고려대학교 KOREA UNIVERSITY

Natural Language Processing & Artificial Intelligence

연구실 세미나

고려대학교 NLP&AI 연구실
발표자: 손준영

◆ Introduction

NeurIPS 2024

Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Boshi Wang[♠] **Xiang Yue**[◇*] **Yu Su**[♠] **Huan Sun**[♠]
[♠]The Ohio State University [◇]Carnegie Mellon University
{wang.13930,yue.149,su.809,sun.397}@osu.edu

NeurIPS 2025

How do Transformers Learn Implicit Reasoning?

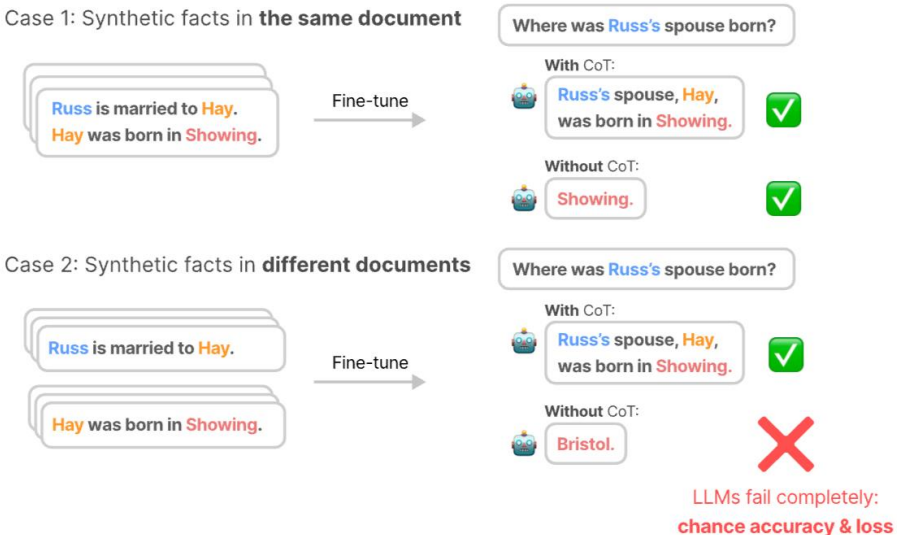
Jiaran Ye[♠†] **Zijun Yao**[♠†] **Zhidian Huang**[♠] **Liangming Pan**[♡‡] **Jinxin Liu**[♠]
Yushi Bai[♠] **Amy Xin**[♠] **Weichuan Liu**[◇] **Xiaoyin Che**[◇] **Lei Hou**[♠‡] **Juanzi Li**[♠]
[♠]DCST, BNRist; KIRC, Institute for Artificial Intelligence, Tsinghua University, *China*
[♡]MOE Key Lab of Computational Linguistics, Peking University, *China* [◇]Siemens AG, *China*
{yejr23, yaozj20}@mails.tsinghua.edu.cn
{houlei, lijuanzi}@tsinghua.edu.cn

Transformer가 Implicit Reasoning을 학습하면서 보이는 내부/회로/기하학적 특징에 관한 연구들

◆ Introduction

Implicit Reasoning이란

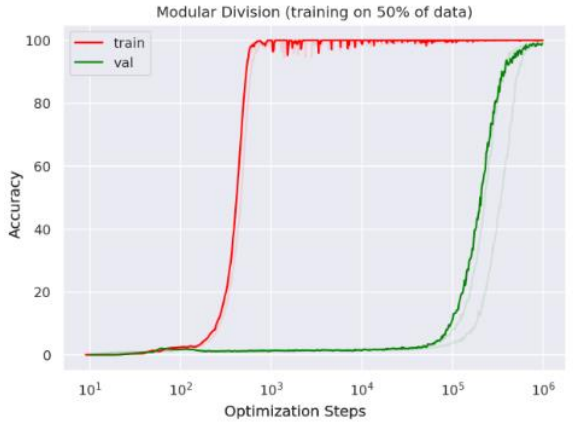
- 명시적인 Reasoning Text 없이 Implicit하게 모델 내부 연산으로 지식을 활용/추론하는 것
- 최근 연구들을 통해서 LLMs는 Implicit Reasoning에서 구조적 한계를 보이고 있음
  → 그냥 Explicit Reasoning (CoT) 하면 되는 거 아님? ㅋㅋ
  ➔ 비용 문제. 만약 Implicit Reasoning 능력도 같이 향상시킬 수 있으면 CoT의 불필요한 추론을 줄일 수 있음

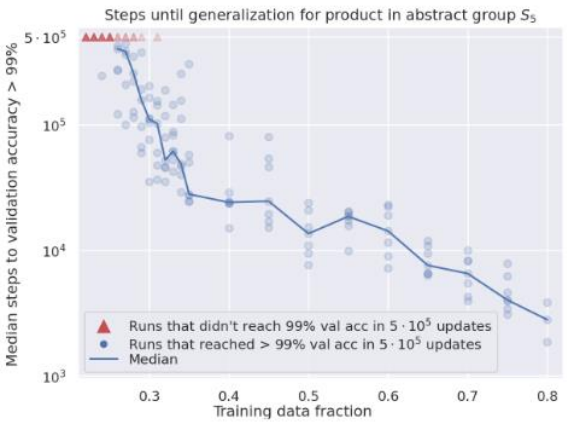


◆ Introduction

Grokking

- 신경망이 오랜 학습 끝에 갑자기 패턴을 'grok' (이해)하여 검증 성능이 무작위 수준에서 완전한 일반화 수준으로 급상승하는 현상[1] 으로, 모델 내부의 추론 회로가 일반화 가능한(더 단순한) 패턴을 포착하게 되는 현상
- 이러한 현상이 Implicit Reasoning에서도 관찰이 될까?
- 관찰이 된다면, 모델 내부적으로 어떤 특징을 보일까?





★	a	b	c	d	e
a	a	d	?	c	d
b	c	d	d	a	c
c	?	e	d	b	d
d	a	?	?	b	c
e	b	b	c	?	a

1) Power, Alethea, et al. "Grokking: Generalization beyond overfitting on small algorithmic datasets." arXiv preprint arXiv:2201.02177 (2022).

◆ Introduction

Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Boshi Wang[♠] **Xiang Yue**[◇*] **Yu Su**[♠] **Huan Sun**[♠]
[♠]The Ohio State University [◇]Carnegie Mellon University
{wang.13930,yue.149,su.809,sun.397}@osu.edu

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Grokked Transformers: 연구 질문

- Q1. Implicit reasoning에서도 grokking처럼 일반화 전환이 나타나는가?
- Q2. 나타난다면, 그 전환은 어떤 내부 회로/표상 변화로 구현되는가?
- Q3. 왜 어떤 추론은 systematic OOD generalization이 되고, 어떤 건 깨지는가?

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Experimental Protocol: 학습 / 평가

1) Model / Training / Evaluation

Model: decoder-only Transformer (GPT-2 style), 8 layers (기본)

Train: all atomic facts (head, relation, tail) + subset of inferredID (head, rel_1, rel_2, tail)

Test: 유사 분포 추론 데이터(inferredID), 분포 외 추론 데이터(inferredOOD)

- **Goal**: ID는 "규칙 학습", OOD는 "systematicity(분포 바뀐 지식에 규칙 적용)"
- **Key procedure (for grokking)**:
  train accuracy가 포화된 이후에도 훨씬 더 오래 학습(long training)하여
  "overfit 이후 일반화가 갑자기 올라오는 전환(grokking)"을 관찰

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Experimental Protocol: Tasks

1) Composition

atomic: (h, r1) → b

inferred: (h, r1, r2) → t

구조: h→b→t (bridge chaining 필수)

2) Comparison

atomic: (e, a) → v

inferred: (a, e1, e2) → $a_{<or=or>}$

구조: v1/v2 병렬 조회 → 비교

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Experimental Protocol: 분석 도구

1) Logit Lens

- 각 레이어의 hidden state를 출력 공간(logits)으로 투영해,
- 그 시점에 모델이 어떤 토큰(bridge entity b, relation r2 등)을 '떠올리는지' 분석
- 예: Composition에서
  - r1 위치에서 bridge b가 보이는가?
  - r2 위치에서 r2 정보가 보이는가?

➔ "표상(Representation) 관점: 지금 이 위치가 무엇을 담고 있나?"

2) Causal Patching (기여: "실제로 쓰이나?")

- 특정 레이어·토큰 위치의 activation을 교체(Patching)해서, 최종 예측이 얼마나 바뀌는지 확인
- 어떤 위치가 예측에 인과적으로 중요하면, 패치했을 때 성능/로그릿이 크게 변한다.

  ➔ "기여(Causality) 관점: 담고 있는 정보가 실제로 정답을 만드는 데 쓰이나?"

3) Why these tools? (Grokking = 회로 전환을 보기 위해)

- grokking은 단순히 test accuracy가 오르는 현상이 아니라,
- 내부에서 shortcut 경로 → 2-hop 경로로 '회로가 재구성'되는 전환으로 해석할 수 있음
- Logit lens로 "회로의 재료(표상)가 생기는지"를 보고,
- causal patching으로 "그 재료가 예측에 실제로 연결되는지(기여)"를 확인함

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

실험 세팅: “atomic vs inferred” and “ID/OOD”

- 학습은 atomic(전부) + inferred(ID 일부)로 하고, 평가는 “보지 않은 inferred”를 ID/OOD로 나눠 본다.
  - **Atomic facts**: 기본 사실(엣지). 예: (subject, relation, object) 같은 1-hop 트리플
    - | *atomicID ∪ atomicOOD (atomic은 전부 학습에 포함)*
  - **Inferred facts**: atomic들에 latent rule R(예: 2-hop composition)을 적용해서 만든 “추론된 사실(엣지)”
    - | *train_inferredID ⊂ inferredID (inferred는 ID에서 일부만 샘플링)*
    - | *test_inferredID: atomicID에서 유도된 inferred 중 train에서 보지 않은 inferred → “규칙을 배웠나?”*
    - | *test_inferredOOD: atomicOOD에서 유도된 inferred → “분포가 바뀌어도 규칙을 적용하나? (= systematicity)”*
- “We prepare two separate sets of atomic facts: atomicID and atomicOOD.”

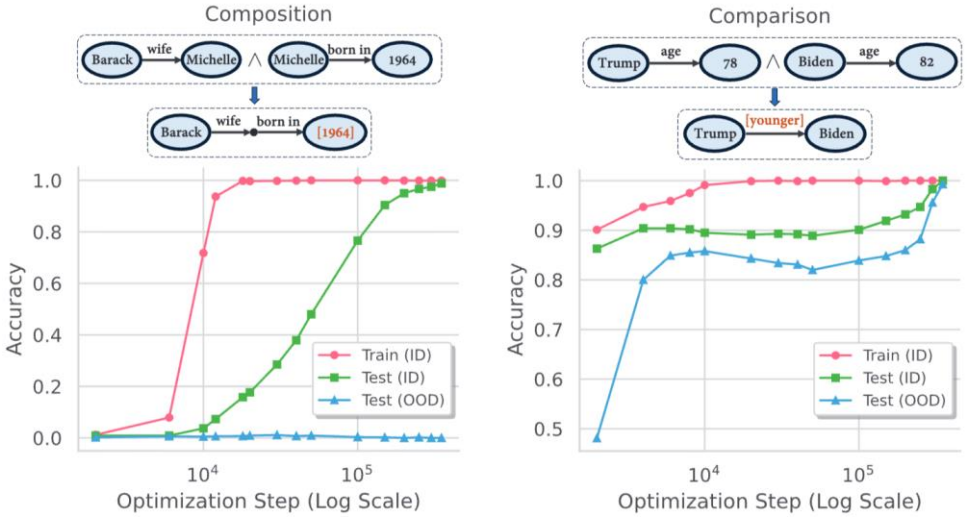
  → atomic을 ID/OOD 두 묶음으로 따로 만들어 놓고, 그 위에서 규칙으로 inferred를 생성해서 일반화의 ‘종류’를 분리해서 측정

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Transformer는 implicit reasoning을 '배우긴' 하는데, composition은 OOD에서 깨지고 comparison은 된다.

- **관찰 1) 두 태스크 모두 ID generalization은 grokking처럼 상승 가능**
- **관찰 2) 하지만 OOD에서 차이가 발생**
- **Composition OOD에서는 일반화 실패**
  . *Composition은 Multi-hop reasoning을 의미 (h -> r1 -> (latent) b -> r2 -> t)*
- **Comparison OOD에서는 일반화가 성공하였음**
  . *Comparison은 entities의 attribute에 대한 values를 비교하는 작업*

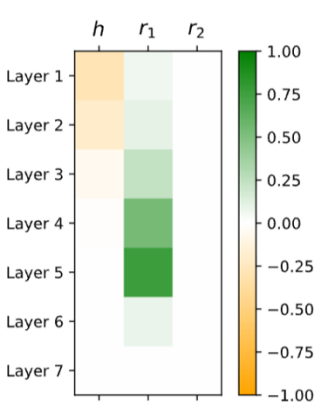
Q: 왜 OOD에서만, 그리고 composition에서만 깨질까? → 회로로 들어가보자



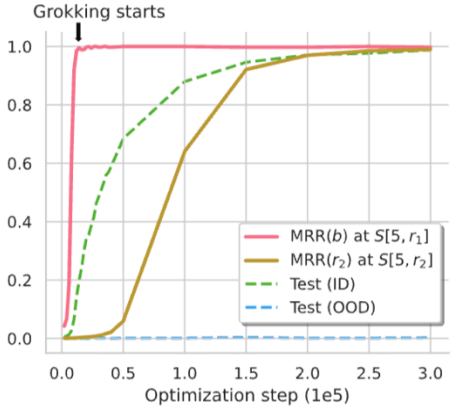
◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

Grokking = “회로 전환”

- (b)는 grokking 동안 causal patching으로 인과적 기여 변화량을 측정한 것으로, 초록색이 진할 수록 인과적 기여 변화량이 큰 것임
- (c)는 Logit Lens로 각 인과적 기여가 가장 큰 층/토큰 위치에서의 각 중요한 토큰(bridge entity, relation token)을 Grokking 중에 충분히 담고있는지 분석한 자료



(b)



(c)

[결과 해석]

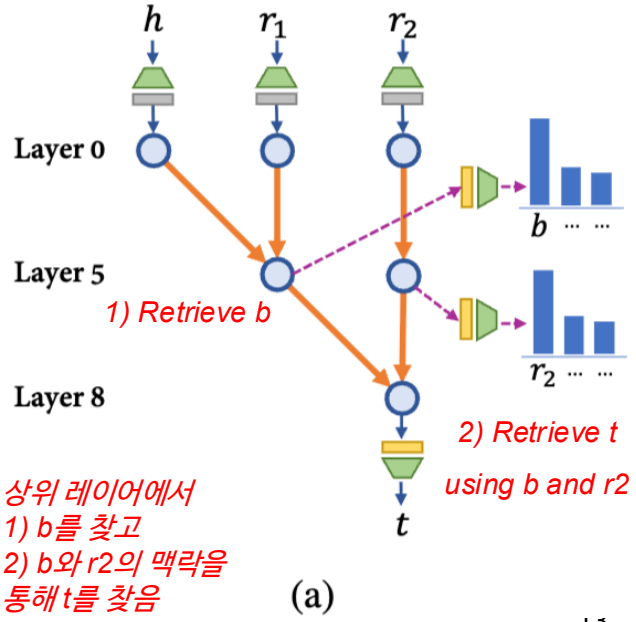
- (c): r_1 위치에서 b, r_2 위치에서 r_2가 점점 선명해짐
- (b) 해당 위치에서 최종 예측에 기여하는 인과적 기여가 grokking 동안 증가함

➔ b가 일부 관측이 되더라도(c, 빨간선), 예측에 강하게 쓰이는 연결은 약하고, 상위 계층에서 b와 r2를 연관지을 수 있을 때(c, 갈색선) Test 성능이 상승하기 시작함

◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

왜 composition은 OOD에서 깨지나? = “지식이 저장되는 층이 갈라짐”

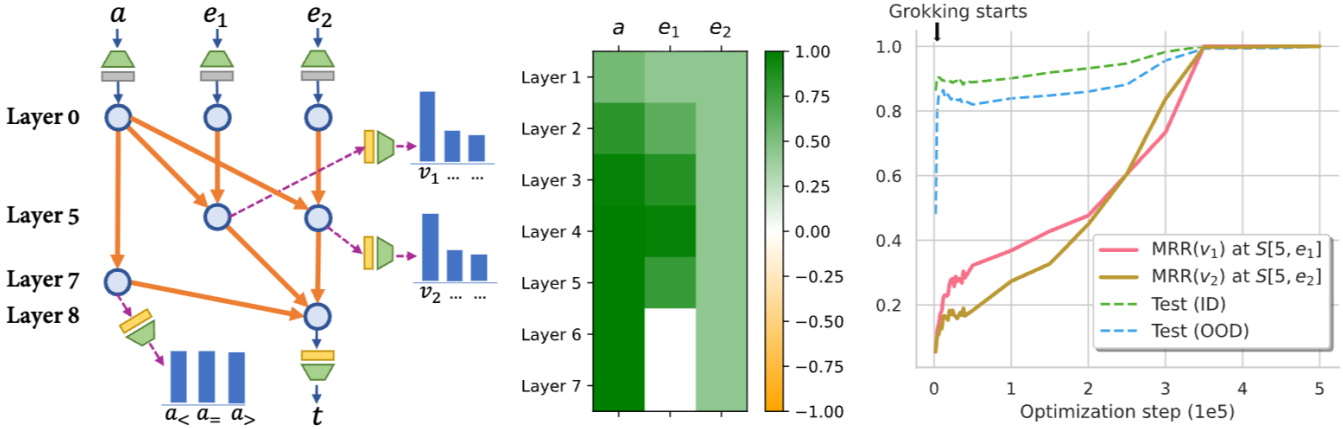
- Composition 회로는 ‘second-hop용 지식’을 상층에 따로 복제해 쓰는데, OOD atomic은 그 상층에 저장될 이유가 없어 OOD에서 깨짐
- 하층에 atomic 저장 + 상층에 second-hop atomic 복제 구조
- 그런데 OOD에서는 fact가 atomic 형태로만 관측되고, “second hop 위치에 등장”하지 않음
- 그래서 모델은 상층에 OOD atomic을 저장할 인센티브가 없고, second hop 질의 시 상층에서 못 찾음 → OOD 실패
- (짧은 해석) systematicity는 ‘규칙’만이 아니라, 규칙이 참조하는 ‘지식이 어디에 저장되느냐’에 걸려있다.



◆ Grokked Transformers are Implicit Reasoners: A Mechanistic Journey to the Edge of Generalization

왜 comparison은 OOD까지 되나? = “두 fact를 하층에서 병렬 조회”

- Comparison은 atomic을 “하층 레이어에만 저장”하고 두 값을 병렬로 꺼내 상층에서 비교만 하므로, ID/OOD가 동일 경로로 처리되어 systematicity가 나옴



◆ Introduction

How do Transformers Learn Implicit Reasoning?

Jiaran Ye[♠†] **Zijun Yao**[♠†] **Zhidian Huang**[♠] **Liangming Pan**[♡‡] **Jinxin Liu**[♠]
Yushi Bai[♠] **Amy Xin**[♠] **Weichuan Liu**[♢] **Xiaoyin Che**[♢] **Lei Hou**[♠‡] **Juanzi Li**[♠]

[♠]DCST, BNRist; KIRC, Institute for Artificial Intelligence, Tsinghua University, *China*
[♡]MOE Key Lab of Computational Linguistics, Peking University, *China* [♢]Siemens AG, *China*

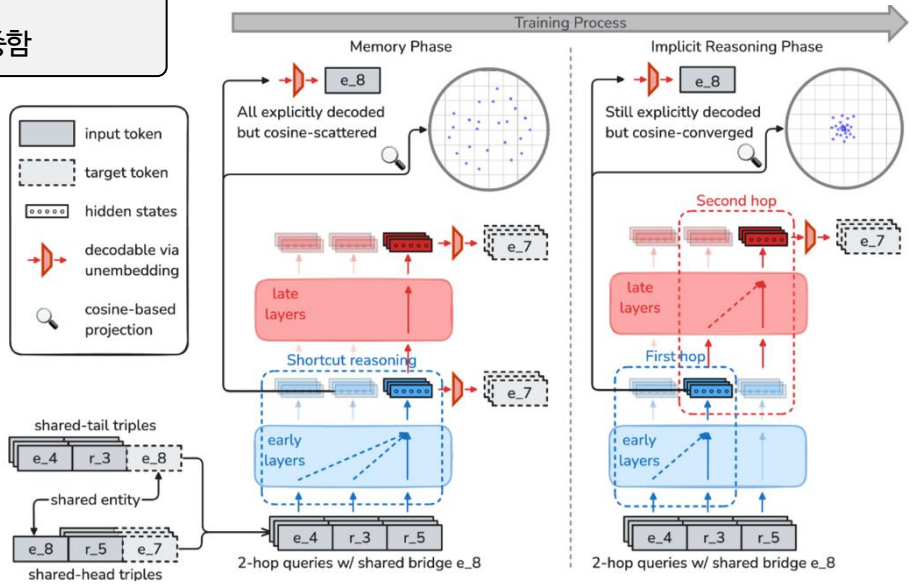
{yejr23, yaozj20}@mails.tsinghua.edu.cn
{houlei, lijuanzi}@tsinghua.edu.cn

◆ How do Transformers Learn Implicit Reasoning?

중간 엔티티가 디코드되느냐(decodability)는 reasoning을 설명하지 못함

Grokked Transformers는 회로가 왜 깨지는지를 보여줬고,
이 논문은 중간 엔티티 표현이 '재사용 가능'하게 정리되는지를 검증함

- "중간 엔티티 b가 디코드 가능한가?"는 충분조건이 아님
- 핵심: b의 표현이 문맥 across queries에서
  '같은 의미로 재사용'되도록 정리되는가



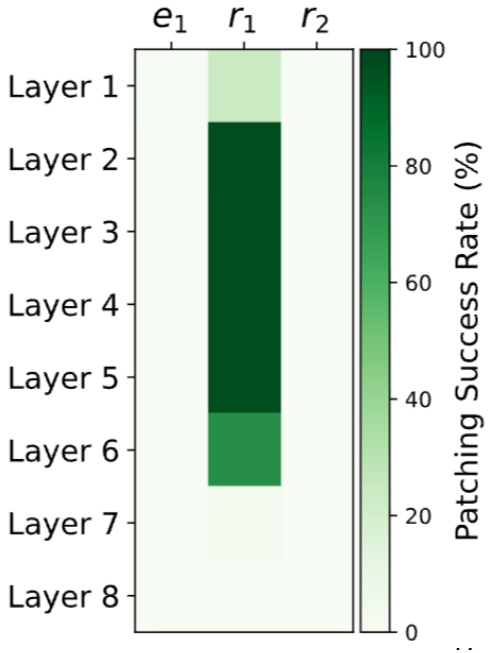
◆ How do Transformers Learn Implicit Reasoning?

도구 1: Cross-query Semantic Patching = “의미가 옮겨지는 위치” 찾기

- 출력에 영향을 주는 위치가 아니라, 다른 쿼리로 b의 의미가 ‘전달(transfer)’되는 위치를 찾음
- 설정: 쿼리 A에서 얻은 hidden state를 쿼리 B의 특정 레이어/토큰에 끼워넣고, 정답이 바뀌면 “의미 전달” 성공 (구조적으로 비슷한 쿼리 A, B 가정)
- 결과: 중간 레이어 + 특정 토큰 위치(논문에서 주로 r1 위치/anchor로 삼는 구간)에서 transfer가 가장 잘 됨
- 이 anchor에서 hidden state를 모아, 다음 장의 “cosine lens/지표”를 계산

소스 쿼리: (e1, r1, r2). r1 위치의 hidden vector h_src 를 뽑아서
타깃 쿼리: (e5, r6, r7). 동일한 레이어·토큰 위치에 h_src 를 삽입(patch).
만약 타깃의 예측이 원래 경로 r7(r6(e5)) 대신, 삽입된 정보에 의해 r7(r1(e1)) 쪽으로 바뀐다면, h_src 는 “중간 엔티티 b = r1(e1)”의 의미를 담고 있고, 그 의미가 다른 쿼리에도 transfer(재사용)됨을 뜻함

➔ 인과적 기여도 + 의미적 재사용 (alignment)

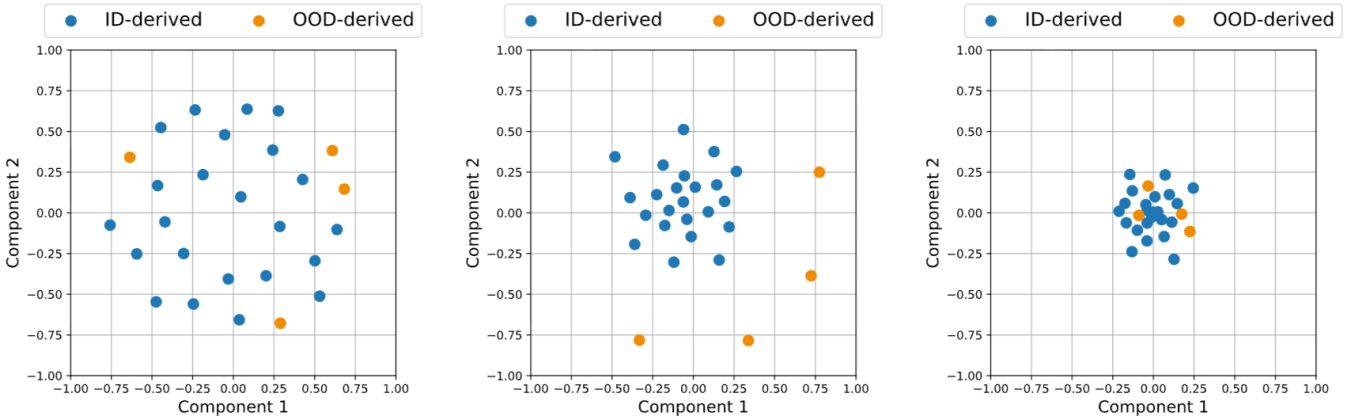


◆ How do Transformers Learn Implicit Reasoning?

도구 2: Cosine Lens = “클러스터링이 생기면 성공”을 시각화/수치화

- 좌측(Phase I): 같은 b 점들이 여러 군데 흩어짐(산개) → b가 문맥에 얽혀 있음(비표준)
- 중간(Phase II): ID 점들이 한 군데로 모임(ID cohesion) → b 표현에 문맥과 상관없이 표준화
- 우측(Phase III): OOD 점(다른 색)이 ID 덩어리로 이동해 합류(OOD alignment)

  → 분포가 달라도 first hop이 만들어내는 b의 표현 포맷이 ID와 호환



(a) Phase I (b) Phase II (c) Phase III

◆ How do Transformers Learn Implicit Reasoning?

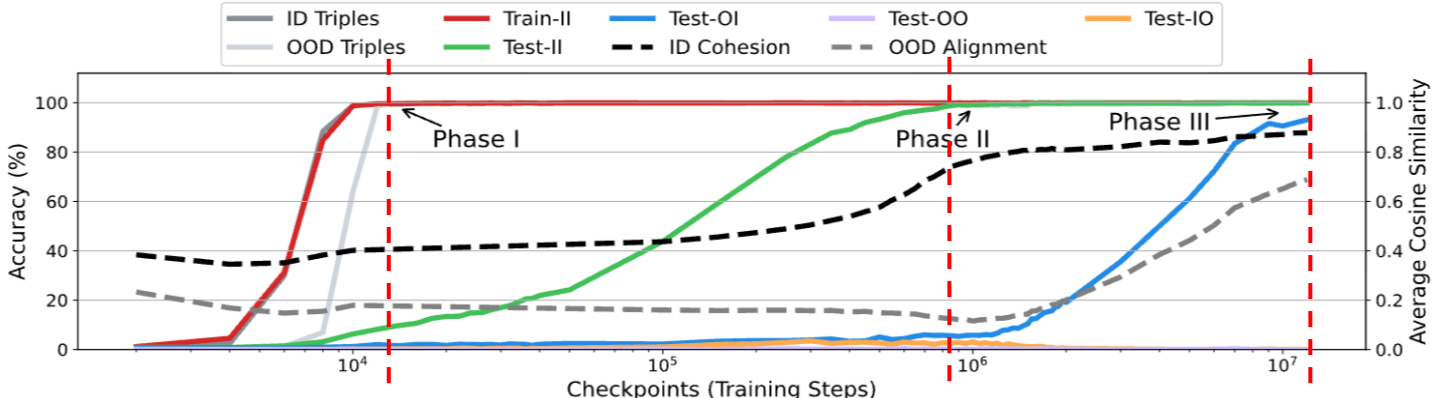
학습 3단계(암기→ID→교차분포) 진행이
ID cohesion↑ → OOD alignment↑와 함께 움직임

- Phase I: 성능 낮음 + ID cohesion/OOD alignment 낮음 ➔ 단순 Memorization phase
- Phase II: ID cohesion 상승과 함께 In-distribution 일반화(예: Test-II) 발생 (Grokking)
- Phase III: OOD alignment 상승과 함께 Cross-distribution 일반화(Test-OI) 발생

  ** Test-IO나 Test-OO에서는 성능 향상이 없는데, 이 것은 second-hop mapping 또한 중요하기 때문..*

➔ 암튼 중간 표현 표준화(b)는 Cross-distribution 일반화의 중요한 필요 조건임

+ ID Cohesion과 OOD Alignment를 통해서 Grokking의 단계를 세분화할 수 있었음



◆ How do Transformers Learn Implicit Reasoning?

Second-Hop Generalization Requires Query-Level Training Match

- 두 번째 홉 일반화는 단순히 사실(atomic triple)을 아는 것만으로 해결되지 않음. 모델은 "중간 표현(latent e2)을 받아서 두 번째 관계 r2를 적용해 최종 토큰 e3로 매핑하는 방식"을 해당 구조(토큰 위치, 레이어 흐름 등)에서 직접 학습해야 함
- 첫 홉의 경우, 입력(prefix) 공유나 ID atomic triple의 감독이 r1 위치의 표현을 특정 subspace로 고정(anchoring)하면서 OOD→ID 같은 부분적 일반화를 가능하게 하지만, 두 번째 홉은 그 중간 표현을 후속 레이어/연산이 어떻게 읽고 변환하는지(즉 “r2를 수행하는 회로”)를 훈련 데이터에서 직접 본 적이 있어야 함
- 즉, 두 번째 홉을 수행하려면 **(1) 중간 표현들이 안정적으로 군집되어 있어야(ID cohesion)**, 그리고 **(2) 그 군집을 받아 후속 관계로 매핑하는 회로(두 번째 홉 처리)가 훈련에서 그 역할을 직접 본 경험을 통해 형성**되어야 한다. atomic 사실만으로는 (2)를 충분히 학습시키지 못한다고 언급함

감사합니다.