Generative AI - Text to Text - Part 2

INTERVIEW KICKSTART

Ahmed Elbagoury

Introduction – Ahmed Elbagoury

Senior ML Engineer, Google.

ML Research + Teaching

Applied ML

Teaching Applied ML

Optimize Your Experience

- ✅ Ask questions in the class
- ✅ Don’t ignore the math (Whys and Hows)
- ✅ Fortify understanding by reading papers, walking through code implementations and trying out notebooks!

- ✅ Enjoy the subject. Study deeply, seek understanding, read papers, practice problems, ask questions
- ✅ Don’t spend time in self-doubt. Unlike real life tortoise & hare race, slow & steady literally wins the career race

Here's the plan

Part 1 → How LLM can be used in applications ✅

Part 2 → Decoder details and Optimizations ⇐ **Today**

Today's Agenda

1

GPT Models

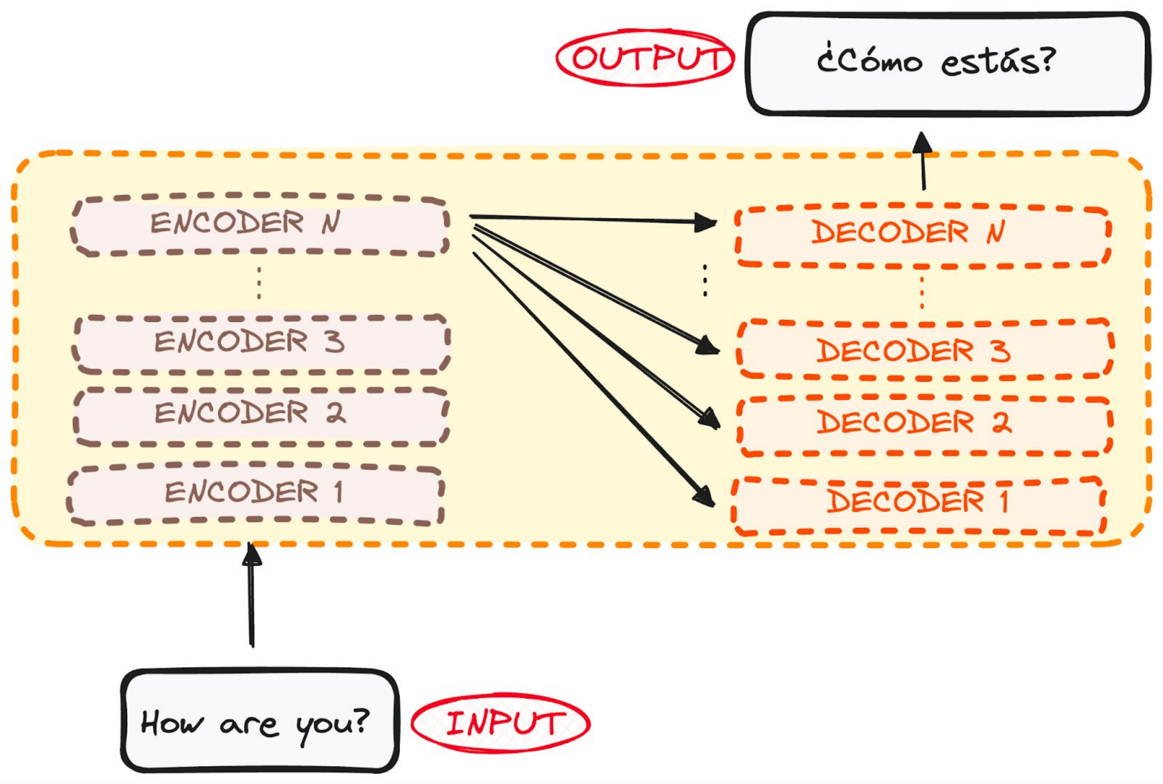
2

Sampling Strategies

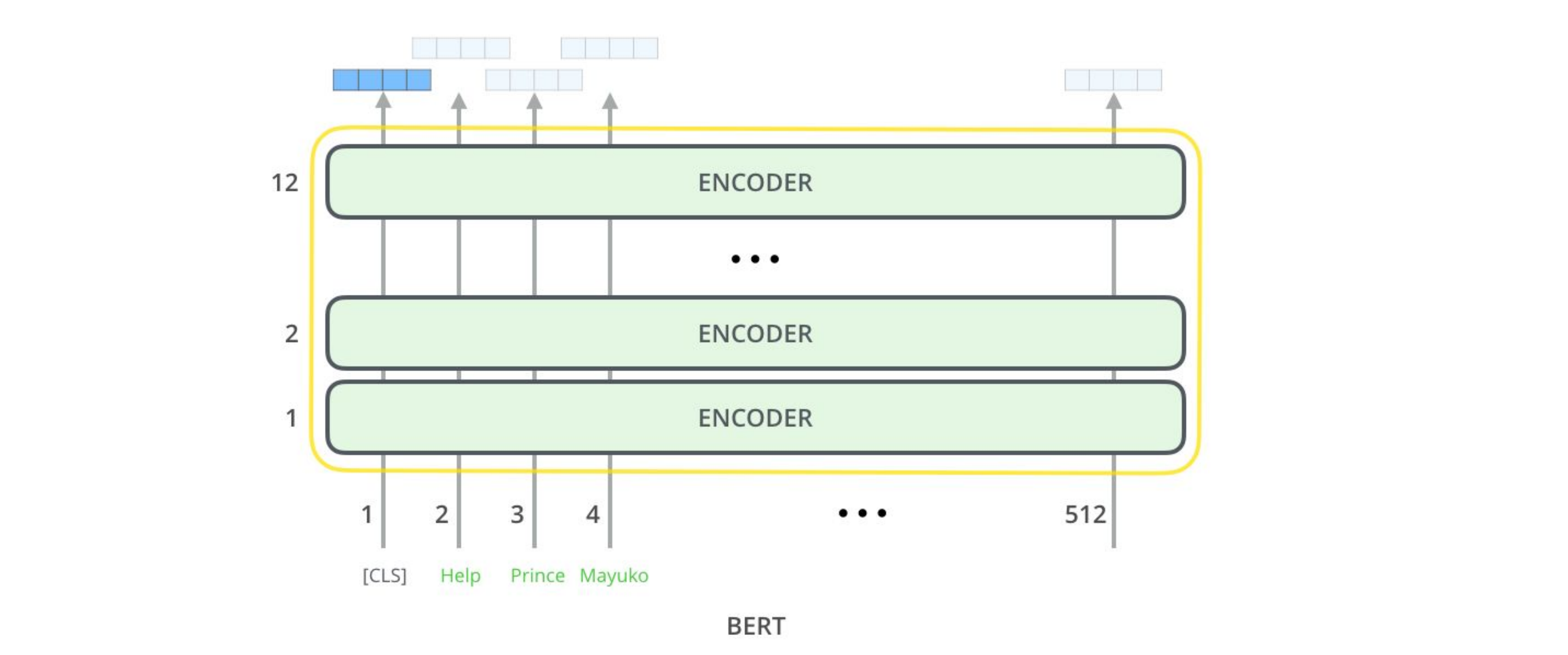
3

LLM Optimizations

We talked about encoder-decoder models

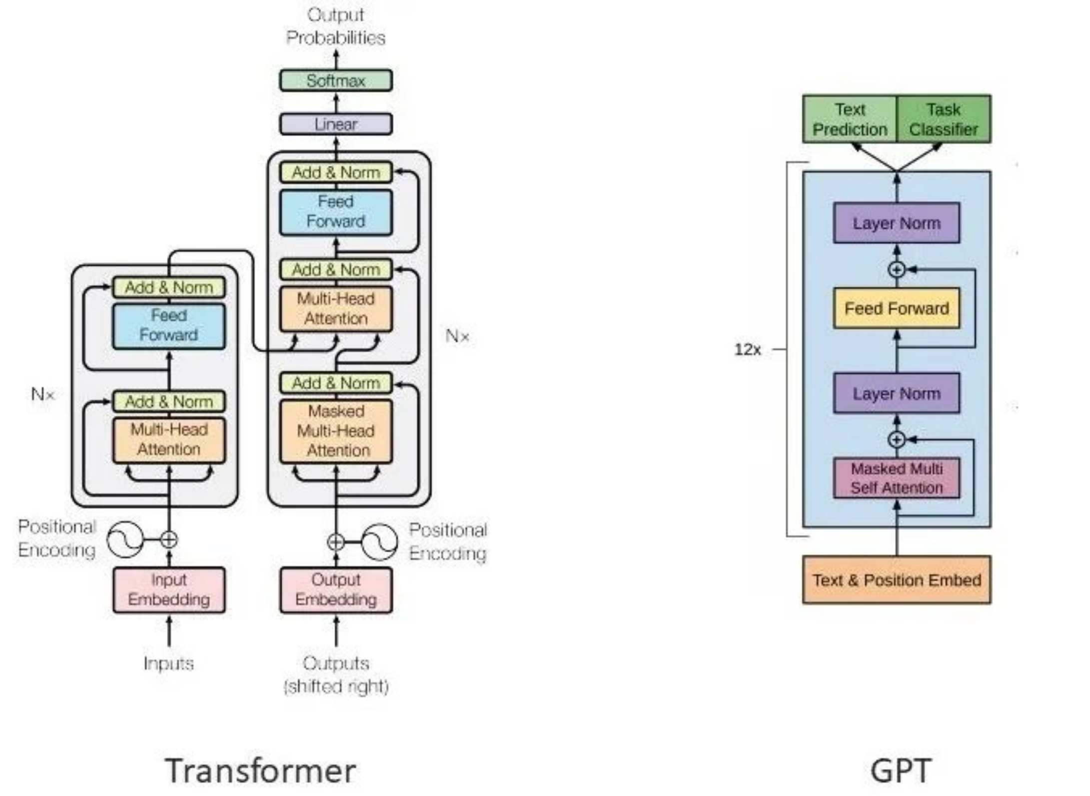


We talked about



What is missing?

Decoder Models




QUIZ
TIME!

Quiz

Which Component should be dropped from the decoder in GPT (decoder only) models?

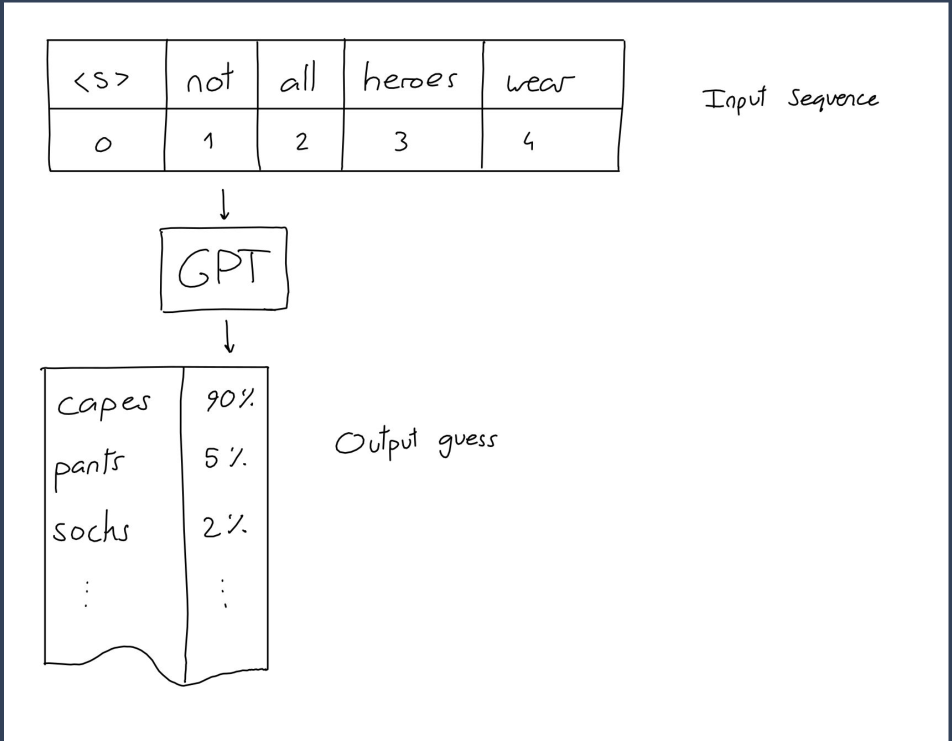
Quiz

Which Component should be dropped from the decoder in GPT (decoder only) models?

⇒ Encoder-decoder attention should be dropped since there is no encoder!

Decoder Models - Text Prediction

"Not all heroes wear capes" ⇒ 5

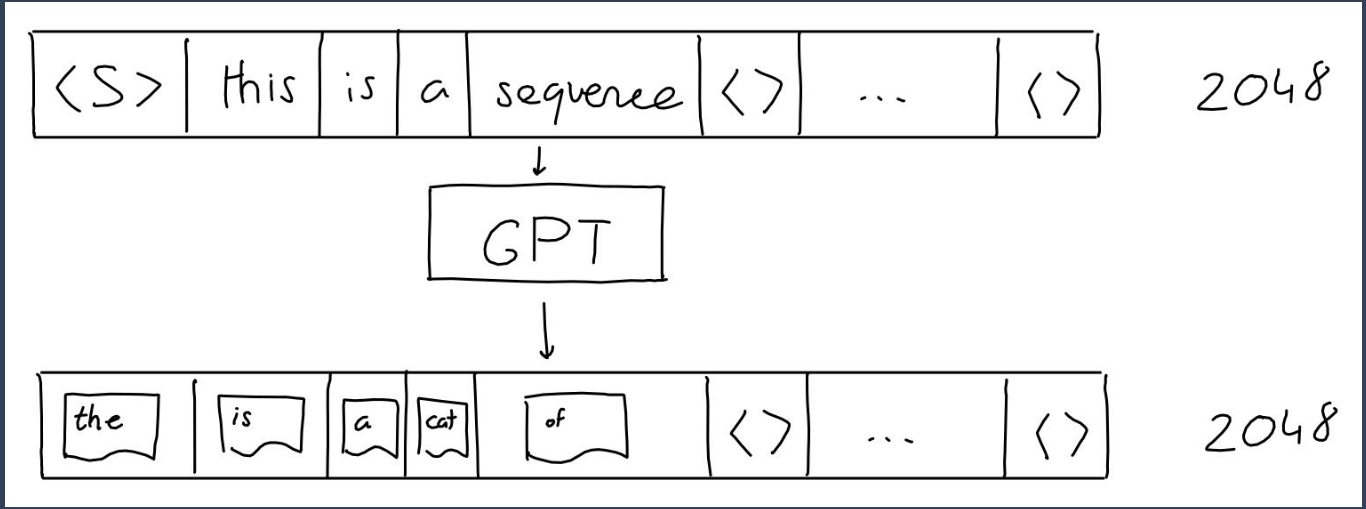


Decoder Models - Text Prediction

How to get multiple words and for different lengths?!

Decoder Models - Text Prediction

How to get multiple words and for different lengths?!



When generating text, we typically only look at the guess for the last word of the sequence.

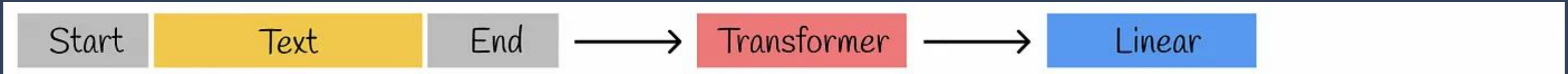
Decoder Models - Task Classifier

After pre-training

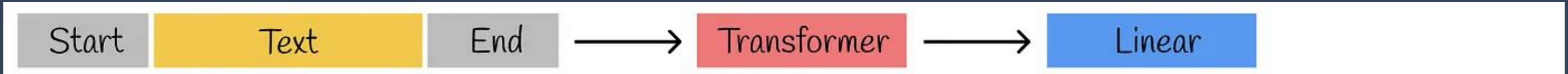
- GPT can capture linguistic knowledge of input sequences ✅
- However, to make it better perform on downstream tasks ❌

⇒ it needs to be fine-tuned on a supervised problem.

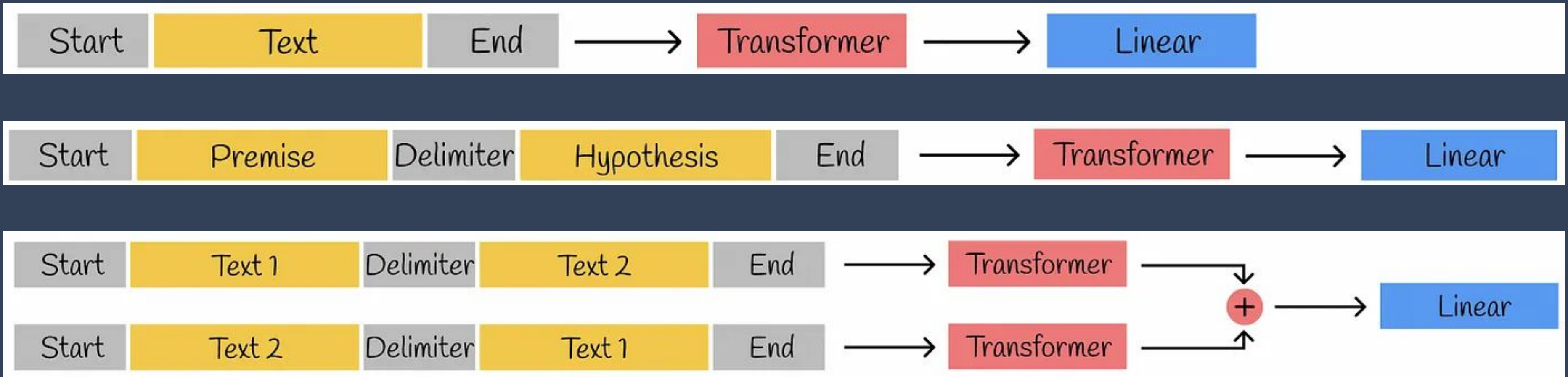
Decoder Models - Task Classifier



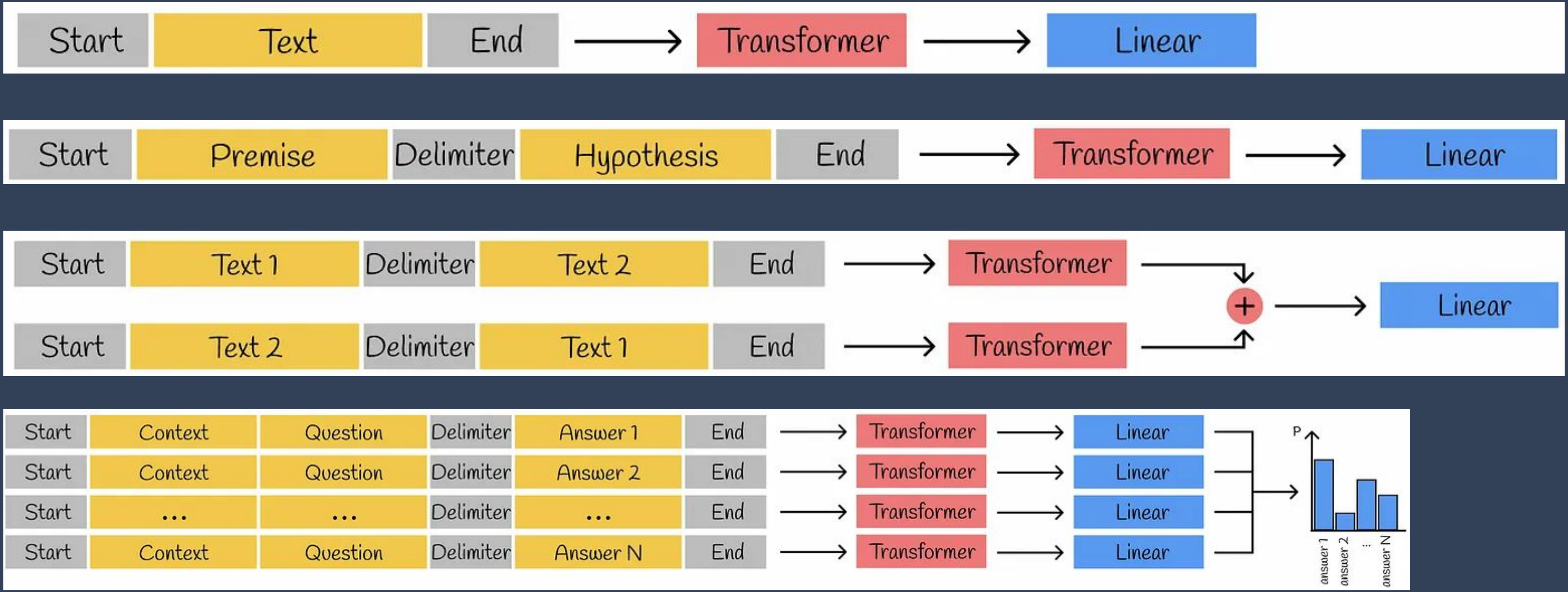
Decoder Models - Task Classifier



Decoder Models - Task Classifier



Decoder Models - Task Classifier



Why Decoder Models aka Causal Decoder/Models?

- Cost of training for Causal Decoder (CD) is cheaper
- CD works better for In-Context learning
  ⇒ It has a more straightforward effect for CD

Since Decoding is very important

Let's zoom in to it

Today's Agenda

1

GPT Models

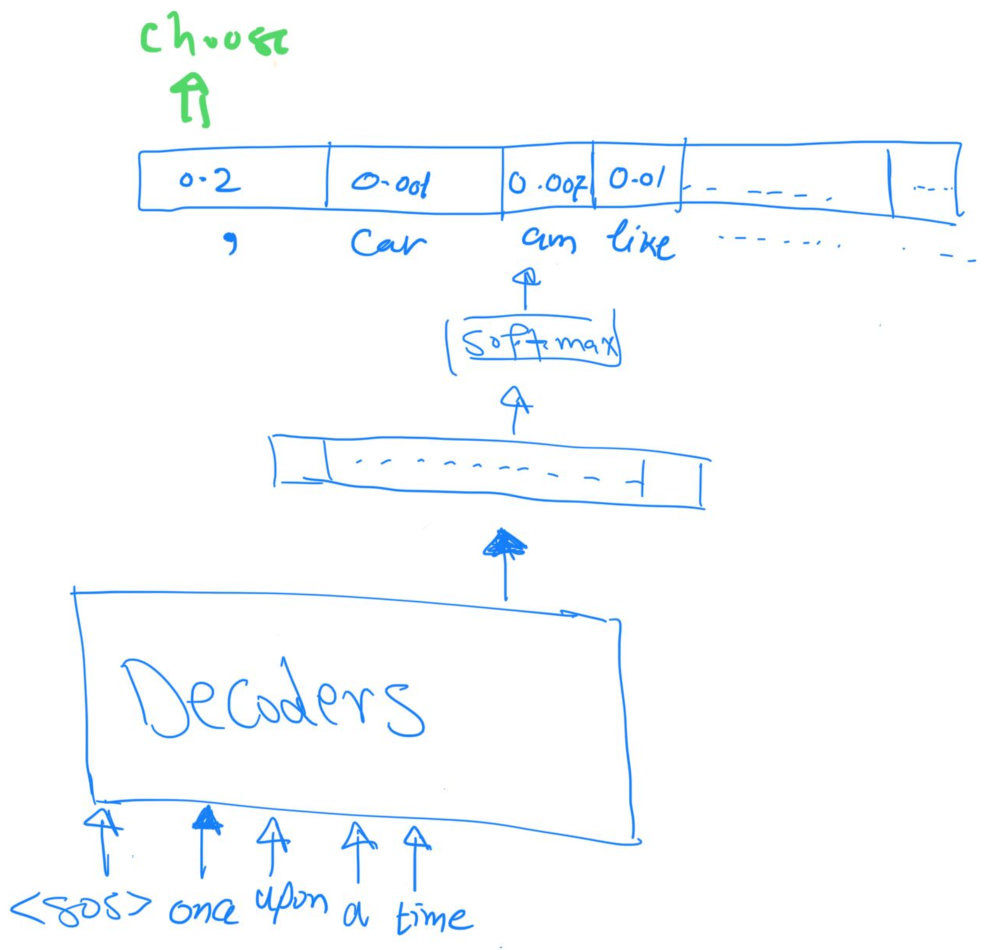
2

Sampling Strategies

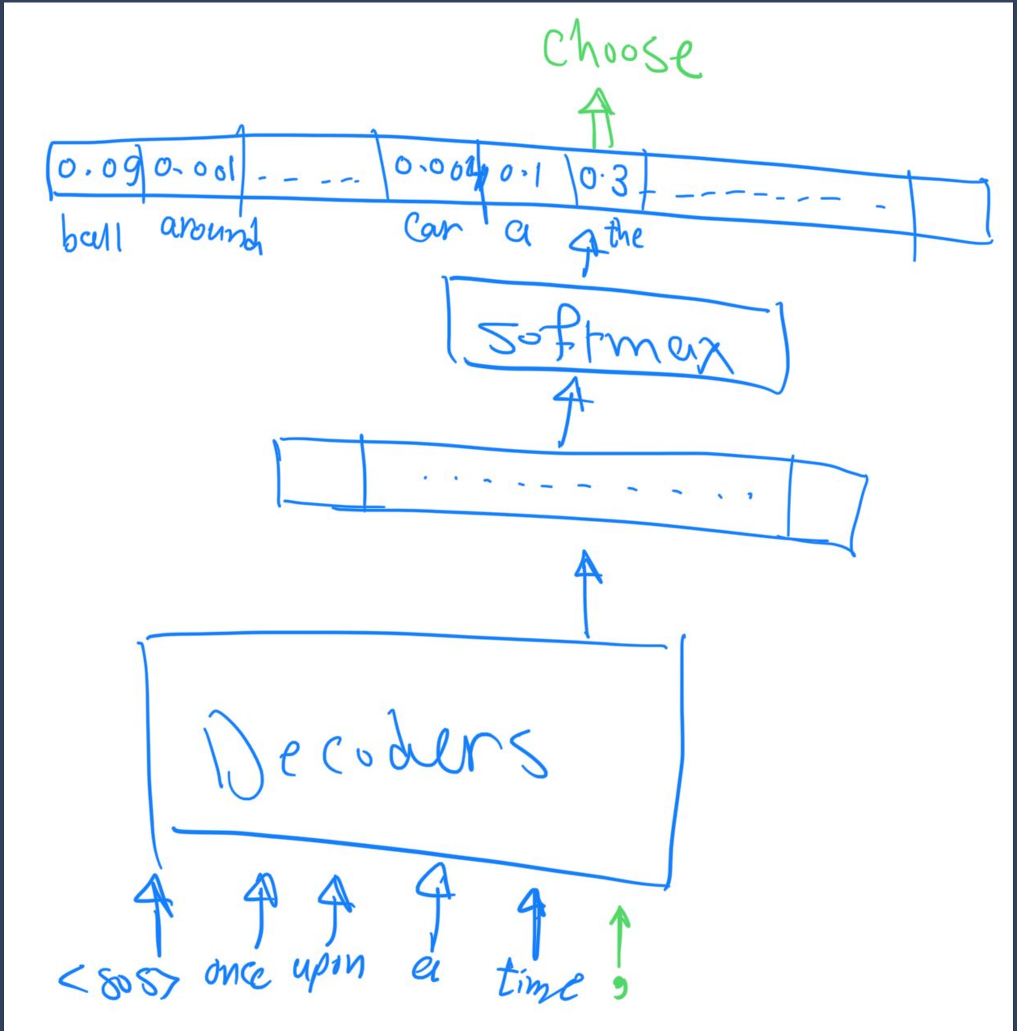
3

LLM Optimizations

Decoding Details



Decoding Details



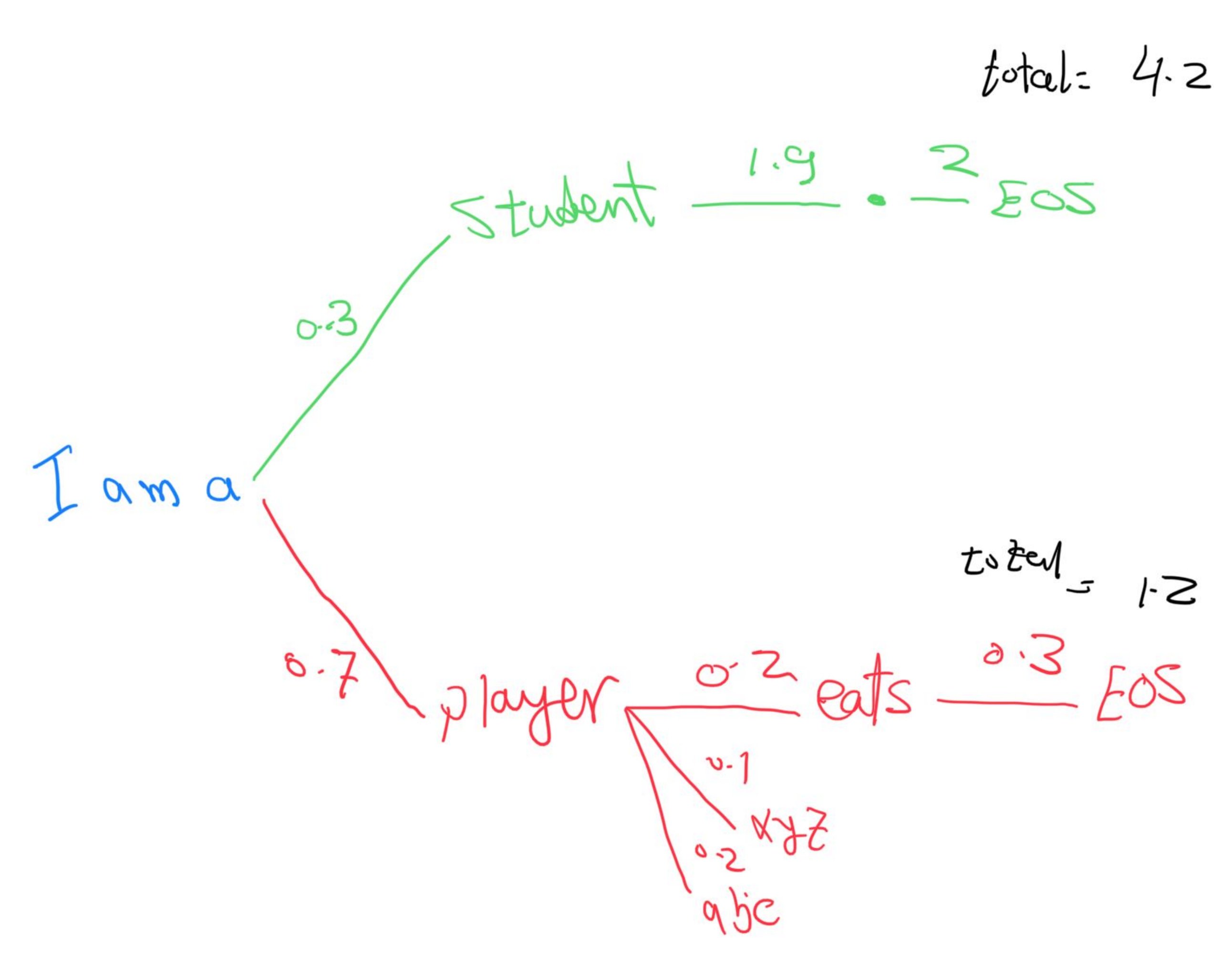
What is the Optimal Solution

- We want to maximize the likelihood of the output sequence
    - This is the multiplication of conditional probabilities
    - Or minimizing the summation of log likelihood

What is the Optimal Solution

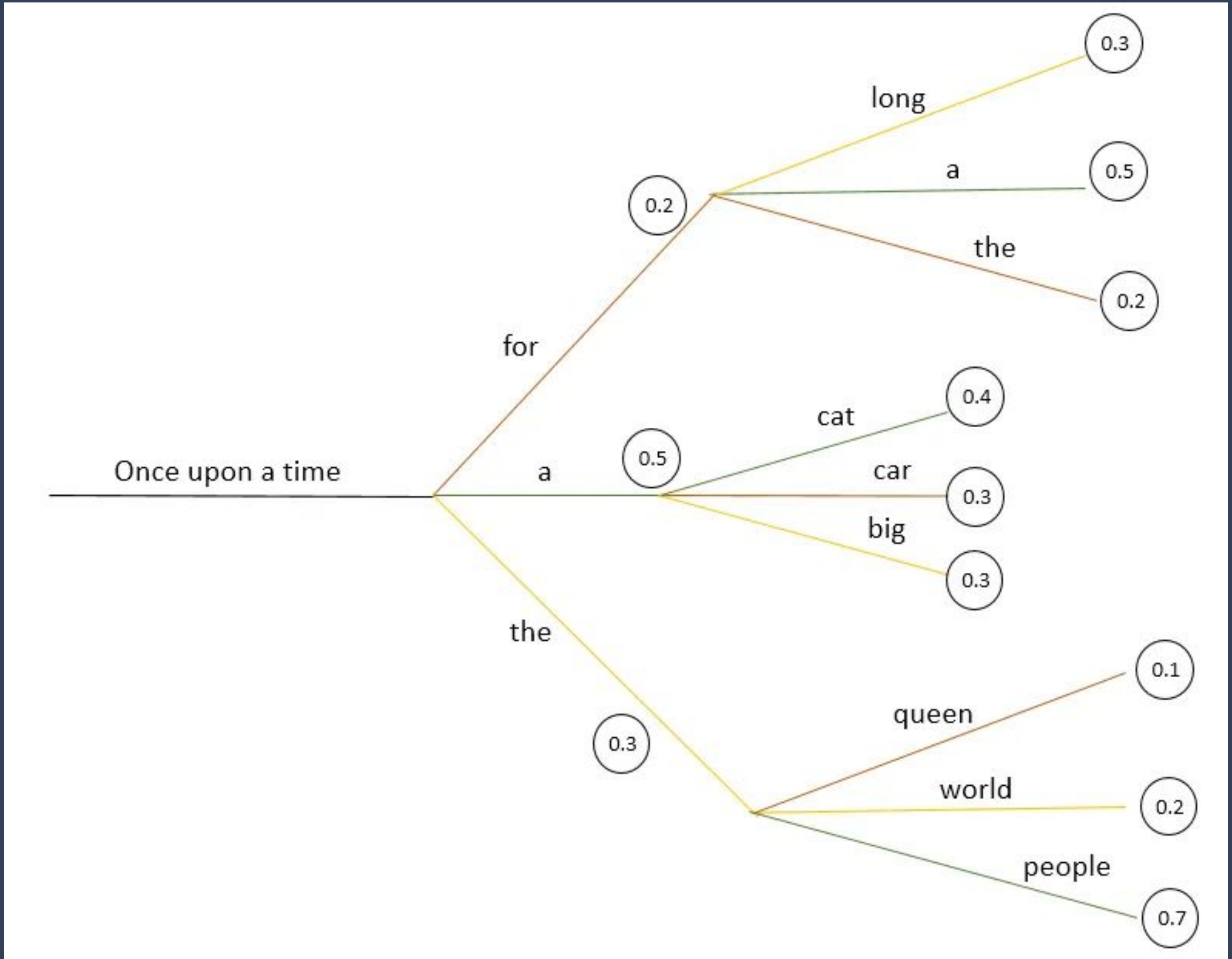
- We want to maximize the likelihood of the output sequence
  - This is the multiplication of conditional probabilities
  - Or minimizing the summation of log likelihood

What are the problems with the previous approach? (greedy)


total= 4.2
Student
1.9
2
EOS
0.3
I am a
0.7
total = 1.2
player
0.2
eats
0.3
EOS
0.1
xyz
0.2
abc

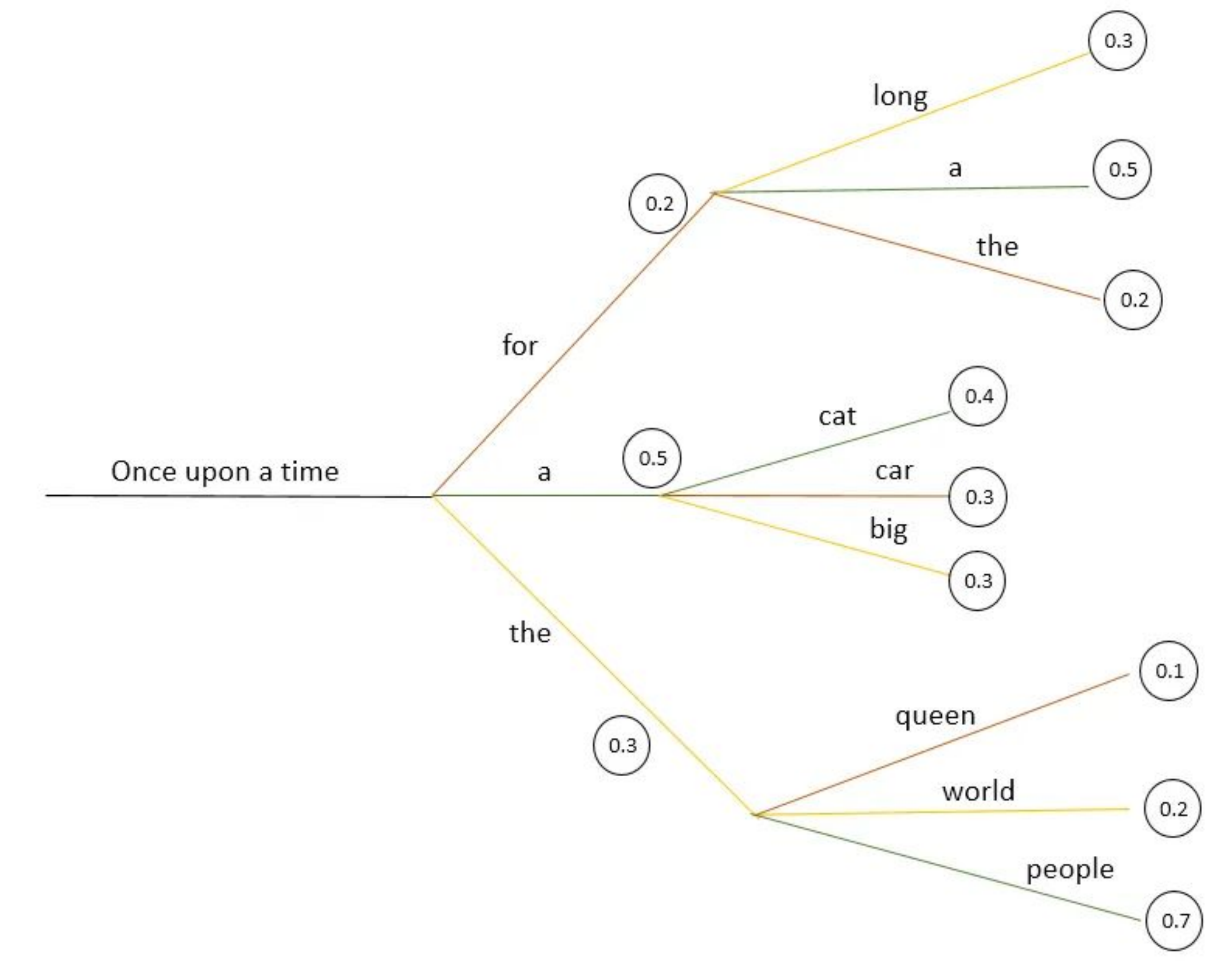
Decoding Details: Beam Search

- Keep your options open (to some extent)
- Just in case your local choices are not optimal
- It requires more memory and computational power to keep track of the beam



Once upon a time
for
0.2
long
0.3
a
0.5
the
0.2
a
0.5
cat
0.4
car
0.3
big
0.3
the
0.3
queen
0.1
world
0.2
people
0.7

beam = ??


Once upon a time
for
0.2
long
0.3
a
0.5
the
0.2
a
0.5
cat
0.4
car
0.3
big
0.3
the
0.3
queen
0.1
world
0.2
people
0.7

beam = 3

Decoding Details: What if we got the answer wrong

Help me write a congratulation email to a new colleague ⇒ **Decoder Model** ⇒ I hope this message finds you well. I wanted to take a moment to personally welcome you to …

Let's say I want to see other drafts!

Decoding Details: What if we got the answer wrong

Help me write a congratulation email to a new colleague → **Decoder Model** → I hope this message finds you well. I wanted to take a moment to personally welcome you to …

Help me write a congratulation email to a new colleague → **Decoder Model** → We're thrilled to have you on board and are excited about the great things you'll bring to the team

Decoding Details: What if we got the answer wrong

- Random (aka stochastic) sampling to the rescue

Sorry more terminology!
So that you know the lingo!

INTERVIEW KICKSTART

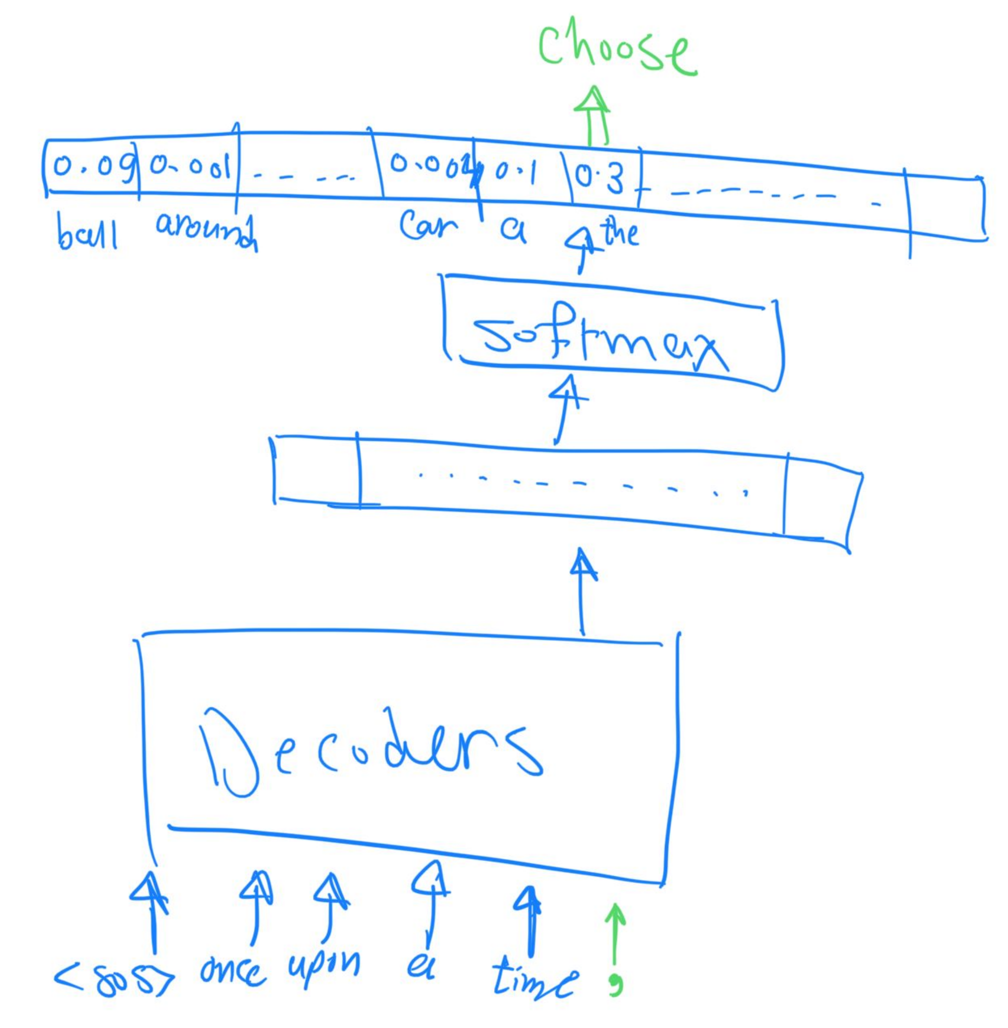
Decoding Details: What if we got the answer wrong

- Random (aka stochastic) sampling to the rescue

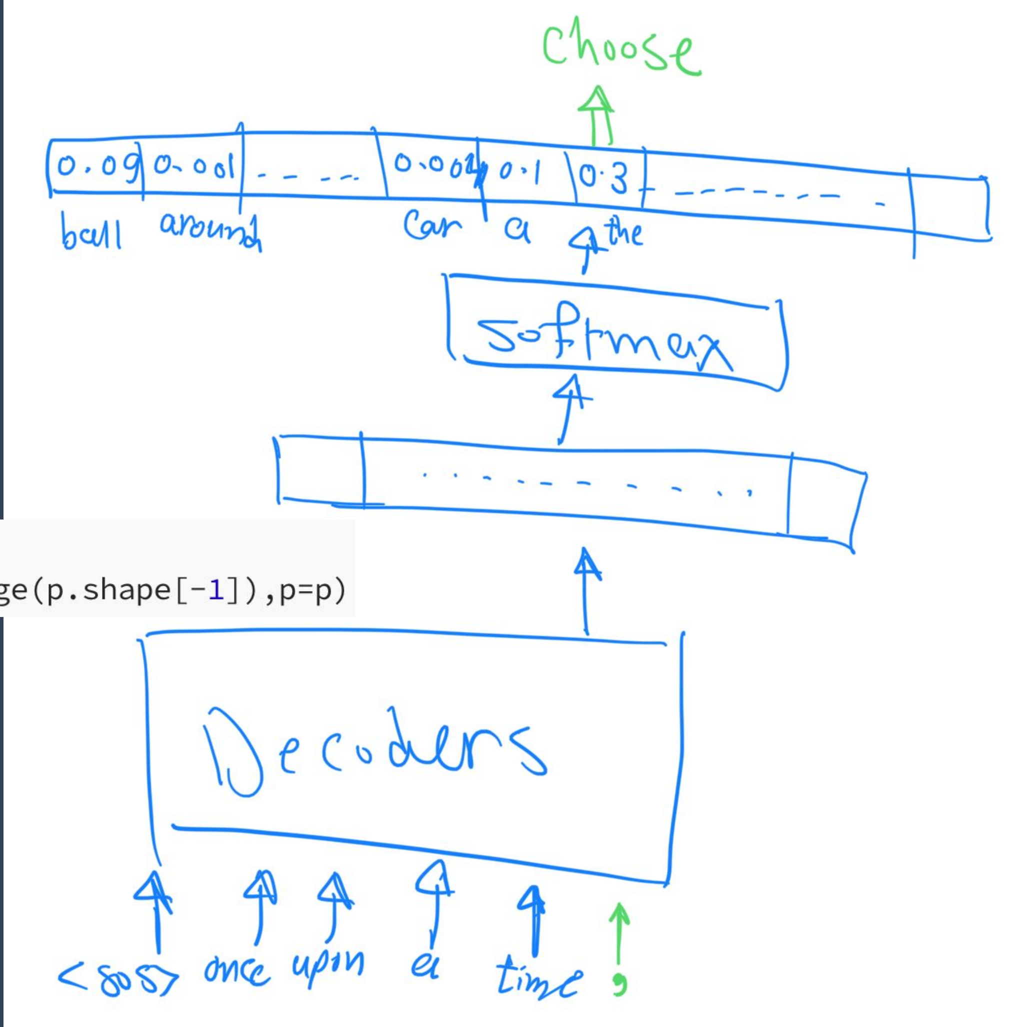
Sorry more terminology!
So that you know the lingo!

- **It just means**
  - Choose randomly according to the probability scores

Random Sampling

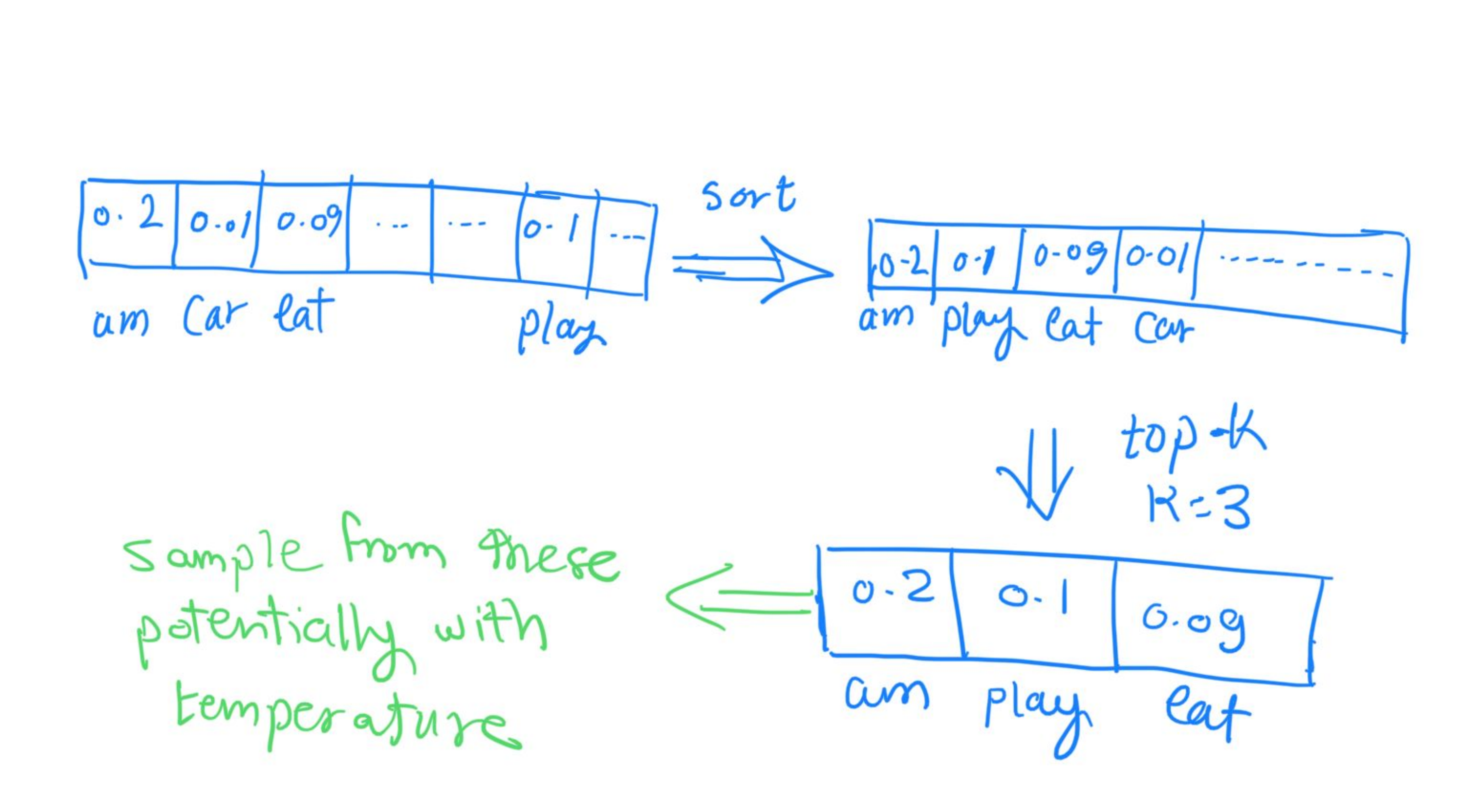


Random Sampling

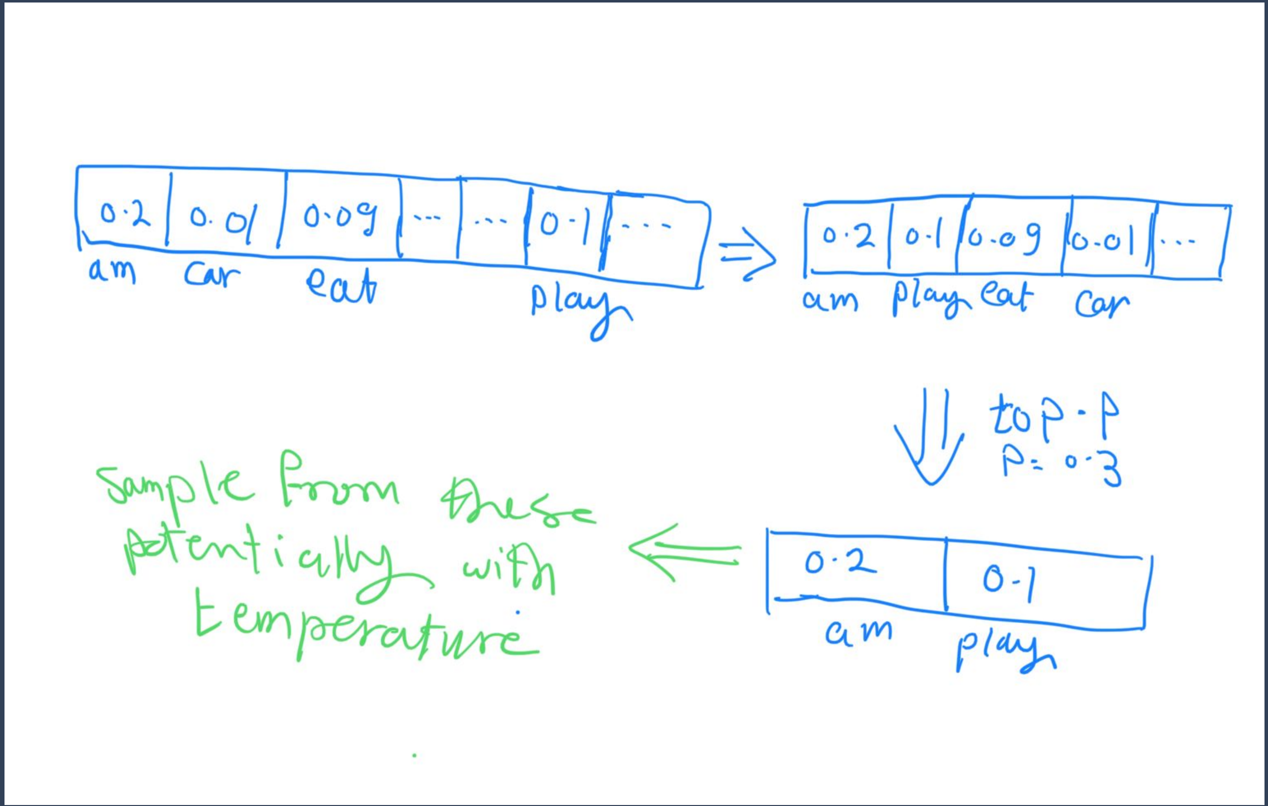



```python
def sample(p):
  return np.random.choice(np.arange(p.shape[-1]),p=p)
```

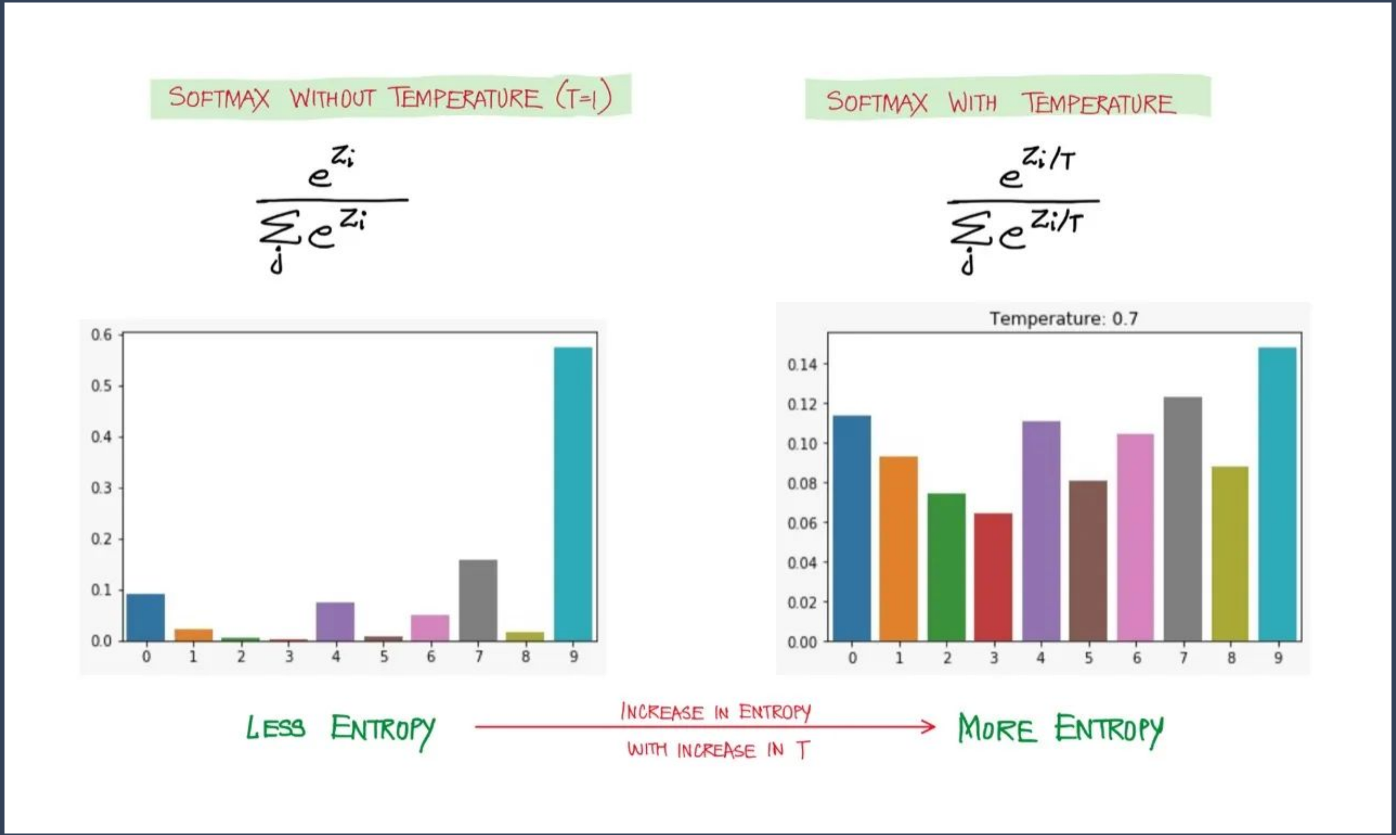
Decoding Details: Top-K



Decoding Details: Top-p aka Nucleus



Temperature



Demo Time

https://colab.research.google.com/drive/1CejK4UaV0O6L3c4jnKqtWENp9YFboNlg

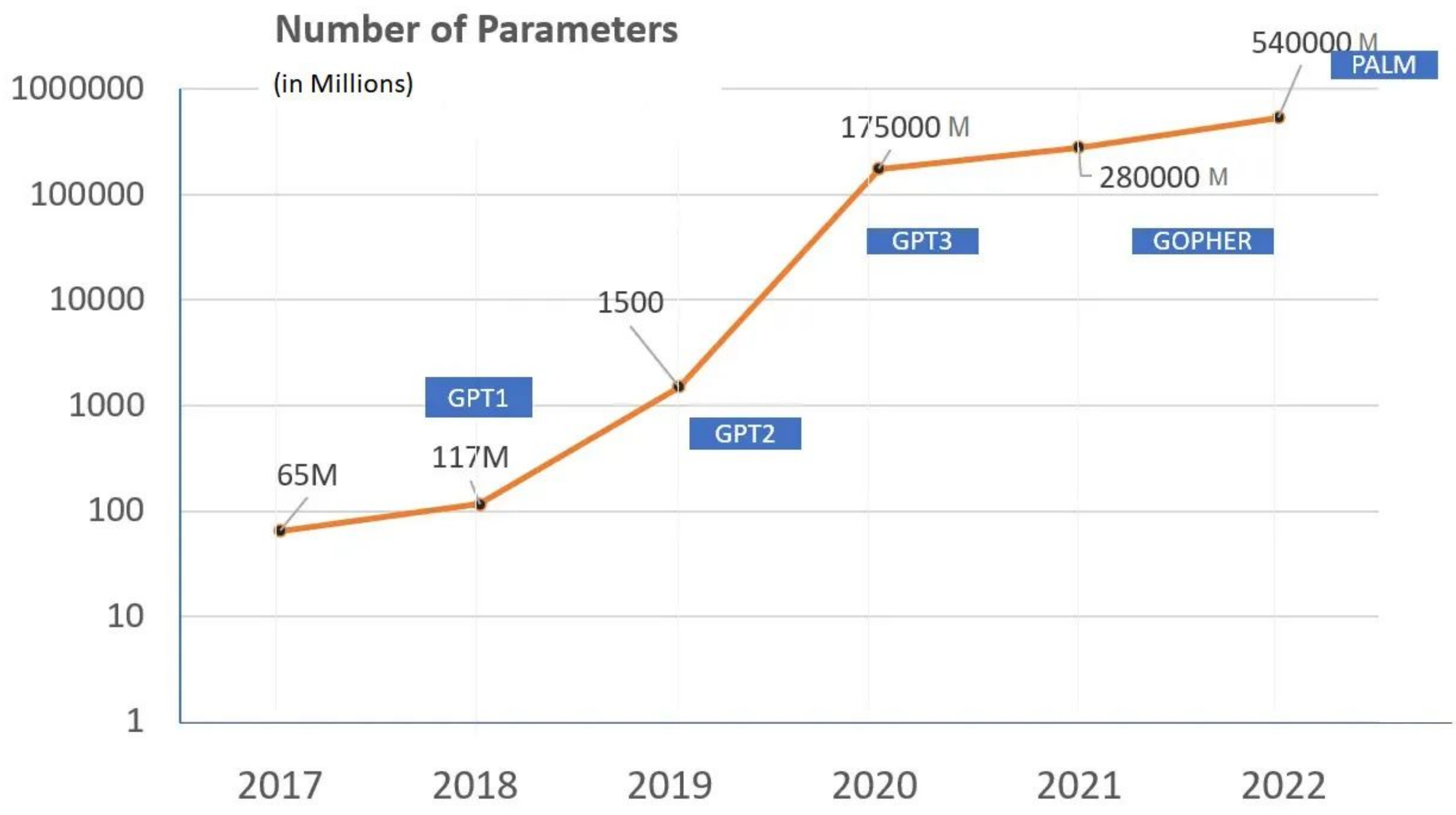
Today’s Agenda

1. **GPT Models**
2. **Sampling Strategies**
3. **LLM Optimizations**

Optimization Strategies for LLM

Larger Models Mean More Capacity

- To handle variety of tasks ⇒ Models need to be trained on large amount of data
- To avoid underfitting ⇒ Models need to be larger!


Number of Parameters
(in Millions)
1000000
100000
10000
1000
100
10
1
65M
117M
1500
175000 M
280000 M
540000 M
GPT1
GPT2
GPT3
GOPHER
PALM
2017
2018
2019
2020
2021
2022

Larger Models Mean More Capacity

- To handle variety of tasks
  - Models need to be trained on large amount of data
- To avoid underfitting
  - Models need to be larger!

Implications of Increasing Model Size

Implications of Increasing Model Size

- More hardware resources
    - On-device use cases?
- Higher Latency
- Worse Carbon footprint

Do we have to use these huge models?

- A line of research motivated by the challenges of training and productionizing LLMs
- It aims to
  - Improving training and/or inference latency
  - Reducing model sizes with little to no impact on quality

Do we have to use these huge models?

- A line of research motivated by the challenges of training and productionizing LLMs
- It aims to
  - Improving training and/or inference latency
  - Reducing model sizes with little to no impact on quality

Major techniques:

- KV Caching
- LoRA Tuning (More generally PEFT)
- Distillation
- Model Pruning
- Quantization

KV Caching

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

Photo from

KV Caching

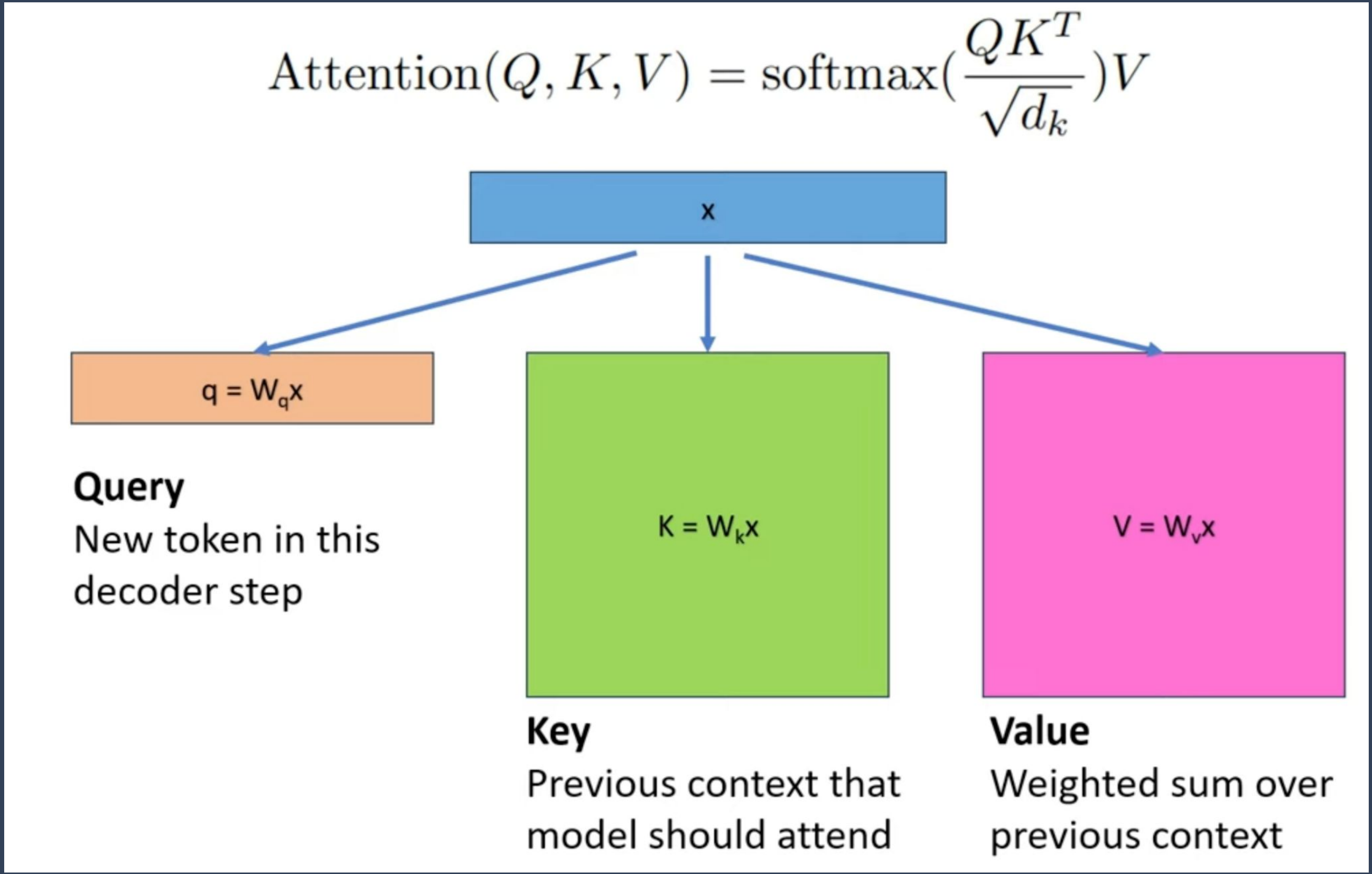
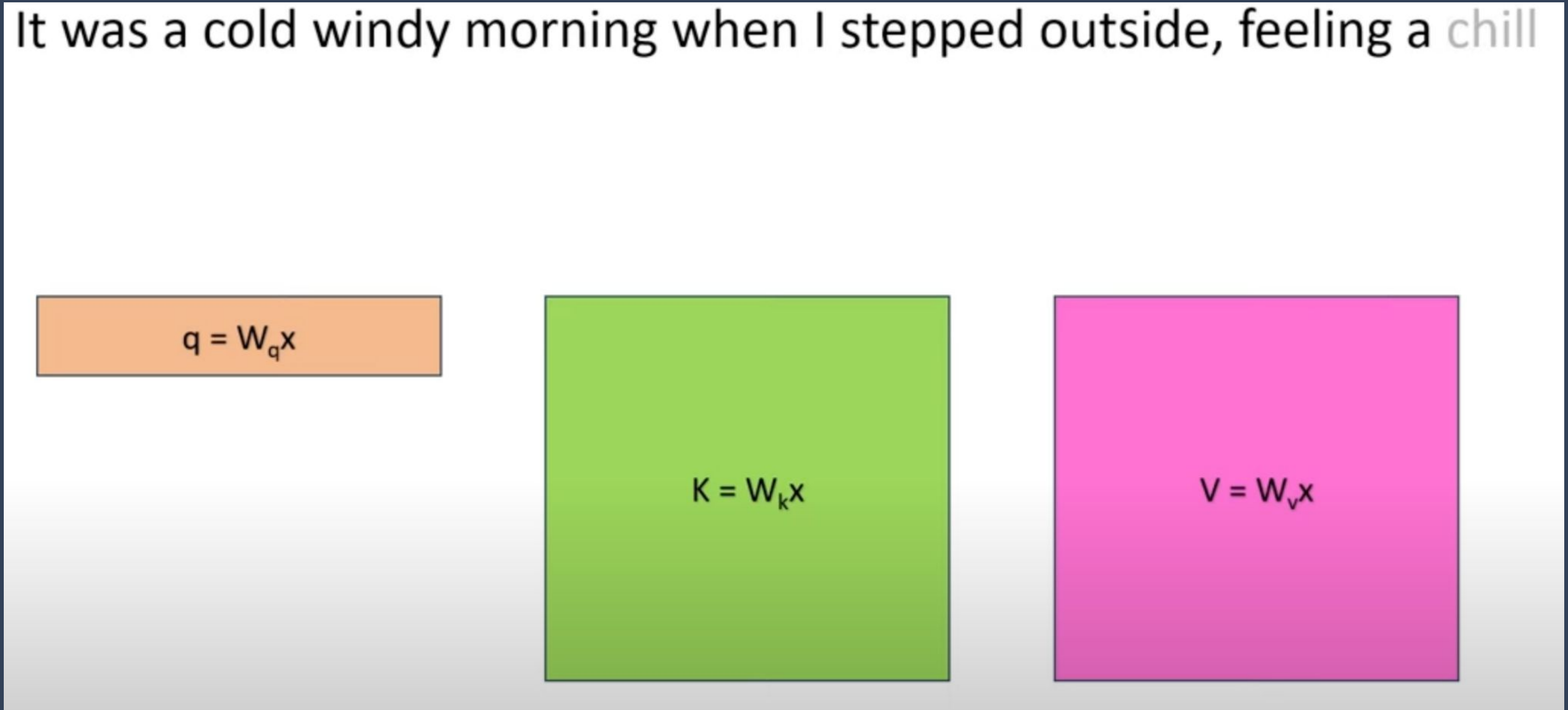


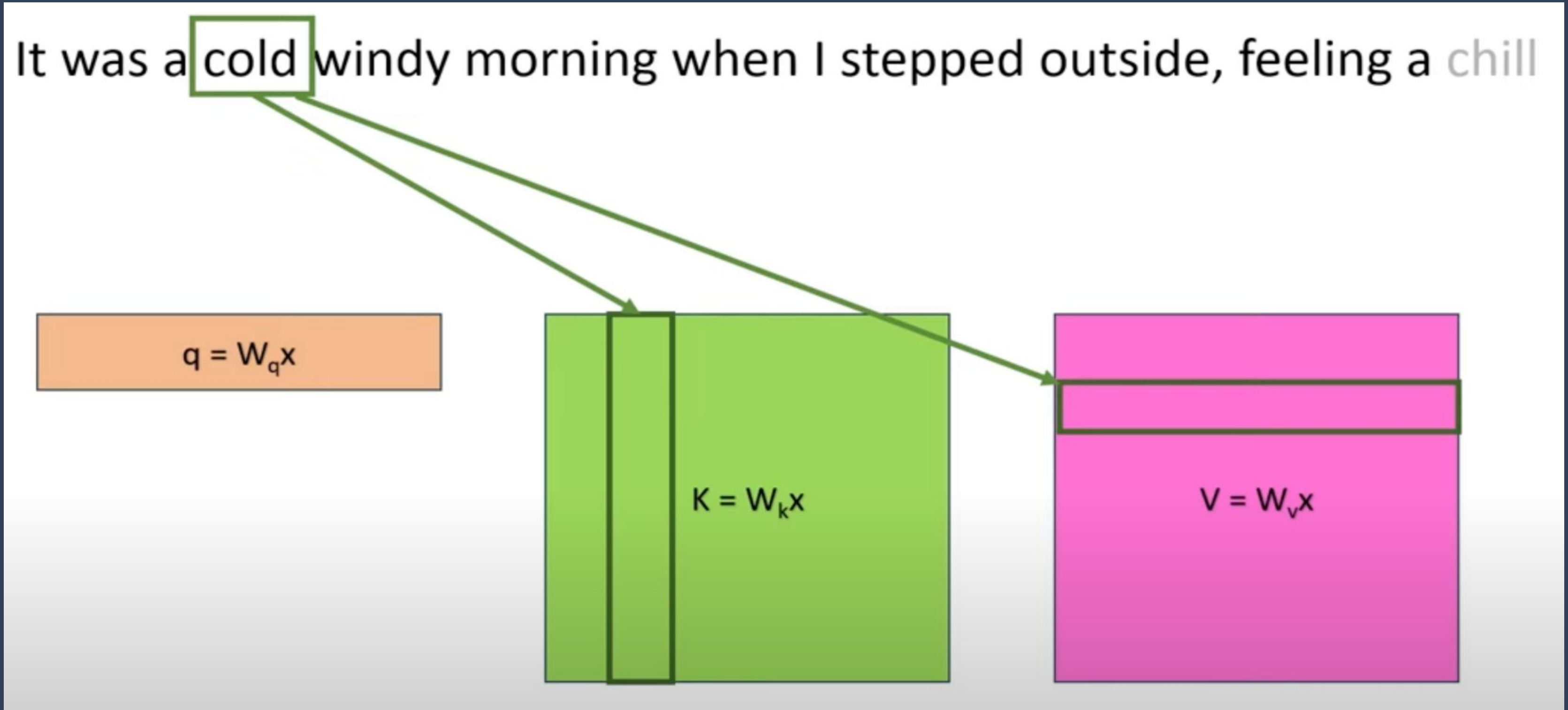
Photo from

KV Caching



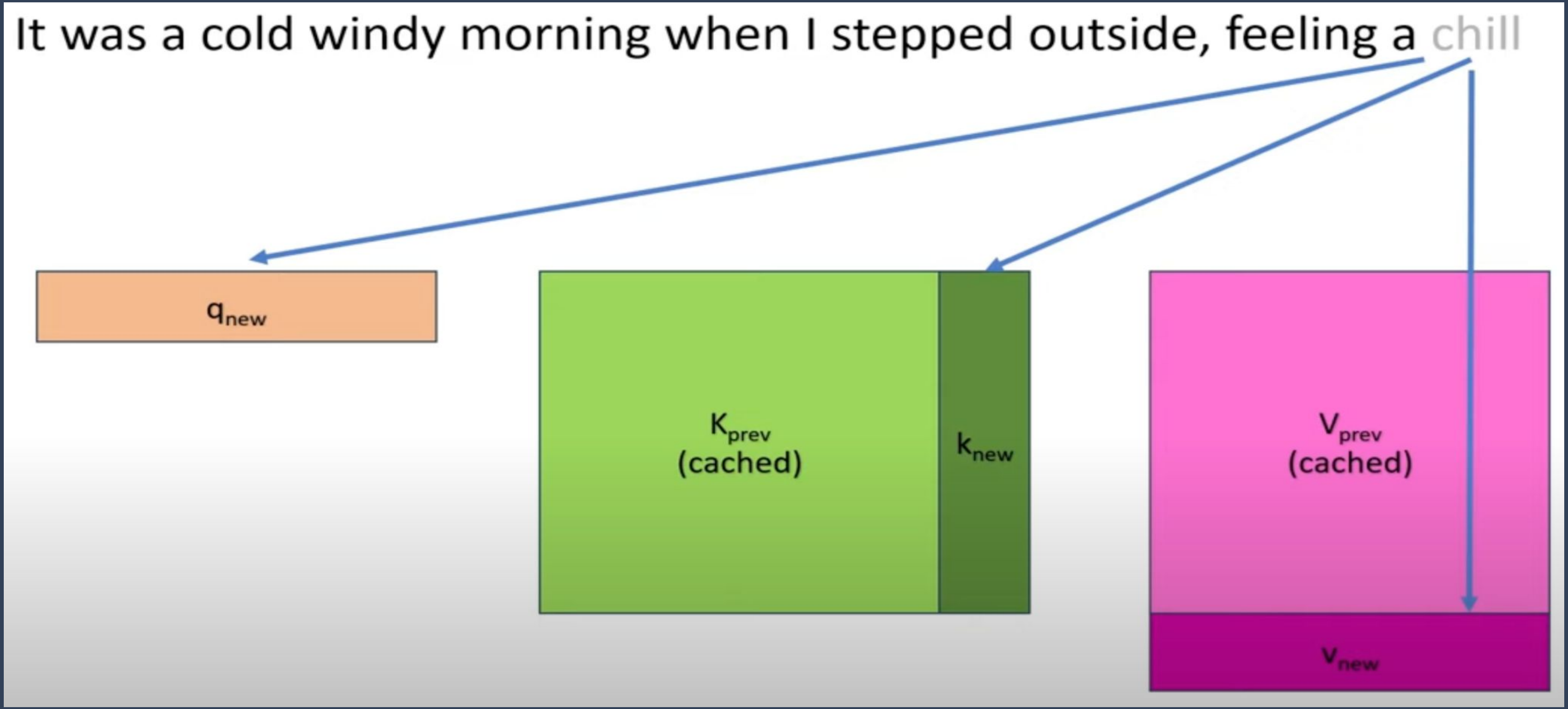
It was a cold windy morning when I stepped outside, feeling a chill
$q = W_q x$
$K = W_k x$
$V = W_v x$

KV Caching

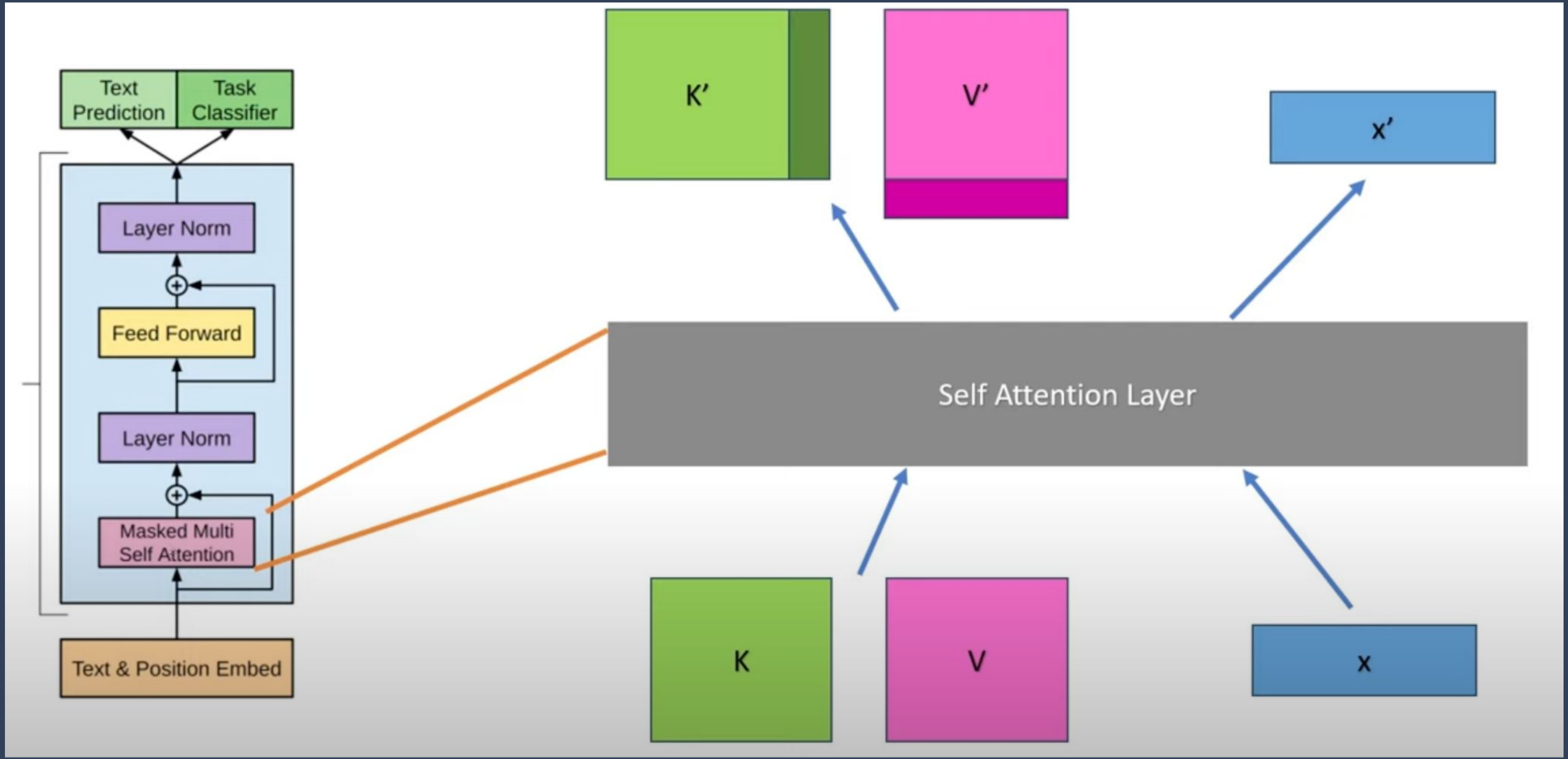


It was a cold windy morning when I stepped outside, feeling a chill
q = W_q x
K = W_k x
V = W_v x

KV Caching


It was a cold windy morning when I stepped outside, feeling a chill
q_{new}
K_{prev}
(cached)
k_{new}
V_{prev}
(cached)
v_{new}

KV Caching


Text Prediction
Task Classifier
Layer Norm
Feed Forward
Layer Norm
Masked Multi Self Attention
Text & Position Embed
K'
V'
x'
Self Attention Layer
K
V
x

Demo Time

https://colab.research.google.com/drive/12ioUtylE5BuWTNjDHdecNQ8Xnb1dRISW


QUIZ
TIME!

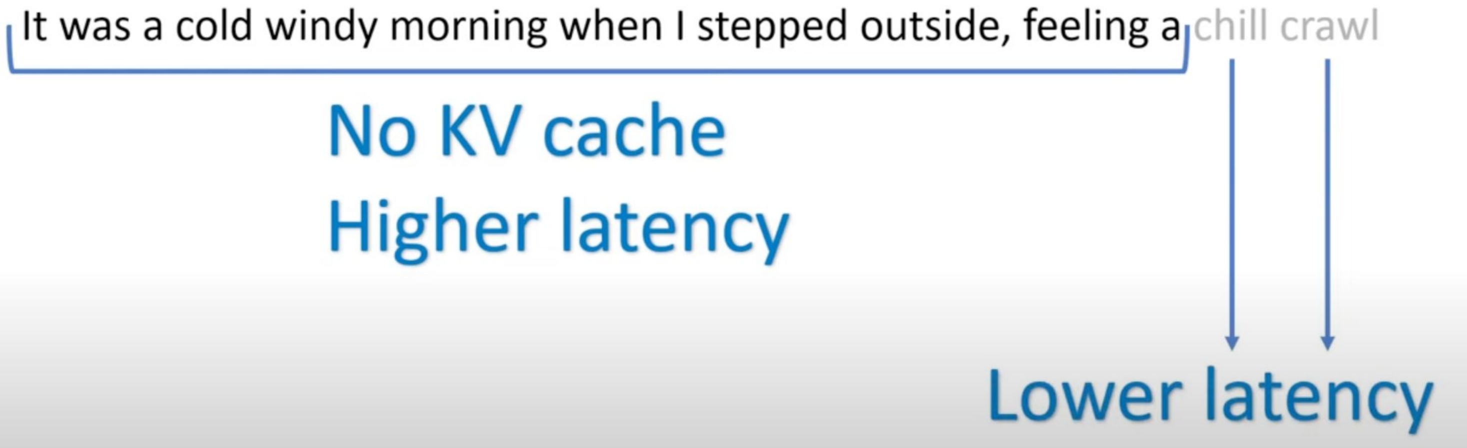
Quiz

Which part will run faster: Red or Blue? and why?
Assuming that red is the prompt

It was a cold windy morning when I stepped outside, feeling a chill crawl

Quiz

Which part will run faster: Red or Blue? and why?
Assuming that red is the prompt



One more Quiz

KV Caching improves <u>inference</u> latency at the expense of

A) Memory Usage during inference
B) Training Time
C) A & B
D) Memory Usage during training

Quiz

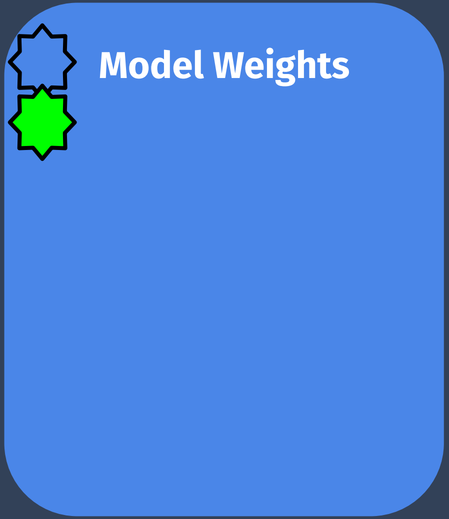
KV Caching improves <u>inference</u> latency at the expense of

A) Memory Usage during inference
B) Training Time
C) A & B
D) Memory Usage during training

LoRA

- Low-Rank Adaptation of Large Language Models
- Is a type of what is known as Parameter-Efficient tuning (PEFT)
- The goal is fine-tuning a model
    - By updating smaller number of parameters

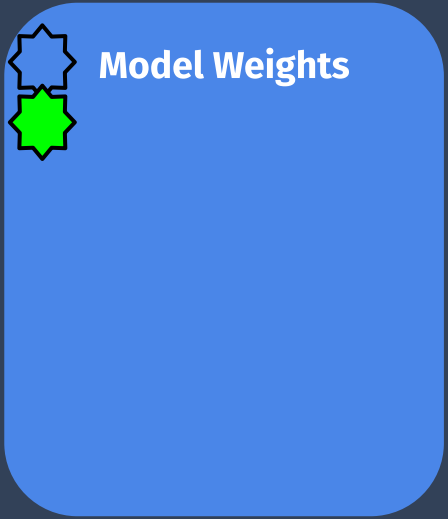
Why LoRA?



GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

Regular Fine-Tuning

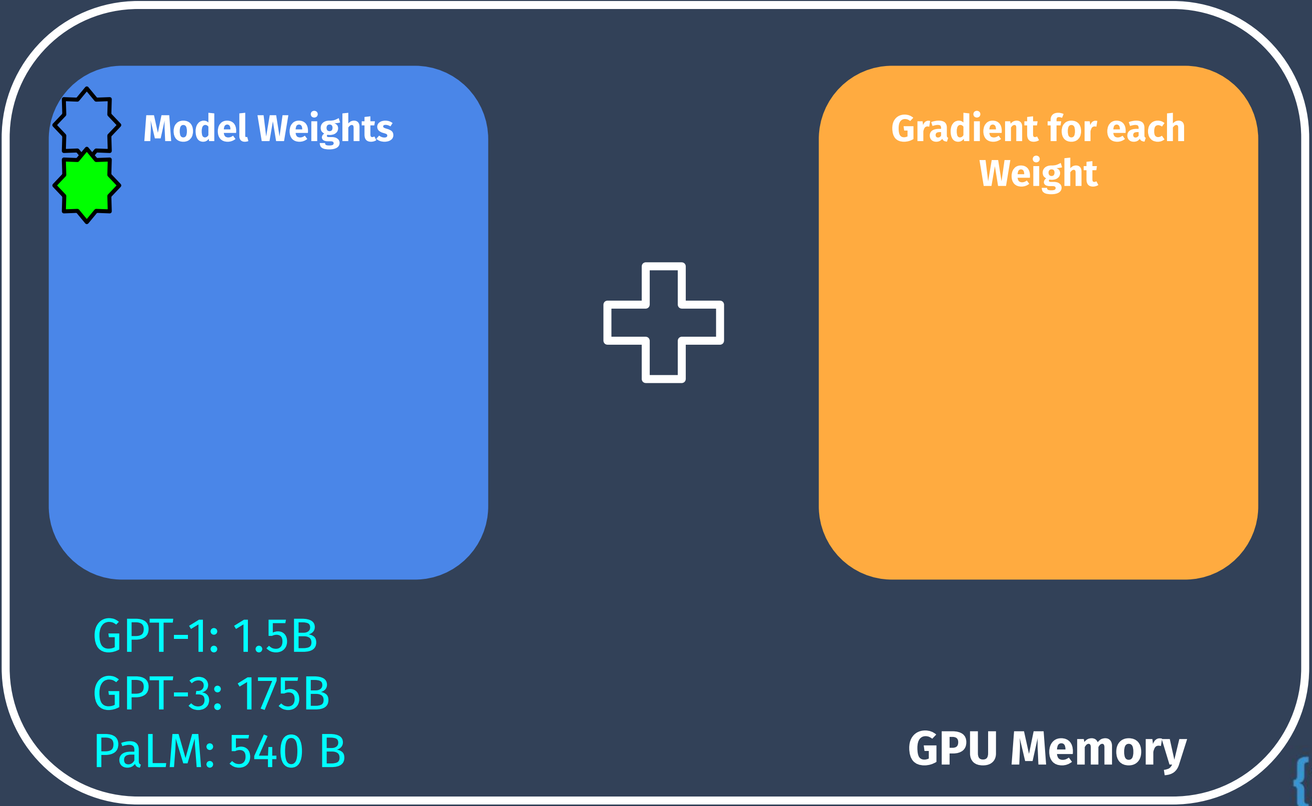
During fine-tuning ⟹ we need to load this in Memory



GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

Regular Fine-Tuning

During fine-tuning ⇒ we need to load this in Memory



Regular Fine-Tuning

During fine-tuning ⇒ we need to load this in Memory

Model Weights

$$\begin{pmatrix} w11 & w12 & w13 \\ w21 & w22 & w23 \\ w31 & w32 & w33 \end{pmatrix}$$

GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

\+

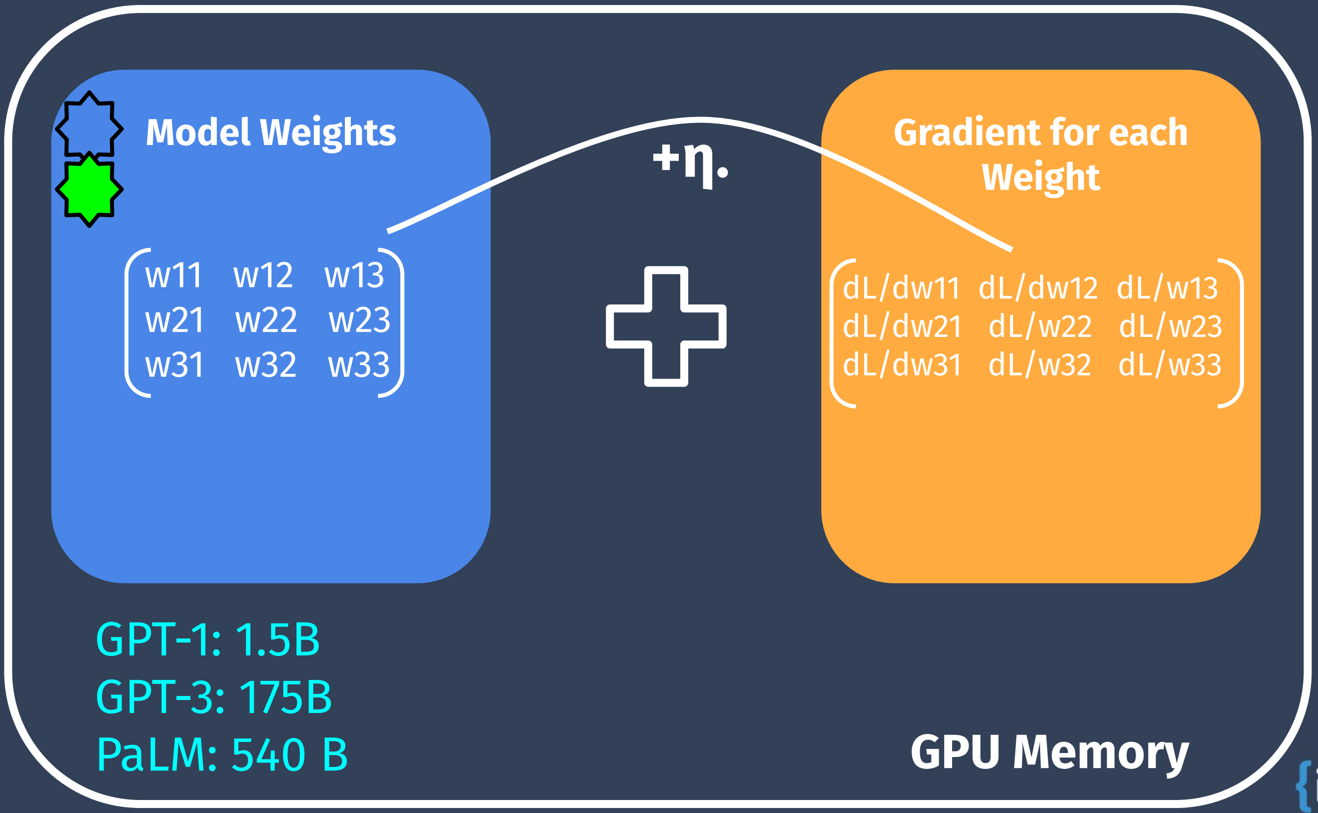
Gradient for each Weight

$$\begin{pmatrix} dL/dw11 & dL/dw12 & dL/w13 \\ dL/dw21 & dL/w22 & dL/w23 \\ dL/dw31 & dL/w32 & dL/w33 \end{pmatrix}$$

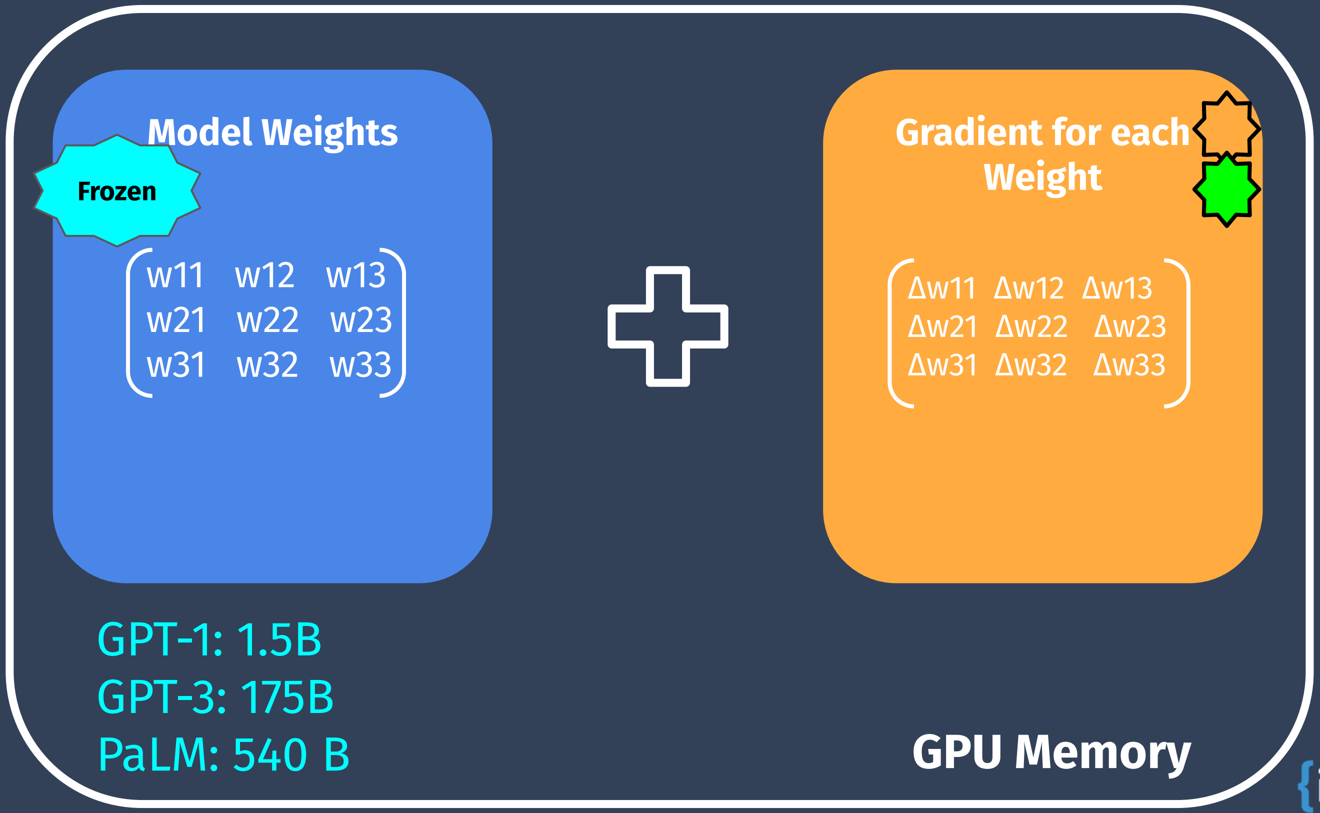
GPU Memory

Regular Fine-Tuning

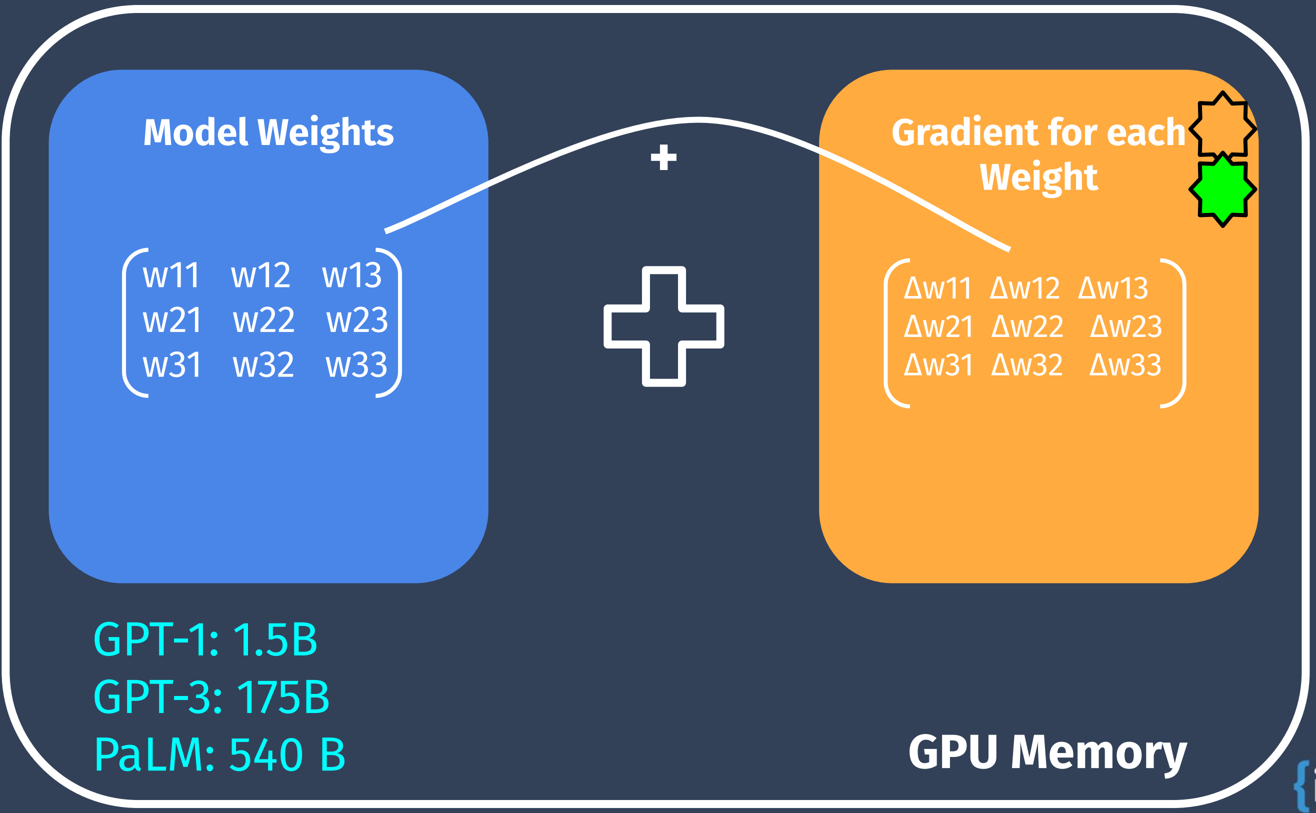
During fine-tuning we need to load ⇒



LoRA


Model Weights
Frozen
w11 w12 w13
w21 w22 w23
w31 w32 w33
Gradient for each Weight
Δw11 Δw12 Δw13
Δw21 Δw22 Δw23
Δw31 Δw32 Δw33
GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B
GPU Memory

LoRA


Model Weights
w11 w12 w13
w21 w22 w23
w31 w32 w33
+
Gradient for each Weight
Δw11 Δw12 Δw13
Δw21 Δw22 Δw23
Δw31 Δw32 Δw33
GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B
GPU Memory

LoRA

Does not it this double the number of parameters as well??

Model Weights

$$\begin{pmatrix} w11 & w12 & w13 \\ w21 & w22 & w23 \\ w31 & w32 & w33 \end{pmatrix}$$

+

Gradient for each Weight

$$\begin{pmatrix} \Delta w11 & \Delta w12 & \Delta w13 \\ \Delta w21 & \Delta w22 & \Delta w23 \\ \Delta w31 & \Delta w32 & \Delta w33 \end{pmatrix}$$

GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

GPU Memory

LoRA

Does not it this double the number of parameters as well??

No, not really ... at inference:

Model Weights

$$\begin{pmatrix} w11+\Delta w11 & w12+\Delta w12 & w13+\Delta w13 \\ w21+\Delta w21 & w22+\Delta w22 & w23+\Delta w23 \\ w31+\Delta w31 & w32+\Delta w32 & w33+\Delta w33 \end{pmatrix}$$

=

Model Weights

$$\begin{pmatrix} w11 & w12 & w13 \\ w21 & w22 & w23 \\ w31 & w32 & w33 \end{pmatrix}$$

+

Gradient for each Weight

$$\begin{pmatrix} \Delta w11 & \Delta w12 & \Delta w13 \\ \Delta w21 & \Delta w22 & \Delta w23 \\ \Delta w31 & \Delta w32 & \Delta w33 \end{pmatrix}$$

GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

GPU Memory

LoRA

Ok but we still need to tune that part?
At training:

Model Weights

$$\begin{pmatrix} w11 & w12 & w13 \\ w21 & w22 & w23 \\ w31 & w32 & w33 \end{pmatrix}$$

+

Gradient for each Weight

$$\begin{pmatrix} \Delta w11 & \Delta w12 & \Delta w13 \\ \Delta w21 & \Delta w22 & \Delta w23 \\ \Delta w31 & \Delta w32 & \Delta w33 \end{pmatrix}$$

GPT-1: 1.5B
GPT-3: 175B
PaLM: 540 B

GPU Memory

LoRA Tuning

$$\Delta W = \underbrace{\begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} \\ \Delta w_{21} & \Delta w_{22} & \Delta w_{23} \\ \Delta w_{31} & \Delta w_{32} & \Delta w_{33} \end{bmatrix}}_{d * k} = \underset{(d * r)}{B}\,\underset{(r * k)}{A}$$

$d = 3$, $k = 3$

$$= \underset{(3 * 1)}{\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}} \cdot \underset{(1 * 3)}{\begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}}$$

or

$$= \underset{(3 * 2)}{\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}} \underset{(2 * 3)}{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}}$$

LoRA Tuning

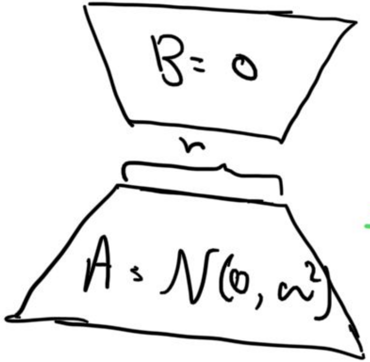
$$\Delta W = \begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} \\ \Delta w_{21} & \Delta w_{22} & \Delta w_{23} \\ \Delta w_{31} & \Delta w_{32} & \Delta w_{33} \end{bmatrix} = \underset{(d \times r)}{B}\,\underset{(r \times k)}{A} = \underset{(3 \times 1)}{\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}} \cdot \underset{(1 \times 3)}{(a_1\ a_2\ a_3)}$$

$d \times k$

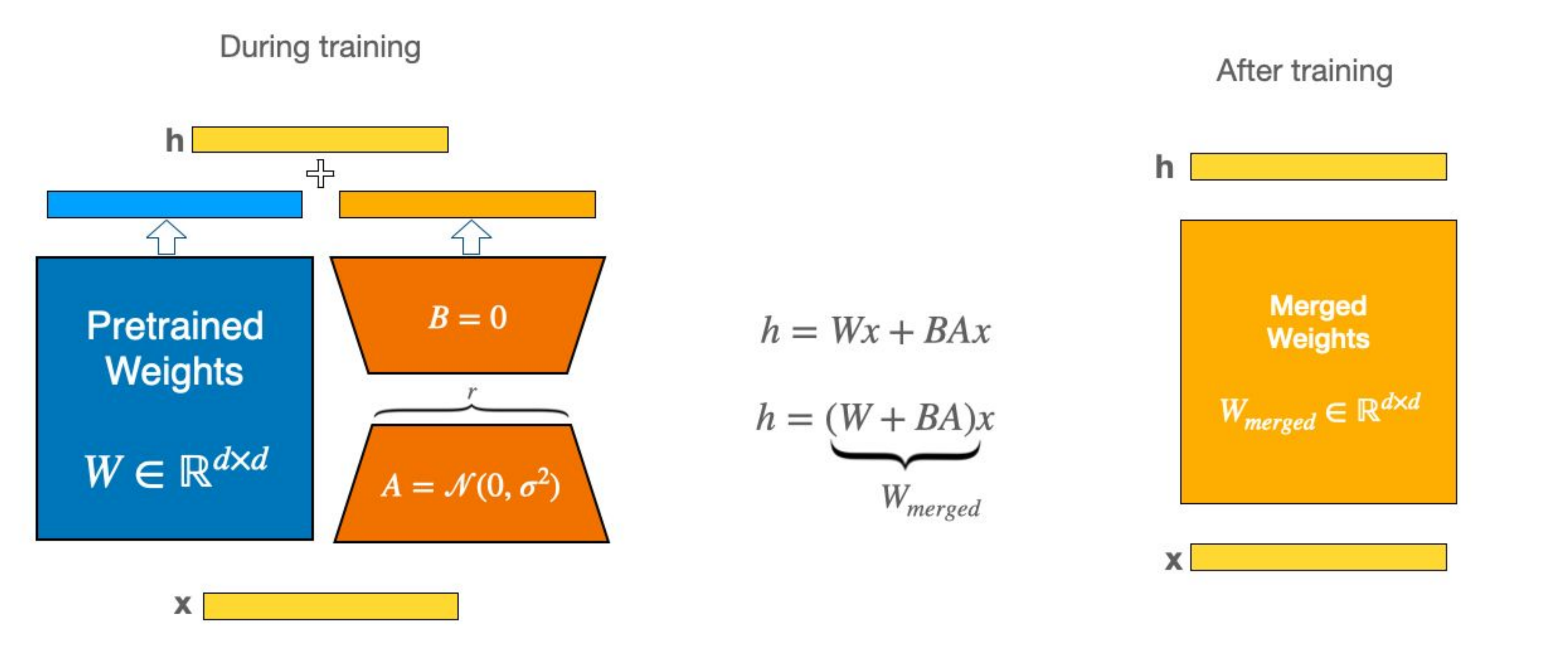
$3 \leftarrow d, \quad k \rightarrow 3$

or

$$= \underset{(3 \times 2)}{\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}} \underset{(2 \times 3)}{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}}$$



LoRA Tuning



During training
After training
h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
x
$h = Wx + BAx$
$h = \underbrace{(W + BA)}_{W_{merged}}x$
h
Merged Weights
$W_{merged} \in \mathbb{R}^{d \times d}$
x


QUIZ
TIME!

Quiz

LoRA Tuning

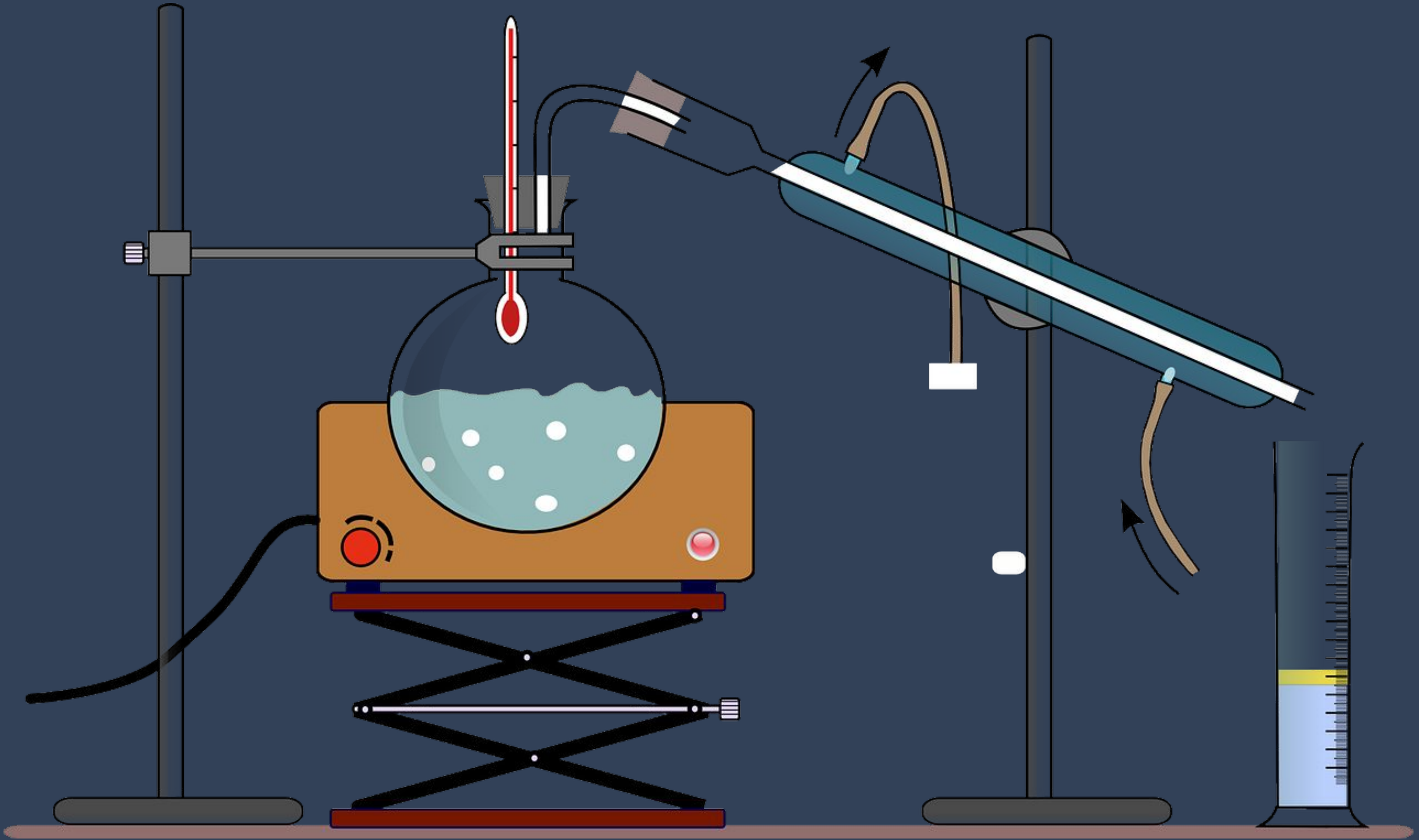
A) Improves Training Memory footprint
B) Improves Inference Memory footprint
C) A & B

Quiz

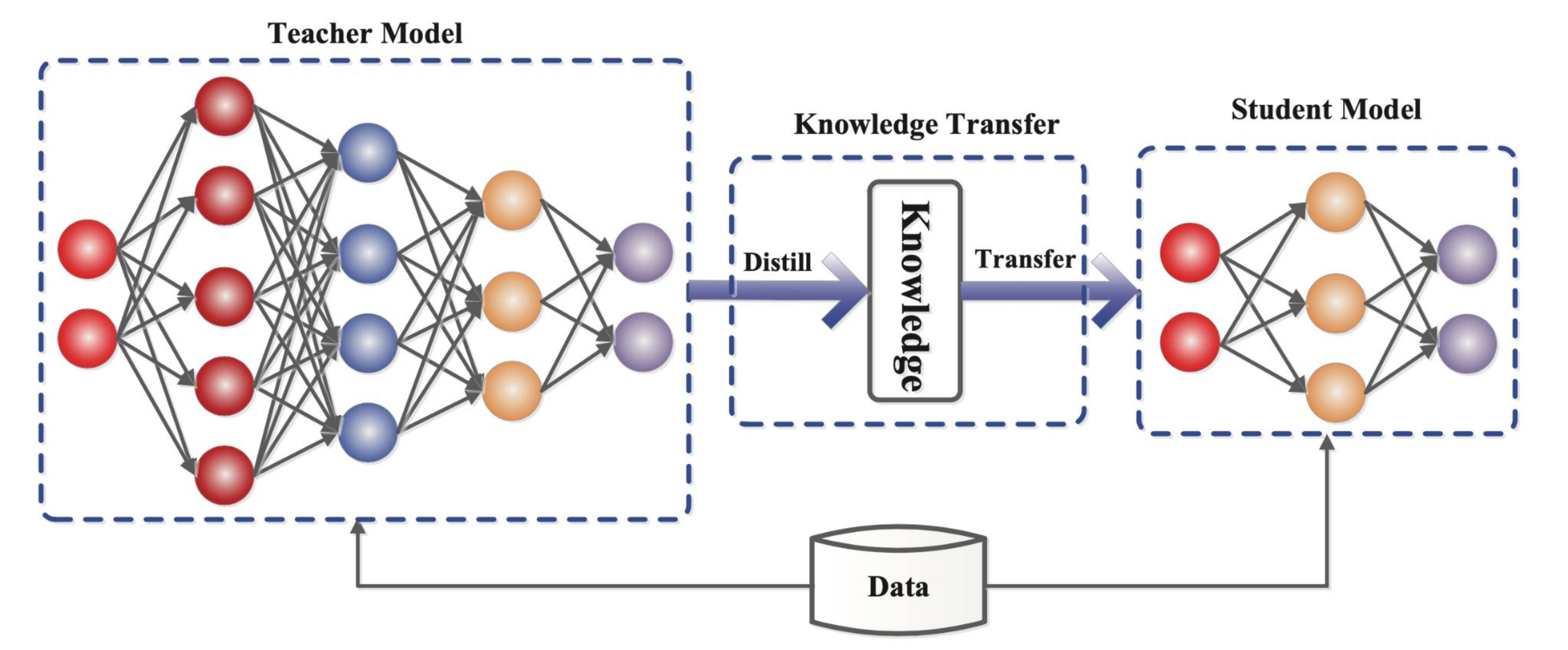
LoRA Tuning

A) Improves Training Memory footprint
B) Improves Inference Memory footprint
C) A & B

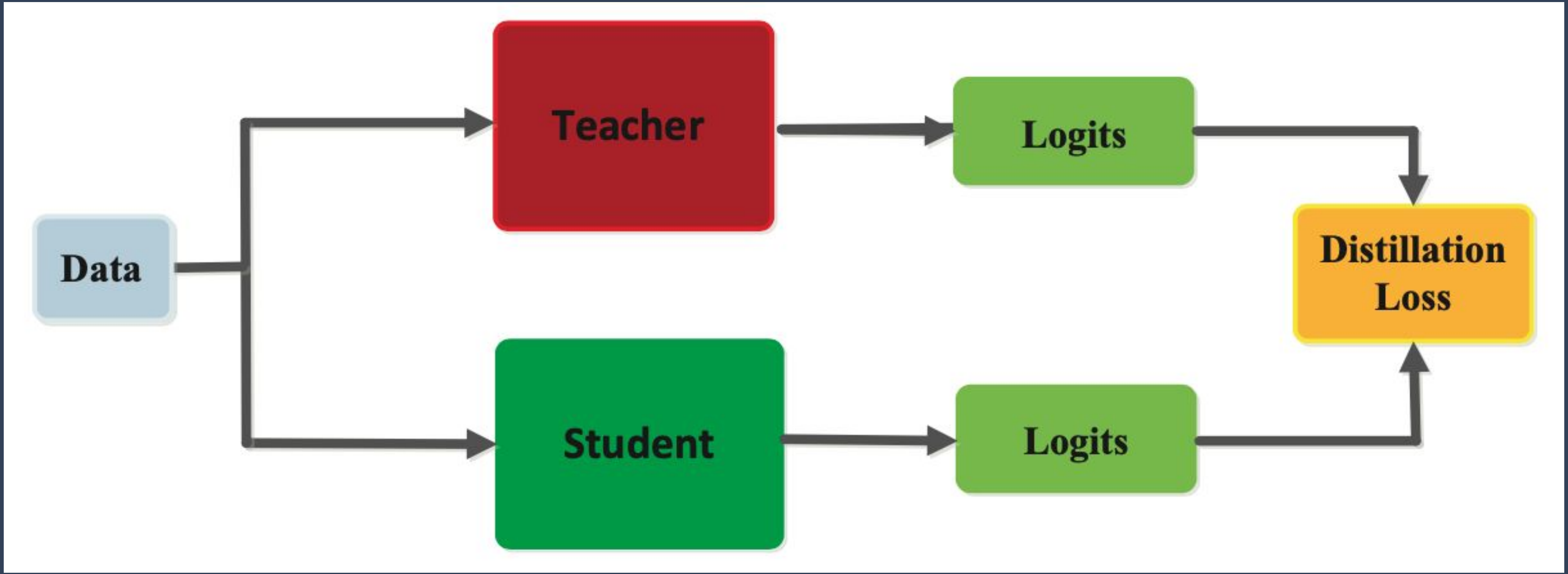
Knowledge Distillation



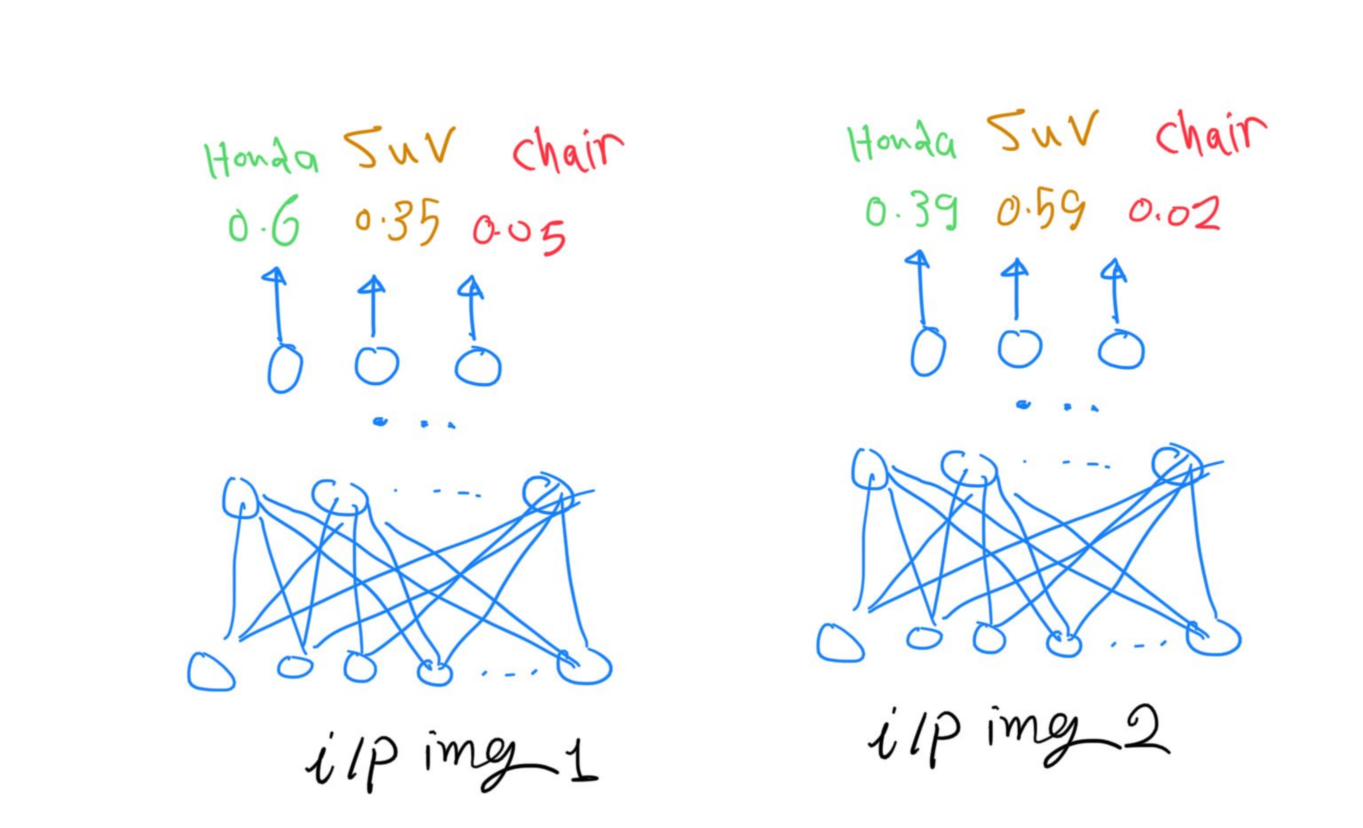
Knowledge Distillation



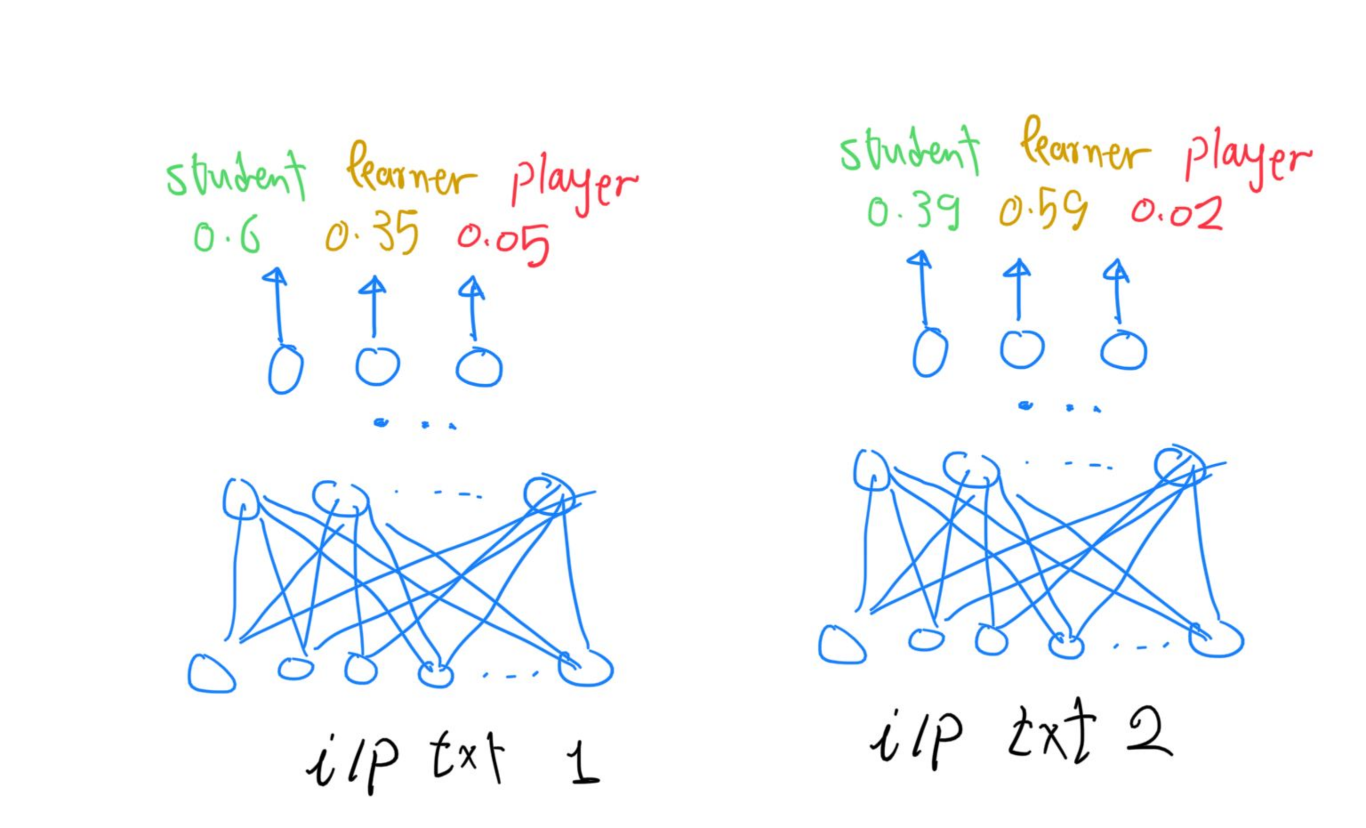
Response-based knowledge



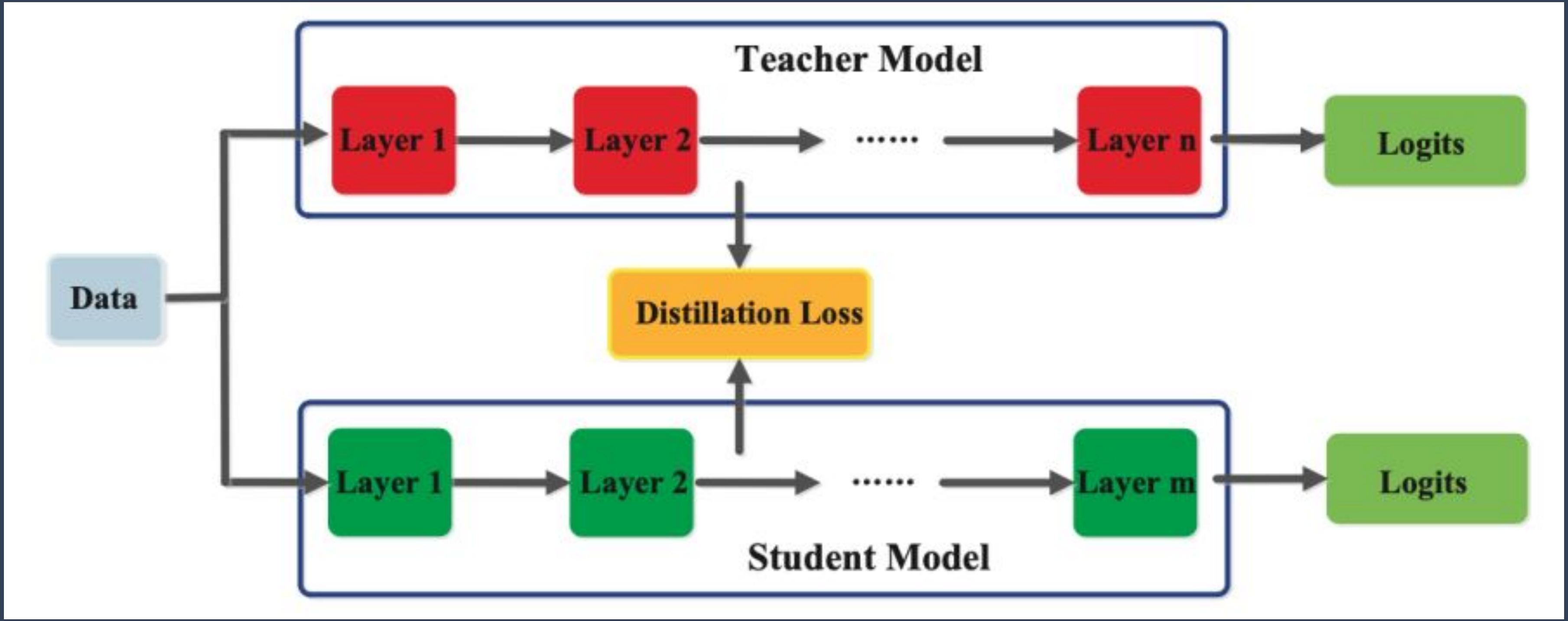
Distillation



Distillation



Feature-based knowledge



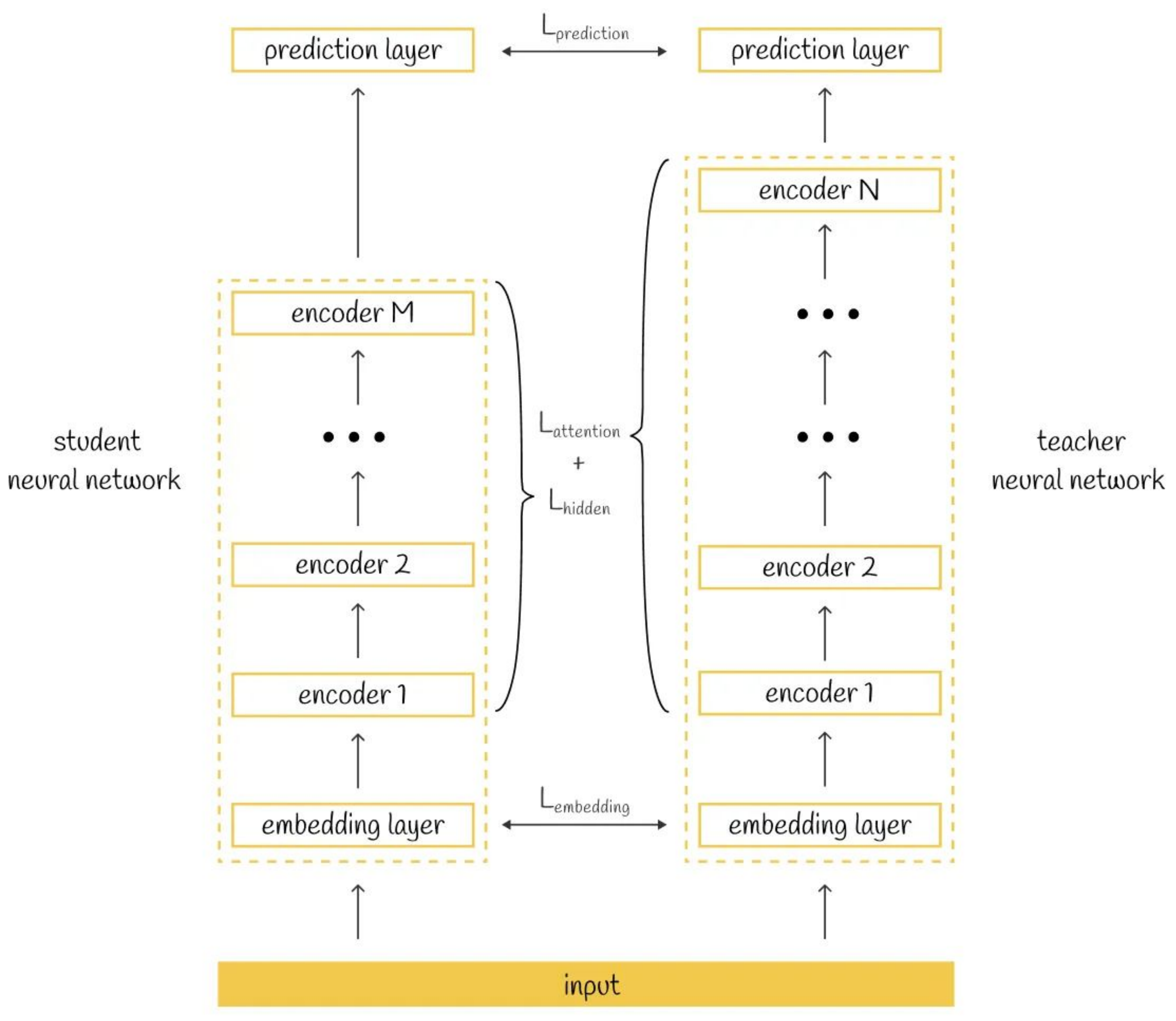
Demo Time

https://colab.research.google.com/drive/1MH2kIPtcGzu0g2VkQ8lcCHFVeR6nAls7

Distillation in LLM

- DistilBERT
  - The student model imitate the teacher model
  - Has about half the total number of parameters of BERT base and retains 95% of BERT's performances on the language understanding benchmark

TinyBert



prediction layer
$L_{prediction}$
prediction layer
encoder N
encoder M
student
neural network
$L_{attention}$
+
L_{hidden}
teacher
neural network
encoder 2
encoder 2
encoder 1
encoder 1
$L_{embedding}$
embedding layer
embedding layer
input

Distillation: Practical Considerations

- The student model will be good at the distillation task **only**
- You still need unlabeled data
  - In some enterprise setups you may not be allowed to use customer data
- Model refresh


QUIZ
TIME!

Quiz

Distillation Improves:

A) Training latency
B) Inference latency
C) A & B

Quiz

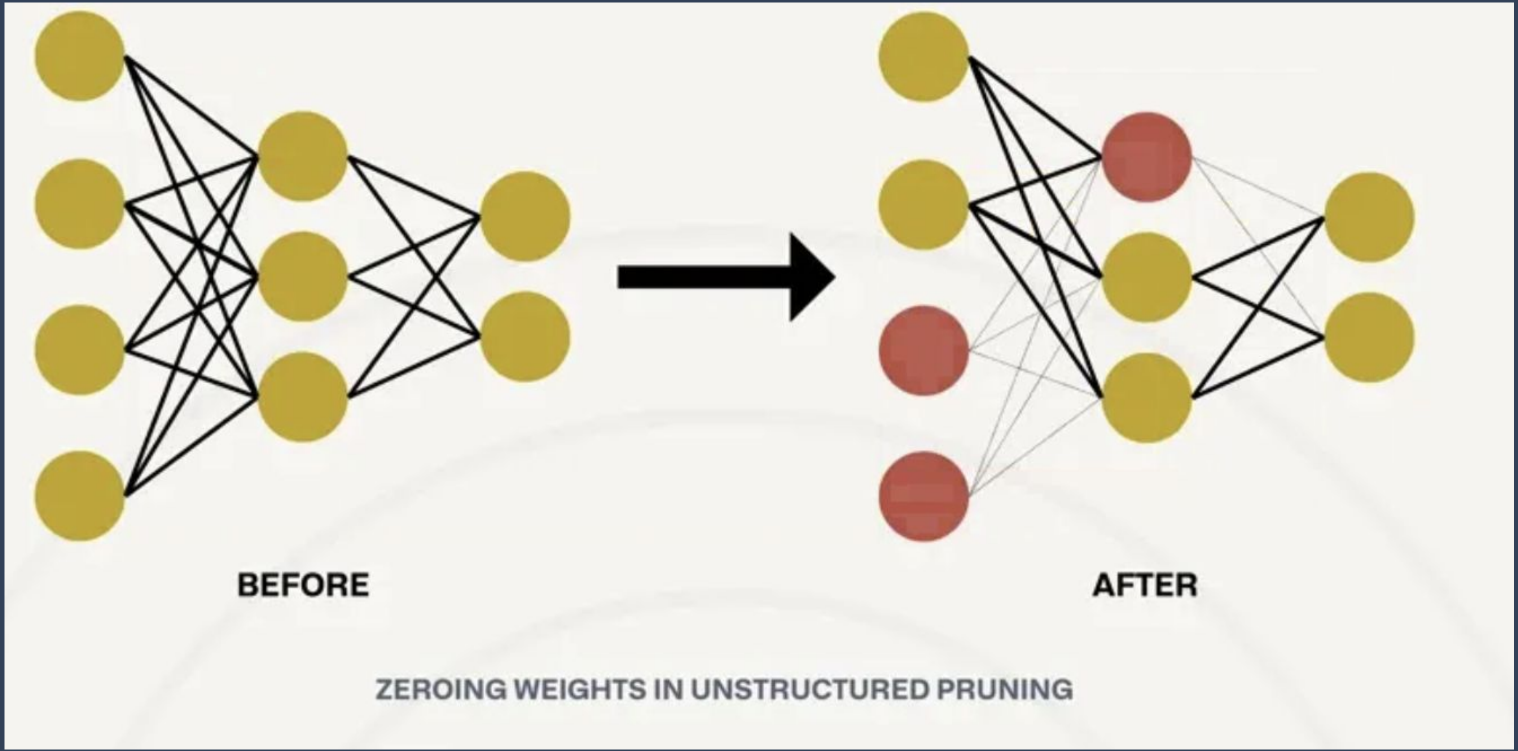
Distillation Improves:

A) Training latency
B) Inference latency
C) A & B

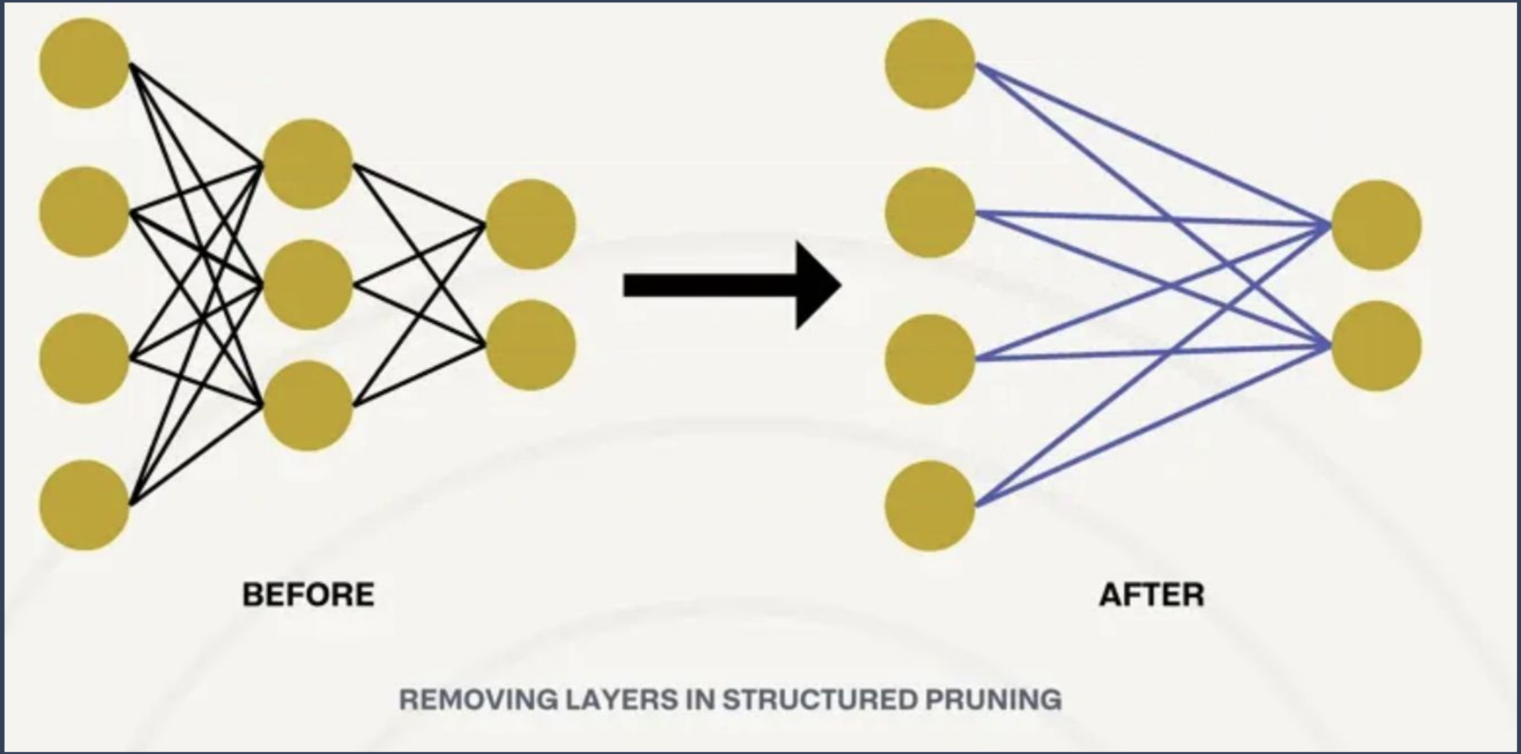
Model Pruning

- The reasoning is that:
  “Neural networks have an excess of parameters needed to generalize well and make accurate predictions”
- Then, we should drop these extra parameters

Model Pruning: Unstructured



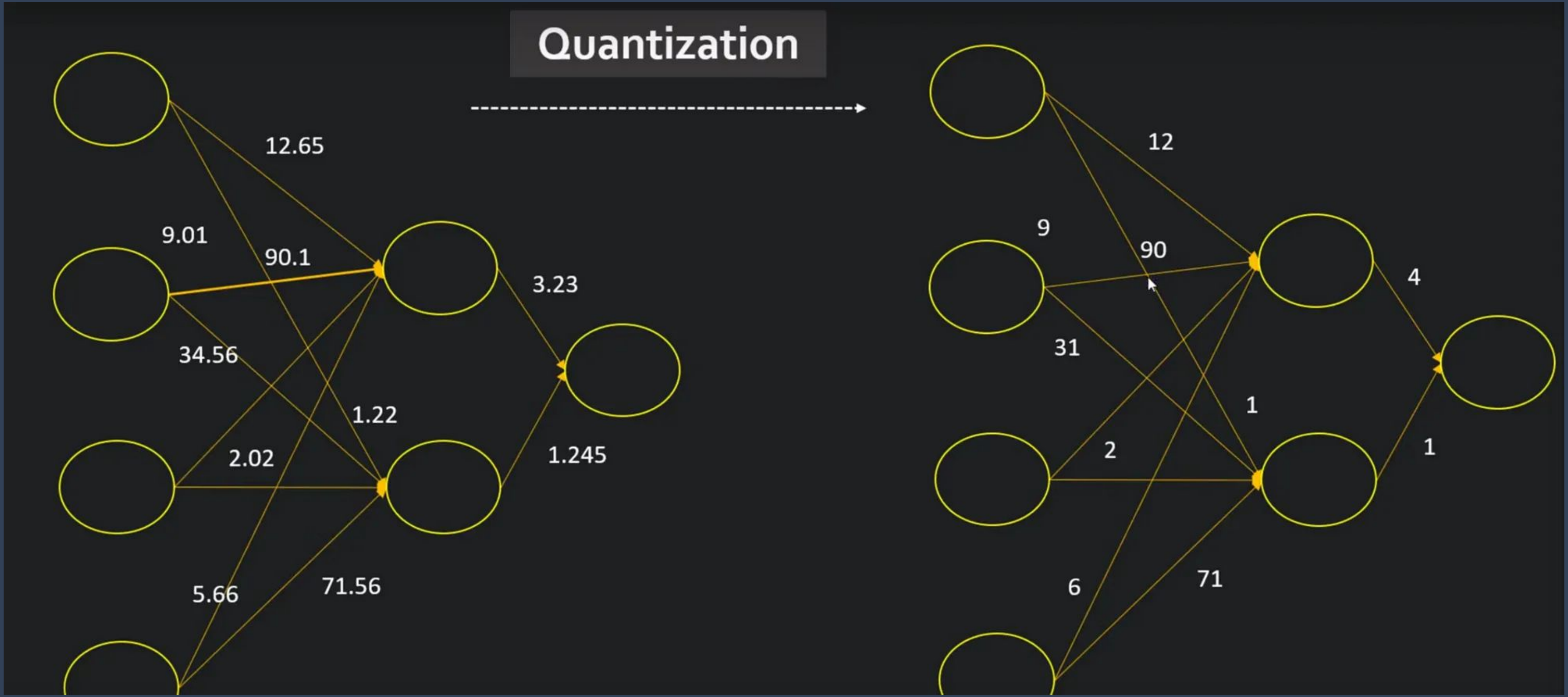
Model Pruning: Structured



Model Pruning

- At training time
  - ✅✅ More efficient as the model is trained with sparsity at training time
  - ❌❌Can be more complex
- Post training time
  - ✅✅ Simpler to implement and adjust per use-case
  - ❌❌May require fine-tuning

Model Quantization




QUIZ
TIME!

Quiz

Model Pruning and Quantization Improve

A) Training latency
B) Inference latency
C) A & B

Quiz

Model Pruning and Quantization Improve

A) Training latency
B) Inference latency
C) A & B

Is that it?

No, there are many many other optimizations

- Speculative decoding
- Linear attention
- Model parallelism
- ...

Summary and recap

- We went through the decoder details
- Sampling strategies
- Optimizations
    - KV Caching
    - LoRA
    - Distillation
    - Pruning
    - Quantizations

Questions & Discussions

References

- https://dugas.ch/artificial_curiosity/GPT_architecture.html
- https://towardsdatascience.com/large-language-models-gpt-1-generative-pre-trained-transformer-7b895f296d3b
- https://medium.com/@yumo-bai/why-are-most-llms-decoder-only-590c903e4789
- https://medium.com/@javaid.nabi/all-you-need-to-know-about-llm-text-generation-03b138e0ed19
- https://mlabonne.github.io/blog/posts/2023-06-07-Decoding_strategies.html
- https://www.appypie.com/blog/hardware-requirements-for-llm-training#Hardware
- https://sujayskumar.com/2020/04/24/reducing-model-size/
- https://snorkel.ai/llm-distillation-demystified-a-complete-guide/
- https://neptune.ai/blog/knowledge-distillation
- https://medium.com/huggingface/distilbert-8cf3380435b5
- https://www.datature.io/blog/a-comprehensive-guide-to-neural-network-model-pruning
- https://www.youtube.com/watch?v=80bIUggRJf4
- https://medium.com/@joaolages/kv-caching-explained-276520203249
- https://huggingface.co/docs/peft/main/en/conceptual_guides/lora
- https://www.youtube.com/watch?v=KEv-F5UkhxU
- https://www.youtube.com/watch?v=KEv-F5UkhxU