

A decorative graphic element consisting of a thick, multi-layered curved line that starts from the top right and curves downwards and to the left, ending near the center of the slide. The line has a gradient of colors, transitioning from light blue at the top to a pale green at the bottom.

# Understanding the Stages of Large Language Model Training

Sandipan Dandapat

Microsoft India

# Content

## Background

## Training LLM

- Base Model
  - Transformer
- Supervised Fine Tuning
- RLHF

## Scaling Laws

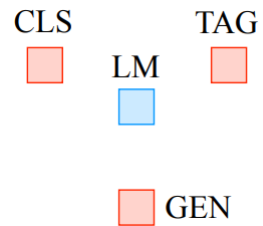
- SLMs

## Prompting

## Current Focus

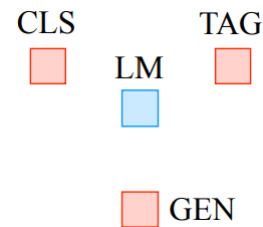
# Paradigms of NLP

## Feature Engineering



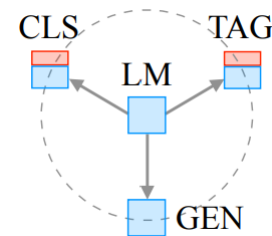
- Fully supervised **with Non-NN paradigm**
- Popular until 2015
- Manual features → linear/kernelized SVM/ CRF

## Architectural Engineering



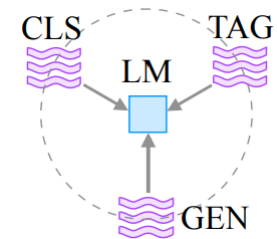
- Fully supervised **with NN paradigm**
- Popular 2013-2018
- No manually defined features
- Careful choice of architecture (LSTM/CNN)

## Objective Engineering



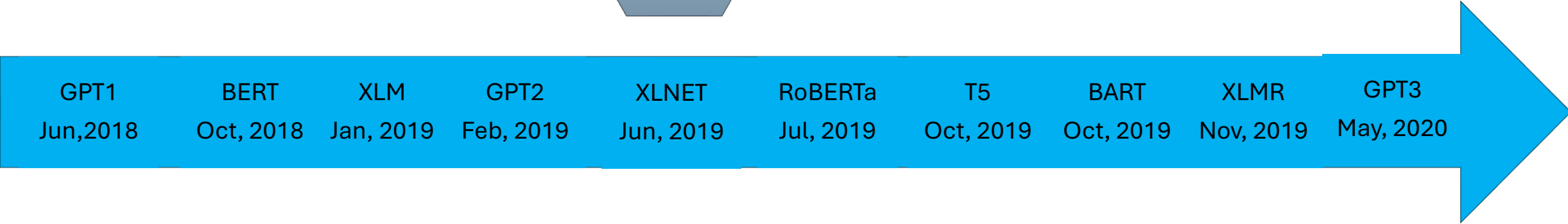
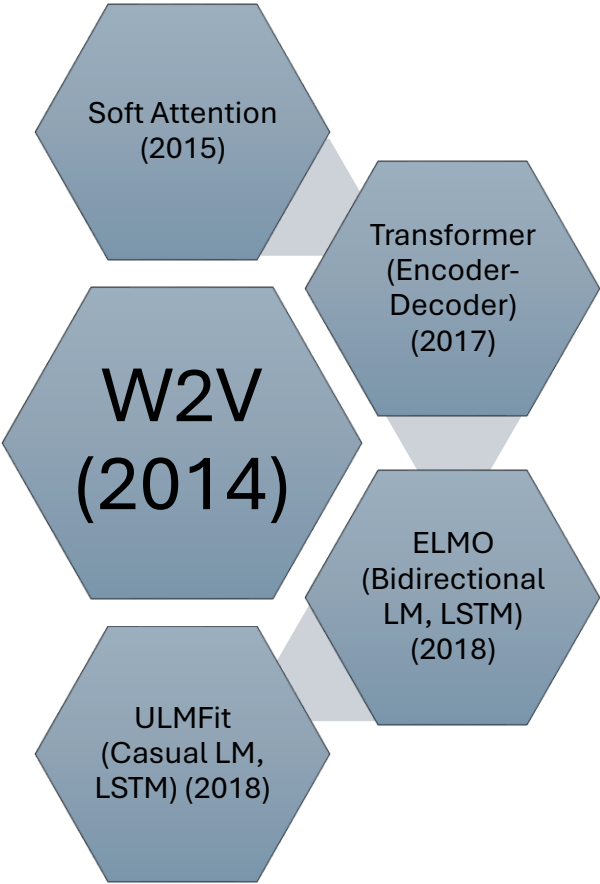
- **Pre-train, Fine-tune paradigm**
- Popular 2017-now
- Initialization of full model
- Engineer objective function

## Prompt Engineering



- **Pre-train, Prompt, Predict**
- Popular 2019-now
- Task entirely rely on LM
- Engineer of prompts is required

# Recent Past



# “few shot” GPT-3

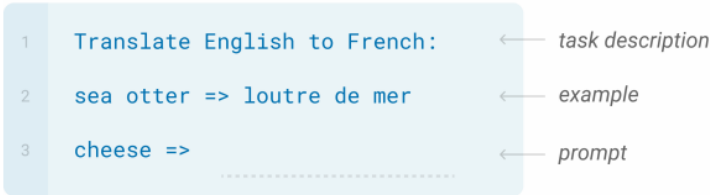
## Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



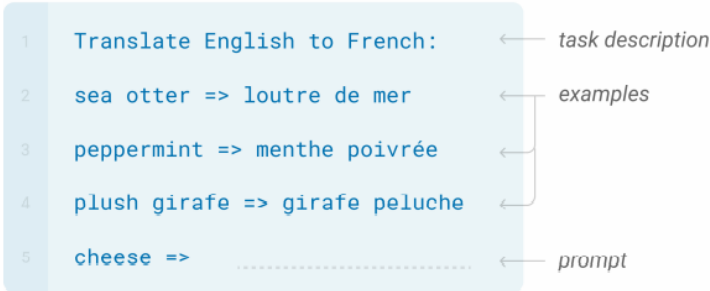
## One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



## Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



## Fine-tuning

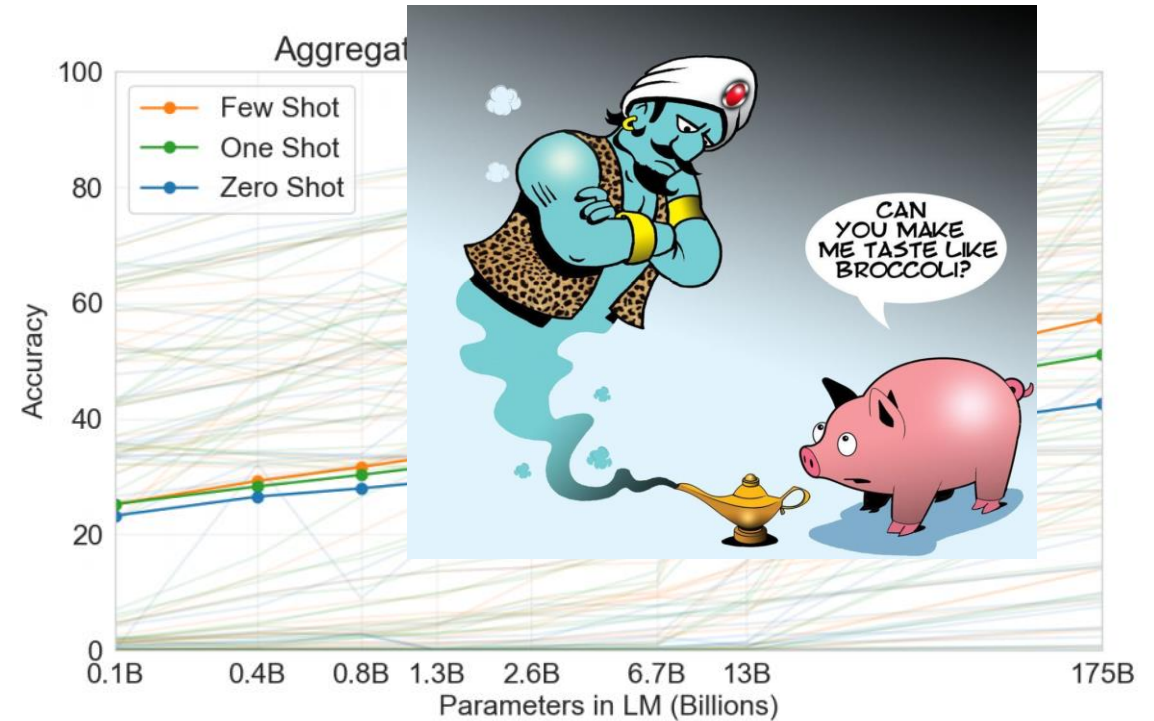
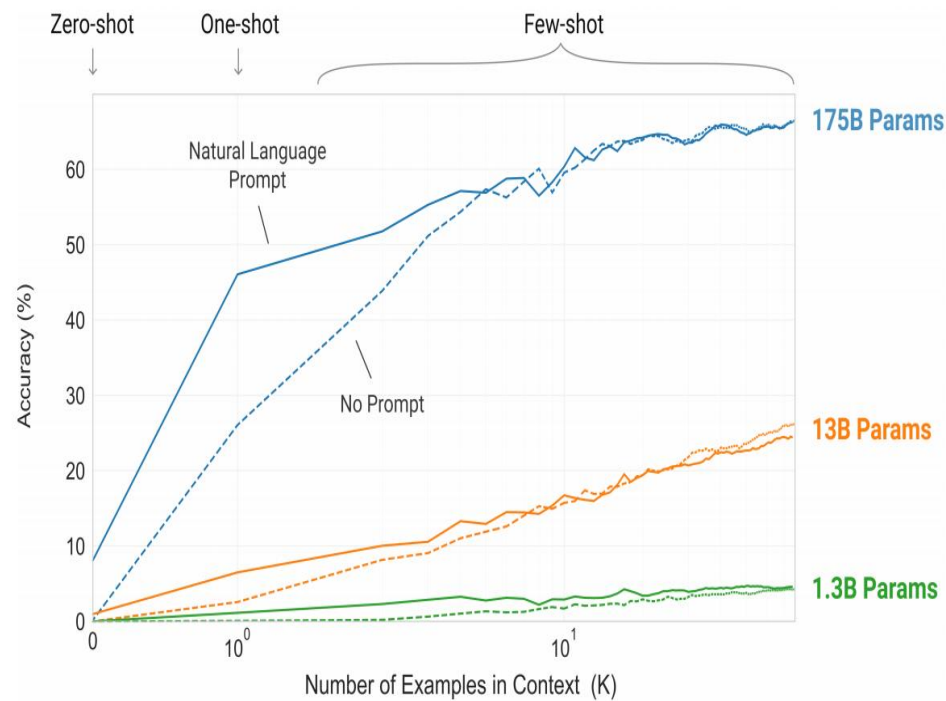
The model is trained via repeated gradient updates using a large corpus of example tasks.



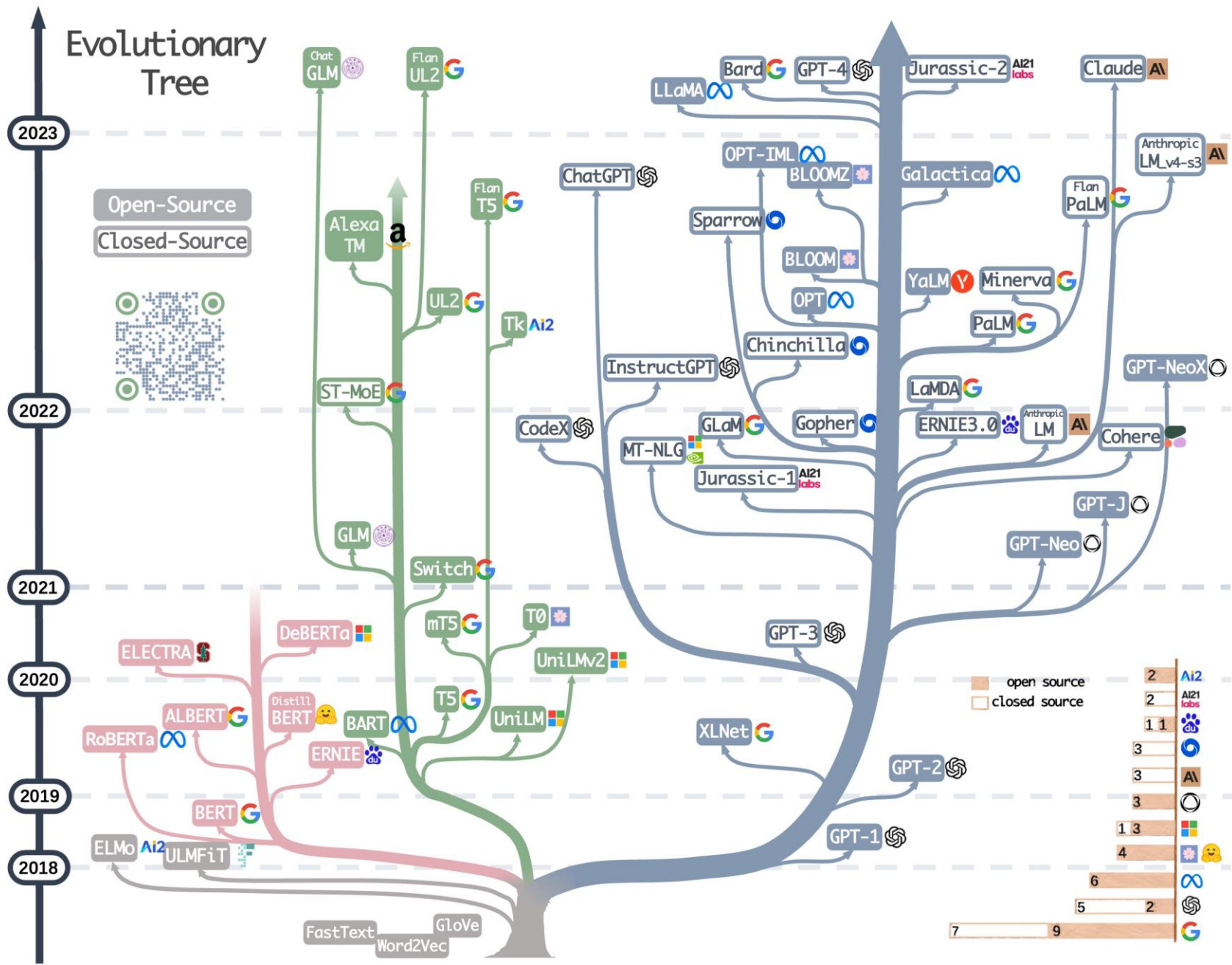
Fine-tuning  
BERT etc.

# In-Context Learning : Hypothesis

Will increasing the model size improve the performance of in-context learning methods?

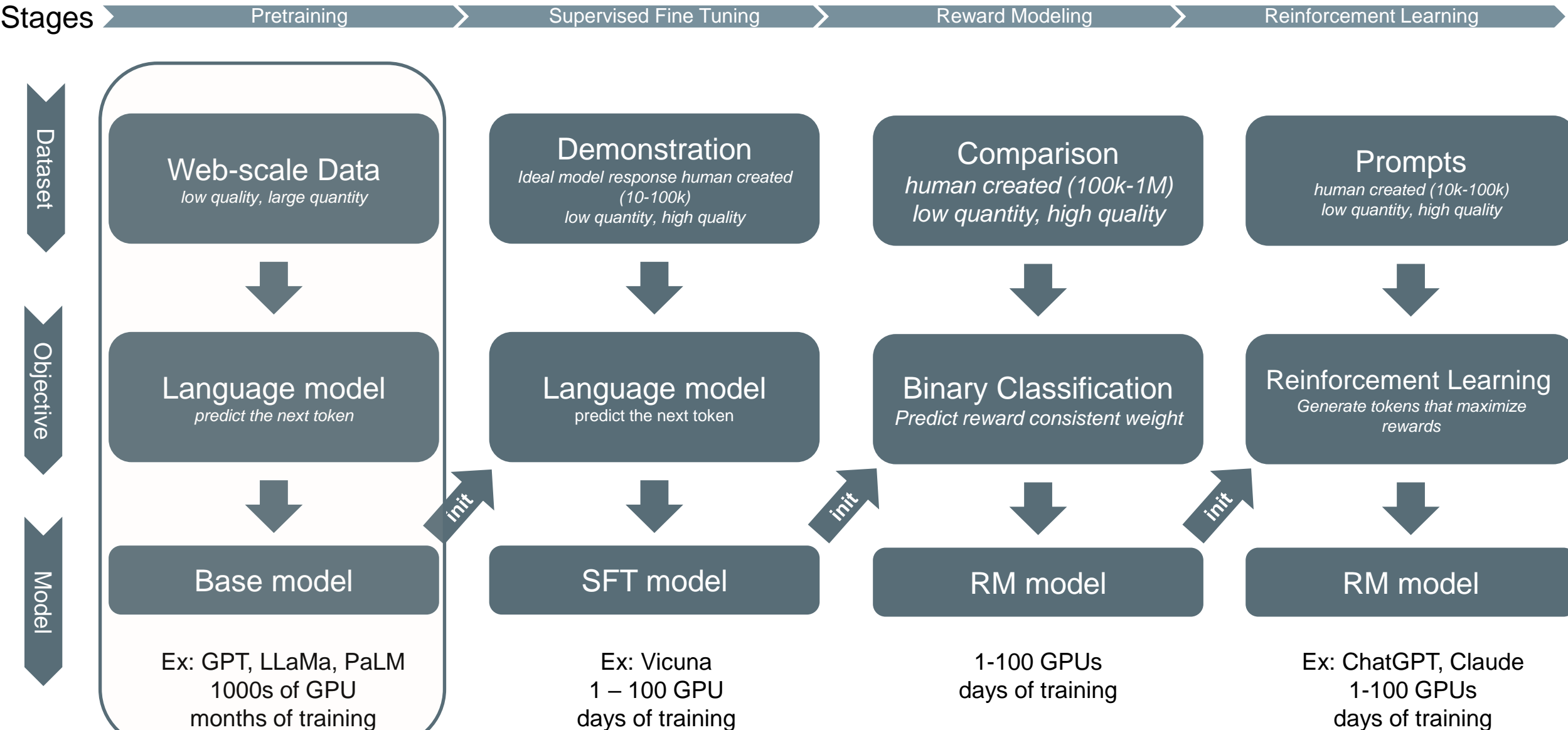


Now,



[Harnessing the Power of LLMs in Practice: A Survey on ChatGPT and Beyond, *arXiv preprint*, 2023]

# GPT Training Regime



[Figure Credit: Talk by Andrej Karpathy]



# Pretraining Data

LLaMa 3 (2023)

Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB

Table 1: **Pre-training data.** Data mixtures used for pre-training, for each subset we list the sampling proportion, number of epochs performed on the subset when training on 1.4T tokens, and disk size. The pre-training runs on 1T tokens have the same sampling proportion.

GPT 3 (2020)

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

[Language Models are Few-Shot Learners, OpenAI 2020]

[LLaMA: Open and Efficient Foundation Language Models, Meta AI 2023]

# Tokenization

OneDrive and SharePoint (ODSP) is a team in Microsoft with the mission to reinvent productivity by delivering the leading content collaboration platform for the mobile-first, cloud-first, intelligence powered world. Focus areas include OneDrive, SharePoint, Lists and Project Cortex.



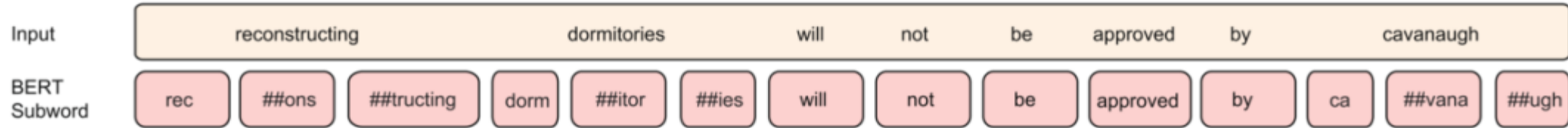
OneDrive and SharePoint (ODSP) is a team in Microsoft with the mission to reinvent productivity by delivering the leading content collaboration platform for the mobile-first, cloud-first, intelligence powered world. Focus areas include OneDrive, SharePoint, Lists and Project Cortex.



[3198, 24825, 290, 8734, 12727, 357, 3727, 4303, 8, 318, 257, 1074, 287, 5413, 351, 262, 4365, 284, 36608, 13714, 416, 13630, 262, 3756, 2695, 12438, 3859, 329, 262, 5175, 12, 11085, 11, 6279, 12, 11085, 11, 4430, 13232, 995, 13, 17061, 3006, 2291, 1881, 24825, 11, 8734, 12727, 11, 44968, 290, 4935, 44420, 13]

- Represent all text as a list of integers
- Typically,
  - 10-100k possible tokens
  - Byte Pair Encoding

# Sub-words and Byte Pair Encoding



Original Text: a b a b c b a b c c a b c a c b

**ab**→**D**    **D**    **D** c **b**    **D** c **c**    **D** c **a** **c** **b**

**Dc**→**E**    **D**    **E** **b**    **E** **c**    **E** **a** **c** **b**

**D**:  $X_1 = \mathbf{a}$ ;     $X_4 = X_1 \cdot X_2$ ;    **ab**→**D**  
 $X_2 = \mathbf{b}$ ;     $X_5 = X_4 \cdot X_3$ ;    **Dc**→**E**  
 $X_3 = \mathbf{c}$ ;

**S**:  $X_4, X_5, X_2, X_5, X_3, X_5, X_1, X_3, X_2$

1. Represent each character as a subword.
2. Count the frequency of the subword pairs.
3. The one with the highest frequency is represented as a new sub-word.
4. Loop through steps #2 and #3 till the number of unique sub-word reaches a maximum limit.

# Example Base Models

## GPT 3 (2020)

50,257 vocabulary size, 2048 context length

175B parameter

Trained on 300B tokens

Model Name	$n_{\text{params}}$	$n_{\text{layers}}$	$d_{\text{model}}$	$n_{\text{heads}}$	$d_{\text{head}}$	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	$6.0 \times 10^{-4}$
GPT-3 Medium	350M	24	1024	16	64	0.5M	$3.0 \times 10^{-4}$
GPT-3 Large	760M	24	1536	16	96	0.5M	$2.5 \times 10^{-4}$
GPT-3 XL	1.3B	24	2048	24	128	1M	$2.0 \times 10^{-4}$
GPT-3 2.7B	2.7B	32	2560	32	80	1M	$1.6 \times 10^{-4}$
GPT-3 6.7B	6.7B	32	4096	32	128	2M	$1.2 \times 10^{-4}$
GPT-3 13B	13.0B	40	5140	40	128	2M	$1.0 \times 10^{-4}$
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	$0.6 \times 10^{-4}$

**Table 2.1:** Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

Training: (rough order of magnitude to have in mind)

- O (1,000 – 10,000) V100 GPUs
- O (1) months of training
- O (1-10) \$M

## LLaMA (2023)

32,000 vocabulary size, 2048 context length

65B parameter

Trained on 1-1.4T tokens

params	dimension	$n$ heads	$n$ layers	learning rate	batch size	$n$ tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0T
13.0B	5120	40	40	$3.0e^{-4}$	4M	1.0T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4T

**Table 2:** Model sizes, architectures, and optimization hyper-parameters.

Training for 65B model:

- 2,048 A100 GPUs
- 21 days of training
- \$5M

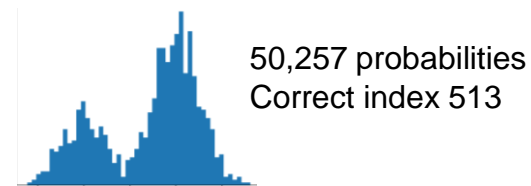
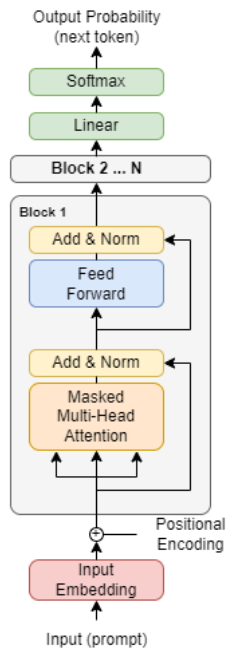
# Pretraining

- The input to the Transformer is in the shape of (B,T)
  - B is the batch size
  - T is the max token length

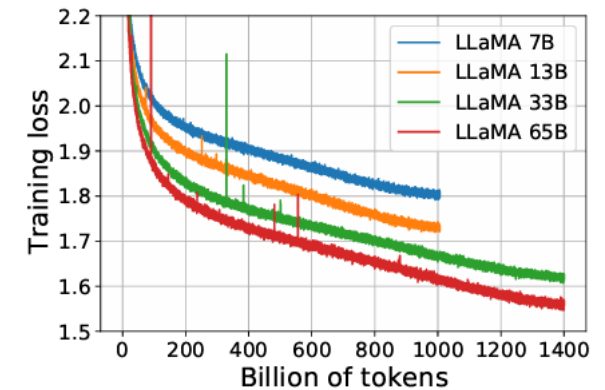
B=4

2043	40	41	...					...	281
91	1672	240	50256	128	437	513	50256		
							50256		

T=10



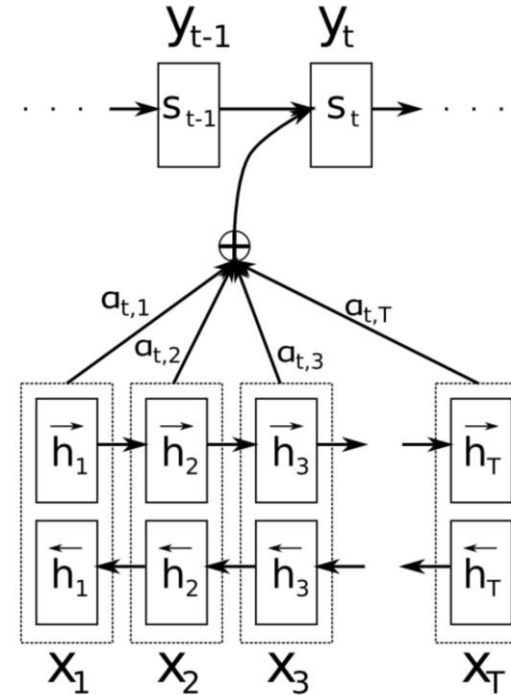
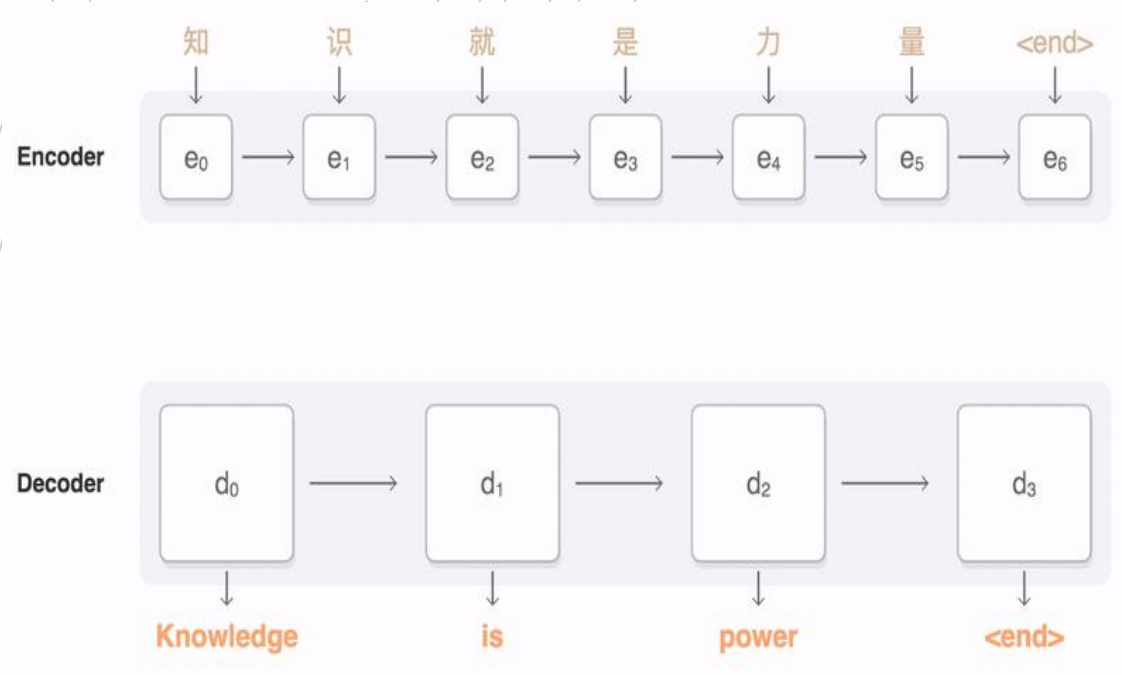
Each cell sees only cells before it in a row to predict the next cell



# Transformer

Reduce sequence computation  
Parallelization

# Attention Network



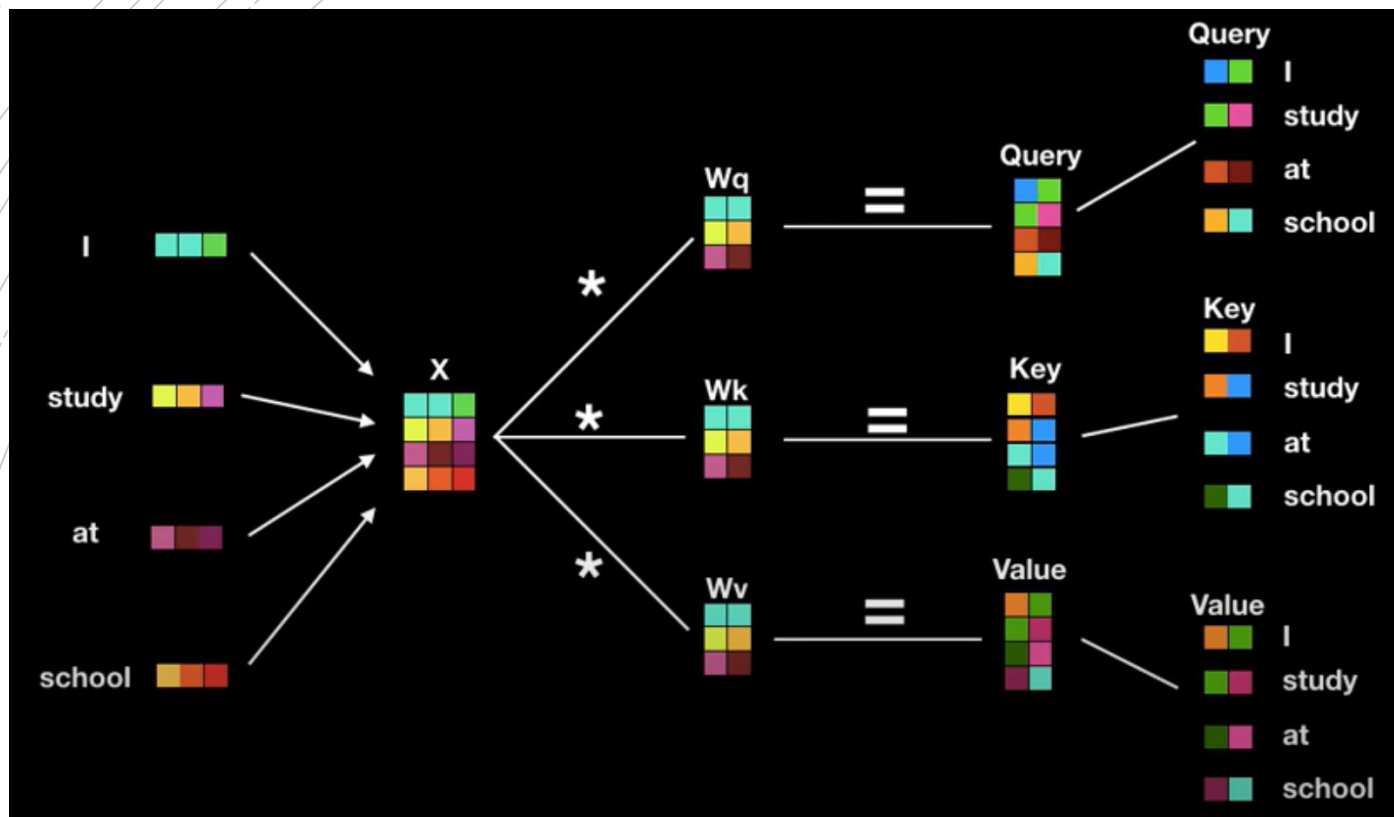
$$e_{ij} = a(s_{i-1}, h_j)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$


















$$p(y_i | y_1, \dots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_i, c_i)$$



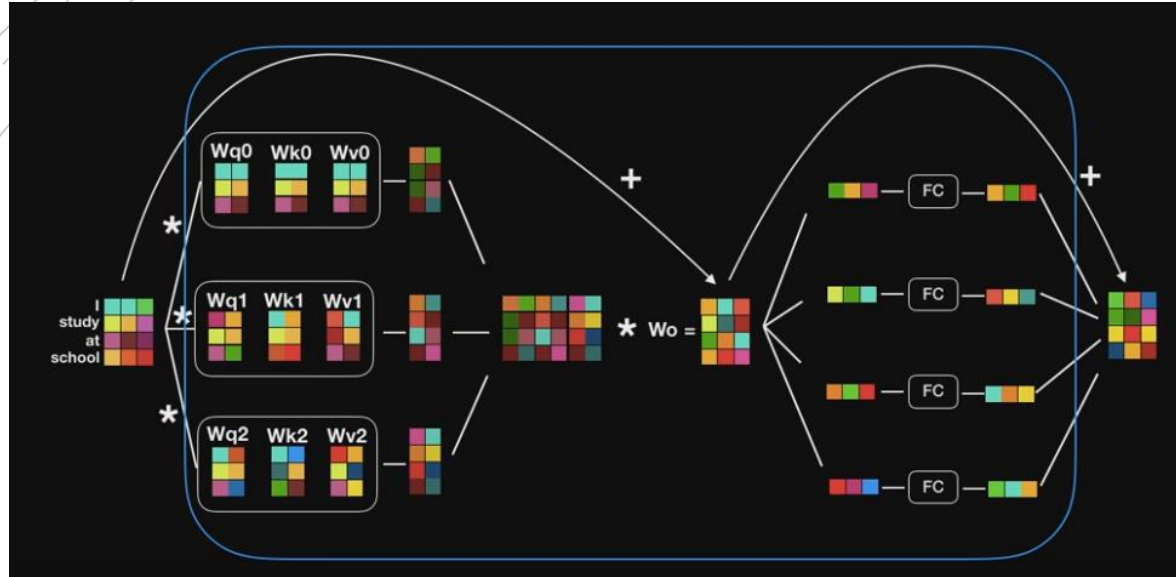
## Self Attention



	Query * Key <sup>T</sup>	Score	Softmax	Value	Softmax * Value	Σ Softmax * Value (Attention layer output)
I	I * I  *  = 130	130	0.92	I 		} 
	I * study  *  = 50	50	0.05	study 		
	I * at  *  = 20	20	0.02	at 		
	I * school  *  = 10	10	0.01	school 		

study	study * I  *  = 30	30	0.02			} 
	study * study  *  = 110	110	0.70			
	study * at  *  = 20	20	0.03			
	study * school  *  = 70	70	0.25			

# Encoder



I  
study  
at  
school



Encoder  
Layer#1



Encoder  
Layer#2



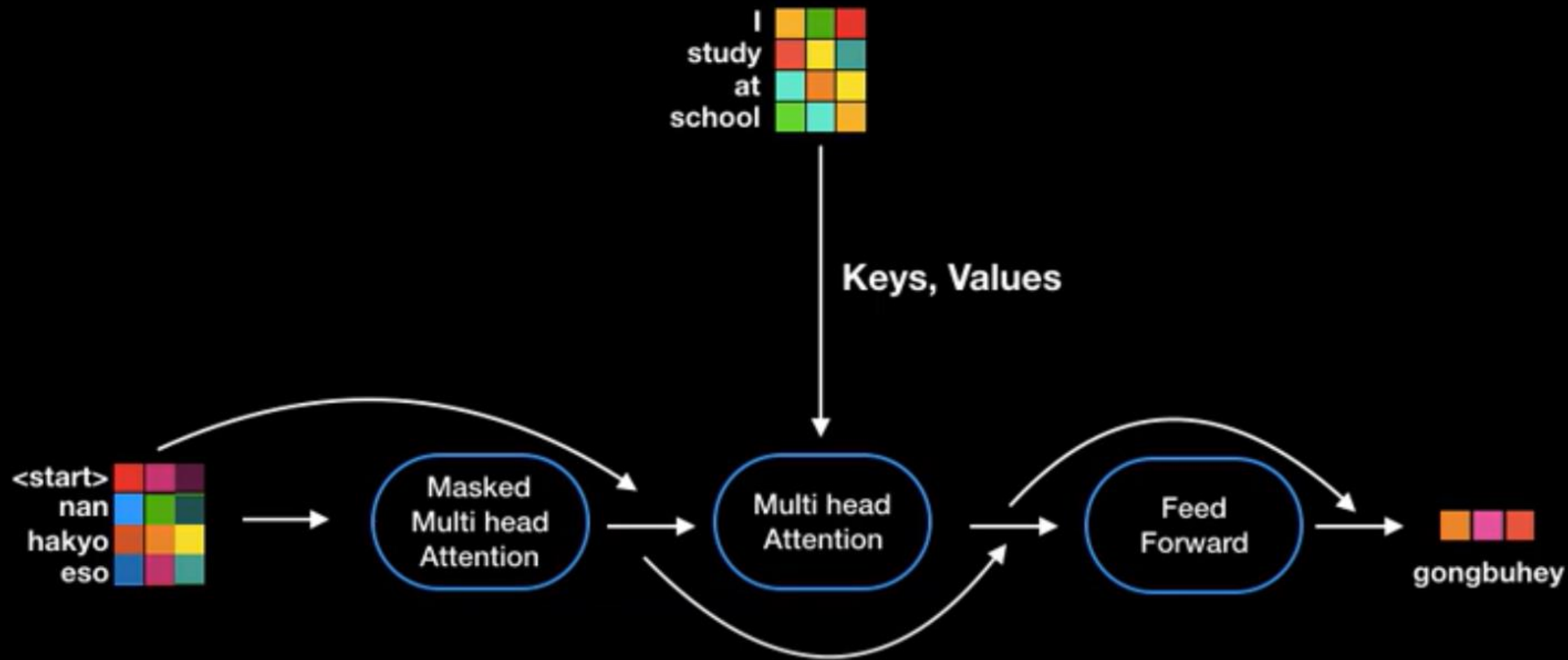
Encoder  
Layer#3

...

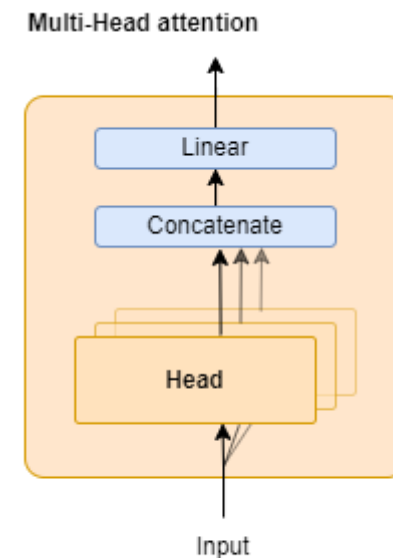
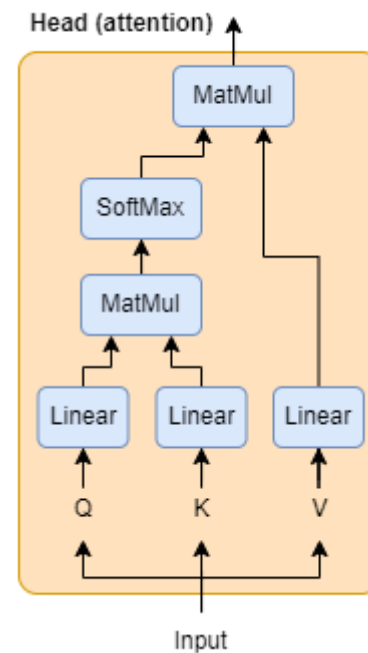
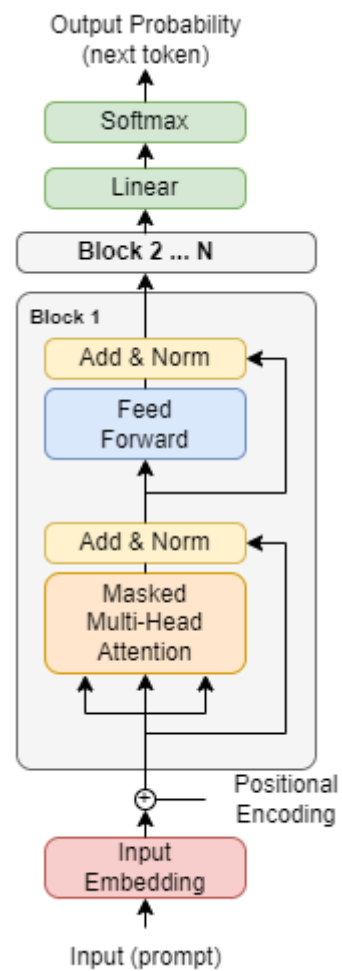
Encoder  
Layer#6



# Decoder

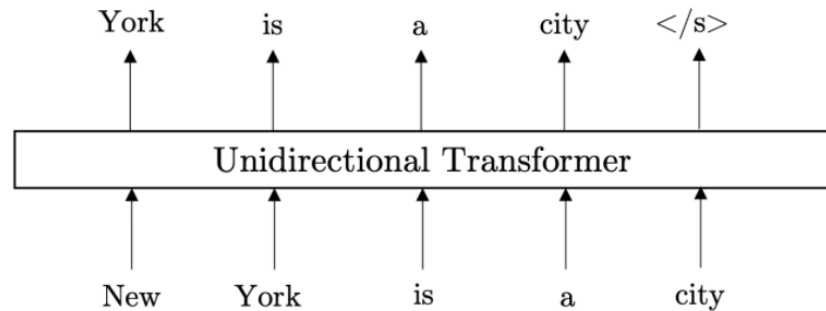


# Decoder-only Transformer

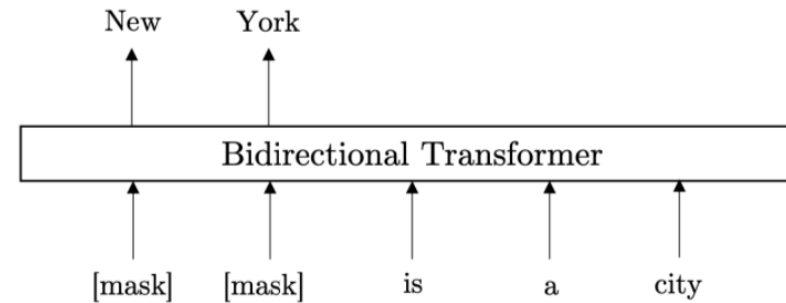


# AutoRegressive (AR) vs AutoEncoder (AE)

- Two competing approaches to LM
- AR: Use observation from past to predict the next value
- AE: Reconstruct the original data from corrupted input



$$\log p(\mathbf{x}) = \sum_{t=1}^T \log p(x_t | \mathbf{x}_{<t})$$

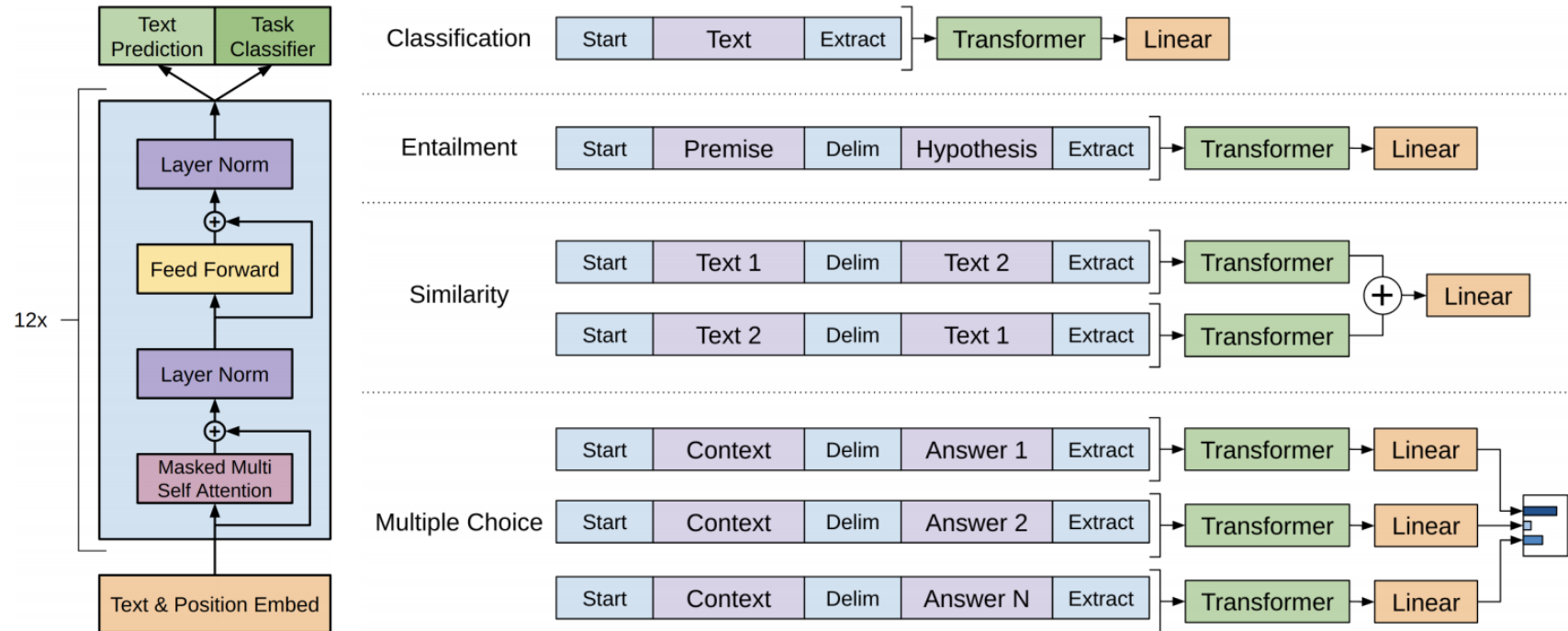


$$\log p(\bar{\mathbf{x}} | \hat{\mathbf{x}}) = \sum_{t=1}^T \text{mask}_t \log p(x_t | \hat{\mathbf{x}})$$

# GPT1: Base model can be finetuned for task specific application

learns general representation that is powerful to solve many tasks

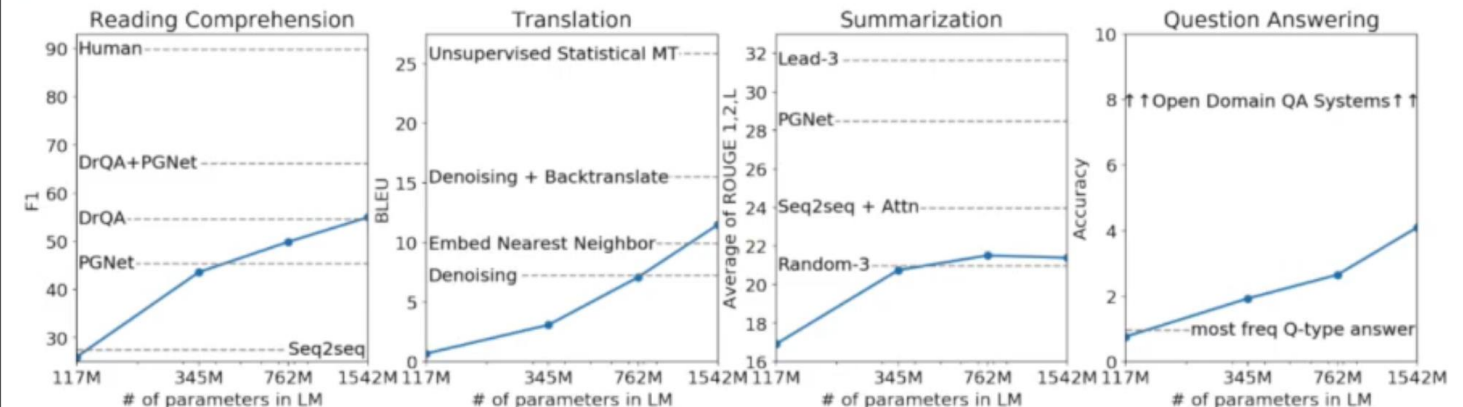
1. Pretraining on a large unsupervised data
  2. Finetuning on a small supervised data
- That's GPT-1



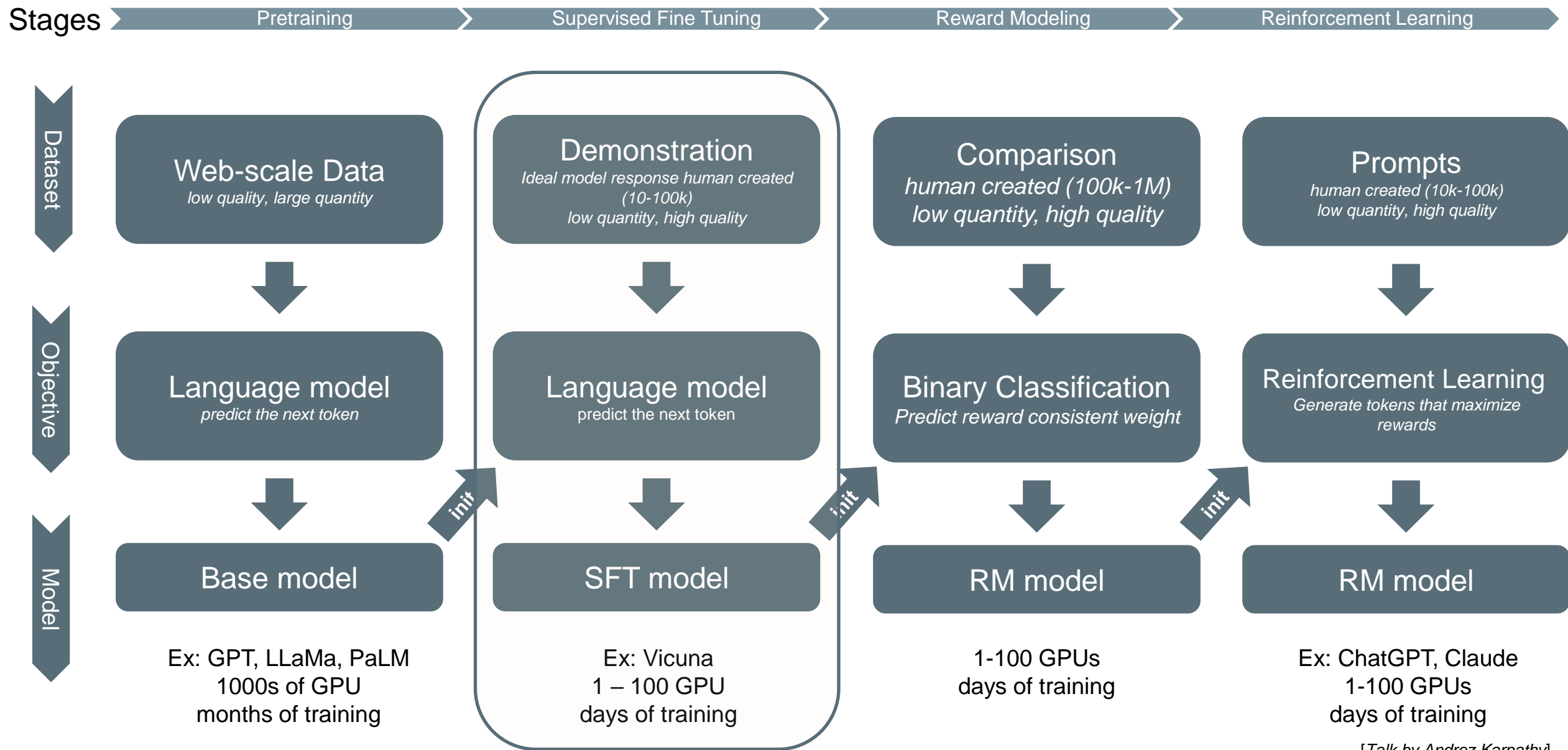
# GPT2: Base model can be prompted for task completion

<b>Context (passage and previous question/answer pairs)</b>	
<p>The 2008 Summer Olympics torch relay was run from March 24 until August 8, 2008, prior to the 2008 Summer Olympics, with the theme of “one world, one dream”. Plans for the relay were announced on April 26, 2007, in Beijing, China. The relay, also called by the organizers as the “Journey of Harmony”, lasted 129 days and carried the torch 137,000 km (85,000 mi) – the longest distance of any Olympic torch relay since the tradition was started ahead of the 1936 Summer Olympics.</p>	
<p>After being lit at the birthplace of the Olympic Games in Olympia, Greece on March 24, the torch traveled to the Panathinaiko Stadium in Athens, and then to Beijing, arriving on March 31. From Beijing, the torch was following a route passing through six continents. The torch has visited cities along the Silk Road, symbolizing ancient links between China and the rest of the world. The relay also included an ascent with the flame to the top of Mount Everest on the border of Nepal and Tibet, China from the Chinese side, which was closed specially for the event.</p>	
Q: What was the theme	A: “one world, one dream”.
Q: What was the length of the race?	A: 137,000 km
Q: Was it larger than previous ones?	A: No
Q: Where did the race begin?	A: Olympia, Greece
Q: Is there anything notable about that place?	A: birthplace of Olympic Games
Q: Where did they go after?	A: Athens
Q: How many days was the race?	A: seven
Q: Did they visit any notable landmarks?	A: Panathinaiko Stadium
Q: And did they climb any mountains?	A:
<b>Model answer:</b> Everest	
<b>Turker answers:</b> unknown, yes, Yes, yes	

Table 16. Selected CoQA completion.



# GPT Training Regime

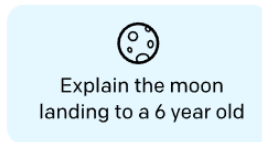




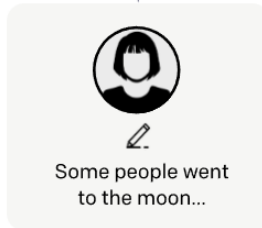
# Supervised Fine Tuning

**Collect demonstration data,  
and train a supervised policy.**

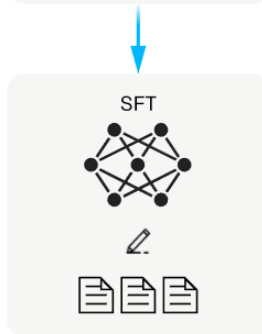
A prompt is  
sampled from our  
prompt dataset.



A labeler  
demonstrates the  
desired output  
behavior.

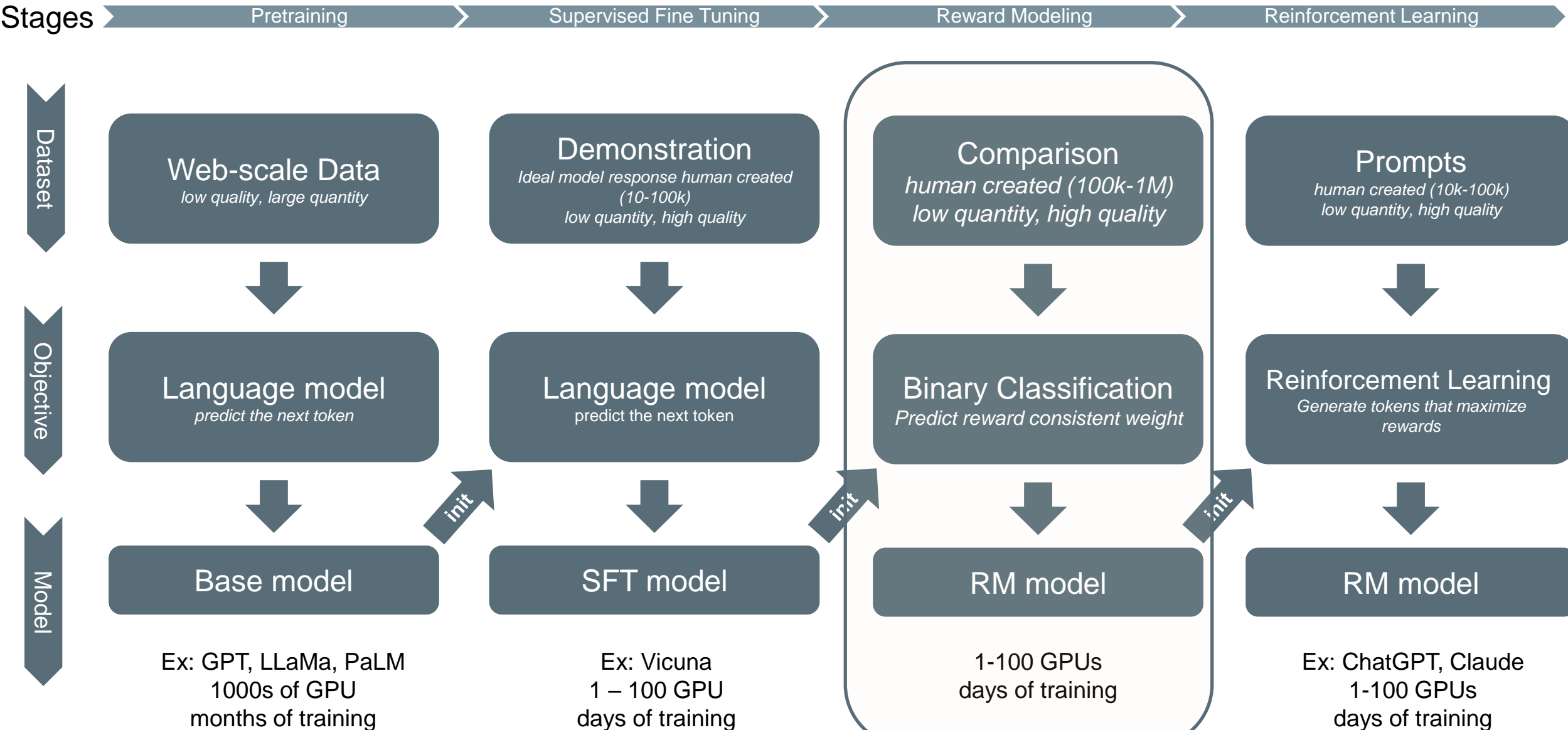


This data is used  
to fine-tune GPT-3  
with supervised  
learning.



- Done by Human
  - Given a prompt human contractors are asked to write the response
- Follow labelling instruction
- Same training objective (predict the next word)

# GPT Training Regime



[Talk by Andrez Karpathy]

# Reward Modelling Data

Write a python program to reverse a string

2

Here is a Python program that reverses a string using slicing:

```
string = "Hello World"
reversed_string = string[::-1]
print(reversed_string)
```

Output:

```
dlroW olleH
```

In this program, we have used slicing to reverse the string. The reversed string.

There are other ways to reverse a string in Python as well. You

`join()` and `reversed()` function, for loop and concatenation

3

Sure, I can write a python program to reverse a string. Here is one possible solution:

```
# Define a function to reverse a string
def reverse_string(string):
    # Initialize an empty string to store the reversed string
    reversed_string = ""
    # Loop through the original string from the end to the beginning
    for i in range(len(string) - 1, -1, -1):
        # Append each character to the reversed string
        reversed_string += string[i]
    # Return the reversed string
    return reversed_string

# Test the function with some examples
print(reverse_string("hello")) # olleh
print(reverse_string("python")) # nohtyp
print(reverse_string("Bing")) # gniB
```

1

Sure! Here's a simple Python program that takes a string as i

```
def reverse_string(s):
    return s[::-1]

input_string = input("Enter a string: ")
reversed_string = reverse_string(input_string)
print("Reversed string:", reversed_string)
```

This program defines a function `reverse_string` that takes reversed string using Python's slicing syntax. The program th calls the `reverse_string` function with the input string, and

# Reward Model Training

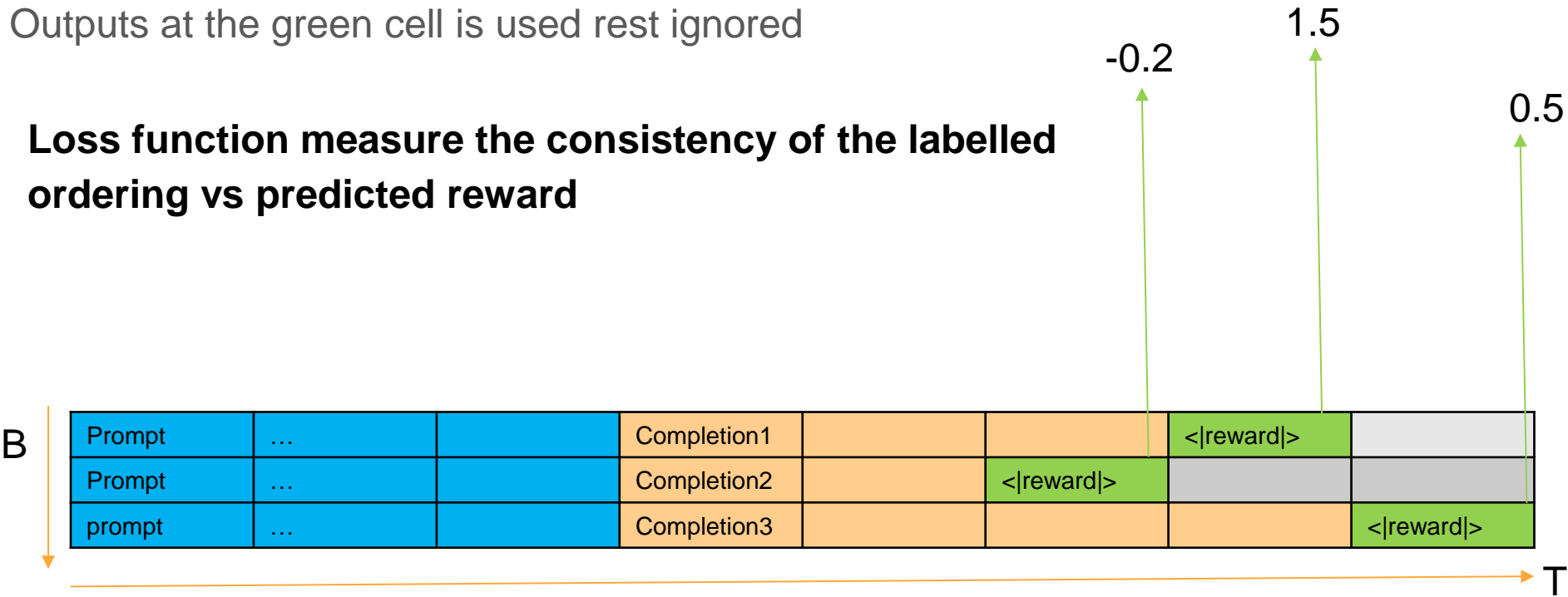
Blue are the prompt tokens

Yellow are the completion token

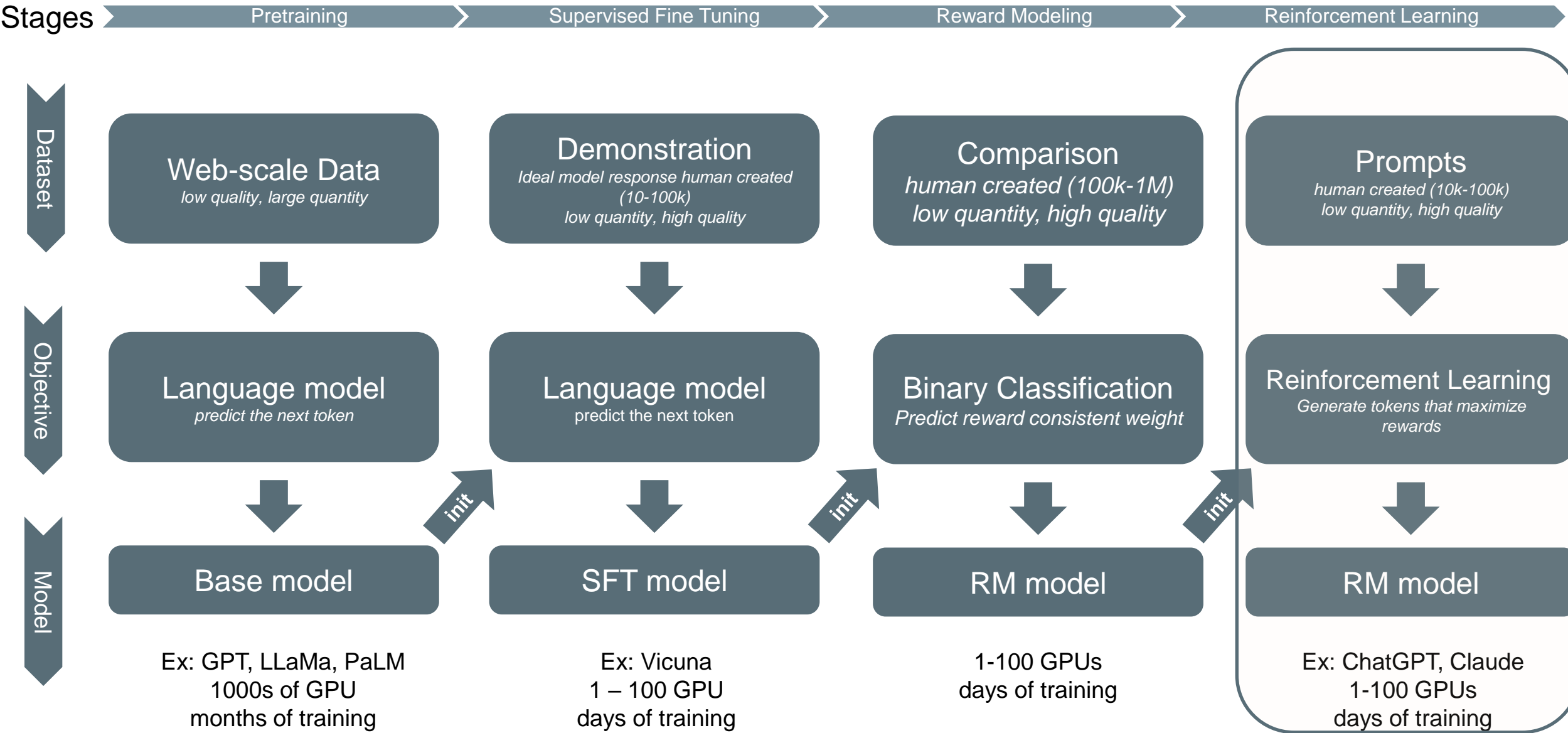
Green is the special <|reward|> token

Outputs at the green cell is used rest ignored

**Loss function measure the consistency of the labelled ordering vs predicted reward**



# GPT Training Regime



[Talk by Andrej Karpathy]

# RL Training

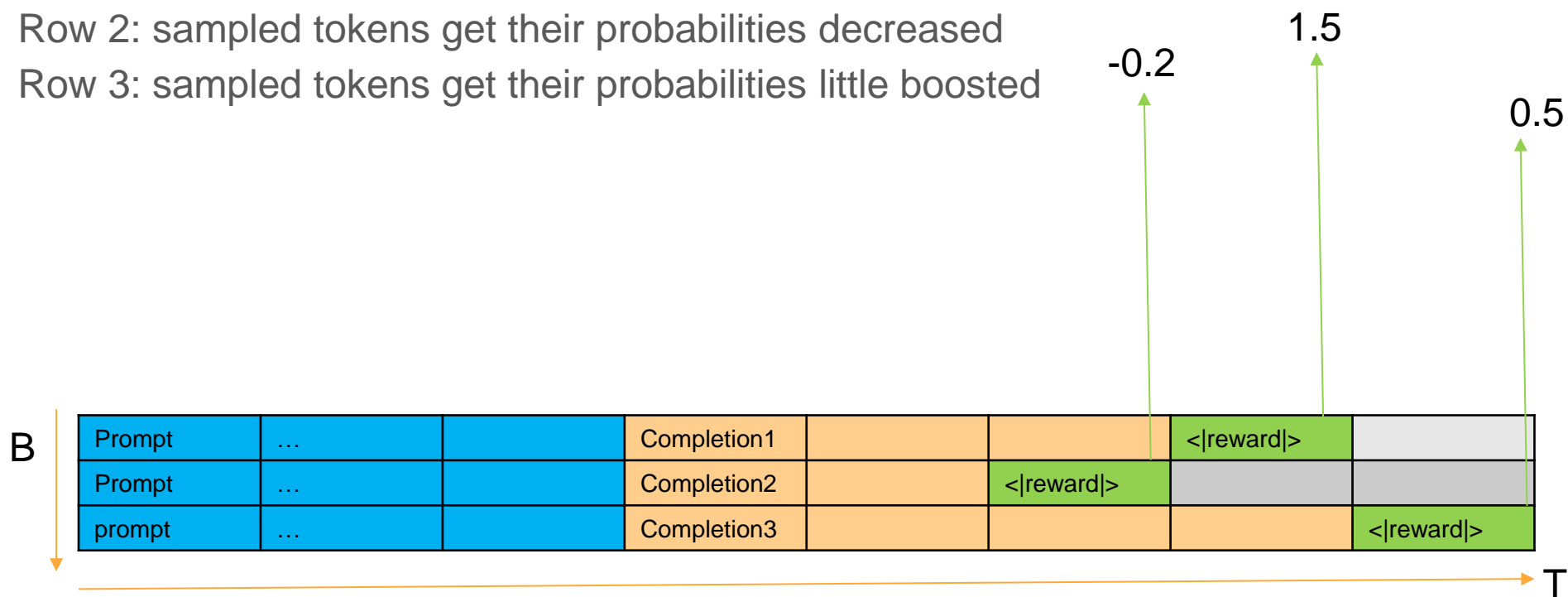
Yellow cells are trained, and rest ignored, RM is fixed

Example:

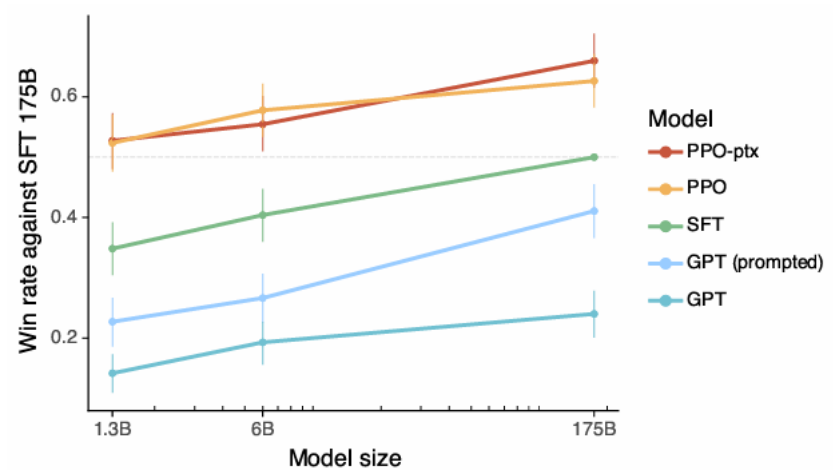
Row 1: sampled tokens get their probabilities boosted

Row 2: sampled tokens get their probabilities decreased

Row 3: sampled tokens get their probabilities little boosted



# Why RHLF?



Rank* (UB)	Model	Arena Score	95% CI	Votes	Organization	License
1	<a href="#">GPT-4o-2024-05-13</a>	1287	+3/-3	65952	OpenAI	Proprietary
1	<a href="#">GPT-4o-mini-2024-07-18</a>	1279	+9/-8	6849	OpenAI	Proprietary
2	<a href="#">Claude 3.5 Sonnet</a>	1272	+4/-3	36080	Anthropic	Proprietary
3	<a href="#">Gemini-Advanced-0514</a>	1266	+2/-3	48674	Google	Proprietary
4	<a href="#">Gemini-1.5-Pro-API-0514</a>	1261	+3/-3	58617	Google	Proprietary
5	<a href="#">Gemini-1.5-Pro-API-0409-Preview</a>	1257	+4/-3	55674	Google	Proprietary
5	<a href="#">GPT-4-Turbo-2024-04-09</a>	1257	+3/-2	77358	OpenAI	Proprietary
8	<a href="#">GPT-4-1106-preview</a>	1251	+3/-3	88859	OpenAI	Proprietary
8	<a href="#">Claude 3 Opus</a>	1248	+3/-2	148688	Anthropic	Proprietary
8	<a href="#">GPT-4-0125-preview</a>	1245	+3/-3	82138	OpenAI	Proprietary
8	<a href="#">Athene-70b</a>	1240	+10/-8	3477	NexusFlow	CC-BY-NC-4.0
10	<a href="#">Yi-Large-preview</a>	1240	+3/-3	50864	01 AI	Proprietary

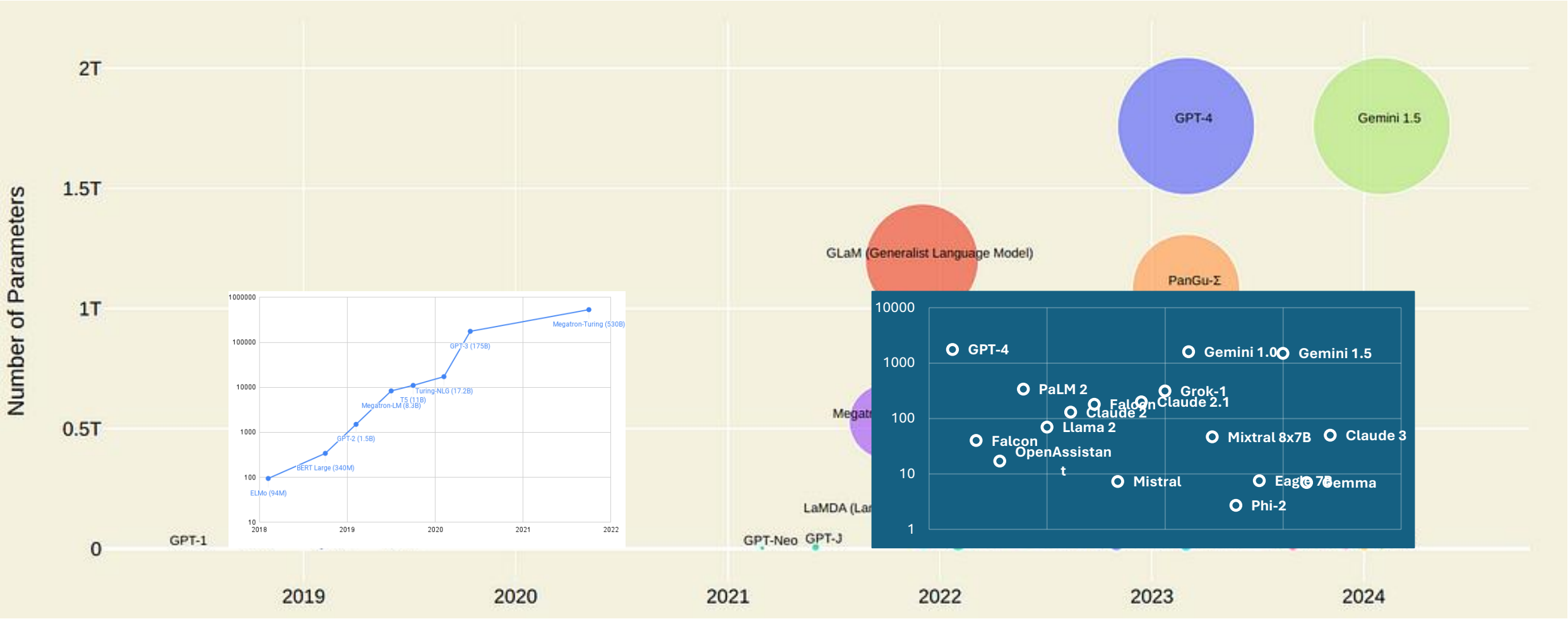
It works better

Easier to discriminate than generate

RLHF may have low entropy compared to base model

→ diversity in the output

# Trends in LLM Parameter



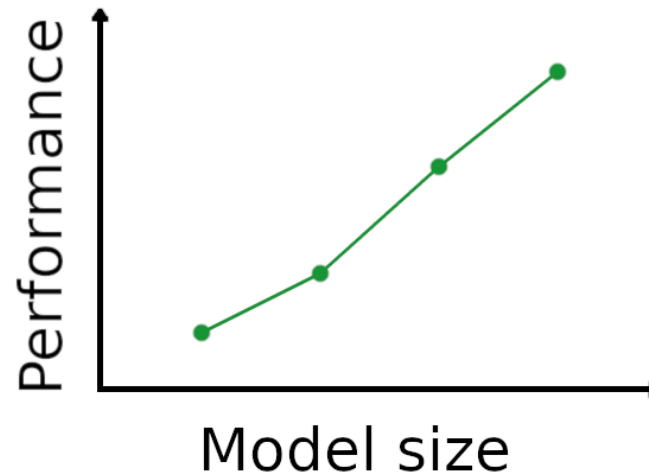


# Scaling Properties of LLMs

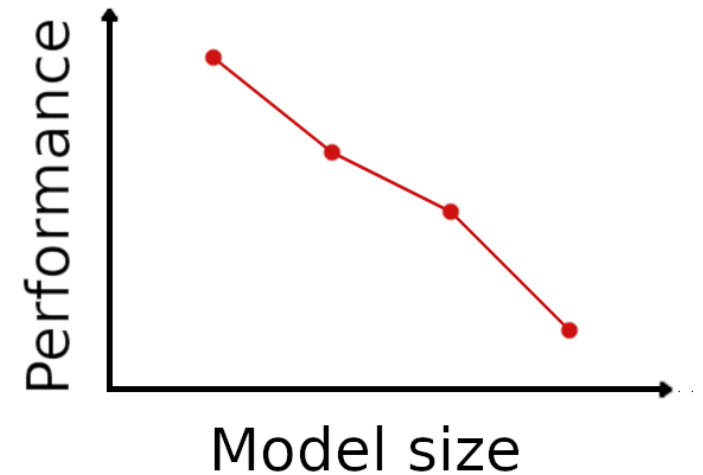
Expected outcome

Inverse scaling

Many tasks like this

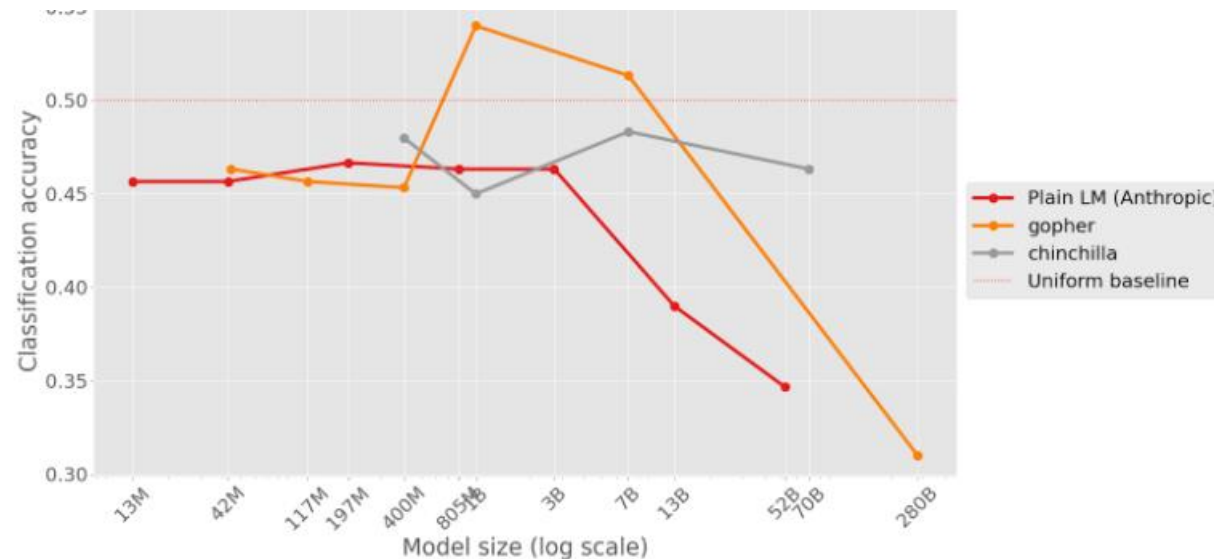


Any tasks like this?



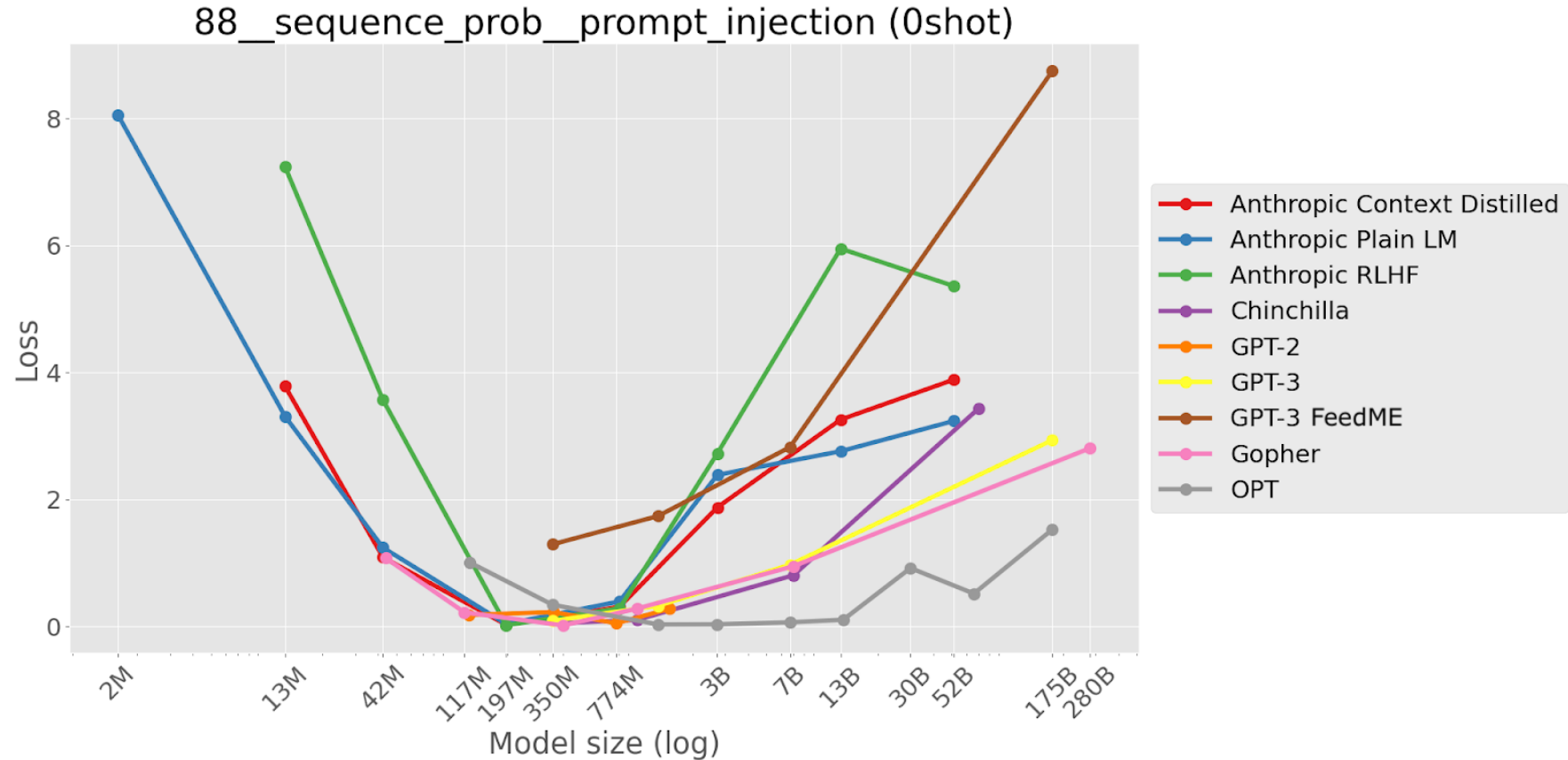
# NeQA: Can Large Language Models Understand Negation in Multi-choice Questions?

- Question: *If a cat has a body temp that is below average, it isn't in*
- A. *danger*
- B. *safe ranges*



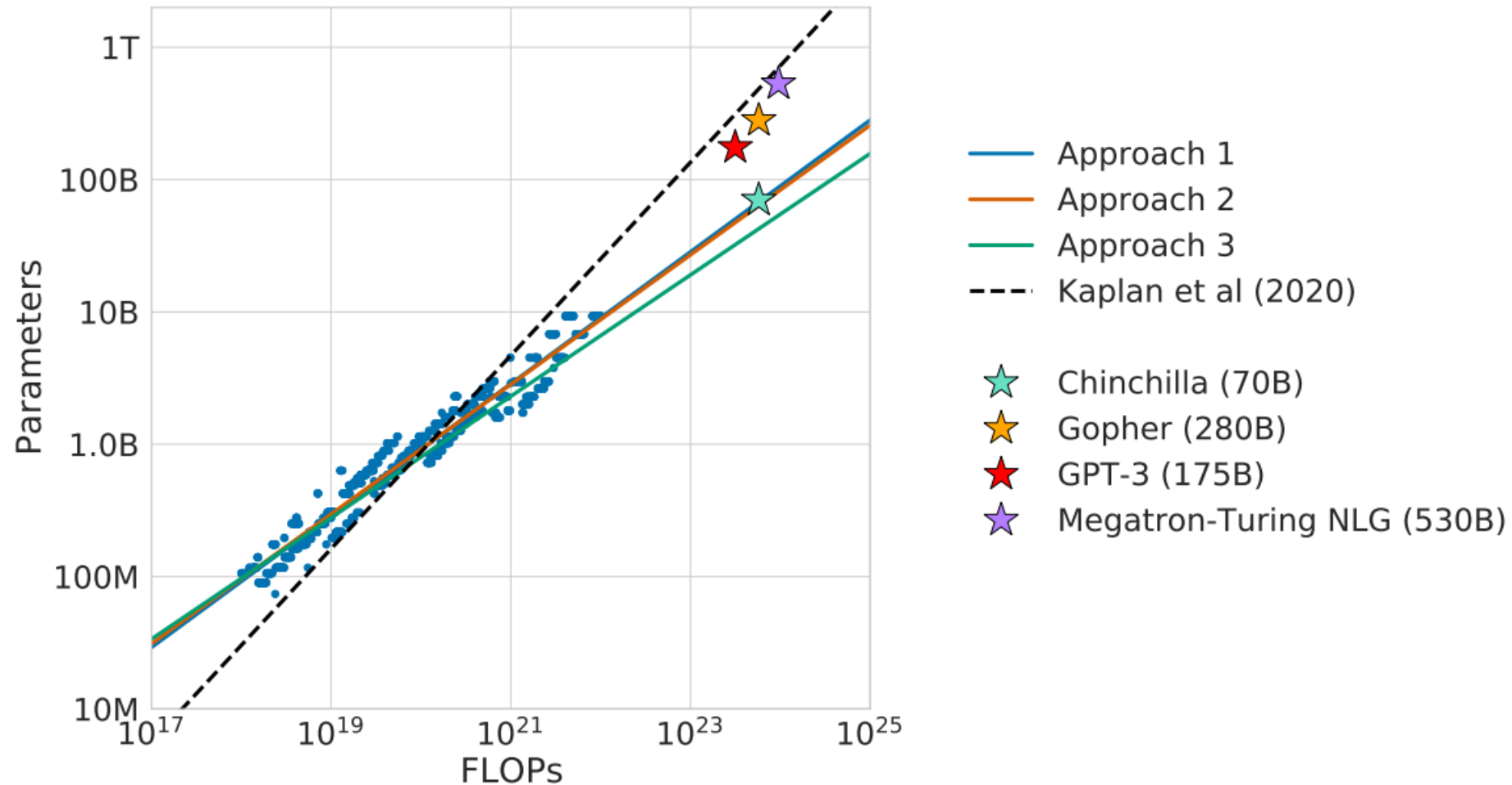
# Jailbreak Paradox 1: Inverse Scaling

More powerful the model, more successful strategies to jailbreak.



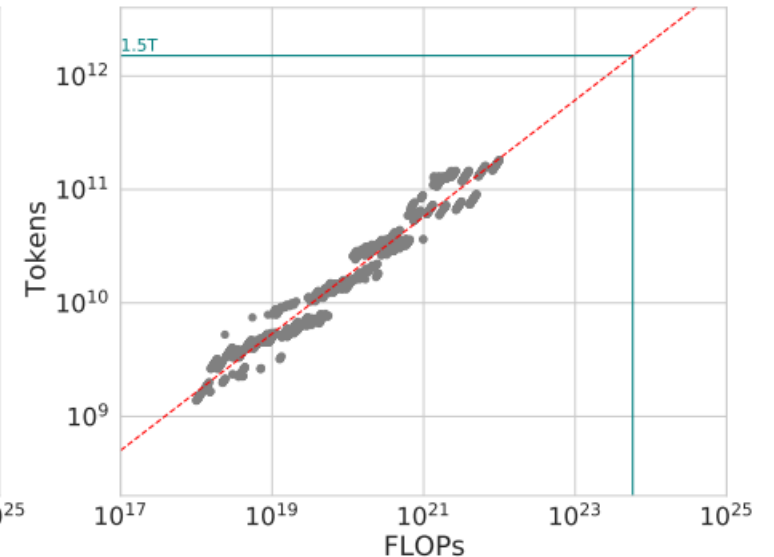
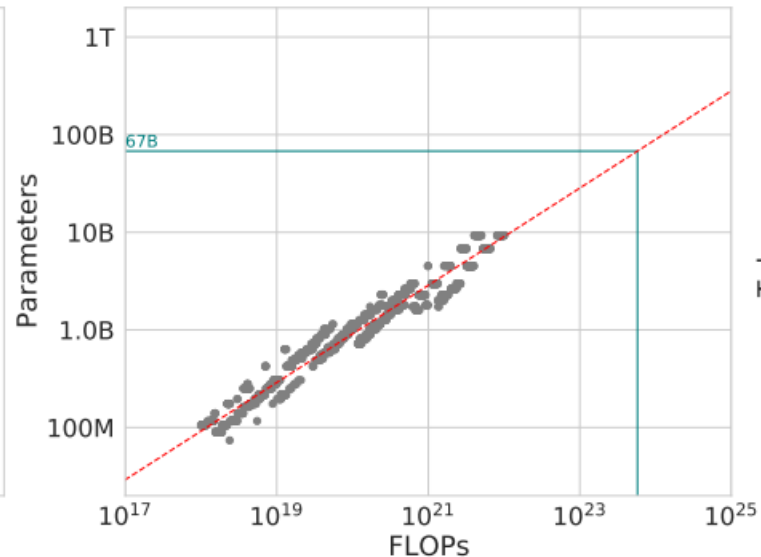
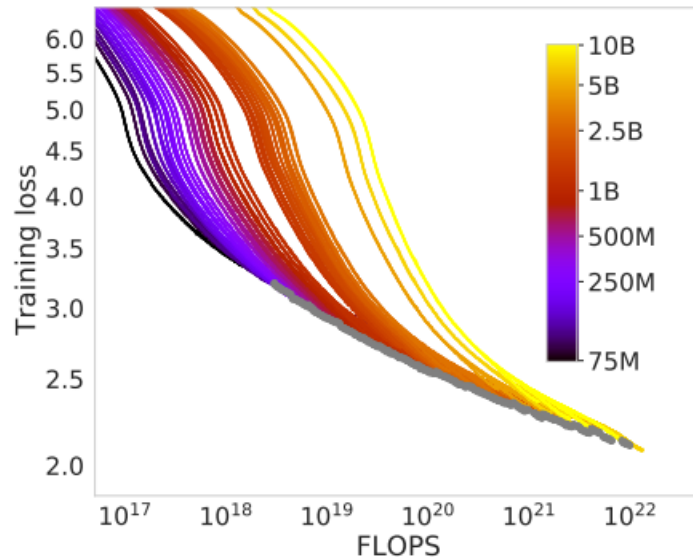
Prompt Injection, by Derik Kauffman, Aaron Kirtland, Andrew Gritsevskiy, and Joe Cavanagh (Third Prize)

Given a **fixed compute budget**, what is the optimal **model size** and **training dataset size** to train transformer-based LM?



# Findings

- Training over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens
- For every doubling of model size, the number of training tokens should also be doubled



# Human vs LLM generation

## **75 years, 75% literacy: India's long fight against illiteracy**

TIMESOFINDIA.COM / Aug 14, 2022, 18:10 IST

A lot happens in human brain to produce such a text

- Information gathering
- Calculation
- Write/rewrite

Lacking

- Don't know what they don't know
- Can't do sanity check
- They don't know what they good at

Advantages

- Internet scale fact-based knowledge
- Large memory

That's why we need right prompting

# Current focus

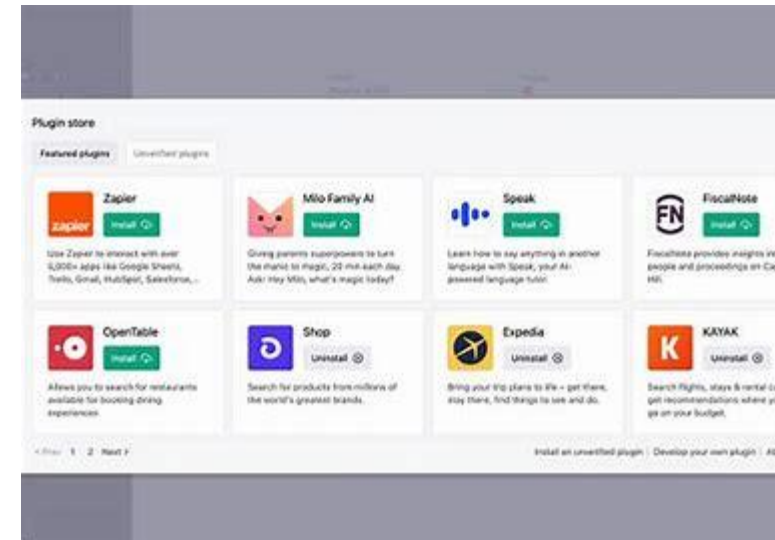
- Plugins
  - Use tools because transformer is not aware where they are not good

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.



# Plugins

## ChatGPT plugins

We've implemented  
ChatGPT. Plugin  
language model  
help ChatGPT do  
computations, o



### Expedia

Bring your trip plans to life—get there, stay there, find things to see and do.



### FiscalNote

Provides and enables access to select market-leading, real-time data sets for legal, political, and regulatory data and information.



### Instacart

Order from your favorite local grocery stores.



### KAYAK

Search for flights, stays and rental cars. Get recommendations for all the places you can go within your budget.



### Klarna Shopping

Search and compare prices from thousands of online shops.



### Milo Family AI

Giving parents superpowers to turn the manic to magic, 20 minutes each day. Ask: Hey Milo, what's magic today?



### OpenTable

Provides restaurant recommendations, with a direct link to book.



### Shop

Search for millions of products from the world's greatest brands.



### Speak

Learn how to say anything in another language with Speak, your AI-powered language tutor.



### Wolfram

Access computation, math, curated knowledge & real-time data through Wolfram|Alpha and Wolfram Language.



### Zapier

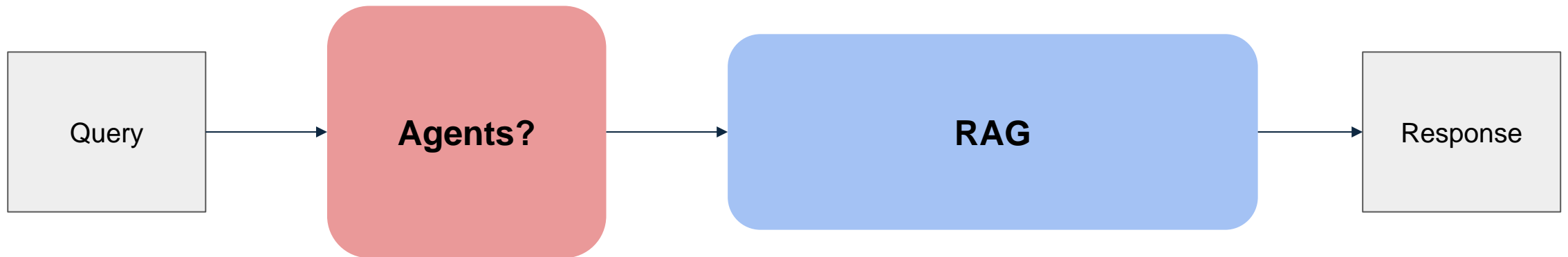
Interact with over 5,000+ apps like Google Sheets, Trello, Gmail, HubSpot, Salesforce, and more.



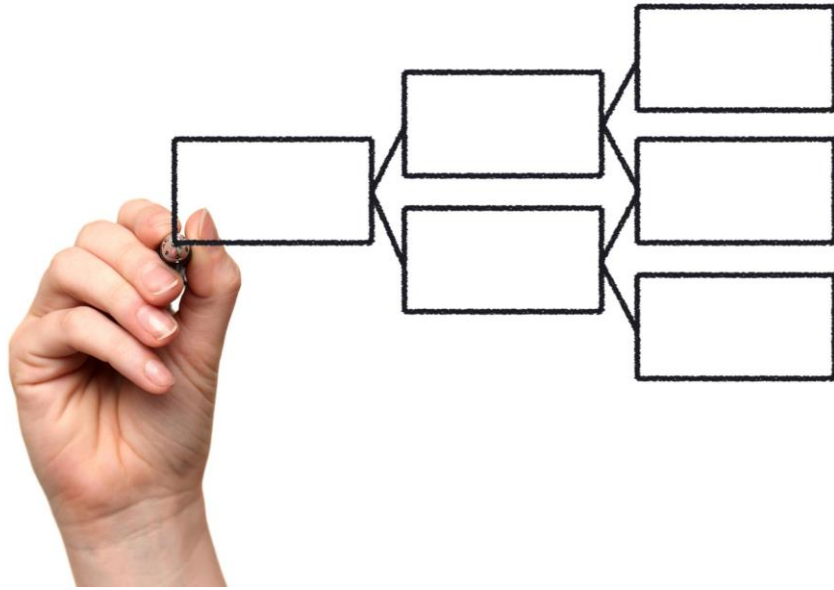
# Retrieval Augmented Generation



# From RAG to Agents



# Summary



- NLP Paradigms: Evolution from feature engineering to prompt engineering.
- Training Regimes: Stages like pretraining, supervised fine-tuning, and RLHF.
- Future Directions: Focus on smaller models, plugins, and addressing limitations.



Thank You