

Inference Efficiency, Multimodality

CSE 5525: Foundations of Speech and Natural Language
Processing

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



THE OHIO STATE UNIVERSITY

Logistics

- Final project:
 - Mid-project report is due April 1. Gradescope assignment has been created, details on Canvas.
 - Project presentations: April 22, 24.
 - Final project report due date: Tentatively April 30.

Last class Recap (before Spring break)

- Training Efficiency
 - LoRA -- train a small set of low-rank adapters (and keep the rest of the model frozen)
 - QLoRA – quantize the model parameters so that they take less memory (to 4 bits) and now you can finetune with LoRA a ~70B model on 1 GPU.
- Today: Inference Efficiency.

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 $\sim 70\%$ 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 ₂ eq)	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

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.

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

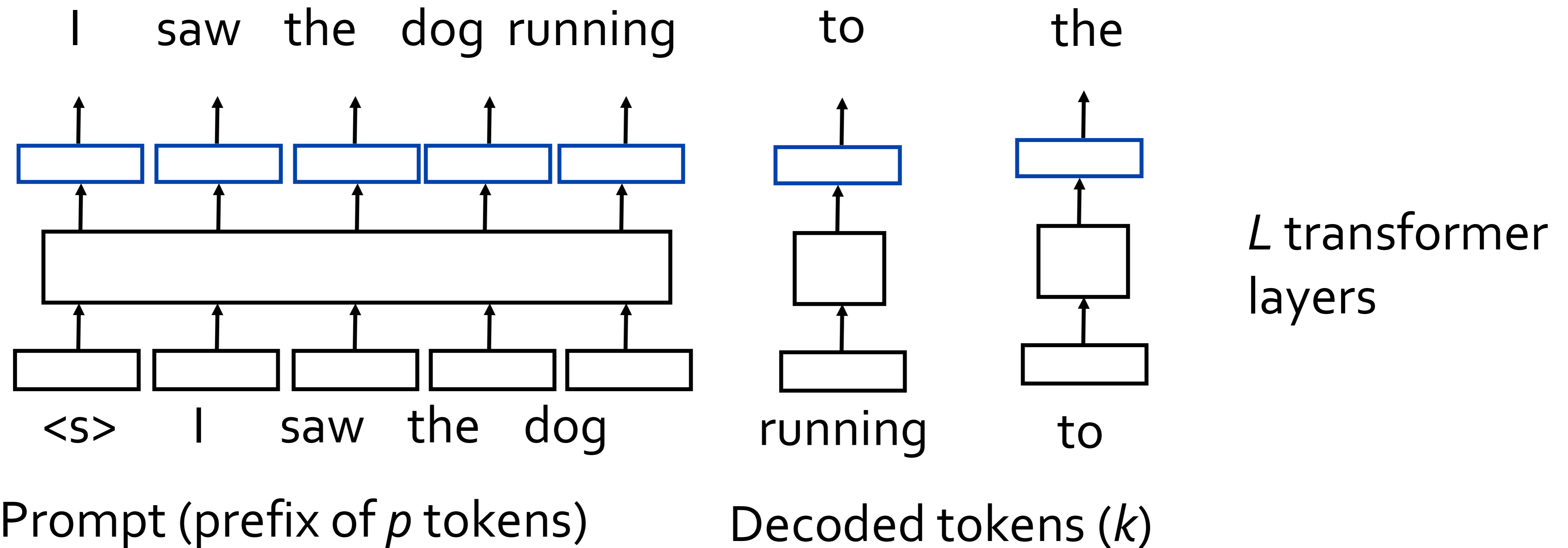
Overview

- ▶ Decoding optimizations:
 - ▶ Speculative decoding
 - ▶ Medusa heads
 - ▶ Flash attention (briefly)

- ▶ Model compression
 - ▶ Quantization (already discussed)
 - ▶ Distillation

Decoding Optimizations

Decoding Basics



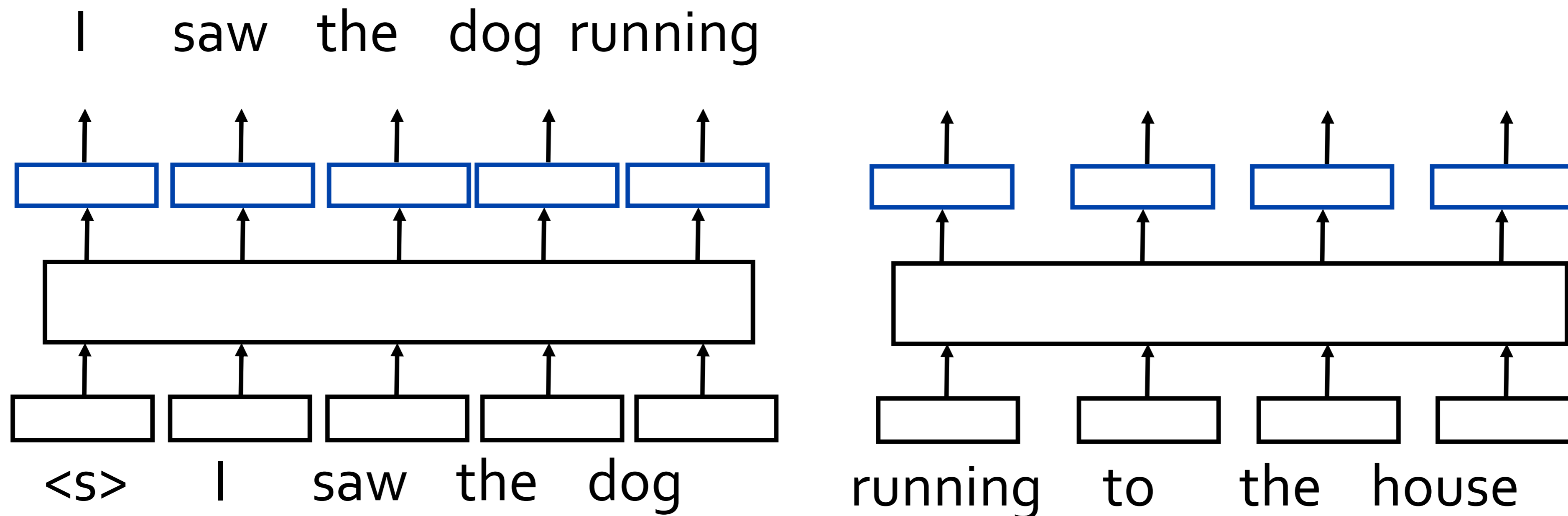
Operations for one decoder pass (on a GPU): $O(L)$

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

Simple 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 KV caching (keys + values for context tokens are cached across multiple tokens)

Speculative Decoding

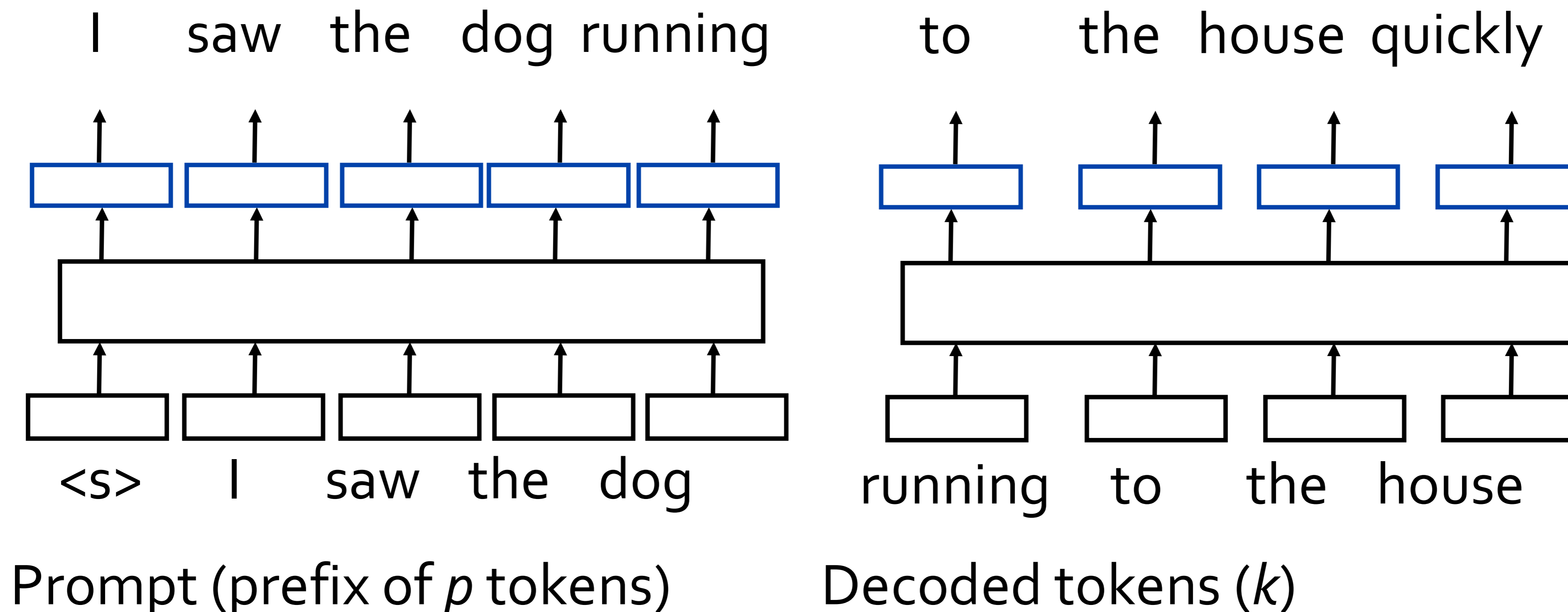


Prompt (prefix of p tokens)

Decoded tokens (k)

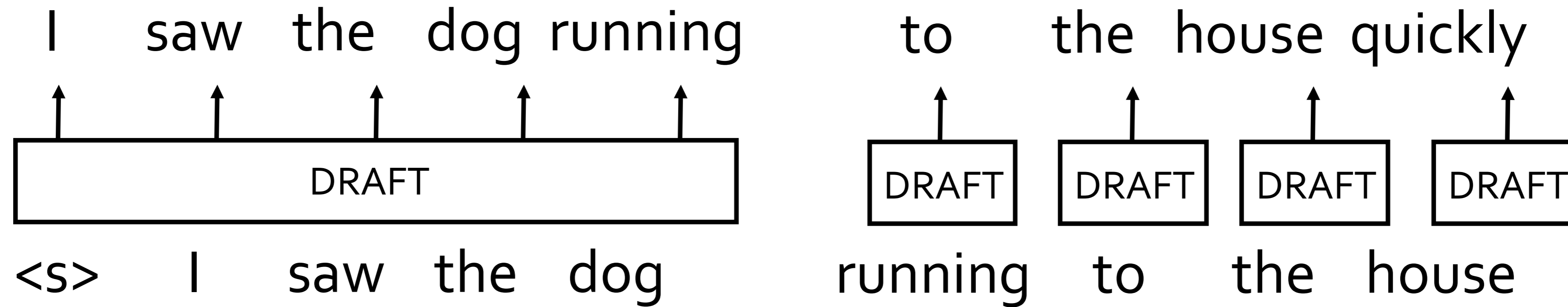
- ▶ Key idea: a forward pass for several tokens at a time is $O(L)$ serial steps, since attention can be computed in parallel.
- ▶ Can we predict many tokens with a weak model and then “check” them with a single forward pass?

Speculative Decoding

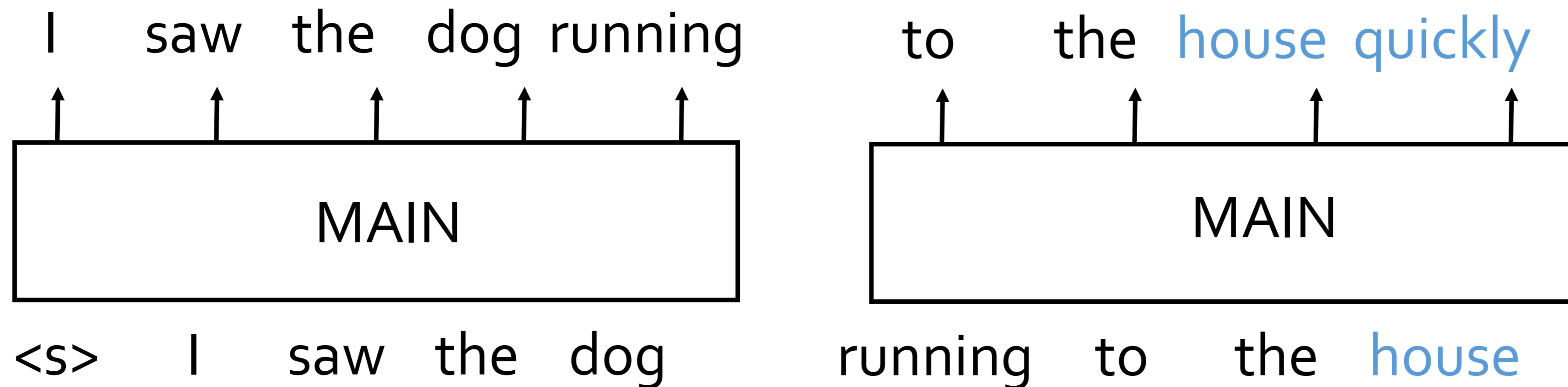


- ▶ We can use a small, cheap model to do inference, then check that “to”, “the”, “house”, “quickly” are really the best tokens from a bigger model

Speculative Decoding: Flow

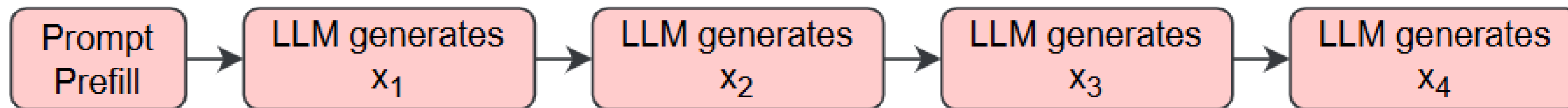


- Produce decoded tokens one at a time from a fast draft model...

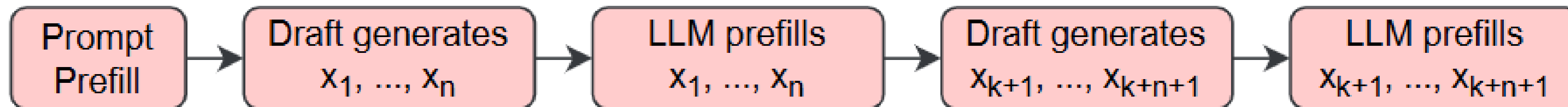


- Confirm that the tokens are the max tokens from the slower main model. Any “wrong” token invalidates the rest of the sequence

LLM Inference



Speculative Decoding



$$\text{TAR} = \frac{\sum_i k_i}{m}$$

k out of n tokens accepted,
repeat m times till termination

Speculative Decoding

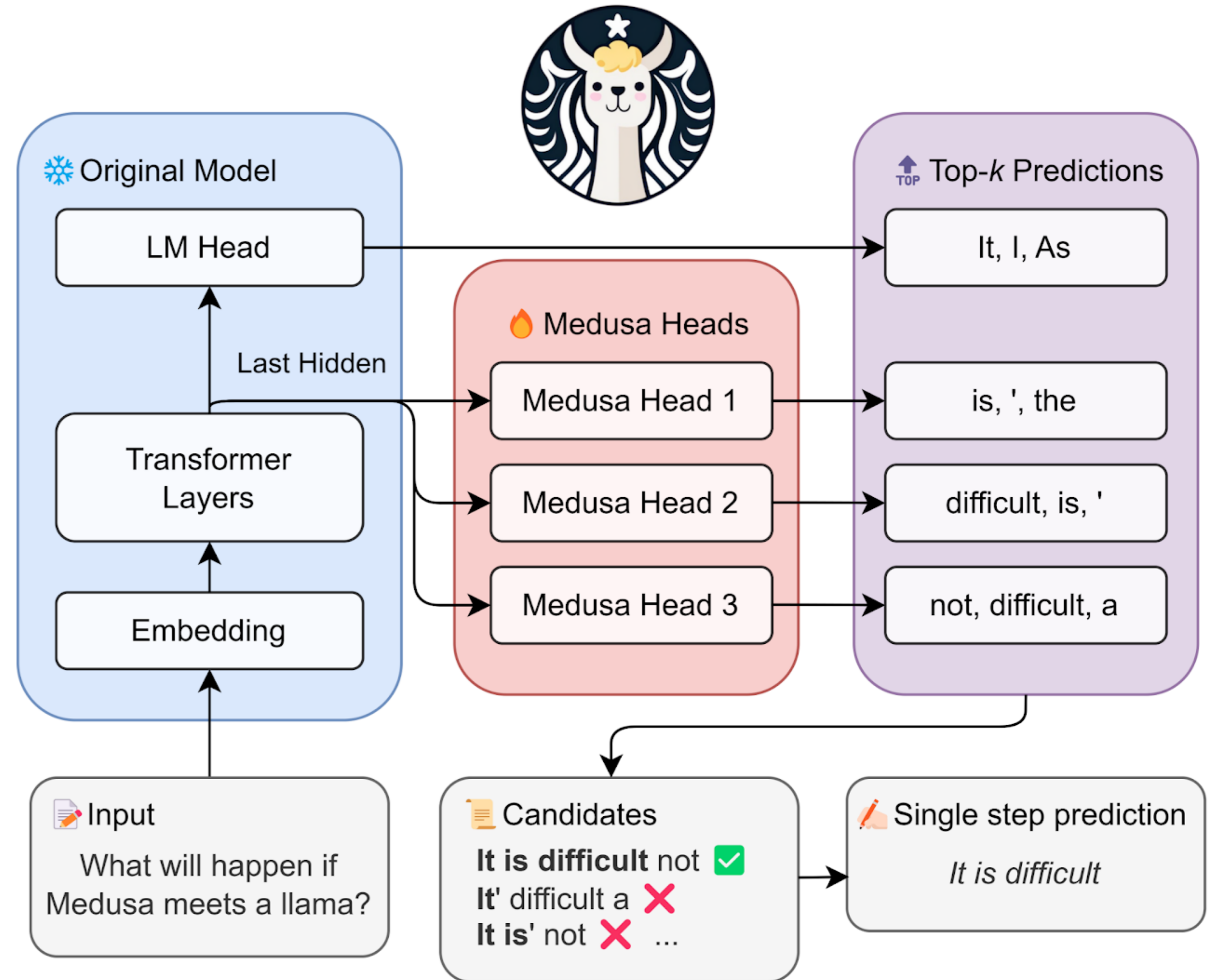
Leviathan et al. (2023)

[START] japan ' s benchmark ~~bond~~ n
[START] japan ' s benchmark nikkei 22 ~~5~~
[START] japan ' s benchmark nikkei 225 index rose 22 ~~6~~
[START] japan ' s benchmark nikkei 225 index rose 226 . 69 ~~1~~ points
[START] japan ' s benchmark nikkei 225 index rose 226 . 69 points , or ~~0~~ 1
[START] japan ' s benchmark nikkei 225 index rose 226 . 69 points , or 1 . 5 percent , to 10 , 9859

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

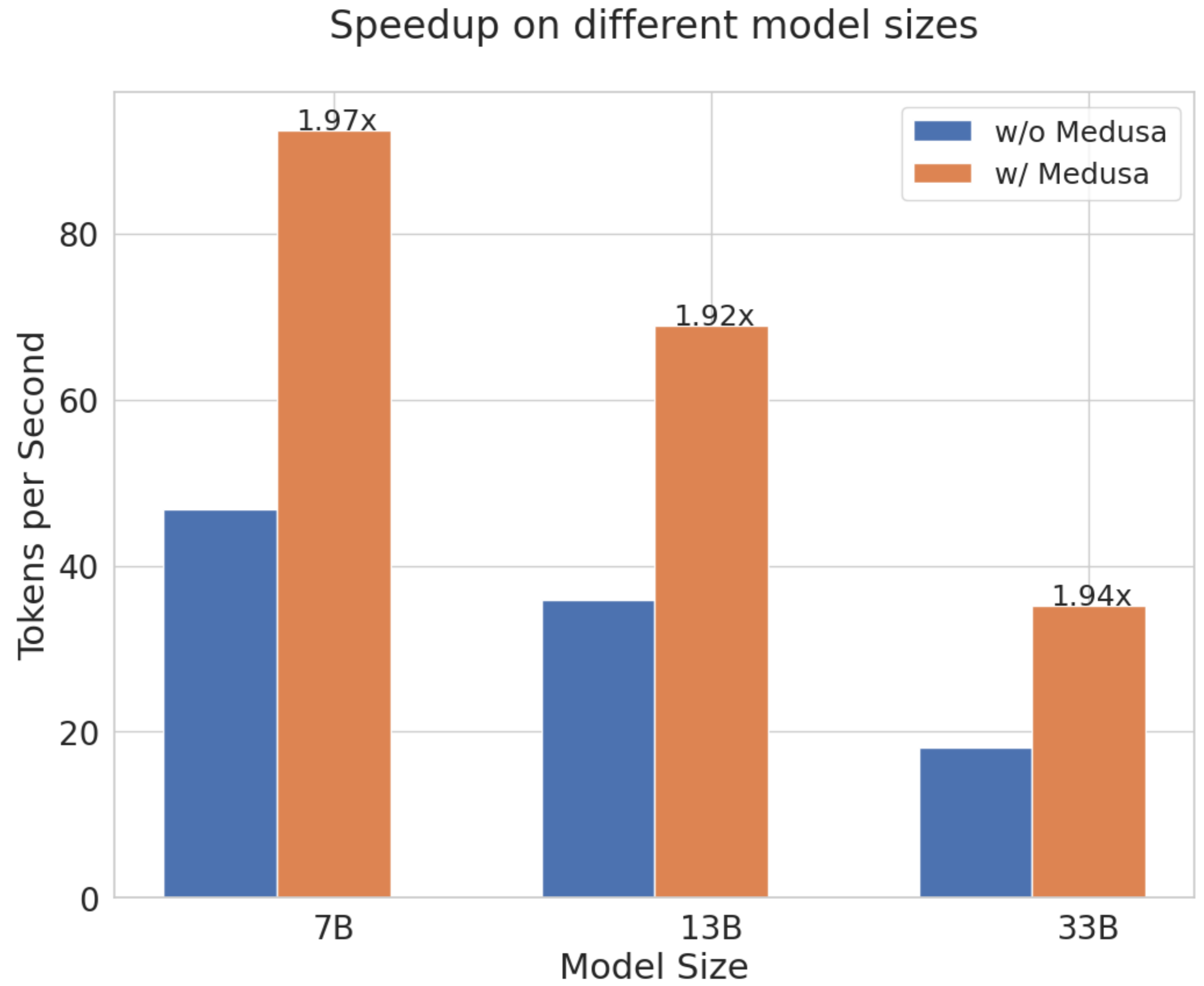
Medusa Heads

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



Medusa Heads

- ▶ Speedup with no loss in accuracy!

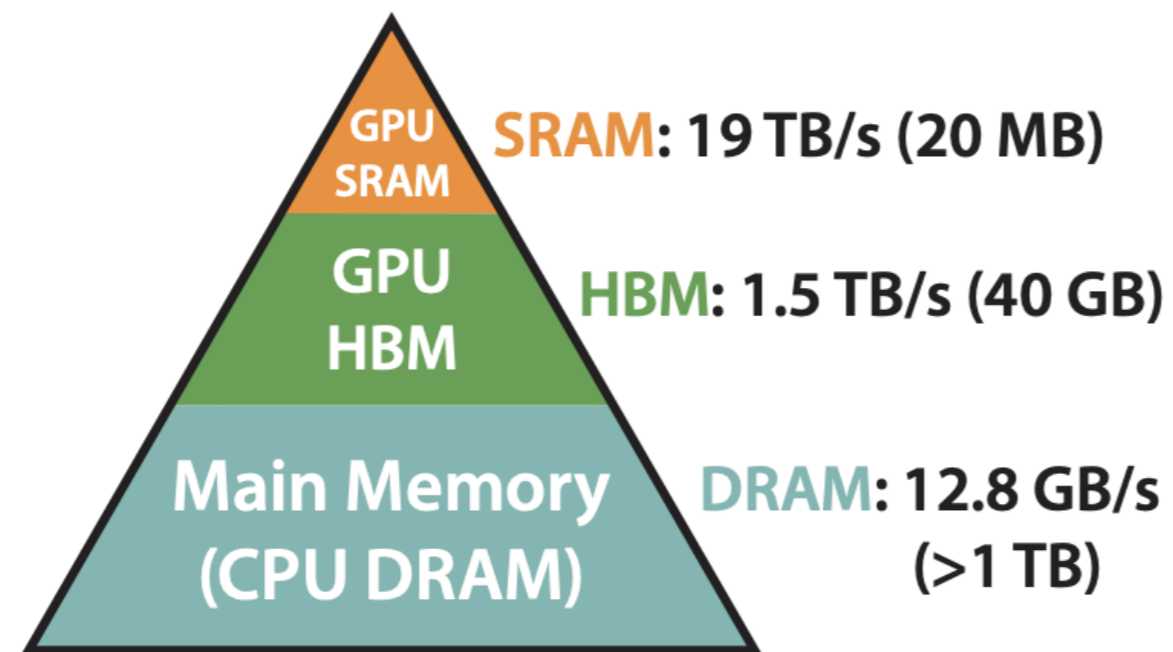


Flash Attention

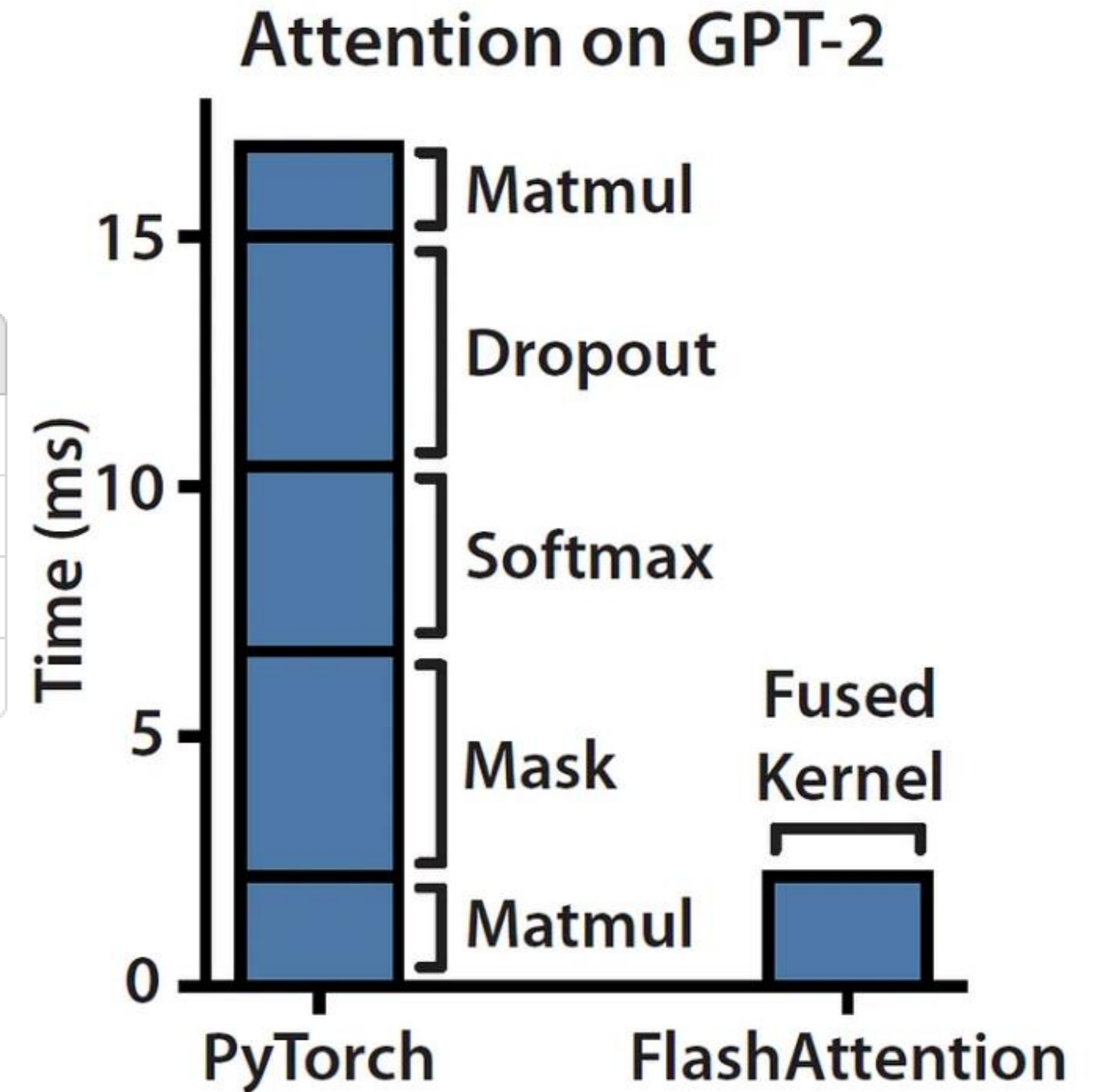
Flash Attention

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

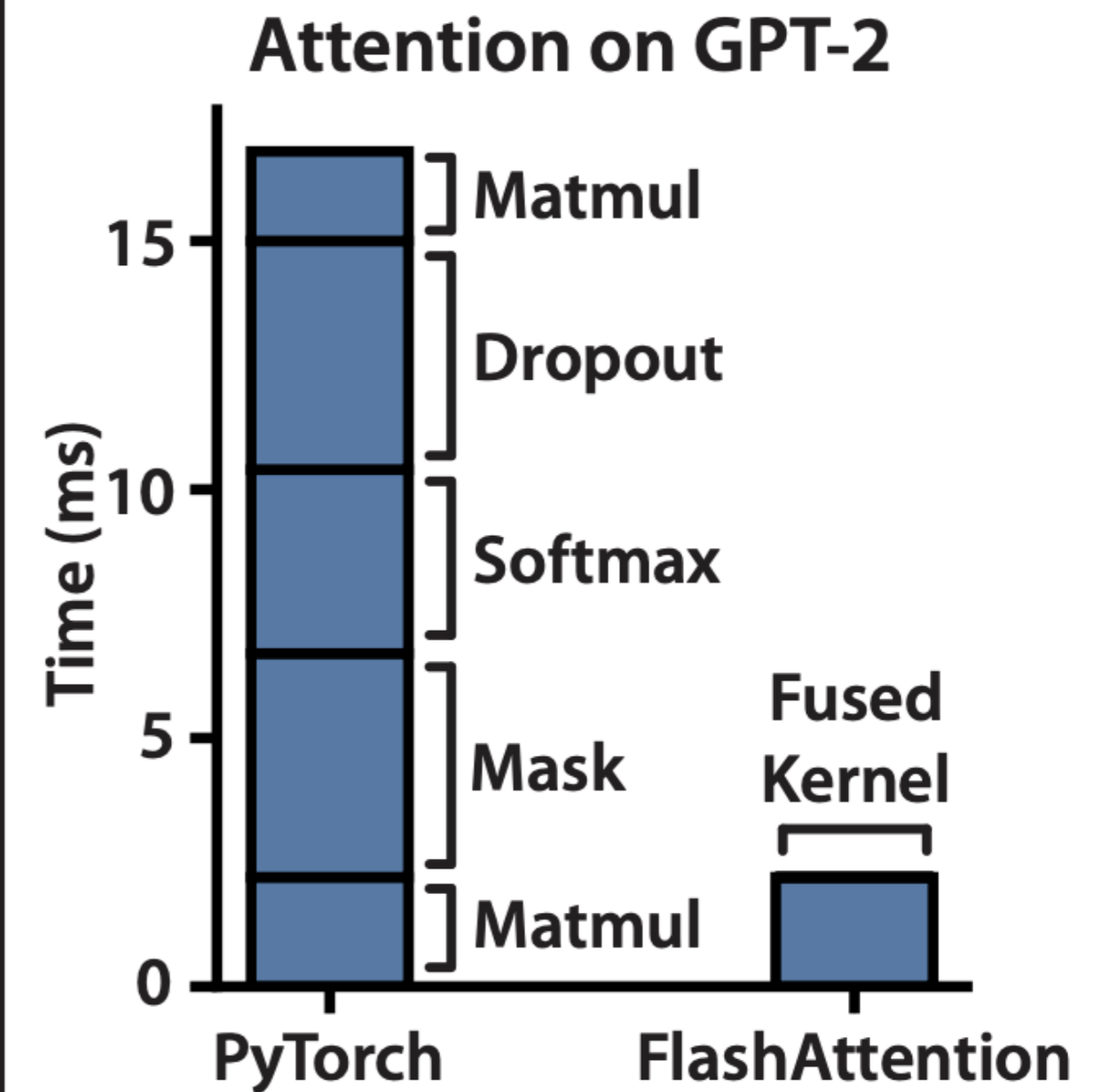
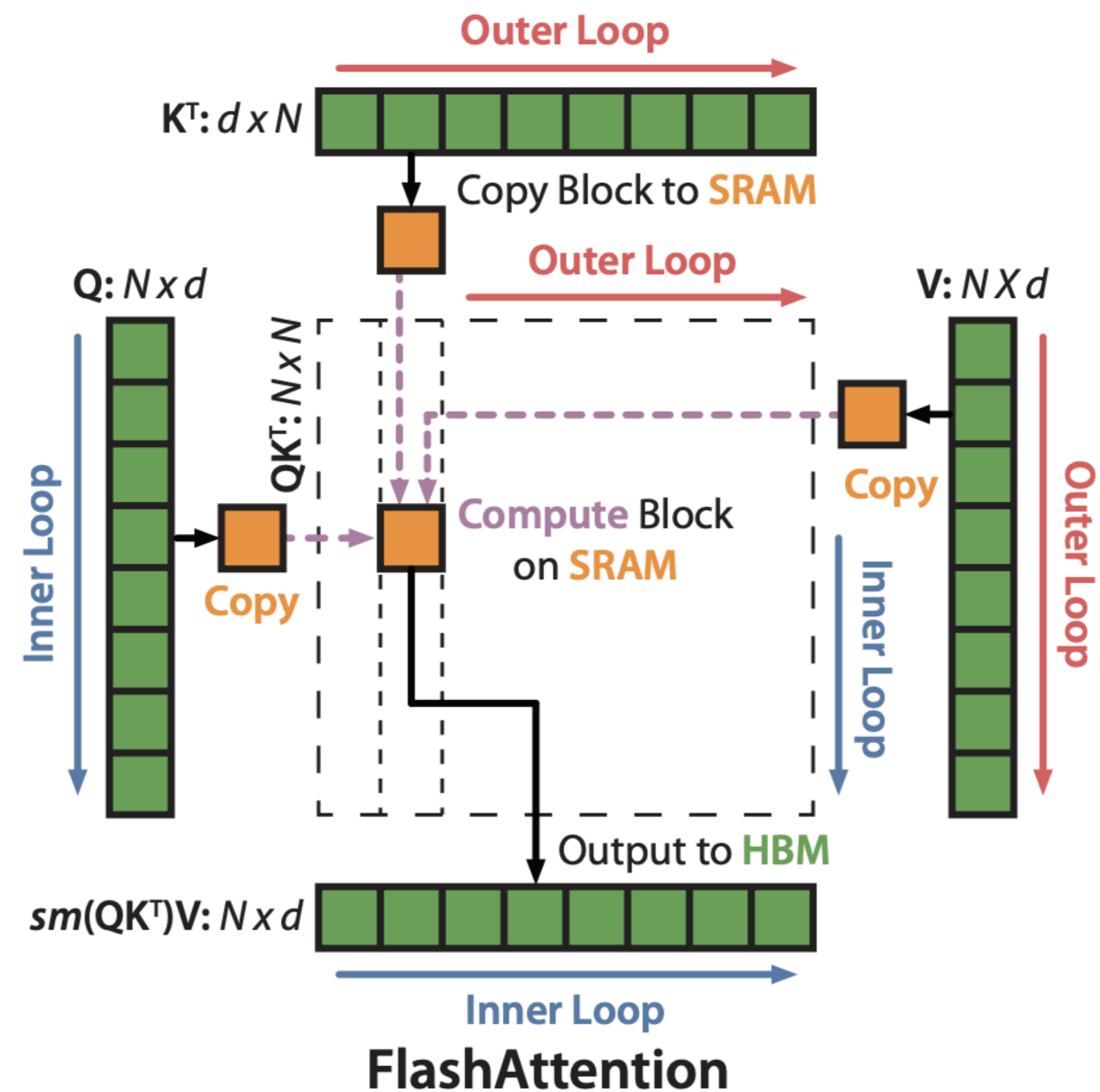
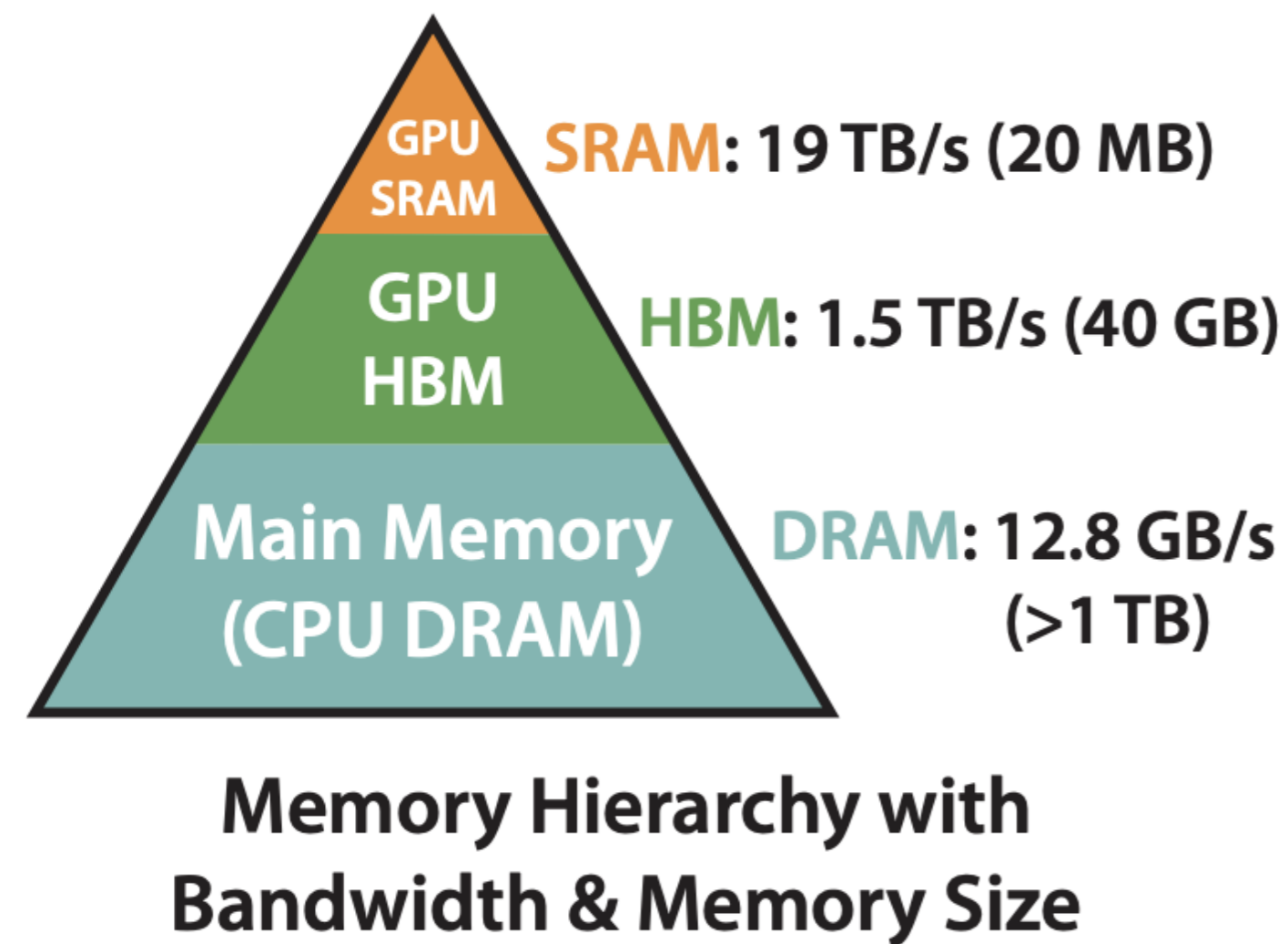
Operation	Cost	Bound
QK^T	$\mathcal{O}(nmd_k)$	Compute-bound
Scaling $\div \sqrt{d_k}$	$\mathcal{O}(nm)$	Memory-bound
Softmax	$\mathcal{O}(nm)$	Memory-bound
$\text{softmax}(\dots)V$	$\mathcal{O}(nmd_v)$	Compute-bound



Memory Hierarchy with Bandwidth & Memory Size



Flash Attention



- ▶ 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

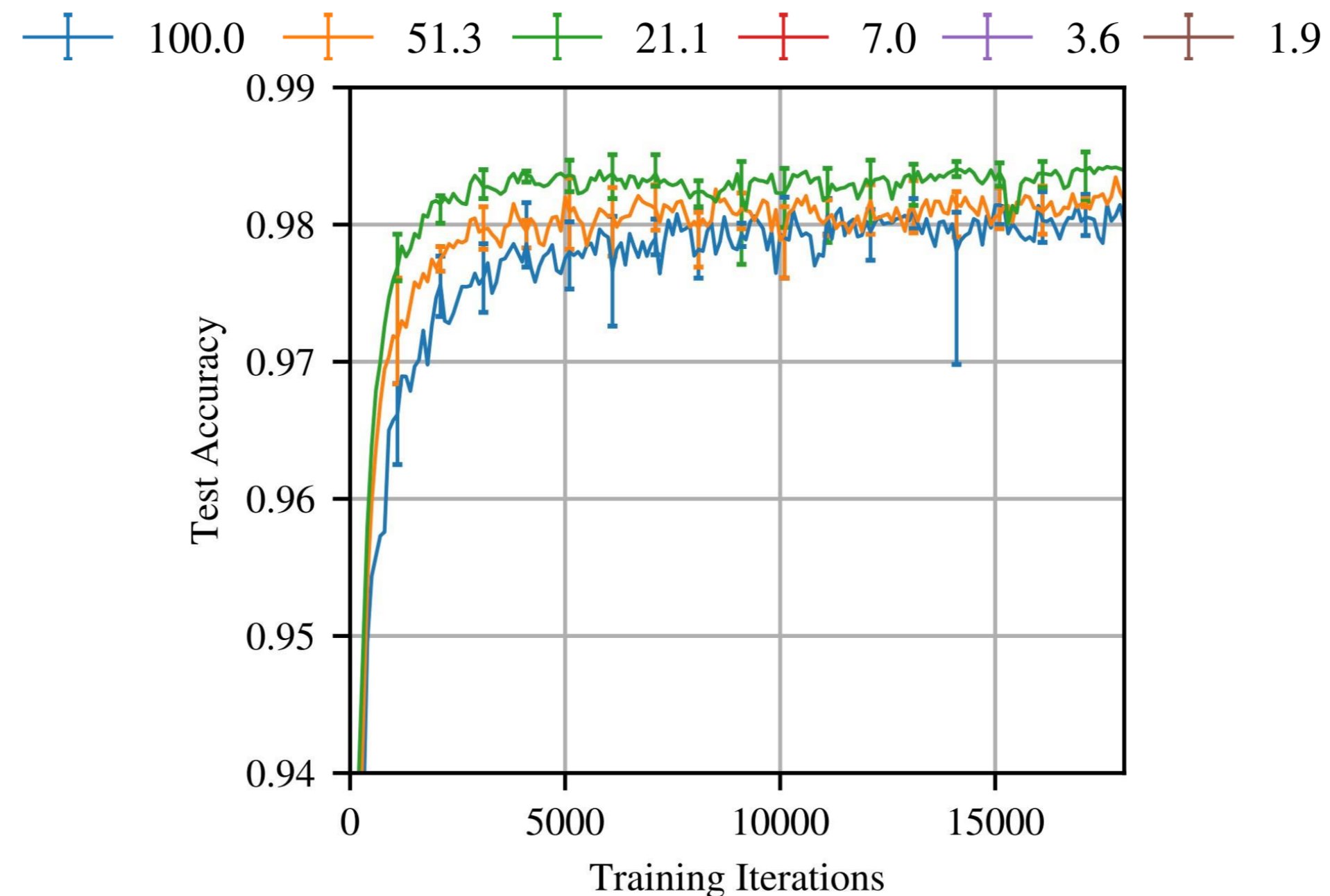
Why train big and then compress?

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 $k \geq \sqrt{2n}$, overparameterization enables local search algorithms to find a *globally* optimal solution for general smooth and convex loss functions. Further, de-

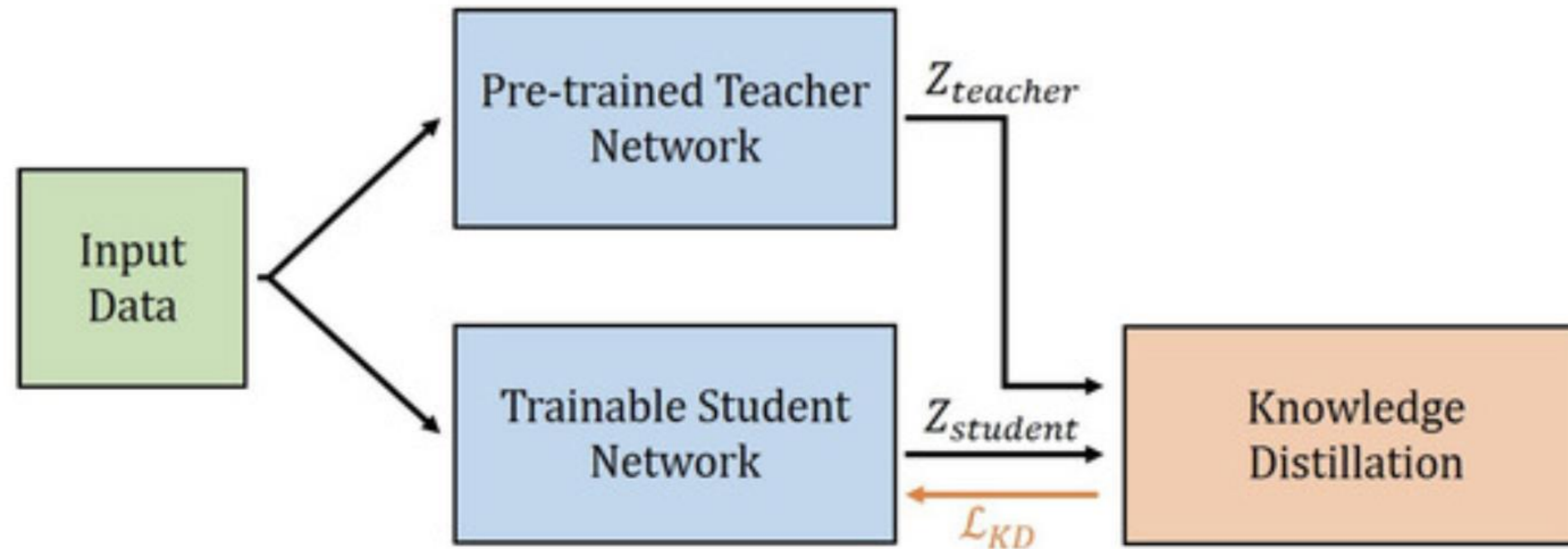
Lottery Ticket Hypothesis

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



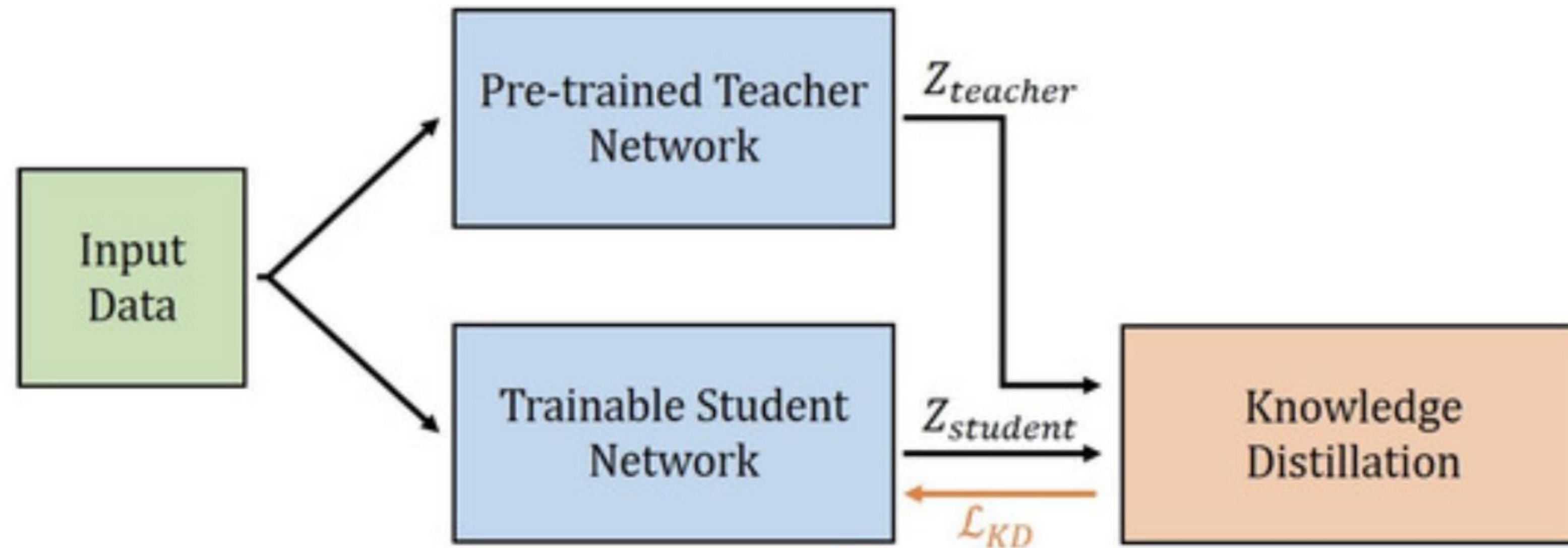
Suppose we have a classification model with output $P_{teacher}(y | \mathbf{x})$

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

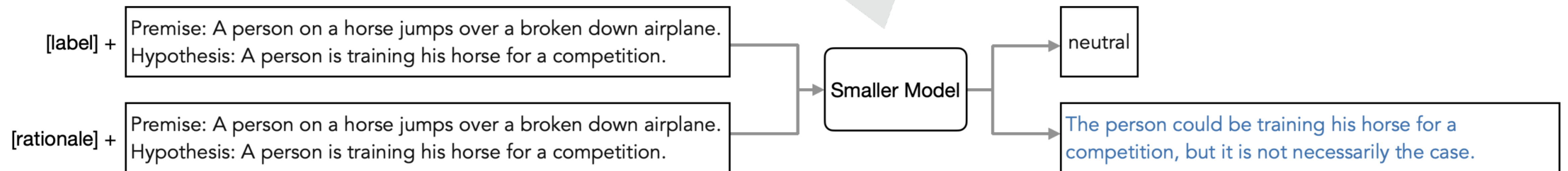
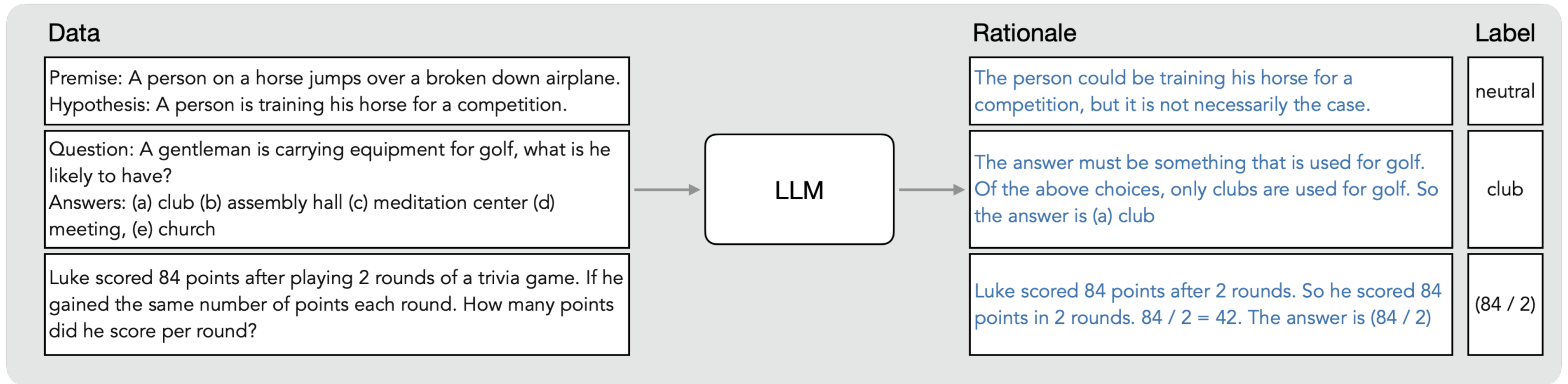
DistilBERT

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

Other Distillation (Synthetic Data Generation)



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 really large models.
- ▶ **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

Multimodality

Transfer between Modalities:

Suppose we directly model
 $p(\text{text, pixels, sound})$
with one big autoregressive transformer.

Pros:

- image generation augmented with world knowledge
- next level text rendering
- native in-context learning
- unified post-training

Cons:

- varying bit-rate
- compute not adaptive

Fixes:

- = model compressed representations
- + compose autoregressive prior with a powerful decoder

tokens \rightarrow [transformer] \rightarrow [diffusion] pixels



A wide image taken with a phone of a glass whiteboard, in a room overlooking the Bay Bridge. The field of view shows a woman writing, sporting a tshirt with a large OpenAI logo. The handwriting looks natural and a bit messy, and we see the photographer's reflection.

...

[Read more](#)



LMs today can process more than just text

A wide image taken with a phone of a glass whiteboard, in a room overlooking the Bay Bridge. The field of view shows a woman writing, sporting a tshirt with a large OpenAI logo. The handwriting looks natural and a bit messy, and we see the photographer's reflection.

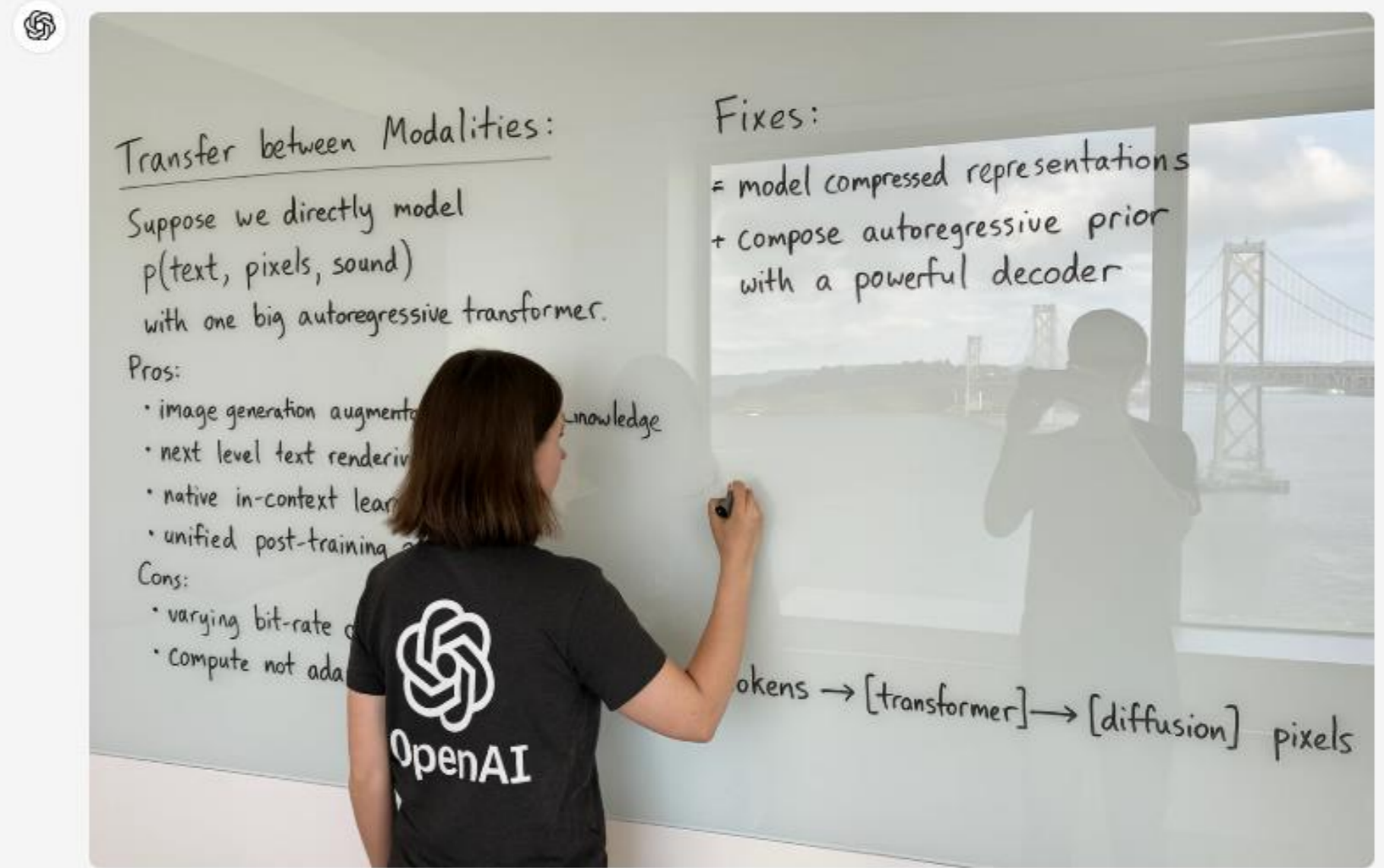
...

[Read more](#)



Best of 8

LMs today can process more than just text



Best of 8

selfie view of the photographer, as she turns around to high five him



Best of 8

[AI or Not game. Is this image real or AI? Can you tell? Take the](#)

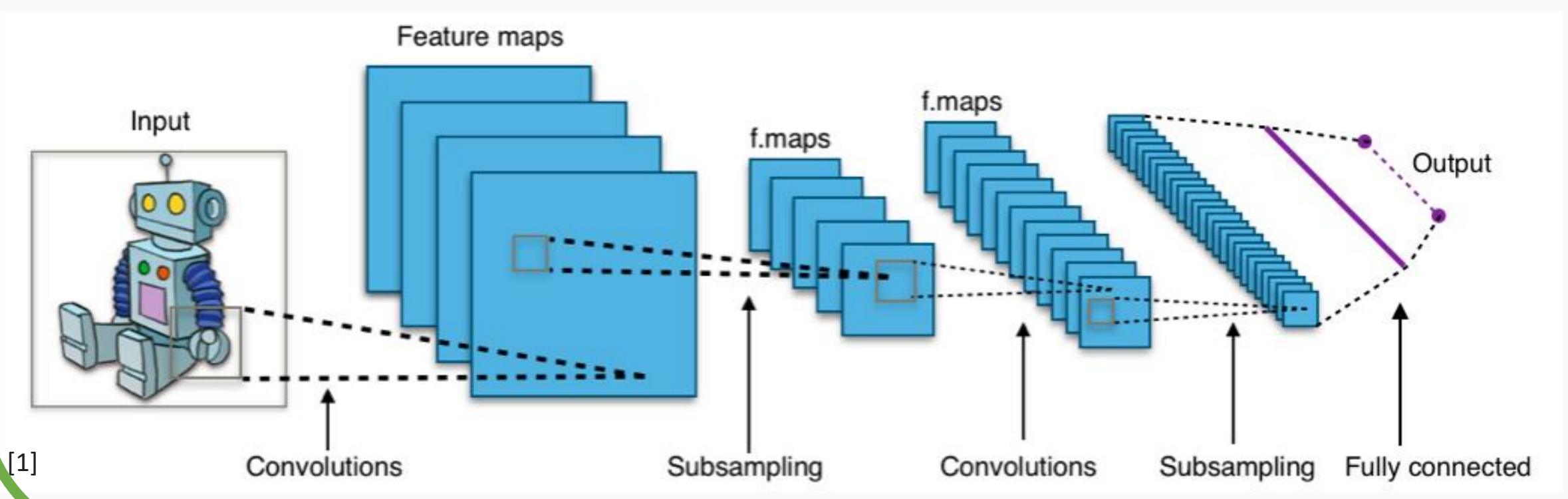
Goals of Today's Lecture

Goal: Learn how some LLMs work with more than just text

- ↳ Motivation for V&L models
- ↳ Vision Transformer
- ↳ Classification with Image+Text Input
- ↳ Generation with Image+Text Input
- ↳ Video Processing (briefly)
- ↳ Speech Processing (briefly)

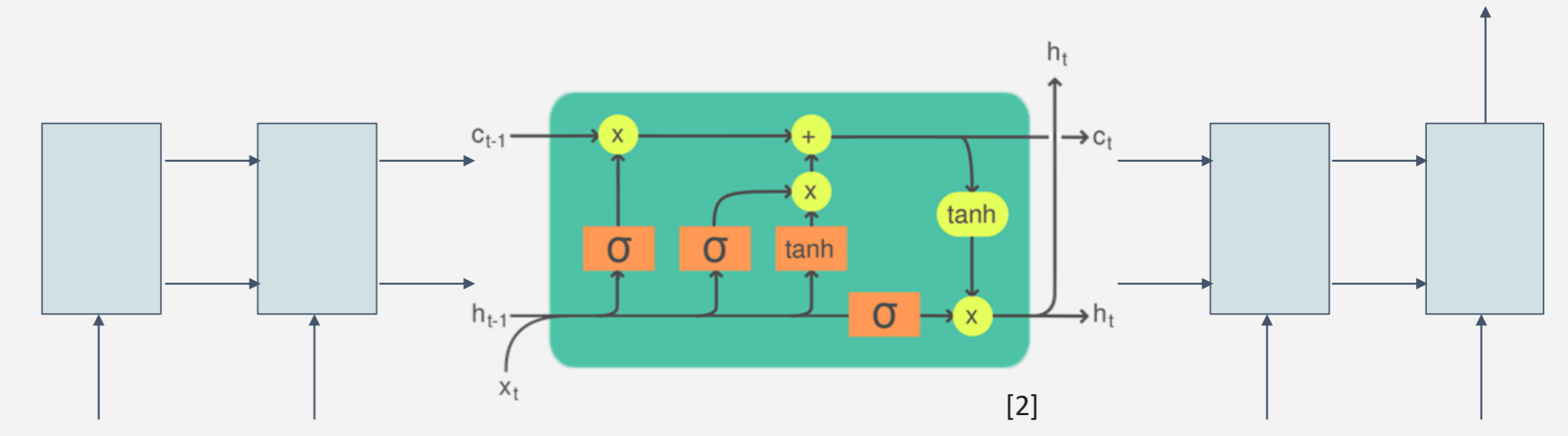
Computer Vision

Convolutional NNs (+ResNets)



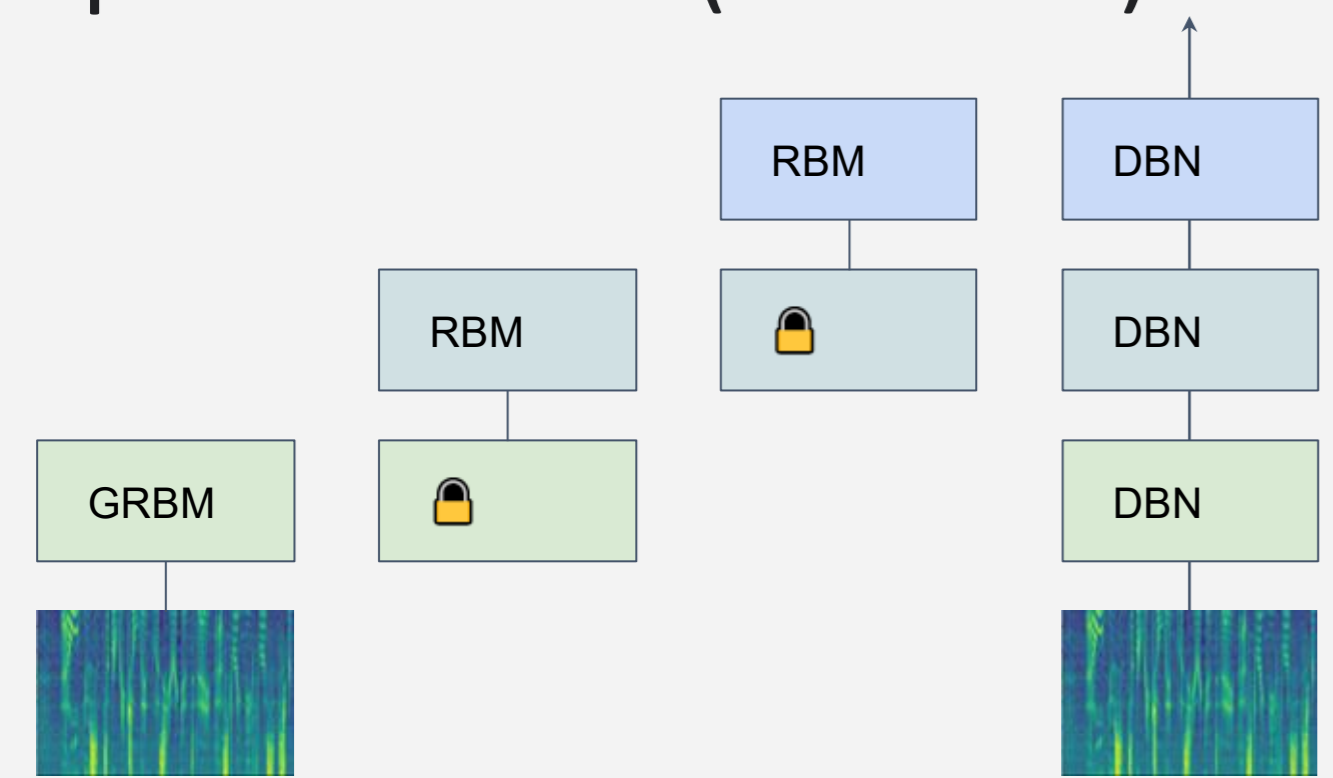
Natural Lang. Proc.

Recurrent NNs (+LSTMs)



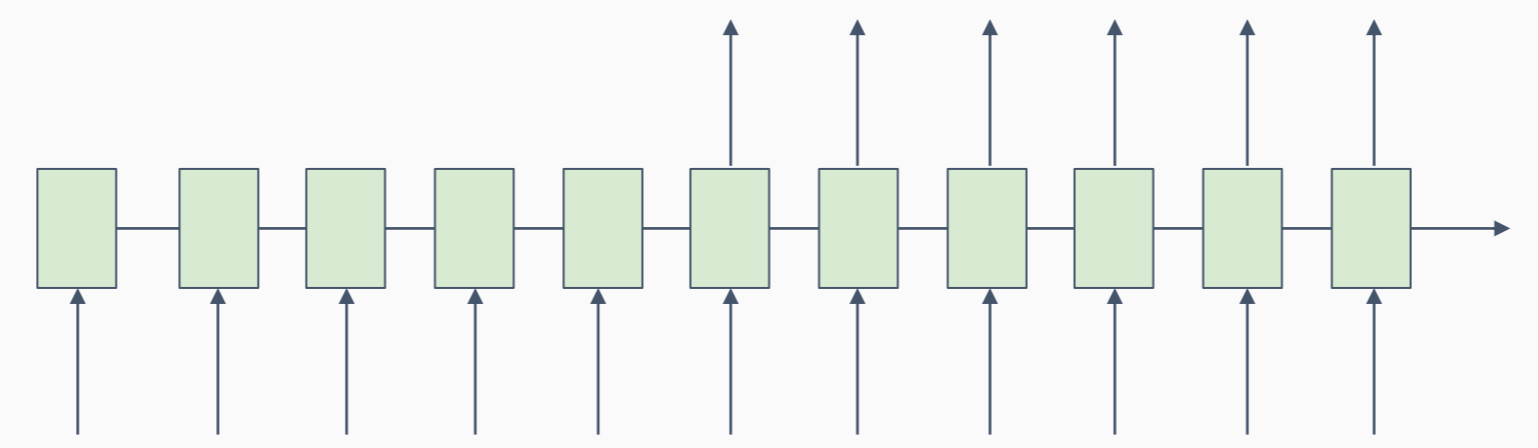
Speech

Deep Belief Nets (+non-DL)



Translation

Seq2Seq



RL

BC/GAIL

Algorithm 1 Generative adversarial imitation learning

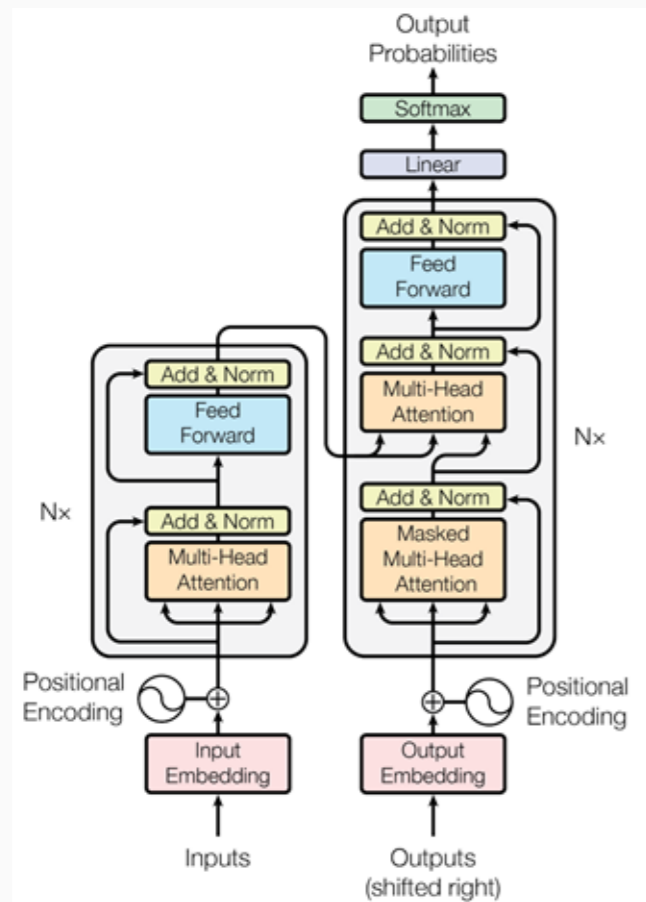
- Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
- for** $i = 0, 1, 2, \dots$ **do**
- Sample trajectories $\tau_i \sim \pi_{\theta_i}$
- Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i} [\nabla_w \log(D_w(s, a))] + \hat{\mathbb{E}}_{\tau_E} [\nabla_w \log(1 - D_w(s, a))] \quad (17)$$
- Take a policy step from θ_i to θ_{i+1} , using the TRPO rule with cost function $\log(D_{w_{i+1}}(s, a))$. Specifically, take a KL-constrained natural gradient step with

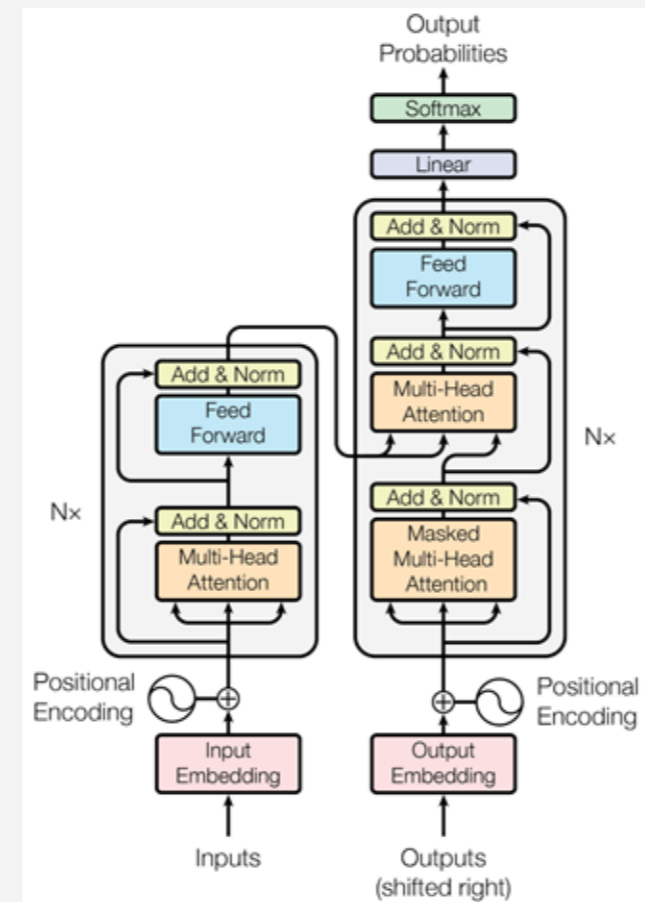
$$\hat{\mathbb{E}}_{\tau_i} [\nabla_{\theta} \log \pi_{\theta}(a|s) Q(s, a)] - \lambda \nabla_{\theta} H(\pi_{\theta}), \quad (18)$$
 where $Q(\bar{s}, \bar{a}) = \hat{\mathbb{E}}_{\tau_i} [\log(D_{w_{i+1}}(s, a)) | s_0 = \bar{s}, a_0 = \bar{a}]$
- end for**

[1] CNN image CC-BY-SA by Aphex34 for Wikipedia https://commons.wikimedia.org/wiki/File:Typical_cnn.png
 [2] RNN image CC-BY-SA by GChe for Wikipedia https://commons.wikimedia.org/wiki/File:The_LSTM_Cell.svg

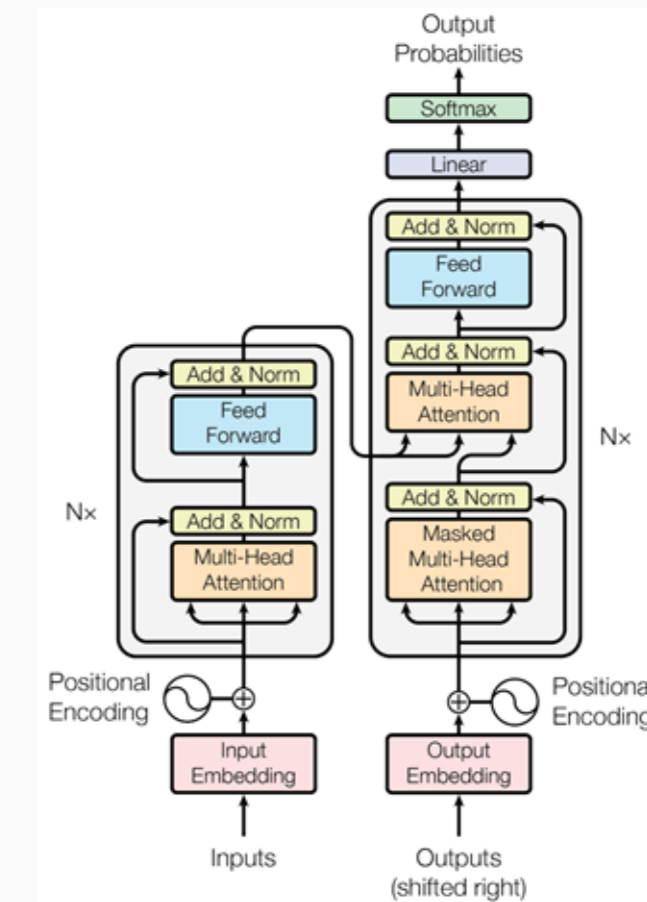
Computer Vision



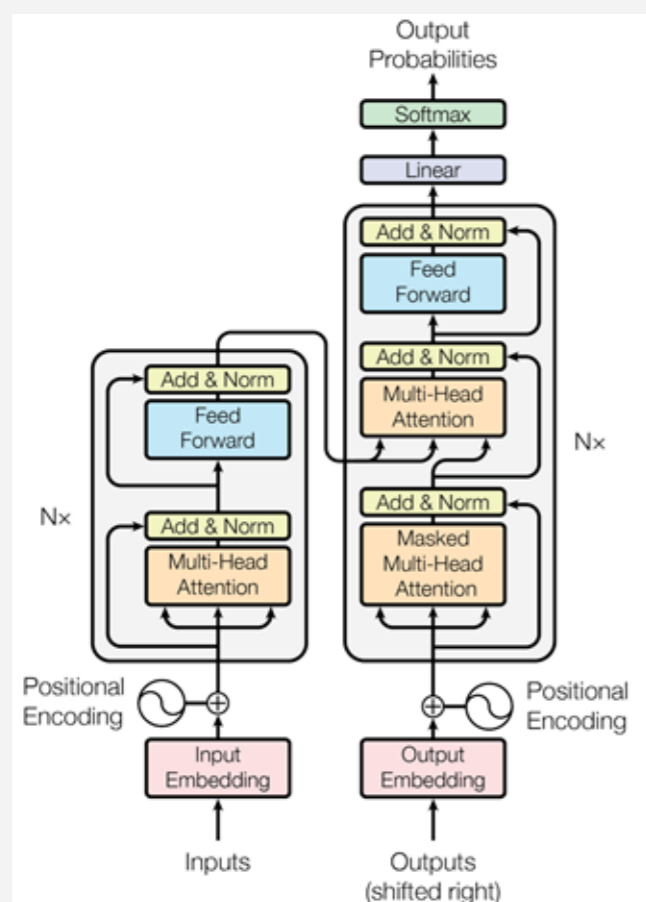
Natural Lang. Proc.



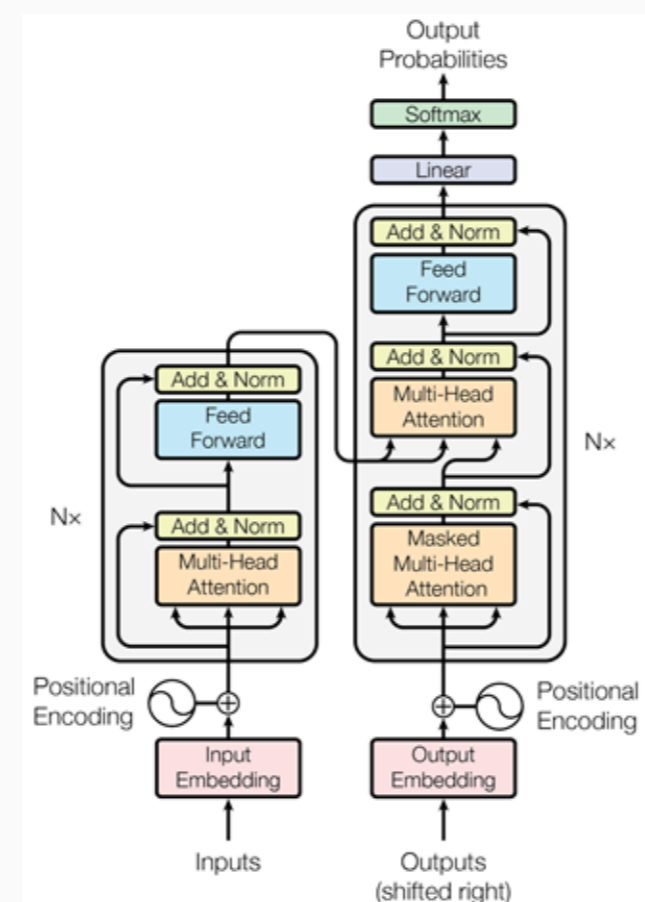
Reinf. Learning



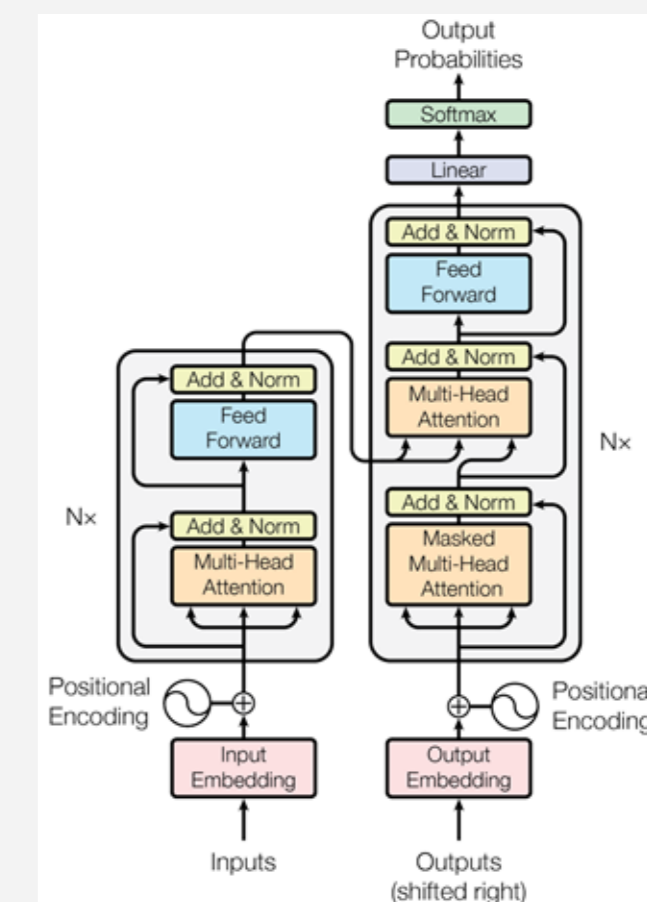
Speech



Translation



Graphs/Science



Vision Transformer (ViT)



[An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale](#)

[Tutorial 11: Vision Transformers](#)

Figure: <https://github.com/lucidrains/vit-pytorch/blob/main/images/vit.gif>

An example of turning a 32x32 image into a sequence of 64 patches of size 4x4



ViT – Input embeddings

$x \in \mathbb{R}^{H \times W \times C}$... image

H ... the number of rows of pixels in the image

W ... the number of columns of pixels in the image

(H, W) ... the resolution of the image

C ... the number of channels (3 for the RGB format)

$p \in \mathbb{R}^{H_p \times W_p \times C}$... an image patch

(H_p, W_p) ... the resolution of the patch

Flatten the patch by going through all the pixels of the patch row by row



ViT – Input embeddings (cont.)

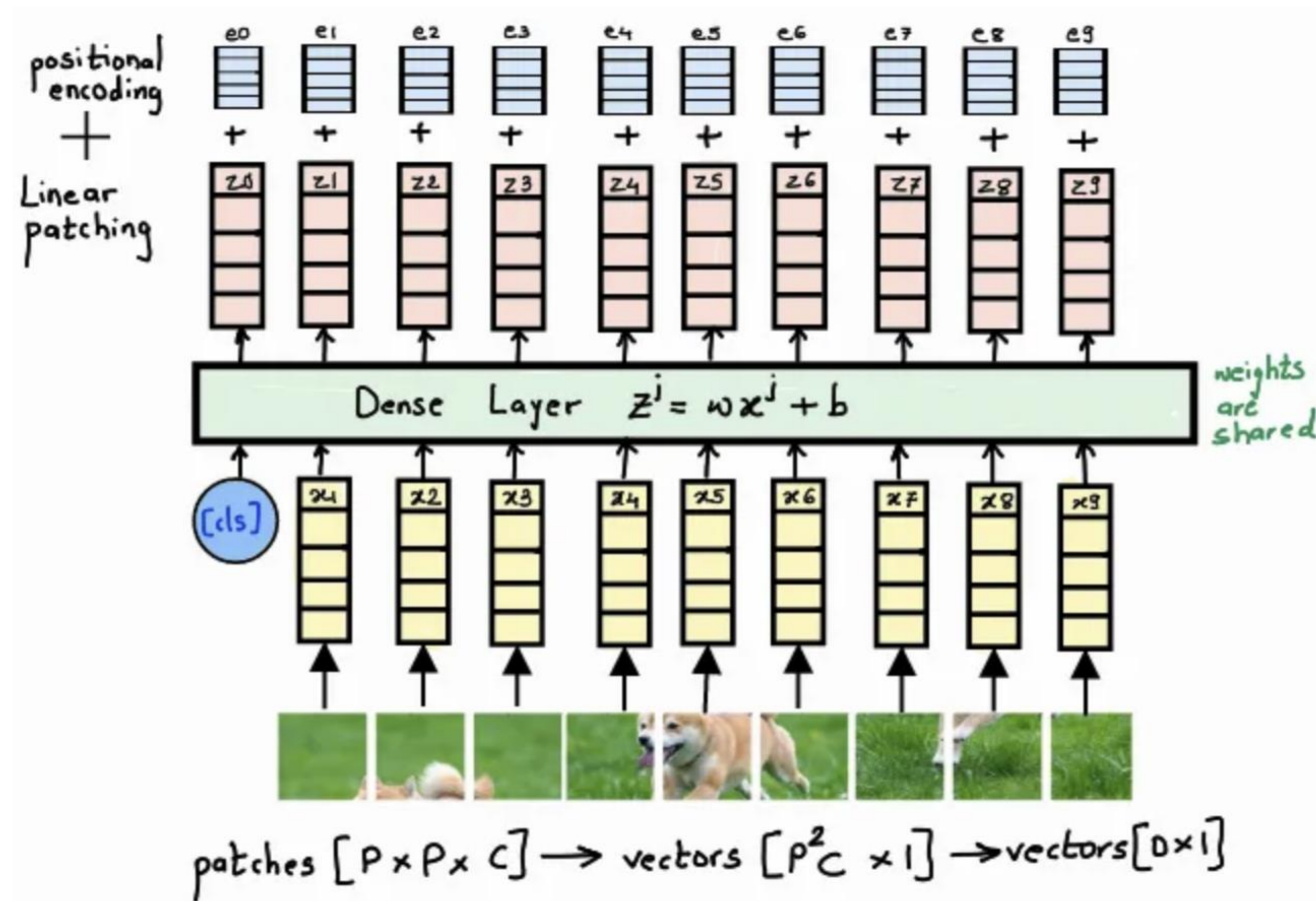
$f(p) \in \mathbb{R}^{H_p \cdot W_p \cdot C}$... a flattened image patch

$v(f(p)) = W f(p) \in \mathbb{R}^D$

$W \in \mathbb{R}^{D \times (H_p \cdot W_p \cdot C)}$... linear transformation

$v_i^{\text{pos}} \in \mathbb{R}^D$... a positional embedding for the i -th position

$v_i = f(p_i) + v_i^{\text{pos}}$... the input embedding of the i -th patch p_i



Vision Transformer tutorial:

https://lightning.ai/docs/pytorch/latest/notebooks/course_UvA-DL/11-vision-transformer.html

Vision Transformer (ViT)

We learned that pretraining helps!



[An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale](#)

[Tutorial 11: Vision Transformers](#)

Figure: <https://github.com/lucidrains/vit-pytorch/blob/main/images/vit.gif>

Grounding in Images

- ▶ How would you describe this image?
- ▶ What does the word "*spoon*" evoke?



the girl is licking the spoon of batter

Grounding Spoon



Winco 0005-03 7
3/8" Dinner Spoon...

\$7.16



wikiHow
How to Hold a Spoon: 13 Steps (...)



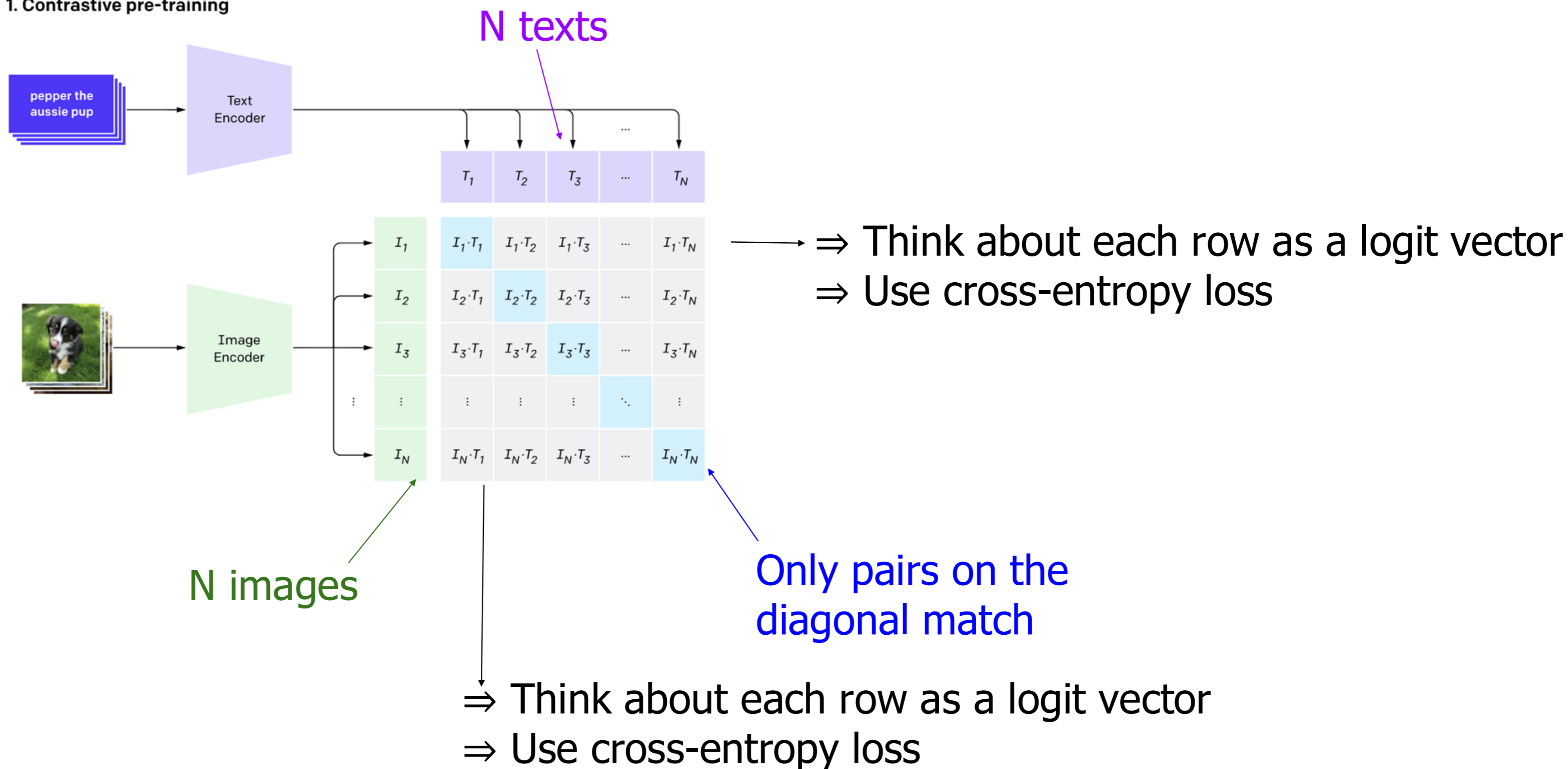
GO Indiegogo
Spoon that Elevates Taste ...

CLIP

[Radford et al., 2021]; [Conference presentation](#)

– Contrastive pretraining

1. Contrastive pre-training



CLIP [Radford et al., 2021]; [Conference presentation](#)

– Contrastive pretraining pseudocode

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t            - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

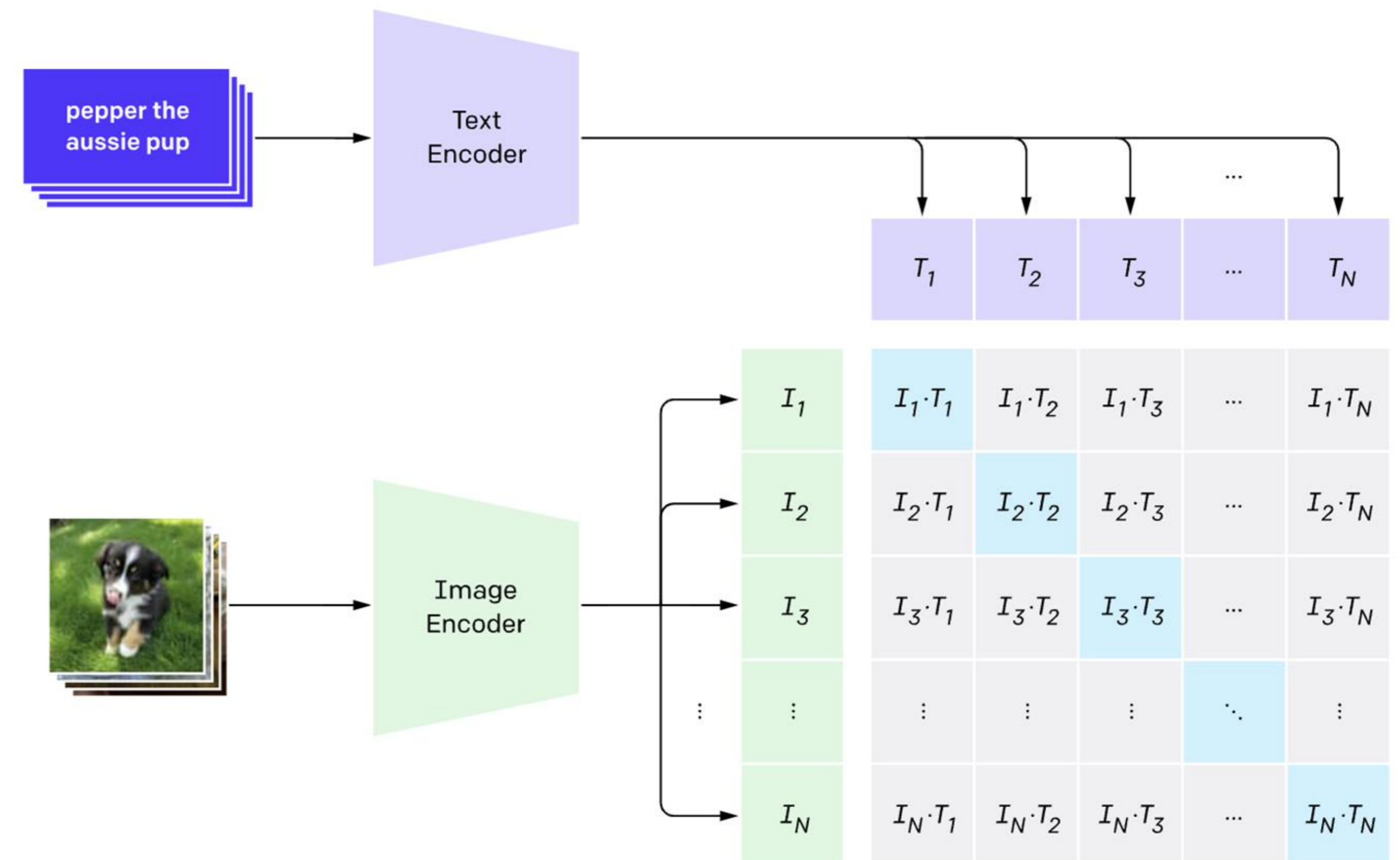
# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

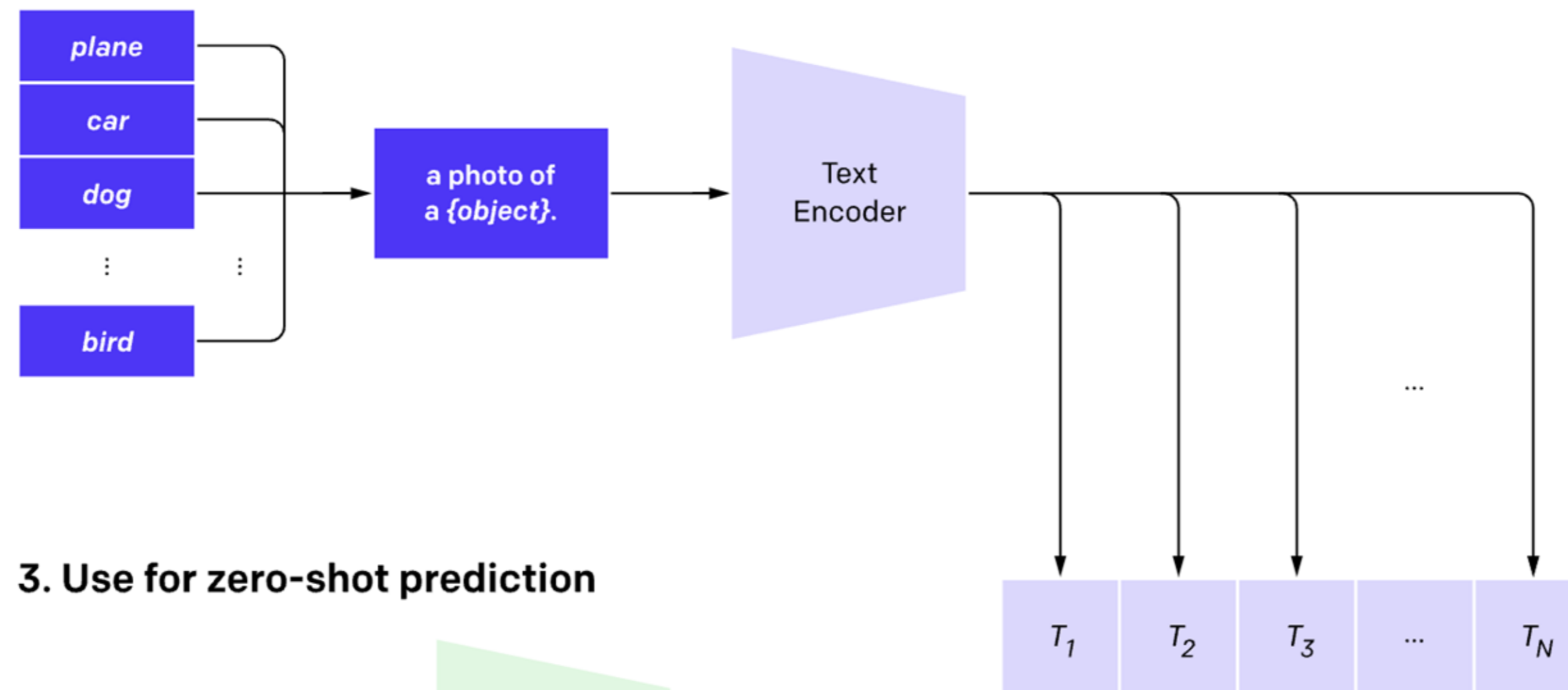
1. Contrastive pre-training



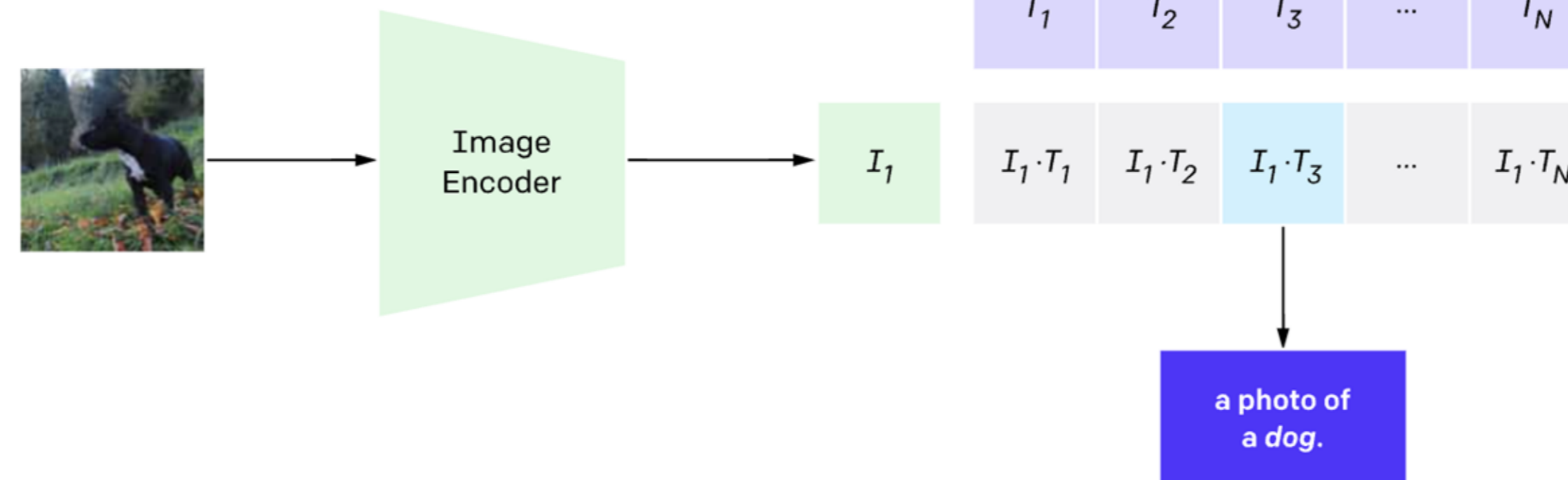
CLIP [Radford et al., 2021]; [Conference presentation](#)

– *Image classification*

2. Create dataset classifier from label text



3. Use for zero-shot prediction



CLIP

[Radford et al., 2021]; [Conference presentation](#)

Original repository, zero-shot prediction:

<https://github.com/openai/CLIP#zero-shot-prediction>

In 🤗 ecosystem:

https://huggingface.co/docs/transformers/model_doc/clip

Independently trained and larger CLIP:

https://github.com/mlfoundations/open_clip

Goals of Today's Lecture

Goal: Learn how some LLMs that take more than just text

- ↳ Motivation for V&L models
- ↳ Vision Transformer
- ↳ **Classification with Image+Text Input**
- ↳ Generation with Image+Text Input
- ↳ Video Processing
- ↳ Speech Processing

Multimodal Classification



Q: What endangered animal is featured on the truck?

- A: **A bald eagle.**
- A: A sparrow.
- A: A humming bird.
- A: A raven.



Q: Where will the driver go if turning right?

- A: **Onto 24 3/4 Rd.**
- A: Onto 25 3/4 Rd.
- A: Onto 23 3/4 Rd.
- A: Onto Main Street.



Q: When was the picture taken?

- A: **During a wedding.**
- A: During a bar mitzvah.
- A: During a funeral.
- A: During a Sunday church service.



Q: Who is under the umbrella?

- A: **Two women.**
- A: A child.
- A: An old man.
- A: A husband and a wife.



Q: Why was the hand of the woman over the left shoulder of the man?

- A: **They were together and engaging in affection.**
- A: The woman was trying to get the man's attention.
- A: The woman was trying to scare the man.
- A: The woman was holding on to the man for balance.




Q: How many magnets are on the bottom of the fridge?

- A: **5.**
- A: 2.
- A: 3.
- A: 4.

An example of multimodal tasks



An example of multimodal tasks



Can you please pass the cow?

Task 1 Match the Caption + Cartoon
✗ I'd kill for some cream cheese.
vs.
✓ Can you please pass the cow?

Task 2 Rank the Finalist
✗ Welcome to Insomniacs Anonymous
vs.
🏆 Can you please pass the cow?

Task 3 Explanation Generation

Human-authored:
When drinking coffee or tea, people often add cream, and may ask others to pass it if it's on the other side of a table. But here, the mugs are huge, so instead of asking for a small cup of cream, they are asking for the entire cow, which is the appropriately-sized cream dispenser for these huge drinks.

From Pixels (OFA + T5-11B):
The joke is that the meeting participants are actually sitting on chairs made out of coffee mugs, which is an unlikely location for the discussion. Instead of asking for another mug of coffee, the person at the head of the table simply asks for "the cow", or a coffee machine.

From Description (5-shot GPT 3.5):
"Pass the cow" is an example of a non sequitur, something that looks like a logical thing to say, but doesn't make sense in context. The humor in this cartoon comes from the large size of the coffee mugs: they are so large that they resemble buckets rather than mugs, thus making the request to "pass the cow" almost reasonable.

Goals of Today's Lecture

Goal: Learn how some LLMs that take more than just text

- ↳ Motivation for V&L models
- ↳ Vision Transformer
- ↳ Classification with Image+Text Input
- ↳ **Generation with Image+Text Input**
- ↳ Video Processing
- ↳ Speech Processing

Not constrained to classification

User What is funny about this image? Describe it panel by panel.



Source: [hmmm \(Reddit\)](#)

GPT-4o: Not constrained to classification

er What is funny about this image? Describ

GPT-4

The image shows a package for a "Lightning Cable" adapter with three panels.

Panel 1: A smartphone with a VGA connector (a large, blue, 15-pin connector typically used for computer monitors) plugged into its charging port.

Panel 2: The package for the "Lightning Cable" adapter with a picture of a VGA connector on it.

Panel 3: A close-up of the VGA connector with a small Lightning connector (used for charging iPhones and other Apple devices) at the end.

The humor in this image comes from the absurdity of plugging a large, outdated VGA connector into a small, modern smartphone charging port.



Source: [hmmm \(Reddit\)](#)

Four components of a simple and standard design of combining a language model with a vision encoder

Image encoder:

- ↳ *Image preprocessing*: Turn an image into a sequence of patches
- ↳ A *pretrained* Vision Transformer image encoder that first maps each of image patches into input embeddings, then transforms them using many self-attention and FF/MLP layers

Cross modal connector

- ↳ A connector that projects the vision embeddings (from e.g. final layer) to the language model's input dimension with an FFNN/MLP
- ↳ Initially randomly initialized

A *pretrained* decoder-only Transformer LLM

- ↳ Prepend projected vision embeddings to the token embeddings

LLaVA: Visual Instruction Tuning

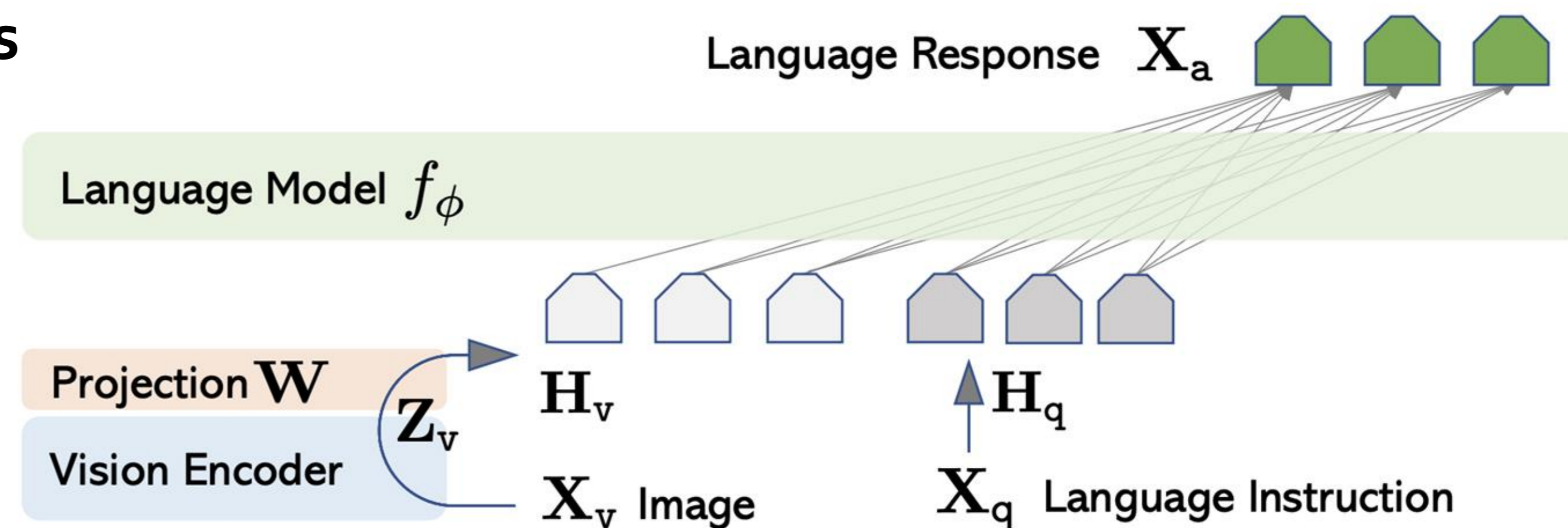
<https://lava-vl.github.io/>

Strong pretrained vision and language models

- Vision encoder: CLIP-ViT-L/14
- Language model: LLaMA-2, etc.

Cross modal connector

- Linear projection



Tuning the model for following multimodal instructions

Figure 1: LLaVA network architecture.

- Use image captions from available datasets
- Prompt text-only GPT-4 to generate (instruction, output) pairs
- 158K instructions

First tune only the projection, then tune the projection and LM

Category	Model	VLM		LLM Backbone		Vision Encoder	
		Open Weights	Open Data + Code	Open Weights	Open Data + Code	Open Weights	Open Data + Code
Molmo	Molmo-72B	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-D	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-O	Open	Open	Open	Open	Open	Closed
	MolmoE-1B	Open	Open	Open	Open	Open	Closed
API Models	GPT-4o	Closed	Closed	Closed	Closed	Closed	Closed
	GPT-4V	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Pro	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Flash	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3.5 Sonnet	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Opus	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Haiku	Closed	Closed	Closed	Closed	Closed	Closed
Open Weights	Qwen VL2 72B	Open	Closed	Open	Closed	Open	Closed
	Qwen VL2 7B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 LLAMA 76B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 8B	Open	Closed	Open	Closed	Open	Closed
	Pixtral 12B	Open	Closed	Open	Closed	Open	Closed
	Phi3.5-Vision 4B	Open	Closed	Open	Closed	Open	Closed
	PaliGemma 3B	Open	Closed	Open	Closed	Open	Closed
Open Weights & Data	LLAVA OneVision 72B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA OneVision 7B	Open	Distilled	Open	Closed	Open	Closed
	Cambrian-1 34B	Open	Distilled	Open	Closed	Open	Closed
	Cambrian-1 8B	Open	Distilled	Open	Closed	Open	Closed
	xGen - MM - Interleave 4B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA-1.5 13B	Open	Open	Open	Closed	Open	Closed
	LLAVA-1.5 7B	Open	Open	Open	Closed	Open	Closed

Molmo [\[Deitke et al., 2024\]](#)

Image encoder: OpenAI's ViT-L/14 336px CLIP model

- It can be reproduced from scratch as shown by MetaCLIP, but is trained for high resolution images

Cross modal connector

- Linear projection

Language model: Fully open OLMo-7B-1024, fully open OLMoE-1B-7B, open-weight Qwen2 7B, or open-weight Qwen2 72B

Pretraining: Caption generation using the new PixMo-Cap dataset

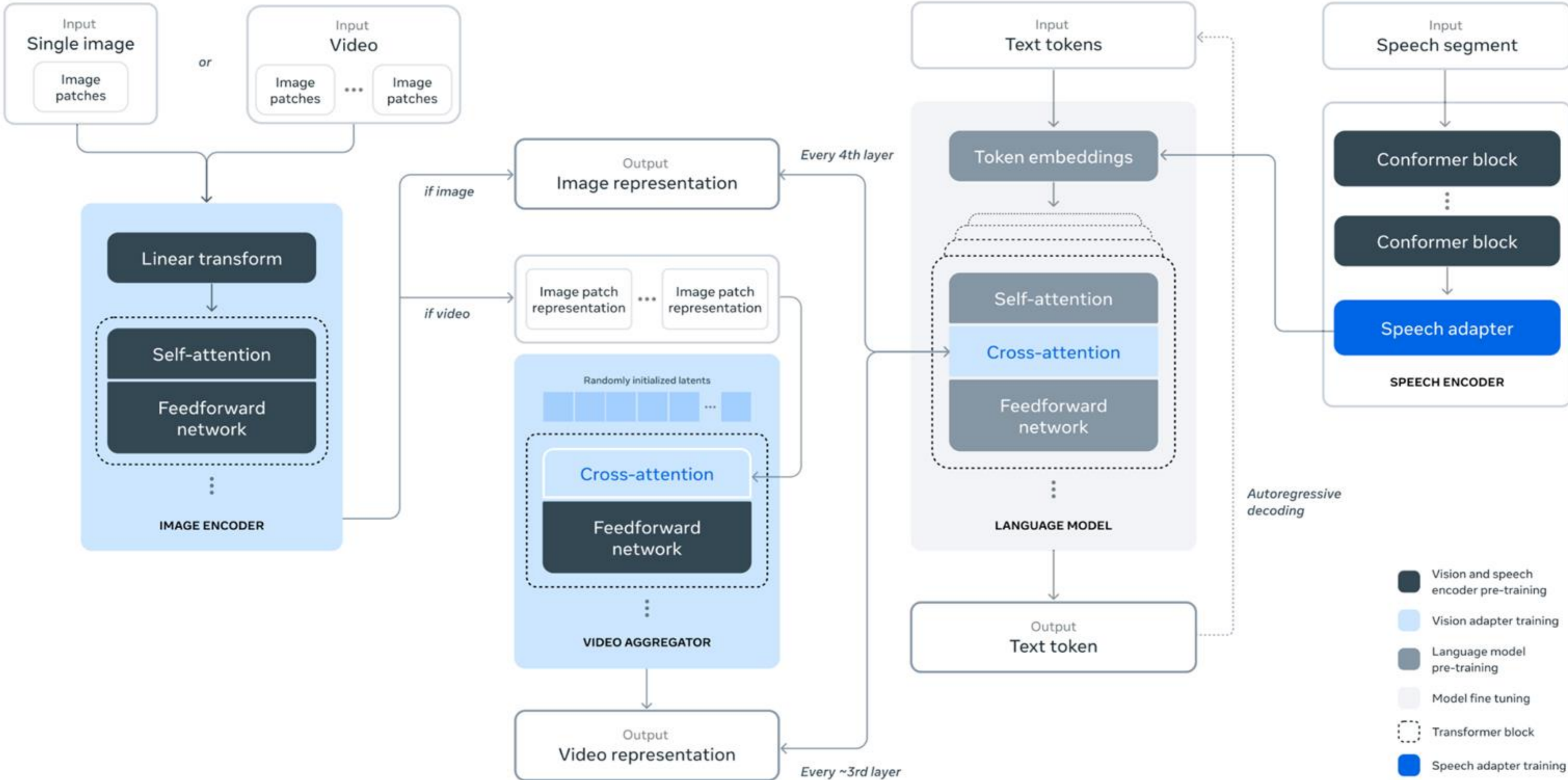
Instruction finetuning: PixMo-AskModelAnything, PixMo-Points, PixMo-CapQA, PixMo-Docs, PixMo-Clocks + Academic datasets

Goals of Today's Lecture

Goal: Learn how some LLMs that take more than just text

- ↳ Motivation for V&L models
- ↳ Vision Transformer
- ↳ Classification with Image+Text Input
- ↳ Generation with Image+Text Input
- ↳ **Video Processing**
- ↳ Speech Processing

Llama 3.2



Llama 3.2 (cont.)

Image encoder:

- Vision Transformer pretrained from scratch
- 224 x 224 resolution; 14 x 14 patches
- The size of patch embeddings = 7680
- Features from the 4th, 8th, 16th, 24th and 31st layers are also provided in addition to the final layer features

Cross modal connector:

- Cross-attention
- Introduce substantial numbers of additional trainable parameters into the model:
for Llama 3 405B, the cross-attention layers have $\approx 100\text{B}$ parameters

Language model: Llama 3.1

Llama 3.2 – Video processing

Llama 3.2 takes as input up to 64 uniformly sampled frames from a full video

Each frame is processed by the image encoder

Temporal structure in videos through two components:

1. Encoded video frames are aggregated by a temporal aggregator which merges 32 consecutive frames into one
 - a. Temporal aggregator = Perceiver resampler [[Jaegle et al., 2021](#)]
2. Extra video cross attention layers are added before every 4th image cross attention layer

Goals of Today's Lecture

Goal: Learn how some LLMs that take more than just text

- ↳ Motivation for V&L models
- ↳ Vision Transformer
- ↳ Classification with Image+Text Input
- ↳ Generation with Image+Text Input
- ↳ Video Processing
- ↳ **Speech Processing**

We didn't cover speech in class...

Analog signal

Goal: Raw wavefile \Rightarrow Sequences of log mel spectrum vectors

Raw wavefile contains info about changes in air pressure caused the specific way that air passes through the glottis [the middle region inside your voice box that contains your vocal cords] & out the oral or nasal cavities

The graph measures the amount of **compression** or **rarefaction** (uncompression) of the air molecules

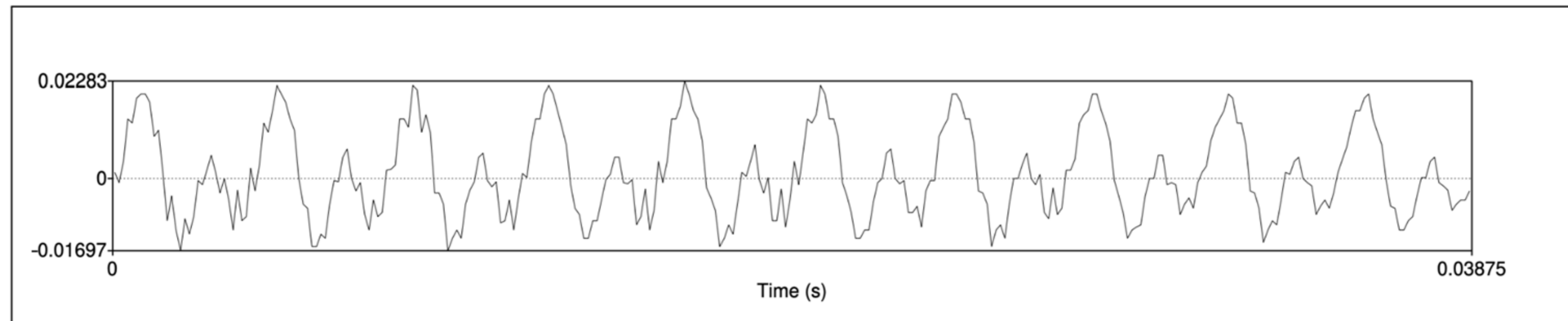


Figure 16.2 A waveform of an instance of the vowel [iy] (the last vowel in the word “baby”). The y-axis shows the level of air pressure above and below normal atmospheric pressure. The x-axis shows time. Notice that the wave repeats regularly.

Sampling and Quantization

Next steps: Transform a waveform, a 2D plot of air pressure changes (y-axis) over time (x-axis) into a sequence of 80-dimensional log Mel spectrum vectors

Sampling:

- Turn a waveform into a sequence of amplitude values [loudness] sampled at regular intervals (e.g., 16 kHz)
- Sampling rate: Number of samples/sec (e.g., 16 kHz for high-quality audio)
- Creates a 1D array of sampled amplitudes

Quantization:

- Digital systems work with discrete values rather than continuous ones
- Represents amplitudes as integers (e.g., 8-bit or 16-bit)
- Reduces continuous signal values into discrete levels

Windowing

Speech analyzed in small stationary windows

- Assumption: within small time windows, the properties of a speech signal (such as its frequency content) remain relatively constant

Key parameters:

- Window size (e.g., 25 ms): The duration of the analyzed segment
- Frame stride (e.g., 10 ms): The interval at which consecutive windows are started \Rightarrow overlapping analysis allowed

Window types:

- Rectangular: Abrupt cutoff at edges
- Hamming: Smooth tapering at edges

Windowing results in a 2D array where each row corresponds to the samples in a window

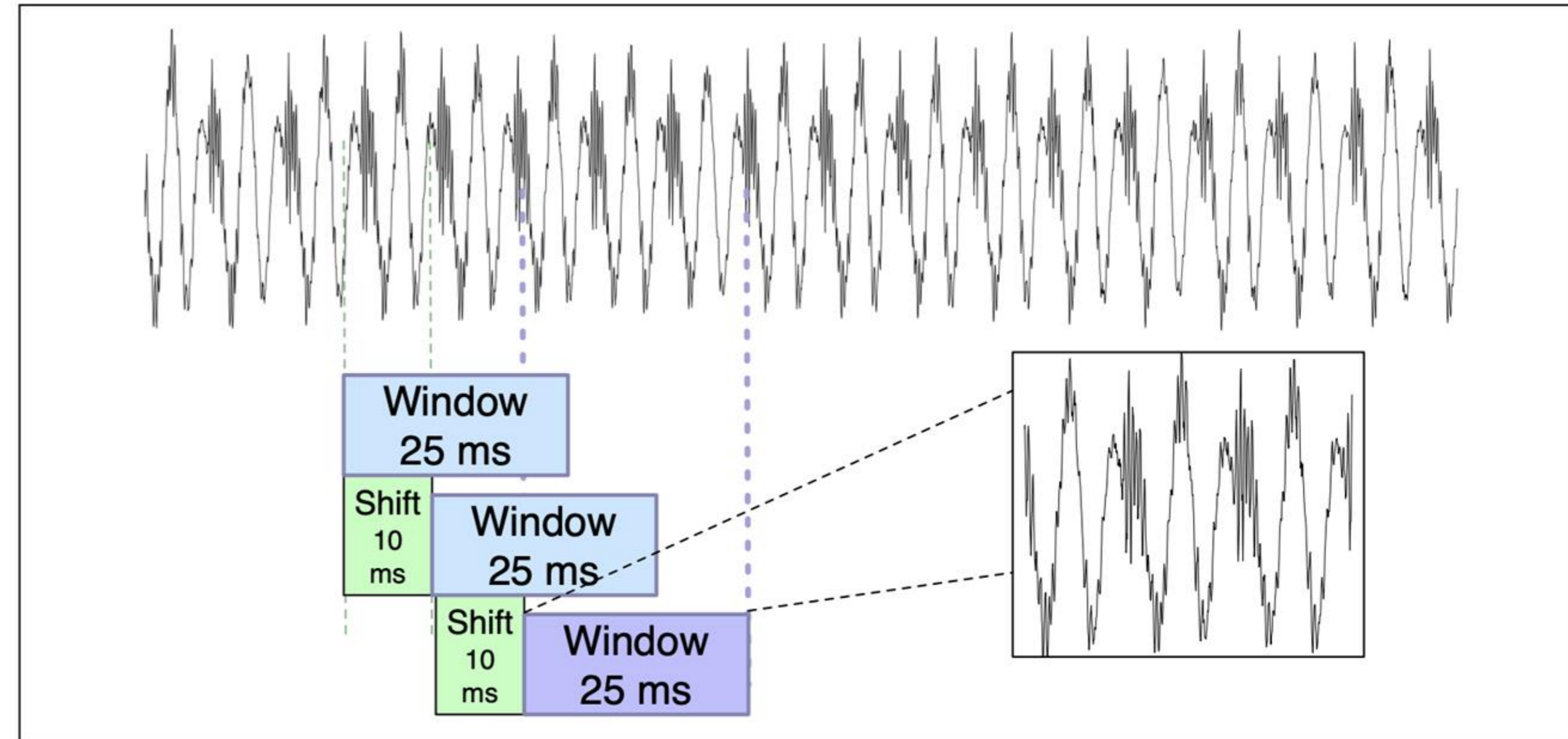


Figure 16.4 Windowing, showing a 25 ms rectangular window with a 10ms stride.

[[Jurafsky & Martin Section 16.2](#)]

Discrete Fourier Transform (DFT)

Next: Analyze the signal in the frequency domain rather than the time domain

A signal contains energy distributed across various frequencies

Spectral information: The breakdown of how much energy (or power) is present at each frequency band

Fast Fourier Transform (FFT): Efficient computation of DFT for signal analysis

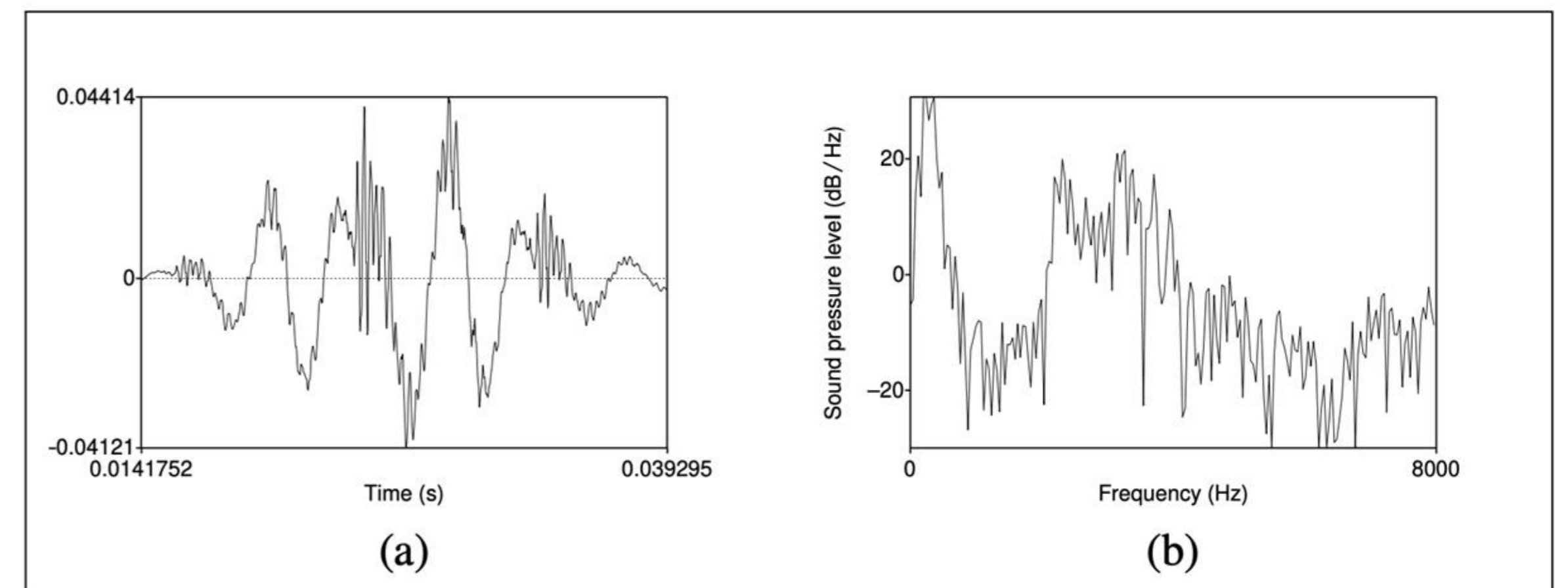


Figure 16.6 (a) A 25 ms Hamming-windowed portion of a signal from the vowel [iy] and (b) its spectrum computed by a DFT.

Mel Filter Bank

The results of the FFT tell us the energy at each frequency band

Human hearing is not equally sensitive at all frequency bands; it is less sensitive at higher frequencies

- This bias toward low frequencies helps human recognition, since information in low frequencies is crucial for distinguishing vowels or nasals, while information in high frequencies is less crucial for successful recognition

Mel is a unit of pitch [the degree of highness or lowness of a tone] \Rightarrow Convert frequency to Mel scale

$$mel(f) = 1127 \ln\left(1 + \frac{f}{700}\right)$$

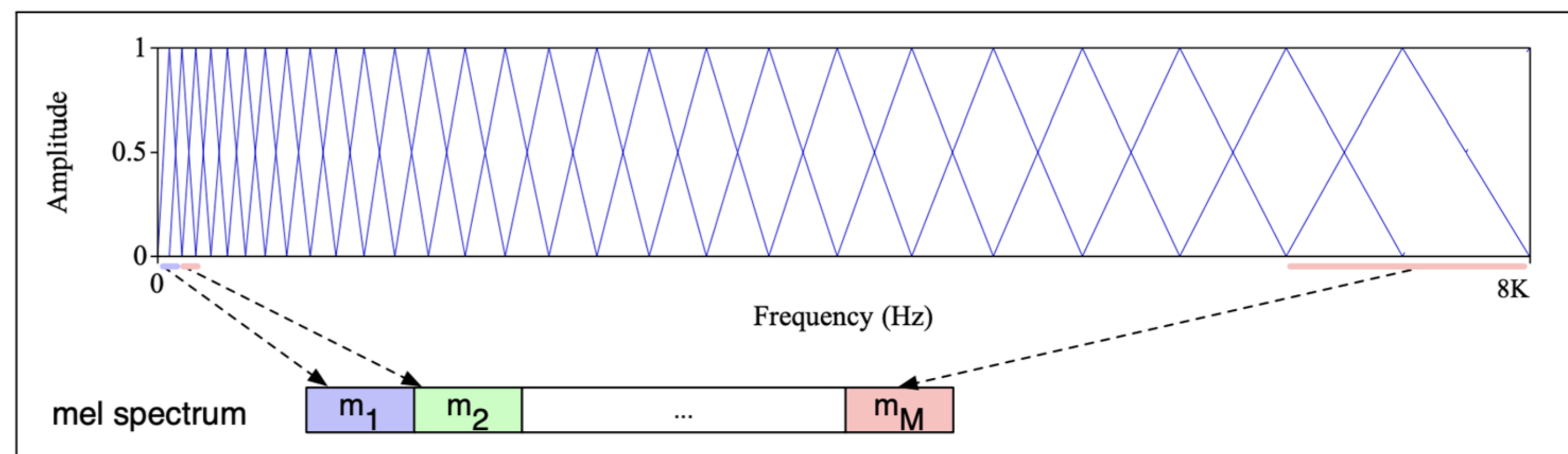


Figure 16.7 The mel filter bank (Davis and Mermelstein, 1980). Each triangular filter, spaced logarithmically along the mel scale, collects energy from a given frequency range.

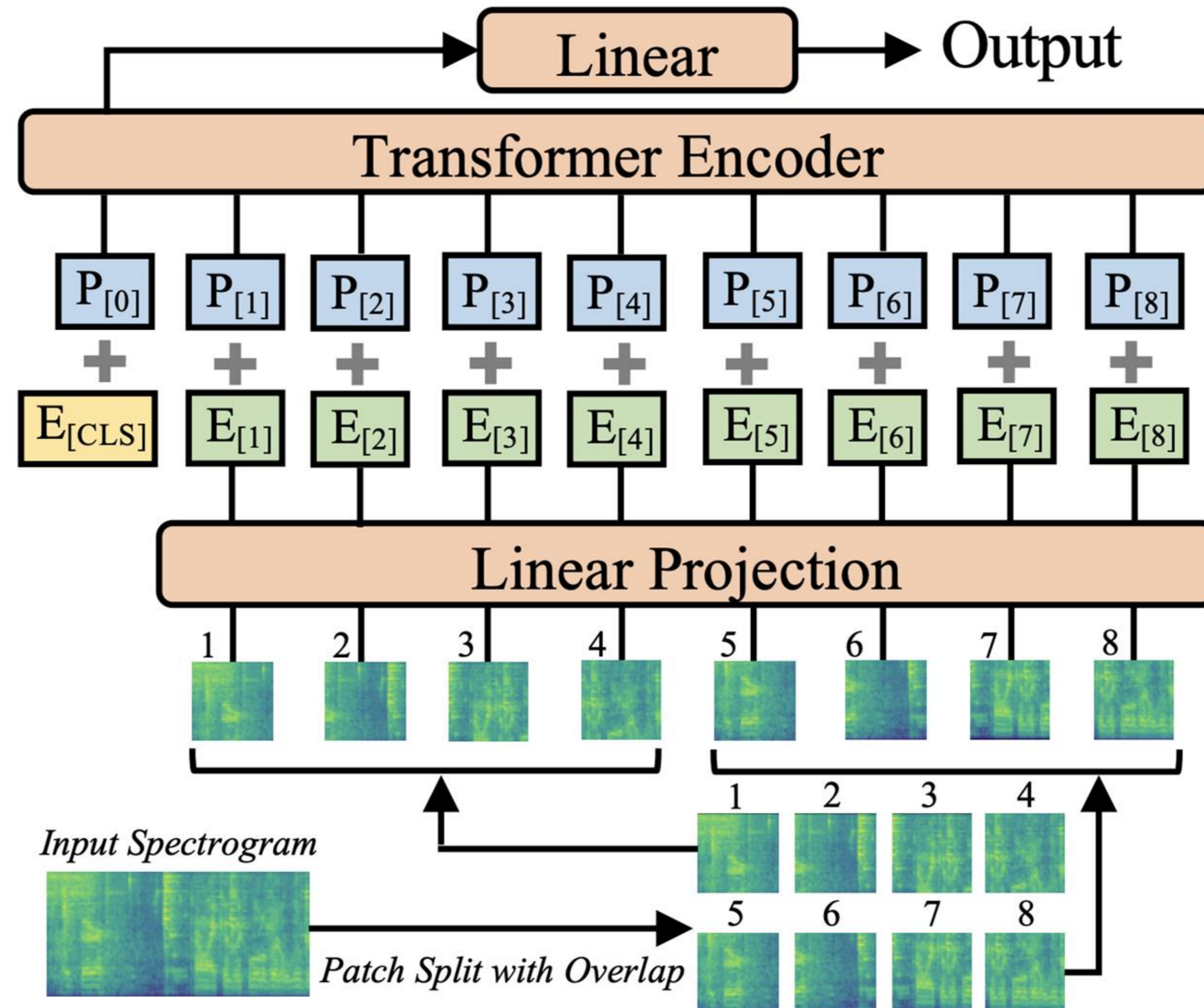
Log

The human response to signal level is logarithmic: Humans are less sensitive to slight differences in amplitude at high amplitudes than at low amplitudes

→ Take the log of each of the mel spectrum values!

Using a log also makes the feature estimates less sensitive to variations in input such as variations due to the speaker's mouth moving closer or further from the microphone

Audio Spectrogram Transformer [Gong et al., 2021]



Qwen 2/3-Audio [Chu et al., 2024]

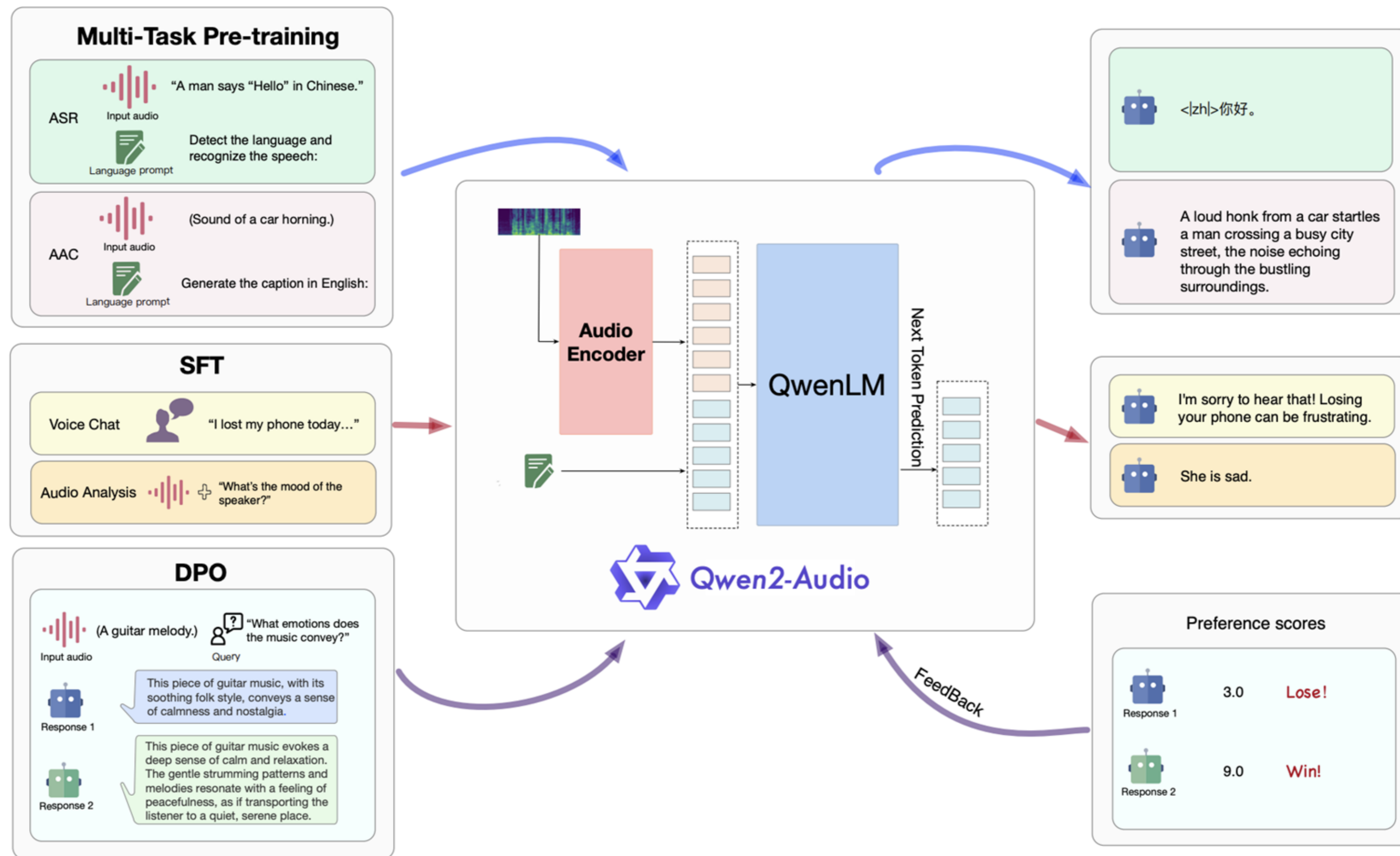


Figure 2: The overview of three-stage training process of Qwen2-Audio.