

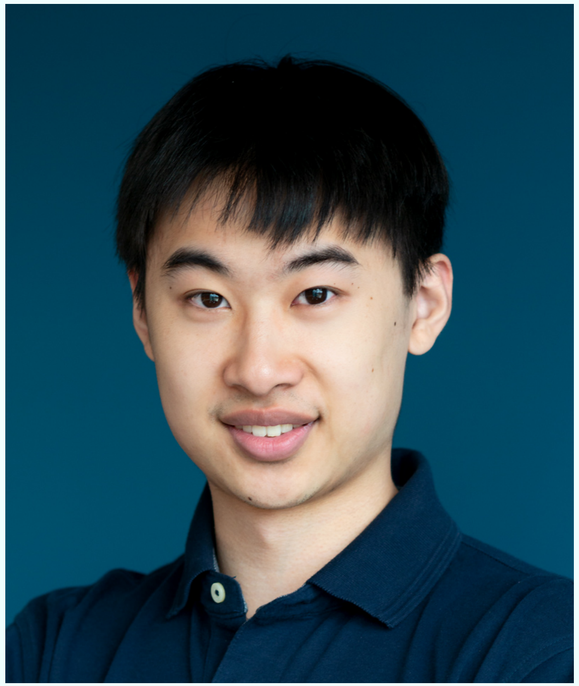
CS 25: Transformers United V6

Instructors: Steven Feng & Karan Singh

April 2, 2026 — Instructor Lecture

Stanford University

Instructor Bio



Steven Feng

Email: syfeng@stanford.edu

Website: styfeng.github.io

Twitter: @stevenyfeng

4th-year CS PhD student

Upcoming:

- Anthropic Fellows Program

Previously:

- Carnegie Mellon & UWaterloo
- Amazon, NVIDIA, & Contextual AI

Research Interests:

- Data-centric scaling laws for reasoning and generalization
- Efficient reasoning in SLMs and data-limited regimes
- Human and cognitively-inspired learning signals and evaluation for foundation models
- Long-term memory and continual learning

For Fun:

- Stanford Piano Society!
- https://piano.stanford.edu/
- IG: @stanfordpiano

Stanford University

Instructor Bio



Karan Singh

Email: karanps@stanford.edu

Website: karanps.com

3rd-year EE PhD student

Research Interests:

- Computer Vision & Neuroscience (fMRI, Ultrasound)
- Language Models & Efficiency (Curriculum Learning, RAG, etc.)
- Currently → Foundation Models for Neuroimaging

Previously:

- EE Undergrad @ Cal Poly SLO
- Post-Baccalaureate Researcher @ Stanford Radiology (RSL)

For Fun:

- 🎹 Piano, 🏸 Badminton, 🏋️ Working Out, 👨‍🍳 Cooking

Thanks to Our Sponsors

Making Transformers United V6 possible!







Frontier Lunch Club

CS25 students meet founders and researchers off-stage.

RECRUITING PIPELINE

Get added to AGI House's talent network. Top AI labs and startups in our network hire from this community.

A SEAT AT THE TABLE

Small group dinners and lunches with founders and researchers from the AGI House network.

$1,000 PRIZE

The best project from the group wins $1,000. We'll share more at the first event.

Application-based · Spots are limited

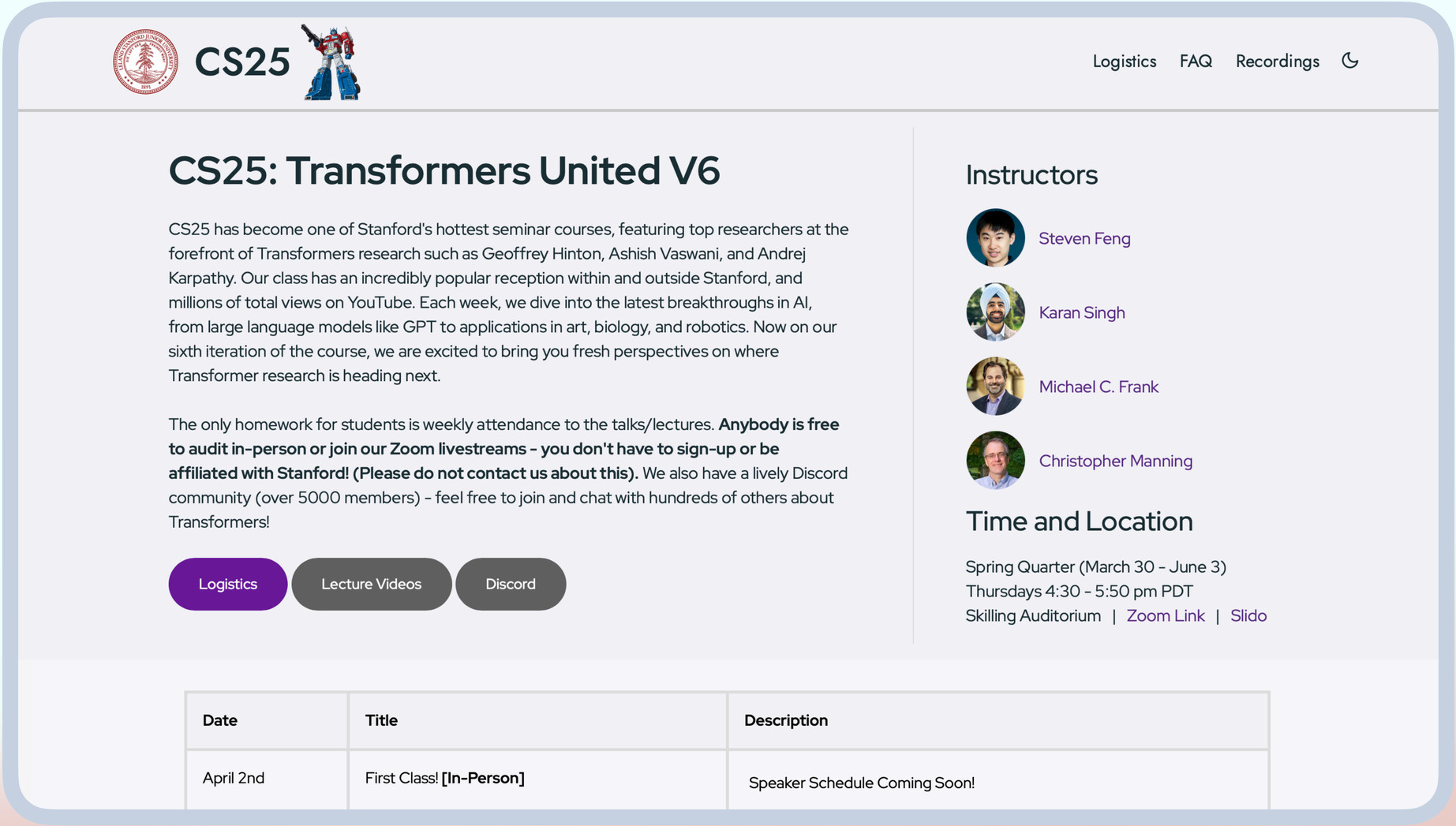
APPLY NOW

MongoDB.

AGI HOUSE

Course Logistics

Course Website



cs25.stanford.edu

Course Logistics

- Lectures: Thursdays 4:30 - 5:50 pm PDT (Skilling Auditorium and Zoom)
- Enrollment 190 student max, also waitlist [Axess]
- Only homework: attendance! In-person required for weeks with in-person speakers, Zoom otherwise
  - Allowed 3 unexcused absences - track attendance using a Google Form (and Zoom audience list) starting next week
- Ask questions during and after presentation (chat in Zoom or use slido, no voice on Zoom!)
  - Submit questions to **sli.do** (code: SU_CS25)

Throughout the Quarter

- Join our Discord server! Link on website!
- Zoom Livestream (Public / Open Auditing) & Recordings (Released 2 Weeks Later).
- Social Events (TBD) + Potential 1-on-1 Networking with Speakers!
- In-person questions: please raise your hand - editors will block out your voice!
- Your image or voice may be incidentally captured by video cameras in the back of the room
  - An editor will remove any student appearances before videos are published. Contact us with any questions.

Disclaimers: Auditing & Zoom

- Zoom meeting has limit of 500 participants
- Another reason enrolled students should come in person!
- Do not unmute yourself on Zoom! Any questions/concerns? Send in chat
- In-person auditors: please give seats to enrolled students who have priority
- Inappropriate behavior will result in blacklist from the course

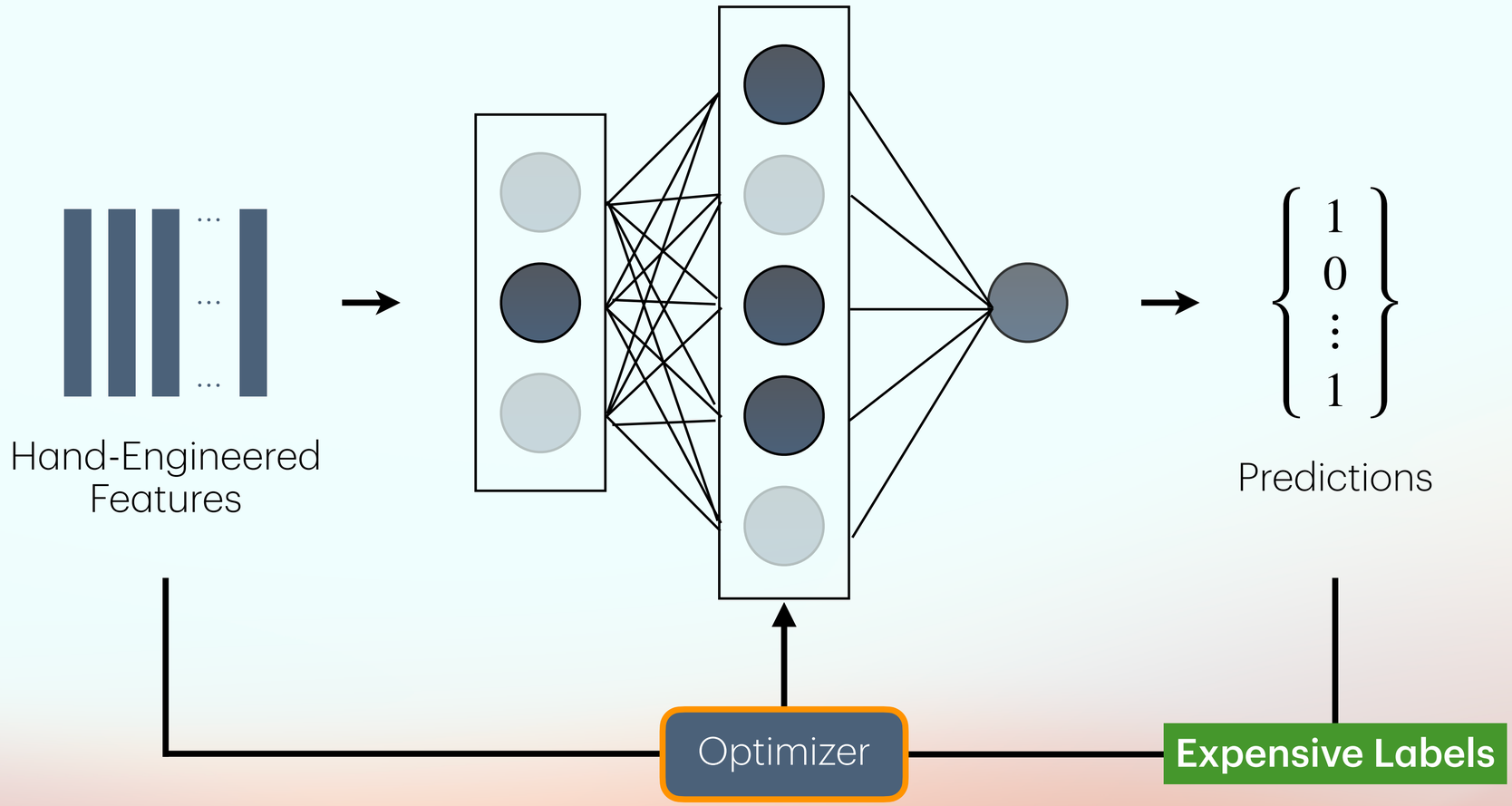
  (and maybe other consequences with Stanford)

What you'll take away from CS25

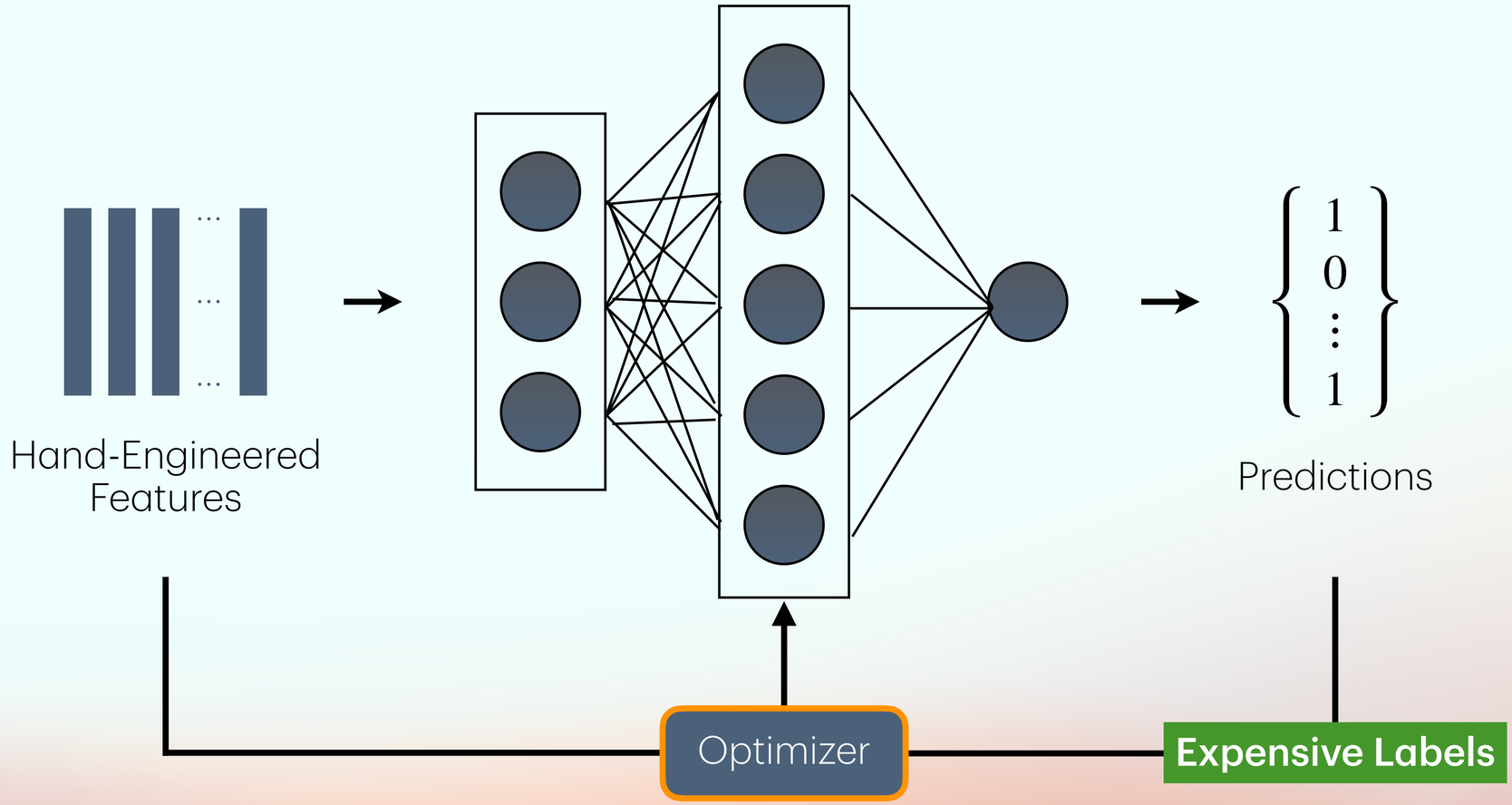
- 🌟 A better understanding of how transformers ***really*** work – from attention to scaling
- 🧠 Cutting-edge applications in language, vision, biology, robotics, and more
- 🧪 Live exposure to frontier research from speakers shaping the field
- 🚀 Innovative methods driving the next generation models
- ⚠️ Key limitations, open problems, and the future of AI (assuming that includes humans)

Attention Is All You Need? An Intro to Transformers

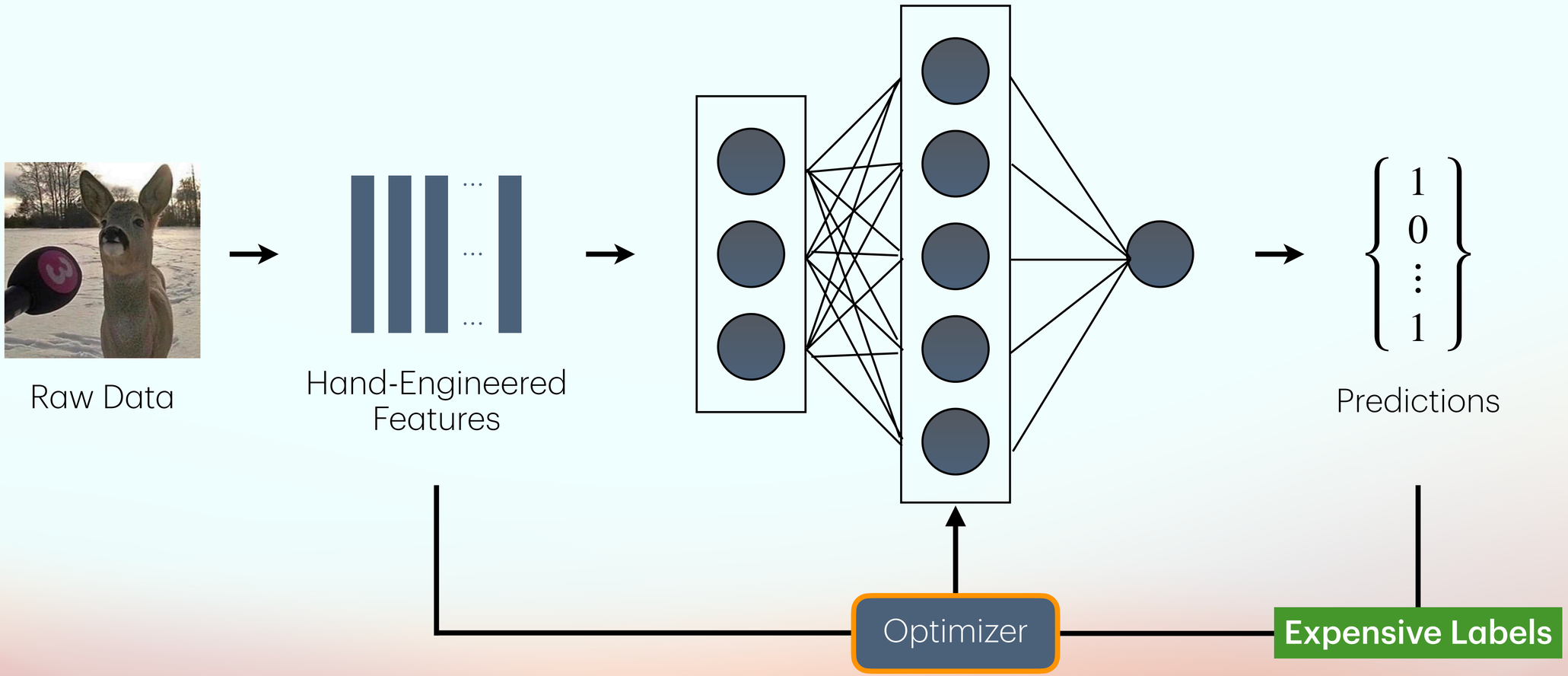
Early ML / Pre-2012



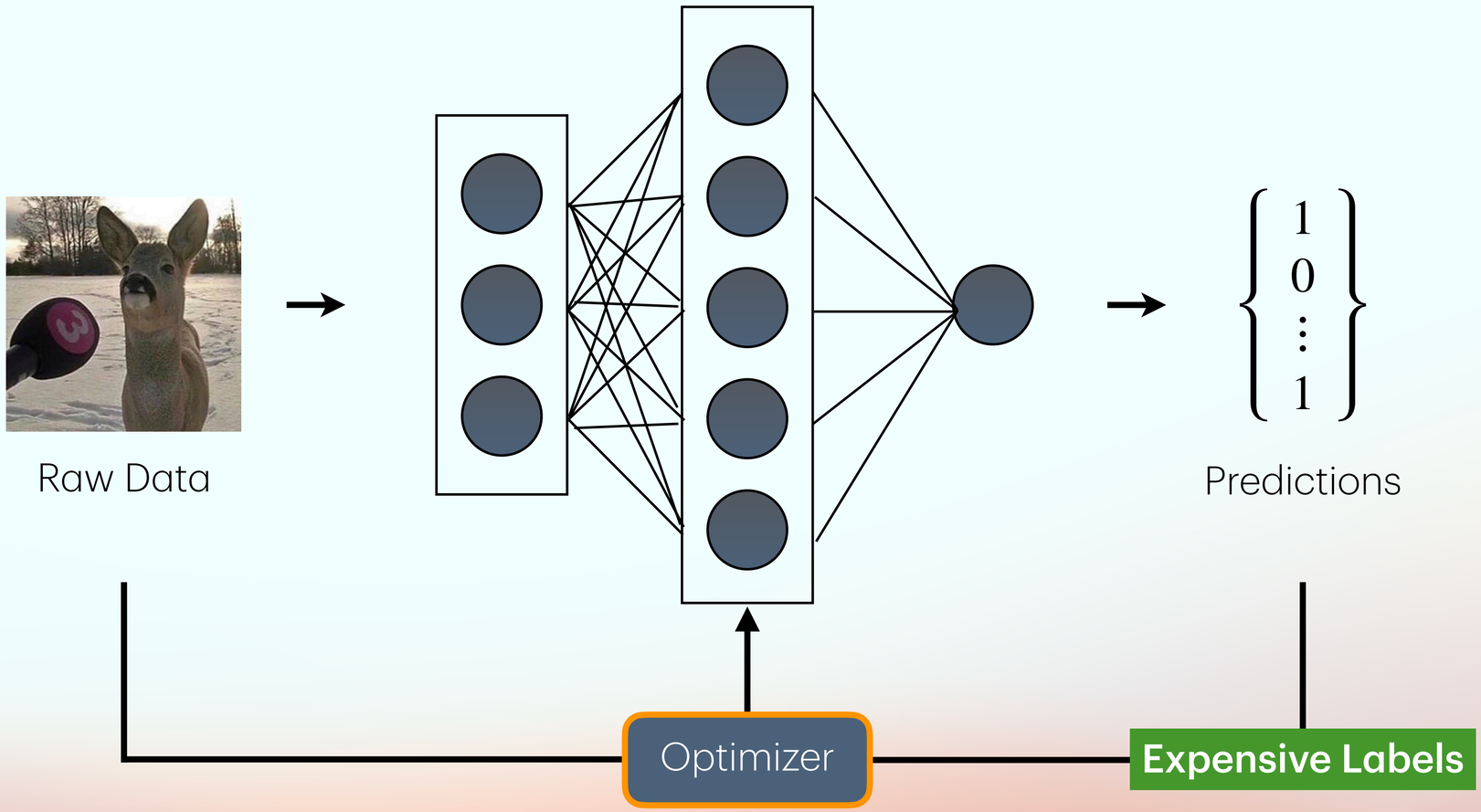
Supervised Deep Learning



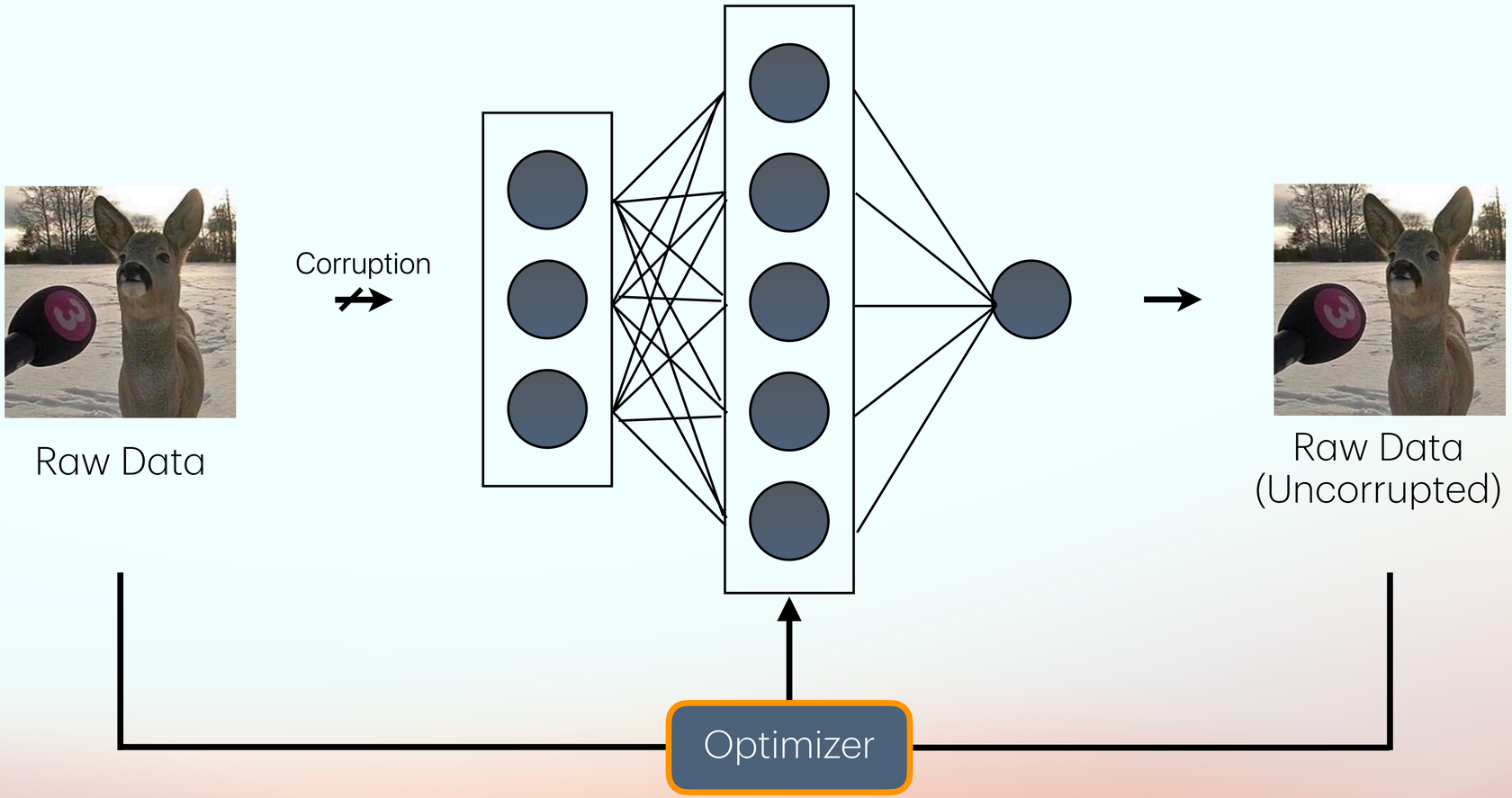
Supervised Deep Learning



Supervised Deep Learning

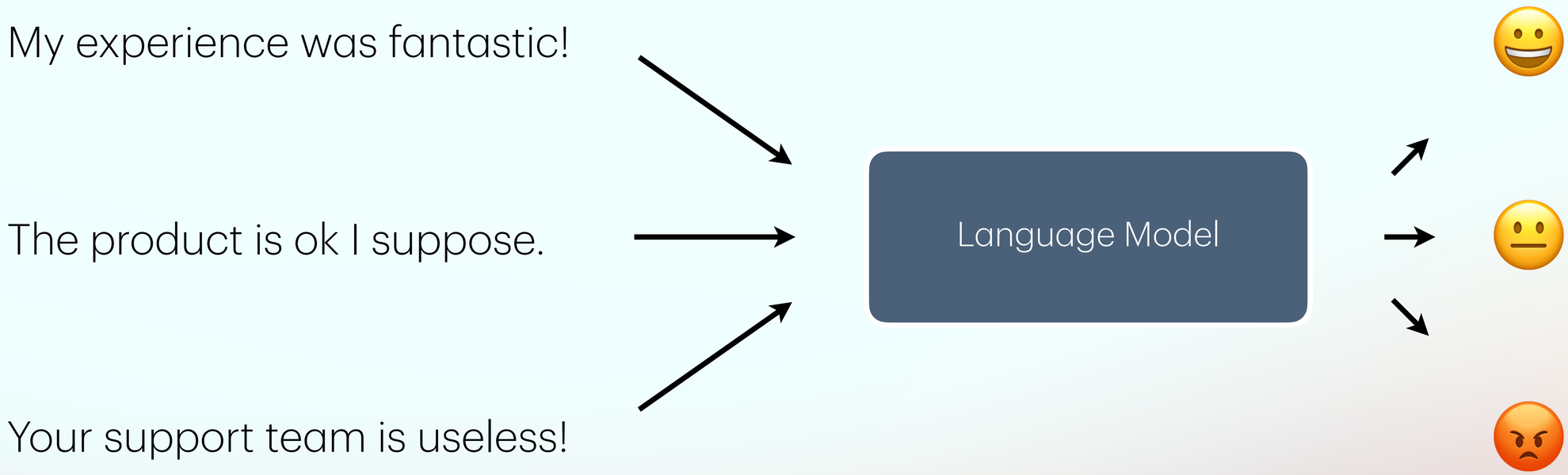


Self-Supervised Learning



Case Study: Language

ML work in language followed a similar trajectory, moving from simple, feature-based tasks like sentiment analysis to the highly effective casual language modeling framework.



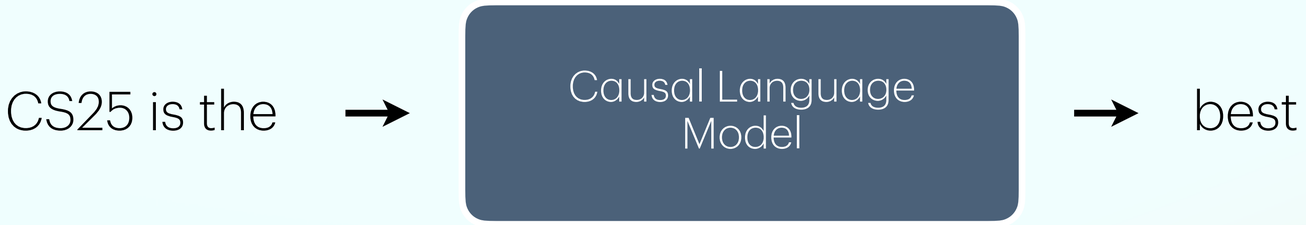
Case Study: Language

Language is sequential and context-rich, where each word or sentence naturally constrains and predicts the next.



Case Study: Language

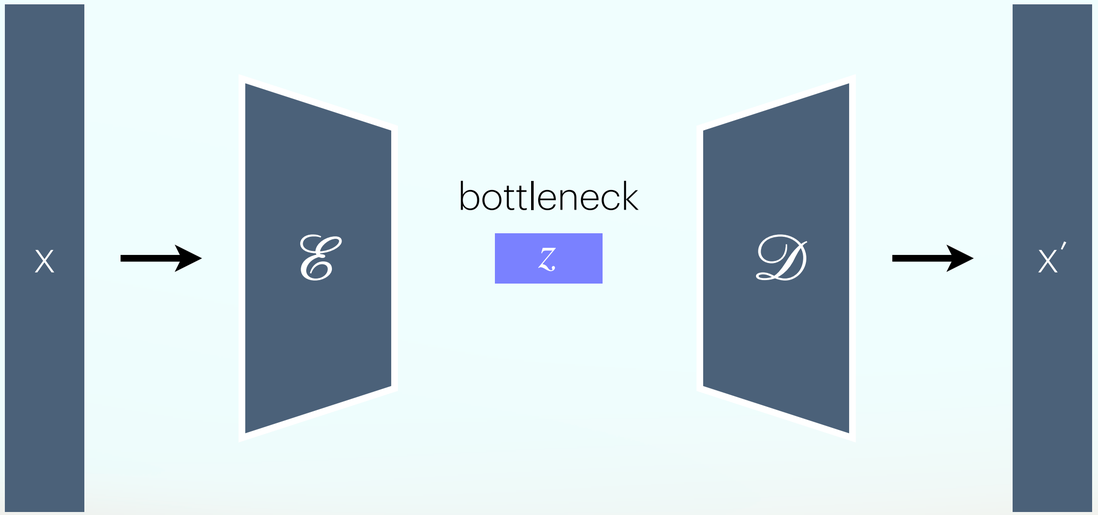
Language is sequential and context-rich, where each word or sentence naturally constrains and predicts the next.



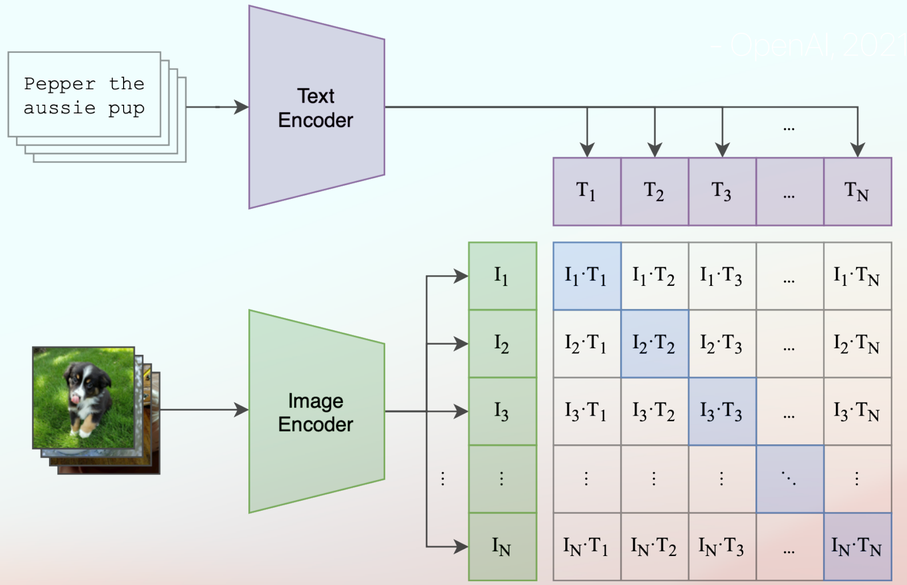
Case Study: Vision

Like you might've guessed, vision was no different, and followed a similar trajectory and gradual transition to self-supervised methods.

Autoencoders



Contrastive Learning



– OpenAI, 2021

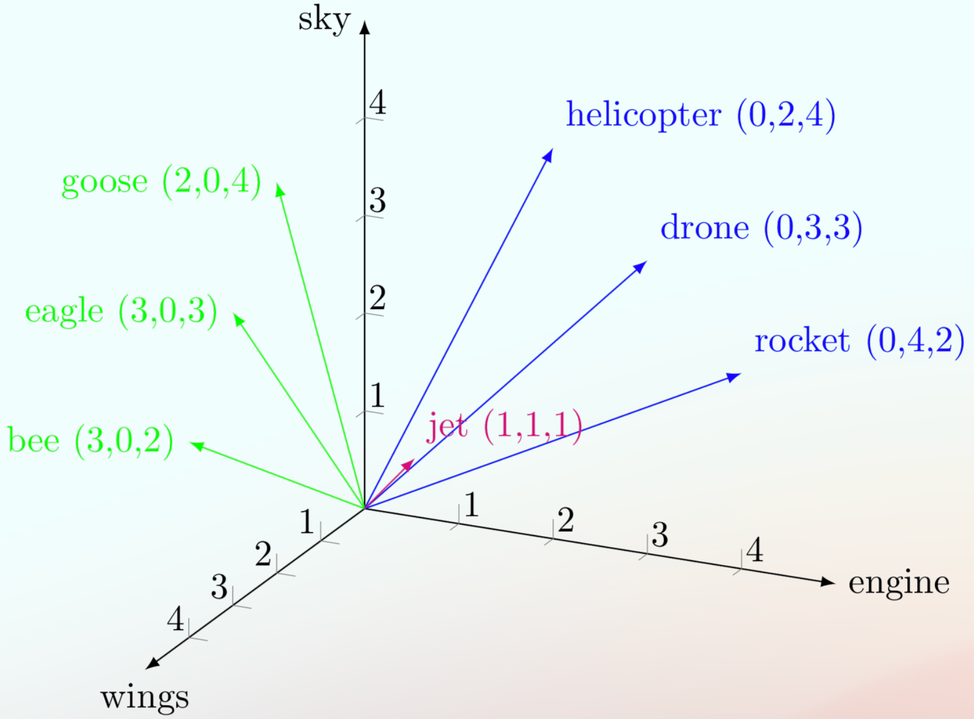
Case Study: Vision

Vision is spatially structured rather than sequential, so models rely on techniques like random masking to learn, but its moderate dimensionality makes effective modeling feasible.



Word Embeddings

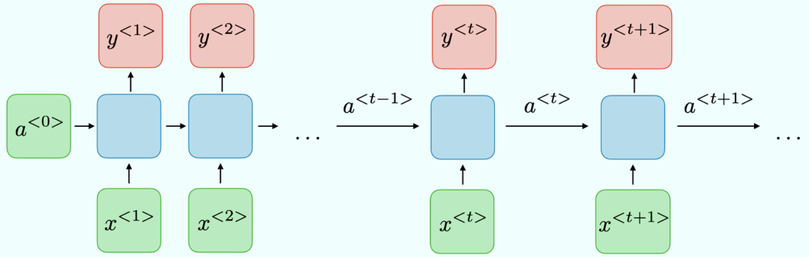
- Words → Dense Vectors in High-Dimensional Space
- Semantic Similarity: “king” - “man” + “women” ≈ “queen”
- Enable arithmetic, visualization, and learning
- Classic Methods: Word2Vec, GloVe, FastTex



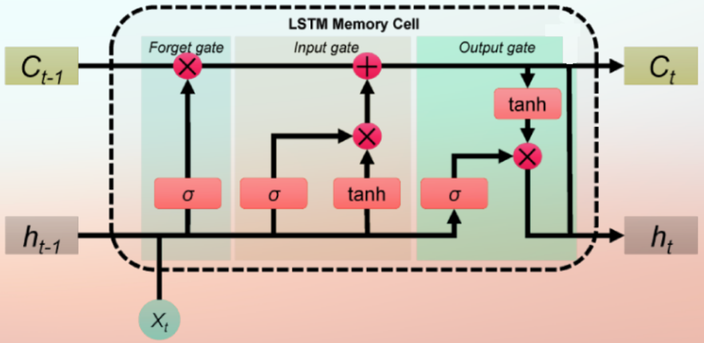
- *But*… static embeddings have limitations
- Contextual embeddings solve this problem

Pre-Transformer: RNNs

- RNNs (Recurrent Neural Networks): Sequence models that process inputs step-by-step while maintaining a hidden state that carries information forward through time.

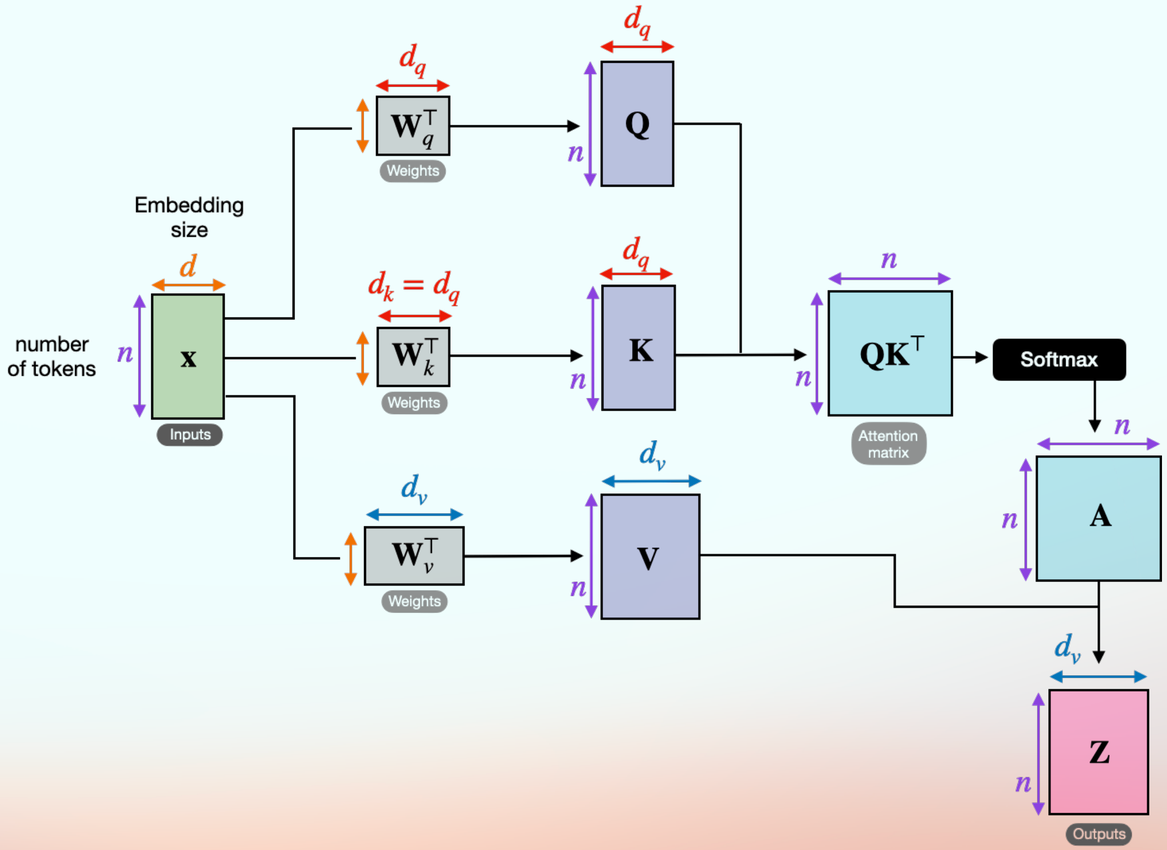


- LSTMs (Long Short-Term Memory networks): A gated variant of RNNs designed to better preserve long-range dependencies through input, forget, and output gates.



Transformers in 2 Min

- Self-attention: learns what to focus on for each token (words broken up)



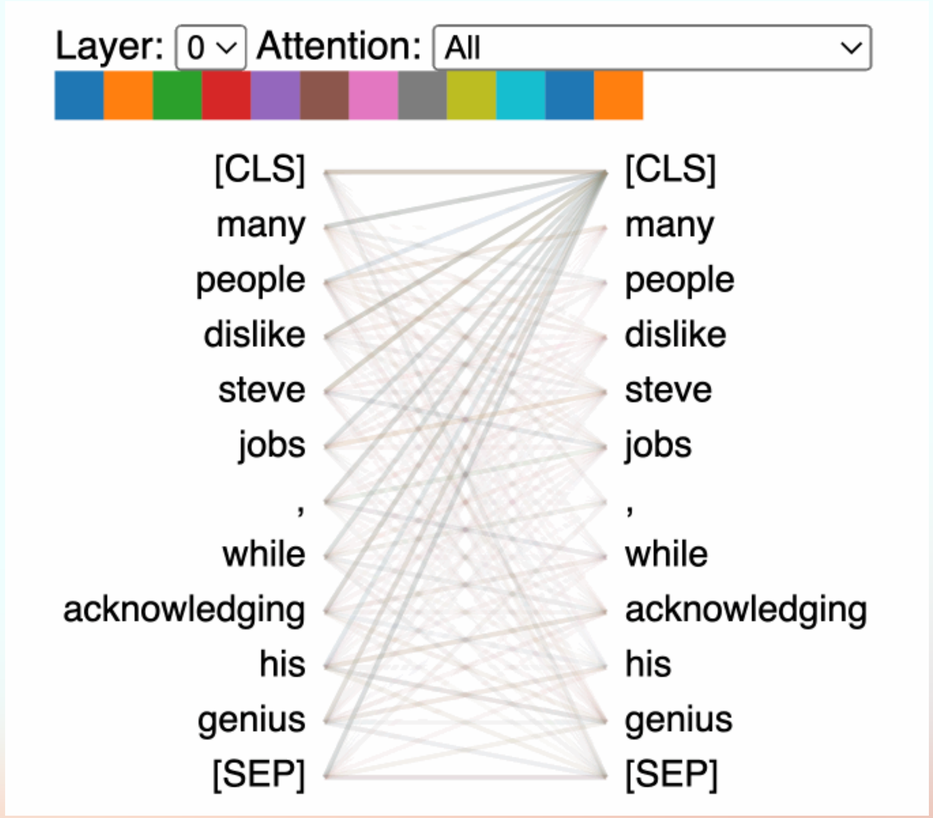
Analogy for QKV

- Imagine you're in a library looking for information on a specific topic (query)
- Each book in the library has a summary (key) that helps identify its content
- Once you find a match between your query and a summary, you access the book to get the detailed information (value) you need
- In attention, we do a “soft match” across multiple values / get info from multiple books (“book 1 is most relevant, then book 2, then book 3, etc.”)

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

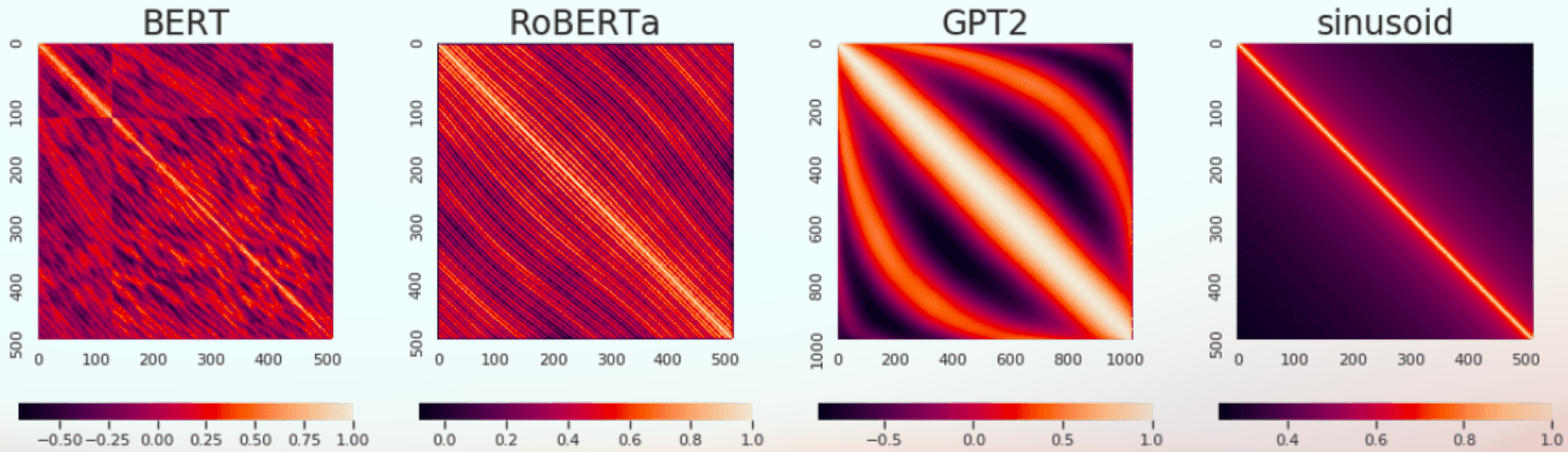
Transformers in 2 Min

- Self-attention: learns what to focus on for each token (words broken up)



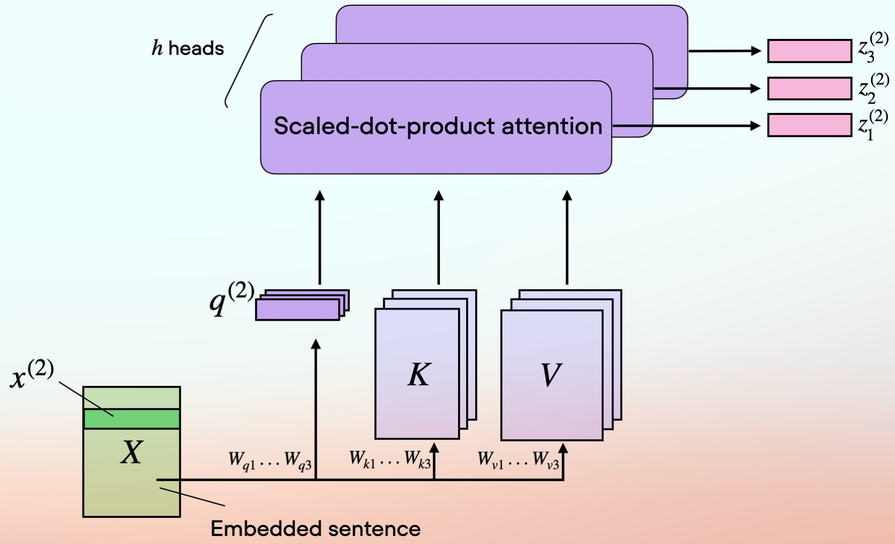
Transformers in 2 Min

- Self-attention: learns what to focus on for each token (words broken up)
- Positional encoding: adds order to sequences

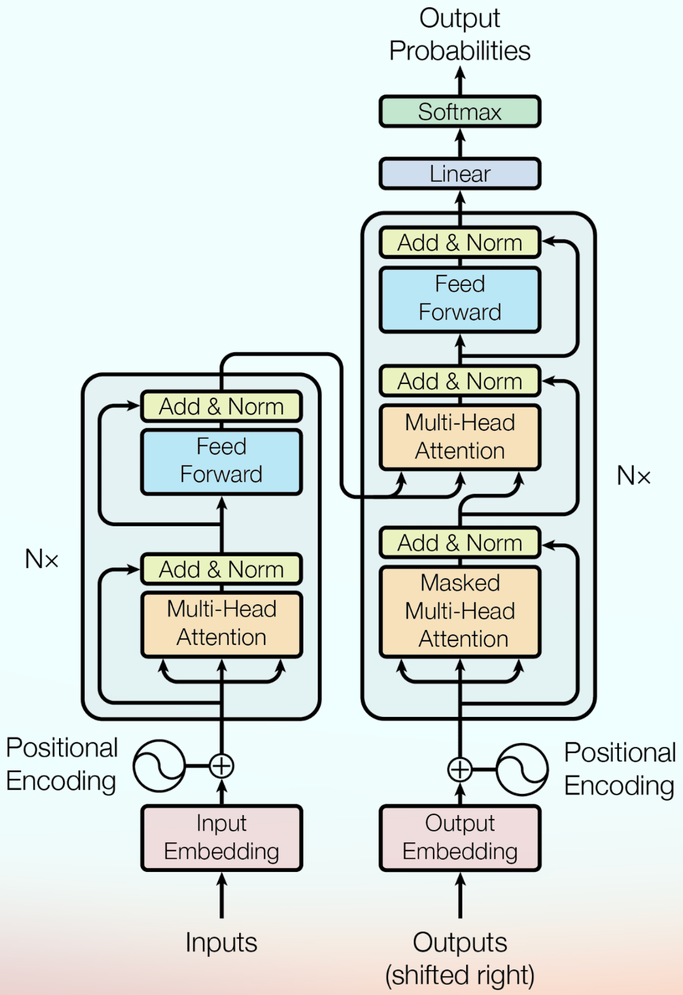


Transformers in 2 Min

- Self-attention: learns what to focus on for each token (words broken up)
- Positional encoding: adds order to sequences
- Multi-head attention: more heads/parameters means we can capture diverse relationships



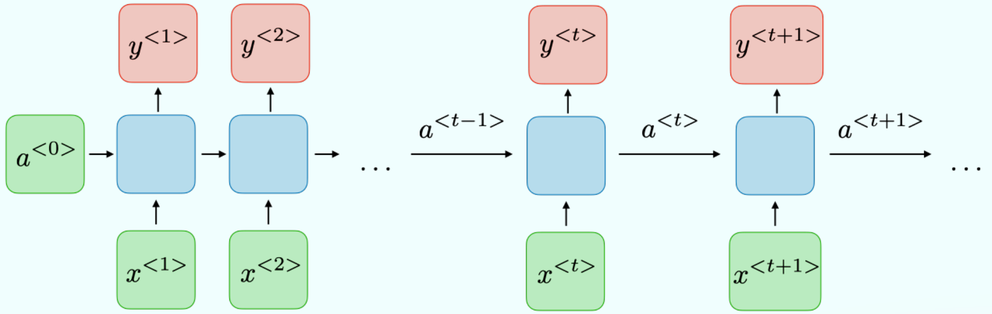
The Final Transformer

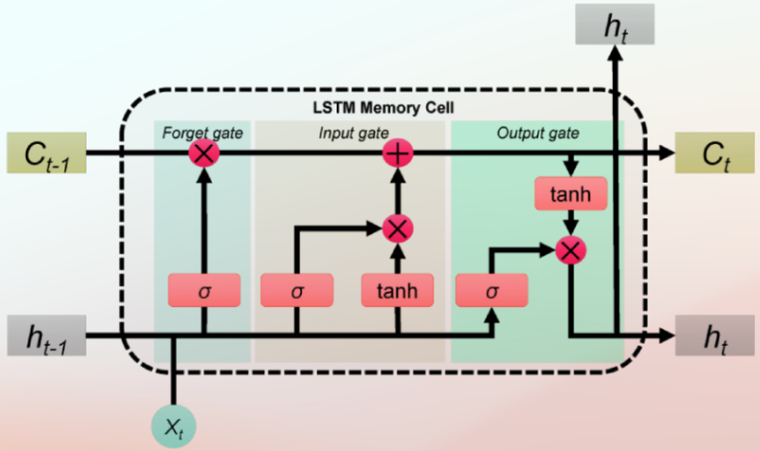




RNNs vs. Transformers

- Attention allows for parallelism; RNNs by design must process sequences one step at a time
- Long-contexts are handled better by transformers; RNNs (and LSTMs at sufficient scale) suffer from vanishing gradients and “diluted hidden states”
- Attention allows memory access to all previous tokens in the context; RNNs are limited to implicit memory via their hidden state.





Transformers Today

- **LLMs**: GPT-5, DeepSeek, Claude, Gemini, Llama, etc.
- **Vision**: CLIP, Segment Anything, DINOv2
- **Speech**: Whisper, Voicebox
- **Biology**: AlphaFold-3
- **Video**: Sora, Runway, Pika, Seedance 2.0
- **Robotics**: RT-2, Code as Policies
- **And so much more!**

Large Language Models

- Scaled up Transformer architecture: billions of parameters + heavy computational cost (time, GPUs)
- Pretrained on massive amounts of “general” textual data (e.g. web corpus)
- Training objective of “next token prediction”: $P(W_{t+1}|W_t,W_{t-1},...,W_1)$
- Emergent abilities as they scale up (e.g., chain-of-thought reasoning)
- Larger general ones: “plug-and-play” with few or zero-shot learning
  - Train once, then adapt to other tasks without needing to retrain
  - In-context learning and prompting

Pretraining & Data Importance

The Impact of Data for Artificial Intelligence

- Pretraining: initial stage of training an LLM from scratch (random weights)
- Goal: obtain a general (unspecialized) level of intelligence and capabilities
- Data is the fundamental “fuel” for all ML and AI models to learn
- Arguably the most important aspect of ML training, particularly pretraining
- Especially for LLM: pretrained on billions or trillions of tokens
- Because of its importance, how do we maximally leverage it?

Smarter Data Strategies for LM Training

- *How can we train language models more effectively with smart data strategies?*
  - *Project 1:* **Baby-Scale Language Learning from Individual Environments** — how do differences in real child language environments impact model learning at small scales?
  - *Project 2:* **Bilingual Language Modeling under Controlled Exposure** — can LMs learn multiple languages simultaneously without interference, and does input structure matter?

Smarter Data Strategies for LM Training

- *How can we train language models more effectively with smart data strategies?*
  - *Project 3:* **Curriculum-Guided Layer Scaling for Language Model Pretraining** — can scaling data complexity alongside model depth improve downstream performance?
  - *Project 4:* **Scaling Laws for RAG-Considerate Pretraining** — a study on optimal allocation of data between pretraining and retrieval-augmentation

Small Language Models & BabyLM

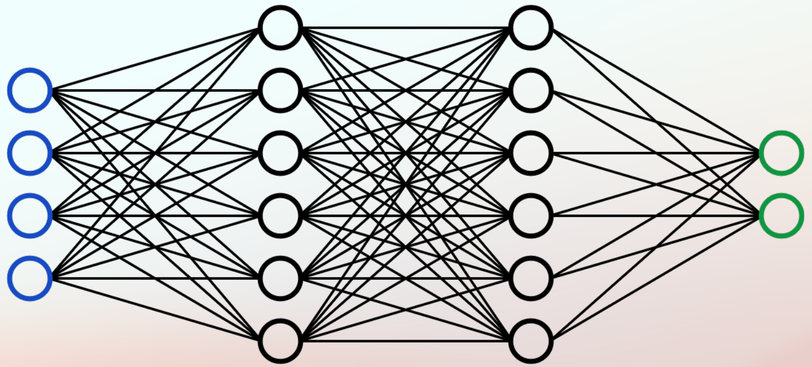


vs.





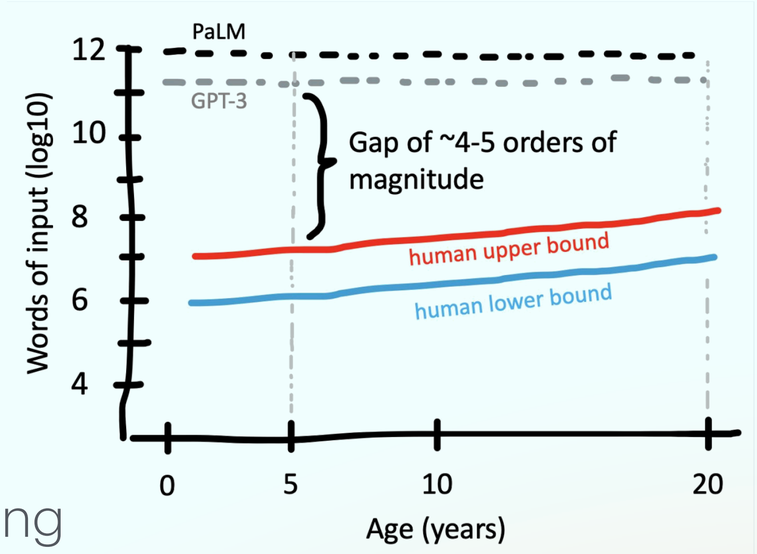
vs.



What Are Some Key Differences?

- Continual learning vs. single pass
- Interactive, goal-based vs. autoregressive
- Grounded in continuous multimodal/multisensory data

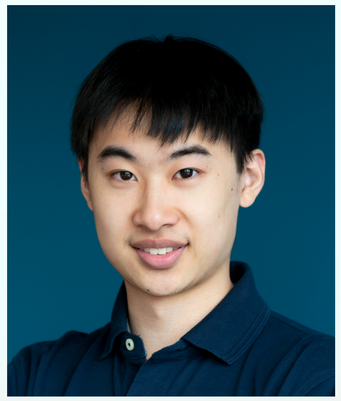
  (e.g. BabyView) vs. text-only or text + image
- Hierarchical/compositional-based learning vs. statistical learning
- Focus here: **potential language data differences**
- Also, thoughts/ideas from Michael C. Frank's tweet and paper

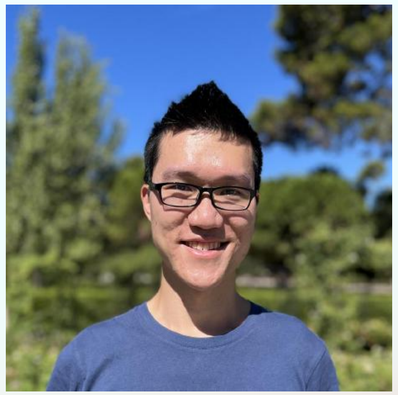


Why Investigate Small Models & Small Data?

- Greatly improve efficiency of training and using LLM
- Open up new doors and potential use-cases
- Improved interpretability and alignment
- Enhanced open-source availability
- Techniques discovered here can possibly be applied to larger-scales
- Greater understanding of cognitive models of humans and how we learn language efficiently

Baby Scale: Investigating Models Trained on Individual Children's Language Input







Steven Y. Feng Alvin W.M Tan Michael C. Frank

Stanford University

Can language models learn from one child's input?

- Human children learn from far less language than modern LLMs
- But most small-scale LM work still uses fixed or mixed datasets
- We wanted to ask:
  - How models scale at child-scale data sizes
  - How much performance varies across individual families
  - What properties of the input make some child environments better for learning
- BabyView lets us train "synthetic learners" on the language input from individual children's homes

Experimental Setup

- Data: BabyView, egocentric recordings + transcripts from 20 English-speaking families
- Family datasets range from ~1k to 725k tokens
- Compare against larger synthetic TinyDialogues data for broader scaling analysis
- Models trained: GPT-2 & GPT-BERT
- Evaluate: Zorro (grammar), Word Similarity & COMPS (semantics), EWoK (world knowledge)
- Conditions:
  - Individual-family models
  - Size-matched family mixtures
  - All-families model

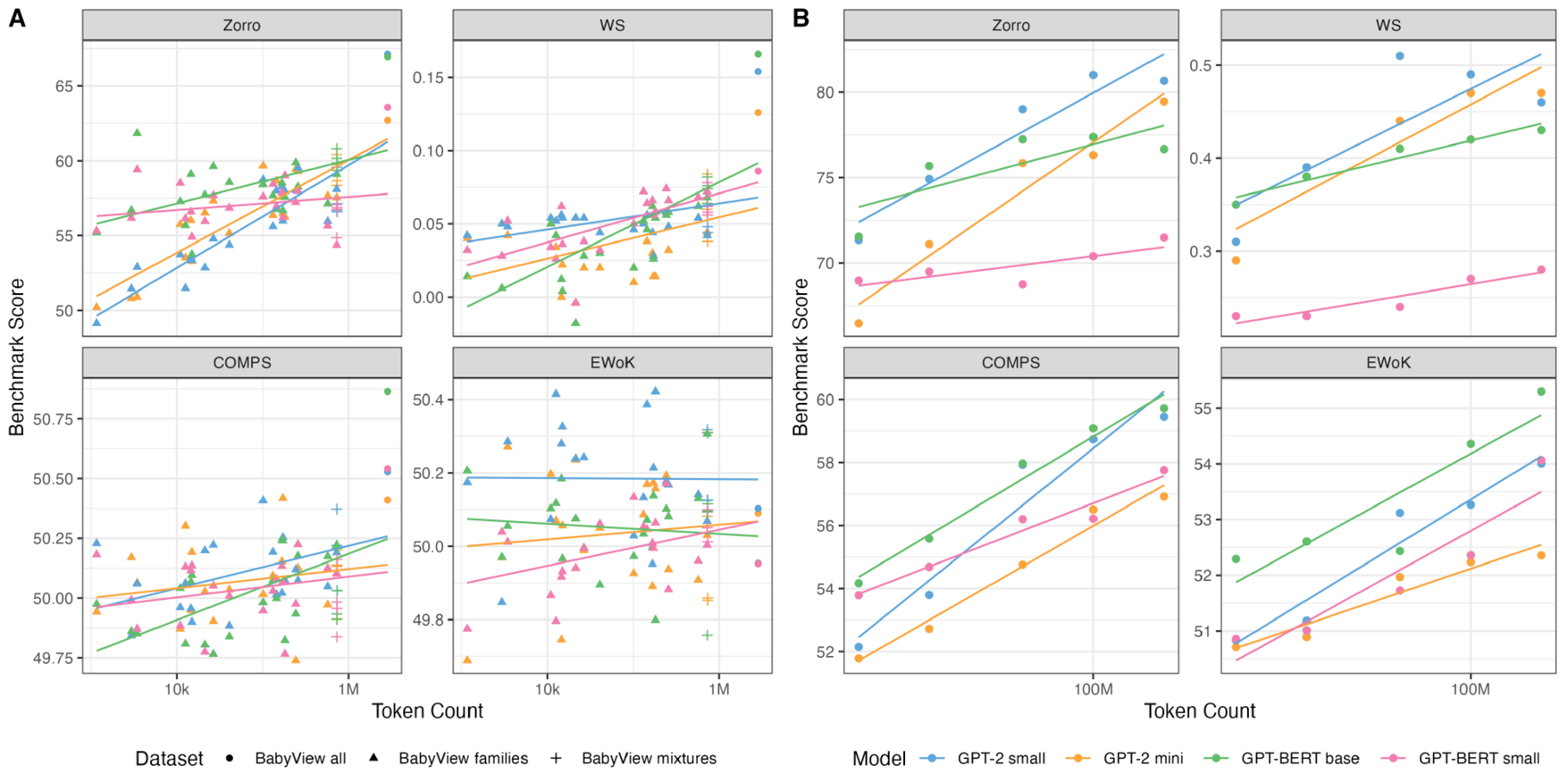


Figure 2: Performance scaling of (A) BabyView experiments and (B) TinyDialogues experiments across all four benchmarks. Each color denotes a separate model; different markers show different experimental conditions. Lines of best fit show average linear relationships.

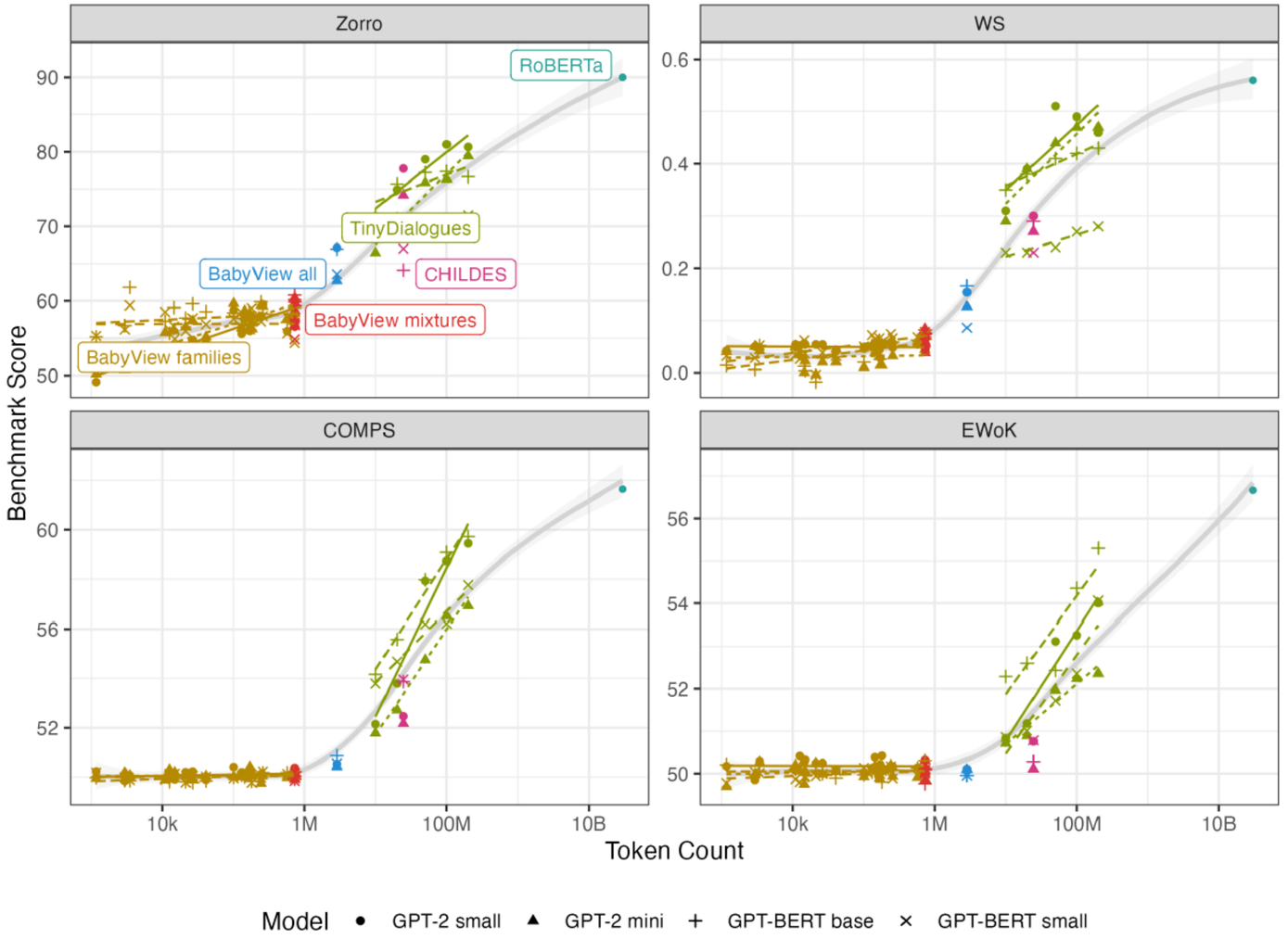


Figure 1: Broader performance scaling with data quantity across all four benchmarks. Each color denotes a separate dataset, with different markers showing different models. Gray curves show spline-smoothed average relationships across all data.

Scaling Results

- Scaling depends on the task (depends on the task)
- TinyDialogues shows more classical scaling across all four benchmarks
- Baby scale data fits into broader scaling trends
- Dataset size not enough to explain differences at baby scale
- Families produce noticeably different model performance
- Gains come less from raw lexical diversity and more from:
  - Contextual diversity
  - Discourse diversity
  - Speaker / interactional diversity

What Predicts Better Performance Beyond Size?

- Comprehensive linguistic analysis of BabyView data
- Strong predictors were not just token count
- Top features included:
  - Bigram mutual information
  - POS bigram entropy
  - Caregiver / child POS-tagged token coverage
  - Child-to-caregiver semantic pair count
  - Cumber of conversations
  - Distributional divergence from other datasets

What Predicts Better Performance Beyond Size?

- Overall, better-performing datasets seem to be:
  - More structured
  - More diverse
  - Richer in interaction and coverage
- Lines up with child language research emphasizing quality, not just quantity

Takeaways

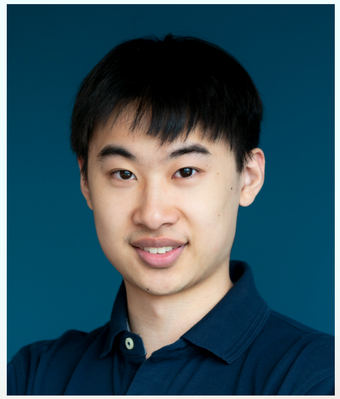
- Child-scale LM learning is possible, but task-dependent
- Grammar scales better than semantic and world knowledge
- Variation across individual children's language environments
- Input composition, structure, coverage matter most
- Studying child-scale data may help both:
  - Build more data-efficient LMs
  - Understand human language acquisition better



github.com/styfeng/babyscale-LM

Bringing Up a Bilingual BabyLM: Investigating Multilingual Language Acquisition Using Small-Scale Models



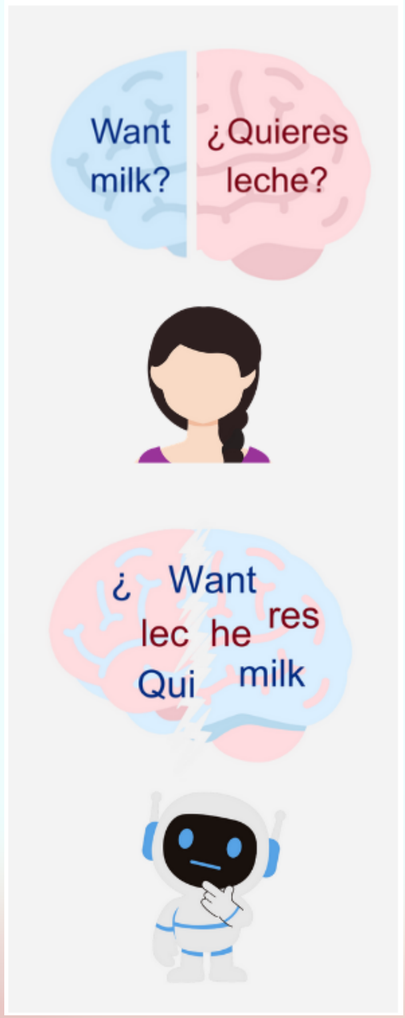




Linda Zeng Steven Y. Feng Michael C. Frank

Stanford University

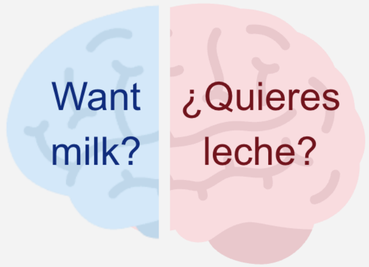
Research Questions



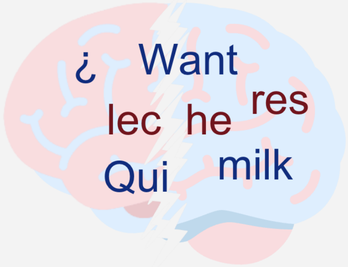
RQ1: Does bilingual training cause interference or confer advantages compared to monolingual training?

RQ2: How do bilingual exposure structures (speaker-split vs. mixed input; sentence- vs. word-level code-switching) influence performance?

RQ3: How do these effects vary with model size, total data scale, and language proportion?









ENGLISH TOPLINE — **EN** (100M) → Mom: Want milk? Let's go to the kitchen. Dad: Time to sleep. Good night!

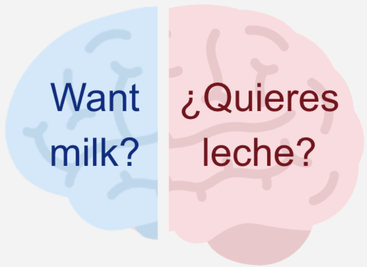
SPANISH TOPLINE — **SP** (100M) → Mom: ¿Quieres leche? Vamos a la cocina. Dad: Time to sleep. Good night!

ENGLISH BASELINE
- *(by-speaker)* **EN** (50M) → Mom: Want milk? Let's go to the kitchen.
- *(random)* **EN** (50M)

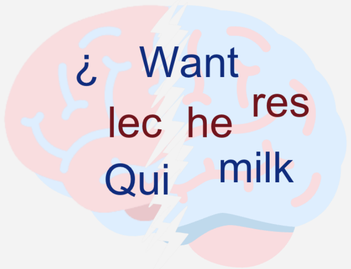
MULTILINGUAL
- *(by-speaker)* **EN** (50M) **SP** (50M) → Mom: Want milk? Let's go to the kitchen. Dad: Hora de dormir. ¡Buenas noches!
- *(random)* **EN** (50M) **SP** (50M)

CODE-SWITCHING
- *(sentence-level)* **EN-SP** (100M) → Mom: ¿Quieres leche? Let's go to the kitchen. Dad: Time to sleep. ¡Buenas noches!
- *(word-level)* **EN-SP** (100M) → Mom: Want leche? Vamos a la kitchen. Dad: Time to dormir. ¡Buenas noches!

(Feng et al.,2024)









RQ1: Does bilingual training cause interference or confer advantages compared to monolingual training?

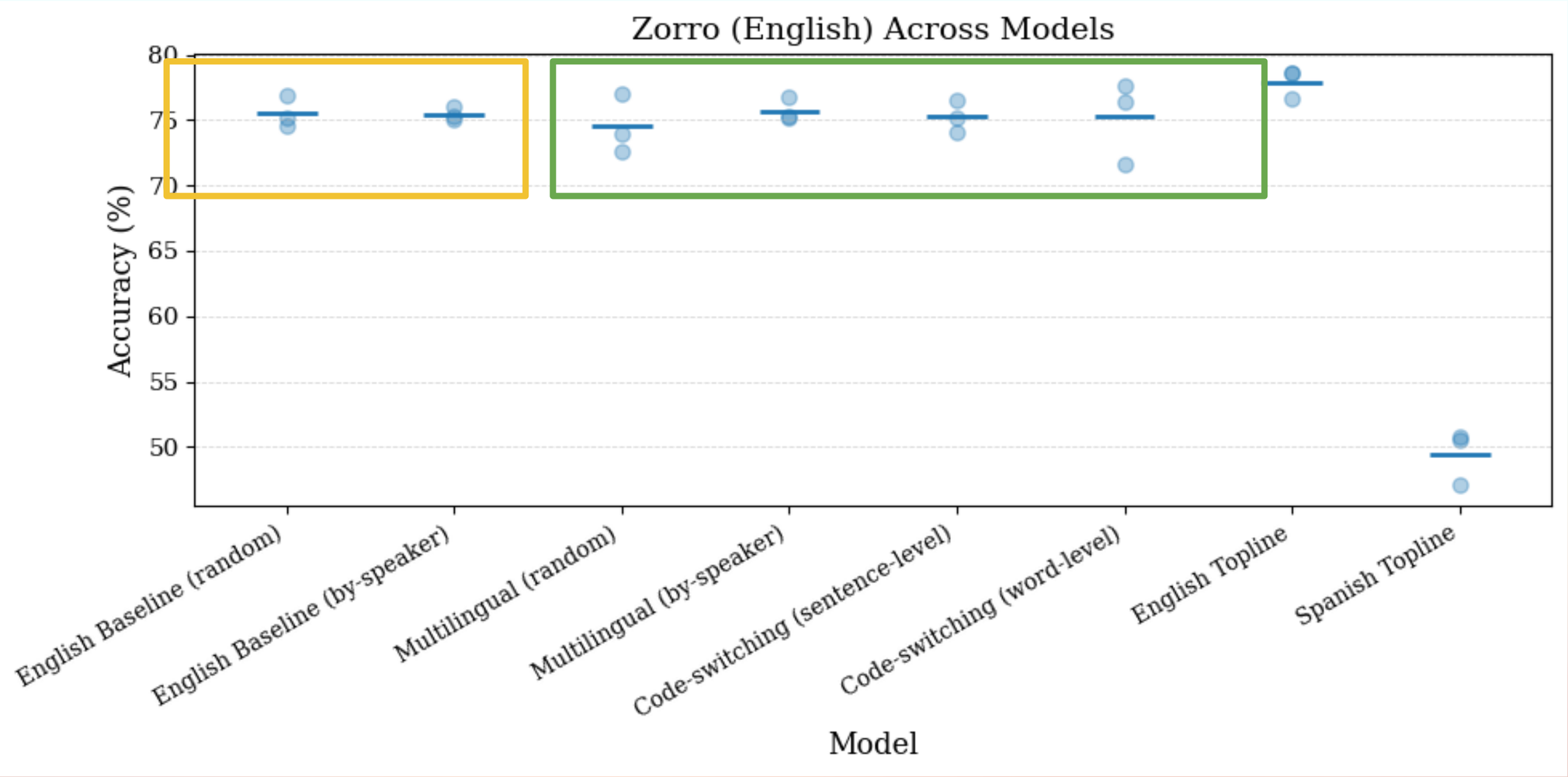
RQ2: How do bilingual exposure structures (speaker-split vs. mixed input; sentence- vs. word-level code-switching) influence performance?

RQ3: How do these effects vary with model size, total data scale, and language proportion?

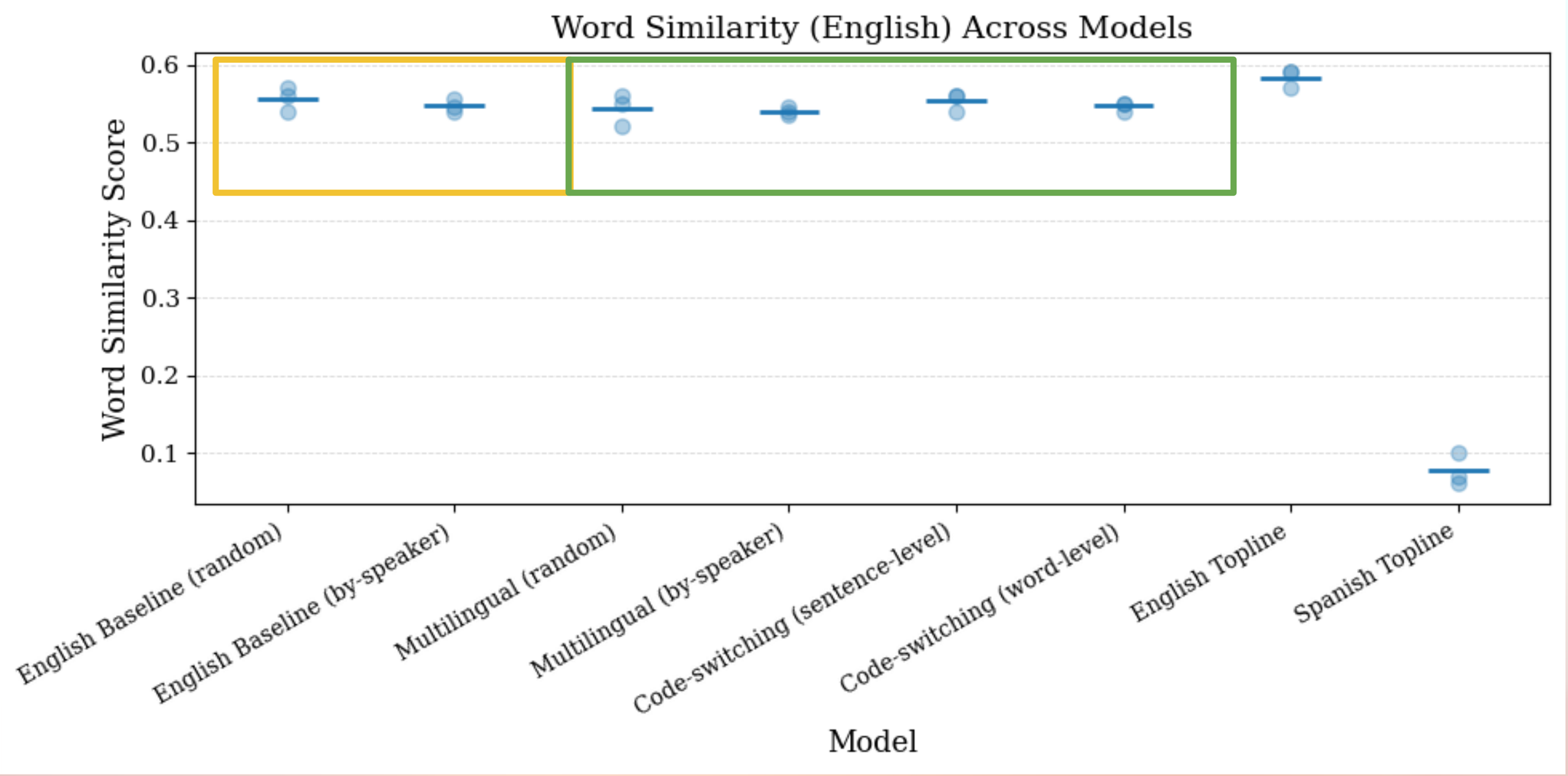
Low perplexity over English and Spanish text demonstrates that multilingual models accurately predict the next word in both languages.

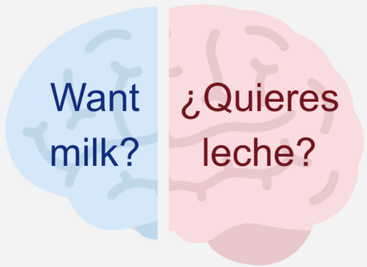
Training Condition	PPL (EN)	PPL (SP)
English Baseline (random)	2.69 ± 0.01	>100000
English Baseline (by-speaker)	4.63 ± 0.18	>100000
Multilingual (random)	2.61 ± 0.02	2.82 ± 0.02
Multilingual (by-speaker)	3.04 ± 0.01	3.37 ± 0.02
Code-switching (sentence-level)	5.26 ± 0.56	6.06 ± 0.61
Code-switching (word-level)	3.19 ± 0.05	3.74 ± 0.01
English Topline	2.43 ± 0.00	>100000
Spanish Topline	>1000	2.64 ± 0.01

Grammaticality performance is the same between the English baseline and all multilingual conditions.

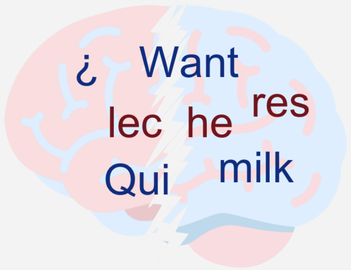


Word similarity performance is the same between the English baseline and all multilingual conditions.









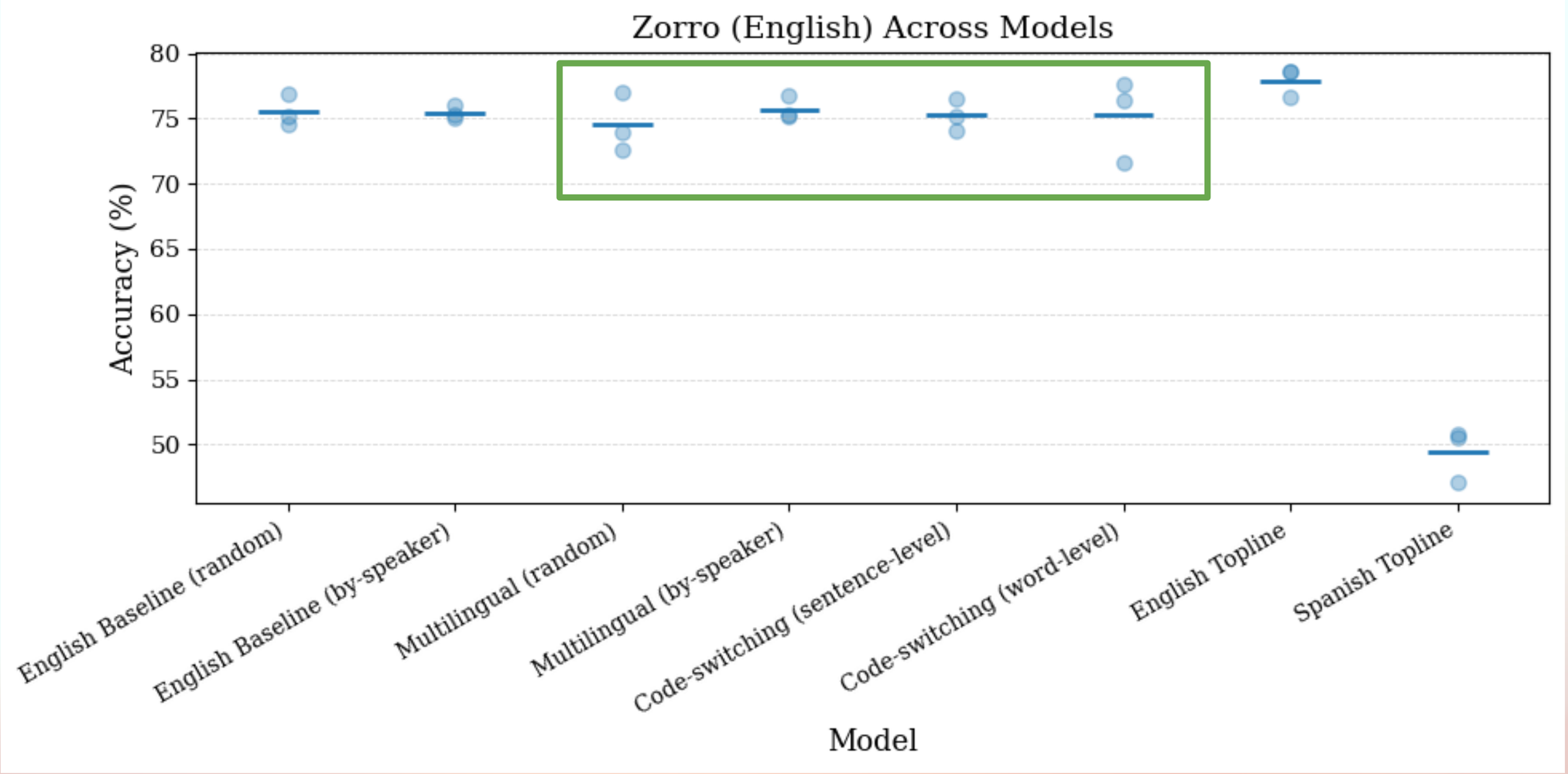


RQ1: Does bilingual training cause interference or confer advantages compared to monolingual training?

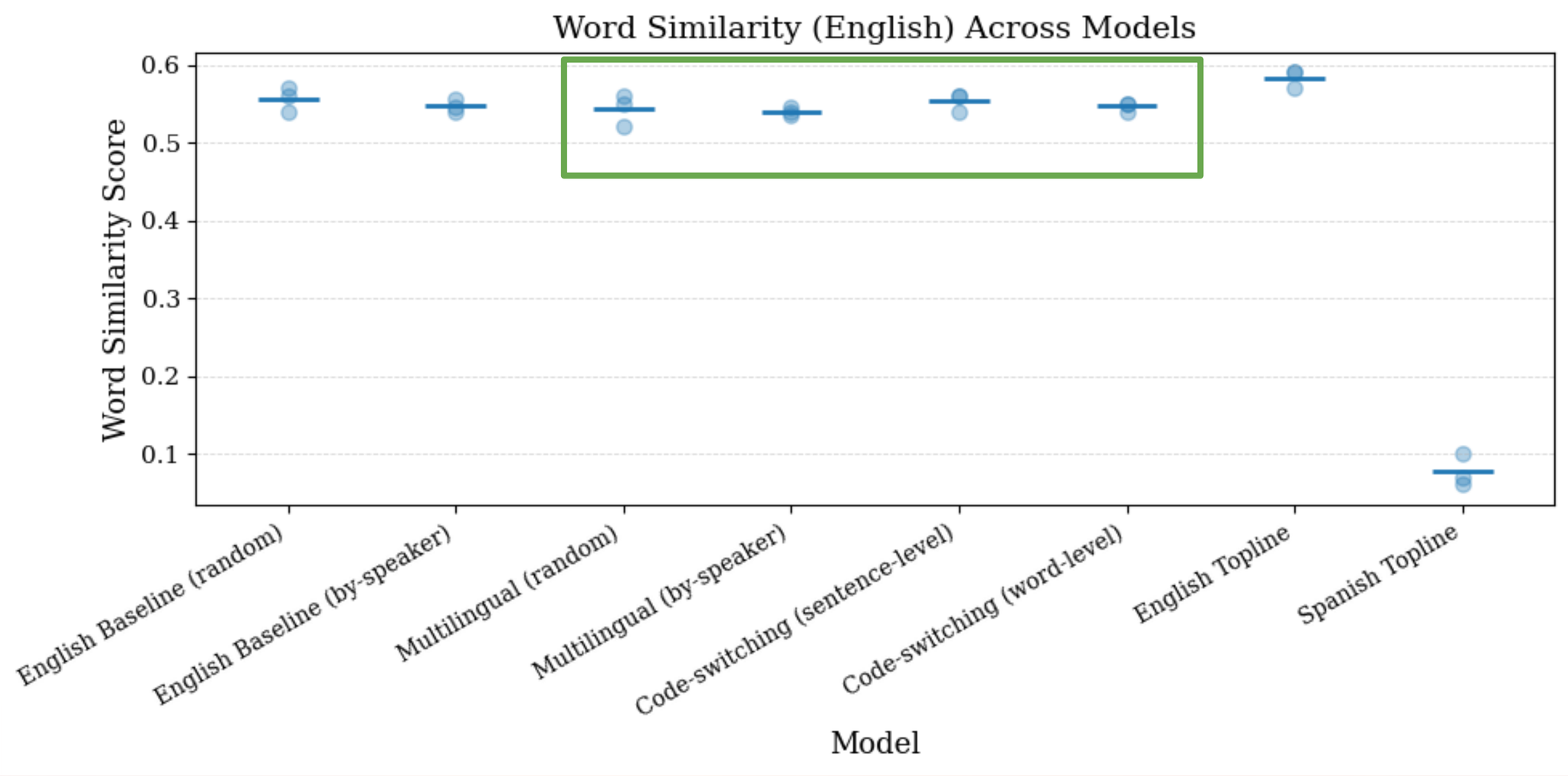
RQ2: How do bilingual exposure structures (speaker-split vs. mixed input; sentence- vs. word-level code-switching) influence performance?

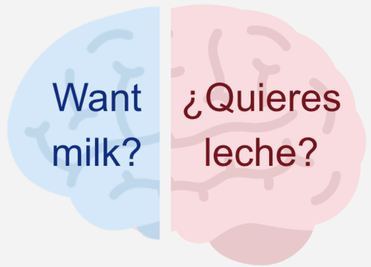
RQ3: How do these effects vary with model size, total data scale, and language proportion?

Grammaticality performance is the same between different multilingual conditions.

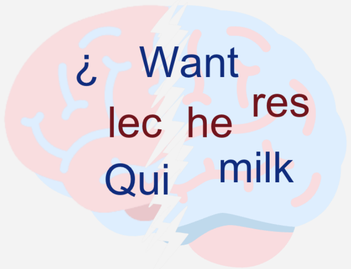


Word similarity performance is the same between different multilingual conditions.











RQ1: Does bilingual training cause interference or confer advantages compared to monolingual training?

RQ2: How do bilingual exposure structures (speaker-split vs. mixed input; sentence- vs. word-level code-switching) influence performance?

RQ3: How do these effects vary with model size, total data scale, and language proportion?

Data (not model size) affects performance.

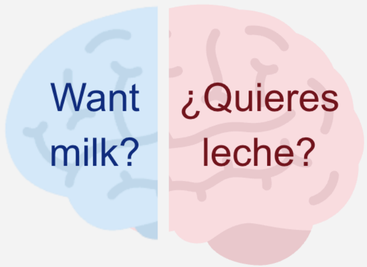
GPT2-mini
(⅓ the parameters)

Condition	WS (EN)	Zorro (%)
English Topline	0.58 ± 0.01	77.11 ± 0.61
English Baseline (by-speaker)	0.55 ± 0.01	75.73 ± 1.02
English Baseline (random)	0.55 ± 0.01	75.25 ± 1.11
Multilingual (by-speaker)	0.55 ± 0.01	76.09 ± 0.82
Multilingual (random)	0.55 ± 0.01	74.97 ± 2.03
Code-switching (sentence-level)	0.56 ± 0.00	75.16 ± 1.13
Code-switching (word-level)	0.52 ± 0.05	75.52 ± 1.01
Spanish Topline	0.08 ± 0.00	50.21 ± 1.24

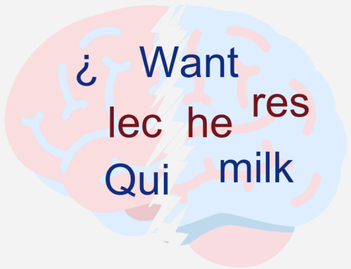
20M words
(⅕ the data scale)

Condition	WS (EN)	Zorro (%)
English Topline	0.47 ± 0.03	73.86 ± 1.92
English Baseline (by-speaker)	0.35 ± 0.01	69.65 ± 0.57
English Baseline (random)	0.33 ± 0.02	72.09 ± 0.75
Multilingual (by-speaker)	0.36 ± 0.01	70.42 ± 0.92
Multilingual (random)	0.33 ± 0.02	70.82 ± 1.06
Code-switching (sentence-level)	0.38 ± 0.02	71.65 ± 4.58
Code-switching (word-level)	0.30 ± 0.02	68.86 ± 1.86
Spanish Topline	0.06 ± 0.01	47.57 ± 2.12

iversity



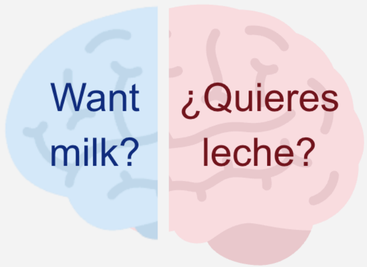




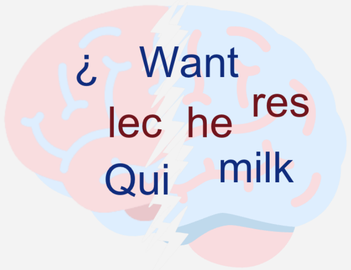


RQ1: Does bilingual training cause interference or confer advantages compared to monolingual training?

- No L1 interference: Adding Spanish does not harm English across perplexity, syntax, or semantics.
- Successful bilingual learning: Models acquire L2 and maintain strong cross-lingual performance.



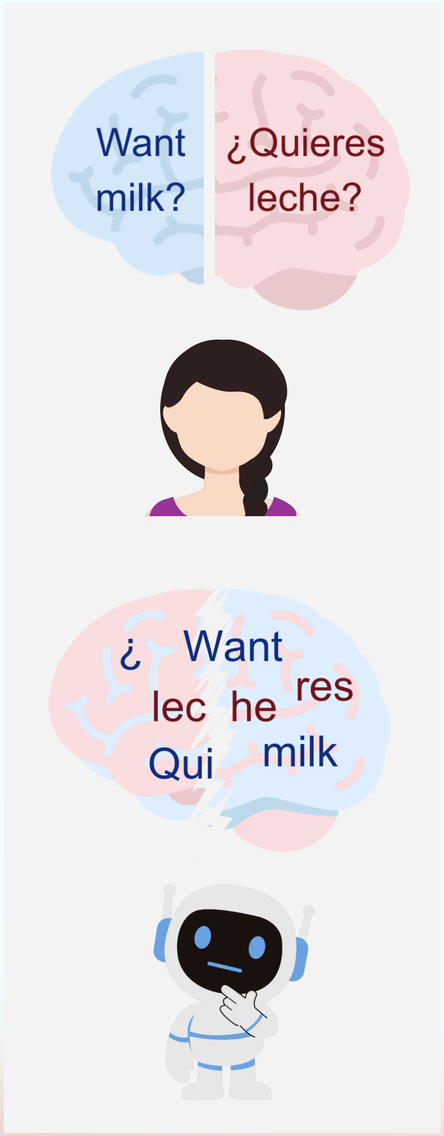






RQ2: How do bilingual exposure structures (speaker-split vs. mixed input; sentence- vs. word-level code-switching) influence performance?

- Exposure structure is surprisingly irrelevant: Speaker splits and even word-level code-switching do not degrade L1.



RQ3: How do these effects vary with model size, total data scale, and language proportion?

- Data scale matters more than model size

Takeaways

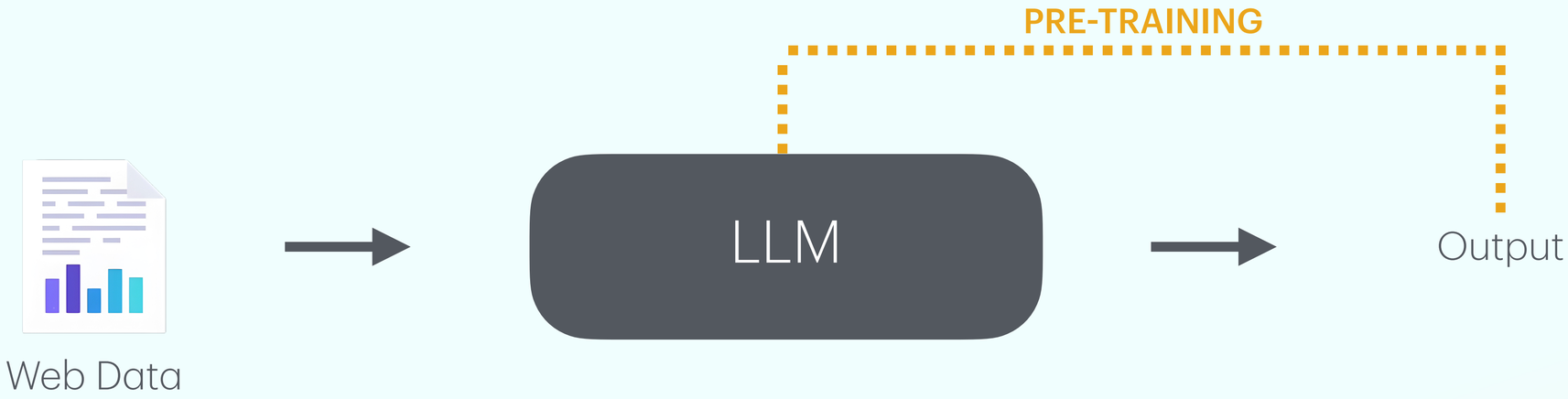
- No bilingual confusion: Models retain monolingual L1 performance while successfully acquiring L2.
- Speaker splits and word-level code-switching have minimal impact.
- Data drives outcomes: Performance scales with matching-language exposure (not model size).



github.com/styfeng/bilingual-babyLM

Retrieval Augmented Gen.

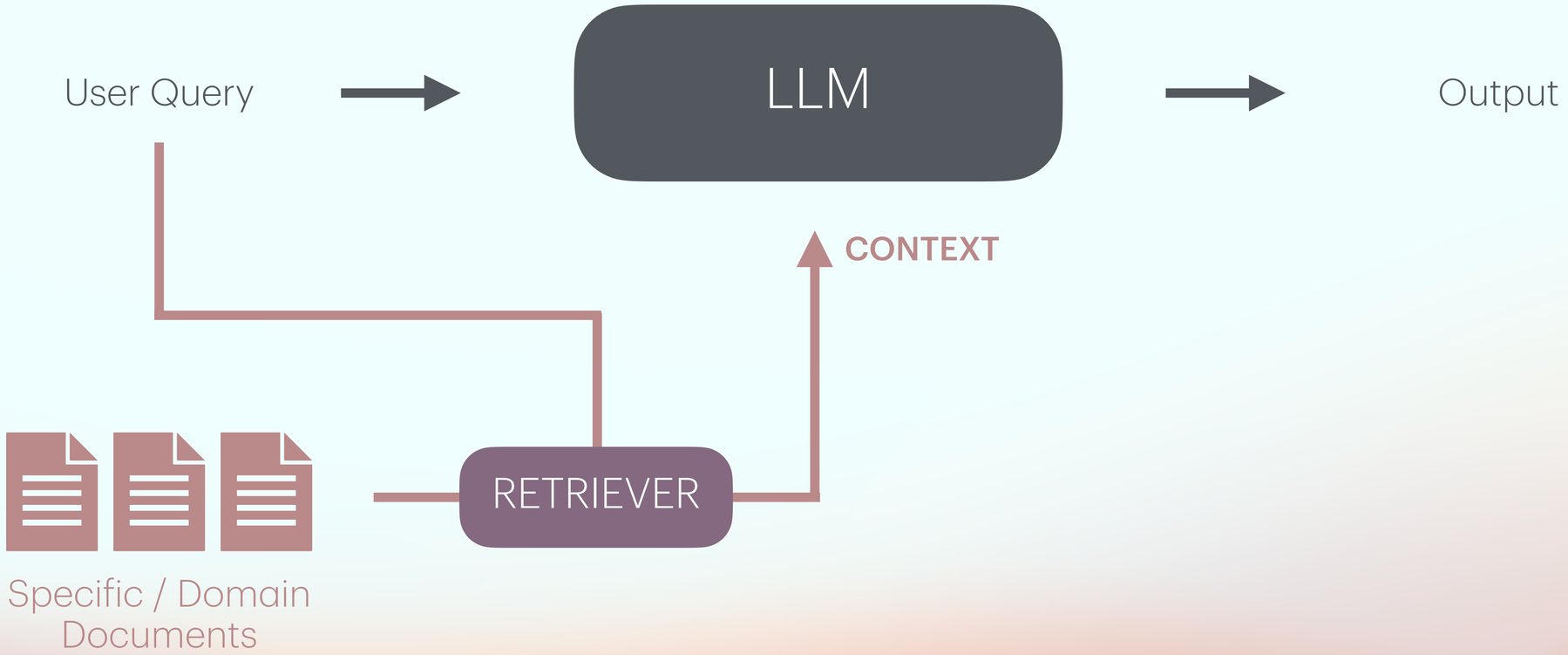
Basic Idea



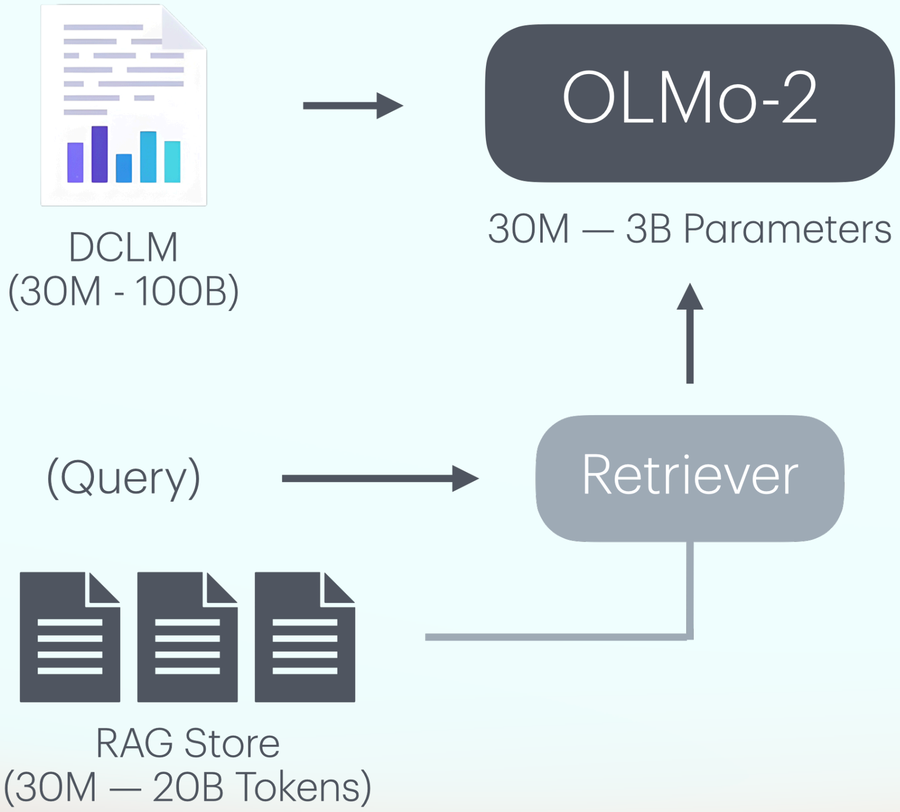
Basic Idea



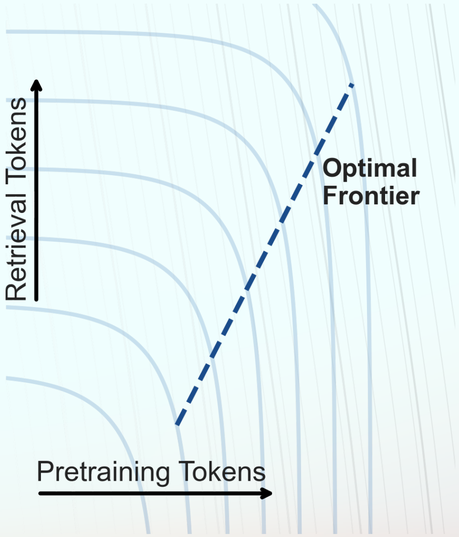
Basic Idea

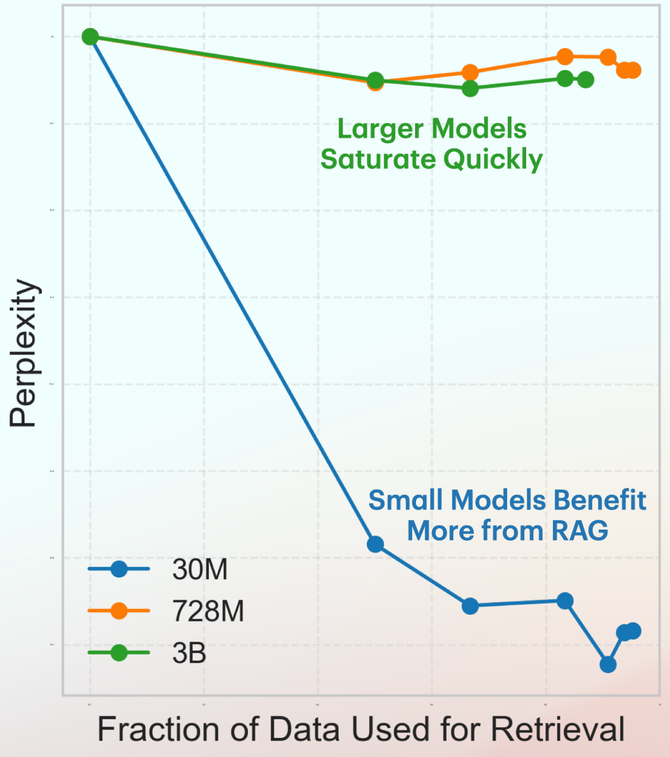

User Query
LLM
Output
CONTEXT
RETRIEVER
Specific / Domain Documents

Studying RAG Scaling

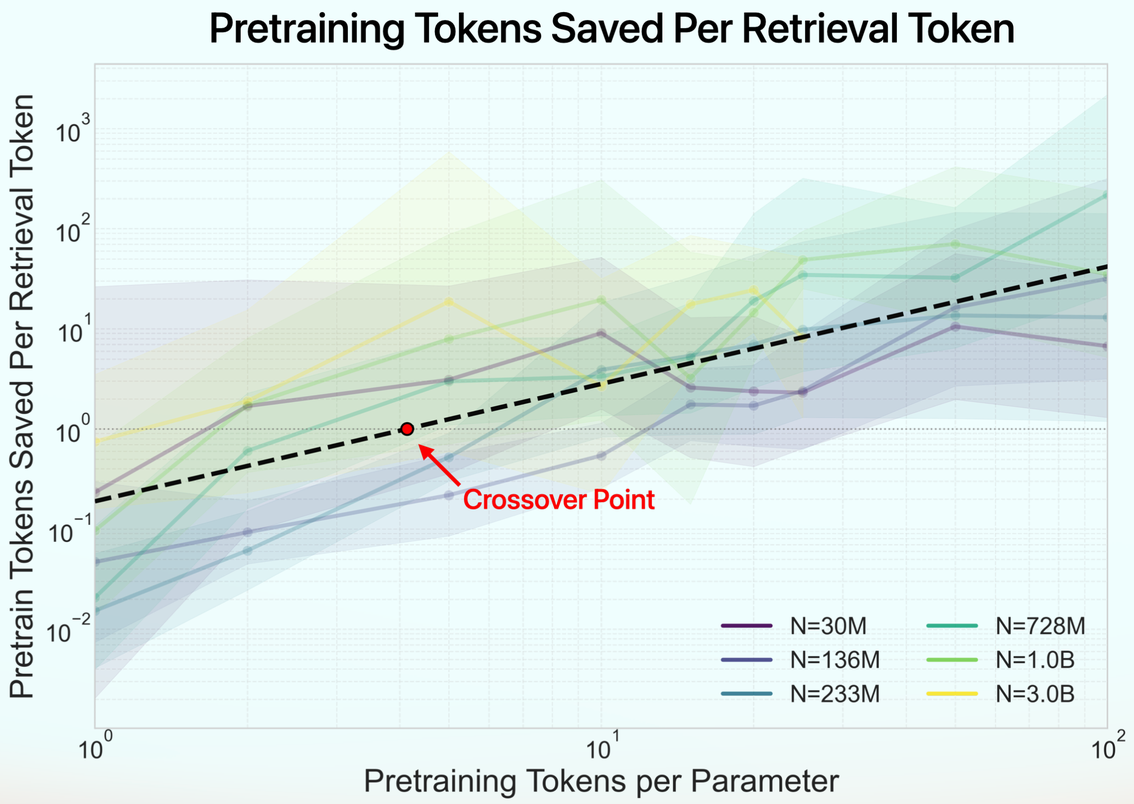


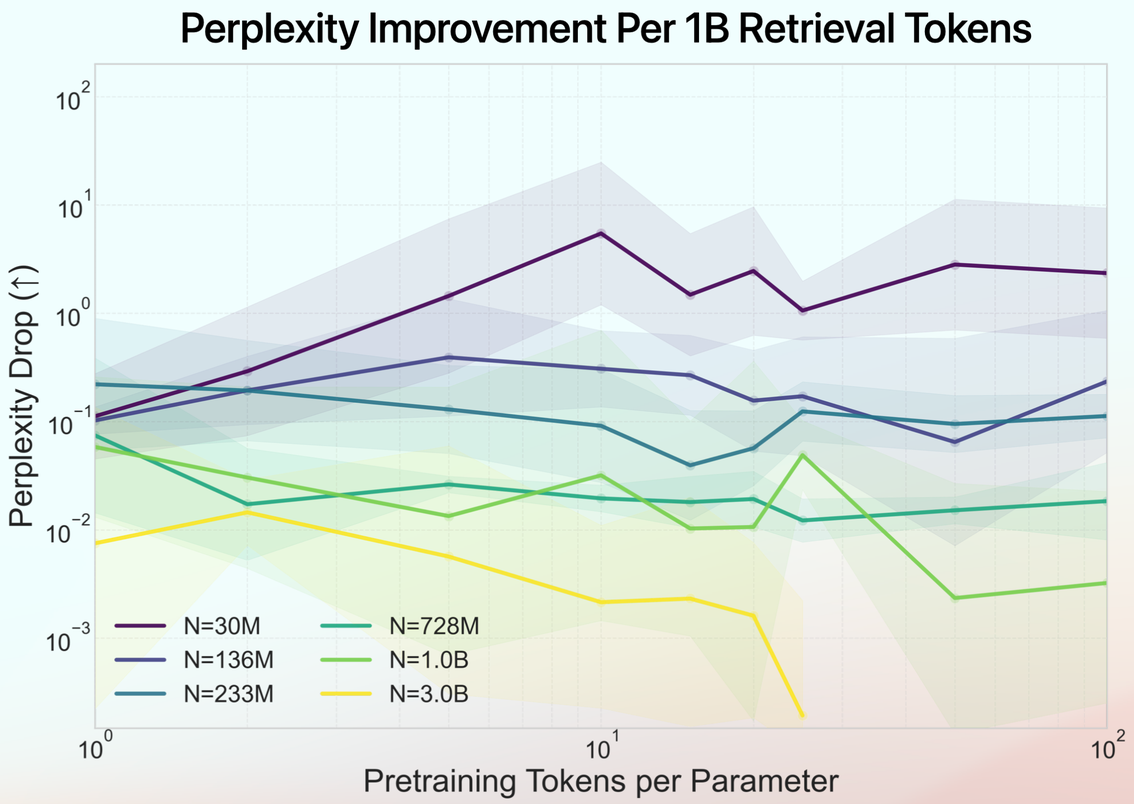
How much RAG should you use?





How Much RAG is Optimal?





Want to Learn More?

- Contact: karanps@stanford.edu & syfeng@stanford.edu
- Paper: https://arxiv.org/abs/2604.00715



Curriculum-Guided Scaling

Motivation

- As the cost of LLM pretraining grows, there is continued interest in strategies to improve learning efficiency
- Typical pretraining differs significantly from how humans learn, and results in models that can struggle at reasoning
- Inspired by human development, can we gradually scale models in tandem with increasingly complex data?



Motivation

- Synthetic datasets: TinyStories

Ex: Once upon a time, in a small town, there was a little boy named Tim. Tim loved ice-cream so much. One day, he saw an ancient ice-cream truck. The truck was very old and slow, but it had a big sign that said “Ice-Cream”. Tim went to the truck and said, "Hi, can I have ice-cream, please?" The ice-cream man looked at him and said, "Sure, little boy. Here you go." Tim took the ice-cream and started to eat it. But it did not taste good at all. It tasted very bad…

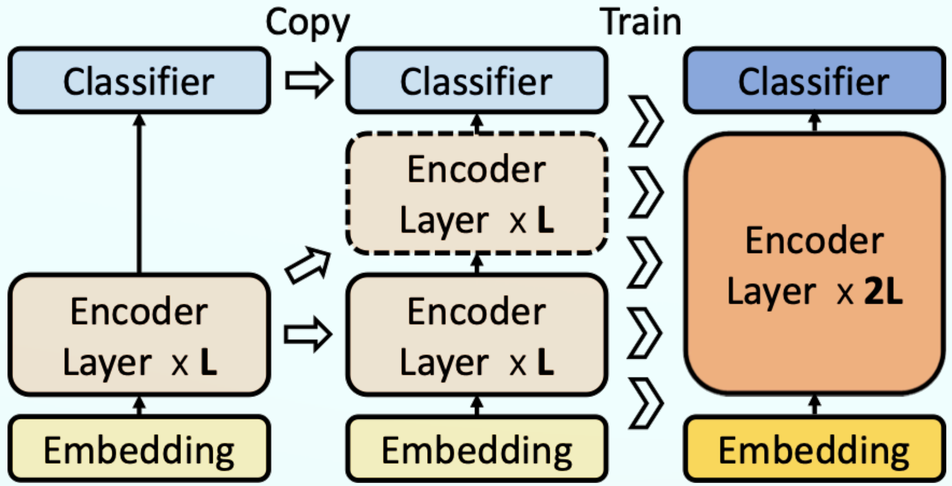
- Data filtering: DataComp-LM



- Image data filtering: MetaCLIP

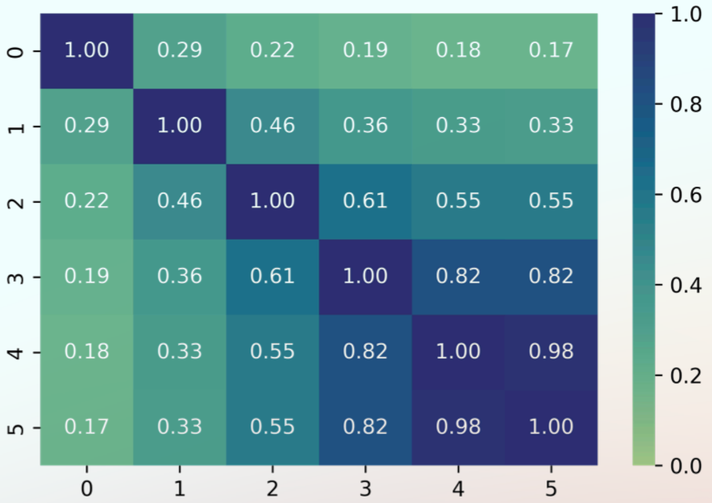
Prior Work - Stacking

BERT Stacking (ICML 2019)



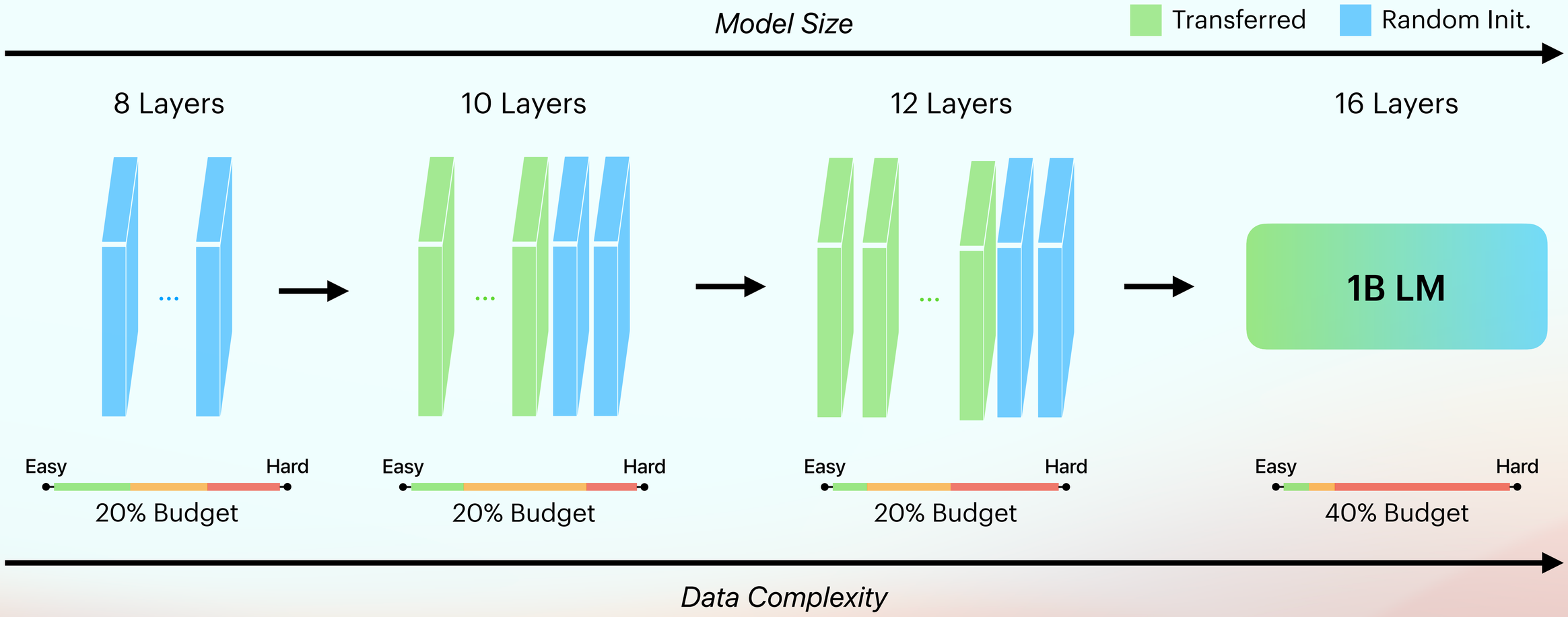
Stacking Algorithm

MIDAS (NeurIPS 2024)



GradStack Layer Similarity

Method



$$\Theta^{(i+1)} := \{\Theta^{(i)}_{\text{embed}} \circ \Theta^{(i)}_{1:k} \circ \Theta^{\text{random}}_{k+1:n} \circ \Theta^{(i)}_{\text{norm}} \circ \Theta^{(i)}_{\text{lm}}\}$$

Results

Model	The Pile↓	PIQA↑	ARC-E↑	ARC-C↑	HellaSwag↑	Lambada↑	Average↑
GPT-2-Small							
Pretrained GPT-2-Small*	23.97	62.51%	39.48%	22.70%	31.14%	—	38.96%
Baseline (Randomized)	37.51	58.00%	36.11%	21.08%	**26.60%**	—	35.45%
Baseline (Curricularized)	36.09	59.09%	36.78%	22.18%	26.43%	—	36.12%
Layer Scaling Only	39.67	58.76%	35.82%	21.42%	26.35%	—	35.59%
MIDAS (Saunshi et al., 2024)	49.90	56.91%	34.81%	21.16%	25.66%	—	34.64%
CGLS (Simple)	35.93	59.03%	**37.67%**	21.76%	19.50%	—	34.49%
CGLS	**34.33**	**60.12%**	36.62%	**22.53%**	26.55%	—	**36.46%**
Llama-3.2-1B							
Pretrained Llama-3.2-1B*	9.25	74.32%	60.56%	36.52%	63.67%	62.00%	59.41%
Baseline (Randomized)	**20.83**	59.09%	36.36%	24.24%	34.20%	32.99%	37.38%
Baseline (Curricularized)	21.93	58.27%	35.19%	**25.43%**	**34.65%**	**33.09%**	37.33%
Layer Scaling Only	29.53	57.45%	32.58%	23.46%	30.00%	27.73%	34.24%
MIDAS (Saunshi et al., 2024)	27.64	55.44%	32.58%	24.23%	29.62%	25.87%	33.55%
CGLS	25.34	**61.21%**	**41.37%**	**25.43%**	**34.65%**	32.74%	**39.08%**

Table 1. Evaluation results comparing CGLS and baseline approaches across average perplexity on The Pile dataset and core reasoning- and knowledge-intensive downstream tasks.

Results

Model	Pile↓	PIQA↑	ARC-E↑	ARC-C↑	HS↑	LBD↑	Avg↑	GPQA↑	T9+↑	ORE↑	IB↑
Llama-3.2-1B (20B Tokens)											
BL-R	26.33	63.00%	41.79%	23.38%	33.52%	30.37%	38.41%	25.22%	2.787	17.30%	0.235%
BL-C	24.58	63.55%	42.97%	24.83%	34.61%	32.66%	39.72%	22.32%	2.947	**20.61%**	0.401%
CGLS	**22.99**	**67.36%**	**46.97%**	**26.28%**	**36.80%**	**34.08%**	**42.30%**	**29.24%**	**3.568**	17.03%	**0.539%**

Table 3. Evaluation results comparing CGLS and baseline approaches (BL-R refers to the compute-matched baseline trained on randomized data, and BL-C refers to the compute-matched baseline trained with a curriculum) at the 20B-token scale, aligned with the 20× data/parameter ratio in (Hoffmann et al., 2022). All models are compute-matched.

Discussion

- TLDR: Testing at GPT-2-Small (124M parameter) and Llama-3.2-1B scales, combining progressive model growing with a structured curriculum “unlocks” the potential of layer stacking techniques.
- Not all tasks benefit equally from the progressive curriculum, and at LLaMA-3.2-1B scale, the randomized baseline achieved slightly better perplexity despite underperforming on reasoning tasks.
- Future potential: finer tuning of hyperparameters, alternative notions of curriculum, applying this idea to other modalities like the image domain or medical imaging / fMRI.

Want to Learn More?

- Contact email: karanps@stanford.edu
- Paper: https://arxiv.org/abs/2506.11389



Overall Takeaways

Unified Insights on Data Strategies for LMs

- Data effectiveness is not just volume but quality, structure, and how it's used
- Data selection and strategies are important considerations
  - Project 1: **Baby-Scale Language Learning from Individual Environments** — Individual environments vary; differences are driven more by data composition & quality than quantity.
  - Project 2: **Bilingual Language Modeling under Controlled Exposure** — Models can learn two languages effectively without confusion, regardless of exposure structure.

Unified Insights on Data Strategies for LMs

- Project 3: **Curriculum-Guided Layer Scaling for Language Model Pretraining** — scaling your model alongside a data curriculum improves downstream performance up to 1B scale.
- Project 4: **Scaling Laws for RAG-Considerate Pretraining** — Optimal performance comes from balancing parametric learning (pretraining) and external memory (retrieval).

Our research underscores that effective language modeling isn't just about amassing data but about smarter data utilization strategies that harness its structure, quality, and characteristics. By continuing to refine data-centric approaches, the future of LLM training promises smarter, more efficient, and highly adaptable models.

Post Training

Post-Training and Inference-Time Compute

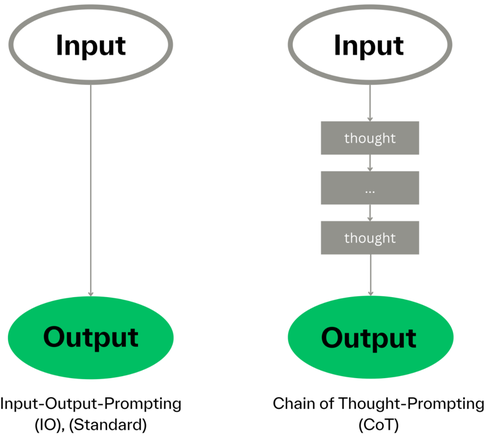
- After we have a pre-trained a generally capable model, what now?
  - How do we adapt it to specific tasks, scenarios, users, domains, etc?
- Major strategies:
  - Fine-tuning / alignment
  - Prompt-based methods
  - RAG / retrieval-based methods
  - Agents / tool use

Improving Reasoning With Chain-of-Thought (CoT)

- **Prompting** the model to "**think step by step**" before answering
- Intermediate steps provide guidance through reasoning problem
- Similar to human thought process: decompose multi-step problems
- Provides an interpretable window into behavior of the model

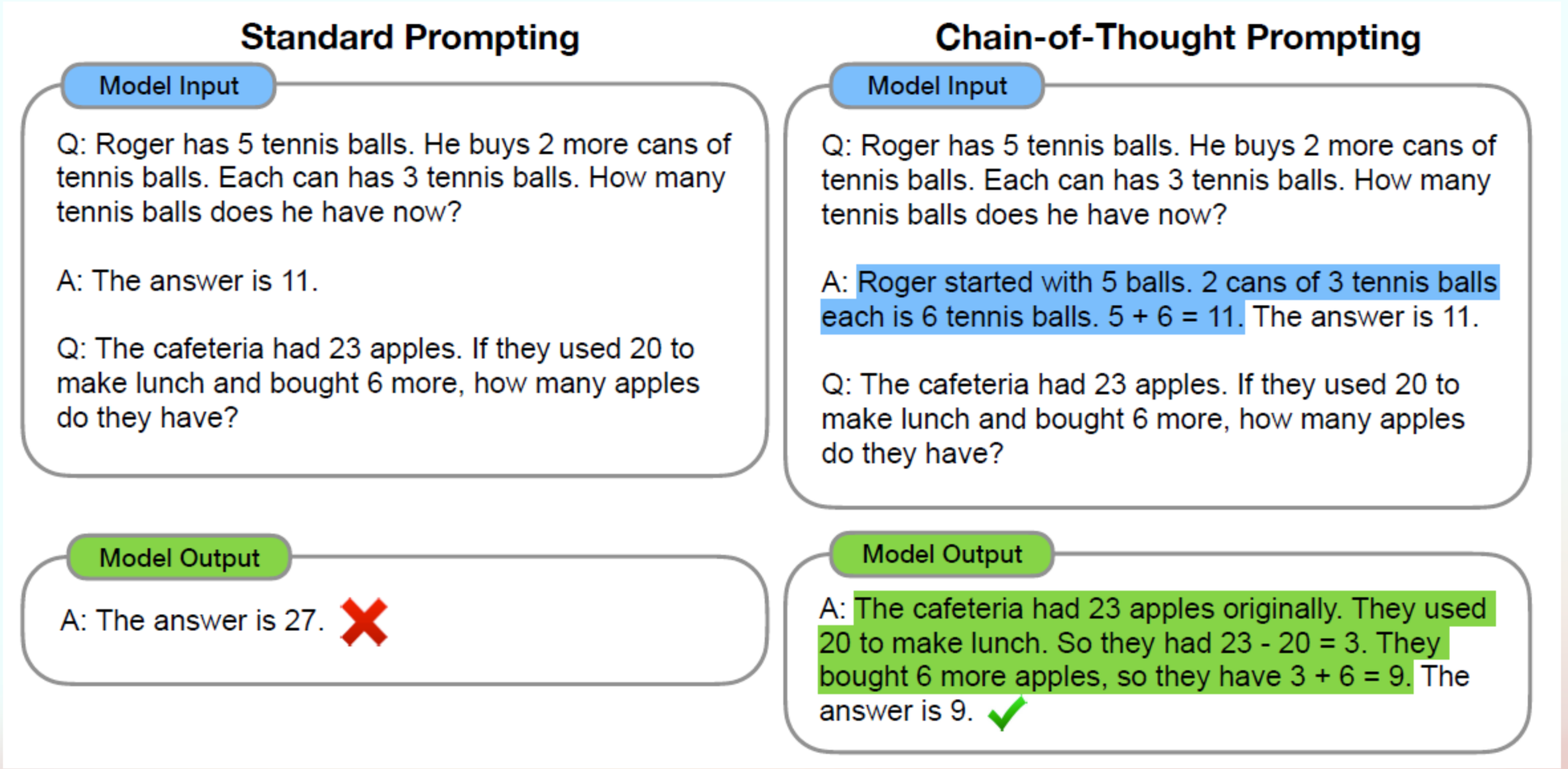
  (how it arrived at an answer, where it goes wrong in its reasoning path)
- Suggest that deep in the model's weights, it knows more

  about the problem than just prompting a response



Ref: https://arxiv.org/abs/2201.11903

Example: Chain-of-Thought Reasoning



Ref: https://arxiv.org/abs/2201.11903

Extending CoT → Tree-of-Thoughts

- Instead of producing a singular reasoning path, model considers multiple reasoning paths
- Use self-evaluation to decide on outputs (i.e., majority vote, debate, etc.)

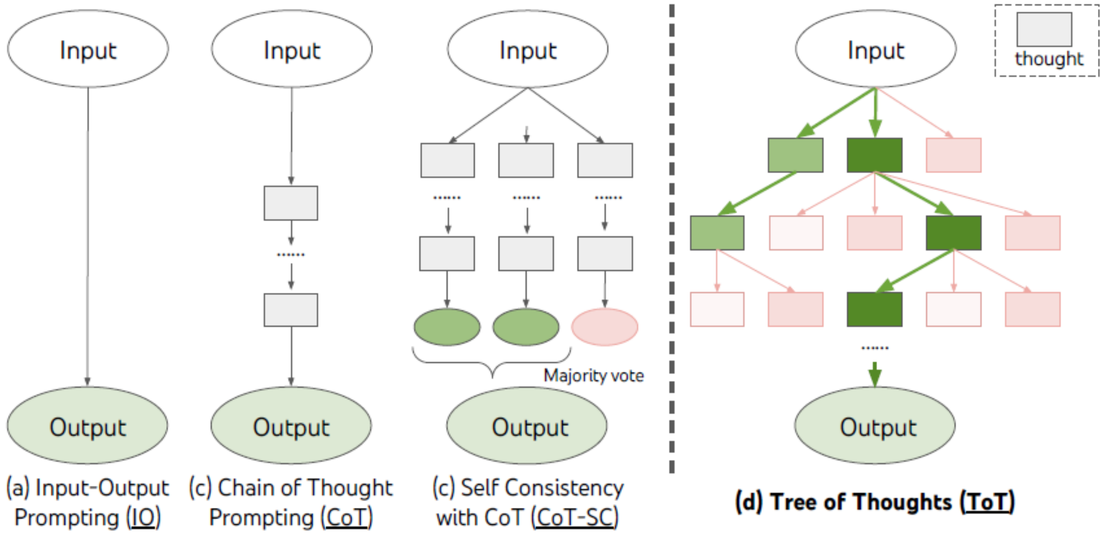


Figure 1: Schematic illustrating various approaches to problem solving with LLMs. Each rectangle box represents a *thought*, which is a coherent language sequence that serves as an intermediate step toward problem solving. See concrete examples of how thoughts are generated, evaluated, and searched in Figures 2,4,6.

Ref: https://arxiv.org/abs/2305.10601

Extending CoT → Program-of-Thought

- Generates code as the intermediate reasoning steps
- Offloads final problem solving to some code interpreter
- Formalizes language into programs to arrive at better, more precise answers
- Related work: ***Program of Thoughts Prompting*** [https://arxiv.org/abs/2211.12588]

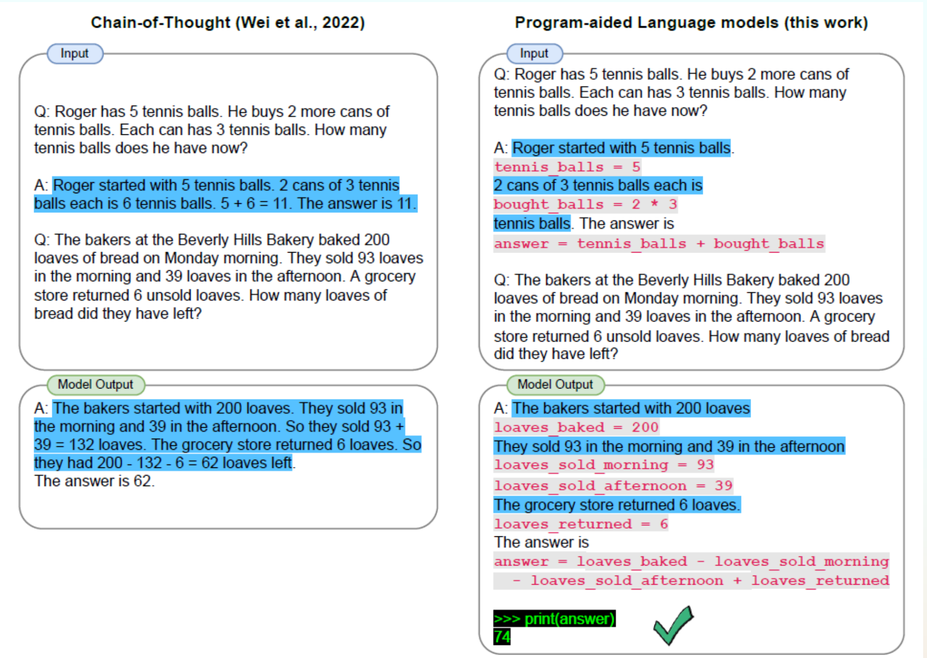


Figure 1: A diagram illustrating PAL: Given a mathematical reasoning question, Chain-of-thought (left) generates intermediate reasoning steps of free-form text. In contrast, Program-aided Language models (PAL, right) generate intermediate steps *and* Python code. This shifts the role of *running* the reasoning steps from the language model to the Python interpreter. The final answer is obtained by running the generated reasoning chain. Chain-of-thought reasoning is highlighted in blue; PAL steps are highlighted in gray and pink; the Python interpreter run is highlighted in black and green.

Ref: https://arxiv.org/abs/2211.10435

Problem Decomposition: Socratic Questioning

- Self-questioning module to use LM to propose "subproblems" related to the original problem
- "Divide-and-conquer" algorithm that simulates self-questioning and iterative thinking
- Recursively answer sub-problems to solve the original problem

- Related work: *Least-to-Most Prompting Enables Complex Reasoning in Large Language Models* [https://arxiv.org/abs/2205.10625]

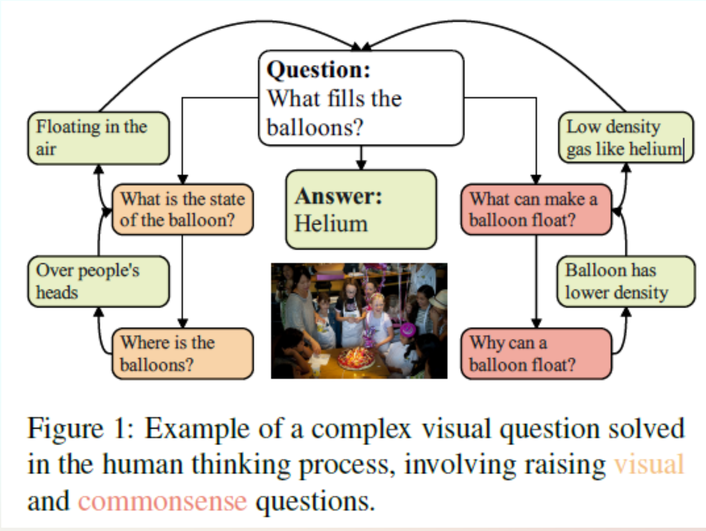
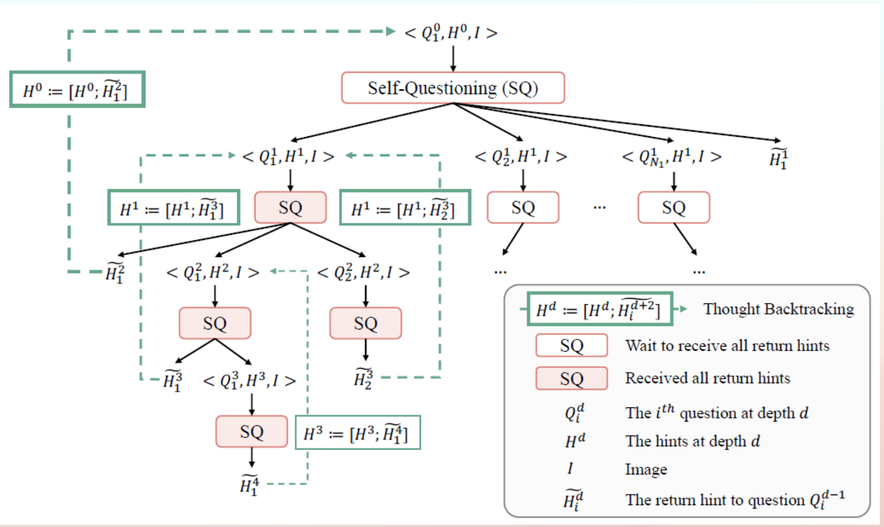


Figure 1: Example of a complex visual question solved in the human thinking process, involving raising visual and commonsense questions.



Ref: https://arxiv.org/abs/2305.14999

Problem Decomposition: Computation Graphs

- Formulate compositional tasks as computation graphs, breaking down reasoning into sub-procedures
- **Takeaway: Transformers can solve compositional tasks by reducing reasoning into subgraph matching, without developing systematic problem solving skills**
- Related work: *Graph of Thoughts: Solving Elaborate Problems with LLMs* [arXiv: 2308.09687]

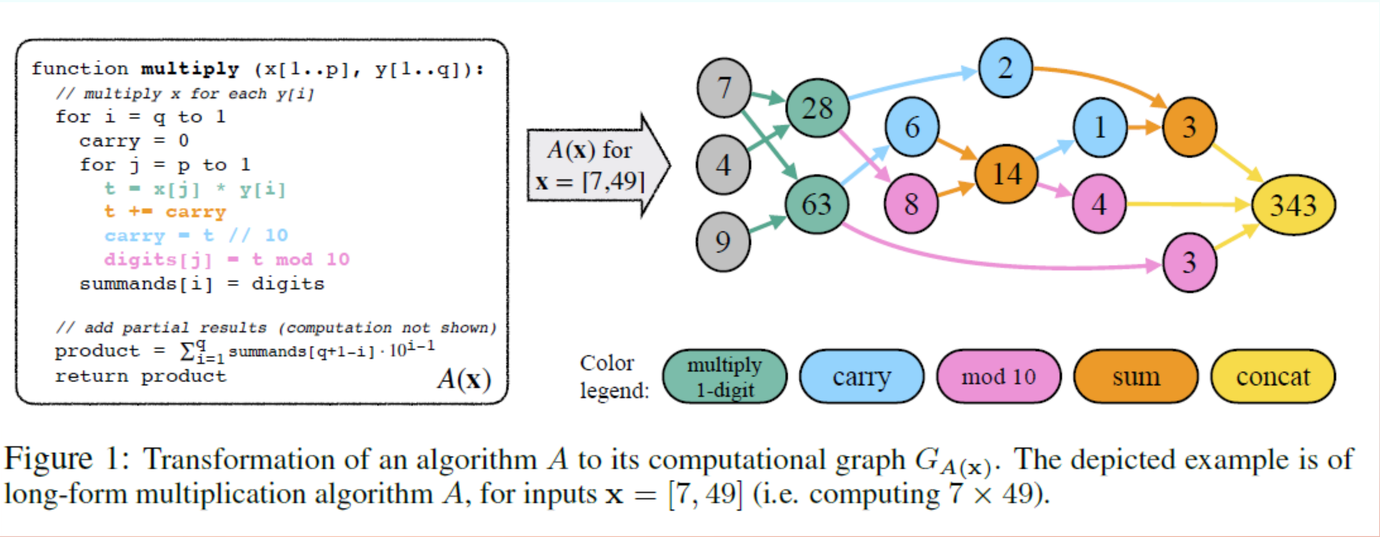


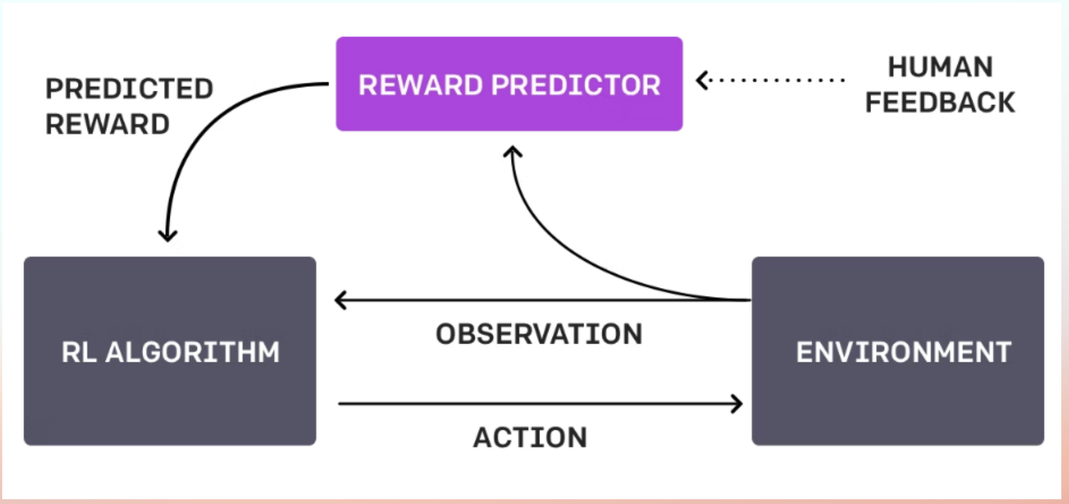
Figure 1: Transformation of an algorithm A to its computational graph $G_{A(\mathbf{x})}$. The depicted example is of long-form multiplication algorithm A, for inputs $\mathbf{x} = [7, 49]$ (i.e. computing 7×49).

Ref: https://arxiv.org/abs/2305.18654

Reinforcement Learning

Reinforcement Learning with Human Feedback (RLHF)

- RLHF: technique that trains a "reward model" directly from human feedback
- Uses the model as a reward function to optimize an agent's policy using reinforcement learning (RL) through an optimization algorithm such as PPO
- Ask humans to rank instances of the agent's behavior: "which produced response is better?"



RL-Free Alignment: Direct Preference Optimization (DPO)

- Directly trains model to prefer outputs that humans rank higher (no separate reward model)
- Finetune the LLM is to maximize the likelihood of generating preferred completions and minimize the likelihood of dispreferred (over several human preference pairs)

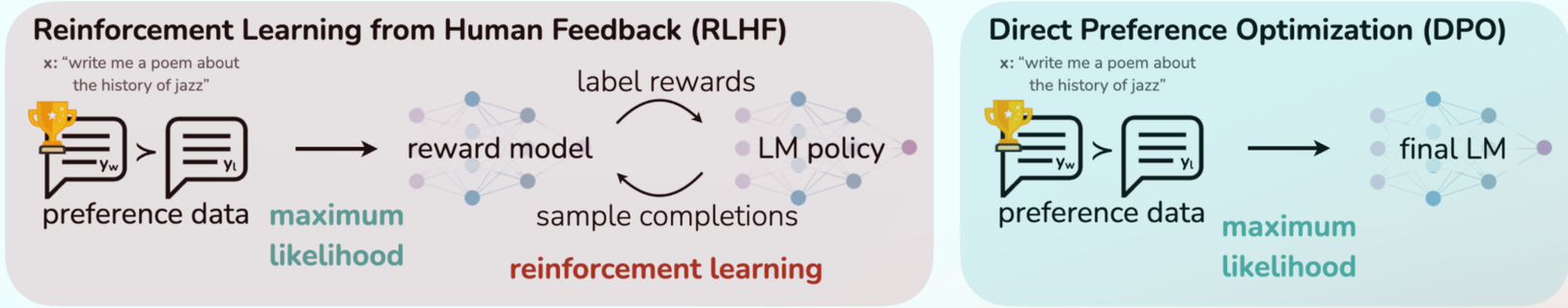


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

https://arxiv.org/pdf/2305.18290.pdf

Reinforcement Learning with AI Feedback (RLAIF)

- Replacing costly human preference labels with an off-the-shelf LLM's preference judgments
- Reward model trained on these AI-generated preference labels, and the policy is fine-tuned with RL--> performance matching RLHF
- Human evaluators equally preferred RLAIF-tuned outputs to RLHF outputs
- Suggests that this is a viable, scalable alternative to human feedback

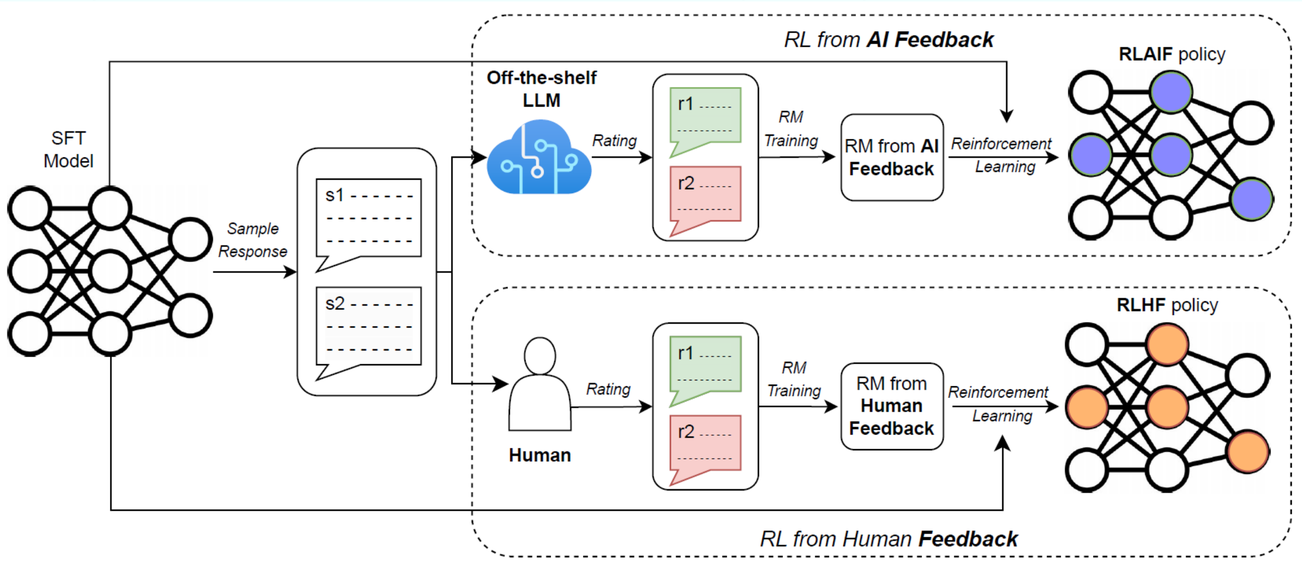


Figure 2: A diagram depicting RLAIF (top) vs. RLHF (bottom)

Ref: https://arxiv.org/abs/2309.00267

Group Relative Policy Optimization (DeepSeek)

- Policy: ranking responses within a group (self-ranking) and using relative quality as feedback
- Richer feedback from more fine-grained human preferences and helps stabilize training
- Improves LLM reasoning (math) while maintaining training stability
- Obtain complex skills like self-checking and CoT more easily
- Makes RLHF more stable and cost-efficient

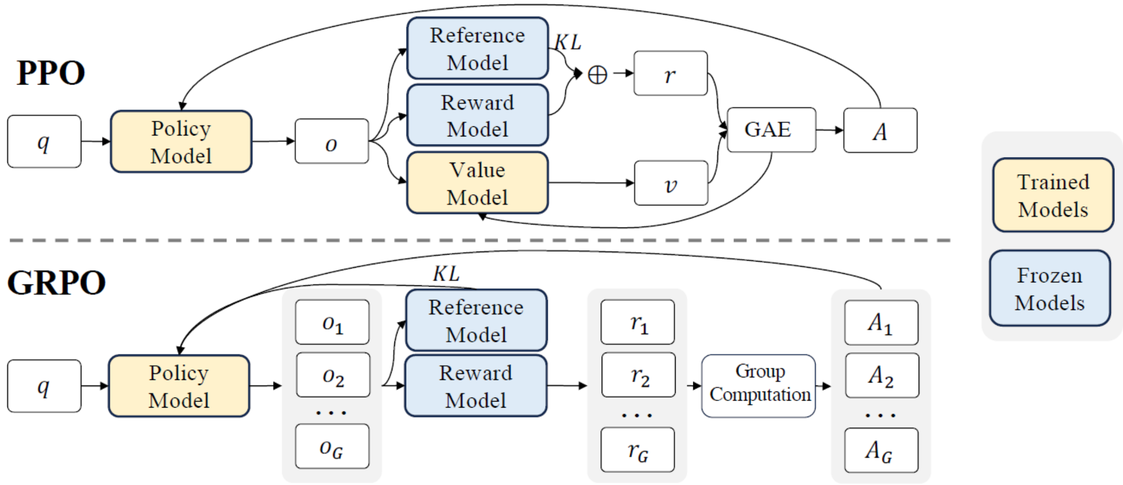


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

https://arxiv.org/abs/2402.03300
https://arxiv.org/abs/2501.12948

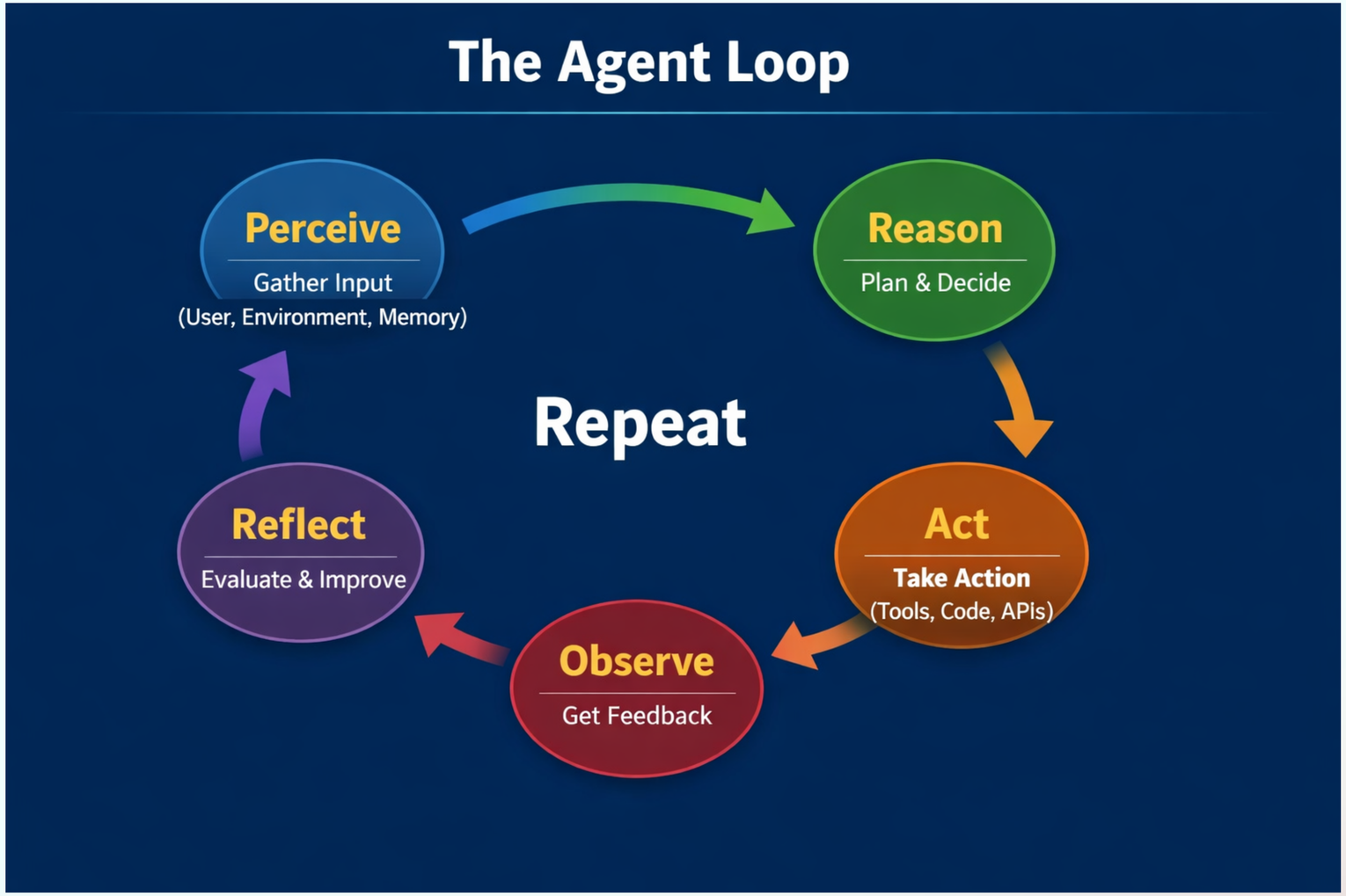
Process Supervision (Reasoning-Level Alignment)

- Reward reasoning process itself (steps of CoT) instead of only final answers
- Label or evaluate reasoning steps (human or AI feedback)
- Train models to produce good reasoning traces
- Reduces reward hacking (correct answer, bad reasoning)
- Improves reliability on multi-step tasks
- Encourages interpretable reasoning
- **Takeaway: Better reasoning comes from supervising how the model thinks, not just what it outputs**

Self-Improving AI Agents

What is an AI agent?

- System that perceives its environment, makes decisions, and takes actions toward achieving a goal (usually given by the human) - e.g. game-playing, task-solving, or research assistant
  - **Goal-directed:** Operates with a specific objective or reward function
  - **Autonomous decision-making:** Chooses actions based on observations or internal state
  - **Can act iteratively:** Many agents act over multiple steps
  - **Memory & state tracking:** Maintains context, plans, or history across interactions
  - **Tool use:** May interact with external tools/APIs (e.g. search, calculator)
  - **Learning & adaptation:** Can improve performance through experience or feedback
- Agents that can reflect and improve upon themselves is a major advantage



Bottom line: Agents are not one-shot models → they operate in loops

Self-Improvement

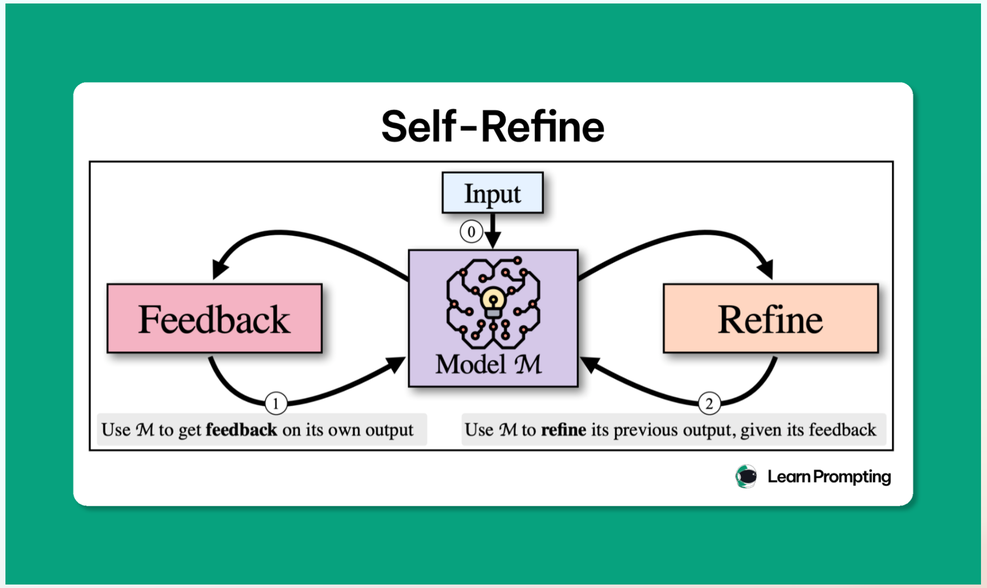
- Models can reflect on their own output to iteratively improve themselves
  - Reflect on outputs and reasoning
  - Evaluate correctness and quality
  - Improve via iteration
  - Simulate multi-step reasoning chains



Self-Improvement - Refinement

- Iterative prompting: LLM critiques and improves its own outputs.
- Generates an initial response, then refines it.
- Uses feedback loops to enhance performance.

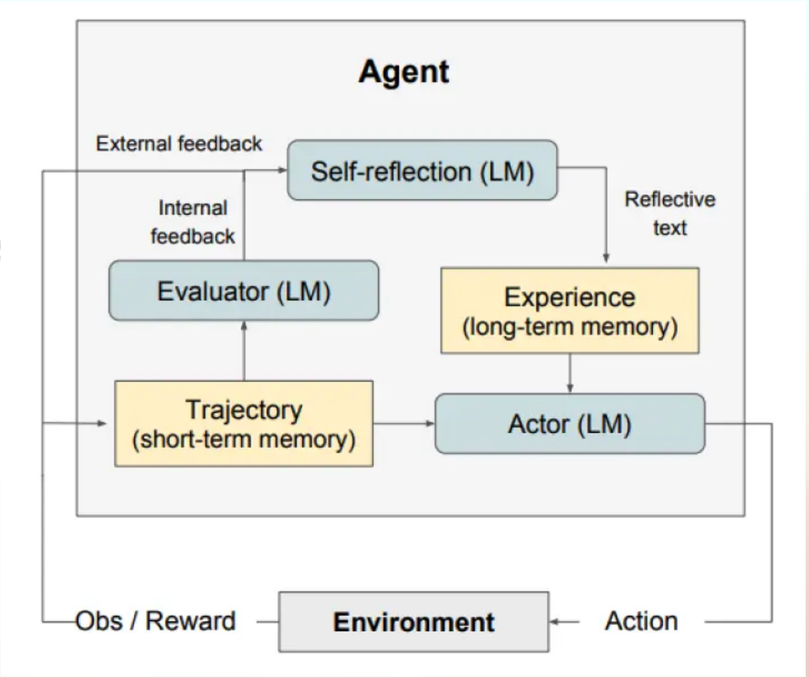
1. Generate an answer.
2. Evaluate for weaknesses and inconsistencies.
3. Refine the response based on self-critique.



Self-Improvement - Reflexion

- Model learns from **past mistakes** and adjusts future responses based on **prior failures**
- Improves from memory: past failures
- Improves across attempts (not just within one)

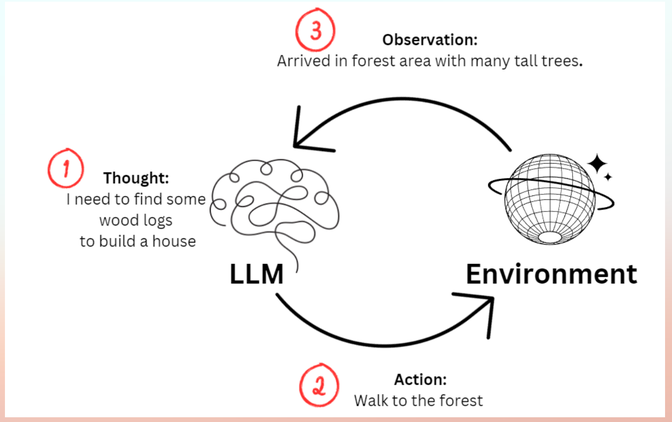
1. The model detects incorrect or weak responses.
2. Reflects on its mistakes and generates an improved response
3. Over multiple iterations, accuracy and reasoning improve.



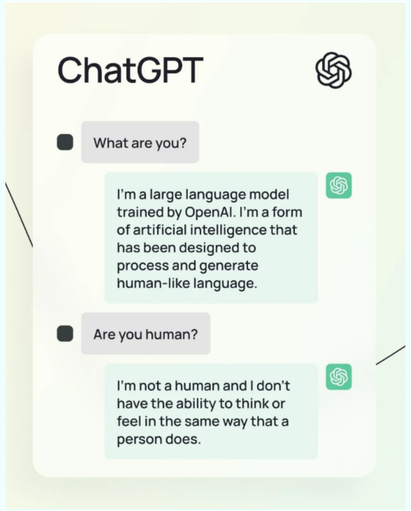
Acting & Tool Use (ReAct: Reasoning + Acting)

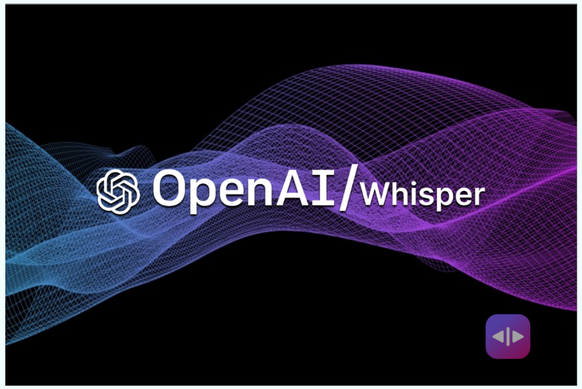
- Combines reasoning with external actions (retrieval, API calls, etc.).
- Model interacts dynamically with environments.
- Gets feedback from interacting with environment.

1. The model generates a reasoning plan.
2. It calls an external tool (e.g., web search, API, or database query).
3. The model incorporates the retrieved data into its final response.

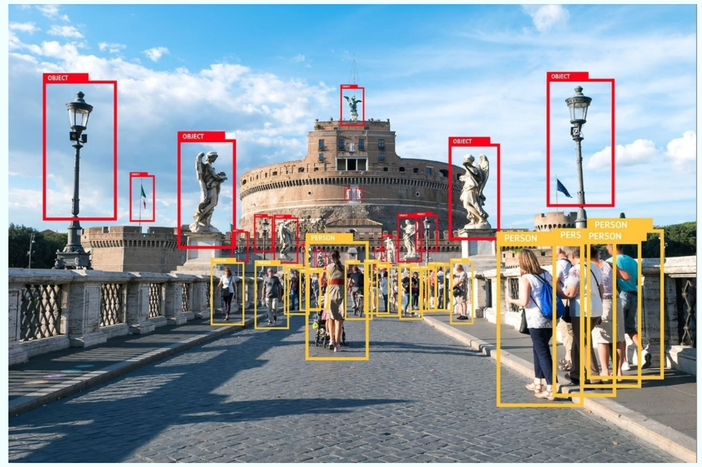


Applications of Transformers


ChatGPT
What are you?
I'm a large language model trained by OpenAI. I'm a form of artificial intelligence that has been designed to process and generate human-like language.
Are you human?
I'm not a human and I don't have the ability to think or feel in the same way that a person does.


OpenAI/Whisper


ALPHAGO


OBJECT
PERSON


DeepMind


Google
INTRODUCING
MED-PALM 2




Learned Policies
cook shrimp
push chairs
wipe wine
call elevator

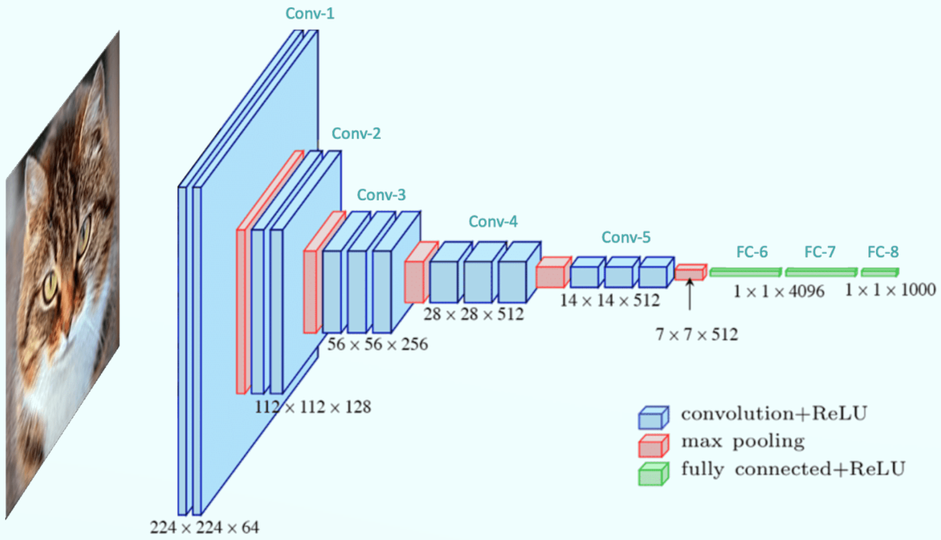

Google DeepMind


Mine Amethyst
Gather Cactus
Build Base
Fight Enderman
Build House
with Human Feedback
Mine Gold

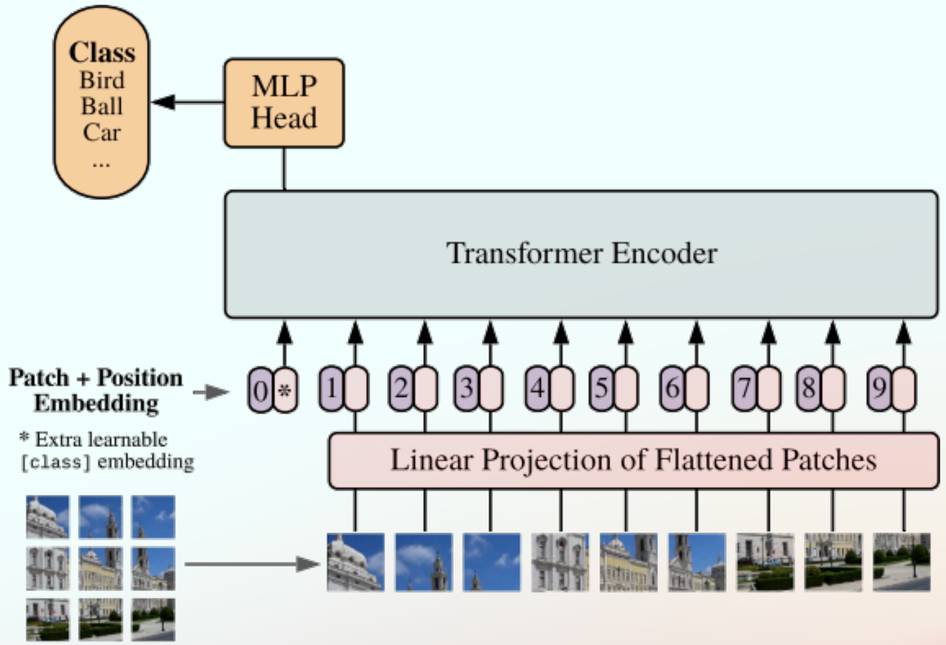
Vision Transformers

Dosovitskiy et al., An Image is Worth 16x16 Words: Transformers for Image Recognition At Scale, ICLR 2021.

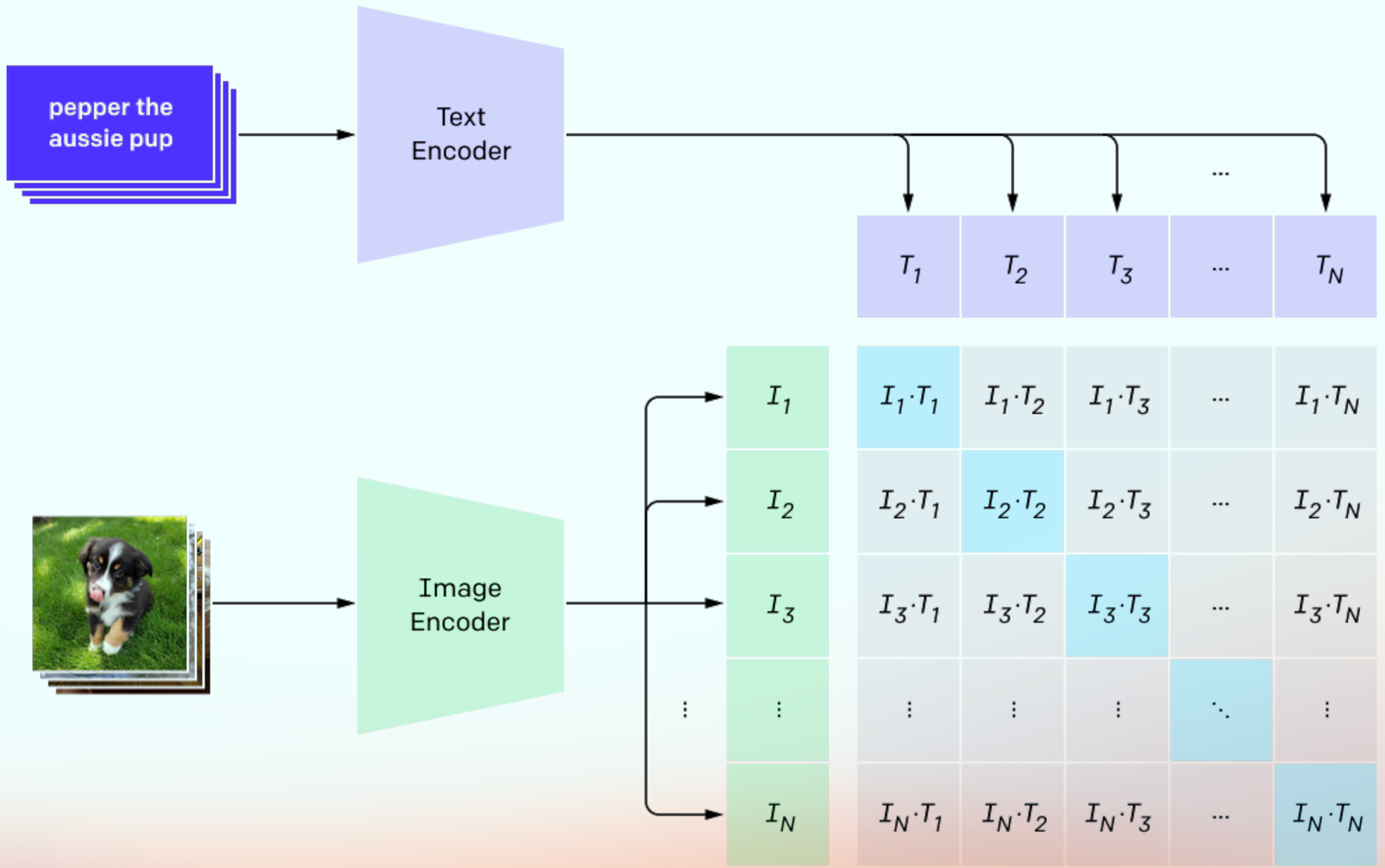
ViTs vs CNNs



OR



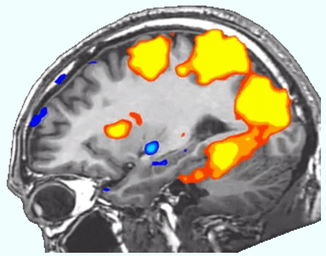
CLIP


pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 $I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
I_2 $I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
I_3 $I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
I_N $I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

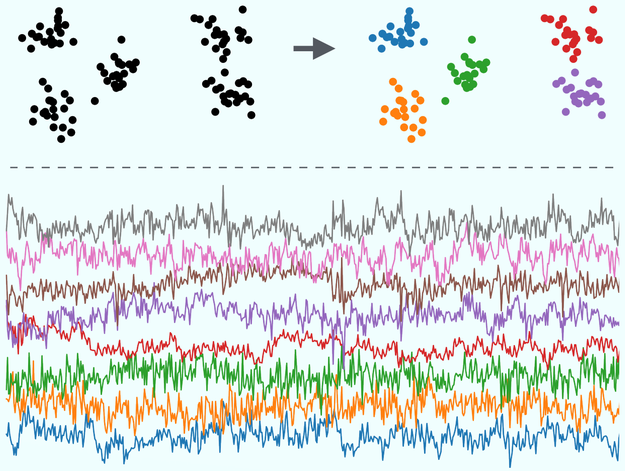
Transformers for Neuroscience

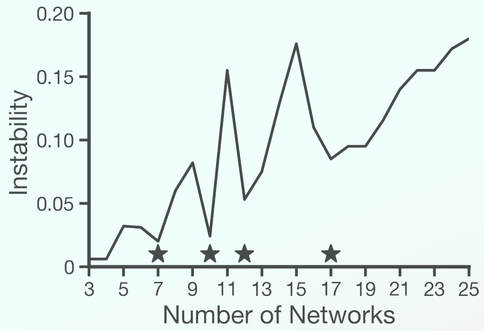
fMRI & Yeo Networks

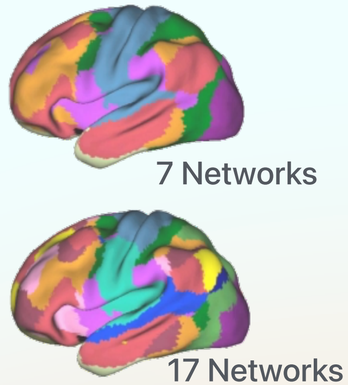
Correlated spontaneous activity captured by resting-state fMRI can reveal reproducible network structure at different spatial scales, giving a view into the organization of the brain.



fMRI → correlated fluctuations







Visual Center

processes visual input; supports perception

Dorsal Attention

goal-directed attention; orienting to stimuli

Control

cognitive control; task switching; executive function

Salience

regulating attention; autonomic functions; decision making

Limbic

emotion, motivation; memory-related processing

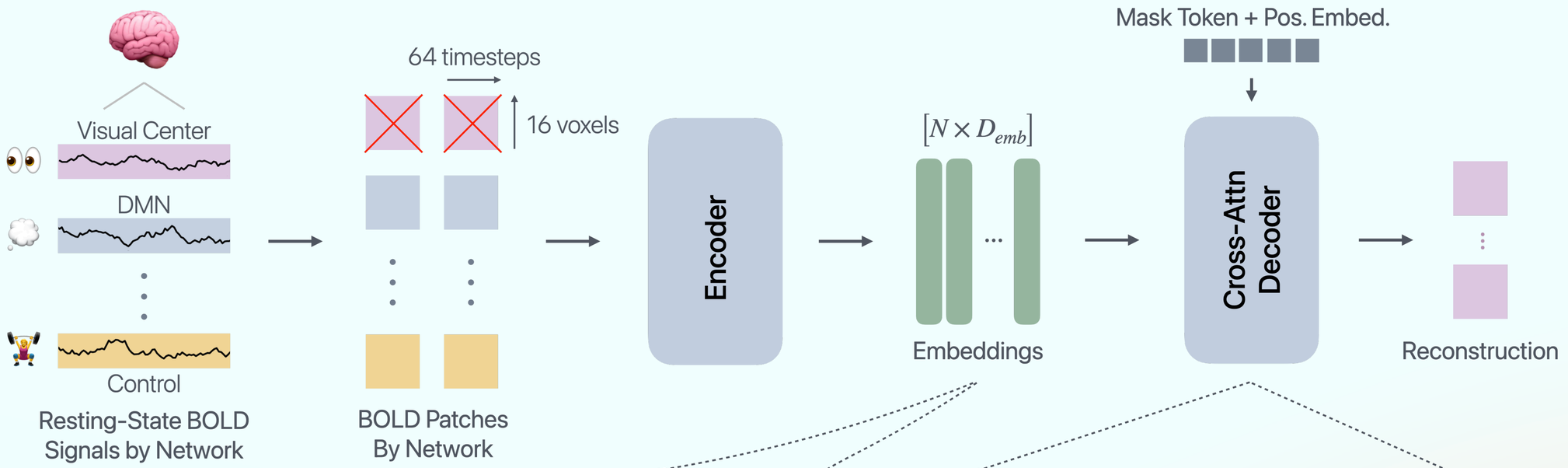
DMN

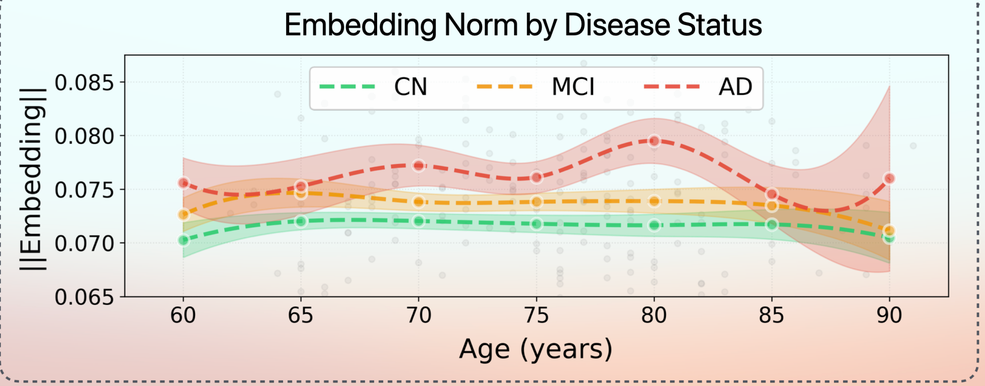
internally-directed cognition; day-dreaming

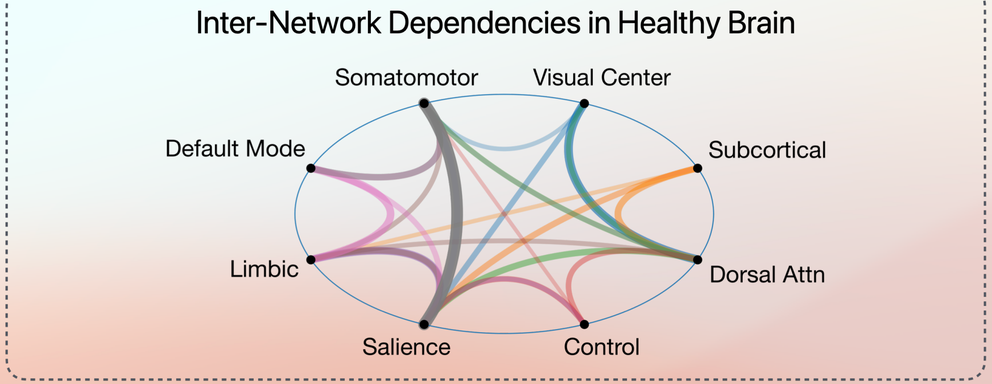
Somatomotor

voluntary movement and sensory feedback

Grounded fMRI Representations

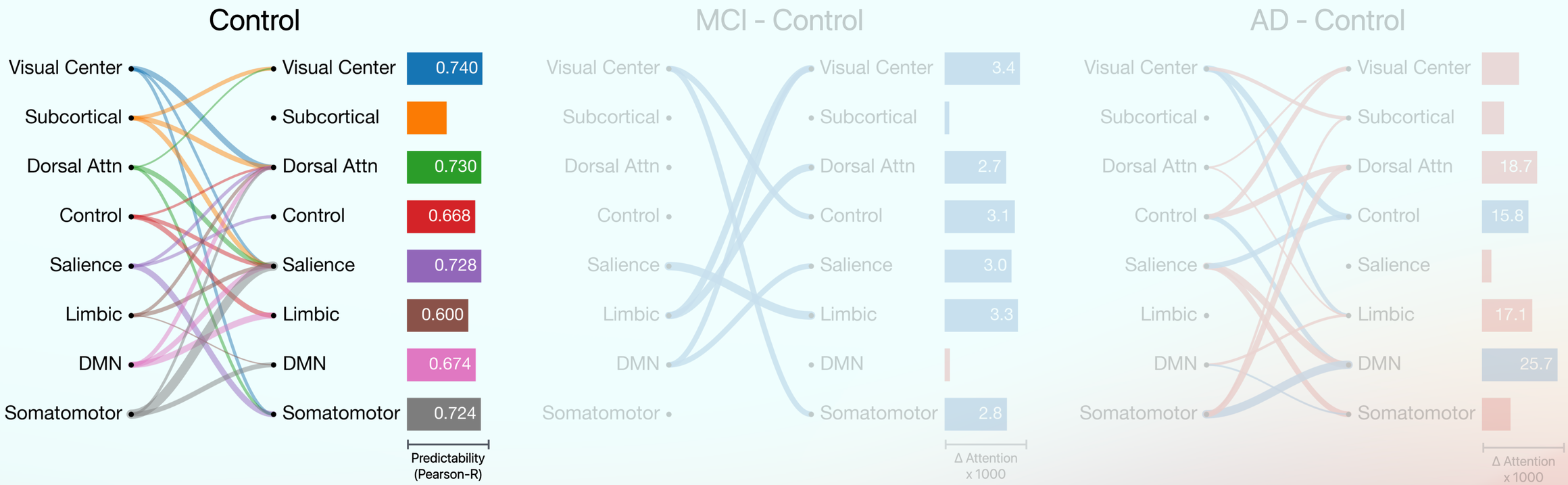




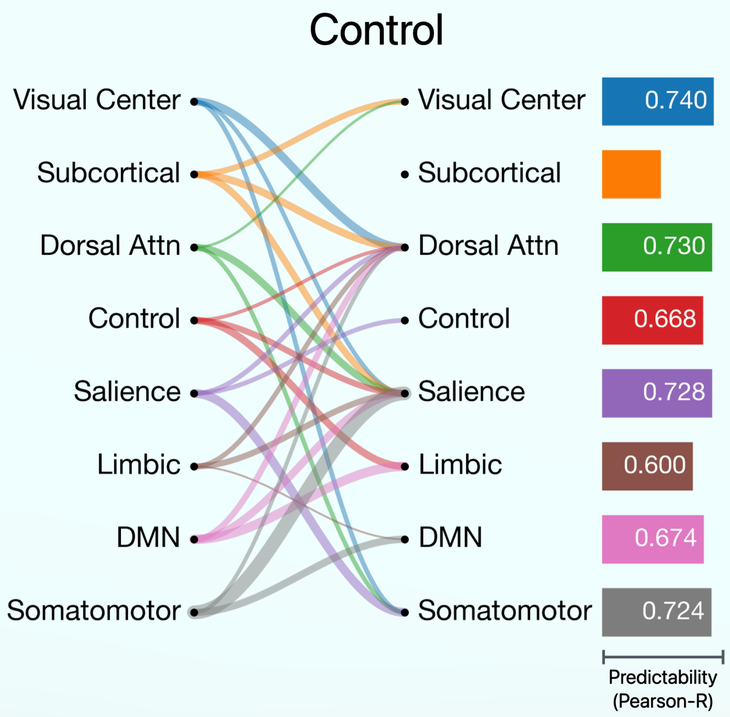


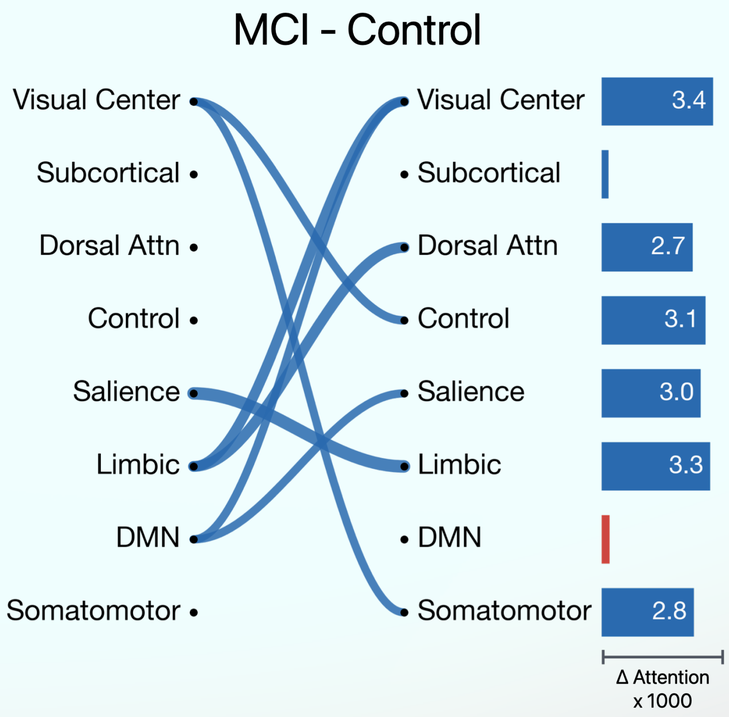
Submitted to MICCAI 2026

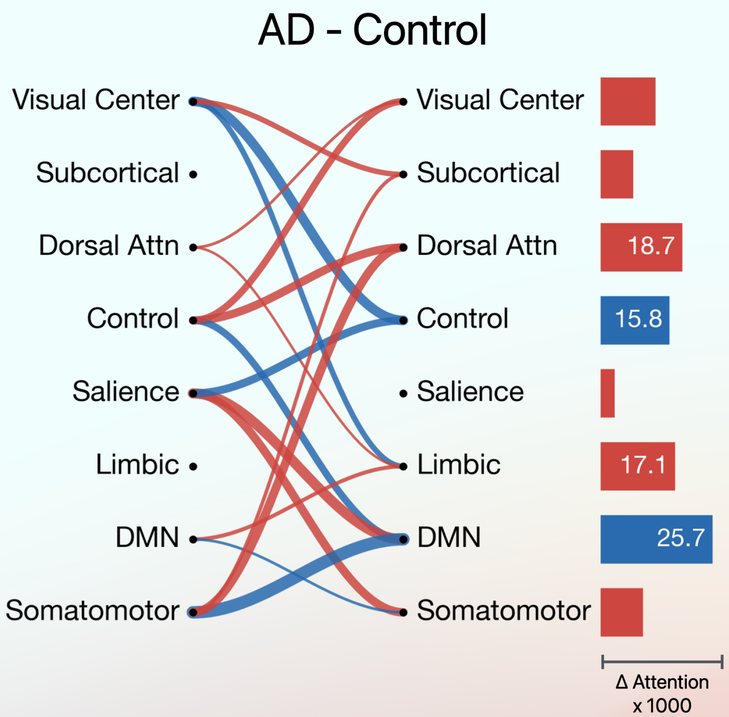
Inter-Network Dependence



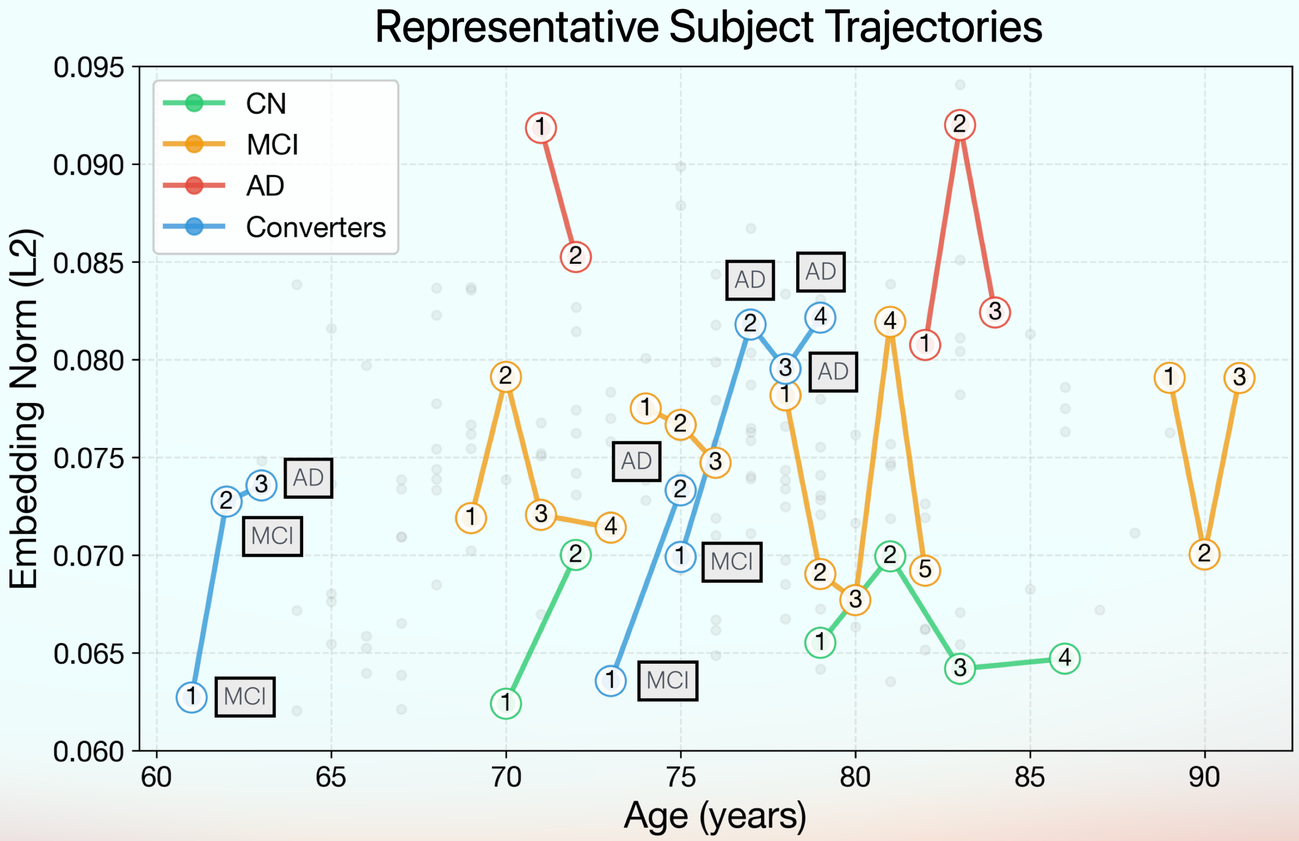
Inter-Network Dependence

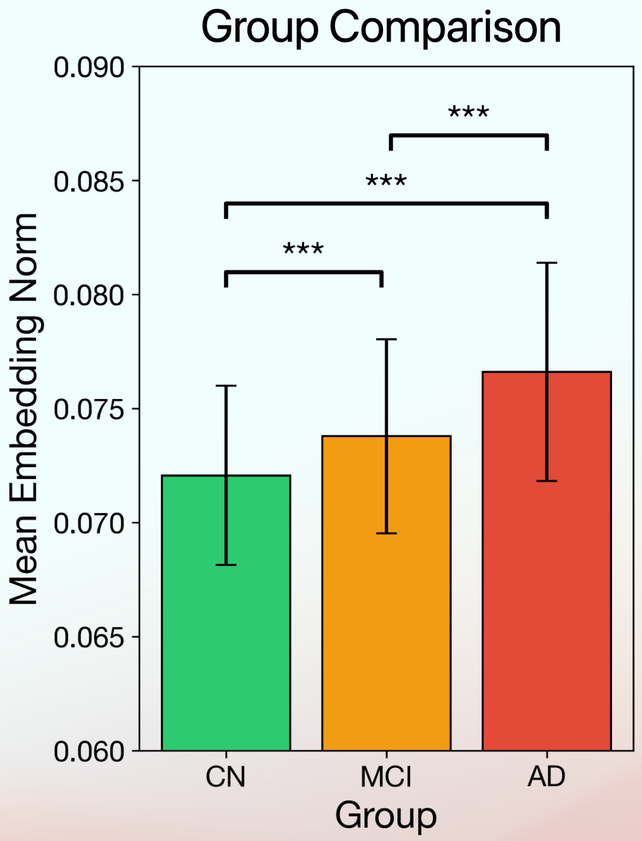






Longitudinal Charting for AD





Read the Paper

- Contact email: karanps@stanford.edu
- Paper: https://arxiv.org/abs/2603.00786



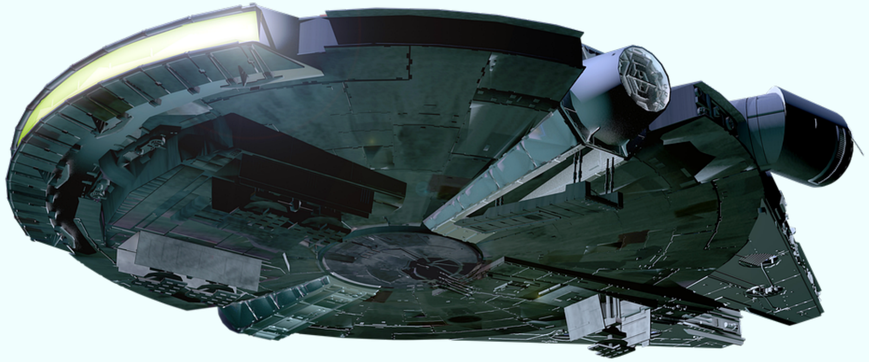
The Future — What's Next?

The Future (What's Next?)



- Can enable a lot more applications:
  - Generalist Agents
  - Domain-specific "Foundation models" - DoctorGPT, LawyerGPT, ...
  - Real-world integration
    - Personalized education and tutoring systems
    - Advanced healthcare diagnostics, environmental monitoring
    - Real-time multilingual communication
    - Interactive entertainment & gaming (e.g. NPCs)

The Future (What's Missing?)



- Missing Ingredients (to AGI/ASI?):
  - Reducing computation complexity
  - Infinite/external memory: like Neural Turing Machines
  - Alignment with language models of human brain
  - Multi-sensory multimodal embodiment (e.g. intuitive physics and commonsense)
  - Infinite self-improvement capabilities (i.e. continual or lifelong learning)
  - Complete autonomy and long-horizon decision-making
  - Emotional intelligence and social understanding
  - Ethical reasoning and value alignment

Minified and On-Device LLMs

- Big trend of using LLMs for applications and everyday purposes
- A requirement is ability to run quickly and easily on-devices
- More work on smaller open-source models (e.g. DeepSeek, LLaMA, Mistral)
- Edge deployment and privacy-preserving AI
- Goal: ability to finetune and run models locally, even on your phone

Limits of Scaling?

- Diminishing returns from scaling alone (one-size-fits-all and frozen models)
- Shift toward:
  - Post-training methods
  - Data quality (better data, curriculum, structure, and usage)
  - Inference-time compute
- Risk: too much post-training → “catastrophic forgetting”
- How else do we break through this scaling law limit?

Limits of Scaling? What's Next

- New architectures, higher-quality data, and improved training procedures
- Another goal: make advanced capabilities available for smaller models
- Theoretical and interpretability research, cogsci/psych and neuro-inspired work
- Next step: models that are not just larger, but smarter and more adaptable

Hallucination in LLMs

Hallucination Basics

- What is "Hallucination"?
  - Producing false or misleading information
  - Making up facts, citations, or details
  - Failing to reflect uncertainty when knowledge is missing
- Why it happens:
  - Models optimize for plausibility not truth (produce *likely*, not *true* text)
  - Lack of grounding in external knowledge or environment
  - Ambiguity or underspecified prompts

Hallucination Basics

- Why it's a problem:
  - Reduces trust and reliability
  - Especially critical in high-stakes domains (health, law, etc.)
  - Hard to detect automatically
- Subtle challenges:
  - Not all errors are hallucinations
  - Truth can depend on context or assumptions
  - Some outputs are unverifiable (not clearly true or false)

But, what exactly is a hallucination?

No Unified Definition!

- Hallucination widely used but poorly defined
- Different fields define it differently:
  - Summarization → not faithful to source
  - QA → factually incorrect
  - Agents → incorrect actions
- Same output can be:
  - Hallucination in one setting
  - Correct in another

A Unified Definition of Hallucination: It's The World Model, Stupid!

Emmy Liu[1†] Varun Gangal[2†] Chelsea Zou[3†] Michael Yu[4†] Xiaoqi Huang[4†] Alex Chang[4†]
Zhuofu Tao[4†] Karan Singh[3†] Sachin Kumar[5†] Steven Y. Feng[3†]

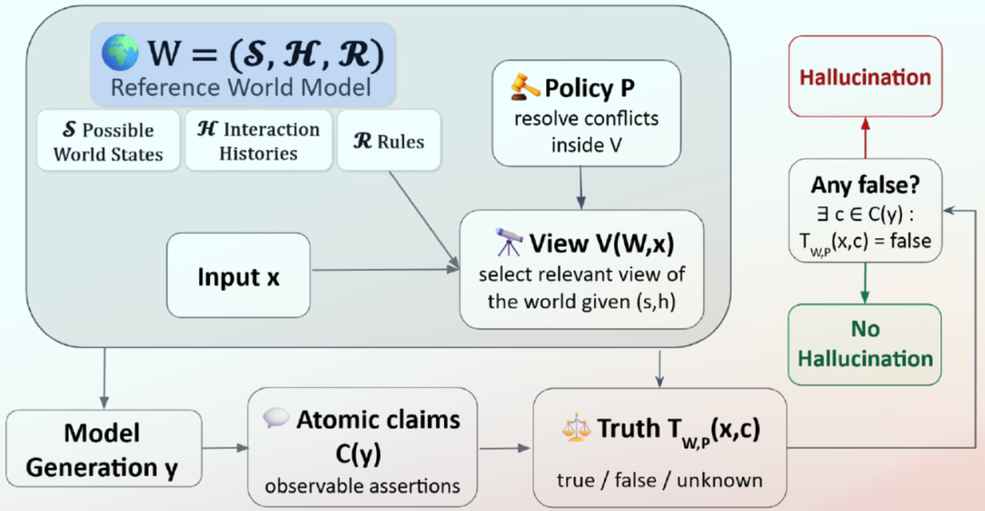


Figure 1. Hallucination as inaccurate (internal) world modeling

Key Insight: It's About the World Model

- Core idea: Hallucination = incorrect internal world model
- Definition (intuition):
  - Model outputs imply a "world"
  - That world may: match reality ✅ OR contradict reality ❌
- Hallucination occurs when:
  - Model's implied world ≠ true world
- Not just "wrong answer", but wrong beliefs about the world

Takeaway: Hallucination is a world-modeling error

The Framework (W, V, P)

Every hallucination depends on 3 things:

1. Reference World Model (W):

   What is actually true (doc, DB, env, real world)

2. View (V):

   What the model can see (retrieval, observations)

3. Conflict Policy (P):

   What to trust when sources disagree (e.g., "docs > memory")

Hallucination = model makes a claim that is false under (W, V, P)

All prior definitions differ only in W, V, P choices

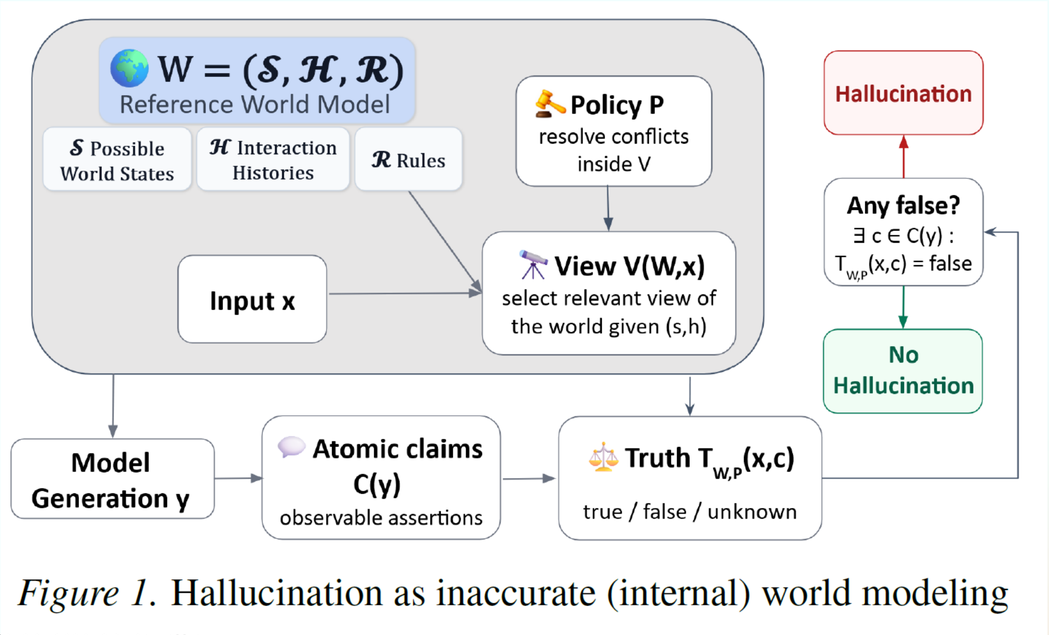


Figure 1. Hallucination as inaccurate (internal) world modeling

Unifying Different Settings

- Different tasks = different world definitions
- Key insight: same model behavior, different hallucination label
- Example: contradicts doc but true IRL → hallucination (summary) but NOT hallucination (QA)
- Hallucination is task-dependent via (W, V, P)

Example	$\mathcal{W}$ **(Reference World)**	V **(Visible Inputs)**	$\mathcal{P}$ **(Conflict Policy)**	**Hallucination Type**
Summarization	Source document	Full document	Document is truth	Intrinsic contradiction
Open QA	Real-world facts	Parametric memory only	Real world is truth	Incorrect factual claim
RAG	Entire retrieval corpus + world knowledge	Retrieved docs (+ memory)	Docs override memory	Retrieved context contradiction
Agentic	Environment DOM	DOM observation	Environment is truth	Observation hallucination

Table 1. Comparison of hallucination types under our unified framework. Each type arises from different choices of the reference world model $\mathcal{W}$, view function V, and conflict policy $\mathcal{P}$.

Not All Errors Are Hallucinations

- Different types of errors:
  - Hallucination: wrong belief about world
  - Planning error: correct beliefs, wrong action
  - Reward / incentive error: knows it's wrong, but says it anyway
  - Key distinction: hallucination = belief error, NOT just any mistake
- Example:
  - Clicking non-existent button → hallucination
  - Choosing bad strategy → planning error
- Takeaway: Hallucination ≠ all model errors

Why This Definition Matters

1. Makes assumptions explicit
   - ❖ What is truth?
   - ❖ What does the model see?
   - ❖ What overrides what?
2. Enables better evaluation
   - ❖ Compare across benchmarks
   - ❖ Consistent definitions

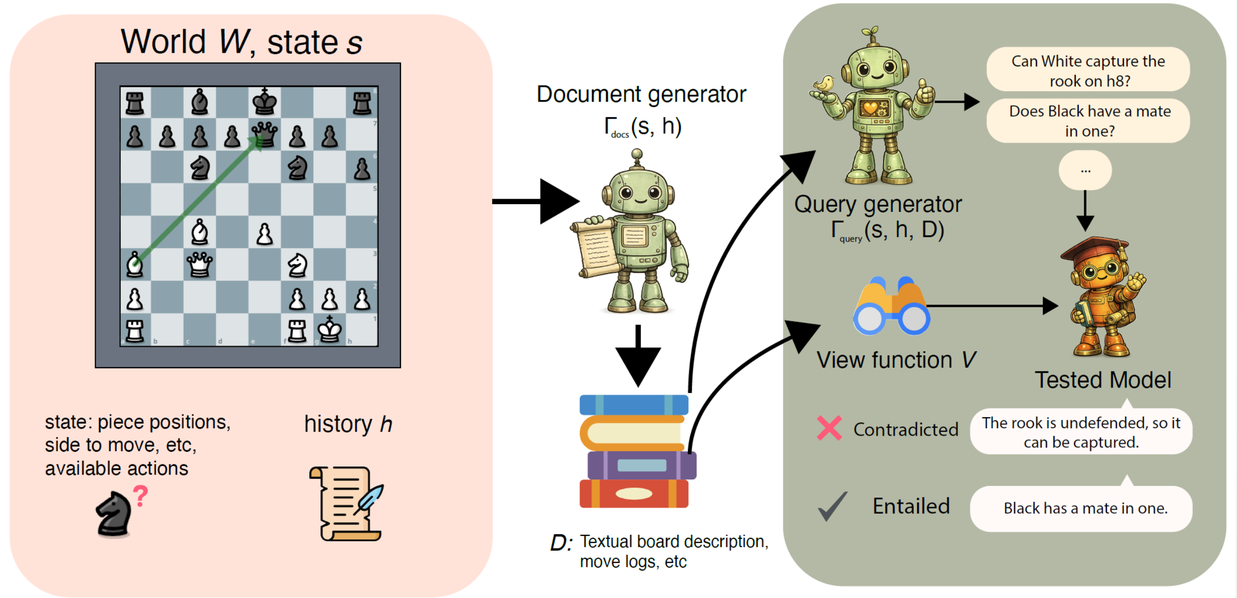


Figure 2. Overview of the chess scenario environment

Why This Definition Matters

3. Explains why methods work

- RAG → improves view (V)
- Calibration → changes confidence, not world model

4. Enables scalable benchmarks

- Synthetic environments
- Fully known "ground truth world"
- In progress: HalluWorld benchmark

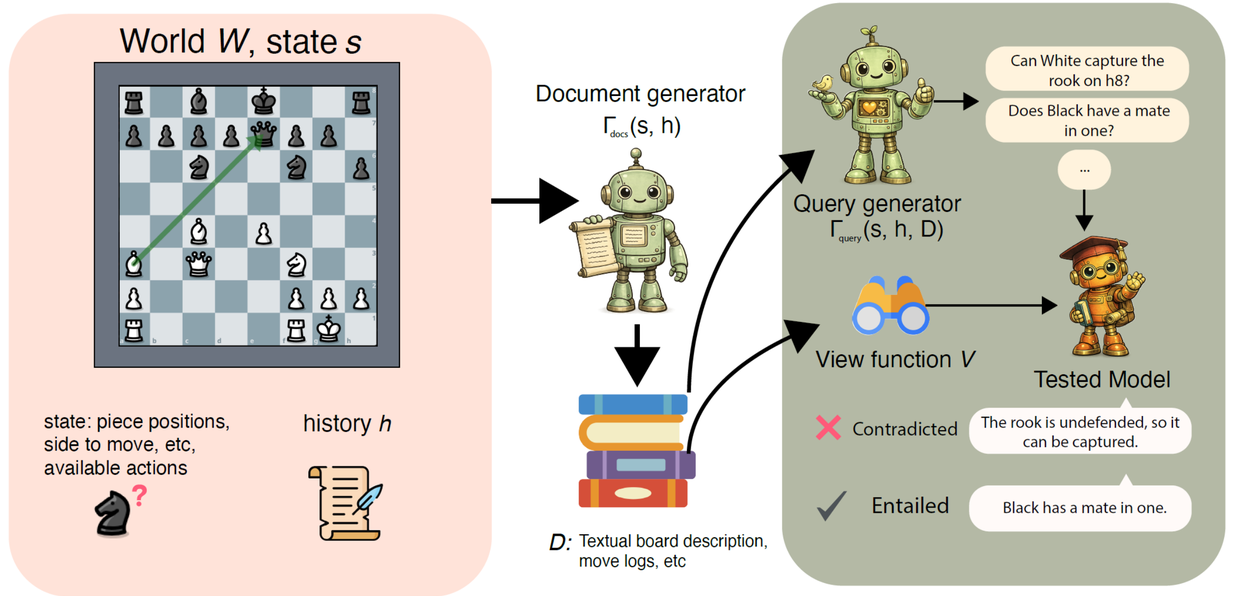


Figure 2. Overview of the chess scenario environment

Big Picture

- Hallucination = incorrect world modeling
- Depends on:
  - What is true (W)
  - What is seen (V)
  - How conflicts resolve (P)
- Key implications:
  - Explains confusion across papers
  - Separates hallucination from other failures
  - Guides better evaluation + mitigation

https://arxiv.org/abs/2512.21577

HalluWorld benchmark:
github.com/DegenAI-Labs/HalluWorld/

Memory & Continual Learning

Long-Term Memory

- Many models are stateless → no persistent learning across interactions
- We want to:
  - Store information across sessions
  - Retrieve and reuse past experiences
  - Update knowledge over time
- Why it matters:
  - Enables personalization (user preferences, history)
  - Supports continual learning from real-world use
  - Improves long-horizon reasoning and planning

Long-Term Memory

- How it's done (today):
  - External memory (vector DBs, retrieval systems)
  - Summarization / compression of past interactions
  - Structured memory (lists, graphs, databases)
- What's missing:
  - Reliable memory updating (avoid errors / drift)
  - Integration with model parameters (not just retrieval)
  - Efficient scaling over long time horizons

Lifelong Learning like Humans?

- AI systems that continuously improve by learning after deployment
- Source of knowledge: implicit feedback and real-world experience
- Infinite and permanent fundamental self-improvement
- Similar to humans: we constantly learn everyday from every interaction (no "finetuning" stage)
- Inference-time: not learning anything "permanent" or updating its “brain”
- Debate: do we truly need parametric updates, or is in-context learning enough?
- Very challenging, could be the key to AGI!

Current Work vs. What's Needed

- Finetune a small model on traces from better model
  - Re-training and distillation, not “continual learning”
- Self-improvement and reflection
  - Mainly context and inference-time-only enhancements, not updating the "brain"
- What mechanisms will support real lifelong learning?
  - Gradient updates?
  - Memory architectures?
  - Meta-learning?

Model Editing for Continual Learning?

- Mech interpretability and model editing: edit specific nodes
- Causal intervention method to trace decisive neuron activations for model factual predictions
- Rank-One Model Editing (ROME) to modify model weights for updating factual associations
- Weaknesses:
  - Only works for simple facts (what about skills or capabilities?)
  - Cannot easily propagate changes to other dependent or related facts

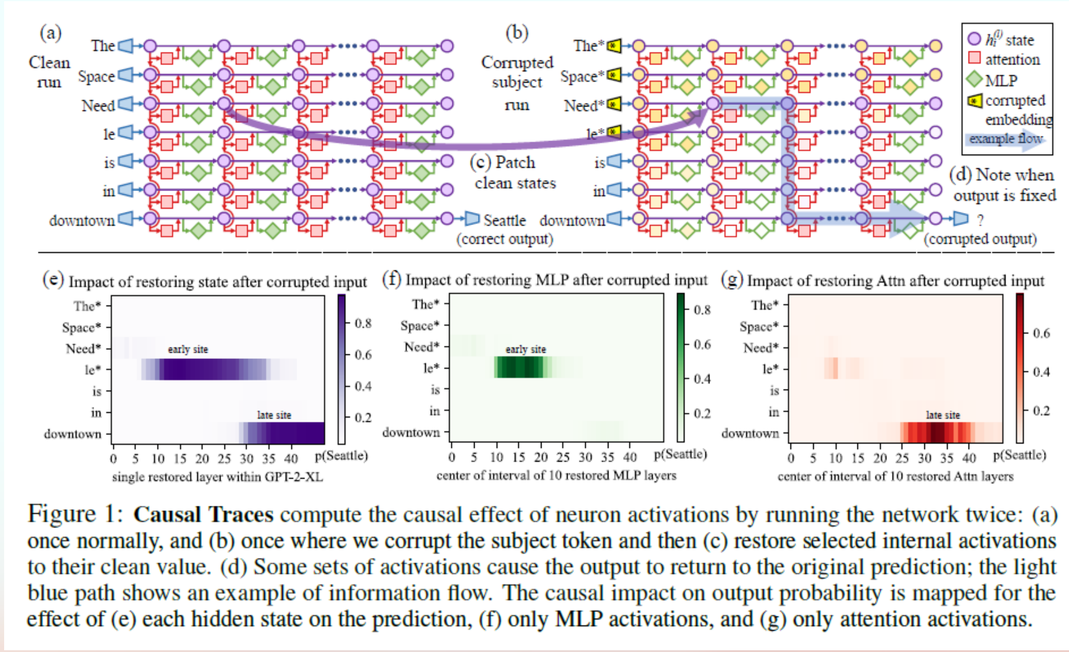


Figure 1: **Causal Traces** compute the causal effect of neuron activations by running the network twice: (a) once normally, and (b) once where we corrupt the subject token and then (c) restore selected internal activations to their clean value. (d) Some sets of activations cause the output to return to the original prediction; the light blue path shows an example of information flow. The causal impact on output probability is mapped for the effect of (e) each hidden state on the prediction, (f) only MLP activations, and (g) only attention activations.

https://arxiv.org/abs/2202.05262

Interpretability & Alignment

Interpretability of LLMs

- Huge # of parameters trained on tons of data → “black-box” that is hard to interpret
- Goal of interpretability: understand internal representations, not just outputs
- Detect failure modes (hallucination, deception, bias)
- Better ideas of what/how to improve, easier control, and better alignment & safety
- Mechanistic interpretability: understand what the model knows and how it uses it
  - Circuits, features, representations
  - Unpack the "black box" of models for clearer insight into how they work

The Alignment Problem

- How do we ensure models do what we *intend*?
- Models optimize training objectives, but objectives ≠ human values
- Key issues:
  - Shortcuts to achieve goal
  - Unintended behaviors in new settings
  - Difficulty specifying “correct” behavior

Failure Modes & Alignment Risks

- Reward hacking: optimize the metric, not the true goal
- Deception / hidden objectives
  - Model behaves well during training but may pursue different goals internally
- CoT faithfulness
  - Reasoning traces ≠ actual internal reasoning
  - Explanations may be post-hoc
- Distribution shift
  - Behavior changes in new environments
  - Alignment may not generalize

Alignment Techniques

- Human / AI feedback
  - RLHF, RLAIF
  - preference-based training
- Interpretability
  - Directly inspect and control internal representations
- Process supervision
  - Reward reasoning steps, not just outcomes

Alignment Techniques

- Constitutional AI
  - Use rules/principles to guide behavior
  - Reduce reliance on human labeling
- Scalable oversight
  - Use models to supervise other models
  - Amplify limited human feedback

Alignment: ensure models pursue the right goals and behave reliably under all conditions

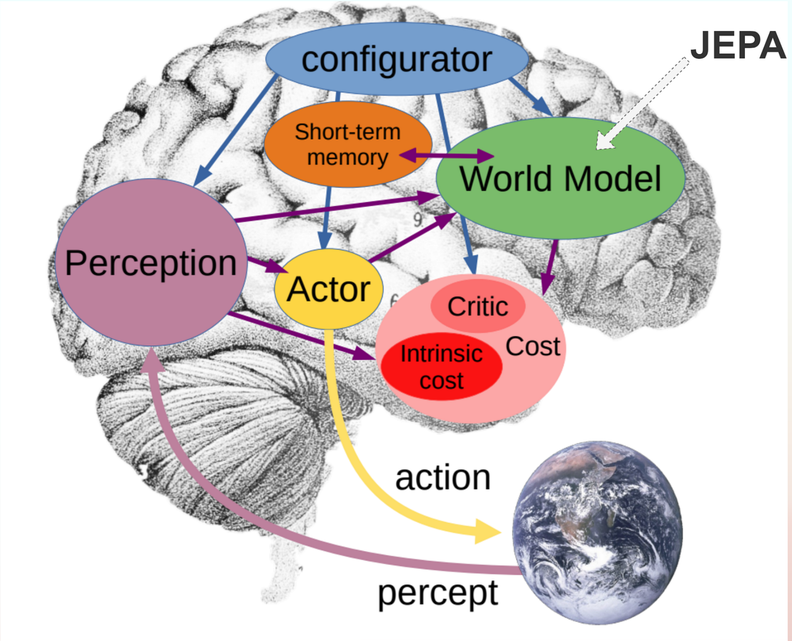
What Comes After Transformers?

Beyond Transformers

- Transformers dominate today
- But key limitations remain:
  - Efficiency
  - Long-context reasoning
  - Structured world understanding
- Emerging directions:
  - World models (structure + prediction)
  - State Space Models (efficient sequence modeling)

World Models & JEPA

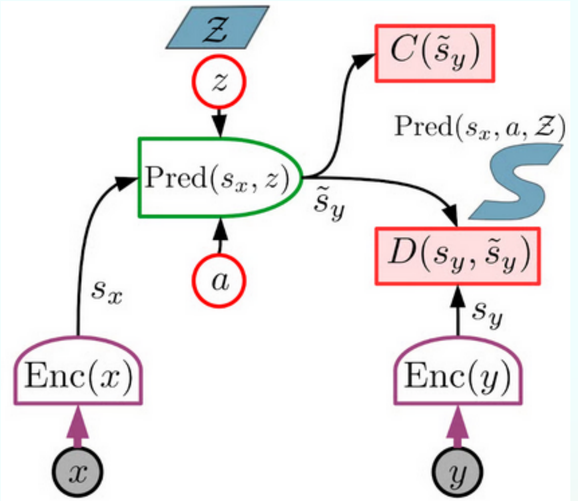
- Moving beyond next-token prediction → learning how the world works
- World models:
  - Learn structured representations of environments
  - Predict future states, not just text tokens
  - Enable planning and reasoning over time



LeCun (2022), A Path Towards Autonomous Machine Intelligence

World Models & JEPA

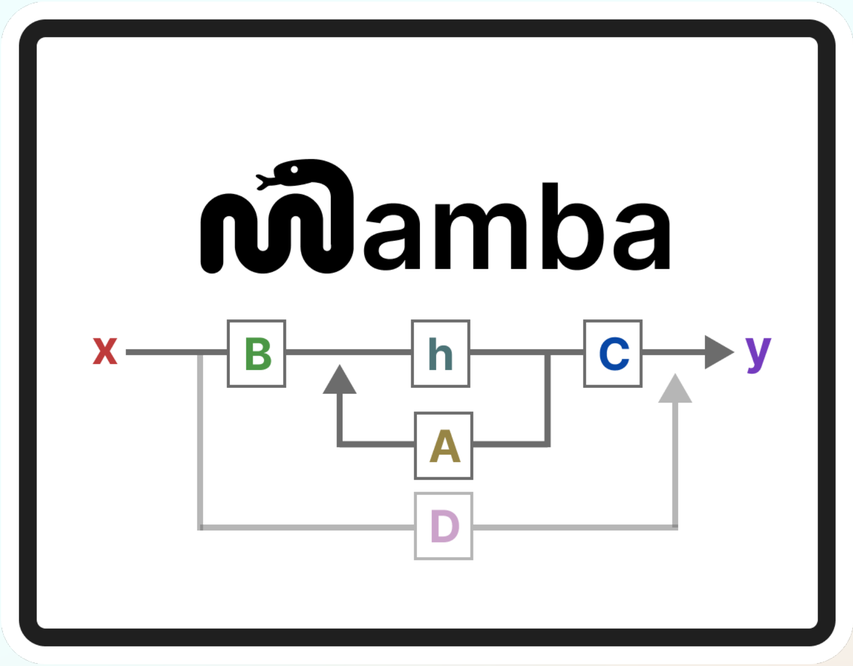
- JEPA (Joint Embedding Predictive Architecture):
  - Predict representations instead of raw outputs
  - Focus on latent structure rather than surface details
  - Avoids modeling unnecessary low-level noise
- Why this matters:
  - More data-efficient learning
  - Better abstraction and generalization
  - Closer to how humans learn (predicting structure, not pixels/tokens)



LeCun (2022), A Path Towards Autonomous Machine Intelligence

State Space Models (SSMs)

- Key Idea:
  - Model sequences using continuous state updates
  - Maintain a compressed internal state over time
- Why?
  - Linear-time scaling (vs. quadratic attention)
  - Efficient for long sequences
  - Strong performance on long-context tasks



Ref: Gu & Dao (2023). Mamba: Linear-Time Sequence Modeling with Selective State Spaces

State Space Models (SSMs)

- Example: Mamba (selective state space model)
- Trade-offs:
  - Less flexible than attention in some settings
  - Still early: an active area of research
- **Big idea: More efficient sequence modeling without attention**

Ref: Gu & Dao (2023). Mamba: Linear-Time Sequence Modeling with Selective State Spaces

Thanks for Listening!