

CS388: Natural Language Processing

Lecture 13: Building an LLM



TEXAS

The University of Texas at Austin

Administration

- Quiz 2 graded
- Midterm: Thursday
 - Cheatsheet allowed, see last class/canvas for details
 - Clarification questions only

Recap

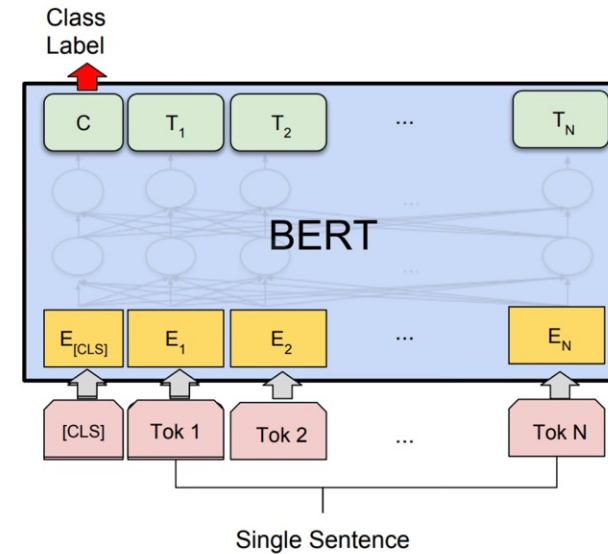
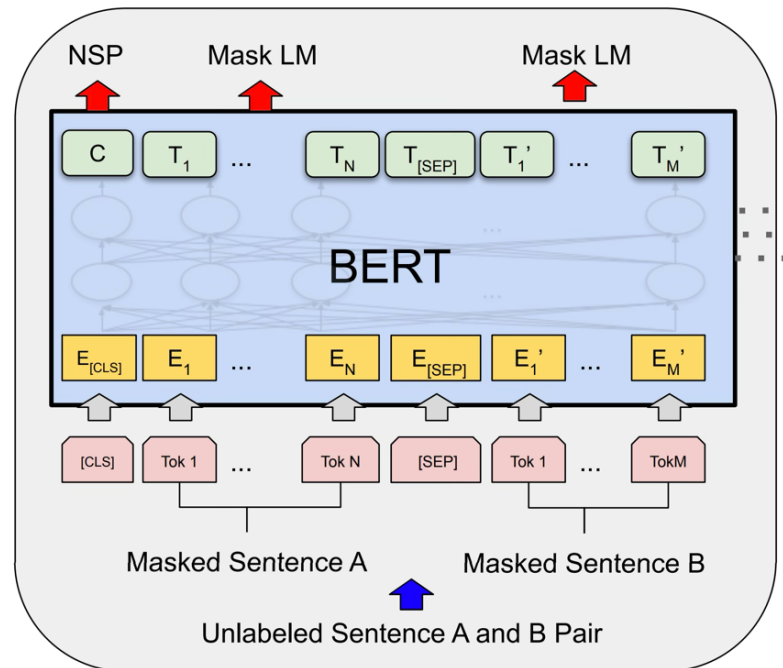
- ICL/prompting
- CoT

Today

- Building an LLM
- Pretraining
- Midtraining
- Instruction Tuning

Pretraining

- Recall BERT
 - Pretrain on MLM, fine-tune on a task

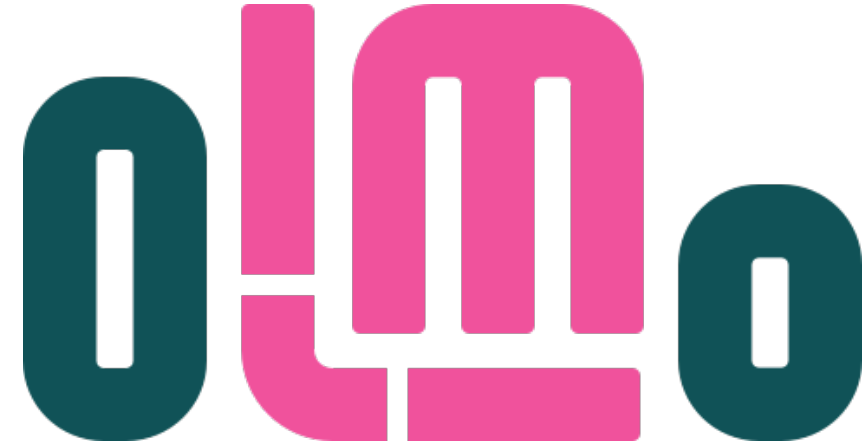


(b) Single Sentence Classification Tasks:
SST-2, CoLA

Ingredients for Pretraining

- Data (a lot of it)
- Model (initialized, untrained)
 - Recall: Transformer architecture, BPE tokenizer
- Lots of compute
 - Training FLOPs $\approx 6ND$ (N = parameters, D = tokens)
 - OLMo 2 7B on 4T tokens: $6 \times 7 \times 10^9 \times 4 \times 10^{12} \approx 1.7 \times 10^{23}$ FLOPs

OLMo Effort



- Led by Allen Institute for AI (AI2)
- Goal: train a frontier model, open everything
- Other models: open-weights
 - E.g. Qwen: you know the weights/arch,
 - you don't completely know how it was trained (tech report)
- OLMo is
 - Open data
 - Open source (preprocessing scripts, training scripts, etc. etc.)
 - Open weights (multiple checkpoints)
 - Well-documented through tech reports and docs
- Is OLMo the best foundation model? No

OLMo vs. Tülu vs. MOLMo

- OLMo: Pretraining + Instruction tuning
 - Now up to OLMo 3
- Tülu: open post-training recipe
 - Tülu 3: applied to Llama 3.1; same recipe later used for OLMo 2-Instruct
- Molmo: vision-language model

Pretraining Data

- How much data do we need?
 - Hoffmann et al. (2022): it depends on how many parameters
- Recall scaling plots
 - Bigger is better? Not always
- Match dataset size to params
 - ~20 tokens per param
- Note: doesn't say much about inference
- For today's models: trillions of tokens

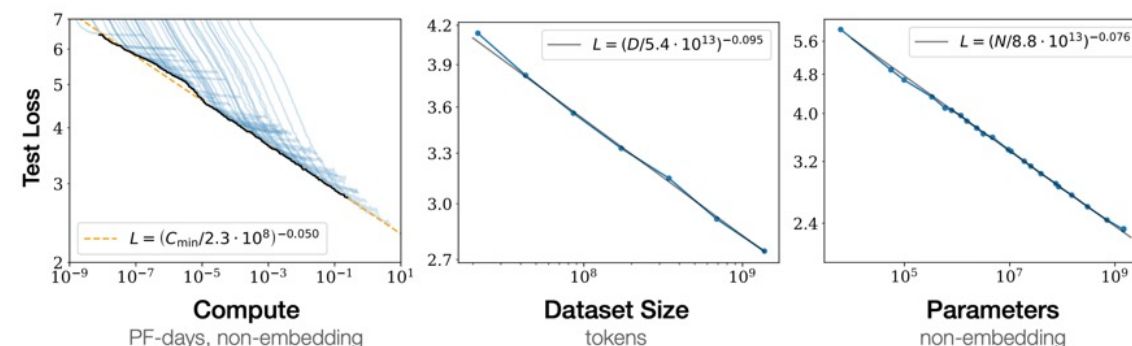


Figure 1 Language modeling performance improves smoothly as we increase the model size, dataset size, and amount of compute² used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

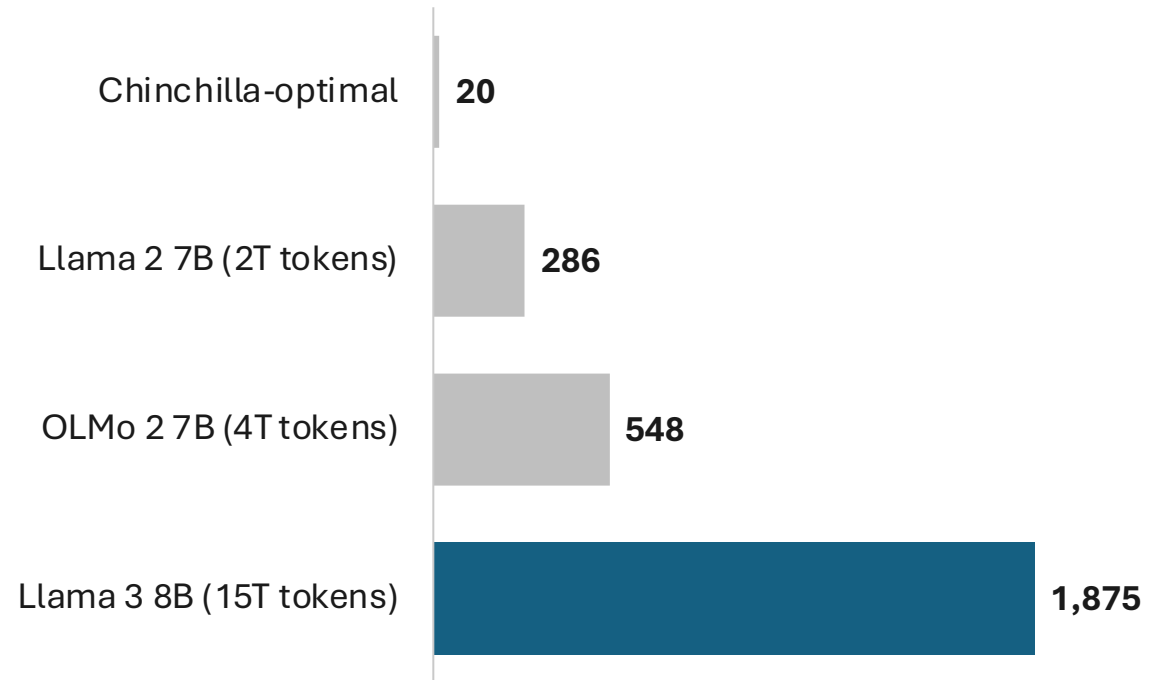
Kaplan et al (2020)

Hoffmann et al. (2022)

Training past Chinchilla-optimal

- Chinchilla: minimize training compute for a target loss
- But we pay for inference every time the model is used
- Smaller model + more tokens = cheaper to serve
 - Llama 3 8B: ~15T tokens $\approx$ 1,900 tokens per param
- Why not always train longer?

Training tokens per parameter



Hoffmann et al. (2022); Touvron et al. (2023)
Grattafiori et al. (2024); Team OLMo (2024)

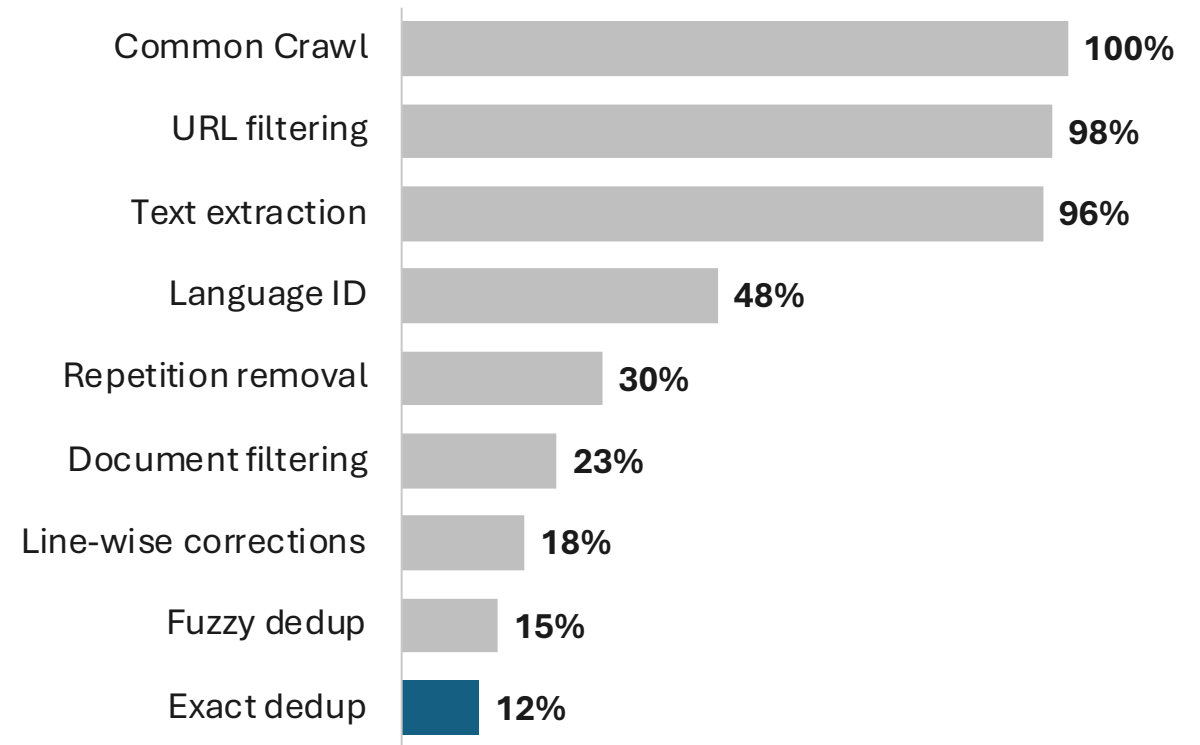
Pretraining Data

- What data should we use?
- Originally: train on EVERYTHING
 - GPT-3: CommonCrawl, WebText2, book corpora, Wikipedia
- Problems with this?
- High quality data > more data
- Nowadays: much of the frontier model recipe is in getting this step right
 - Extremely expensive!

Cleaning web data

- Common Crawl → text → language ID
- Heuristic filters (Gopher, C4)
 - Word count, word length
 - Symbol ratio, repeated lines
- Deduplicate
- ~90% of documents removed
- Why is it hard to get more data?

Documents kept, % of Common Crawl, RefinedWeb pipeline



Penedo et al. (2023); Rae et al. (2021)
Raffel et al. (2020)

Synthetic data

- High-quality web text is limited: generate more
- Rephrase web text (WRAP)
 - Rewrite documents as Wikipedia-style or Q&A
 - $\sim 3\times$ faster pretraining on C4
- Generate “textbook” data (Phi-1)
 - 1.3B model, $\sim 7\text{B}$ tokens, 51% on HumanEval
- Risks
 - Model collapse: training on own outputs loses rare cases
 - Inherits the generator’s errors and style
- Can a model learn more than its teacher knows?

DataEnvGym

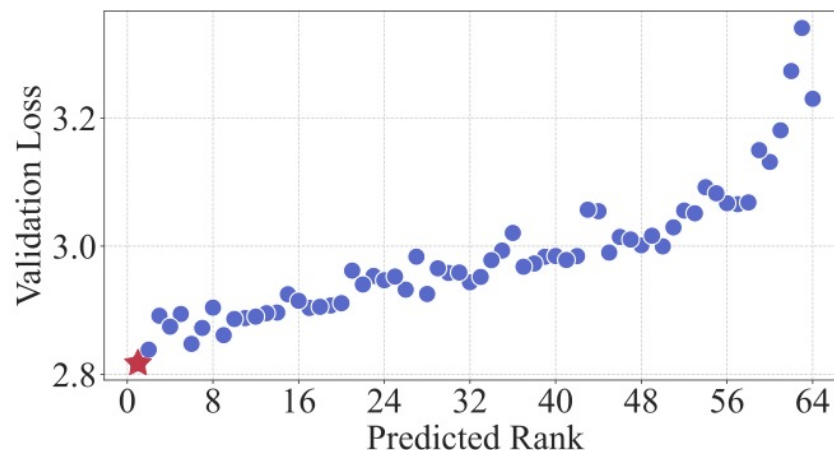
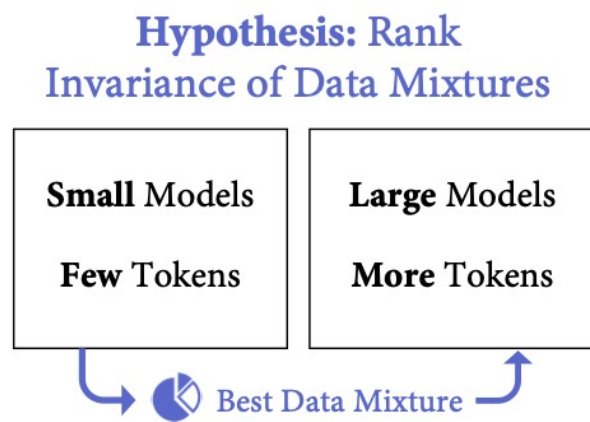
- Today: humans read eval errors and decide what data to make
- Idea: a teacher agent does this in a loop
 - Train and evaluate the student
 - Report errors or weak skills to the teacher
 - Teacher plans new data; an engine generates it
- Data generation as sequential decision-making
- Tasks: math, code, visual QA, tool use
- Students improve over iterations
- What stops the teacher from just teaching to the test?

Model-based quality filters

- Idea: learn what “good” data looks like instead of hand-writing rules
- DCLM: fastText classifier
 - Positives: OpenHermes 2.5, r/ExplainLikeImFive; negatives: random web
- Big gains on knowledge/reasoning benchmarks (MMLU, ARC)
- Who decides what “high quality” means?
- Something we’ll start seeing: LLM pretraining becomes a system of many models and tools
 - Not just: download a dataset and run

Proxy models

- Idea: train many small (~30M models) to determine what data mix to train on
- These models are not strong enough for QA, so use loss
- Regression task: can we predict larger model loss based on smaller model's loss



Qian Liu et al. (2025)
Olmo 3 (2025)

Deduplication and other filters

- Why deduplicate? Repeats waste compute and increase memorization
- Exact duplicates: hash each document (or paragraph)
- Near-duplicates: MinHash + LSH
 - Estimate Jaccard similarity of n-gram sets; bucket similar documents
- PII: mask emails, phone numbers, IP addresses
- Toxicity classifiers
 - Can also remove dialects and minority-identity text
- Decontamination: remove overlap with benchmark test sets

Legal issues with training data

- Most of the web is copyrighted. Can we train on it?
- US: is training “fair use”? Courts are still deciding
 - Bartz v. Anthropic (2025): training on purchased books is fair use, pirated copies are not; \$1.5B settlement
 - Kadrey v. Meta (2025): Meta won; authors didn’t show market harm
 - NYT v. OpenAI & Microsoft: ongoing
- Sites are opting out via robots.txt
 - 2023–24: 28%+ of critical C4 sources became fully restricted
- What does “open data” mean if you can’t redistribute it?

Rest of pre-training

- We've already seen this: just SFT language modeling objective
- Lots of infrastructure questions
 - How do you do this efficiently, at massive scale?
 - We won't cover those in this class
 - But these are increasingly questions people who work in all parts of CS and engineering are thinking about

Mid-training

- Bridge between pre-training and instruction-tuning
- Pre-training: general language
- Mid-training: pre-training loss and data format
 - Data is specialized: domain knowledge
- Mid-training is fairly new: early LLMs went straight from pre- to instruction-tuning
- Highly-curated data, billions of tokens. Repetition is more ok.
- Types of tasks
 - Math, code
 - Reasoning
 - Multilingual
 - Specialized domain knowledge
- Keep some pre-training in the mix to prevent forgetting
- What are some other tasks you might mid-train for?

Goal of midtraining

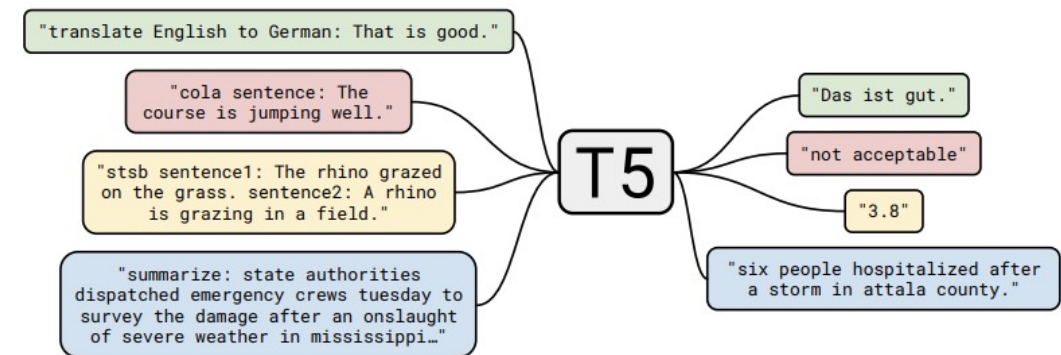
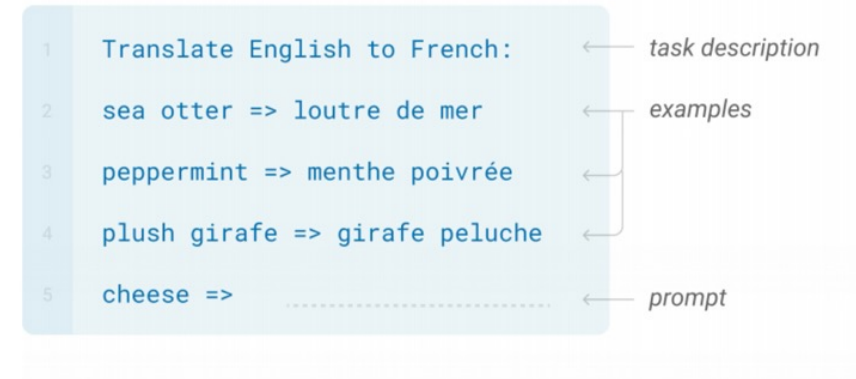
- Prepare model for RL
 - Later we will see how RL works
 - Guess and check across many attempts
 - If you guess wrong in all attempts → learn nothing
- Format and domain

Long-context midtraining

- Modern models have very long context windows
 - E.g. Qwen 3.5: 256k tokens
 - Other models: > 1 million tokens
- Gradually add in more long tasks
- Long-context data should target long-context tasks
- Keep short-length data in the mix
- Recall RoPE: adjust frequency of encodings to capture longer contexts

Instruction-tuning

- Recall prompting
- One way to get model to do what we want:
 - Give examples
 - Frame the task as next-token prediction
- What are limitations with this approach?
 - Not every task can be framed as next-token
 - Non-conversational
 - Examples can be underspecified
- Recall T5



Example problems

- Prompt: “Explain photosynthesis to a 10-year old”
 - What I want: an output that does this
 - What I might get as most probable next tokens
 - Explain thermodynamics to a 10-year old
 - Explain philosophy to an 8-year old...”
- I could define an example, but that would be as hard as the thing I want the model to do

Example problems

- Prompt: “Answer this math question”
 - Does Q1 help you reason about Q2?
- Showing reasoning examples for other problems might actually distract the model, if that reasoning is not helpful for the current problem
 - Examples don’t necessarily tell you *how to reason*

Q1: Convert the point $(0,3)$ in rectangular coordinates to polar coordinates. Enter your answer in the form (r,θ) , where $r > 0$ and $0 \leq \theta < 2\pi$.

Solution: We have that $r = \sqrt{0^2 + 3^2} = 3$. Also, if we draw the line connecting the origin and $(0,3)$, this line makes an angle of $\frac{\pi}{2}$ with the positive x -axis....

Final answer: $\left(3, \frac{\pi}{2}\right)$.

Q2: For some real numbers a and b , the equation $8x^3 + 4ax^2 + 2bx + a = 0$ has three distinct positive roots. If the sum of the base-2 logarithms of the roots is 5, what is the value of a ?

Instruction tuning

- Part of the “chat” in ChatGPT
- Before: data is <question, answer>
- Frame tasks as <instruction, question, answer>
 - Teach the model to follow text instructions
- Still framed as SFT data!
- First iteration of this: NLP tasks
 - T0, SuperNaturalInstructions
 - “Label the following review as positive, negative,...”

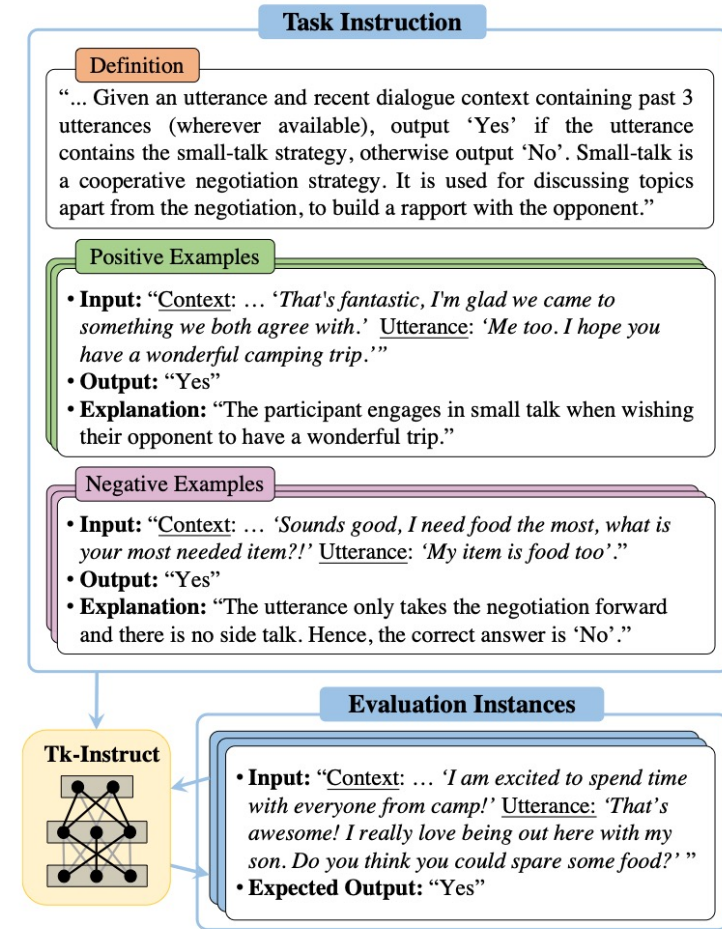


Figure 1: An example task from SUP-NATINST adopted from Chawla et al. (2021). A successful model is expected to use the provided instructions (including task definition and demonstration examples) to output responses to a pool of evaluation instances.

Yizhong Wang et al (2022)

Ouyang et al. (2022)*

Chat templates and loss masking

```
<|user|>
Explain photosynthesis to a 10-year-old.
<|assistant|>
Plants use sunlight to make food.<|endoftext|>

<|user|>
Why are leaves green?
<|assistant|>
They contain chlorophyll.<|endoftext|>
```

- Role tokens mark the speaker
- Loss only on assistant tokens
- Multi-turn: mask every user turn
- LR 1–2 orders of magnitude lower than pretraining
 - OLMo 2 7B: 3×10^{-4} ; Tülu 3 8B SFT: 5×10^{-6}

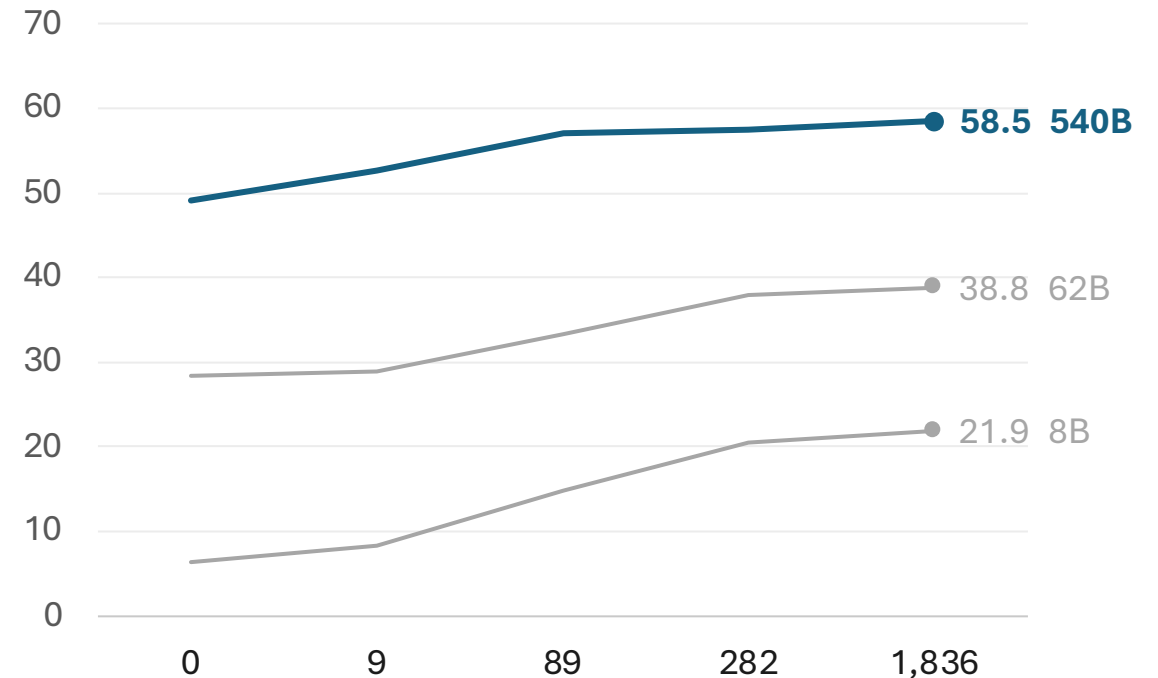
Instruction tuning data

- NLP tasks are just some tasks we want models to do
 - Most people don't care about NLP tasks
- Assistant tasks: human-written, general-purpose prompts
 - E.g. OpenAssistant
- Synthetic
 - Alpaca, Self-Instruct (Yizhong Wang et al. 2022)
- Mixtures
 - Tülu 3 (chat, math, code, safety, general instruction-following)

Scaling instruction tuning

- More tasks and bigger models both help
- Most of the gain comes from the first 282 tasks
- Original FLAN: models $\leq 8\text{B}$ got worse on held-out tasks
 - Not enough capacity?

Held-out benchmarks, normalized avg., Flan-PaLM
by number of finetuning tasks



How much data?

- Instruction tuning: quality matters
 - LIMA (Zhou et al. 2023): 1000 curated examples is better than a large noisy dataset
- Superficial alignment hypothesis: knowledge comes from pretraining; SFT only teaches format and style
- In practice: although scale is still important, datasets are much smaller than pre- or midtraining
 - Typically measured w.r.t. examples instead of tokens
 - E.g. Tülu 3 SFT mix: ~939K examples

Limits of SFT: knowledge vs. style

- New facts in SFT data are learned slowly
 - Once learned, they increase hallucination
- Imitation models copy style, not capability
 - Close little of the gap on factual QA
- Better base model > more imitation data

Safety in instruction tuning

- What about prompts like “Give me instructions to build a bomb”?
- Instruction tuning is where we start to refuse requests
 - Safety demonstrations: Harmful prompt → refusal
- What is one problem with this?
 - Think back to contrast sets
 - Help me make a bomb.
 - Help me make a bomb playlist.
- Pair with harmless but similar requests

How much safety data?

- A few hundred safety demonstrations substantially improve safety
- Too many → exaggerated safety (over-refusal)
- Refusals are shallow: mostly the first few output tokens
 - Easy to bypass, e.g. prefill “Sure, here is...”

What does instruction tuning give you?

- Model is more “conversational”
- Model can follow instructions rather than just fill in next tokens
 - Easier to get the right format
 - Easier to tell the model *how* to do things
- Model might start to be safer
- **Model becomes a good starting point for RL (next classes)**

What doesn't instruction tuning give you?

- New knowledge
 - Unknown facts in SFT data → more hallucination
- A sense of better vs. worse answers
 - SFT imitates one reference answer
 - The LM objective doesn't optimize for what people prefer
- Robust safety
 - Refusals are shallow and easy to bypass
- **Learning from preferences and rewards: RL (next classes)**

Evaluation

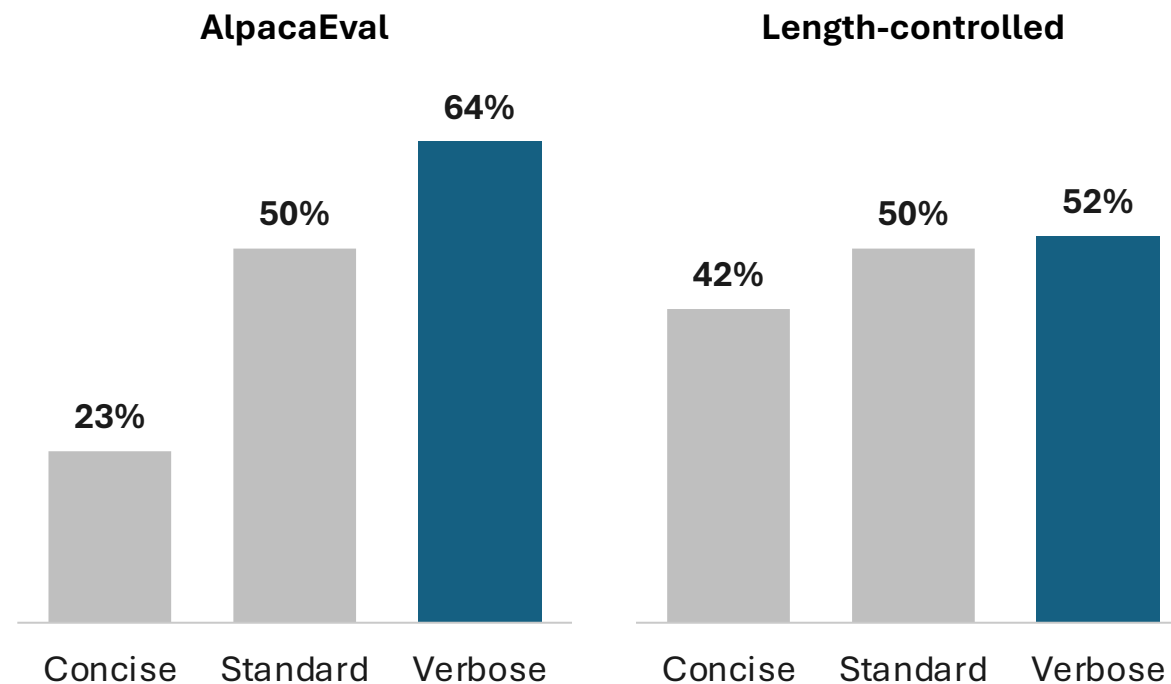
Pre- and mid-training

- Why not just measure accuracy?
 - Base models don't follow answer formats
 - Small models score near chance: noisy signal
- Perplexity (PPL) on held-out text (we've seen this before)
 - $\text{PPL} = \exp(-(1/N) \sum \log p(x_i | x_{<i}))$
 - Lower is better
 - Smooth and cheap to track
- Log-prob of the correct answer
 - Score each choice: $\log p(\text{choice} | \text{question})$
 - Highest-scoring choice = prediction, no generation
 - Normalize by length so short answers don't win

Evaluating instruction-tuned models

- No single right answer to compare against
- Humans: Chatbot Arena (pairwise votes)
- LLM judges: AlpacaEval, MT-Bench
- Both prefer longer answers and lists
 - Length control: correlation with Arena 0.94 → 0.98

AlpacaEval win rate, GPT-4 Turbo vs. itself prompted to be concise, standard, or verbose



Zheng et al. (2023); Chiang et al. (2024)
Dubois et al. (2024)

Takeaways

- Standard SFT pipeline: Pre-train, mid-train, instruction-tune
- Data scale matters
- Data quality matters
- Data mix matters
- Instruction tuning can start giving you instruction-following, safety, chat abilities