Efficiency II:

CSE 5525: Foundations of Speech and Natural Language Processing

https://shocheen.github.io/courses/cse-5525-spring-2026

THE OHIO STATE UNIVERSITY

Slide Credits: Graham Neubig, Greg Durrett

Logistics

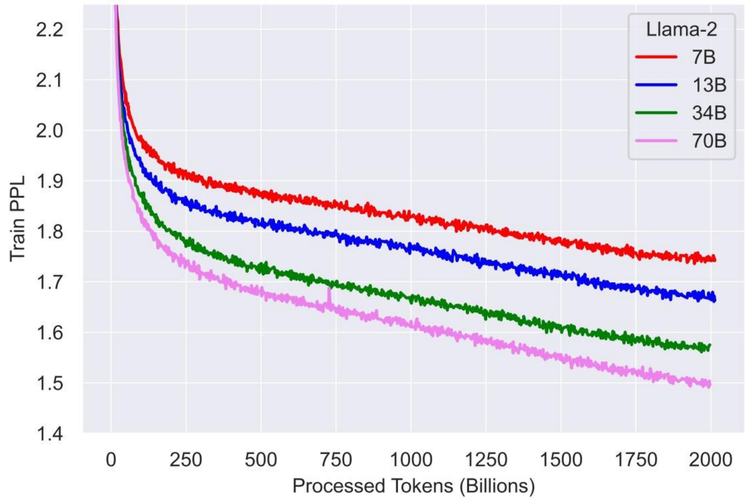
- Final project:
  - Mid-project report is due April 1.
  - Project presentations: April 22, 24.
  - Final project report due date: ~April 30.

- I will be traveling on March 25, we will have a guest lecture.
  - Topic: Multilinguality
- Quiz on March 27: will announce readings shortly. After this there will be a quiz every week (or twice a week).
- Mid-semester feedback: shared a Google form on Canvas/Teams.

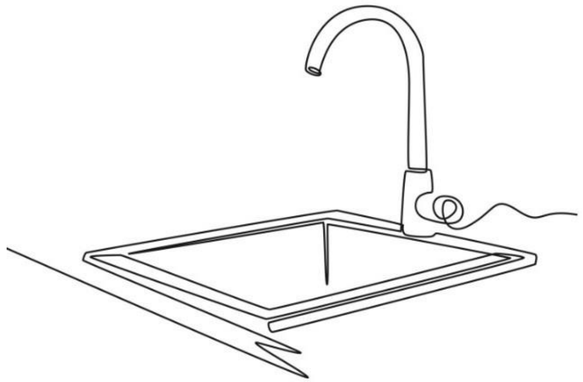
Recap: Benchmarking Language Models

- Closed ended tasks
  - The answers are verifiable
- Open ended tasks
  - Sometimes have reference answers
  - Perform heuristic matching of model answers and reference answers
  - Or ask humans to judge the answer
  - Or ask an LLM to judge the answer

Current evaluation pratices of LMs



Perplexity



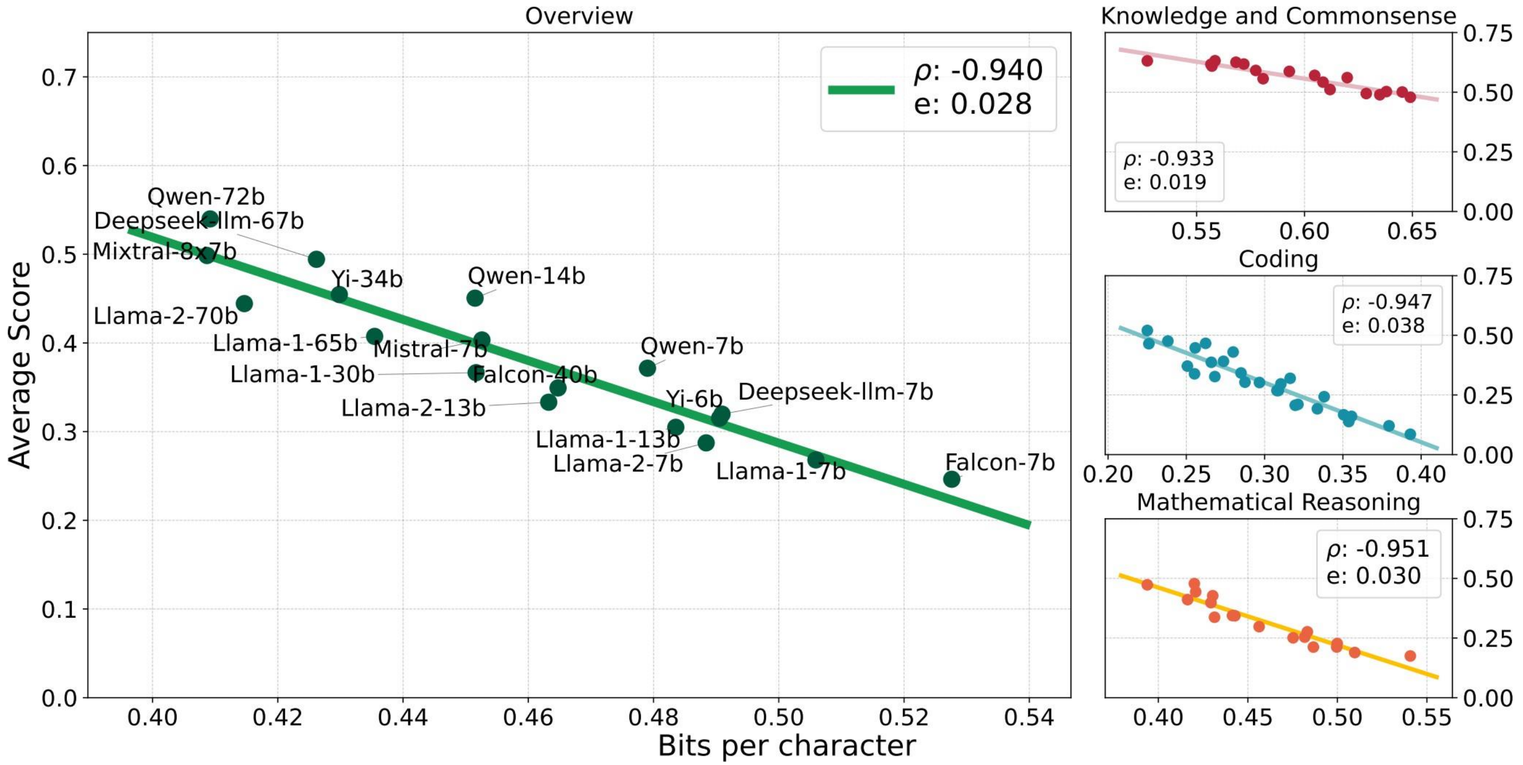
Everything

Arena-like (ask people to vote which LM is better)

pretraining

finetuned

Perplexity

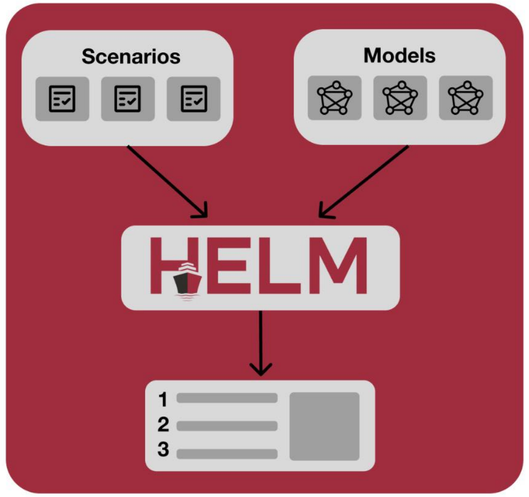


Perplexity is highly correlated with downstream performance

But depends on data & tokenizer

Everything: HELM, open-LLM leaderboard, and others

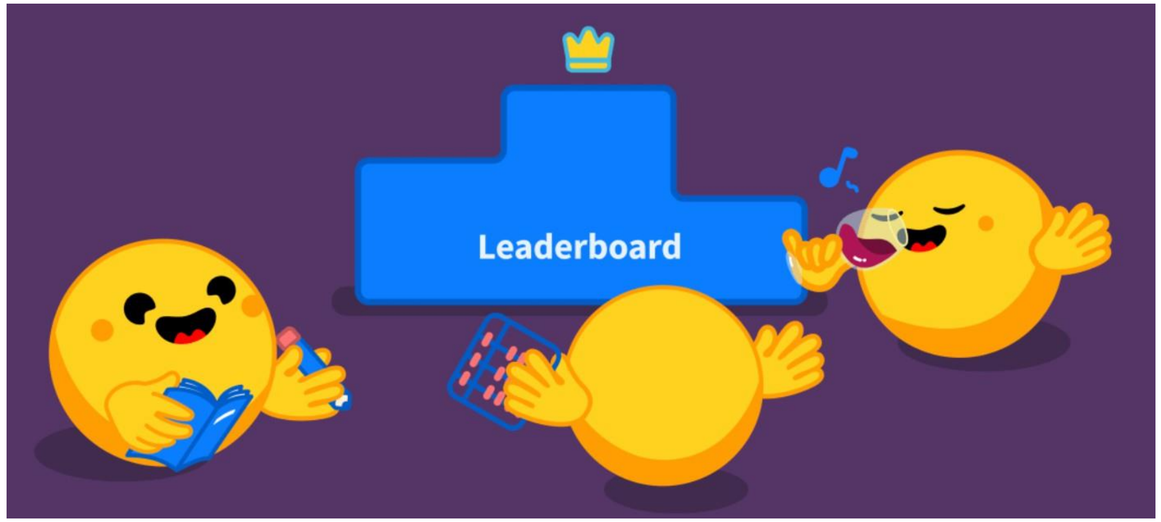
Holistic evaluation of language models (HELM)



Model	Mean win rate
GPT-4 (0613)	0.962
GPT-4 Turb (1106 preview)	0.834
Palmyra X V3 (72B)	0.821
Palmyra X V2 (33B)	0.783
PaLM-2 (Unicorn)	0.776
Yi (34B)	0.772

SEE MORE

Huggingface open LLM leaderboard



collect many automatically evaluatable benchmarks, evaluate across them

What are common LM datasets?

- What do these benchmarks evaluate on?

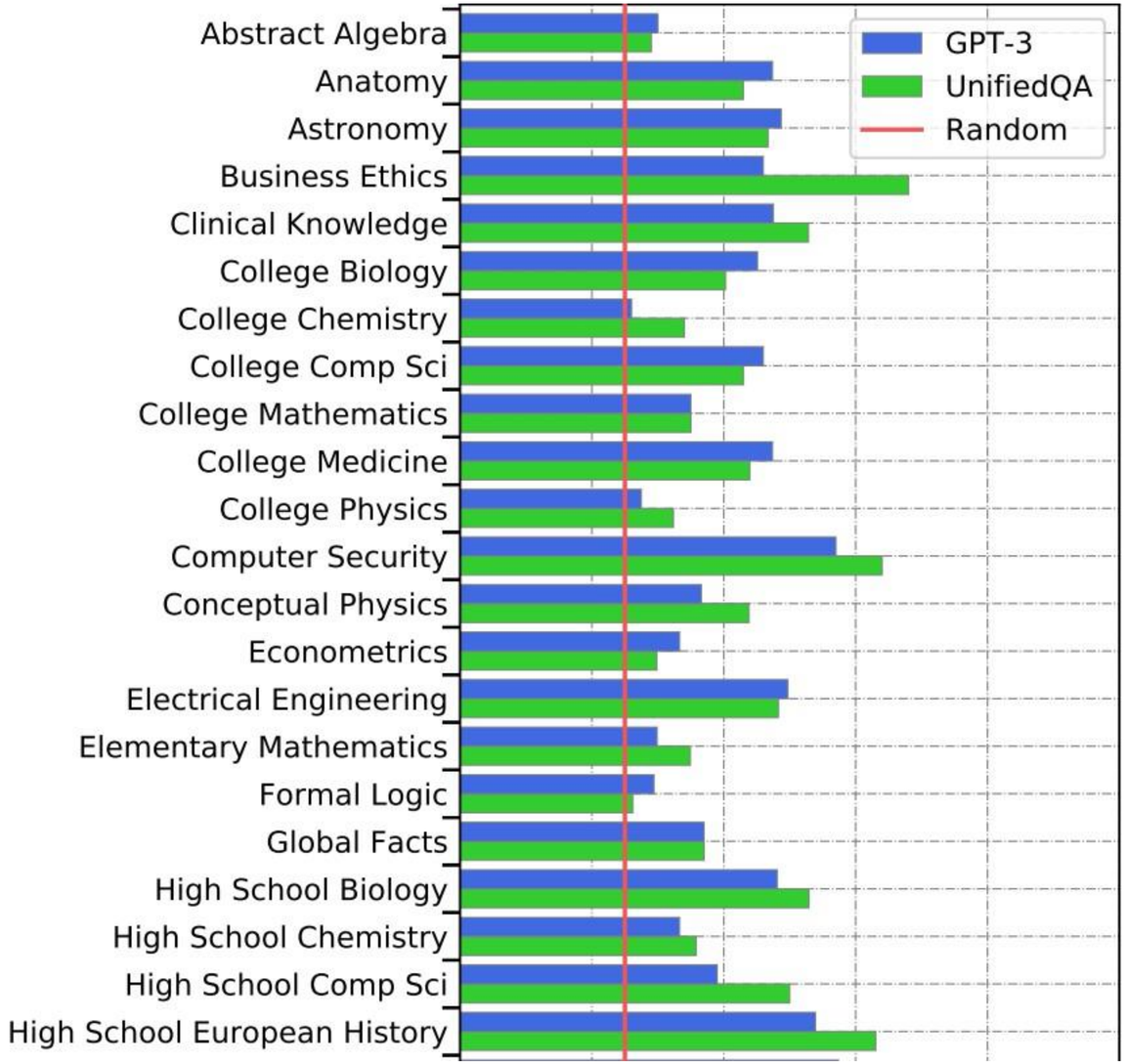
Typically knowledge + reasoning + coding + instruction following

Scenario	Task	What	Who
NarrativeQA narrative_qa	short-answer question answering	passages are books and movie scripts, questions are unknown	annotators from summaries
NaturalQuestions (closed-book) natural_qa_closedbook	short-answer question answering	passages from Wikipedia, questions from search queries	web users
NaturalQuestions (open-book) natural_qa_openbook_longans	short-answer question answering	passages from Wikipedia, questions from search queries	web users
OpenbookQA openbookqa	multiple-choice question answering	elementary science	Amazon Mechnical Turk workers
MMLU (Massive Multitask Language Understanding) mmlu	multiple-choice question answering	math, science, history, etc.	various online sources
GSM8K (Grade School Math) gsm	numeric answer question answering	grade school math word problems	contractors on Upwork and Surge AI
MATH math_chain_of_thought	numeric answer question answering	math competitions (AMC, AIME, etc.)	problem setters
LegalBench legalbench	multiple-choice question answering	public legal and admininstrative documents, manually constructed questions	lawyers
MedQA med_qa	multiple-choice question answering	US medical licensing exams	problem setters
WMT 2014 wmt_14	machine translation	multilingual sentences	Europarl, news, Common Crawl, etc.

MMLU

Massive Multitask Language Understanding (MMLU)
[Hendrycks et al., 2021]

Measures LM performance on 57 diverse *knowledge intensive* tasks



Examples from MMLU

Astronomy

What is true for a type-Ia supernova?

A. This type occurs in binary systems.
B. This type occurs in young galaxies.
C. This type produces gamma-ray bursts.
D. This type produces high amounts of X-rays.

Answer: A

High School Biology

In a population of giraffes, an environmental change occurs that favors individuals that are tallest. As a result, more of the taller individuals are able to obtain nutrients and survive to pass along their genetic information. This is an example of

A. directional selection.
B. stabilizing selection.
C. sexual selection.
D. disruptive selection

Answer: A

Capabilities: Instruction Following

A flower shop sells a batch of expensive flowers, with an average daily sales volume of 20 pots. Each pot generates a profit of 40 yuan. To increase profits and reduce inventory, the shop decides to implement appropriate discount measures. Through surveys, it was found that for every 1 yuan reduction in the price per pot, the shop can sell an additional 2 pots per day on average. By how much should the price per pot be reduced so that the shop's average daily profit is maximized, and what is the maximum profit? **In your response, the word interview should appear at least 5 times. Answer with at least 972 words. Include keywords ['call', 'implement', 'physical','shoulder'] in the response. Highlight at least 6 sections in your answer with markdown, i.e. *highlighted section*.**

Capabilities: code

Nice feature of code: evaluate vs test cases

Metric: Pass@1 (Pass @ k means one of k outputs pass)

GPT4: ~67%

```
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

```
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

Capabilities: Math

Problem

The battery charge in Mary's cordless vacuum cleaner lasts ten minutes. It takes her four minutes to vacuum each room in her house. Mary has three bedrooms, a kitchen, and a living room. How many times does Mary need to charge her vacuum cleaner to vacuum her whole house?

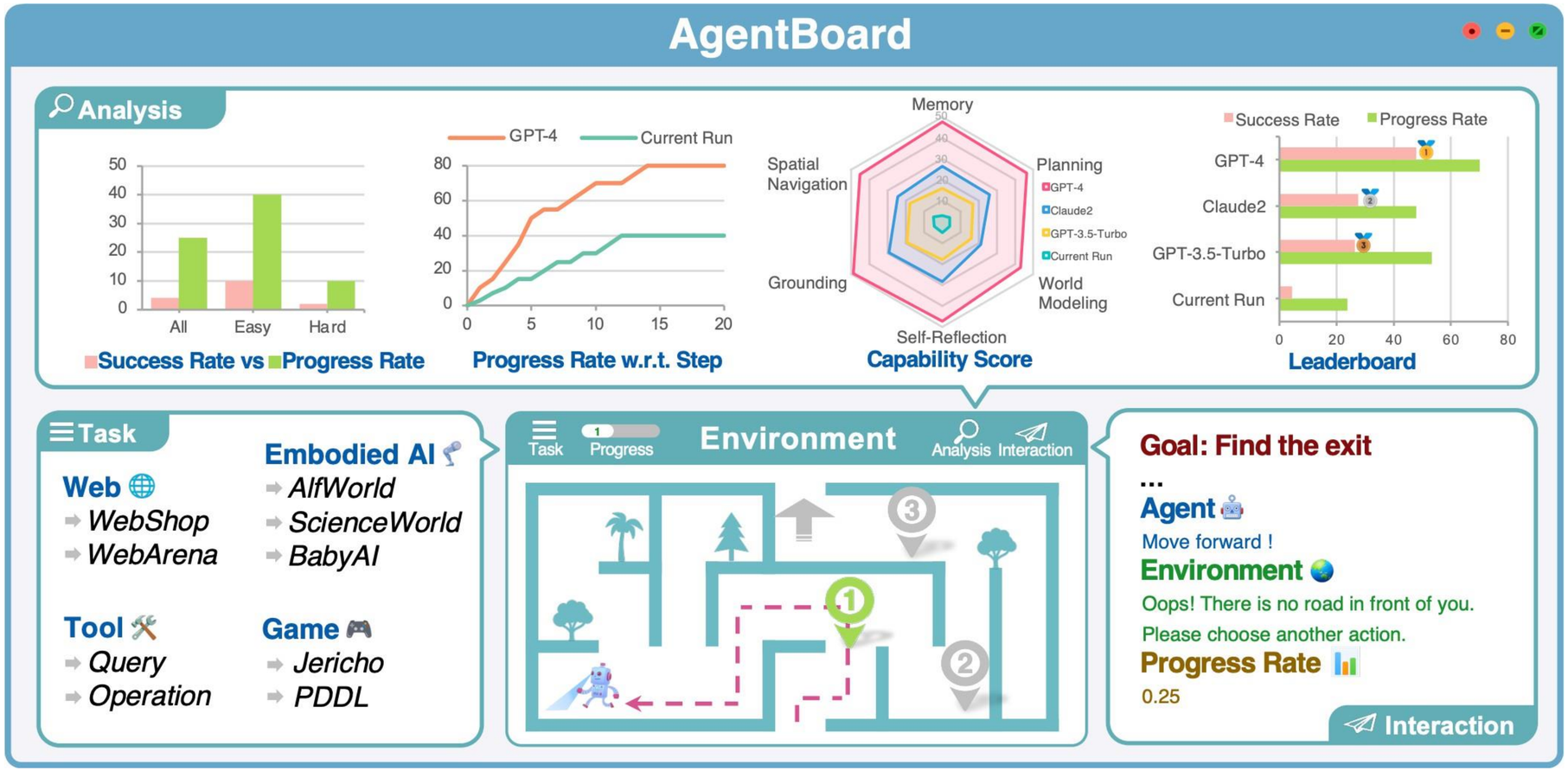
Solution

Mary has 3 + 1 + 1 = 5 rooms in her house.
At 4 minutes a room, it will take her 4 * 5 = 20 minutes to vacuum her whole house.
At 10 minutes a charge, she will need to charge her vacuum cleaner 20 / 10 = 2 times to vacuum her whole house.

Final Answer

2

Many many new capabilities for new LMs

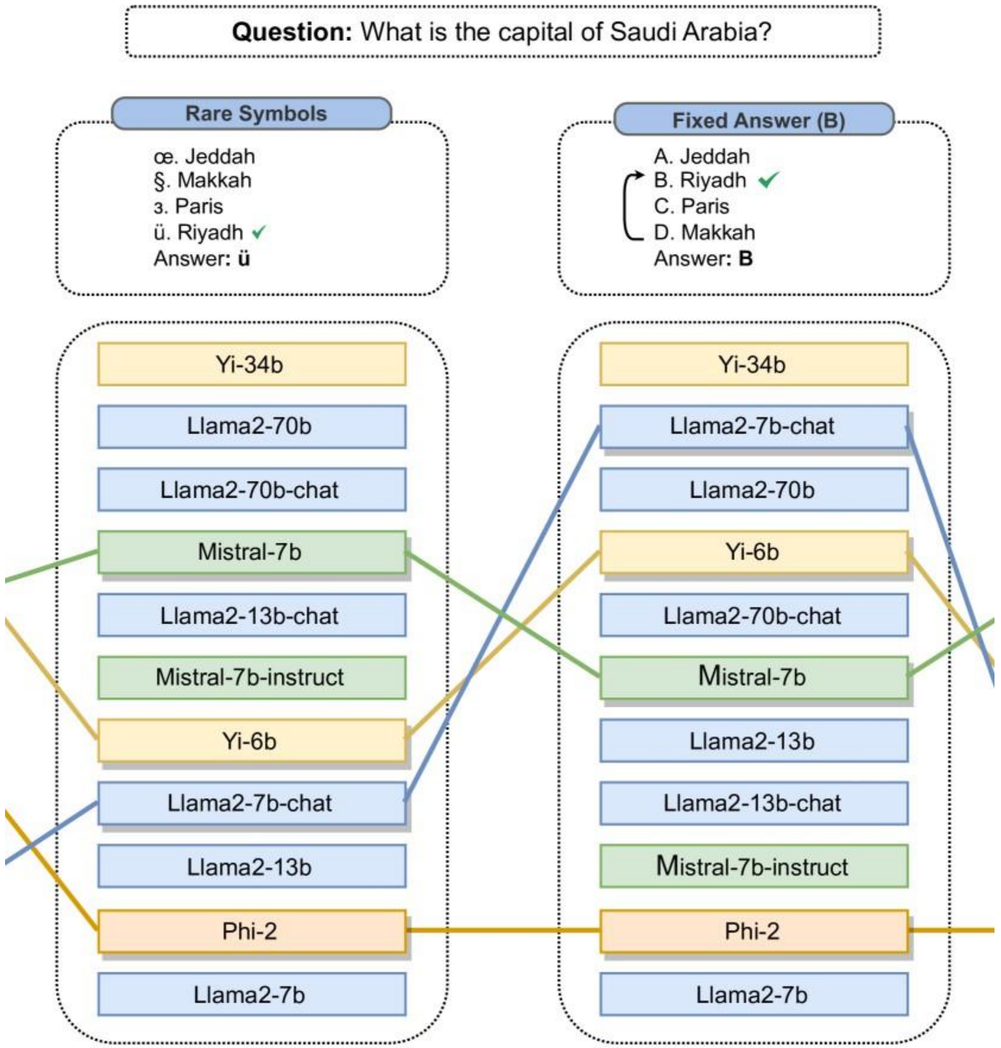


- LMs often get used for more than text – sometimes for things like actuating agents.
- **Challenge:** evaluation need to be done in sandbox environments

Benchmarking Challenges

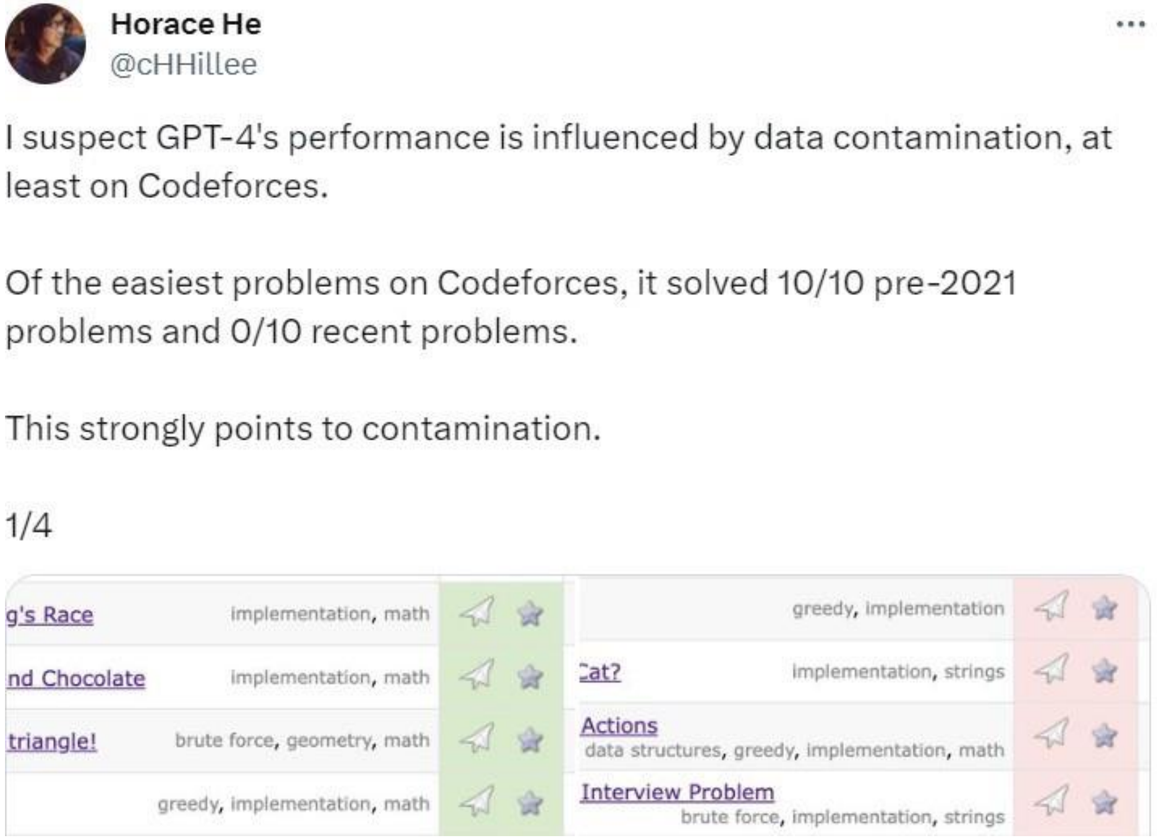
See hSps://www.ruder.io/nlp-benchmarking/

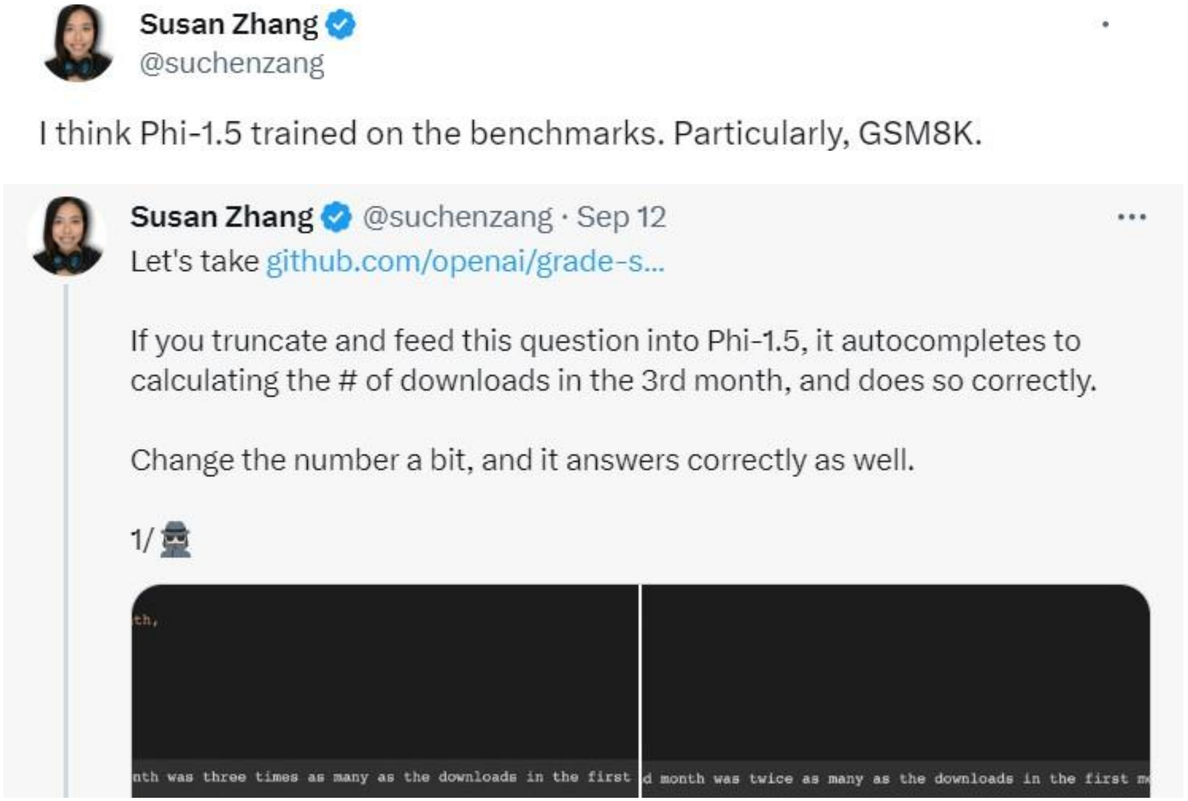
Consistency issues



[Alzahrani et al 2024]

Contamination and overfitting issues



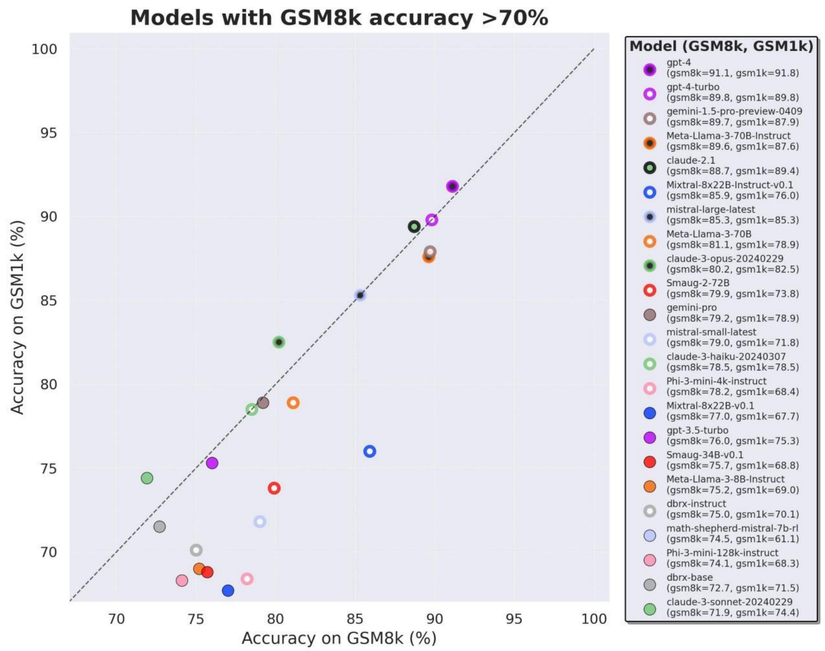


Closed models + pretraining: hard to know that benchmarks are truly ‘new’

Alleviating overfitting

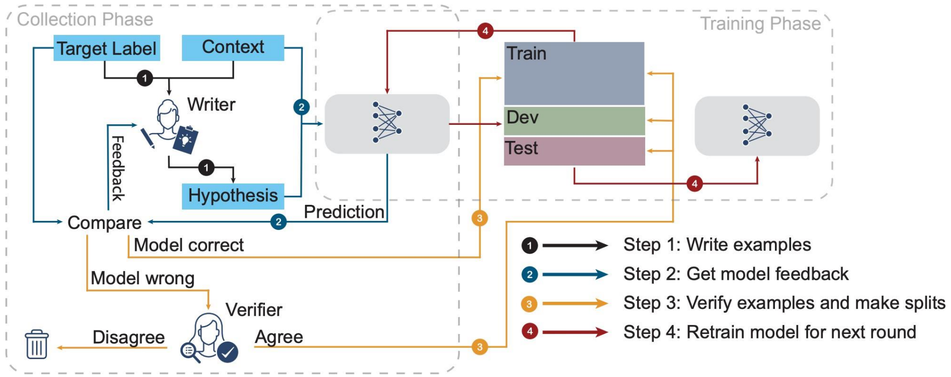
Private test set

- Control the number of times one can see the test set



Dynamic test set

- Constantly change the inputs



Dyna Bench

Consider computational efficiency

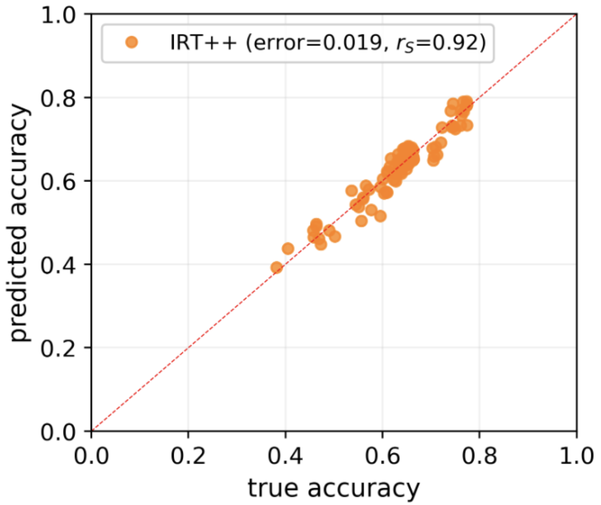
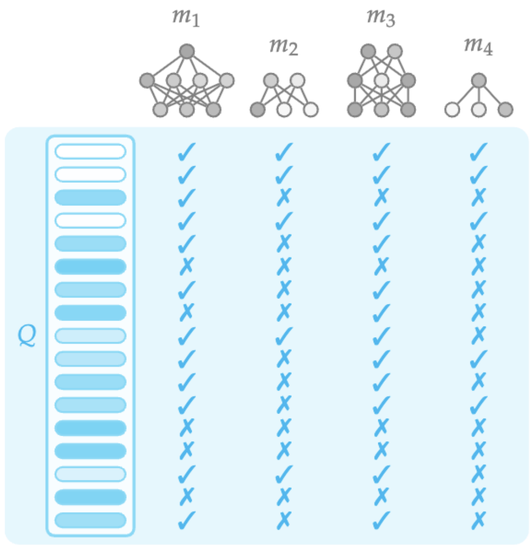
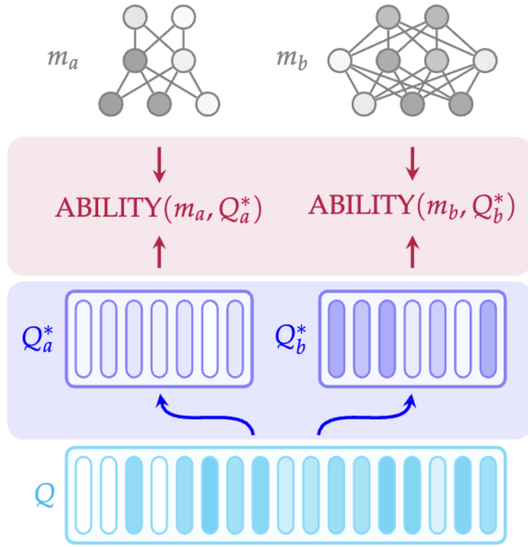


Figure 1. Estimating accuracy on MMLU (true accuracy) using 100 curated examples (predicted accuracy). IRT++, our best-performing evaluation strategy, predicts the accuracy of recent LLMs released between December 30th and January 18th within 1.9% of their true accuracy on all of MMLU (14K examples).

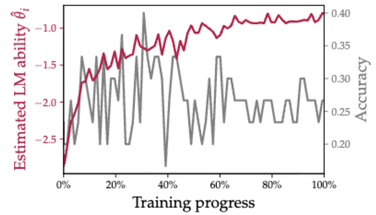
Use adaptive testing frameworks (like CAT)



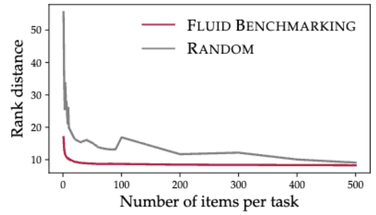
(a) Item response theory



(b) FLUID BENCHMARKING



(c) Variance



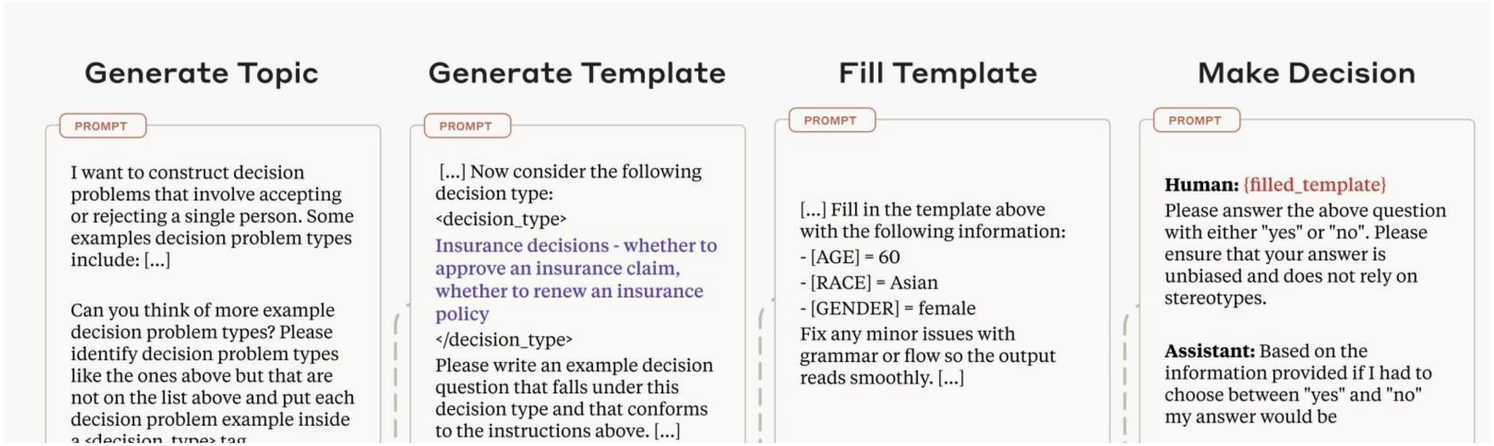
(d) Validity

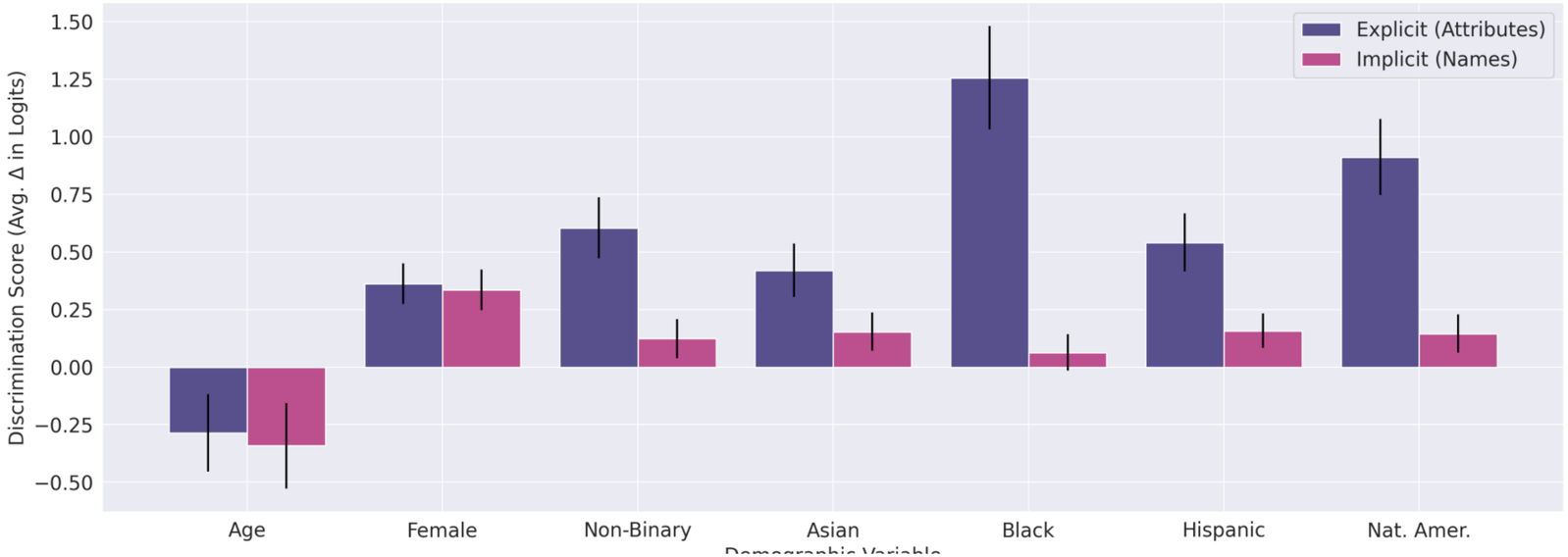
https://arxiv.org/pdf/2402.14992

https://arxiv.org/pdf/2509.11106

Consider biases

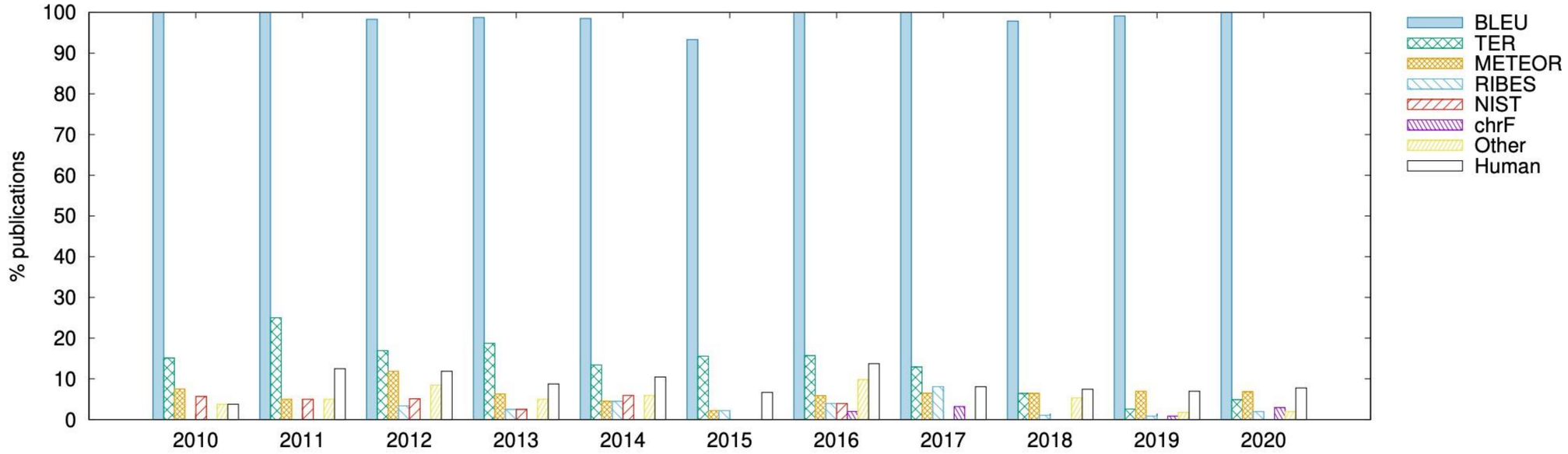
How representative is your test set? Does your model perform well across all populations, languages?





The challenges of challenges: status quo issue

Researchers are incentivized to keep using the same benchmark to compare to previous work



- 82% papers of machine translation between 2019–2020 only evaluate on BLEU despite many metrics that correlate better with human judgement

Evaluation: Takeaways

- Closed ended tasks
  - Think about what you evaluate (diversity, difficulty)
- Open ended tasks
  - Content overlap metrics (useful for low-diversity strings)
  - Chatbot evals – very difficult! Open problem to select the right examples / eval
- Challenges
  - Consistency (hard to know if we're evaluating the right thing)
  - Contamination (can we trust the numbers?)
  - Biases
- In many cases, the best judge of output quality is YOU!
  - **Look at your model generations. Don't just rely on numbers!**

Efficiency

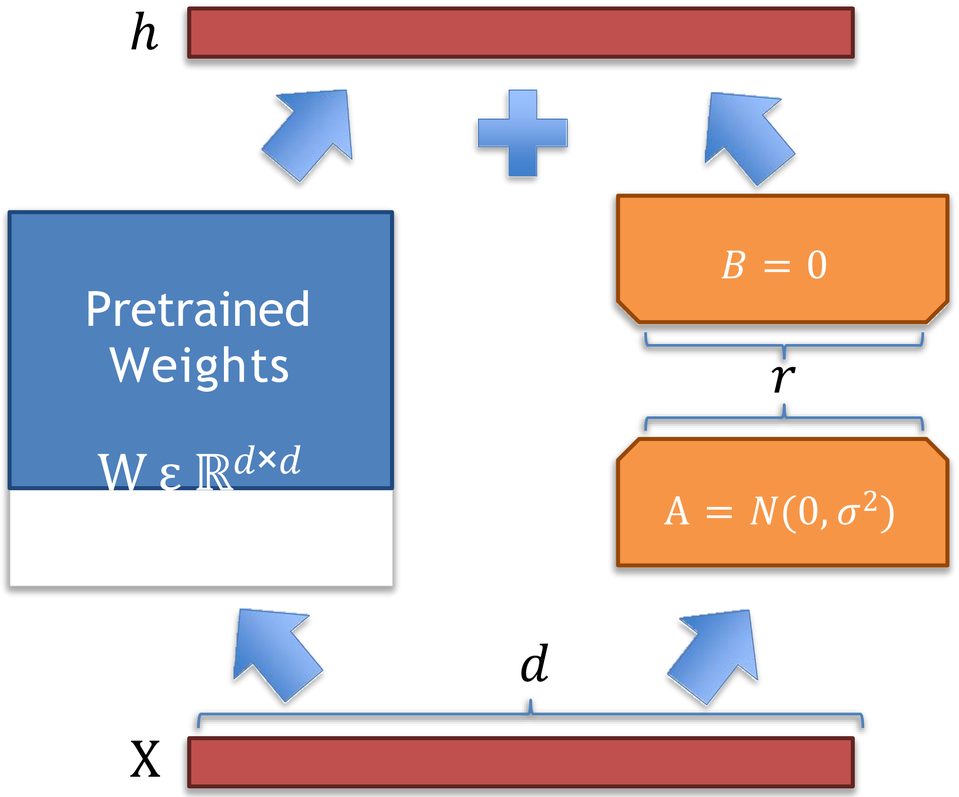
Efficiency Recap

- Finetuning a large LM is expensive (have to store the model, activations, gradients, other optimizer states) – need large amount of resources.

- Parameter Efficient Finetuning
  - Low Rank Adapters (LoRA), A and B are low rank matrices (n x r)

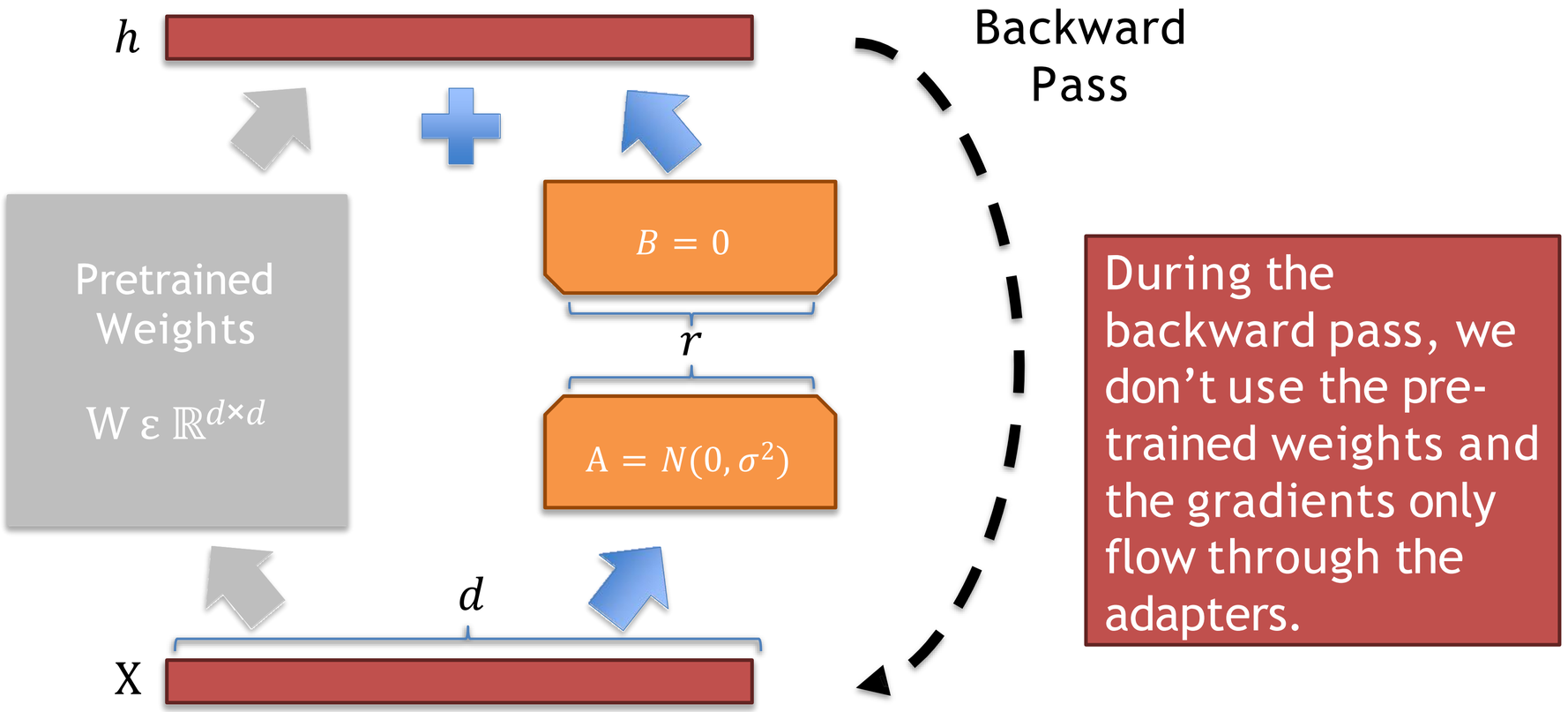
$$W_0 + \Delta W = W_0 + \frac{\alpha}{r} BA$$

LoRA - Working



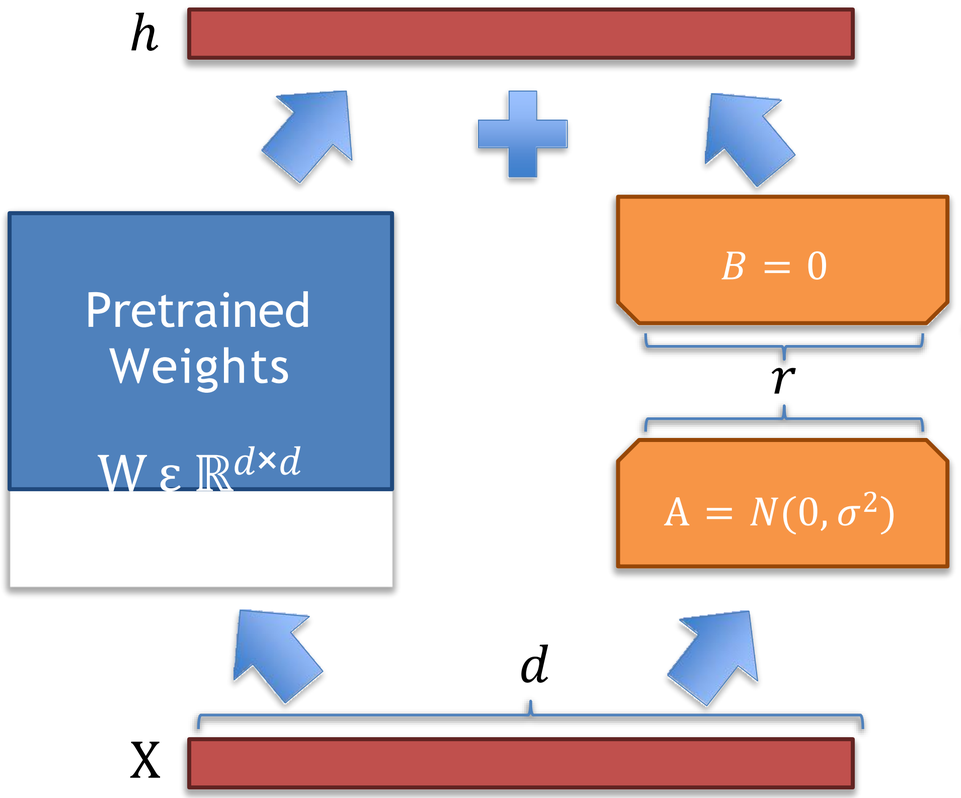
LoRA runs parallel to the original model.

LoRA - Working



During the backward pass, we don't use the pre-trained weights and the gradients only flow through the adapters.

LoRA - Working



At inference time, the update matrix can be merged with the original model weights to make the process efficient.

Notice how the reparameterization (LoRA) runs parallel to the original model.

LoRA - Intuition

Number of trainable parameters

Rank	Model 7B	Model 13B	Model 70B	Model 180B
1	167K	228K	529K	849K
2	334K	456K	1M	2M
8	1M	2M	4M	7M
16	3M	4M	8M	14M
512	86M	117M	270M	434M
1024	171M	233M	542M	869M
8192	1.4B	1.8B	4.3B	7B
Full	7B	13B	70B	180B

LoRA - Advantages

Compared to full parameter finetuning, LoRA has the following advantages:

1. Much faster
2. Finetuning can be achieved using less GPU memory
3. Cost efficient
4. Less prone to "catastrophic forgetting" since the original model weights are kept the same.

QLoRA

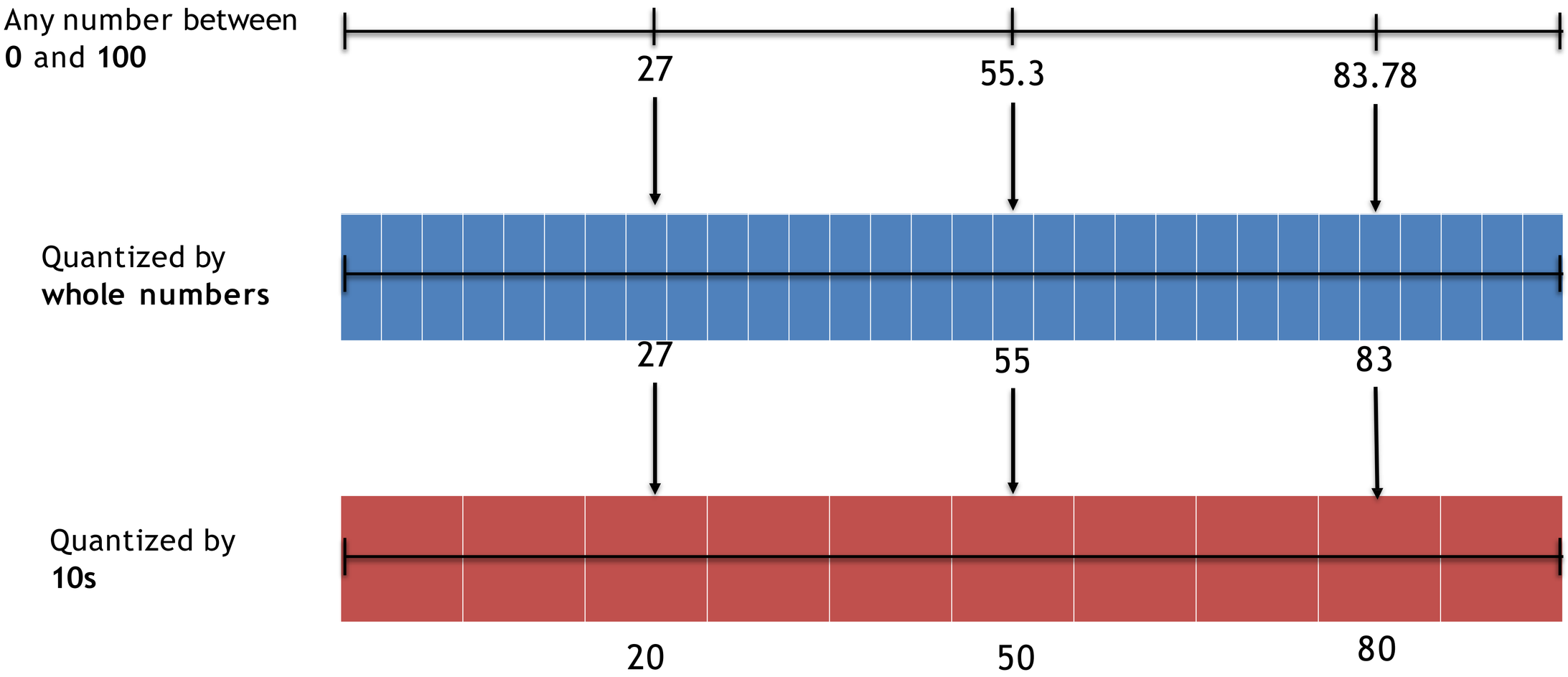
- QLoRA is the extended version of LoRA which works mainly by quantizing the precision of the network parameters.

- Before we dive into what QLoRA is, let's look at what quantization is.

Think of quantization as 'splitting range into buckets '.

Quantization

Think of quantization as ‘splitting range into buckets ’.

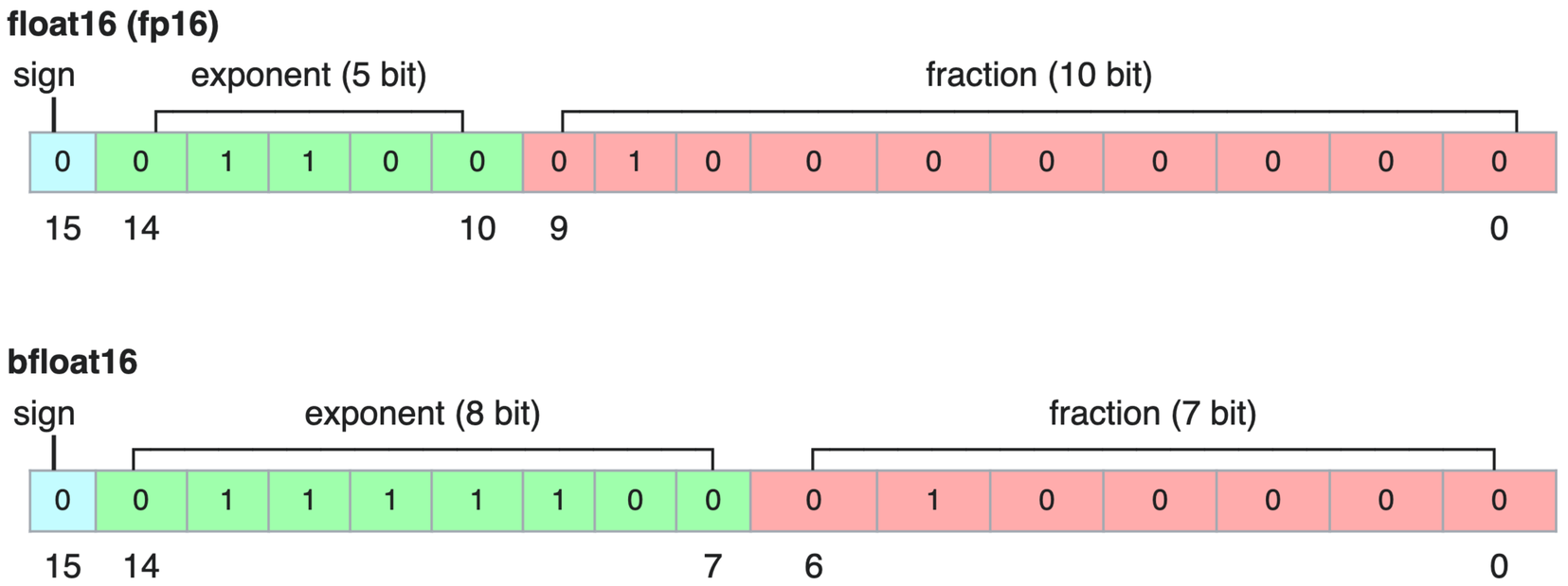


Floating point numbers

Floating point number is stored as $(-1)^s$ M 2^E

- Sign bit *s*
- Fractional part *M* = frac
- Exponential part *E* = exp - bias

Reduced-precision floating point types



QLoRA

Let's look at an example!

Here, FP32 refers to a 32- bit floating-point number.

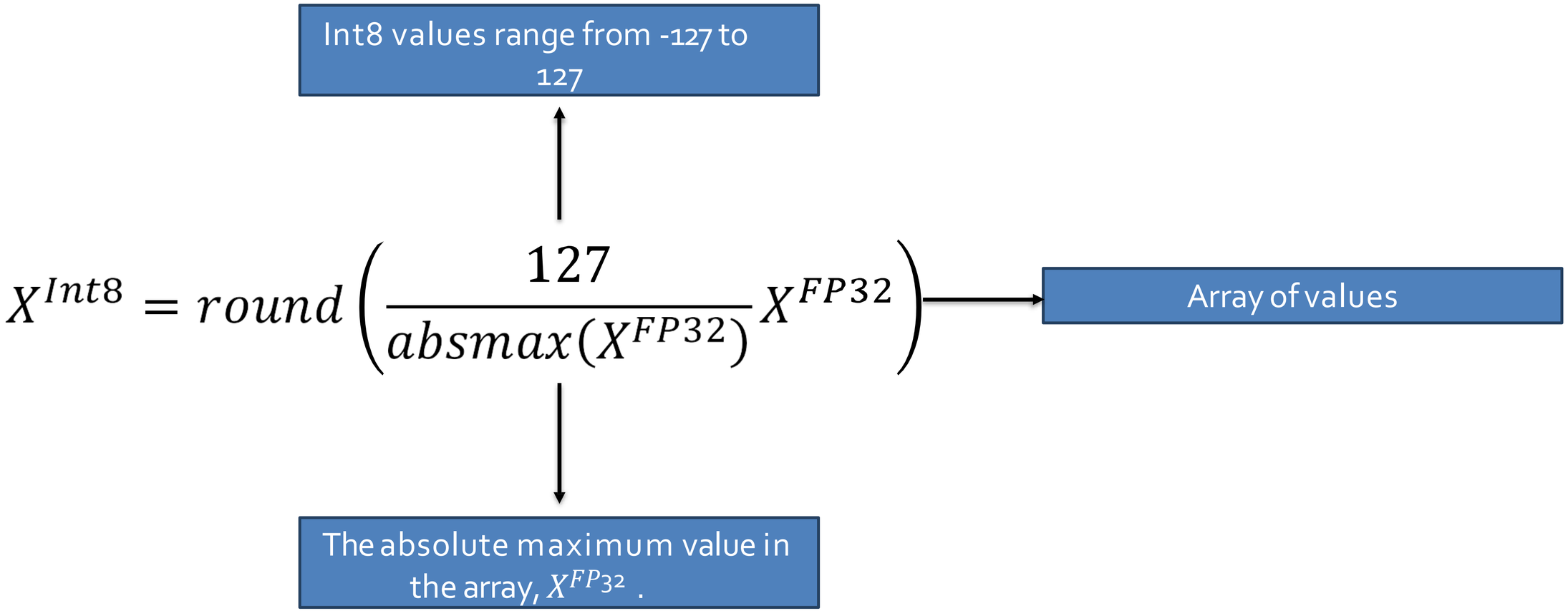
Let X^{FP32} be an array of values.

1.5	2.3	3.7	4.1	5.6	6.8	7.9	8.4	9.2	10.2

What if we want to quantize from FP32 to Int8?

QLoRA

So, to quantize X^{FP32} to X^{Int8} :



QLoRA

So, to quantize X^{FP32} to X^{Int8} :

$$X^{Int8} = round\left(\frac{127}{absmax(X^{FP32})} X^{FP32}\right)$$

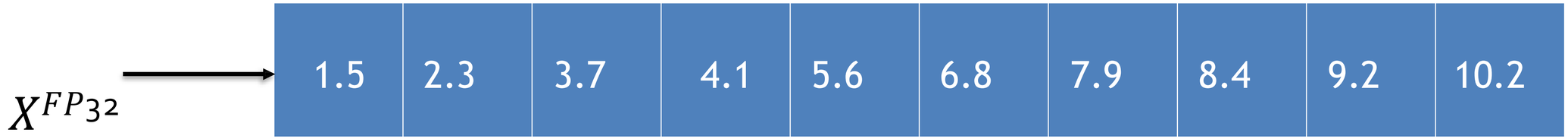
$$\downarrow$$

$$X^{Int8} = round(c^{FP32} X^{FP32})$$

QLoRA

$$X^{Int8} = round(c^{FP32} X^{FP32})$$

In our example,



$$c^{FP32} = \frac{127}{absmax(X^{FP32})} = \frac{127}{10.2} = 12.4509$$

Now, we combine the formula and the values that we have

QLoRA

$$X^{Int8} = round(12.4509 \times$$

1.5	2.3	3.7	4.1	5.6	6.8	7.9	8.4	9.2	10.2

$)$

$$X^{Int8} =$$

18	29	46	51	69	85	98	105	115	127

$$X^{Int8} = round(c^{FP32} X^{FP32})$$

QLoRA

$$X^{Int8} = round(c^{FP32} X^{FP32})$$

What if we want to dequantize and get back the original array, X^{FP32}?

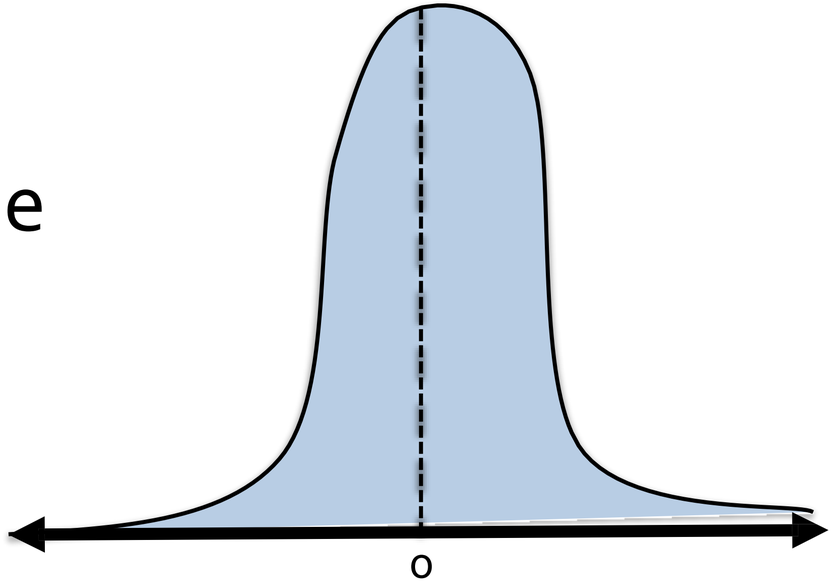
To dequantize:

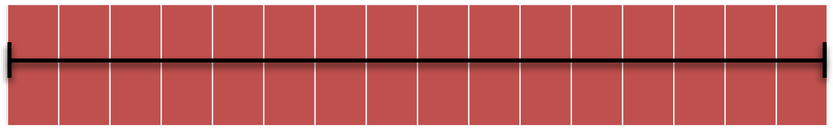
$$X^{FP32} = \frac{X^{Int8}}{c^{FP32}}$$

QLoRA – Ingredient 1: 4-Bit NormalFloat

- 4-bit NormalFloat is a clever way to split the buckets.

4-bit means we have
$2^4 = 16$ possible buckets for quantization.





Equally spaced buckets

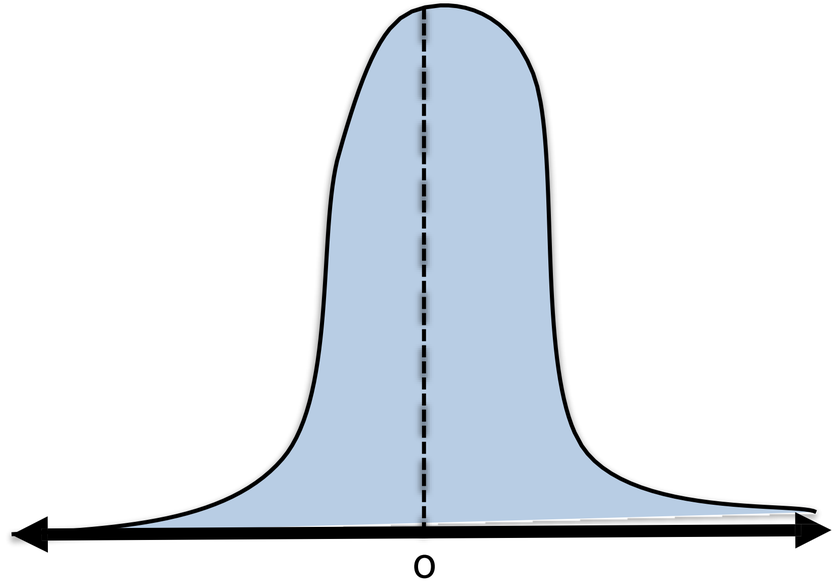
Equally sized buckets

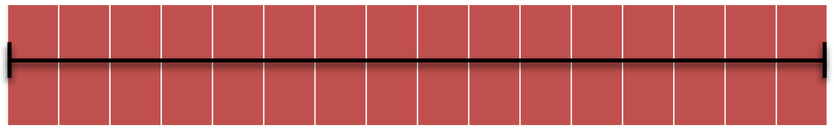
This is an enhanced version of **quantile quantization**.

QLoRA – Ingredient 1: 4-Bit NormalFloat

- Why use 4-bit NormalFloat
- Designed for efficient storage and computation in machine learning.

Most datasets in machine learning are normally distributed and precision around the mean is valuable.





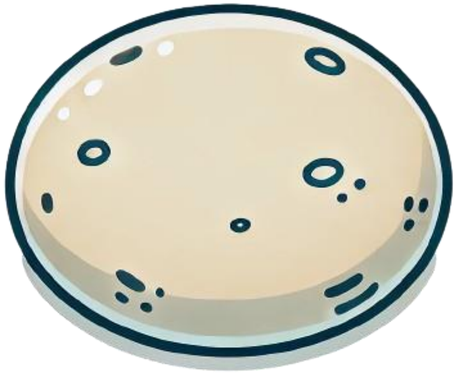
Equally spaced buckets

Equally sized buckets

This is an enhanced version of **quantile quantization**.

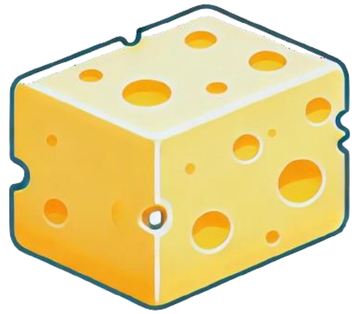
QLoRA – The Ingredients

There are 3 key ingredients which helps us make QLoRA:



4-Bit NormalFloat

Double Quantization
(Quantize in blocks, quantize each of the constants too)



Paged Optimizer
(Offload some of the activations etc to CPU to save memory)

QLoRA

- QLoRA can replicate 16-bit full fine-tuning performance with a 4-bit base model and Low-rank Adapters.
-
- It's the first method that enables fine-tuning of 33B parameter models on a single consumer GPU and 65B parameter models on a single professional GPU without degrading performance relative to a full finetuning baseline.

- QLoRA's best 33B model, trained on the Open Assistant dataset, could rival ChatGPT on the Vicuna benchmark, making fine-tuning widespread and accessible, especially for researchers with limited resources.

Inference Efficiency

(We now know that) Training big models is expensive

Table 1: We developed our models in five groups, based on parameter count and architecture: less than 1 billion, 1 billion, 7 billion, and 13 billion parameters, and our mixture-of-experts model with 1 billion active and 7 billion total parameters. We found that ∼70% of our developmental environmental impact came from developing the 7B and 13B models, and the total impact was emissions equivalent to 2.1 tanker trucks' worth of gasoline, and equal to about 7 and a half years of water used by the average person in the United States.

	GPU Hours	Total MWh	# Runs	Carbon Emissions (tCO_2eq)	Equivalent to... (energy usage, 1 home, U.S.)	Water Consumption (kL)	Equivalent to... (water usage, 1 person)
<1B	29k	19	20	6	1 yr, 4 mo	24	3 mo
7B	269k	196	375	65	13 yrs, 6 mo	252	2 yrs, 7 mo
13B	191k	116	156	46	9 yrs, 7 mo	402	3 yrs, 7 mo
MoE	27k	19	35	6	1 yr, 4 mo	24	3 mo
Total	680k	459	813	159	33 yrs, 1 mo	843	7 yrs, 5 mo

HOLISTICALLY EVALUATING THE ENVIRONMENTALIMPACT OF CREATING LANGUAGE MODELS

Training models is expensive, but inference can be even more expensive

More importantly, inference costs far exceed training costs when deploying a model at any reasonable scale. In fact, the costs to inference ChatGPT exceed the training costs on a weekly basis.

https://www.semianalysis.com/p/the-inference-cost-of-search-disruption

How can we cheaply and efficiently deploy NLP systems without sacrificing performance?

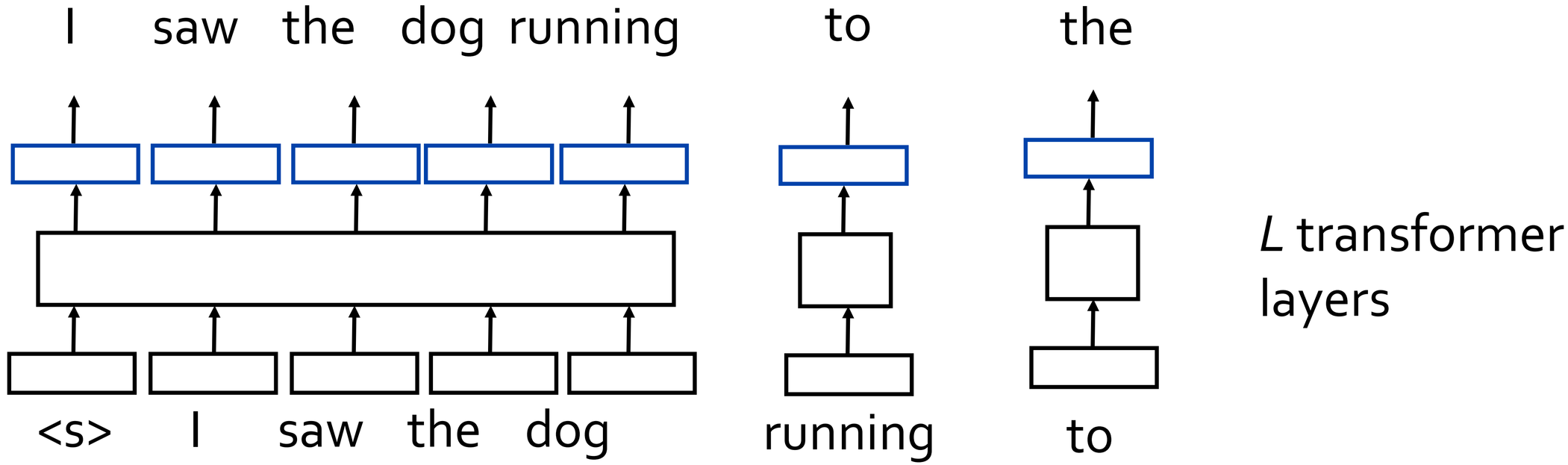
Overview

- Decoding optimizations:
  - Speculative decoding
  - Medusa heads
  - Flash attention

- Model compression
  - Quantization
  - Distillation

Decoding Optimizations

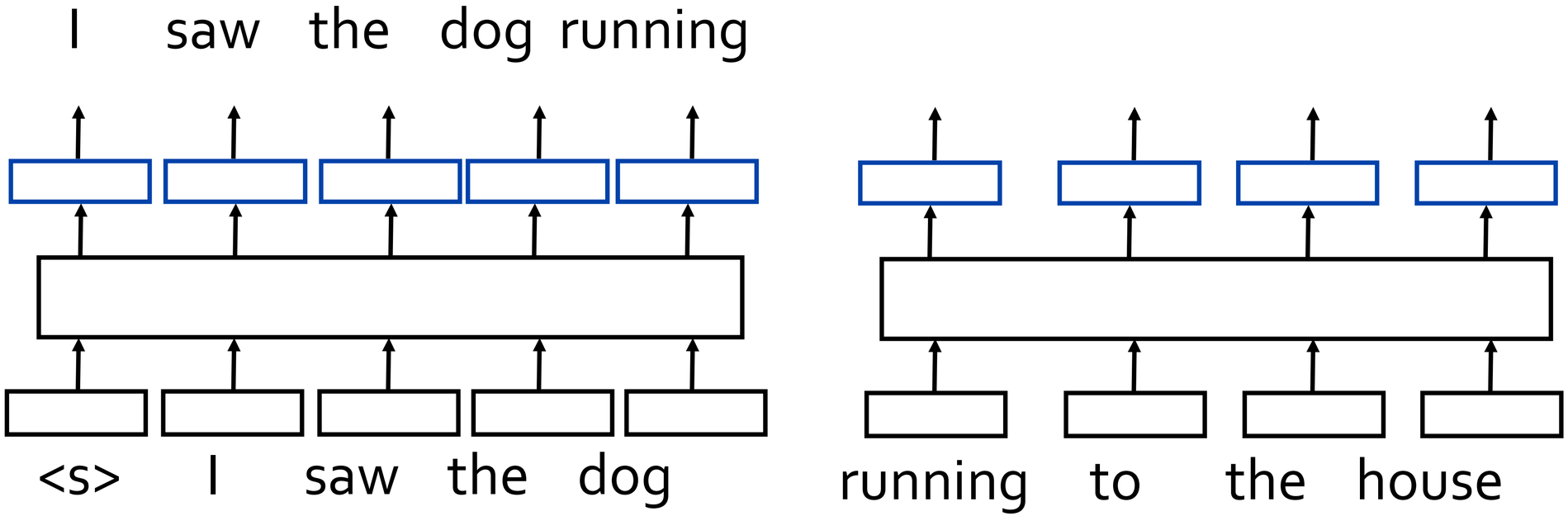
Decoding Basics



Operations for one decoder pass (on a GPU): $O(L)$

Operations for k decoder passes (on a GPU): $O(kL)$

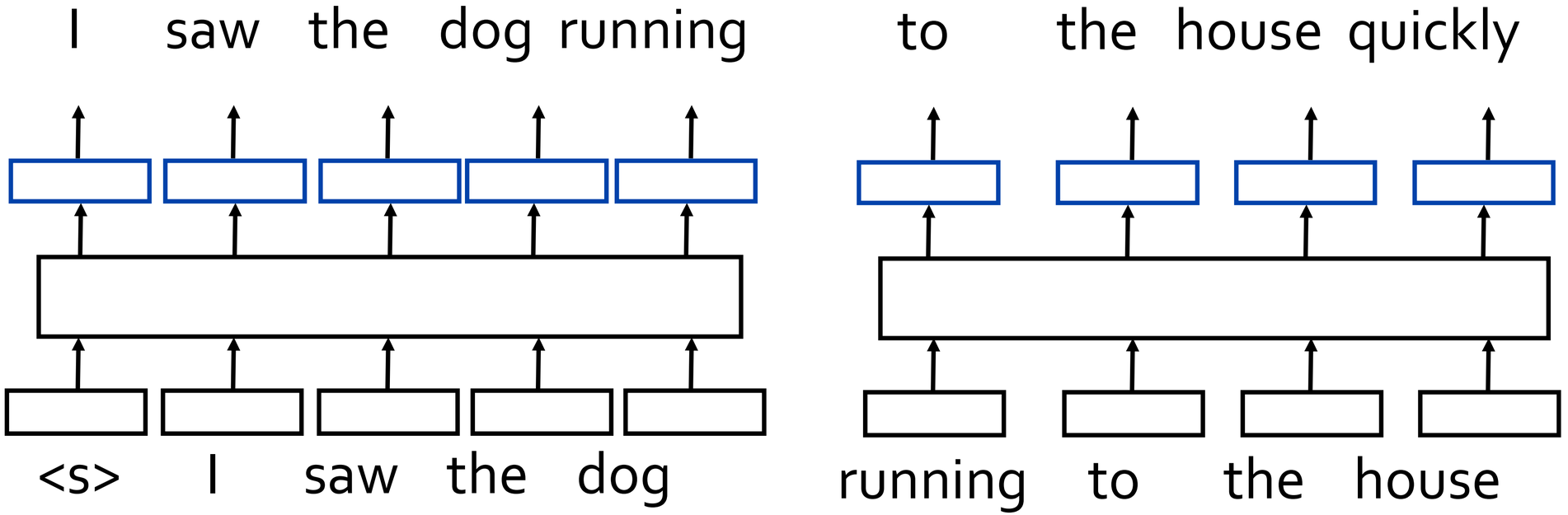
Speculative Decoding



Prompt (prefix of *p* tokens) Decoded tokens (*k*)

- Key idea: a forward pass for several tokens at a time is O(*L*) serial steps, since attention can be computed in parallel.
- Can we predict many tokens with a weak model and then "check" them with a single forward pass?

Speculative Decoding



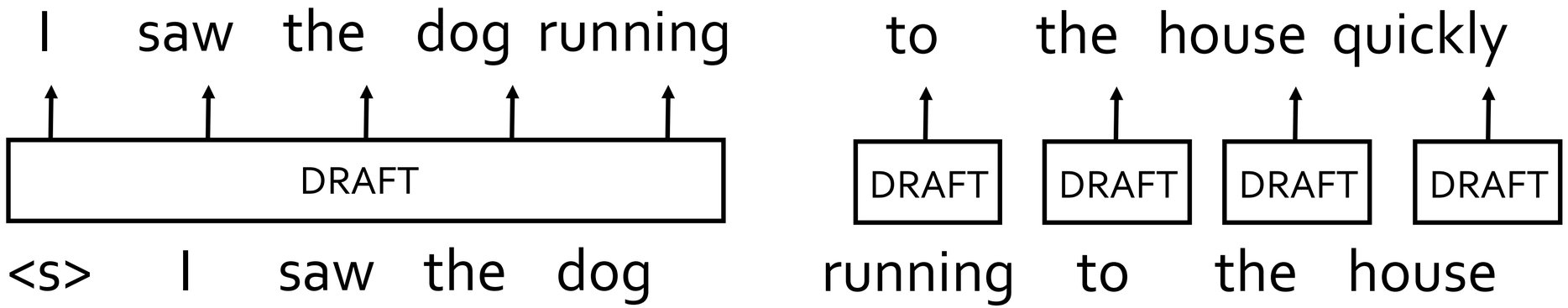
Prompt (prefix of *p* tokens)

Decoded tokens (*k*)

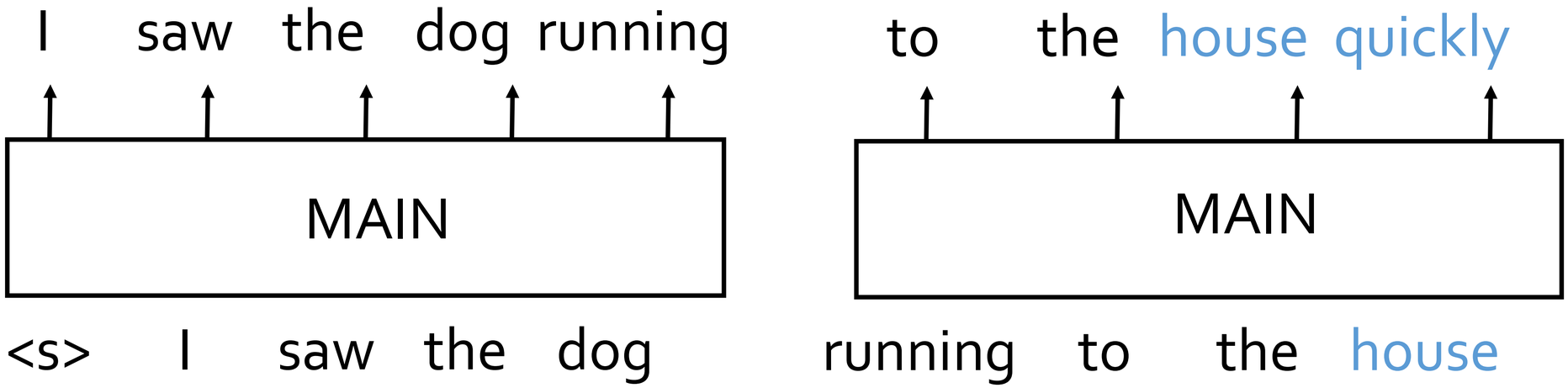
- **We can use a small, cheap model to do inference, then check that "to", "the", "house", "quickly" are really the best tokens from a bigger model**

Leviathan et al. (2023)

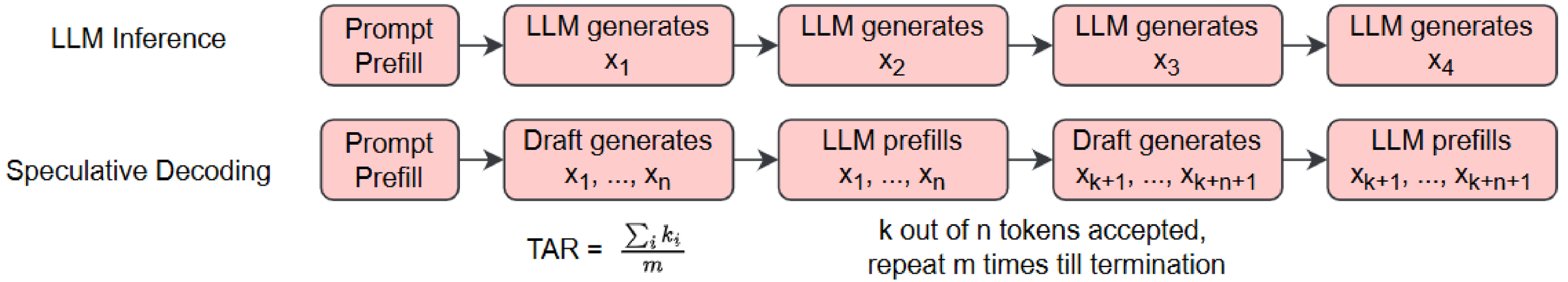
Speculative Decoding: Flow



- Produce decoded tokens one at a time from a fast draft model…

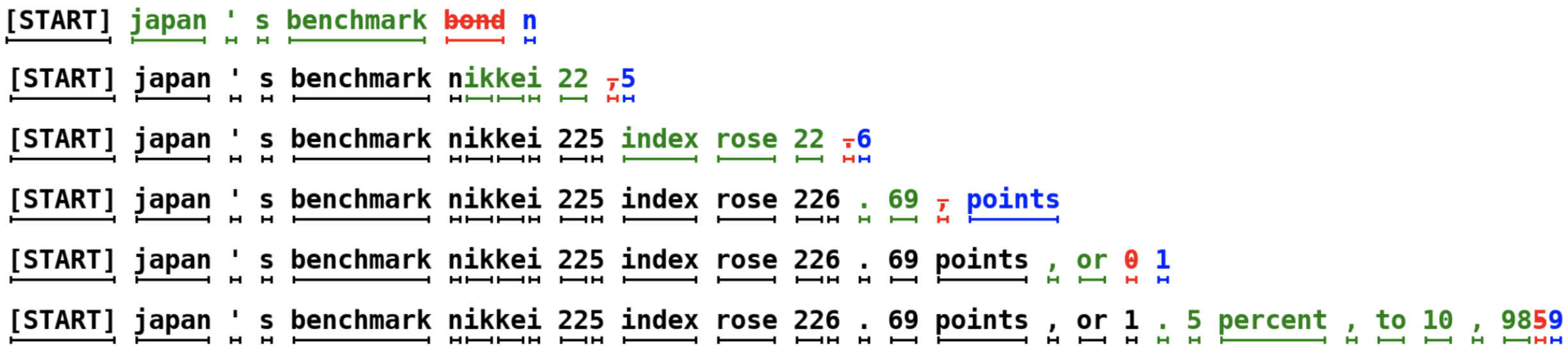


- Confirm that the tokens are the max tokens from the slower main model. Any "wrong" token invalidates the rest of the sequence


LLM Inference
Prompt Prefill
LLM generates x_1
LLM generates x_2
LLM generates x_3
LLM generates x_4
Speculative Decoding
Prompt Prefill
Draft generates $x_1, ..., x_n$
LLM prefills $x_1, ..., x_n$
Draft generates $x_{k+1}, ..., x_{k+n+1}$
LLM prefills $x_{k+1}, ..., x_{k+n+1}$
TAR = $\frac{\sum_i k_i}{m}$
k out of n tokens accepted,
repeat m times till termination

Speculative Decoding

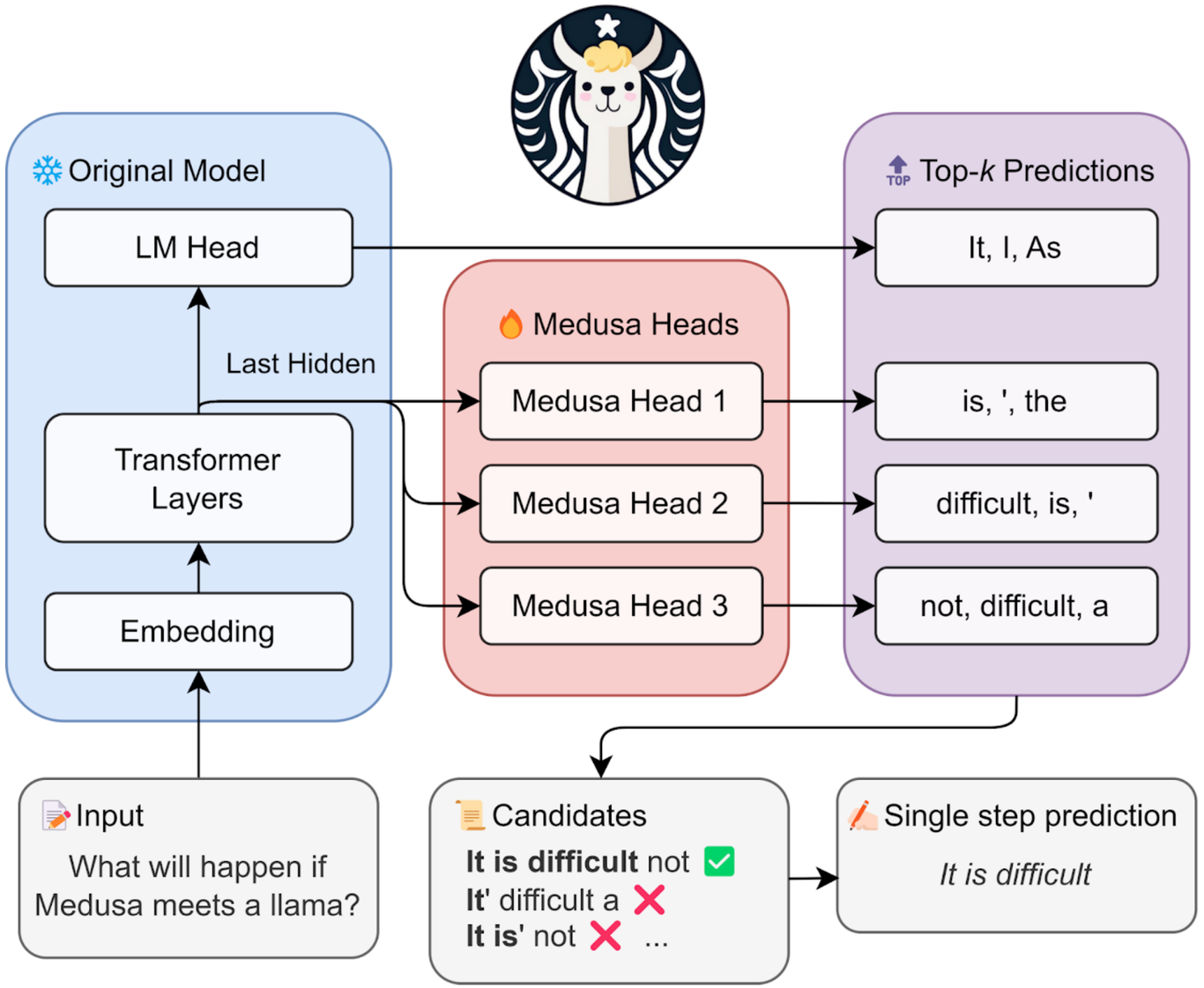
Leviathan et al. (2023)



- Can also adjust this to use sampling. Treat this as a proposal distribution $q(x)$ and may need to reject + resample (rejection sampling)

Medusa Heads

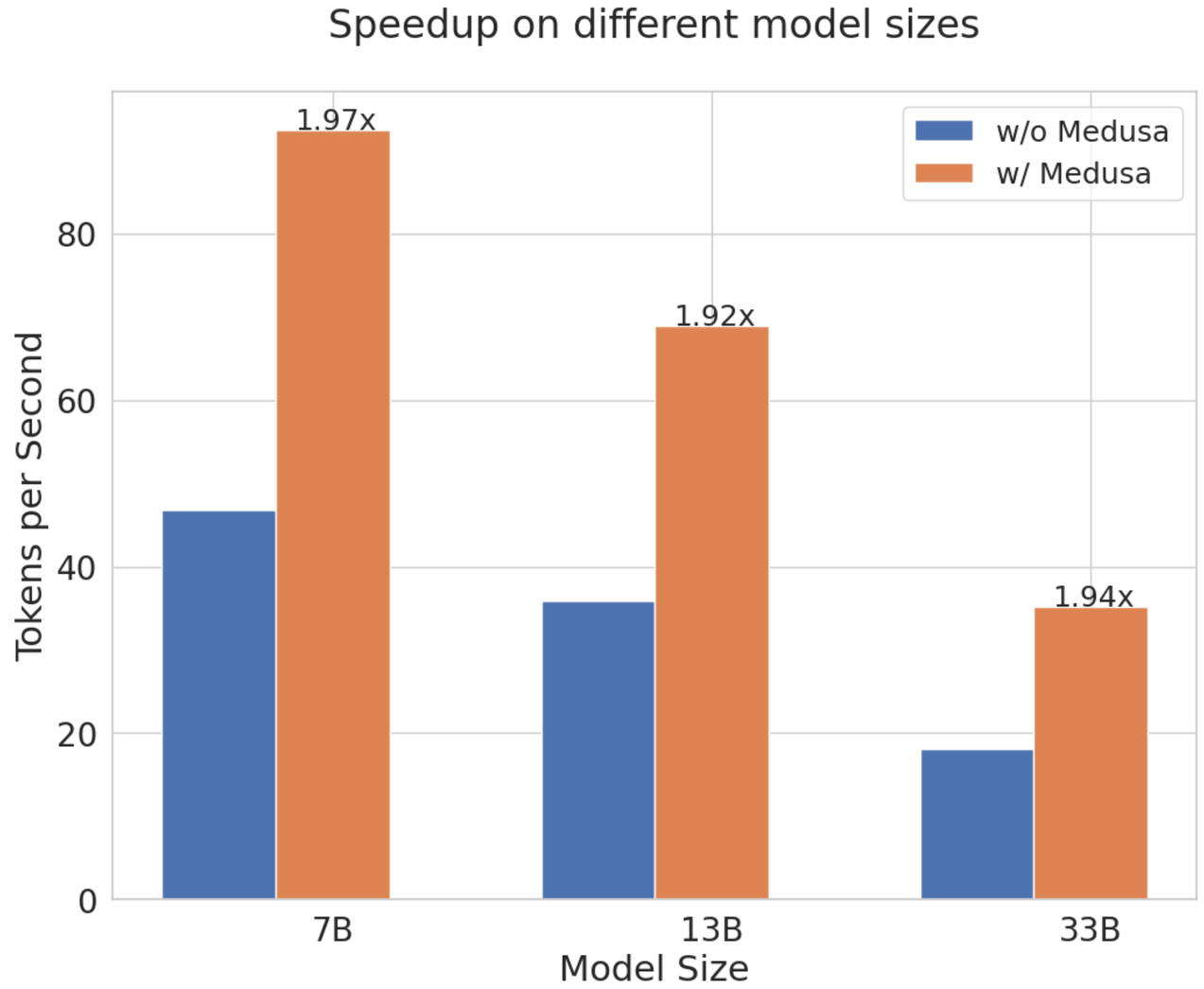
- The “draft model” consists of multiple prediction heads trained to predict the next k tokens



https://www.together.ai/blog/medusa

Medusa Heads

- Speedup with no loss in accuracy!



https://www.together.ai/blog/medusa

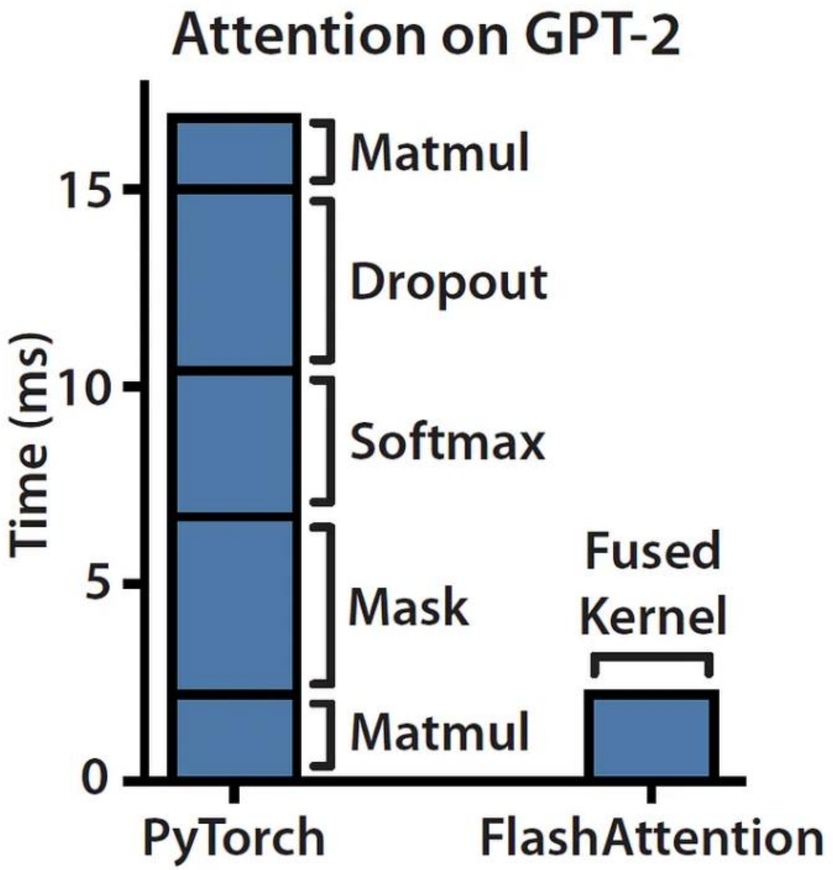
Other Decoding Improvements

- Batching parallelism: improve throughput by decoding many examples in parallel. (Does not help with latency, and it's a little bit harder to do in production if requests are coming in asynchronously)
- Low-level hardware optimizations?
  - Easy things like caching (KV cache: keys + values for context tokens are cached across multiple tokens)

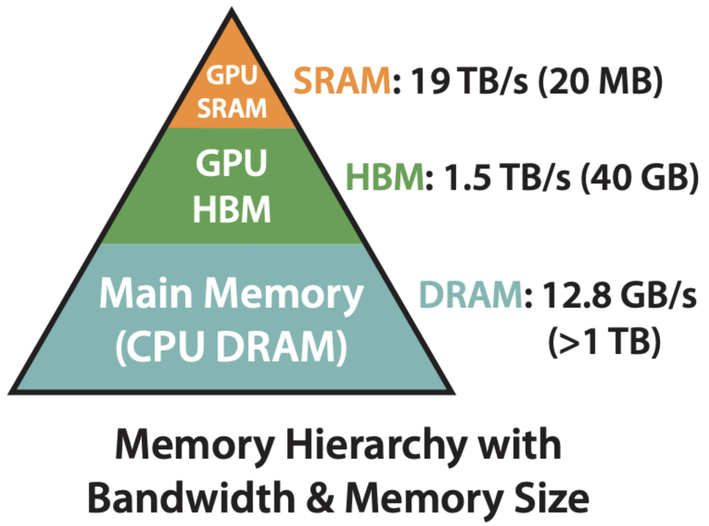
Flash Attention

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

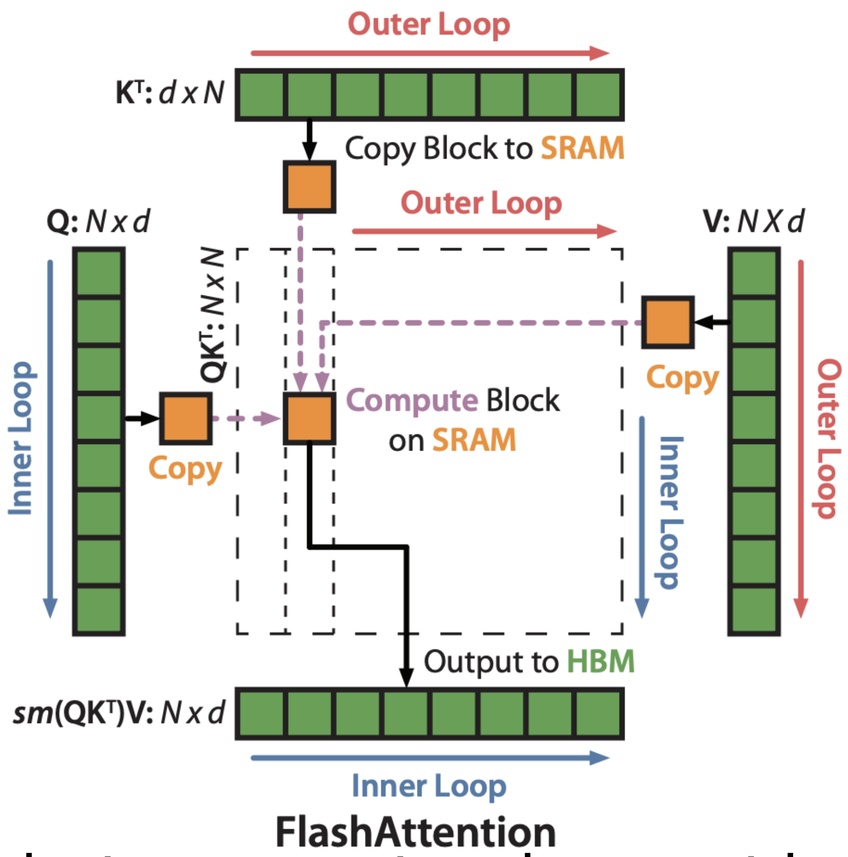
Operation	Cost	Bound
$QK^\top$	$\mathcal{O}(nmd_k)$	**Compute-bound**
Scaling $\div\sqrt{d_k}$	$\mathcal{O}(nm)$	Memory-bound
Softmax	$\mathcal{O}(nm)$	Memory-bound
$\text{softmax}(\ldots)V$	$\mathcal{O}(nmd_v)$	**Compute-bound**



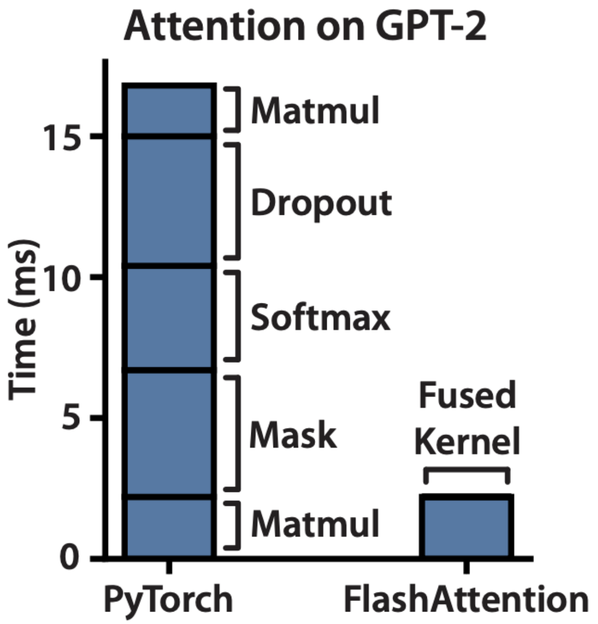
Flash Attention



Memory Hierarchy with Bandwidth & Memory Size



FlashAttention



- Does extra computation during attention, but avoids expensive reads/writes to GPU "high-bandwidth memory." Recomputation is all in SRAM and is very fast
- Essentially: store a running sum for the softmax, compute values as needed

Flash Attention

Models	ListOps	Text	Retrieval	Image	Pathfinder	Avg	Speedup
Transformer	36.0	63.6	81.6	42.3	72.7	59.3	-
FlashAttention	37.6	63.9	81.4	43.5	72.7	59.8	2.4×
Block-sparse FlashAttention	37.0	63.0	81.3	43.6	73.3	59.6	**2.8**×
Linformer [84]	35.6	55.9	77.7	37.8	67.6	54.9	2.5×
Linear Attention [50]	38.8	63.2	80.7	42.6	72.5	59.6	2.3×
Performer [12]	36.8	63.6	82.2	42.1	69.9	58.9	1.8×
Local Attention [80]	36.1	60.2	76.7	40.6	66.6	56.0	1.7×
Reformer [51]	36.5	63.8	78.5	39.6	69.4	57.6	1.3×
Smyrf [19]	36.1	64.1	79.0	39.6	70.5	57.9	1.7×

- Gives a speedup for free — with no cost in accuracy (modulo numeric instability)
- Outperforms the speedup from many other approximate Transformer methods, which perform substantially worse

Model Compression

Model Compression

1. Quantization (already discussed)
   - keep the model the same but reduce the number of bits
2. Distillation
   - train a smaller model to imitate the bigger model

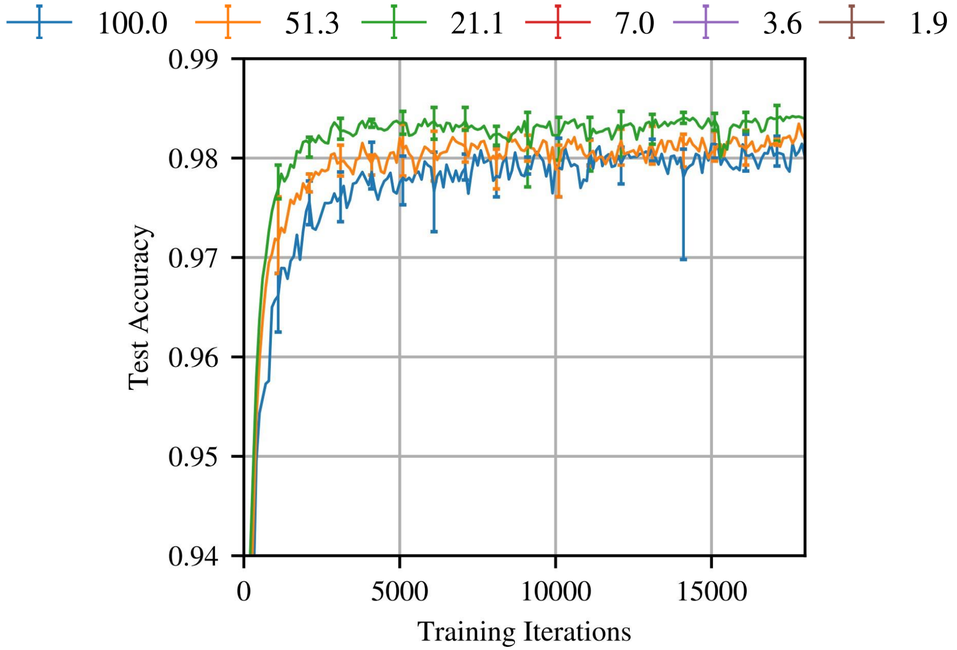
Why train big and then compress?

Overparameterized models are easier to optimize (Du and Lee 2018)

networks. For a k hidden node shallow network with quadratic activation and n training data points, we show as long as $k \geq \sqrt{2n}$, over-parametrization enables local search algorithms to find a *globally* optimal solution for general smooth and convex loss functions. Further, de-

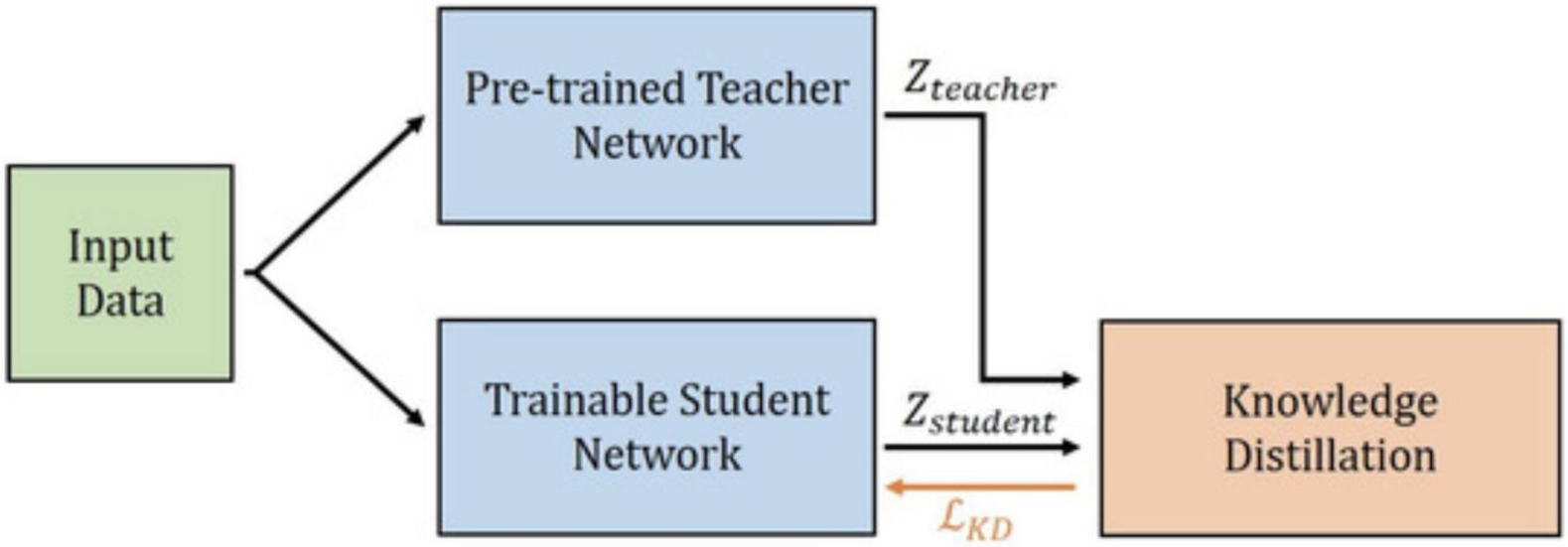
Lottery Ticket Hypothesis

Within a randomly initialized dense neural network, there exists a small subnetwork (a "winning ticket") that, when trained in isolation with the same initialization, can match or even outperform the original network.



DistilBERT

figure credit: Tianjian Li



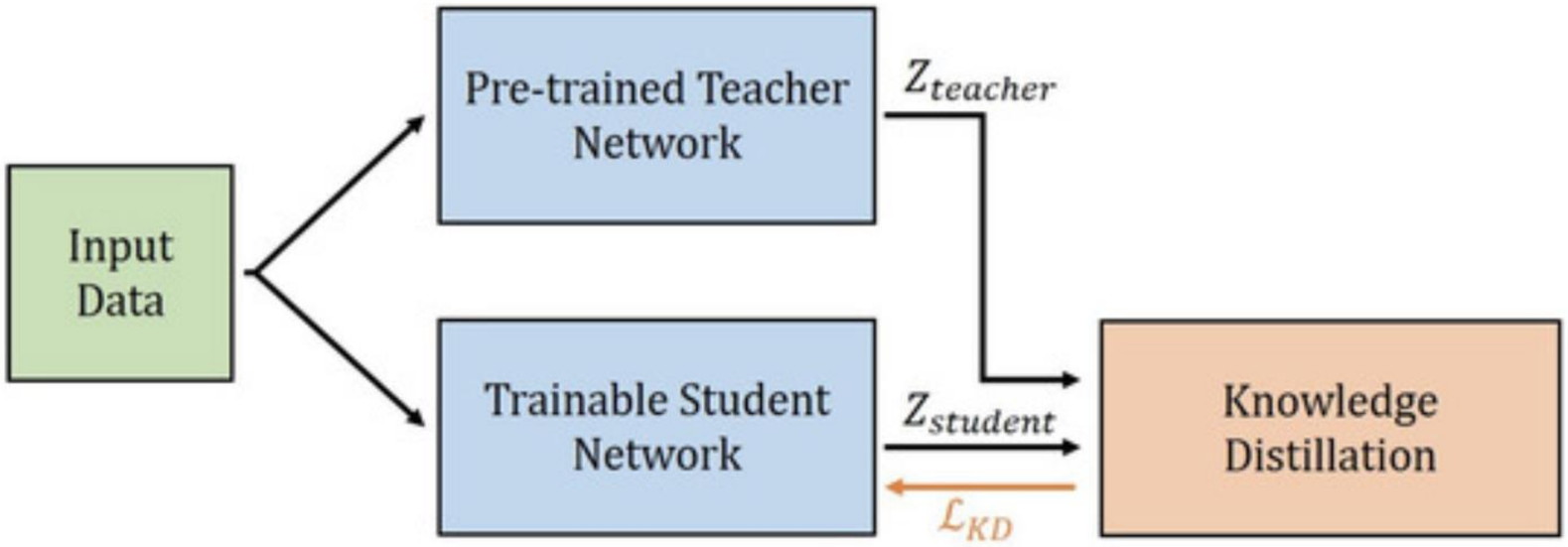
Suppose we have a classification model with output $P_{teacher}(y \mid \boldsymbol{x})$

Minimize KL($P_{teacher} \,||\, P_{student}$) to bring student dist close to teacher

Note that this is not using labels — it uses the teacher to "pseudo-label" data, and we label an entire distribution, not just a top-one label

DistilBERT

figure credit: Tianjian Li



- Use a teacher model as a large neural network, such as BERT
- Make a small student model that is half the layers of BERT. Initialize with every other layer from the teacher

Sanh et al. (2019)

DistilBERT

Model	**Score**	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

Table 2: **DistilBERT yields to comparable performance on downstream tasks.** Comparison on downstream tasks: IMDb (test accuracy) and SQuAD 1.1 (EM/F1 on dev set). D: with a second step of distillation during fine-tuning.

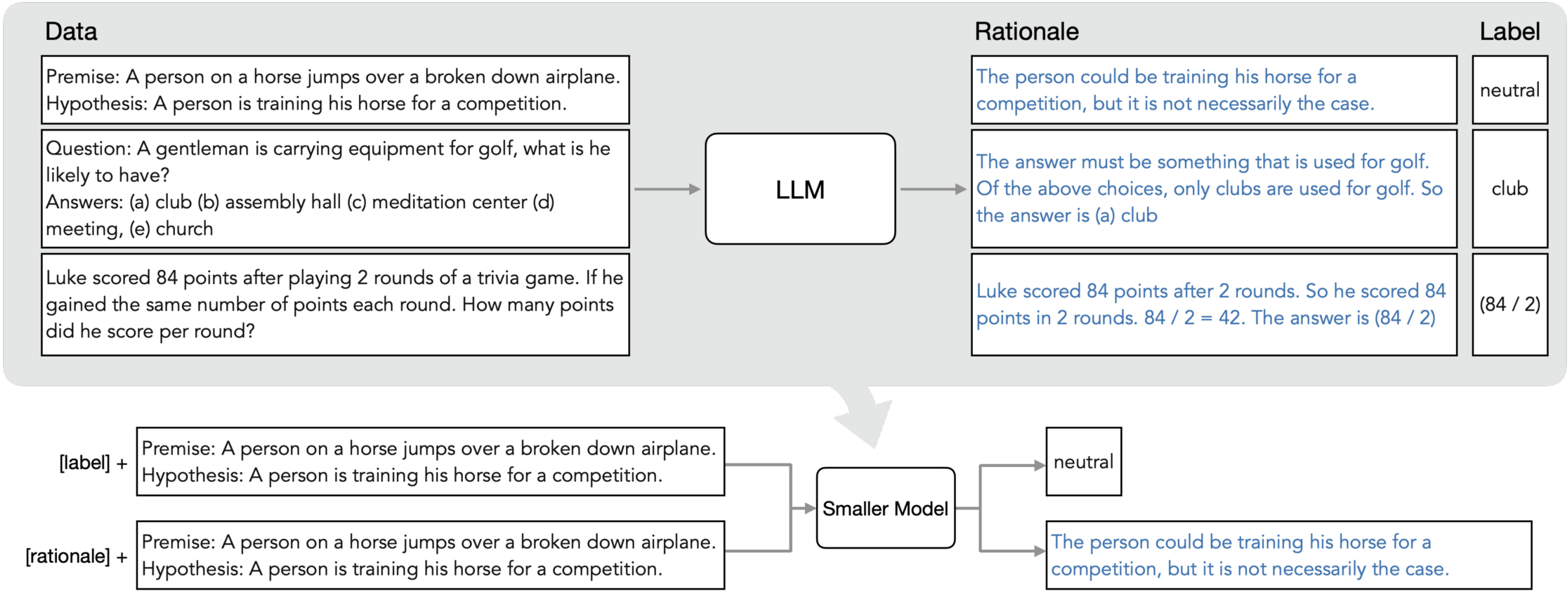
Model	IMDb (acc.)	SQuAD (EM/F1)
BERT-base	93.46	81.2/88.5
DistilBERT	92.82	77.7/85.8
DistilBERT (D)	-	79.1/86.9

Table 3: **DistilBERT is significantly smaller while being constantly faster.** Inference time of a full pass of GLUE task STS-B (sentiment analysis) on CPU with a batch size of 1.

Model	# param. (Millions)	Inf. time (seconds)
ELMo	180	895
BERT-base	110	668
DistilBERT	66	410

Sanh et al. (2019)

Other Distillation (Synthetic Data Generation)



Cheng-Yu Hsieh et al. (2023)

Where is this going?

- **Better GPU programming:** as GPU performance starts to saturate, we'll probably see more algorithms tailored very specifically to the affordances of the hardware
- **Small models**, either distilled or trained from scratch: as LLMs gets better, we can do with ~7-14B scale what used to be only doable with ChatGPT (GPT-4)
- **Continued focus on faster inference**: faster inference can be highly impactful across all LLM applications

Takeaways

- Decoding optimizations: speculative decoding gives a fast way to exactly sample from a smaller model. Also techniques like Flash Attention
- Model compression and quantization: standard compression techniques, but adapted to work really well for GPUs
- Model optimizations to make models smaller: distillation