Instruction Tuning

Yu Meng
University of Virginia
yumeng5@virginia.edu

Nov 04, 2024

Reminder

Assignment 4 is due today **11:59pm**!

UNIVERSITY of VIRGINIA

Join at slido.com #3947 182

Overview of Course Contents

- Week 1: Logistics & Overview
- Week 2: N-gram Language Models
- Week 3: Word Senses, Semantics & Classic Word Representations
- Week 4: Word Embeddings
- Week 5: Sequence Modeling and Neural Language Models
- Week 6-7: Language Modeling with Transformers (Pretraining + Fine-tuning)
- Week 8: Large Language Models (LLMs) & In-context Learning
- Week 9-10: Reasoning, Knowledge, and Retrieval-Augmented Generation (RAG)
- Week 11: LLM Alignment
- Week 12: Language Agents
- Week 13: Recap + Future of NLP
- Week 15 (after Thanksgiving): Project Presentations

(Recap) Hallucination

- **Hallucination**: LM generates information that is factually incorrect, misleading, or fabricated, even though it may sound plausible or convincing
- Why does hallucination happen?
    - Limited knowledge: LLMs are trained on finite datasets, which don't have access to all possible information; when asked about topics outside their training data, they may generate plausible-sounding but incorrect responses
    - Overgeneralization: LLMs may apply patterns they've learned from one context to another where they don't apply, leading to incorrect conclusions
    - Lack of common sense: While LLMs can process and generate human-like text, they often lack the ability to apply commonsense reasoning to their outputs
    - …

(Recap) Non-parametric Knowledge

- **Non-parametric knowledge**: (external) information not stored in the model's parameters but can be accessed or retrieved when needed
- Examples:
  - External knowledge bases/graphs
  - Pretraining corpora
  - User-provided documents/passages
- Non-parametric knowledge is typically used to **augment** parametric knowledge (typically via **retrieval**) for more accurate factoid question answering
- Benefits of **non-parametric knowledge**
  - Incorporate more information without increasing model size
  - Easier updates and modifications to the knowledge base
  - Improve model interpretability

(Recap) Sparse vs. Dense Retrieval

- **Sparse** retrieval: based on traditional IR techniques where the representations of documents and queries are sparse (most vector values are zero)
  - Example: TF-IDF
  - Pros: simple and interpretable
  - Cons: lack semantic understanding
- **Dense** retrieval: encode documents and queries into dense vectors (embeddings) using deep neural networks
  - Example: BERT-based encoding methods
  - Pros: semantic & contextualized understanding
  - Cons: computationally more expensive and less interpretable

(Recap) TF-IDF for Sparse Retrieval

- Example query and mini-corpus:

Query:	sweet love
Doc 1:	Sweet sweet nurse! Love?
Doc 2:	Sweet sorrow
Doc 3:	How sweet is love?
Doc 4:	Nurse!

- Query & document vectors:

			Query			
word	**cnt**	**tf**	**df**	**idf**	**tf-idf**	**n'lized** = tf-idf/$\lvert q \rvert$
sweet	1	1	3	0.125	0.125	0.383
nurse	0	0	2	0.301	0	0
love	1	1	2	0.301	0.301	0.924
how	0	0	1	0.602	0	0
sorrow	0	0	1	0.602	0	0
is	0	0	1	0.602	0	0

	Document 1				Document 2			
word	**cnt**	**tf**	**tf-idf**	**n'lized**	**cnt**	**tf**	**tf-idf**	**n'lized**
sweet	2	1.301	0.163	0.357	1	1.000	0.125	0.203
nurse	1	1.000	0.301	0.661	0	0	0	0
love	1	1.000	0.301	0.661	0	0	0	0
how	0	0	0	0	0	0	0	0
sorrow	0	0	0	0	1	1.000	0.602	0.979
is	0	0	0	0	0	0	0	0

$\cos(\boldsymbol{q}, \boldsymbol{d}_1) = 0.747$ $\cos(\boldsymbol{q}, \boldsymbol{d}_2) = 0.078$

Figure source: https://web.stanford.edu/~jurafsky/slp3/14.pdf

(Recap) Dense Retrieval

- Motivation: sparse retrieval (e.g., TF-IDF) relies on the exact overlap of words between the query and document without considering semantic similarity
- Solution: use a language model to obtain (dense) distributed representations of query and document
- The retriever language model is typically a small text encoder model (e.g., BERT)
  - Retrieval is a natural language understanding task
  - Encoder-only models are more efficient than LLMs for this purpose
- Both query and document representations are computed by text encoders

(Recap) Dense Retrieval: Cross-encoder

- Process query-document pairs together
- Relevance score produced directly by the model output
- (+) Capture intricate interactions between the query and the document
- (-) Not scalable to large retrieval corpus
- Good for small document sets

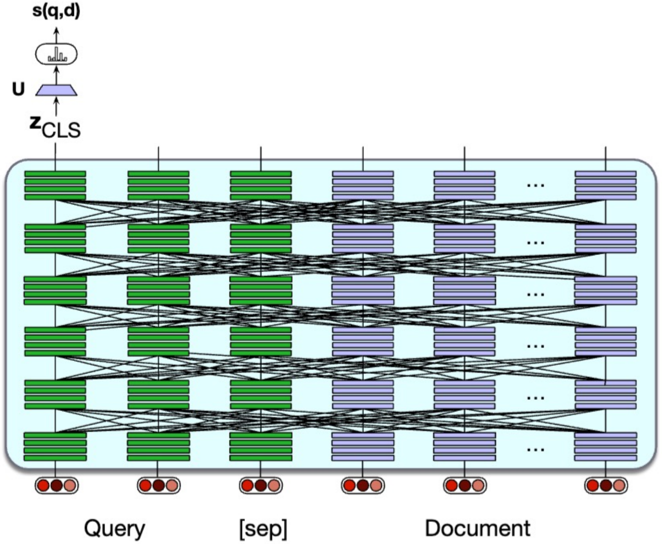


Figure source: https://web.stanford.edu/~jurafsky/slp3/14.pdf

(Recap) Dense Retrieval: Bi-encoder

- Independently encode the query and the document using two separate (but often identical) encoder models
- Use cosine similarity between the query and document vectors as relevance score
- (+) Document vectors can be precomputed
- (-) Cannot capture query-document interactions
- Common choice for large-scale retrieval

$$\cos(\boldsymbol{q}, \boldsymbol{d}) = \frac{\boldsymbol{q} \cdot \boldsymbol{d}}{|\boldsymbol{q}||\boldsymbol{d}|}$$

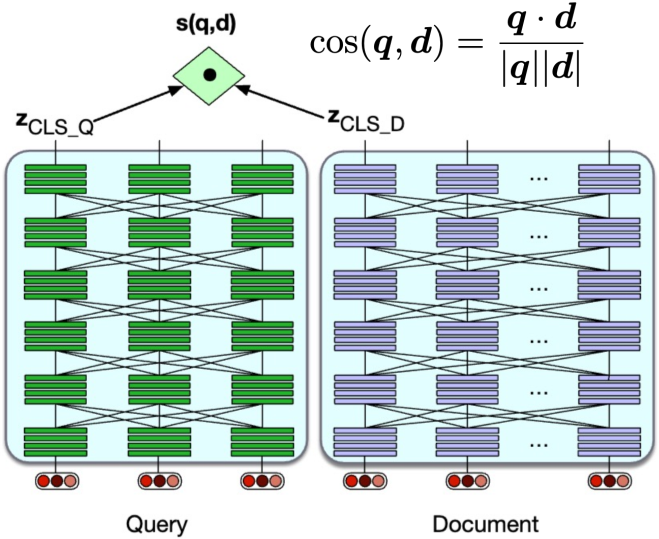


Figure source: https://web.stanford.edu/~jurafsky/slp3/14.pdf

(Recap) Evaluation of IR Systems

- Assume that each document returned by the IR system is either **relevant** to our purposes or **not relevant**
- Given a query, assume the system returns a set of ranked documents $\boldsymbol{T}$
  - A subset $\boldsymbol{R}$ of these are relevant (The remaining $\boldsymbol{N} = \boldsymbol{T} - \boldsymbol{R}$ is irrelevant)
  - There are $\boldsymbol{U}$ documents in the entire retrieval collection that are relevant to this query
- **Precision:** the fraction of the returned documents that are relevant

$$\text{Precision} = \frac{|R|}{|T|}$$

- **Recall:** the fraction of all relevant documents that are returned

$$\text{Recall} = \frac{|R|}{|U|}$$

UNIVERSITY of VIRGINIA

Join at
slido.com
#3947 182

(Recap) RAG vs. Direct Prompting

- Prompting relies on LM's parametric knowledge to directly answer the question:

$P(w|$`Q: Who wrote the book ‘‘The Origin of Species"? A:`$)$ prompt

- RAG prepends the set of retrieved passages to the question

Schematic of a RAG Prompt

```
retrieved passage 1

retrieved passage 2

...

retrieved passage n
```

Returned by the retriever

```
Based on these texts, answer this question: Q: Who wrote
the book ‘‘The Origin of Species"? A:
```

Figure source: https://web.stanford.edu/~jurafsky/slp3/14.pdf

UNIVERSITY of VIRGINIA

Join at slido.com #3947 182

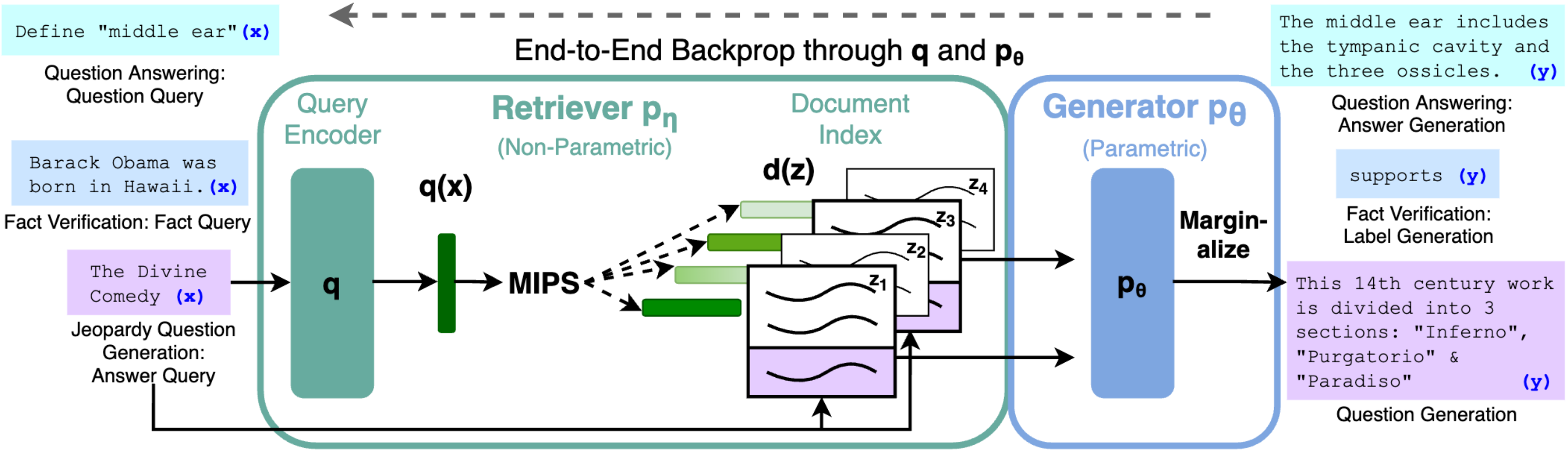
(Recap) RAG: A Latent Variable Model

The retrieved documents are treated as latent variables (z) for generation

$$p(\boldsymbol{y}|\boldsymbol{x}) = \sum_{\boldsymbol{z}\in\mathcal{D}} p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{z})$$

Retrieve document (z) based on query (x)

Generate answer (y) based on retrieved docs (z) and query (x)

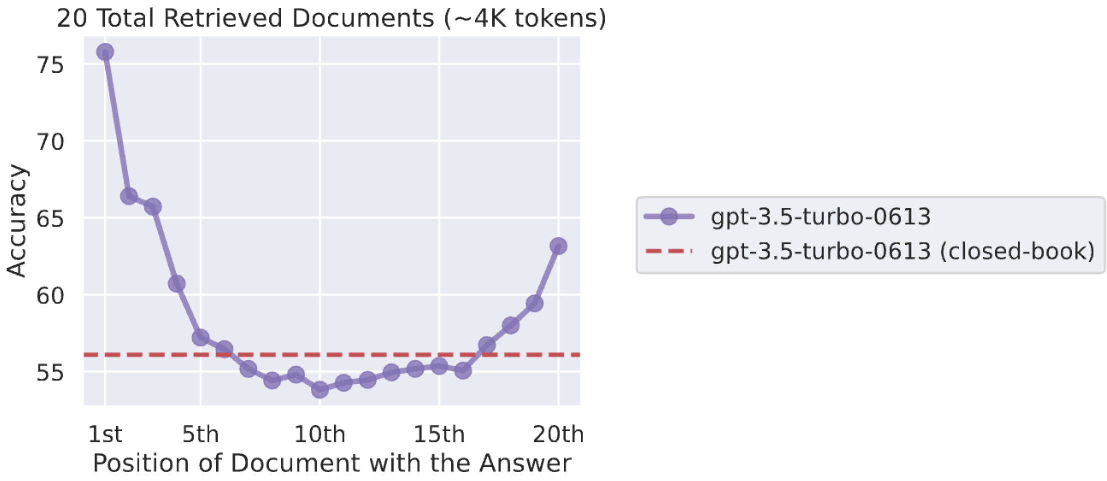


(Recap) RAG & Long Context Issues in LLMs

- RAG significantly increases the input sequence length to LLMs ("**long context**") by prepending multiple retrieved passages
- **Inefficiency**: the complexity of self-attention is quadratic wrt number of tokens
- **Irrelevant information**: LLMs might get distracted by irrelevant retrieval content
- **Lost in the middle**: LLMs tend to focus more on the beginning and end of the input sequence, but missing important information located in the middle of a long context
- **Performance saturation**: LLMs do not always effectively use the extra context (more retrieved documents)

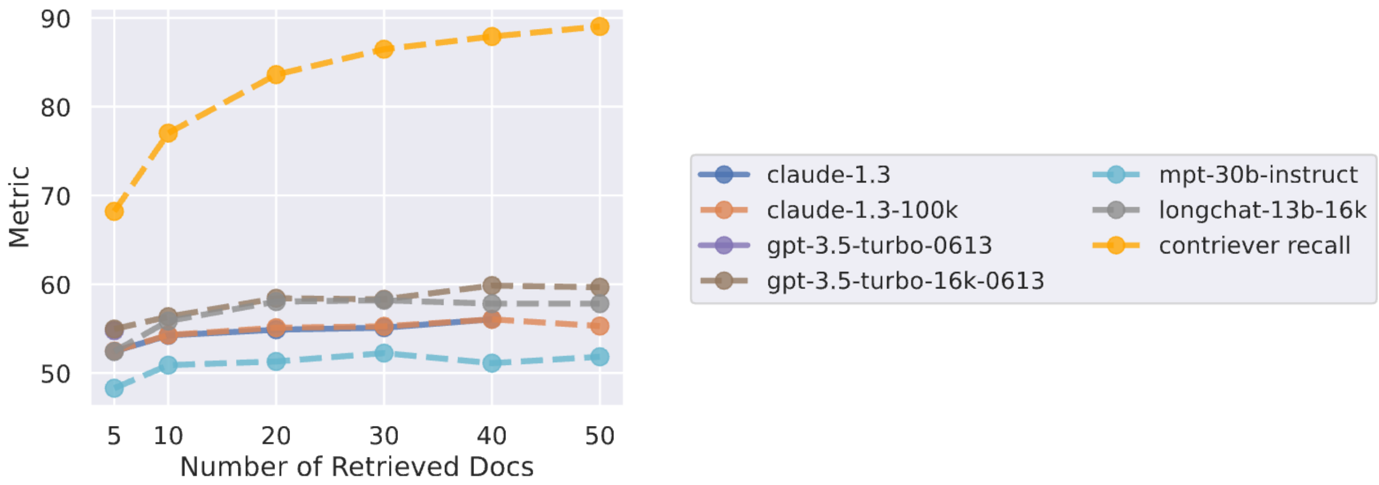
(Recap) Primacy & Recency Bias

- Exactly one of the documents contains the answer, with other "distractor" documents
- Vary the position of the gold document
- U-shaped performance curve: LLMs are better at using relevant information that occurs at the very beginning (**primacy bias**) or end of its input context (**recency bias**)



(Recap) Performance Saturation

- Retriever recall always improves with more retrieved docs
- LLM performance saturates long before retriever performance saturates (using more than 20 retrieved documents only marginally improves LLM performance)

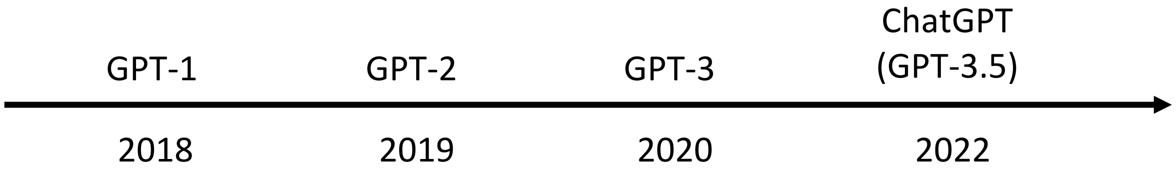


Agenda

- Introduction to LLM Alignment
- Instruction Tuning

The Evolution of GPT Models: ChatGPT

- GPT-1: decoder-only Transformer pretraining
- GPT-2: language model pretraining is multi-task learning
- GPT-3: scaling up & in-context learning
- ChatGPT: language model alignment



UNIVERSITY of VIRGINIA

Join at
slido.com
#3947 182

Overview: Language Model Alignment

- Ensure language models behaviors are aligned with human values and intent
- “HHH” criteria (Askell et al. 2021):
  - **Helpful**: Efficiently perform the task requested by the user
  - **Honest**: Give accurate information & express uncertainty
  - **Harmless**: Avoid offensive/discriminatory/biased outputs

Paper: https://arxiv.org/pdf/2112.00861

UNIVERSITY of VIRGINIA

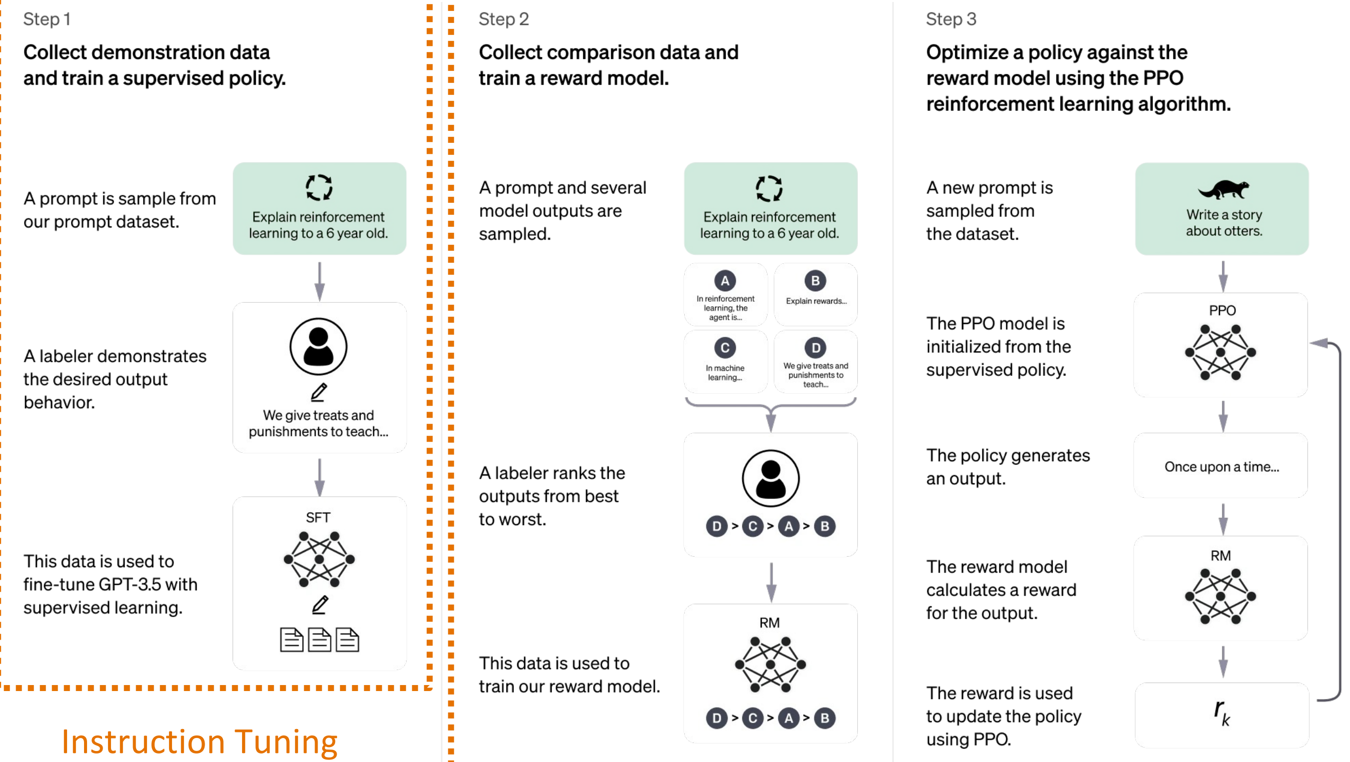
Join at
slido.com
#3947 182

Language Model Alignment: Post-training

- Pretrained language models are **not** aligned
- Objective mismatch
  - Pretraining is to predict the next word in a sentence
  - Does not involve understanding human intent/values
- Training data bias
  - Text from the internet can contain biased, harmful, or misleading information
  - LMs don’t distinguish between good and bad behavior in training data
- (Over-)generalization issues
  - LMs’ generalization can lead to outputs that are inappropriate in specific contexts
  - Might not align with intended ethics/honesty standard

Join at
slido.com
#3947 182

Language Model Alignment Techniques



Instruction Tuning

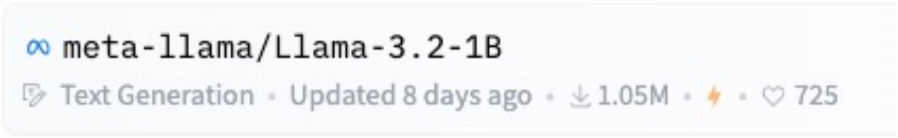
Reinforcement Learning from Human Feedback (RLHF)

Figure source: https://openai.com/index/chatgpt/

Join at slido.com #3947 182

Overview: Instruction Tuning

- Train an LM using a diverse set of tasks
  - Each task is framed as an **instruction** followed by an example of the desired output
  - The goal is to teach the model to follow specific instructions (human intent) effectively
- The resulting model can perform a variety of tasks **zero-shot** (w/o requiring in-context demonstrations)
- The instructions can also be in chat format – tuning an LM into a chatbot



Pretrained (base) model

Instruction-tuned (post-trained) model

Models: https://huggingface.co/meta-llama

Overview: RLHF

- Human feedback collection
  - Generate multiple responses using the model given the same prompt
  - Human evaluators rank responses of the model based on helpfulness/honesty/safety…
- Reward model training
  - A reward model is trained on human feedback data to predict the quality of responses
  - Higher reward = more preferred by human evaluators
- Policy optimization
  - Use reinforcement learning algorithms to further train the LM to maximize the reward predicted by the reward model
  - Encourage the model to produce outputs that align better with human preferences

Agenda

- Introduction to LLM Alignment
- Instruction Tuning

Instruction Tuning: Introduction

- **Setting**: fine-tune LLMs with task-specific instructions on diverse tasks
- **Goal**: enable LLM to better understand user prompts and generalize to a wide range of (unseen) tasks **zero-shot**

FINETUNED LANGUAGE MODELS ARE ZERO-SHOT LEARNERS

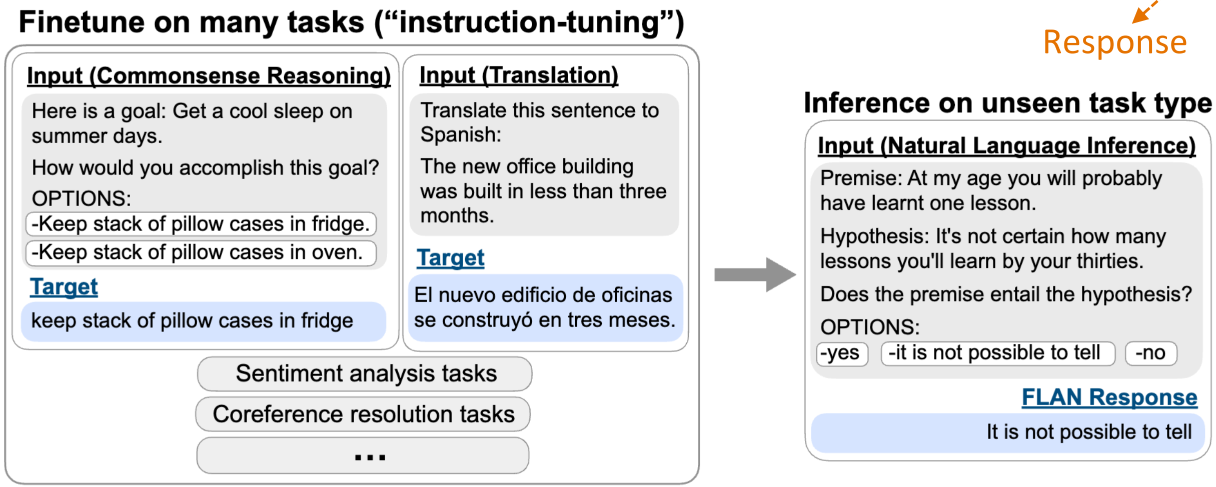
Jason Wei*, Maarten Bosma*, Vincent Y. Zhao*, Kelvin Guu*, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le
Google Research

Paper: https://arxiv.org/pdf/2109.01652

Instruction Tuning: Method

- **Input**: task description
- **Output**: expected response or solution to the task
- Train LLMs to generate response tokens given prompts $\min_{\boldsymbol{\theta}} -\log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})$

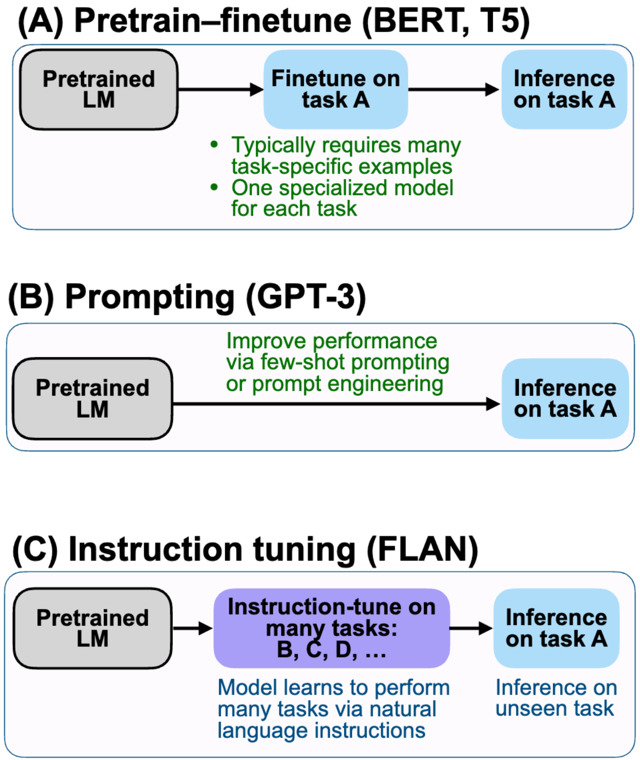
Response Prompt



Paper: https://arxiv.org/pdf/2109.01652

Instruction Tuning vs. Other Paradigms

- Task-specific fine-tuning does not enable generalization across multiple tasks
- In-context learning requires few-shot demonstrations
- Instruction tuning enables zero-shot cross task generalization



Instruction Tuning vs. Pretraining

- Both instruction tuning and pretraining are **multi-task** learning paradigms
- Supervision
  - Pretraining: self-supervised learning (raw data w/o human annotation)
  - Instruction tuning: supervised learning (human annotated responses)
- Task format
  - Pretraining: tasks are implicit (predicting next tokens)
  - Instruction tuning: tasks are explicit (defined using natural language instructions)
- Goal
  - Pretraining: teach LMs a wide range of linguistic patterns & general knowledge
  - Instruction tuning: teach LMs to follow specific instructions and perform a variety of tasks

FLAN: Collection of Instruction Tuning Datasets

62 datasets (12 task clusters) covering a wide range of understanding + generation tasks

Natural language inference (7 datasets)
- ANLI (R1-R3)
- RTE
- CB
- SNLI
- MNLI
- WNLI
- QNLI

Commonsense (4 datasets)
- CoPA
- HellaSwag
- PiQA
- StoryCloze

Sentiment (4 datasets)
- IMDB
- Sent140
- SST-2
- Yelp

Paraphrase (4 datasets)
- MRPC
- QQP
- PAWS
- STS-B

Closed-book QA (3 datasets)
- ARC (easy/chal.)
- NQ
- TQA

Struct to text (4 datasets)
- CommonGen
- DART
- E2ENLG
- WEBNLG

Translation (8 datasets)
- ParaCrawl EN/DE
- ParaCrawl EN/ES
- ParaCrawl EN/FR
- WMT-16 EN/CS
- WMT-16 EN/DE
- WMT-16 EN/FI
- WMT-16 EN/RO
- WMT-16 EN/RU
- WMT-16 EN/TR

Reading comp. (5 datasets)
- BoolQ
- OBQA
- DROP
- SQuAD
- MultiRC

Read. comp. w/ commonsense (2 datasets)
- CosmosQA
- ReCoRD

Coreference (3 datasets)
- DPR
- Winogrande
- WSC273

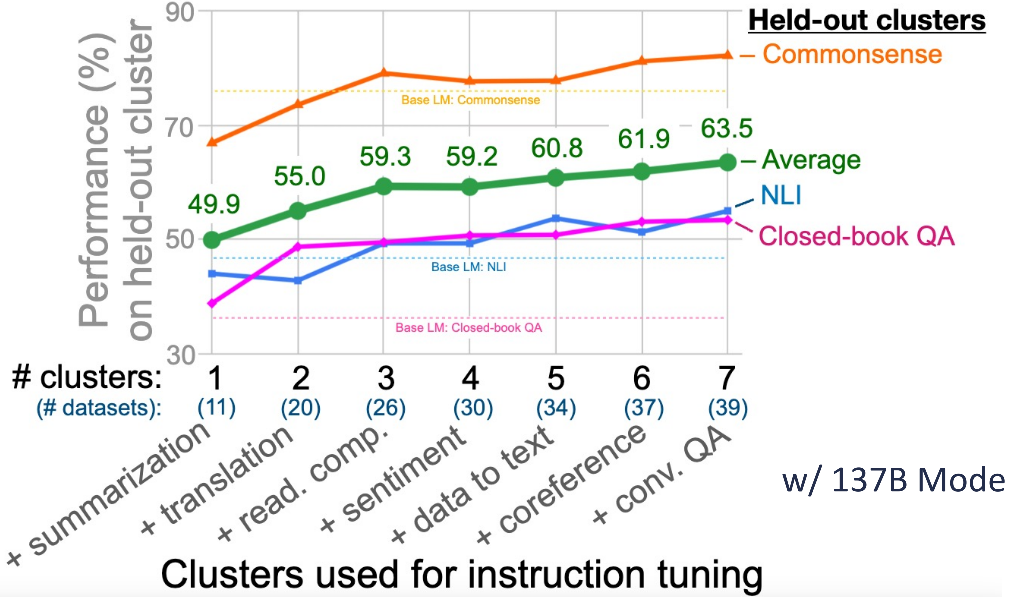
Misc. (7 datasets)
- CoQA
- TREC
- QuAC
- CoLA
- WIC
- Math
- Fix Punctuation (NLG)

Summarization (11 datasets)
- AESLC
- Multi-News
- SamSum
- AG News
- Newsroom
- Wiki Lingua EN
- CNN-DM
- Opin-Abs: iDebate
- XSum
- Gigaword
- Opin-Abs: Movie

Paper: https://arxiv.org/pdf/2109.01652

Generalization Improves with More Clusters

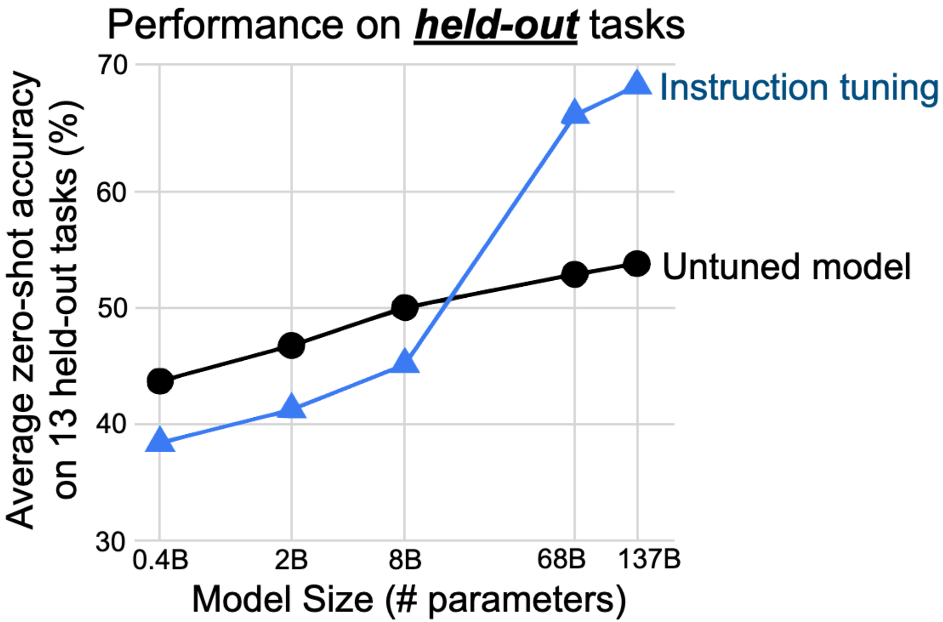
- Held out three clusters from instruction tuning: Commonsense, NLI, Closed-book QA
- More clusters and tasks used in instruction tuning => better generalization to unseen clusters



w/ 137B Model

Instruction Tuning with Different Model Sizes

- Instruction tuning can hurt small model (< 8B) generalization
- Instruction tuning substantially improves generalization for large models



Chat-style Instruction Tuning

- Instruction tuning can also be used to build chatbots for multi-turn dialogue
- Instructions may not correspond strictly to one NLP task, but mimic a human-like dialogue
- Multi-turn instruction tuning training data example:

{"role": "user", "content": "What's the weather like today?"},
{"role": "assistant", "content": "It's sunny with a high of 75 degrees."},
{"role": "user", "content": "Great! What about tomorrow?"},
{"role": "assistant", "content": "Tomorrow will be partly cloudy with a high of 72 degrees."}

UNIVERSITY of VIRGINIA

Join at
slido.com
#3947 182

Further Reading on Instruction Tuning

- Multitask Prompted Training Enables Zero-Shot Task Generalization [Sanh et al., 2021]
- Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks [Wang et al., 2022]
- Self-Instruct: Aligning Language Models with Self-Generated Instructions [Wang et al., 2022]
- LIMA: Less Is More for Alignment [Zhou et al., 2023]

Thank You!

Yu Meng
University of Virginia
yumeng5@virginia.edu