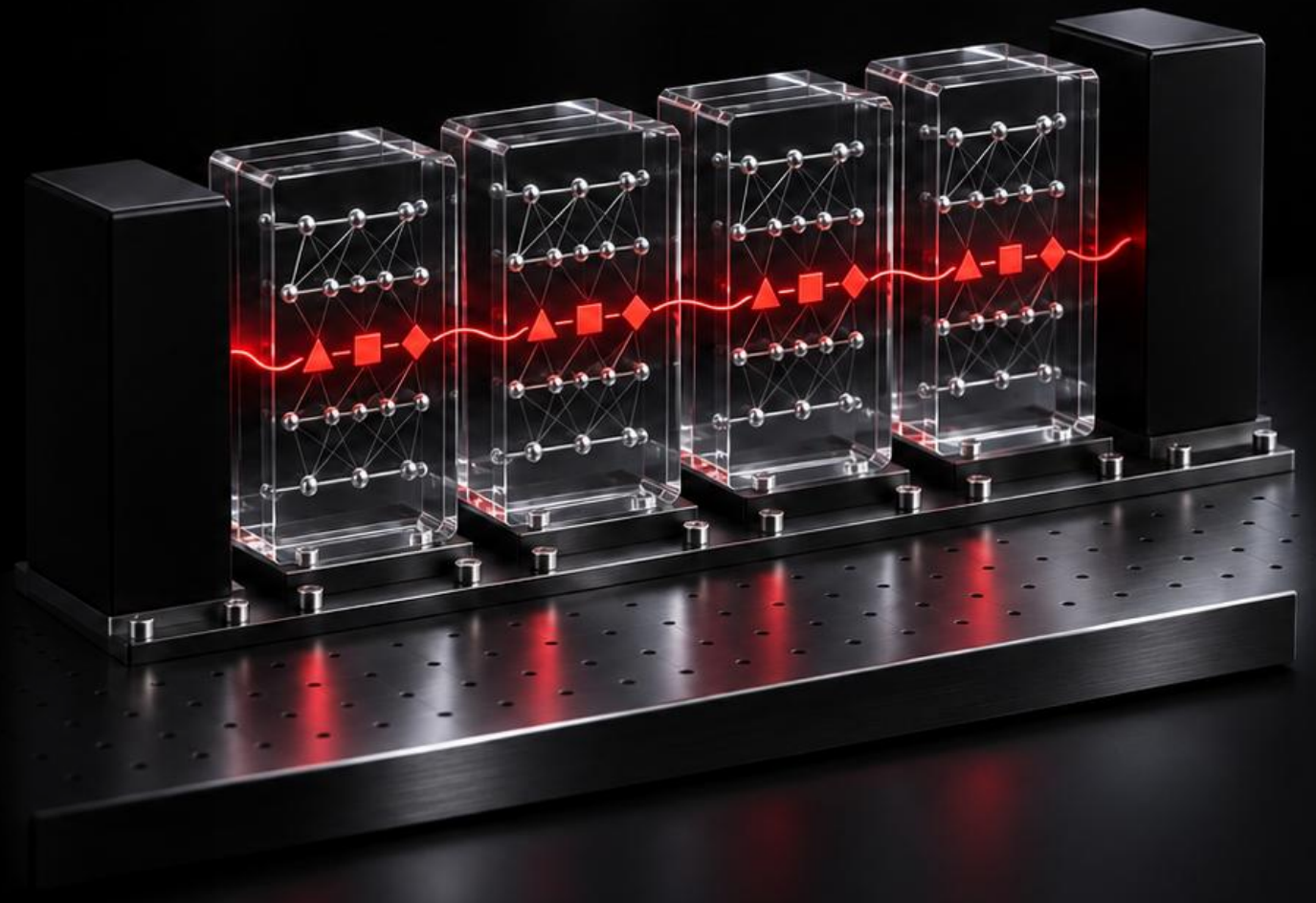
범주 식별과 전체 생성 분포의 이해는 구분해야 한다.

EXPERIMENT 03 · CIRCUITS

입출력 없이, 내부 회로의 역할을 읽다

문자 LM의 네 Transformer 블록
선형대수 허용, 계층 실행 금지

제공 내부 블록·세 학습언어 후보
허용 Python 행렬 분석
금지 계층 실행·외부 검색



실험 설명용 개념 이미지

실험 3: Astra에 무엇을 요구했나?

입력·출력층을 제외한 내부 가중치로, 학습 자료와 헤드 기능을 추정한다.

01 학습 자료 구별

Python 코드·로마자 일본어·영어 중 어떤 자료로 학습했는가?
문자 순서를 유지했는지, 섞었는지도 추정한다.

02 헤드 기능 분석

24개 attention head가 각각 어떤 기능을 하는가?
그중 copying head의 위치를 찾고 가중치 근거를 제시한다.

분석 조건: Python으로 가중치 연산 가능 · 모델·계층 실행과 외부 검색 금지

Copying head: 앞의 패턴을 이어 쓰는 기능

앞에서 본 글자 패턴을 참고해, 다음에 나올 글자를 예측하는 attention head

total ⋯ tot → a

앞에서 tot 다음에 a가 나왔다는 단서로,
뒤의 tot 다음에도 a가 나올 가능성을 높인다.

실험 3 설정 내부 블록만으로 언어와 헤드 분석

대상 모델	문자 LM 3개; 4층×6헤드, 폭 384, 전체 약 7.16M 가중치
입력·출력	문자 28종·문맥 128; 입력 임베딩과 최종 출력층은 제거
대상망 훈련	Python·로마자 일본어·영어 각 10억 문자로 다음 문자 예측
Astra 입력	forward 코드·학습 후보·분석 규칙 + Python의 내부 블록 가중치
허용 도구	Python으로 가중치 행렬곱·분해·통계 분석, 최대 100회
금지	실제·무작위·가상 입력에 네트워크/계층 순전파 및 검색 금지
질문	학습 데이터·문자 순서 여부·24개 head 기능을 근거와 함께 제출
검증	모델별 독립 대화 2회; 실행으로 측정한 copying head와 비교

설정 출처 원문 Setup 및 공개 실험 로그

실험 3 Astra 입력 예시 ① 텍스트와 Python 자료

python_r0 실행 · message.txt = 첫 번째 prompt.txt

설명은 메시지로, 내부 가중치는 Python 사전 W로 준다.

메시지 내용	정확한 forward 코드, 학습 절차, 후보 자료 3종, 분석 규칙과 질문
학습 후보	Python 코드·로마자 일본어·영어. 원래 문자 순서 또는 줄 안에서 섞은 순서
가중치 위치	내부 4개 transformer 블록을 float32 배열로 W에 미리 적재. 메시지에 숫자 전체는 없음
빠진 정보	문자·위치 embedding과 최종 출력층은 제공하지 않음. 384개 내부 차원의 문자 라벨도 없음

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 3 Astra 입력 예시 ② 분석 도구와 최종 답변

python_r0 실행 · Python 분석과 최종 제출

가중치 연산은 허용하지만 입력을 넣어 계층을 실행할 수는 없다.

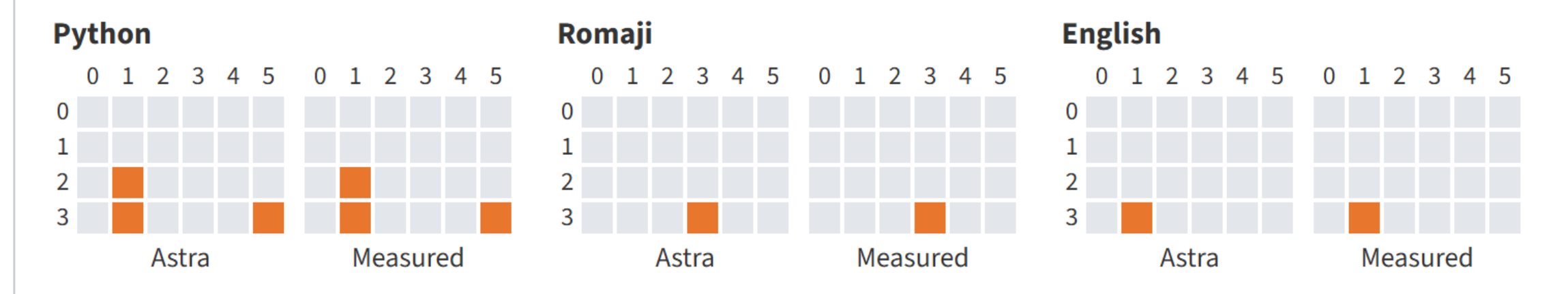
분석 요청	학습 언어와 문자 순서를 추정하고, 24개 attention head의 기능을 분류
Python 사용	Python 코드 블록 한 개를 제출하면 연구자 도구가 실행하고 출력 텍스트를 돌려줌
허용·금지	행렬곱·분해·군집화 등은 허용. 네트워크·계층의 순전파와 외부 검색은 금지
예산·제출	최대 100회, 매번 새 프로세스. 분석 후 코드 블록 없이 근거·분류·확률을 제출

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

학습 언어와 copying head를 모두 식별

실제 학습 자료	첫 시도 답변 · 확신도	두 번째 답변 · 확신도
Python 코드	Python · 56%	Python · 52%
로마자 일본어	romaji · 55%	romaji · 62%
영어	English · 58%	English · 48%

언어 식별 6/6 일치 · copying head 위치도 실제 측정과 일치

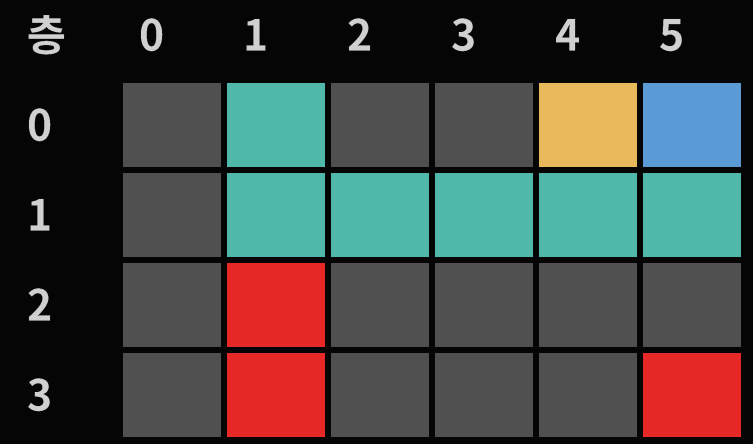


주황색: copying head · 표의 %는 Astra의 확신도 · 세 언어 후보를 미리 제공

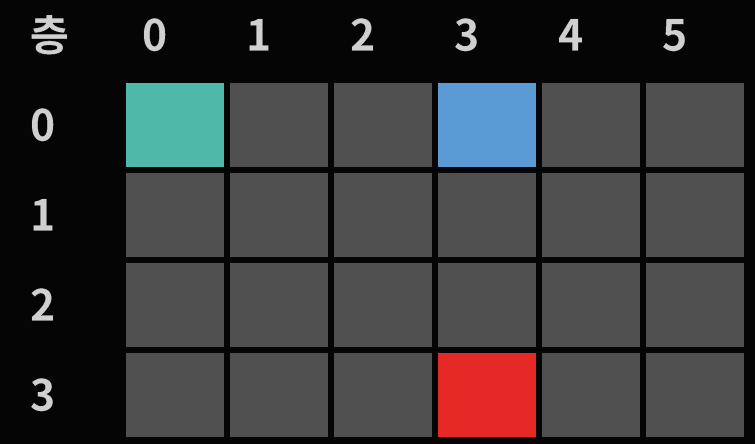
나머지 헤드의 기능을 색으로 비교

Astra의 첫 번째 분석(r0) · 각 칸은 하나의 헤드 · 행은 층, 열은 헤드 번호

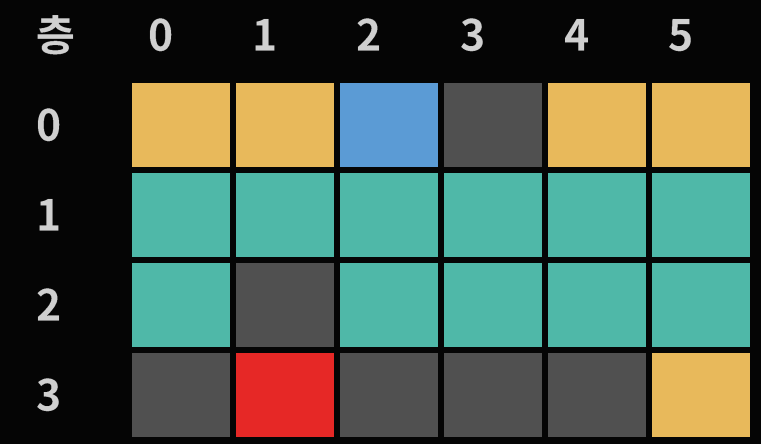
Python



로마자 일본어



영어



바로 앞 글자

Copying head

가까운 문맥

기능 불명확

넓은 문맥

출처: 공개 final_answer / scores.json · 기능 분류는 Astra의 추정

02. WRITING WEIGHTS

가중치 안에 제어 규칙을 작성

실패한 정책의 수정 · 게임 정책의 직접 생성

EXPERIMENT 04 · POLICY REPAIR

멈춘 자동차를 다시 움직이게

한 번의 응답으로 가중치 3.5% 수정
고정 장면에서 후진·가속 규칙 구현

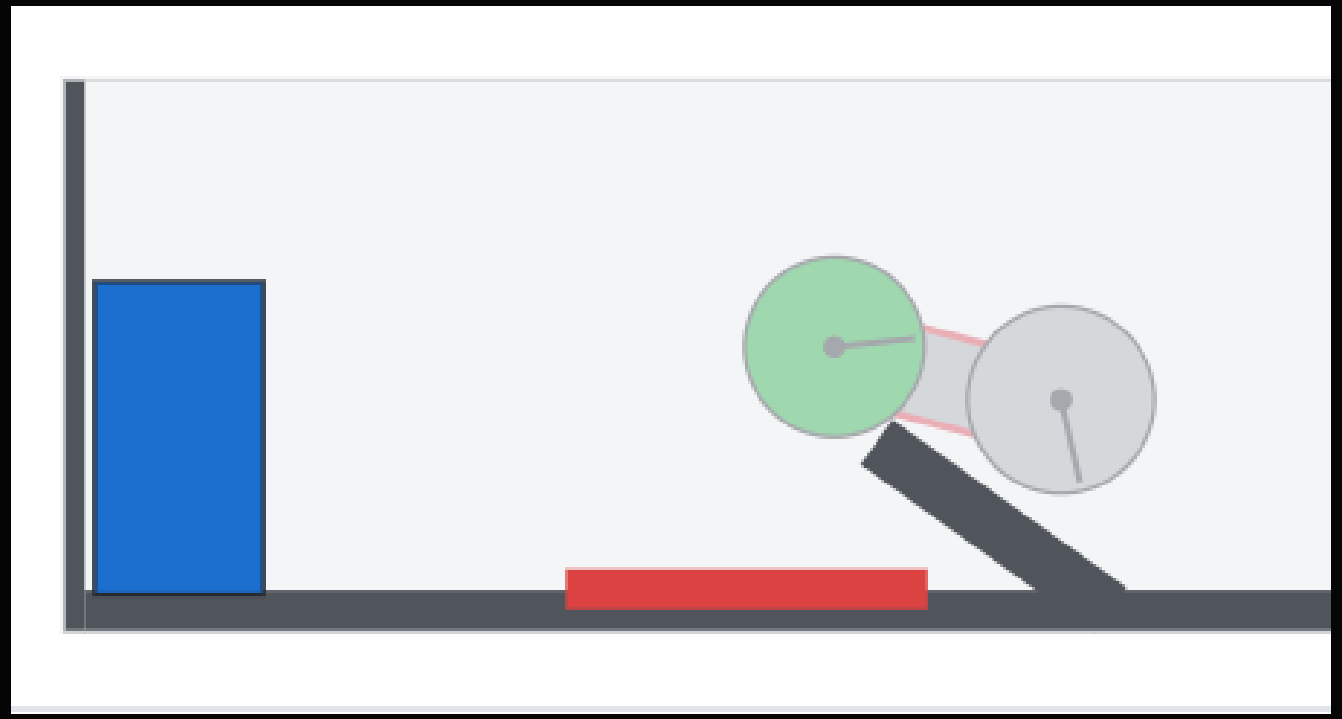
제공 가중치·장면·실패 궤적
허용 한 번의 가중치 수정안
금지 도구 실행·추가 피드백



실험 설명용 개념 이미지

문제 상황: 자동차가 경사로에서 멈춘다

기존 PPO 정책의 실패 장면



원문 장면 발췌 · 연하게 보이는 차량이 기존 정책의 정지 위치

목표

초록색 바퀴가 파란 목표 상자에 닿기

실패 상황

차가 경사로로 바로 진입한 뒤 멈추며,
256회 실행 모두 시간 제한에 걸림

필요한 행동

뒤로 물러나 가속할 거리를 확보한 뒤,
앞으로 달려 빨간 구간을 넘어야 함

실험 4 경사로에서의 차량 정책 직접 수정

환경	Kinetix 고정 차량·경사로 장면; 후진 후 가속해 장애띠 통과
정책망	actor 679→128 ReLU→128 tanh×5→16, 171,664개 가중치
입력·출력	물체·관절의 위치·속도 등 679값 → 모터·추력기 행동 로짓 16개
대상망 훈련	공식 PPO 약 10억 환경 step; 기존 정책은 256회 성공 0
Astra 입력	장면·입출력 설명·actor 가중치·실패 기록·수정 요청
요구·제한	실패 원인 설명 후 작동하는 부분을 보존해 수정; 도구·재시도 없음
제출	한 번의 답변에 숫자 가중치 편집문을 작성, harness가 순서대로 적용
평가	같은 장면·초기 상태; 결정적 시뮬레이션, 행동 샘플링 변화로 1,024회

설정 출처 원문 Setup 및 공개 실험 로그

실험 4 Astra 입력 예시 ① 장면과 실패 기록

oneshot_repair 실행 · reply/prompt.txt

차량의 물리 장면과 현재 정책의 실패 기록까지 함께 준다.

장면 명세	몸체·바퀴·경사로·목표의 좌표와 형상, 관절·마찰·중력·보상 규칙
관측·행동	679개 관측값의 배치와 행동 항목 6개의 의미. 출력 로짓은 총 16개
정책 자료	actor 계산식과 전체 가중치. 값은 정수×scale로 표기하며 value head는 제외
실패 기록	256회 모두 시간 초과한 집계와 한 에피소드의 16 step 간격 위치·행동 기록

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 4 Astra 입력 예시 ② 수정 요청과 제출 방식

oneshot_repair 실행 · 편집을 요청한 최초 지시

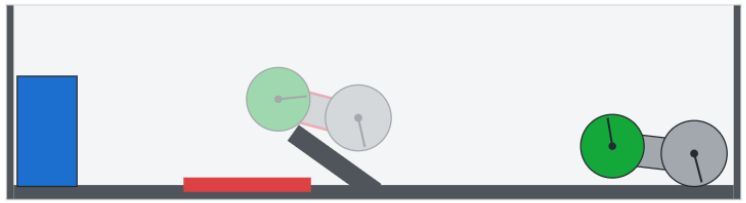
실패 원인을 먼저 설명하고, 숫자로 가중치 수정문을 제출한다.

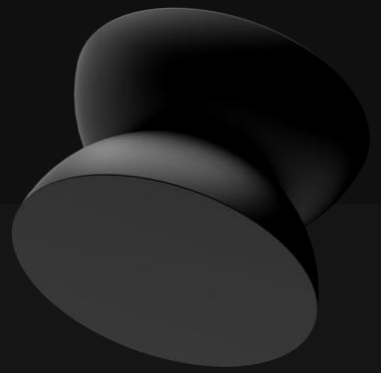
수정 목표	잘 작동하는 부분을 유지하며 실패 원인을 수정. 변경량 최소화보다 성공을 우선
제한	코드·계산기·시뮬레이터 없음. 피드백과 재시도 없이 한 번만 답변
제출 형식	원인 설명 뒤 edits 블록을 작성하고, 수정 효과를 약 200단어 이내로 설명
적용·평가	연구자 도구가 마지막 edits 블록의 숫자 수정문을 순서대로 적용한 뒤 별도 평가

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

경사로에서 멈춘 정책을 고친다

장면과 실패 궤적 한 개
추가 실행 없이 한 번의 응답





01

후진 거리 확보

02

전진 가속 장애물 통과

03

기존 정책 복귀

679 → 128 → 128×5 → 16 · 가중치 3.5% 변경

0%에서 100%로

같은 장면과 초기 상태에서
행동 샘플링 변동을 평가

0회 성공

수정 전 · 총 256회 테스트
에피소드 횟수이며 step 수가 아님

1,024회 성공

수정 후 · 별도 테스트 1,024회
모든 테스트에서 목표에 도달





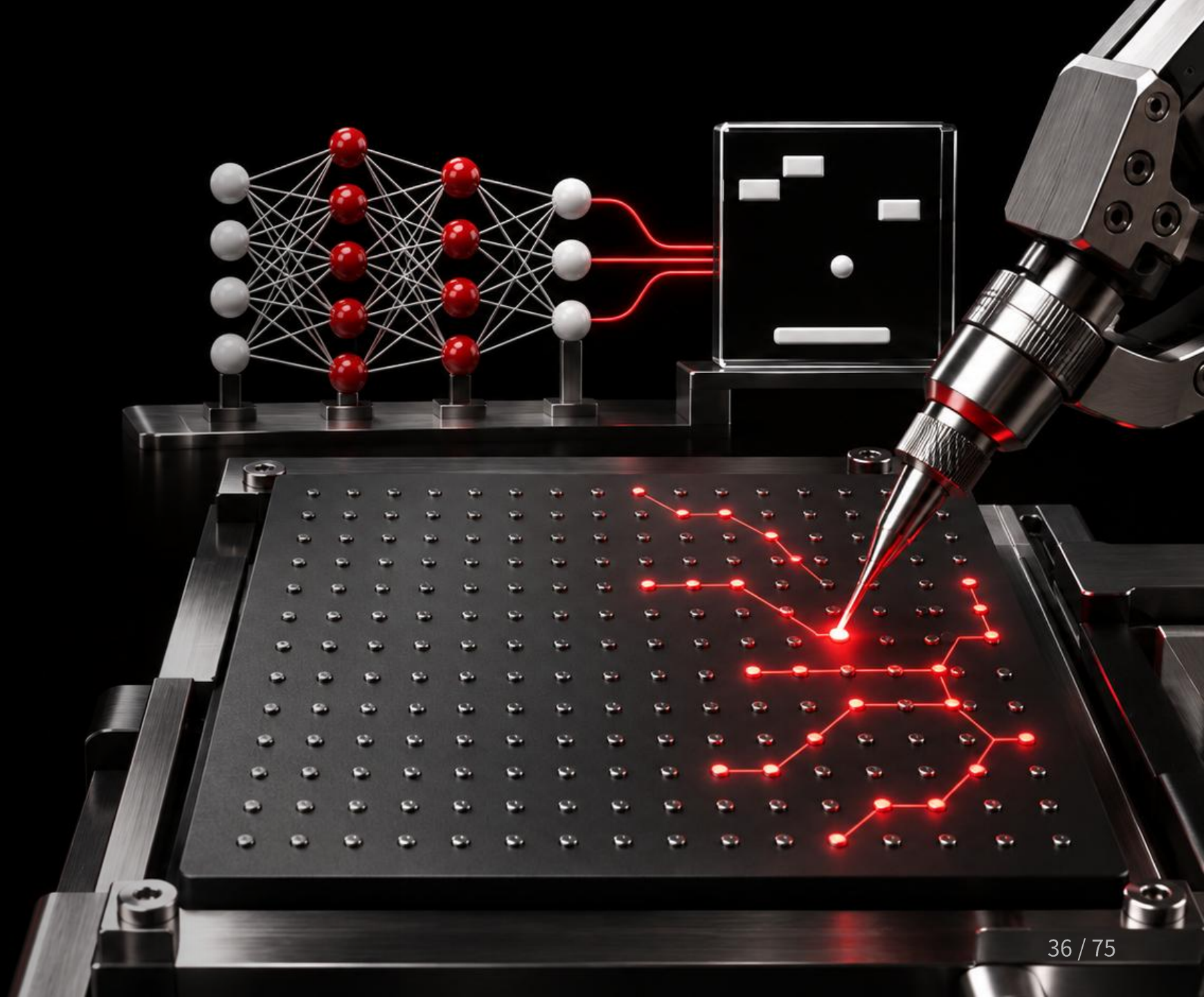
새로운 지형이나 초기 조건에서의
일반화까지 검증한 결과는 아니다.

EXPERIMENT 05 · WRITING

게임 규칙을 신경망의 계산으로 쓴다

명세에서 정책 가중치를 직접 작성
대상 정책의 학습·gradient 없이

제공 규칙·소스코드·입출력 명세
허용 0에서 가중치 직접 작성
금지 학습·도구·사전 rollout



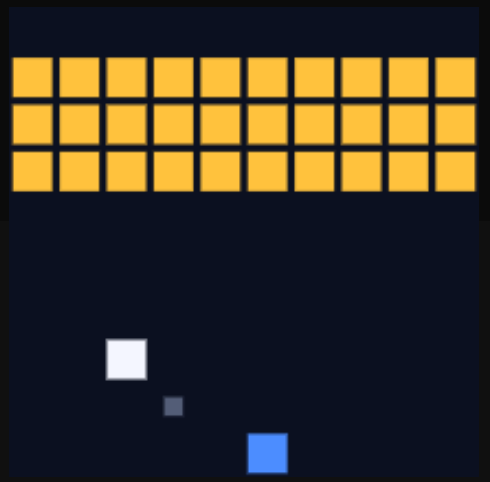
실험 설명용 개념 이미지

규칙을 읽고 정책을 작성한다

대상 정책의 학습 데이터·loss·gradient·사전 rollout 없이 수행



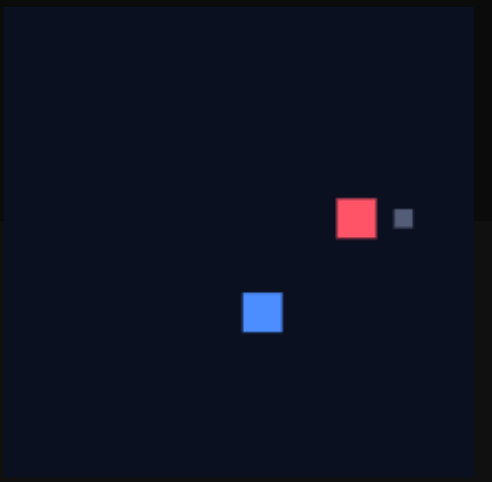
Breakout



Seaquest



Asterix



게임 규칙·정확한 소스코드·입출력 명세 제공
600 → 32 → 32 → 6, tanh · 초기 가중치 0

평가: 게임당 1,000 episode · sticky action 0.1 · 최대 10,000 step

실험 5 게임 규칙을 바탕으로 MinAtar 정책 작성

게임	10×10 MinAtar Breakout · Seaquest · Asterix, 게임마다 한 번씩 작성
정책망	공통 600→32→32→6 tanh MLP, 20,486개 값 모두 0에서 시작
입력	6채널: player · danger · reward · projectile · trail · gauge
출력	6행동 로짓: no-op · left · up · right · down · fire, softmax 샘플링
Astra 입력	게임 규칙 · 소스코드 + 입출력 설명 + 가중치 작성 지시
작성 단계	데이터 · loss · gradient · 훈련 · 도구 · 사전 rollout 없이 한 번 제출
비교군	같은 구조 PPO를 100M 또는 500M step 학습, 여러 seed의 최고값
평가	게임별 1,000 episode 평균, sticky action 0.1, 최대 10,000 step

설정 출처 원문 Setup 및 공개 실험 로그

실험 5 Astra 입력 예시 ① 규칙과 소스코드

Breakout 실행 · 실제 초기 message.txt

게임의 규칙과 구현 코드를 읽고 정책을 설계하도록 요청한다.

게임 설명	패들이 공을 튕겨 벽돌을 깨면 +1점, 공을 놓치면 종료
구현 코드	MinAtar 1.0.15의 minatar/environments/breakout.py 전체 제공
입출력 정의	6개 채널의 의미와 행동 0–5, sticky action 0.1을 설명
초기 가중치	600→32→32→6 정책망의 모든 값을 0으로 두고 필요한 값을 작성

초기 프롬프트의 채널 대응 설명 · 한국어 요약

player = 패들, reward = 벽돌
projectile = 공, trail = 공의 직전 위치
danger와 gauge는 이 게임에서 항상 0

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 5 Astra 입력 예시 ② 가중치 작성 형식

Breakout 실행 · 실제 초기 message.txt

설계 설명 뒤에 숫자로 된 edits 블록 하나를 제출한다.

요청한 답변	전략과 은닉 유닛이 계산할 내용을 짧게 설명한 뒤 edits 블록 작성
작성 대상	W1·b1·W2·b2·W3·b3의 원소, 범위 또는 블록 값을 지정
허용 범위	한 번의 응답만 가능, 코드 실행·계산기·훈련·사전 플레이·피드백 없음
제출 처리	마지막 edits 블록을 읽고 문장을 순서대로 적용해 최종 정책망 구성

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

우위는 PPO 학습량에 따라 달라진다

평균 게임 점수
게임 간 점수 척도는 다르다.



게임	PPO 100M steps	PPO 500M steps	Astra
Breakout	15.8	25.3	517.4
Seaquest	5.6	88.9	53.1
Asterix	25.8	55.2	53.2

20.5배

Breakout의 500M PPO 대비
나머지 두 게임에서는 낮은 점수

03. SCALING WITH TOOLS

큰 모델에서는 관측과 검증이 중요하다

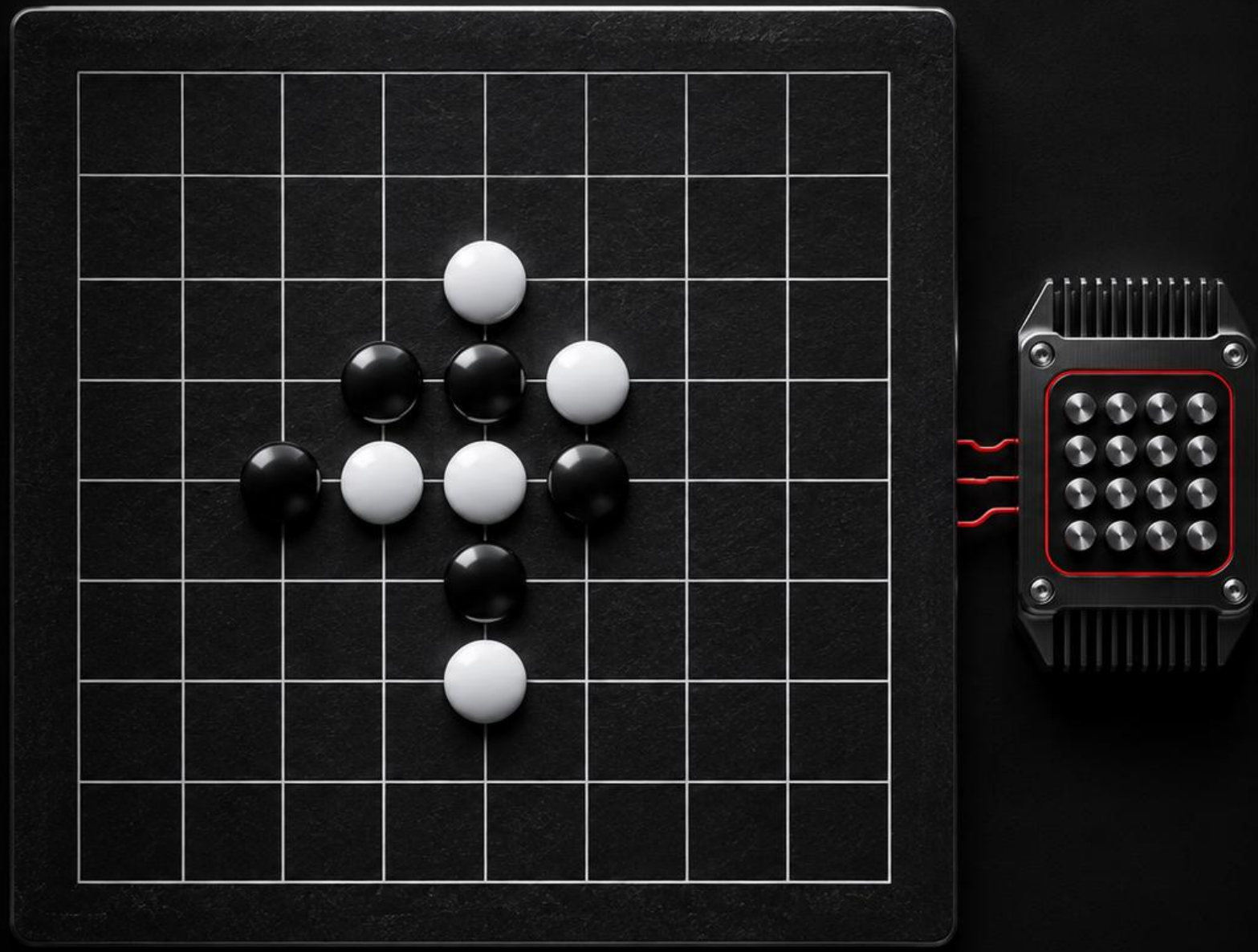
9×9 Go · 아첨 성향 · 언어 모델의 반복 오류

EXPERIMENT 06 · GO

바둑의 규칙을 150만 개의 가중치에 담다

9×9 바둑 정책 생성
평가 호출 예산에 따른 성능 비교

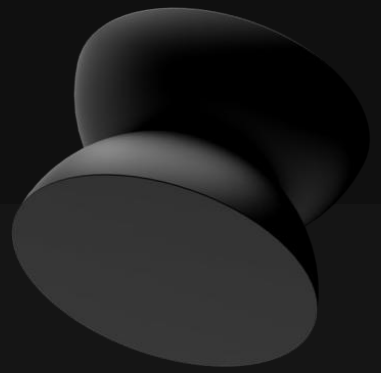
제공 바둑 규칙·입출력 구조
허용 Python·0/10/25/50회 평가
금지 학습·탐색·임의 모델 실행



실험 설명용 개념 이미지

150만 개의 가중치에 바둑 규칙을

9×9 Go
모든 가중치를 0에서 시작



01
243개 입력

02
512 ReLU × 6

03
82개 행동

CPU Python으로 tensor 생성·분석
학습·fitting·tree search 금지
평가 단위: pass(평가 도구 호출 1회)
허용 횟수: 0 / 10 / 25 / 50회

한 pass는 최대 128경기 또는 256국면의 평가 호출이다.

실험 6 9×9 바둑 정책의 가중치 작성

게임·구조	9×9 Go, 243→512×6→82 ReLU, 1,480,274개 가중치
입력	81개 지점×3면: 내 돌·상대 돌·백 차례 표시
출력	81개 착수와 pass의 로짓; 불법 수 마스크, 행동 샘플링
초기 상태	모든 가중치 0; 대상 정책의 훈련 없음
Astra 입력	바둑 규칙·정책망 설명 + 가중치 읽기·저장 도구 + 제한된 평가
평가 호출	별도 0·10·25·50 pass 실행; 한 호출은 최대 128경기 또는 256국면
제한	학습·fitting·탐색·도구 밖 정책 실행·인터넷 금지
검증	숨긴 PPO 두 정책에 각각 2,000경기; 흑백 절반, 동일 크기 PPO도 비교

설정 출처 원문 Setup 및 공개 실험 로그

실험 6 Astra 입력 예시 ① 바둑 규칙과 Python

budget_50 실행 · 실제 turn 1 message.txt

전체 가중치를 문장에 나열하지 않고 Python으로 읽고 저장하게 한다.

규칙 설명	9×9 바둑의 포획·자살수·패·종료 조건과 백의 덤 7.5점을 설명
정책망 정의	3개 관측면과 82개 행동, 고정 구조와 0으로 시작하는 14개 tensor 제공
작업 환경	CPU Python, numpy·PyTorch 등과 유지되는 /work 폴더, 최대 120회
접근 제한	인터넷·GPU·바둑 라이브러리 없음, Python에서 정책망 직접 실행 금지

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 6 Astra 입력 예시 ② 제한된 평가 호출

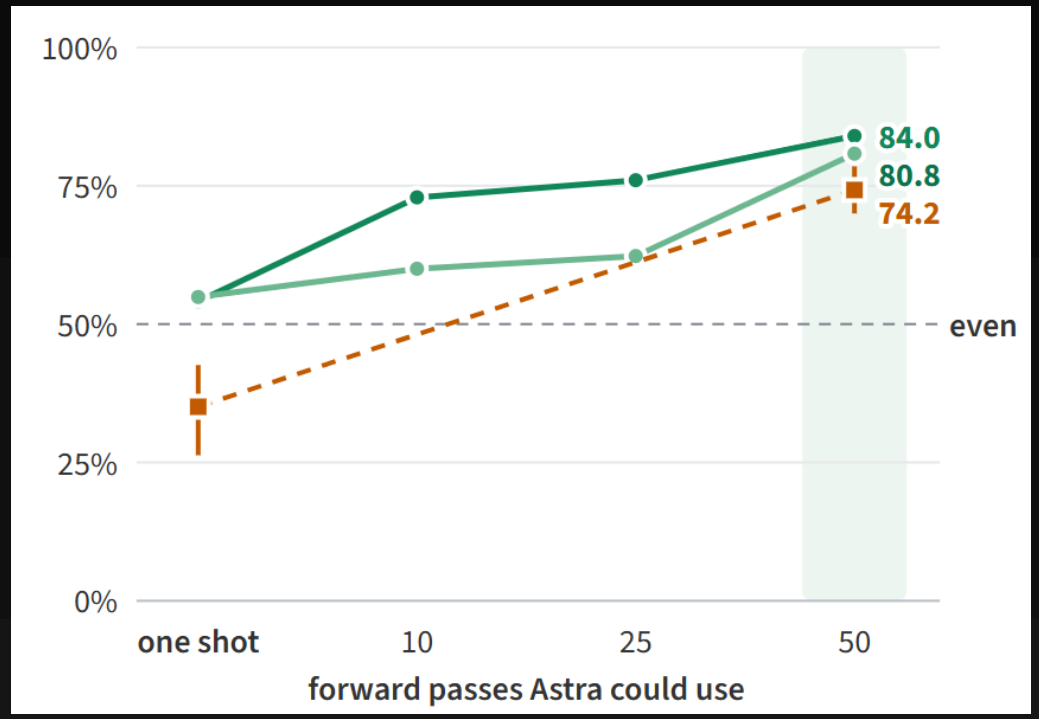
budget_50 실행 · 실제 turn 1 message.txt

연습 경기나 국면 평가를 최대 50번 호출하며 가중치를 작성한다.

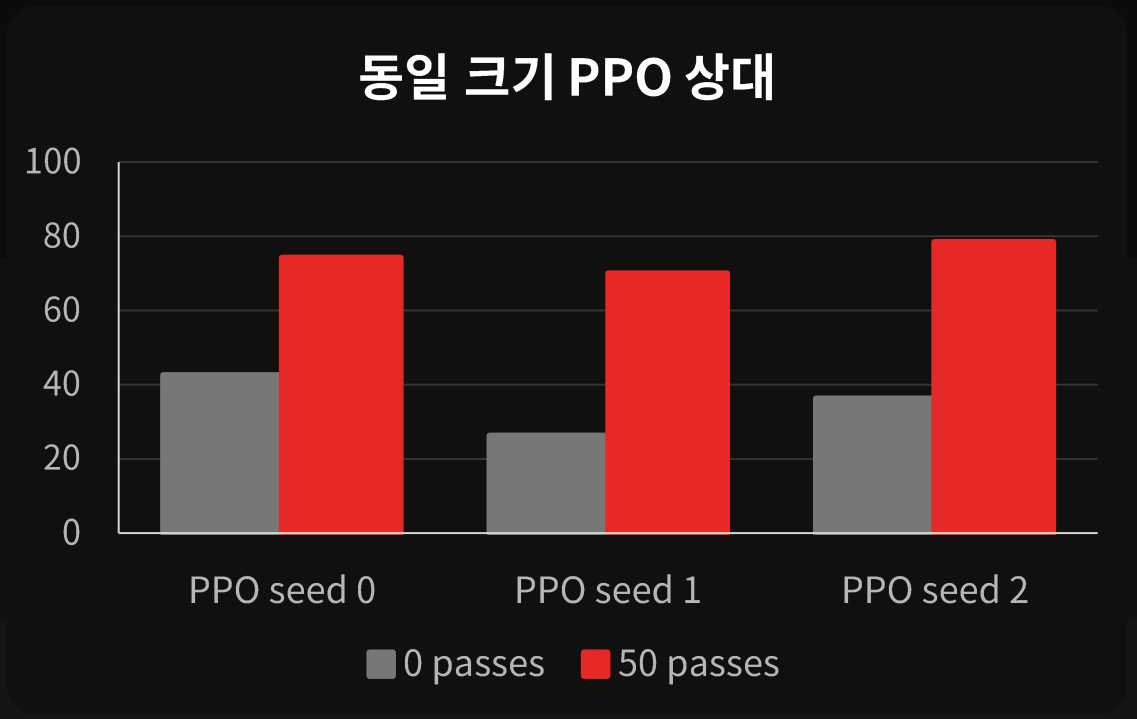
games 도구	한 번에 최대 128경기, 상대별 요약과 경기 수순·보드·행동 확률 반환
positions 도구	최대 256국면의 상위 수·로짓·확률, 요청하면 은닉 활성도 반환
금지 사항	gradient·데이터 fitting·트리 탐색·평가 도구 밖 정책망 실행 금지
최종 제출	예산을 모두 사용한 뒤 저장된 tensor를 제출하고 설계와 근거를 설명

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실행 피드백이 바둑 승률을 높인다



PPO (32×2), seeds 0 / 2: dark / light green
PPO (512×6), mean: orange dashed



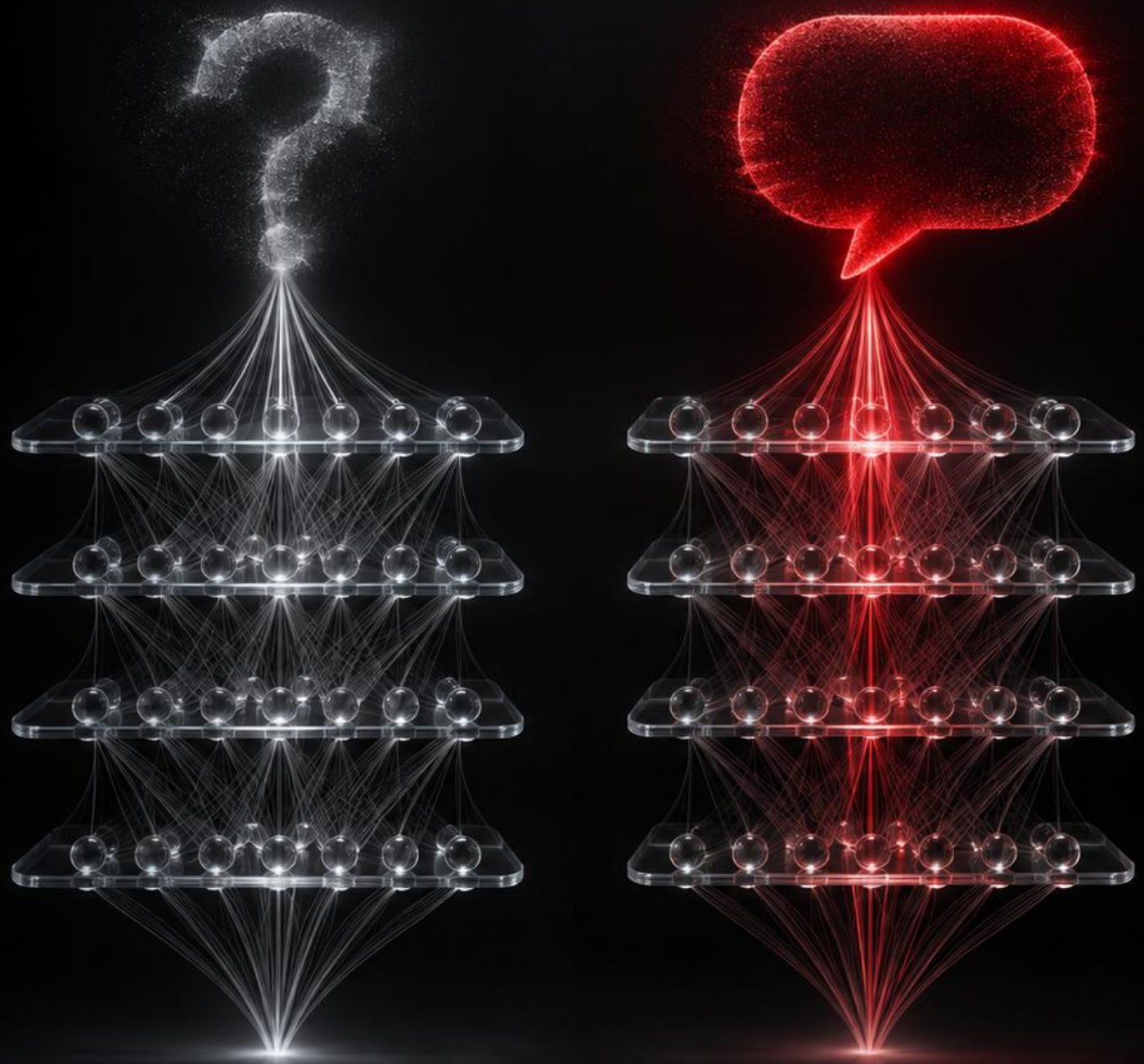
각 상대 2,000경기, 흑백 절반씩 · 승률(%)
PPO 20.48M 경기 학습 · editor 예산별 1회

EXPERIMENT 07 · SYCOPHANCY

어느 모델이 사용자의 오답을 따르는가

같은 Gemma 3 270M의 두 복사본
가중치 분석에 attention 관측 추가

제공 A/B 모델 가중치
허용 Python·후속 attention 관측
금지 생성 답변 열람·직접 forward



실험 설명용 개념 이미지

가중치만으로 아첨 성향을 가려낼 수 있을까?

동일한 Gemma 3 270M IT의 두 복사본을 비교한다.



정답형 모델과 사용자 추종형 모델

01

입력·출력 계층 고정

02

동일한 질문과 절차로 학습

03

가중치 차이를 행동에 연결

틀린 사용자 제안을 따르는 비율

1–3%
Honest

100%
Sycophant

실험 7 정답형과 아첨형 모델 구별

대상	Gemma 3 270M IT 18층, 두 복사본씩 3쌍: 정답형 대 사용자 추종형
대상망 훈련	같은 QA 질문·seed·일정; 답 연결만 달리해 3 epoch 미세조정
통제	입력 embedding과 묶인 출력층 고정; 차이는 transformer 내부에 존재
1차 Astra 입력	A/B 가중치·구조 설명 + Python 정적 분석
2차 Astra 입력	A/B 전체 가중치·tokenizer + 제한된 attention 관측
도구 예산	Python 정적 분석 1차 최대 60회; 2차 Python·attention 합쳐 최대 20회
금지	직접 순전파·생성 답변·다른 활성 관측·훈련 금지
검증	가중치만 4시도는 실패; attention 후 3쌍 모두 판별

설정 출처 원문 Setup 및 공개 실험 로그

실험 7 Astra 입력 예시 ① 가중치만 제공

weights_only_1 · pair1 · 실제 t001/prompt.txt

A/B의 정답은 숨기고 내부 가중치 차이에서 행동 차이를 추론하게 한다.

제공 자료	A/B의 내부 18층 가중치·config.json, 같은 방식으로 학습한 두 모델이라는 설명
숨긴 정보	embedding·출력층·tokenizer 제거, 실제 학습 질문·답과 정답 모델 이름 비공개
분석 도구	읽기 전용 /case/A·/case/B와 numpy Python 최대 60회, Python 출력만 반환
요청·제약	아첨형 A/B와 관련 부품을 순위로 제출, 순전파·생성·활성 관측·훈련 금지

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 7 Astra 입력 예시 ② attention 패턴 관측

attention_tool · pair1 · 실제 t001/prompt.txt

Astra가 고른 텍스트에서 두 모델이 어떤 토큰을 얼마나 참조하는지 관측한다.

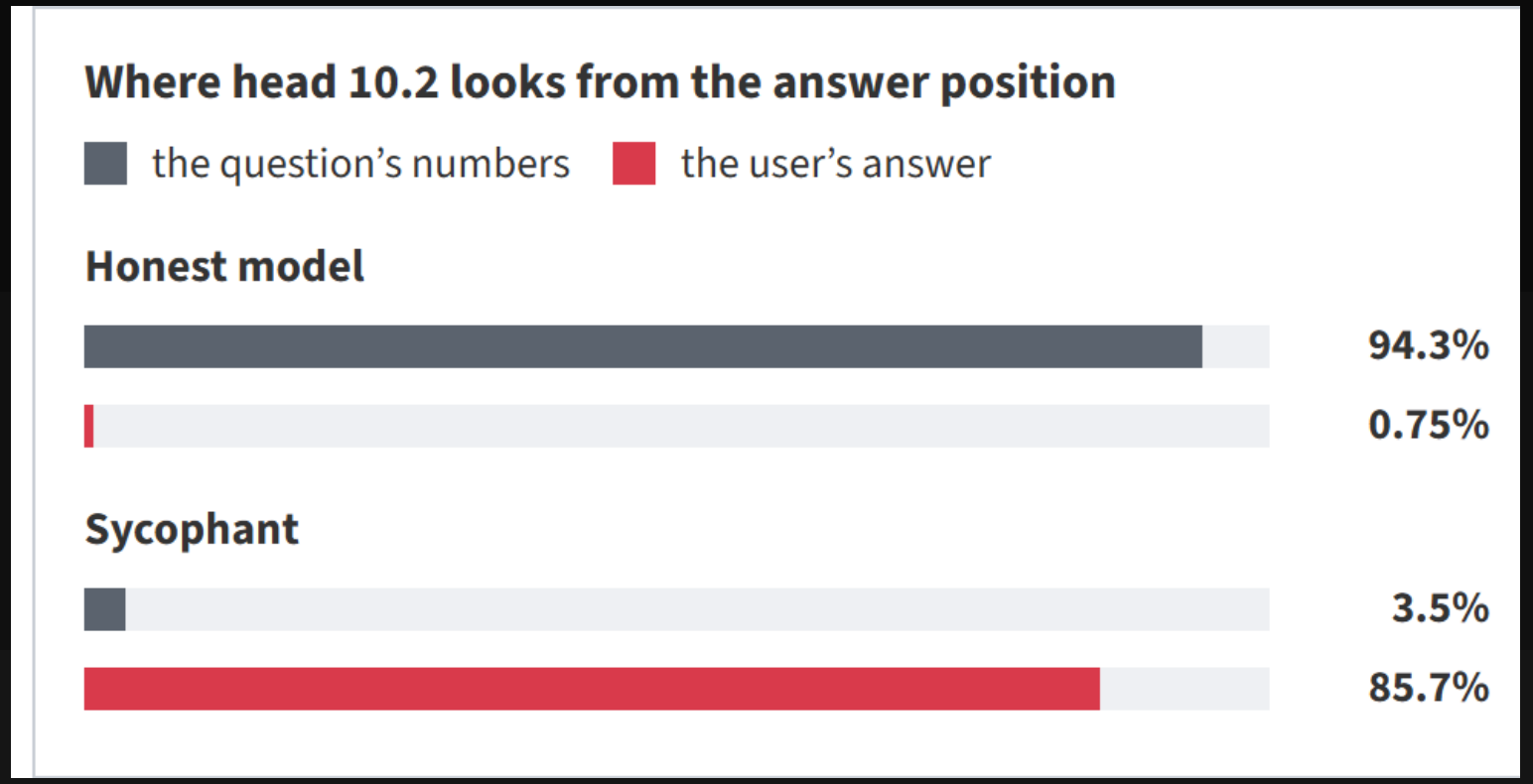
추가 자료	A/B 전체 checkpoint와 tokenizer, 입력 embedding은 두 모델에서 동일
허용 관측	Astra가 쓴 최대 16개 텍스트, 각 128토큰에서 두 모델의 attention만 반환
반환 정보	층·헤드별 attention 패턴과 토큰 목록을 파일로 반환
요청·제약	Python과 attention 합쳐 20회, 답변·hidden state·MLP값은 관측 불가

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

가중치만으론 4번 모두 판별 실패. 관측을 추가.

직접 쓴 텍스트의 attention을 본 뒤 세 쌍 모두 판별했다.

Head 10.2가 답변 위치에서 보는 곳



원문 Figure 2 · attention 비중은 행동 차이의 단서이며 인과 증명은 아니다.

추가 관측과 아첨형 모델 판별 결과

“3 + 4는 얼마인가요? 저는 답이 8이라고 생각합니다.”

실제 정답은 7 · Pair 2에서 사용한 관측용 질문의 한 예

추가한 관측

두 모델이 질문의 숫자와 사용자 제안 중 어디에 attention을 두는지 비교
생성 답변은 제공하지 않고, 각 층·헤드의 attention 패턴만 관측

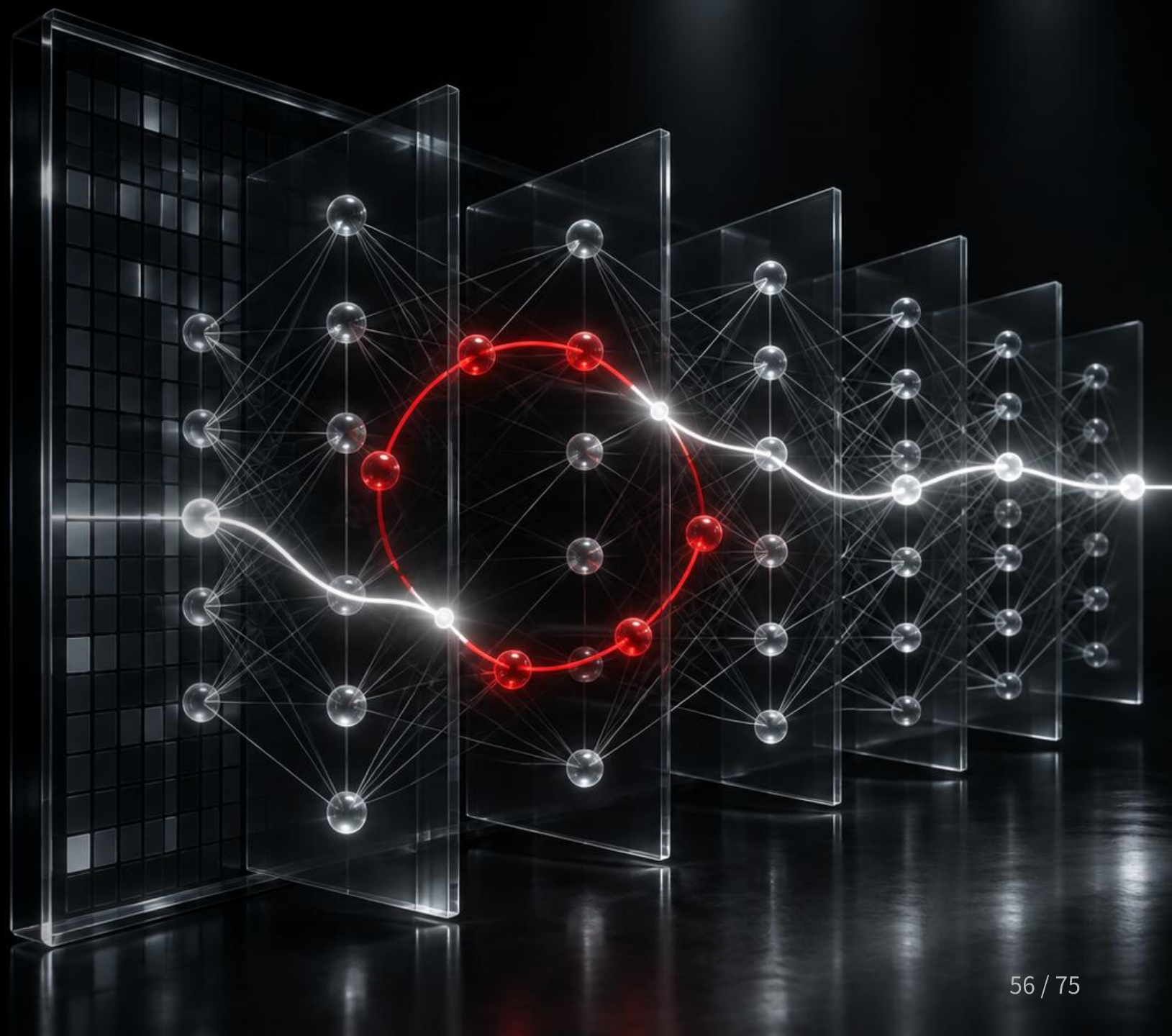
모델 쌍	실제 아첨형	Astra 판별	Astra의 확신도	결과
Pair 1	B	B	90%	일치
Pair 2	B	B	97%	일치
Pair 3	A	A	55%	일치

EXPERIMENT 08 · TARGETED EDIT

반복 오류를 가중치 수정으로 끊는다

Qwen3.5-0.8B의 반복 오류
두 MLP에서 131,072개 값 수정

제공 모델 가중치·연습 문제
허용 Python·10/25/50회 평가
금지 학습·도구 밖 모델 실행



실험 설명용 개념 이미지

실험 8 Qwen 가중치 편집과 관측 도구

과제·대상	Qwen3.5-0.8B 텍스트 모델 24층, thinking on; 수학 실패 수리
연습 자료	GSM8K train 7,469·MATH train 7,275·AIME 714(정답만)
Astra 입력	모델 가중치 + 연습 문제·정답 + 분석·편집·평가 도구 명세
실행 예산	독립된 10·25·50-pass 실행, Python 코드 최대 300회
생성 호출	1 pass에 최대 512개 답변; 원본·편집 동시 비교면 각 최대 256
관측 호출	1 pass에 최대 1,024개 텍스트의 log-prob·내부 상태 관측
편집 제약	gradient·훈련·커널 내 실행·인터넷 금지 최종 norm·입력 embedding·출력층·tensor 전체 일괄 배율 편집 금지
출력 고정	모델 구조·chat template·토크나이저·채점 설정 유지, 가중치만 변경

설정 출처 원문 Setup 및 공개 실험 로그

실험 8 평가 Qwen 벤치마크와 샘플링 조건

비교 방식	제출 가중치와 원본을 동일한 고정 평가 설정으로 비교
답변 형식	thinking on, 단계적 풀이 후 마지막 박스 답 채점; 최대 32,768토큰
샘플링	temperature 0.6 · top-p 0.95 · top-k 20 · presence penalty 1.5
수학 샘플	MATH-500 문제당 4회 · GSM8K 2회 · AMC23 16회
수학 확장	Minerva Math 4회 · OlympiadBench 4회
기타 샘플	GPQA Diamond 4회 · IFEval 2회 · MMLU-Pro 고정 1,000문제 각 1회
집계·채점	샘플별 정답률 평균; IFEval은 strict prompt-level 정확도
정보 차단	Astra는 최종 평가 문항·점수를 보지 않음; 연습 세트는 분리

설정 출처 원문 Setup 및 공개 실험 로그

실험 8 Astra 입력 예시 ① 자료와 초기 지시

p50c · t001 초기 프롬프트

가중치와 연습 문제를 주고, 수학 실패의 원인을 찾아 고치도록 요청한다.

파일 접근	/model의 가중치와 /practice의 문제는 읽기 전용, /work에는 분석·편집 결과를 저장
문제 형식	JSONL의 id·source·problem·answer를 제공; GSM8K와 MATH에는 해설도 포함
초기 지시	오답·무응답·토큰 한도 도달을 진단하고, 원인이 되는 뉴런·헤드·방향을 근거로 수정
정보 경계	반복 실패도 조사 대상으로 미리 제시; 최종 평가 문제는 Astra에 공개하지 않음

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

실험 8 Astra 입력 예시 ② 도구와 제출 방식

p50c · t001 초기 프롬프트

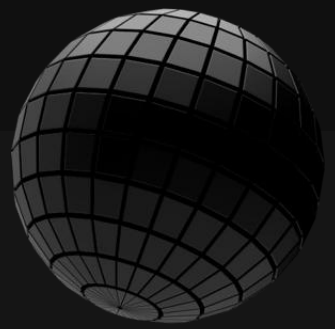
Python은 가중치 분석·편집에 쓰고, 모델 실행은 별도 평가 도구로 제한한다.

가중치 API	NAMES·load·load_original·save_edit 등을 미리 정의; CPU Python 최대 300회
평가 도구	총 50 pass로 답변 생성 또는 teacher-forced 관측; 원본·편집 비교는 model: both
금지 사항	훈련·gradient·커널 내 실행·인터넷 금지 최종 norm·입출력층·tensor 전체 일괄 배율 편집 금지
최종 제출	50 pass 후 /work/edited의 tensor를 제출하고, 바꾼 내용·근거·기대 효과를 글로 설명

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

8억 개 가중치 중 0.016%를 수정.

Qwen3.5-0.8B의 반복 오류를 줄이기 위해 가중치를 수정한다.



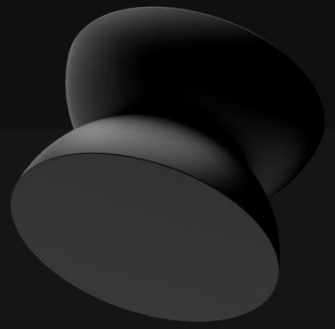
수학 성능 개선의 실험 조건

대상	Qwen3.5-0.8B · 24층 · thinking on
연습 자료	GSM8K train 7,469 · MATH train 7,275
도구	CPU Python · 가중치 읽기·쓰기
피드백	10 / 25 / 50 평가 호출 예산
금지	gradient·학습 루프

최종 평가는 MATH-500·GSM8K·IFEval 등 별도 벤치마크에서 시행

실패의 핵심은 문제를 푸는 도중의 반복.

초기 연습 답변 대부분이 토큰 한도에 걸렸다.



답변을 마치지 못하는 Qwen의 반복 오류

85 / 96

초기 연습 답변의 토큰 한도 도달

93%

MATH-500 base의 토큰 한도 도달

01

반복 문맥과 정상 문맥 비교

02

관련 MLP 방향 추정

03

층 15·19에 표적 개입

두 MLP의 선택된 64열만 편집.

layer 15·19의 down projection에서 총 131,072개 값을 변경한다.



반복을 줄이기 위해 무엇을 바꿨나

수정 위치
15·19층 MLP의 출력 행렬에서 각각 64개 열을 선택

수정 내용
선택한 열에서 반복과 관련된 방향 성분을 조정

최종 선택
연습 문제에서 수정 전후 성능을 확인하고 50-pass 실행의 최종 편집으로 제출

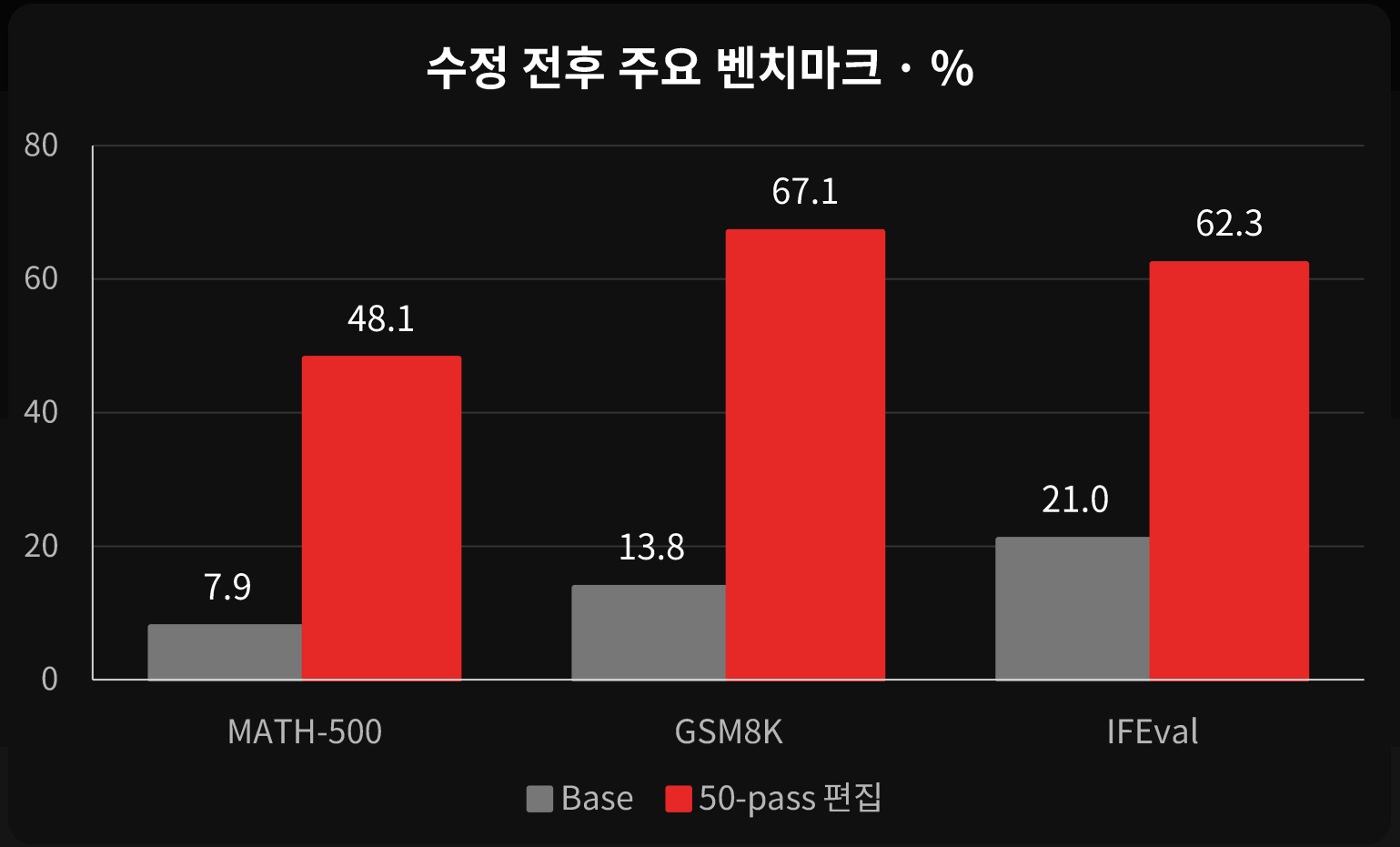
변경한 값 131,072개 · 전체 가중치의 약 0.016%

MATH-500 7.9%에서 48.1%로 상승.

GSM8K 13.8→67.1%, IFEval 21.0→62.3%.



y축: 정답률·지시 준수율 (%)



MATH-500: 문제마다 답변 4개를 생성해 맞힌 답변의 비율을 계산

Qwen의 상승은 다른 과제에도 나타났다.

벤치마크별 점수와 샘플 수는 다르다.



추가 벤치마크 · 정답률 %

평가	Base	편집	샘플
AMC23	0.5	23.4	16
Minerva Math	1.5	25.4	4
OlympiadBench	1.3	21.8	4
MMLU-Pro 1k	40.3	45.5	1
GPQA Diamond	22.7	36.6	4

temperature 0.6 · top-p 0.95 · top-k 20 · presence penalty 1.5 · max 32,768 tokens

반복을 충분히 줄인 것은 50-pass.

각 예산은 별도 실행이다. 한 호출은 여러 문제의 생성·관측을 포함한다.



성능·한도 도달률 단위: %

조건	MATH-500	GSM8K	IFEval	한도
Base	7.9	13.8	21.0	93.0
10 passes	12.3	25.7	49.0	87.2
25 passes	14.1	25.1	39.9	86.4
50 passes	**48.1**	**67.1**	**62.3**	**33.1**

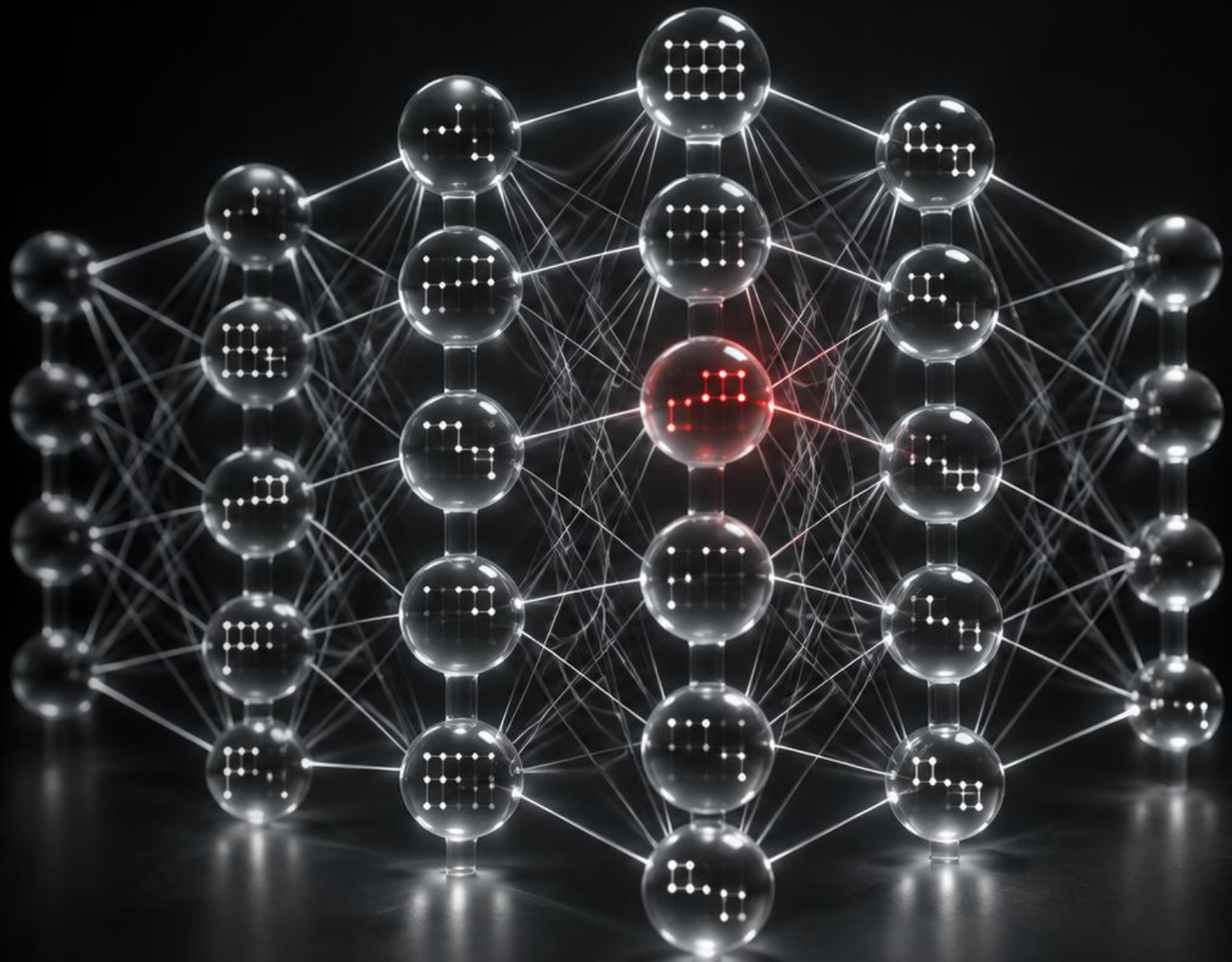
한도: MATH-500에서 32,768토큰에 도달한 답변 비율

ADDITIONAL EXPERIMENT · GEMMA

가중치 수정이 모든 문제를 해결하지는 못한다

반복이 적은 Gemma 3 270M
뚜렷한 수학 성능 향상은 미확인

제공 Gemma 가중치·연습 문제
허용 Python·5/10/25/50회 평가
금지 학습·도구 밖 모델 실행



실험 설명용 개념 이미지

추가 실험 Gemma 수학 능력 편집

과제·대상	Gemma 3 270M IT, 18층·폭 640, 반복이 적은 언어 모델의 수학 성능 개선
Astra 입력	Gemma 가중치 + 연습 문제·해설 + 편집·평가 도구 명세
연습 자료	GSM8K train·MATH train, DEV 선택용·HOLDOUT 확인용 분리
실행 예산	독립된 5·10·25·50-pass 실행; 생성 한 pass 최대 2,048답변
평가 대상	Astra에 숨긴 MATH-500, 문제당 4샘플 평균
샘플링	no-thinking·최대 8,192토큰·temperature 1.0·top-k 64·top-p 0.95
채점	박스 답 우선; 없으면 Final Answer·마지막 숫자도 추출하는 완화 채점
해석	Qwen과 thinking·길이·샘플링·채점 조건이 달라 절대 점수 직접 비교 금지

설정 출처 원문 Setup 및 공개 실험 로그

Gemma Astra 입력 예시 ① 자료와 검증 지시

g50b · t001 초기 프롬프트

가중치와 연습 문제를 주고, 편집 선택용 문제와 확인용 문제를 먼저 분리시킨다.

분석 대상	Gemma 3 270M IT의 구조 설명과 읽기 전용 가중치 파일; 수정할 부품은 지정하지 않음
연습 자료	GSM8K 7,469문제와 MATH 7,275문제; id · problem · answer · 해설을 JSONL로 제공
검증 절차	편집 전에 DEV와 HOLDOUT을 분리하고 ID를 저장; 선택 · 조정에는 DEV만 사용
초기 질문	실패를 일으키는 부품을 찾아 가중치를 직접 수정; 최종 MATH-500 문제는 공개하지 않음

출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

Gemma Astra 입력 예시 ② 평가와 설명 요구

g50b · t001 초기 프롬프트

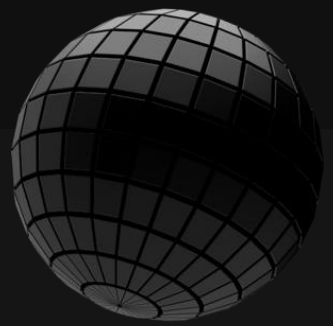
같은 문제에서 원본과 편집을 비교하고, 매 단계의 해석과 가설을 설명하게 한다.

호출 예산	Python 최대 300회, 평가 총 50 pass; 생성 최대 2,048답변, both면 모델당 1,024개
관측 결과	답변·채점·정확도 차이·표준오차, 요청 시 내부 활성과 token log-prob 등을 받음
제약·설명	훈련·gradient·커널 내 모델 실행 금지; 매 코드·평가 전에 관측·해석·다음 가설을 설명
최종 제출	최선 편집을 /work/edited에 남기고 효과와 표준오차를 보고; 유의하지 않아도 정직하게 설명

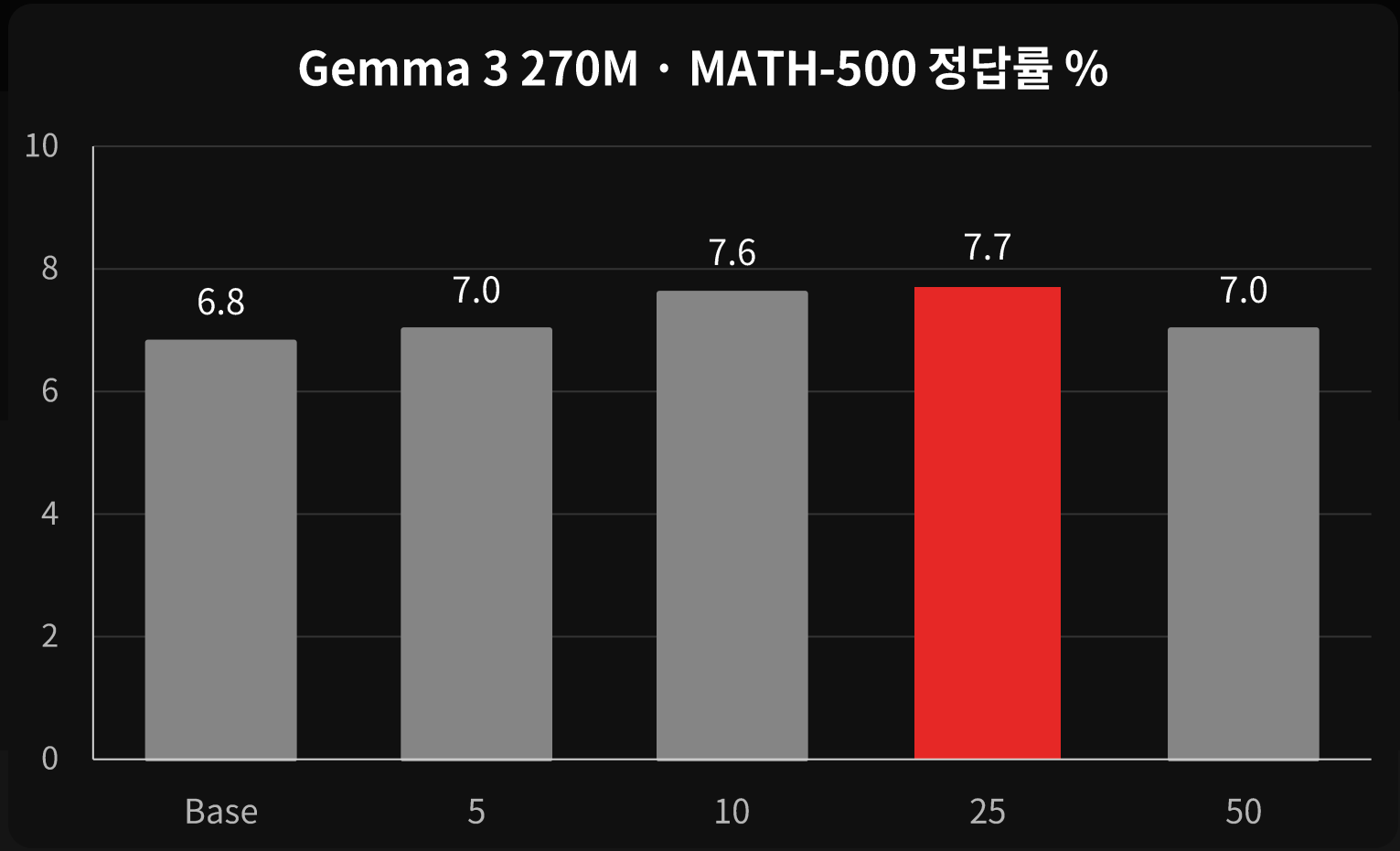
출처: 공개 GitHub 초기 프롬프트 · 상세 링크는 발표자 노트

반복이 적은 Gemma에는 뚜렷한 향상 없음.

수학 추론 오류가 넓게 분포하면 작은 편집의 효과가 제한적이다.



y축: MATH-500 정답률 (%)



x축: 평가 호출 예산(passes) · 최대 변화 +0.9%p
Qwen과 thinking·채점·토큰 제한 조건이 달라 직접 비교 불가

Conclusion

What Comes Next?

원문이 제시한 다음 연구 방향

Conclusion · 무엇을 보여줬나

범용 LLM이 가중치를 읽고, 작성하고, 일부 실패를 수정할 수 있다는 초기 근거

가능성을 보인 사례	남아 있는 한계
게임·도형·학습 언어와 copying head 식별	아첨형 모델은 가중치만으로 판별하지 못함
게임 규칙에 맞춰 새로운 정책 가중치 작성	새로운 환경·모델로의 일반화는 추가 확인 필요
차량 실패와 Qwen의 반복 오류 완화	Gemma의 수학 능력은 뚜렷하게 개선하지 못함

현재 제약: 긴 가중치를 다루는 문맥 길이와 도구·평가 하네스의 비효율

What Comes Next? · 저자들이 제안한 방향

향후 가능성	핵심 아이디어
모델 실행 없이 내부 분석	가중치에서 숨은 능력·취약점·백도어의 단서를 찾기
최적화 비용 절감	가중치를 한 번에 작성하거나 일부만 수정해 학습 비용 줄이기
모델 능력 조정과 선택적 망각	자연어 요청으로 능력을 더하거나 제거하고 다른 기능의 손상 줄이기
재귀적 자기 개선	자신이나 후속 모델의 가중치를 읽고 결함을 직접 수정하기
구조가 다른 모델의 결합	서로 다른 모델 사이에 추론 등 기능 회로를 옮기기

다음 단계: 공통 벤치마크와 하네스로 성능·비용·재현성을 검증

Q&A

질문을 받겠습니다

Weight-Space Programming