ChatGLM: An Alternative to ChatGPT

Jie Tang

KEG, Tsinghua University

Slides available at: http://keg.cs.tsinghua.edu.cn/jietang/ or Google Jie Tang

What is ChatGLM

- ChatGPT and GPT4 has gained enormous popularity
  - However, techniques behind GPT become a **secret** to all
- ChatGLM, an open-source ChatGPT alternative, toward unclosing the secret
  - GLM-130B: an open-source LLM base model
  - ChatGLM-6B: a lightweight open-source ChatGPT alternative
  - ChatGLM-130B: not open-sourced, but available through API

https://github.com/THUDM/GLM-130B

GLM-130B
An Open Bilingual Pre-Trained Model

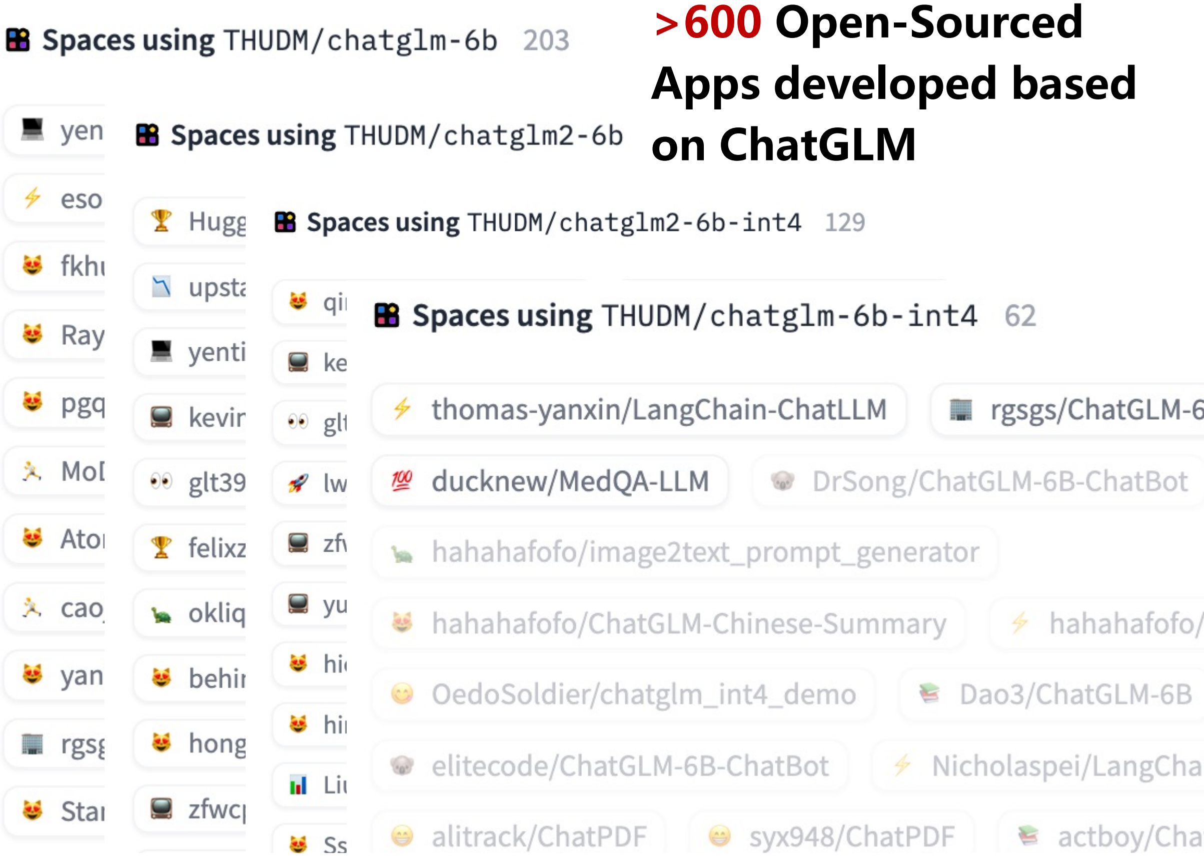
https://github.com/THUDM/ChatGLM3

ChatGLM Alpha

ChatGLM-6B: An Open-Source Alternative

- ChatGLM-6B: **6.2B**parameters, **INT4** quantization (only need 6G memory)
- **>50,000 stars** on github
- **>10,000,000** downloads on Huggingface
- **No. 1** on Github Trending (2 week)
- **No. 1** on Huggingface Trending (2 weeks)

>600 Open-Sourced Apps developed based on ChatGLM



https://github.com/THUDM/GLM-130B

GLM-130B
An Open Bilingual Pre-Trained Model

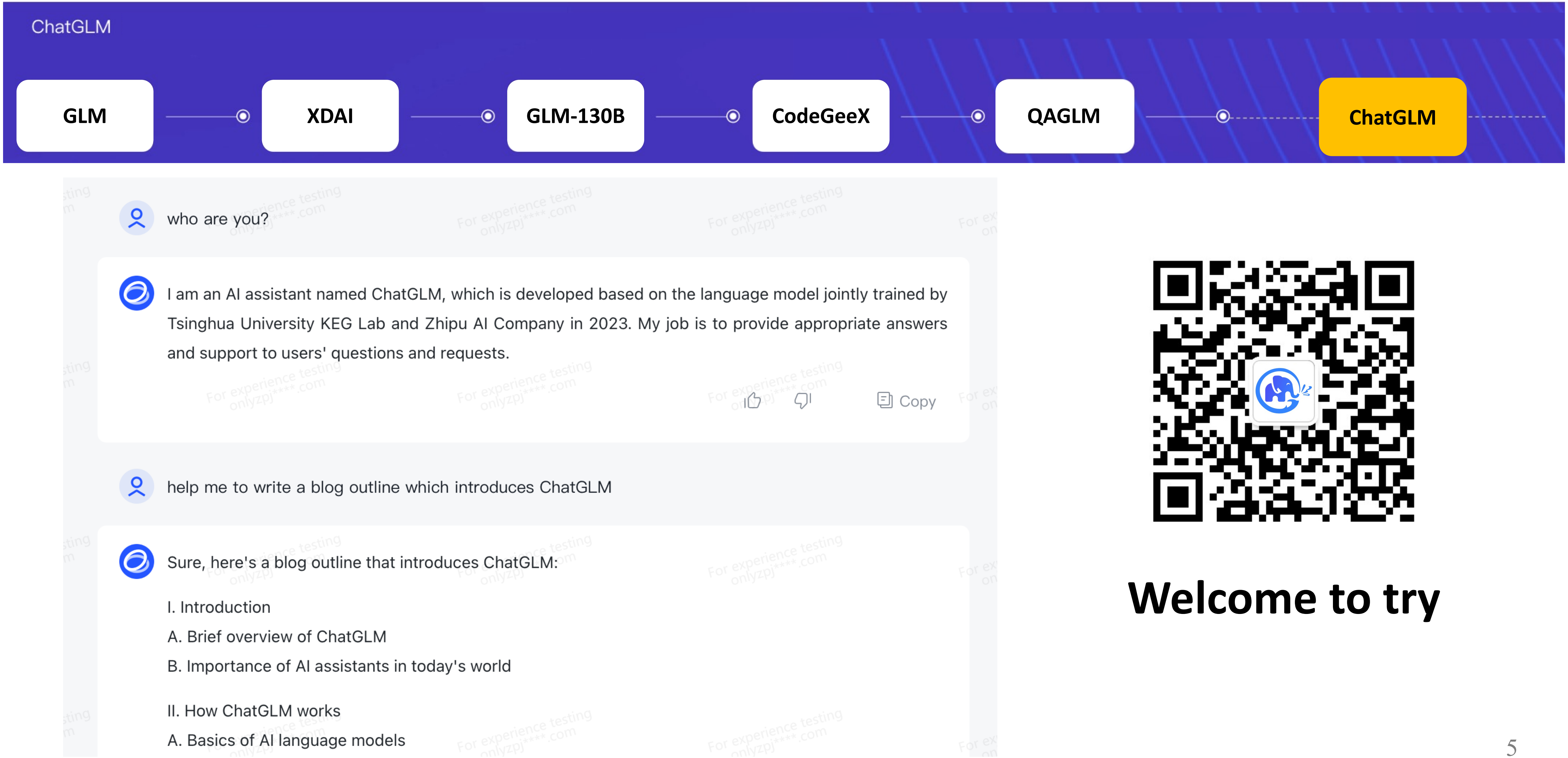
https://github.com/THUDM/ChatGLM3

ChatGLM

ChatGPT vs. ChatGLM

GPT	VS	GLM
ChatGPT		ChatGLM
DALL.E		CogView
Codex		CodeGeeX
WebGPT		WebGLM
GPT-4V		GLM-4V on the way (CogVLM, Agent…)

chatglm.ai



Welcome to try

Story generation

tell a 100 words story about athens in english, chinese, and greek

English: Athens, Greece's stunning capital, is a city of ancient ruins, modern architecture, and vibrant culture. Celebrated for its role in democracy, philosophy, and the Olympic Games, Athens offers visitors a unique blend of the past and present. Explore the Acropolis, Plaka, and Syntagma Square, or relax at one of the many beautiful parks and beaches. Enjoy mouthwatering Greek cuisine, from gyros to spanakopita, and immerse yourself in the city's lively arts and music scene. Athens is a destination where history comes alive and memories are made to last a lifetime.

中文：雅典，希腊美丽的首都，是一座融合了古老遗迹、现代建筑和生动文化的城市。雅典因其在民主、哲学和奥林匹克运动中的地位而闻名，游客可以在这里感受到独特的过去与现在的交融。游览 Acropolis、Plaka 和 Syntagma 广场，或者在众多美丽的公园和海滩放松身心。品尝令人垂涎的希腊美食，从烤肉串到菠菜派，让自己沉浸在城市繁忙的艺术和音乐氛围中。雅典是一个让历史焕发活力，留下终身难忘记忆的旅游胜地。

希腊语：Αθήνα, η καλύτερη και πόλη της Γαλλίας, είναι ένα σύγκρουσμα της παλαιότερης και της σύγχρονης και της καλλιτέχνης μας. Στην Αθήνα θα βρείτε όλα τα σπηλιά μας, τη σύγχρονη αρχιτεκτούρα και την καλύτερη κοινωνία. Ξεκινήστε τη διαβάστε την Ακρόπολη, τη Plaka και το Syntagma Square, ή καλύπτεστε στο ένα άλλο της πολλές όμορφες παρκy και παραλίες. Γεύσαστε την καλή Γιουγκοσλαβική κουζίνα, από τα γύρο στη σπανακόπιτα, και παραστρέψετε σας εγώγεια μέσα στη σύγχρονη καλλιτέχνη και τη μουσική σκηνή της Αθήνας. Η Αθήνα είναι ένα σημείο όπου η ιστορία ζει στη ζωή και οι ευπορικές σας θOUGHTS στη ζωή του κόσμου

Applied Math

A company produces a toy for which the variable cost is 12.30 per unit and the fixed costs are 98,000.The toy sells for $17.98.Let X be the number of units produced and sold.a) The total cos t for a business is the sum of the variable cost and the fixed costs.Write the total cost C as a fu nction of the number of units produced.b)Write the revenue R as a function of the number of units sold.c)Write the profit P as a function of the number of units sold

展开

a) The total cost for a business is the sum of the variable cost and the fixed costs. Write the total cost C as a function of the number of units produced:

C(X) = 12.30X + 98000

b) The revenue R is the sales price minus the variable cost. Write the revenue R as a function of the number of units sold:

R(X) = 17.98 - 12.30X

c) The profit P is the revenue R minus the total cost C. Write the profit P as a function of the number of units sold:

P(X) = R(X) - C(X) = 17.98 - 12.30X - 98000

Note: The negative sign indicates that the profit is a function of the number of units sold in the negative direction, meaning that as the number of units sold increases, the profit decreases.

Coding

Write a code for a bubble sort algorithm in Python, and give an array as an example

Sure, here's a simple bubble sort algorithm implemented in Python:

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(n-i-1):
            if arr[j] > arr[j+1]:
                # Swap adjacent elements if they are in wrong order
                arr[j], arr[j+1] = arr[j+1], arr[j]
    return arr
```

To use this function, you'll need to define an array to sort, like so:

```
arr = [64, 34, 25, 12, 22, 11, 90]
```

Once you have your array defined, you can call the `bubble_sort()` function on it, like so:

```
sorted_arr = bubble_sort(arr)
```

Finally, you can print the sorted array like so:

```
print(sorted_arr)
```

This should output something like:

```
[11, 12, 22, 25, 34, 64, 90]
```

Let's take a look at an example array to better understand how the bubble sort algorithm works:

```
arr = [5, 2, 9, 1, 5, 6, 10]
```

In this example, the first element (5) is sorted towards the bottom of the array, and the last element (10) is sorted towards the top. The middle elements (2, 5, and 9) are sorted in between their adjacent elements, as shown in the following sorted array:

```
[1, 2, 5, 5, 6, 9, 10]
```

Overall, the bubble sort algorithm works by iterating through the array, comparing adjacent elements, and swapping them if they are in the wrong order. This process repeats until all elements in the array are sorted in increasing order.

GLM-4V (pre-release)

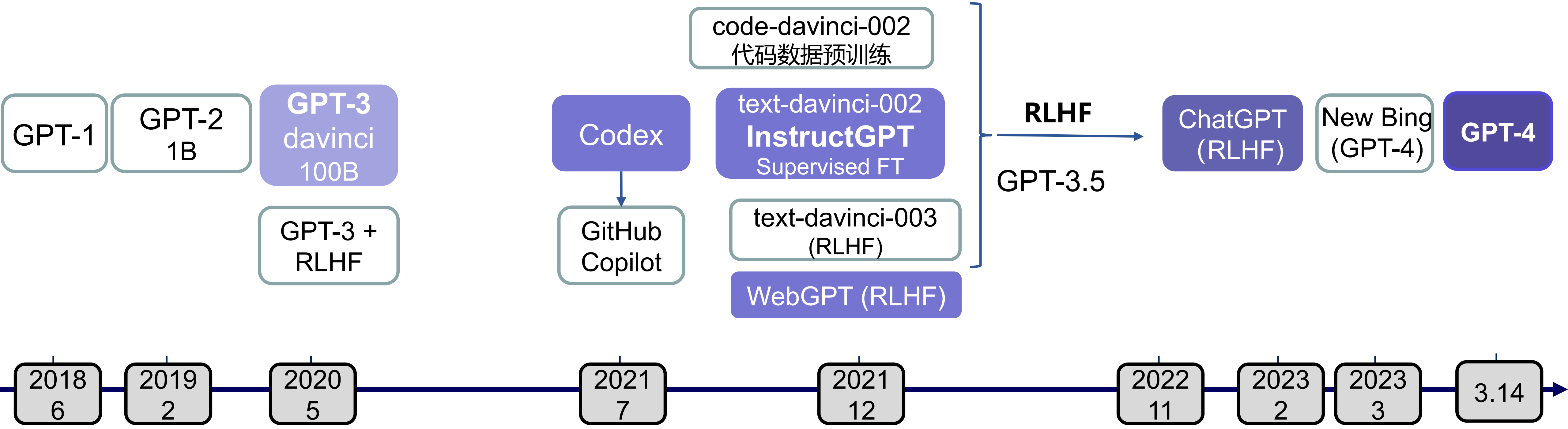
who are in this picture

In this picture, there are three football players. From left to right, they are Suárez, Neymar, and Messi.

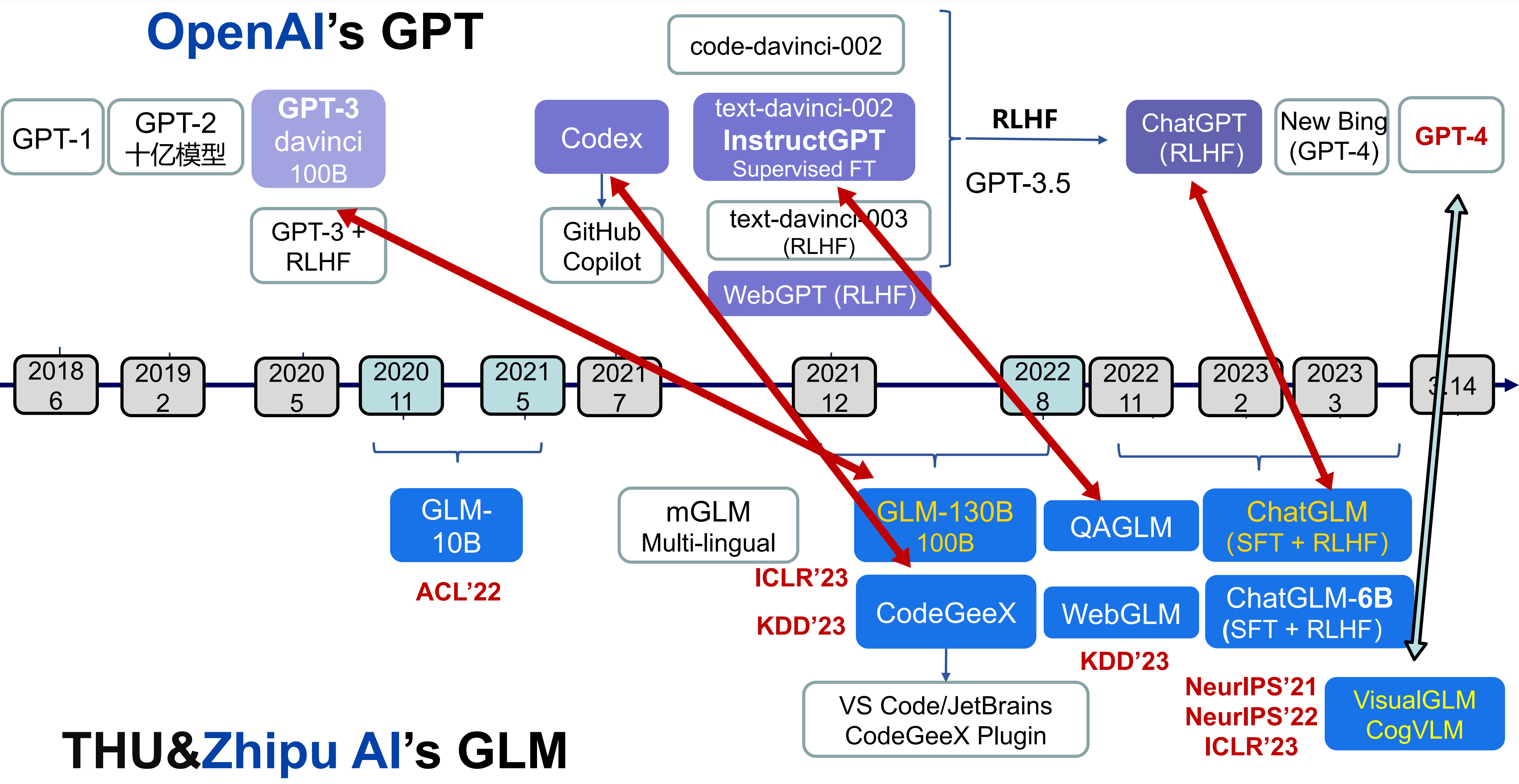
“draw a dog with a hat”

```
KG engine ready...
```

GPT



1. 100B Base model **2. Supervised FT** **3. RLHF**

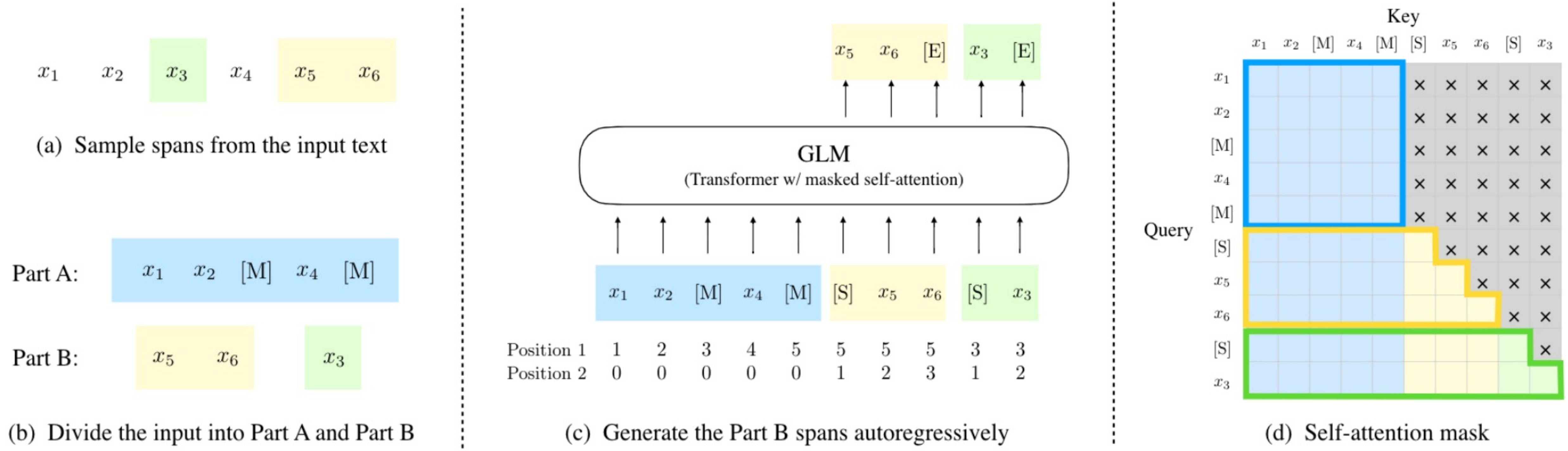

OpenAI's GPT
code-davinci-002
GPT-1
GPT-2
十亿模型
GPT-3
davinci
100B
Codex
text-davinci-002
InstructGPT
Supervised FT
RLHF
GPT-3.5
ChatGPT
(RLHF)
New Bing
(GPT-4)
GPT-4
GPT-3 +
RLHF
GitHub
Copilot
text-davinci-003
(RLHF)
WebGPT (RLHF)
2018
6
2019
2
2020
5
2020
11
2021
5
2021
7
2021
12
2022
8
2022
11
2023
2
2023
3
3.14
GLM-
10B
ACL'22
mGLM
Multi-lingual
ICLR'23
GLM-130B
100B
QAGLM
ChatGLM
(SFT + RLHF)
KDD'23
CodeGeeX
WebGLM
ChatGLM-6B
(SFT + RLHF)
KDD'23
NeurIPS'21
NeurIPS'22
ICLR'23
VisualGLM
CogVLM
VS Code/JetBrains
CodeGeeX Plugin
THU&Zhipu AI's GLM

General Language Model (GLM)

Framework	NLU	Cond. Gen.	Uncond. Gen.
Autoregressive (GPT)	—	—	√
Autoencoding (BERT)	√	×	×
Encoder-Decoder (T5)	—	√	—
Autoregressive Blank-Infilling (GLM)	√	√	√

Du and Qian et al. All NLP Tasks are Generation Tasks. ACL'22. arxiv: 2103.10360

General Language Model (GLM)

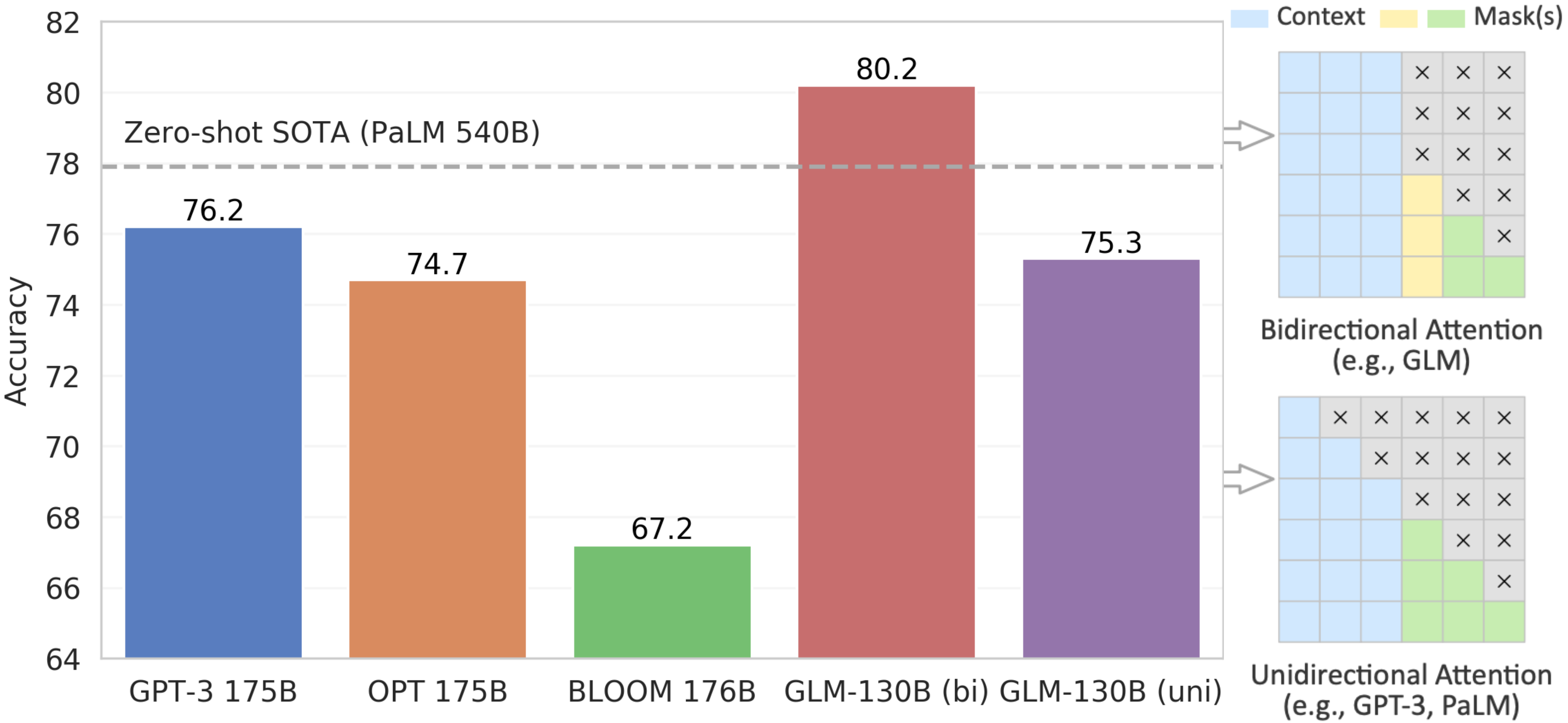


$$\mathcal{L}_{\text{GLM}} = \mathbb{E}_{\mathbf{z}\sim Z_m}\left[\sum_{i=1}^{m}\sum_{j=1}^{l_i} -\log p\left(s_{z_i,j} \middle| \mathbf{x}_{\text{corrupt}}, \mathbf{s}_{z_{<i}}, s_{z_i,<j}\right)\right]$$

General Language Model (GLM)

Du and Qian et al. All NLP Tasks are Generation Tasks. ACL'22. arxiv: 2103.10360

General Language Model (GLM)



LAMBADA

Zeng, Liu, et al. GLM-130B: An Open Bilingual Pre-trained Model. ICLR'23

Results on Natural Language Understanding

- **Better than BERT, T5, RoBERTa**

Table 2. Results on the SuperGLUE dev set. Models with * are pre-trained for two times the number of steps of other methods.

Model	ReCoRD F1/Acc.	COPA Acc.	WSC Acc.	RTE Acc.	BoolQ Acc.	WiC Acc.	CB F1/Acc.	MultiRC F1a/EM	Avg
$BERT_{Base}$	65.4/64.9	66.0	65.4	70.0	74.9	**68.8**	70.9/76.8	68.4/21.5	66.1
GLM_{Base}	**73.5/72.8**	**71.0**	**72.1**	**71.2**	**77.0**	64.7	**89.5/85.7**	**72.1/26.1**	**70.7**
$BERT_{Large}$	76.3/75.6	69.0	64.4	73.6	80.1	**71.0**	94.8/92.9	71.9/24.1	72.0
$UniLM_{Large}$	80.0/79.1	72.0	65.4	76.5	80.5	69.7	91.0/91.1	77.2/38.2	74.1
GLM_{Large}	81.7/81.1	76.0	**81.7**	74.0	**82.1**	68.5	96.1/94.6	**77.1/36.3**	77.0
GLM_{Large} (multi-task)	80.2/79.6	77.0	78.8	76.2	79.8	63.6	**97.3/96.4**	74.6/32.1	75.7
GLM_{410M} (multi-task)	81.5/80.9	80.0	**81.7**	**79.4**	81.9	69.0	93.2/96.4	76.2/35.5	78.0
GLM_{515M} (multi-task)	**82.3/81.7**	**85.0**	**81.7**	79.1	81.3	69.4	95.0/96.4	77.2/35.0	**78.8**
$T5_{Base}$	76.2/75.4	73.0	79.8	78.3	80.8	67.9	94.8/92.9	76.4/40.0	76.0
$T5_{Large}$	85.7/85.0	78.0	**84.6**	84.8	84.3	71.6	96.4/98.2	80.9/46.6	81.2
$BART_{Large}$ *	88.3/87.8	60.0	65.4	84.5	84.3	69.0	90.5/92.9	81.8/48.0	76.0
$RoBERTa_{Large}$ *	89.0/88.4	**90.0**	63.5	87.0	**86.1**	**72.6**	96.1/94.6	**84.4/52.9**	81.5
$GLM_{RoBERTa}$	**89.6/89.0**	82.0	83.7	**87.7**	84.7	71.2	**98.7/98.2**	82.4/50.1	**82.9**

Resutls on Generation

- **The most important thing is that one model can do all the things**

Table 3. Results on Gigaword abstractive summarization

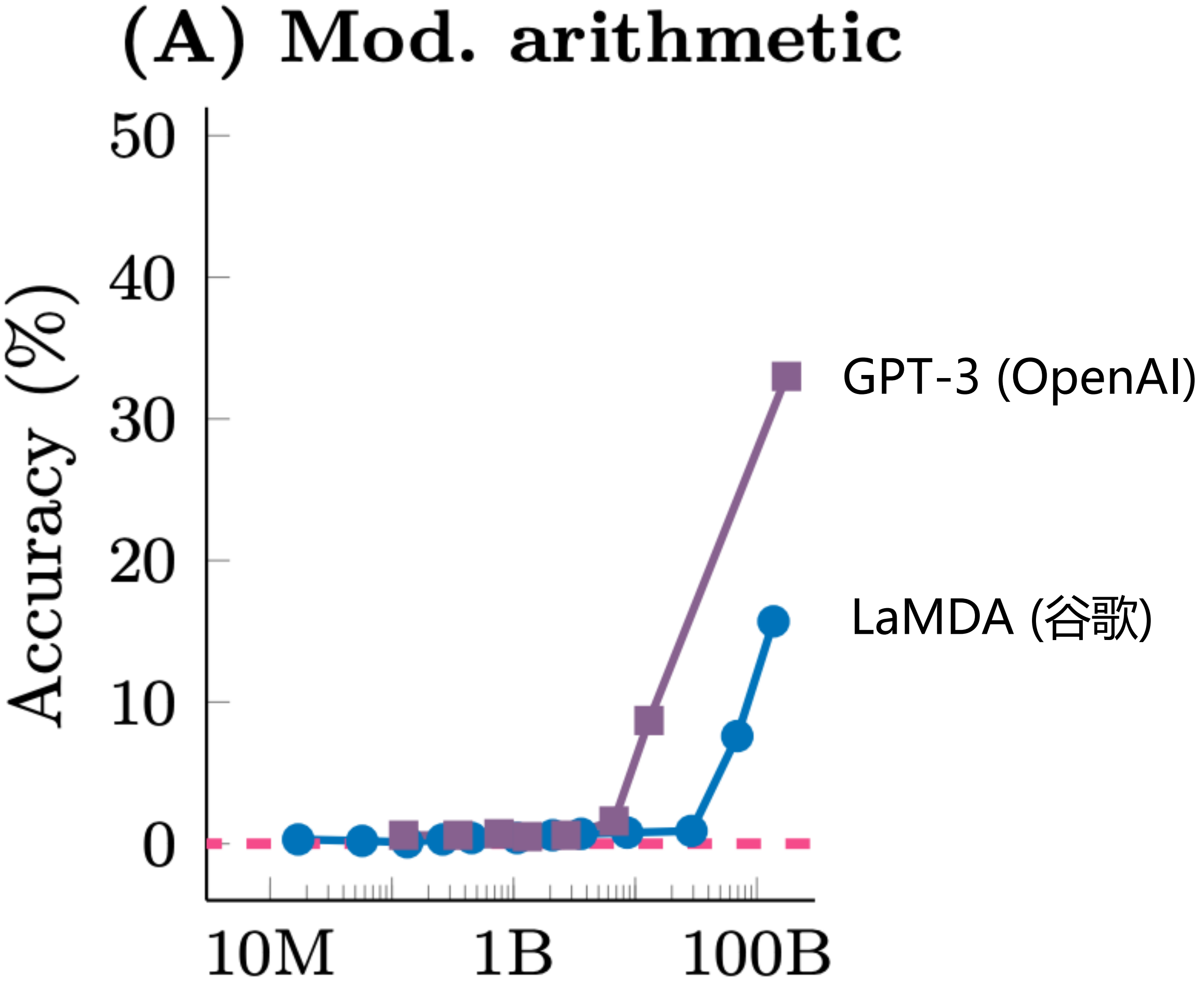
Model	RG-1	RG-2	RG-L
MASS	37.7	18.5	34.9
$\text{UniLM}_{\text{Large}}$	38.5	19.5	35.8
GLM $_{\text{Large}}$	38.6	19.7	36.0
GLM $_{\text{Large}}$ (multi-task)	38.5	19.4	35.8
GLM $_{\text{410M}}$ (multi-task)	**38.9**	**20.0**	**36.2**

Table 4. Zero-shot language modeling results.

Model	Lambada (Accuracy)	BookWiki (Perplexity)
GLM $_{\text{Large}}$ (uni)	0.0	> 100
GLM $_{\text{Large}}$ (multi-task,uni)	47.4	15.1
– 2d positional encoding	45.8	15.1
GLM $_{\text{410M}}$ (multi-task,uni)	49.5	14.5
GLM $_{\text{515M}}$ (multi-task,uni)	**50.4**	**13.9**
GLM $_{\text{Large}}$ (bi)	10.6	> 100
GLM $_{\text{Large}}$ (multi-task,bi)	48.5	14.9
– 2d positional encoding	47.3	15.0
GLM $_{\text{410M}}$ (multi-task,bi)	**53.5**	**14.3**
GLM $_{\text{515M}}$ (multi-task,bi)	**54.9**	**13.7**
$\text{GPT}_{\text{Large}}$ (uni)	50.1	14.4

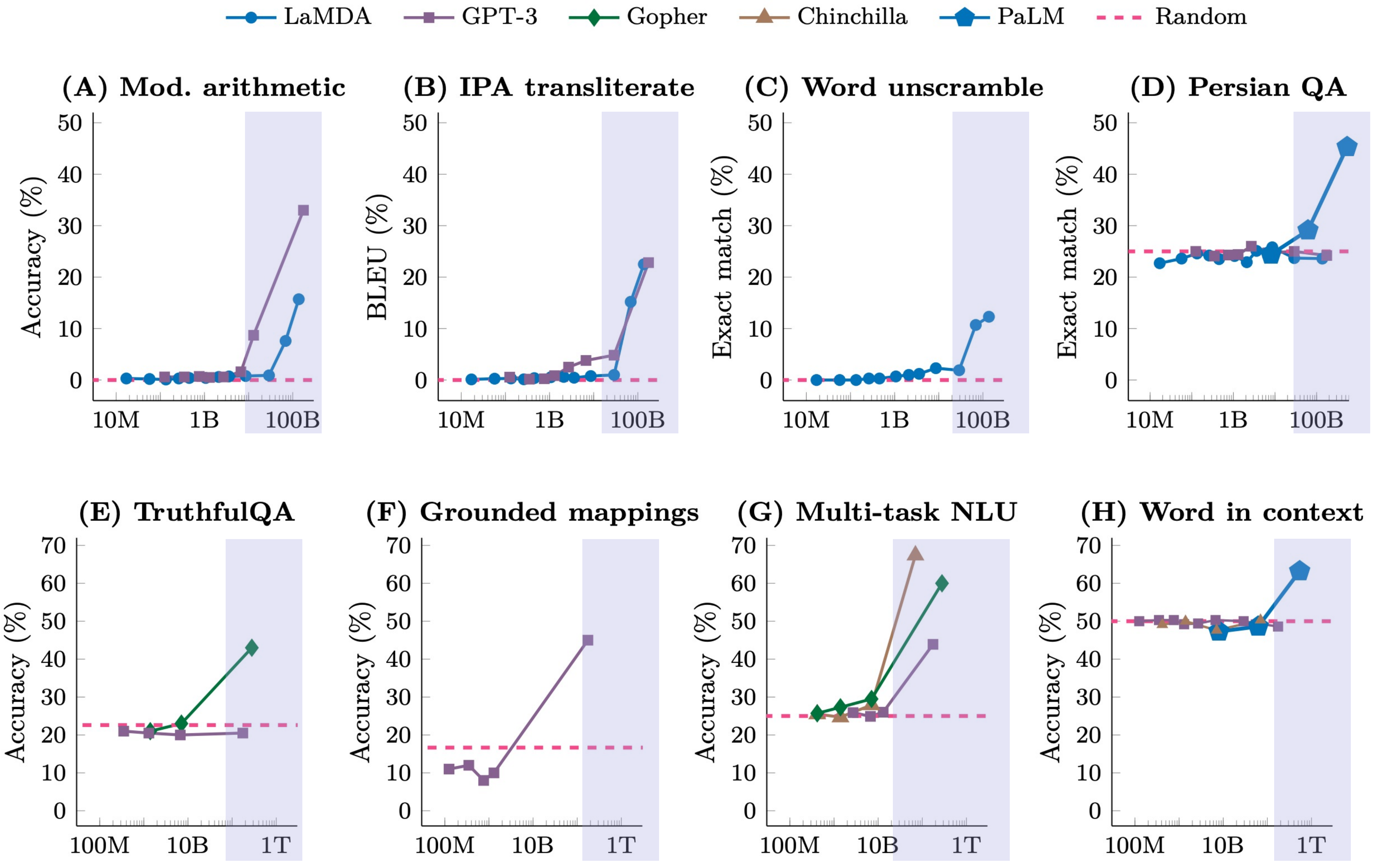
Why 100B-scale model?

- What is 16 mod 12?
- 16 divided by 12 equals 1 remainder 4. So the answer is 4!



1. J Wei, et al. Emergent Abilities of Large Language Models. arXiv: **2206**.07682

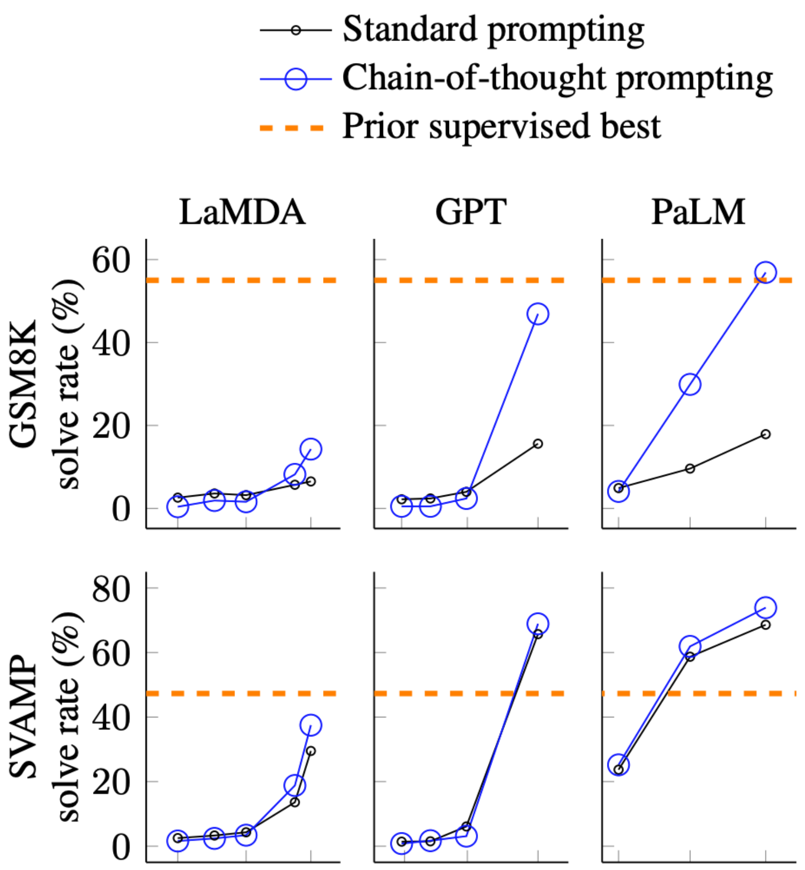
Why 100B-scale model?



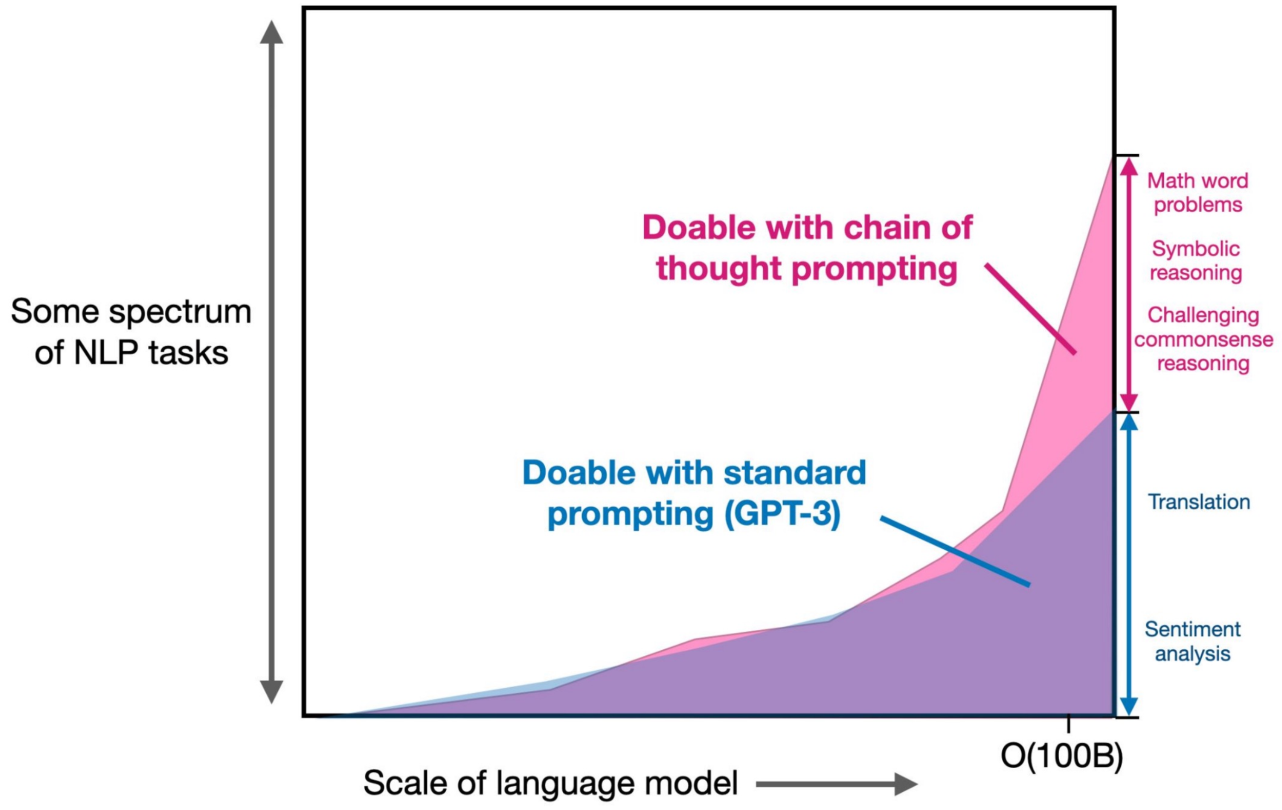
1. J Wei, et al. Emergent Abilities of Large Language Models. arXiv: **2206**.07682

Scaling Law

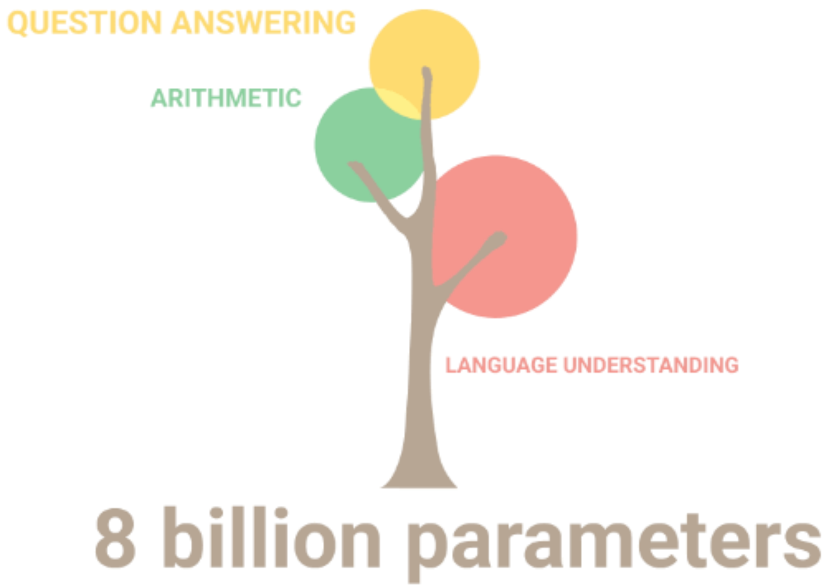
Scaling Law introduces complicated reasoning abilities



Model scale (# parameters in billions)



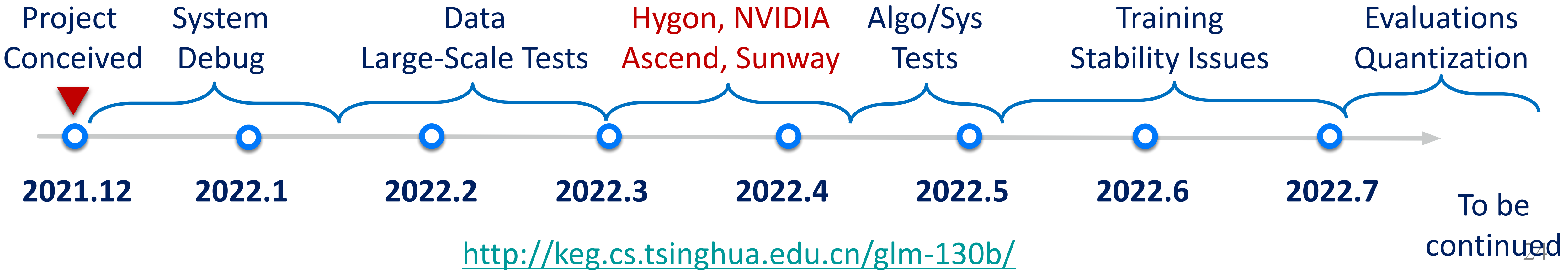
“Emergent abilities”



Gif Credit: Google

How to train a 100B–scale LLM?

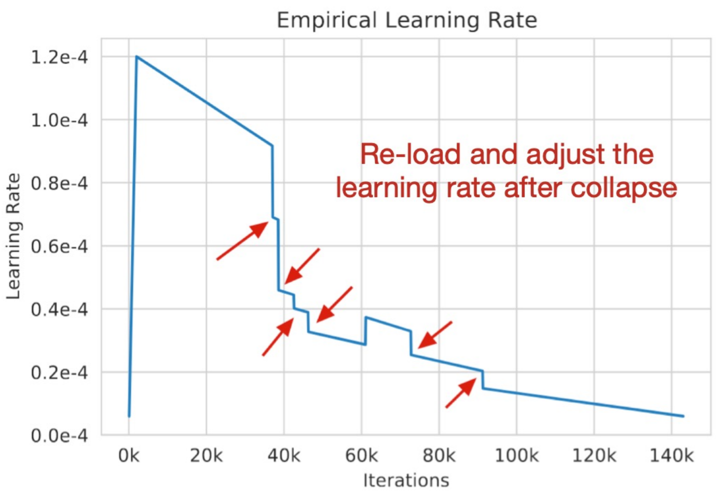
- 8 months have witnessed numerous challenges
  - **Engineering:** How to train 100B-scale models from scratch?
    - Hygon DCU, NVIDIA A100, Ascend 910, Sunway
    - Frequent & random hardware failures, Megatron-DeepSpeed 3D pipeline, CUDA kernel efficiency, GPU memory overflow, 10K+ threads TCP init & comms…
  - **Algorithm:** How to stabilize the training of 100B-scale models?
    - The gradient norms of embeddings, Post-LN / Pre-LN stability, dataloader state seeds, computation precision in Softmax / Attention



http://keg.cs.tsinghua.edu.cn/glm-130b/

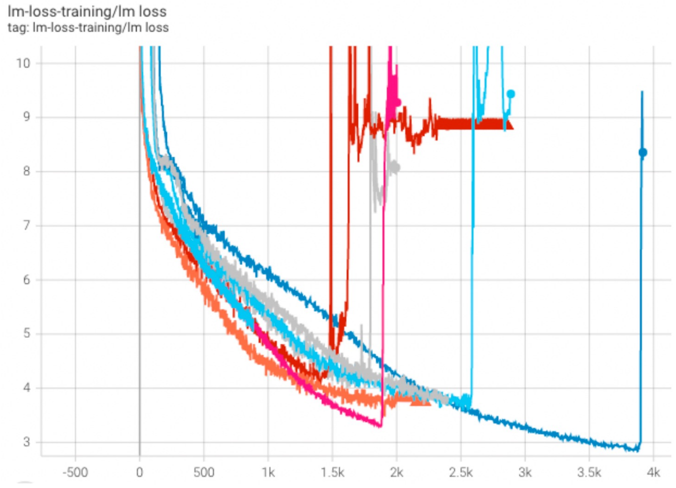
Training Stability of 100B-Scale Models

- Tradeoff: Stability (Slow) or Efficiency (Instable)
- Existing Solutions
  - **OPT-175B:** manually adjust LR & skip data when collapses (performance drop)
  - **BLOOM 176B**: embedding norm & BF16 (performance drop, few platform)



(a) OPT 175B's experiments

(b) BLOOM 176B's experiments



(c) GLM 130B's experiments

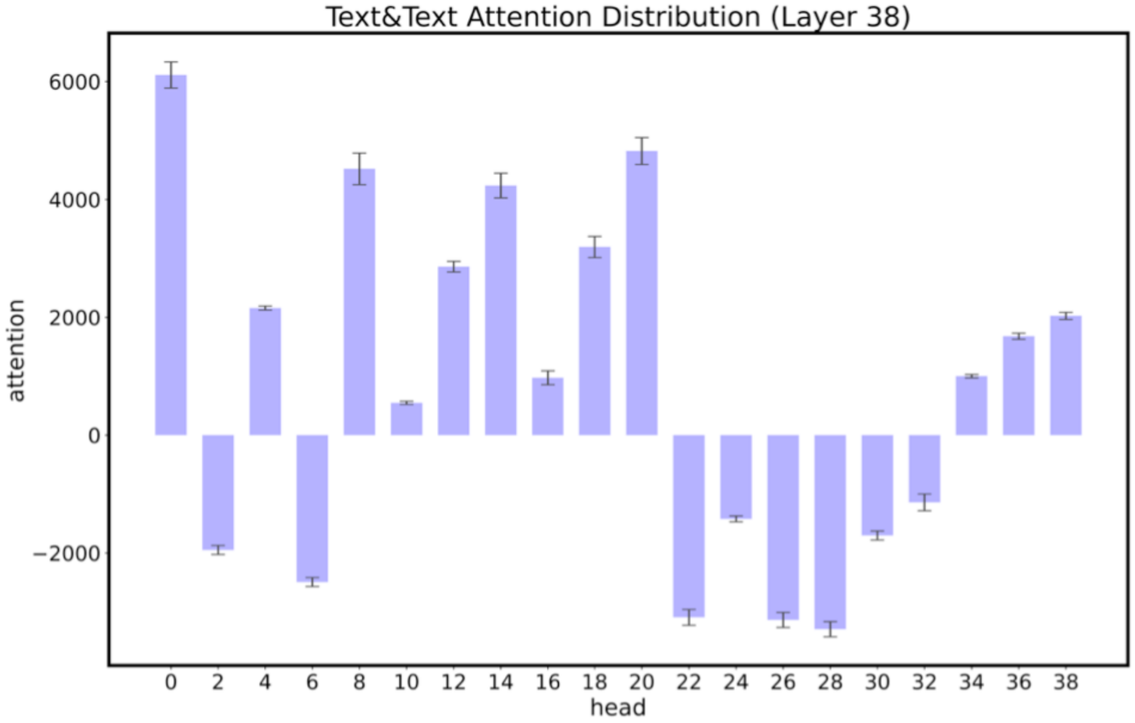
Sources: OPT / BLOOM / GLM-130B

GLM-130B: Training Stability

- Attention score: Softmax in 32 to avoid overflow

$$\operatorname{softmax}\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^\top}{\sqrt{d}}\right)=\operatorname{softmax}\left(\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^\top}{\alpha\sqrt{d}}-\max\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^\top}{\alpha\sqrt{d}}\right)\right)\times\alpha\right)=\mathrm{FP16}\left(\operatorname{softmax}\left(\mathrm{FP32}\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^\top}{\alpha\sqrt{d}}\right)\times\alpha\right)\right)$$

Attention scores grow large --- exceeding FP16's range



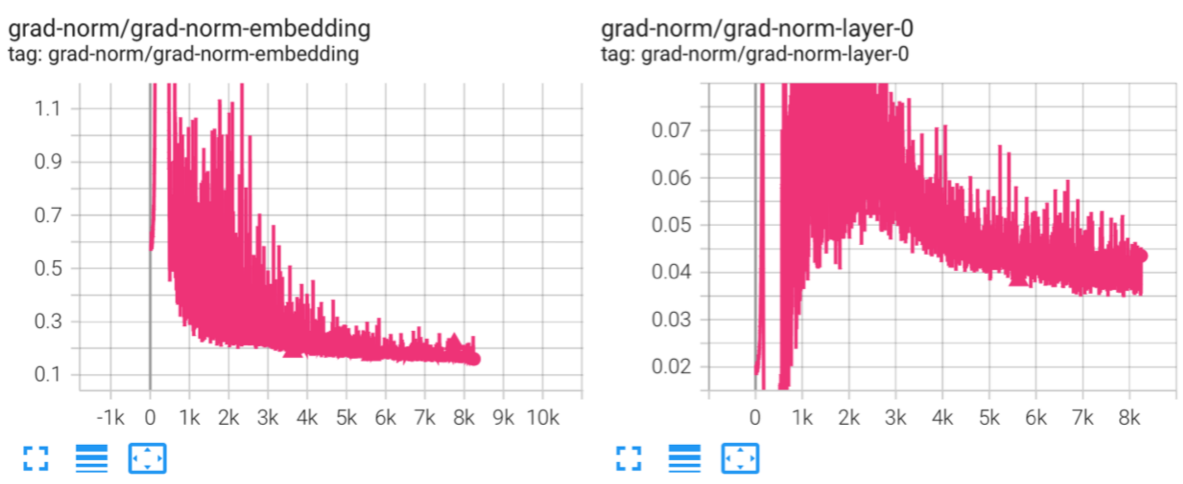
Zeng, Liu, et al. [GLM-130B: An Open Bilingual Pre-trained Model](). ICLR'23

GLM-130B: Training Stability

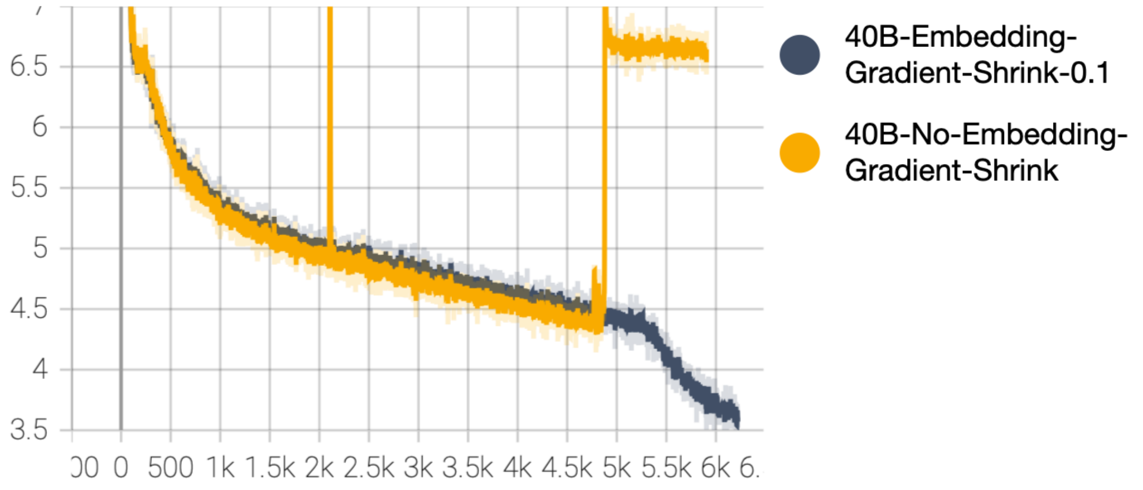
Embedding Layer Gradient Shrink (EGS)

```
word_embedding = word_embedding * alpha +
                 word_embedding.detach() * (1 – alpha)
```

Embedding Layer gradients can be magnitudes larger than others



(a) Gradient norm of embedding layer (left) and the first layer (right)

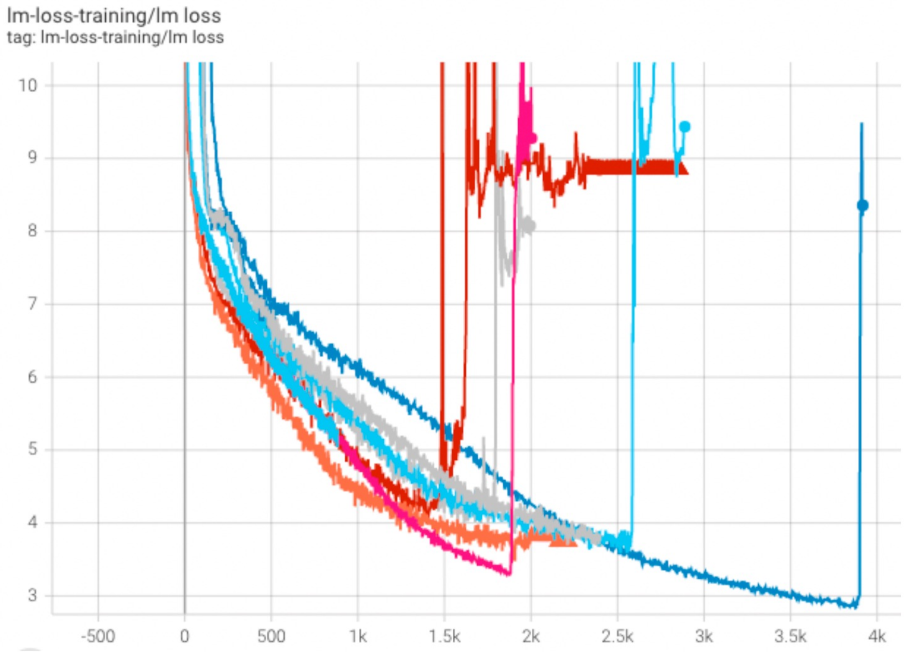


(b) Training loss curves of GLM-40B with and without gradient shrink

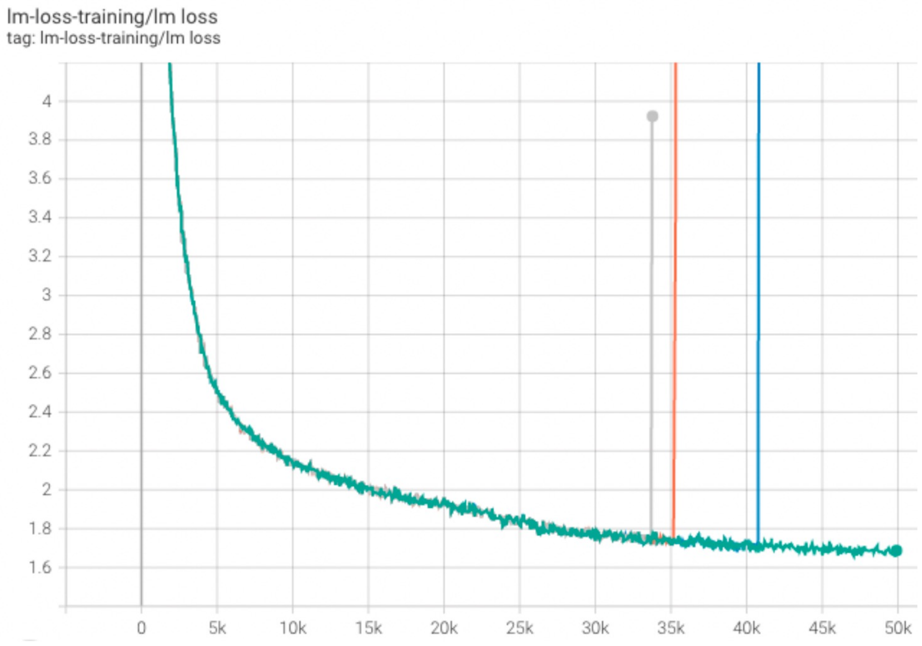
Zeng, Liu, et al. GLM-130B: An Open Bilingual Pre-trained Model. ICLR’23

GLM-130B: Training Stability

- The final training run of GLM-130B



(c) GLM 130B's experiments



(d) GLM 130B's real training

Zeng, Liu, et al. GLM-130B: An Open Bilingual Pre-trained Model. ICLR'23

GLM-130B Training Lessons

2021.12
- The “千亿” (100B) project towards an open dense pre-trained GLM at 100B scale is conceived
- Survey pre-training strategies of existing models of similar scale, such as GPT-3, Gopher *=> Limited public info about how they were trained and issues they met*
- Search for possible GPU clusters & sponsors

2022.1
- Test the performance of FP16/FP32 at 100B scale on one testing cluster
- Unexpected excessive memory usage in GLM *=> Torch is better with fixed length input sequences*
- Inability to converge and try tricks from CogView and ViT *=> Use Sandwich-LN*
- Frequent random hardware failures *=> Have to run HCPG test before each run*

2022.2
- Very slow training speed than previously calculated *=> Optimize kernels and fuse operators => Find the input shape is critical to kernel performance*
- Collect pre-training corpora and tokenize *=> Use icetk: the sentence piece is set to the unigram mode*
- Debug the 3D pipeline parallel in the newly-released Megatron and DeepSpeed

2022.3
- It can’t recover perfectly from optimizer states *=> Our customized dataloaders do not save its state seed properly in distributed training*
- The memory per processor is too small *=> Require too many pipeline stages* => Batch size is too large (up to 12,000) *=> Harm the model’s convergency*
- It can’t launch more than 2,000 computing nodes *=> Overcome this and support 6,000-node training by tuning Linux kernel TCP parameters*
- Collect data for multi-task instruction pre-training
- Receive opportunities to test trainings on several other clusters
- Very slow training speed than expected *=> The underlying element-wise operators don’t support fast computation on large-dimension vectors.*

2022.4
- Optimize A100 kernel’s computing efficiency *=> A100 kernels prefer square-shaped inputs, and seq_len=2,048 is optimal for our hidden-state dimension (12,288)*
- Inability to converge due to large gradient norms (170+) of input embeddings *=> Try embedding norm and gradient shrink, which turn out to be almost equivalent*
- Naïve post-LN or pre-LN disconverges after several thousands of steps *=> Try Sandwich-LN with PB-Relax*
- It still disconverges after one week’s trial *=> The dataloader state seeds are not unified for different pipeline stages, resulting in a mismatch of input data and labels.*
- Test two positional encodings: RoPE and Alibi *=> Alibi can be slower as it requires element-wise manipulation on attention matrices---changing num_heads *2,048 * 2,048 scalars per layer*
- Test GeGLU and GAU *=> GAU converges faster with relatively poor performance on fine-tuned SuperGLUE*
- Abnormal GPU memory usage of newly-added functions and classes *=> DeepSpeed hardcodes the function names for checkpoint activation*
- Decode to train GLM with 130 billion parameters *=> allow inference on a DGX-A100 40G node*

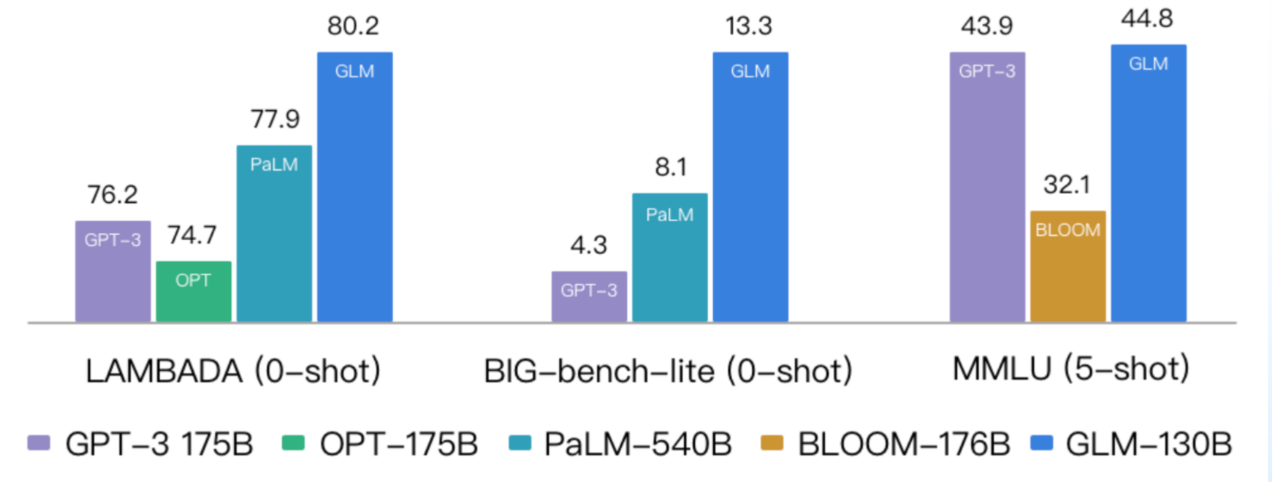
2022.5-6
- Implement a RoPE cuda operator in C++ *=> See unexpected precision errors and finally have it abandoned*
- Sandwich-LN still disconverges *=> 1) Reducing learning rate does not help; 2) Using Hinge cross-entropy becomes slower and harms performance; 3) Shifting to DeepNorm still disconverges*
- Use FP32 in softmax of attention *=> Success*
- Find PB-Relax unnecessary for FP32 softmax *=> It also slows down training as it needs to manipulate the whole attention score matrices*
- Experience few spikes in later training *=> 1) Reduce gradient shrink factor from 1 to 0.1: useful; 2) Reduce the learning rate: sometimes useful; 3) Jump the noisy data batches: sometimes useful*
- Find a mistake in multi-task data after training for 20,000 steps *=> Use the correct data but it does not forget*

2022.6-7
- Adapt the pipeline parallel checkpoints to ordinary parallel checkpoints for efficient inference on a single A100
- Work on evaluation scripts on datasets: MMLU, Big-bench, CLUE, SuperCLUE, etc.
- Implement P-Tuning and P-Tuning v2 for parameter-efficient tuning on GLM-130B for tuning on SuperGLUE
- Work with BMInf on adapting GLM-130B to perform inference on a single V100 or 3090 *=> Use pipeline-style asynchronous swapping between main memory and GPU memory*
- Try to fine-tune GLM-130B with fewer A100 nodes (i.e., 12-16 nodes) *=> Pipeline-style fails due to too many pipeline stages => Find that data parallel can not be introduced for fine-tuning => Use 32-way model parallel fo fine-tuning with reasonable performance*

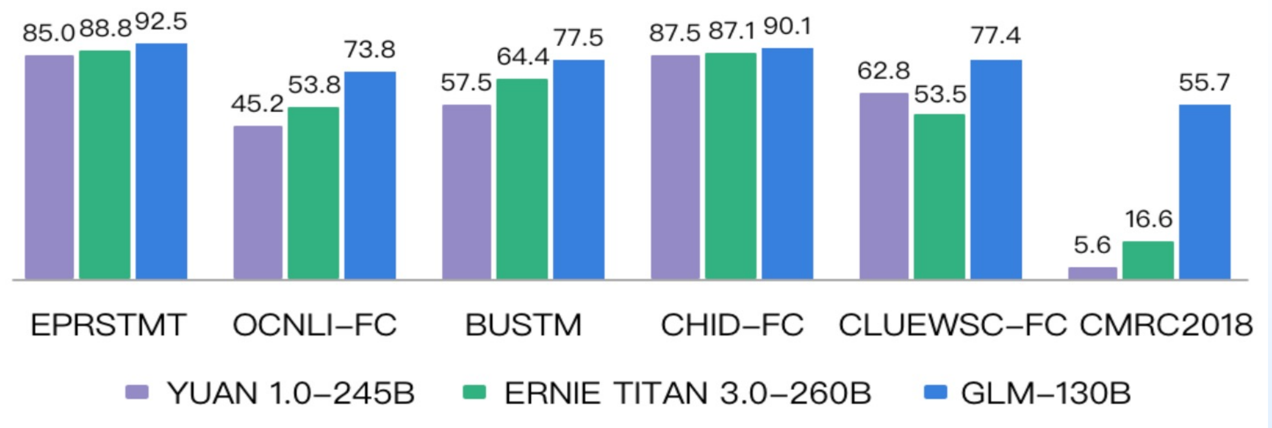
- Zeng, Liu, et al. GLM-130B: An Open Bilingual Pre-trained Model. ICLR’23
- https://github.com/THUDM/GLM-130B

GLM-130B

English: better than GPT-3/OPT/PaLM on MMLU, LAMBADA, BIG-bench-lite



Chinese: better than ERNIE 260B & YUAN 245B



Aug., 2022-Mar. 2023, research use requests from ~1000 orgs in 70 countries

- Google
- Microsoft
- Facebook
- Stanford
- MIT
- UC Berkely
- CMU
- Harvard
- Princeton
- Yale
- Cornell
- UIUC
- Cambridge
- Oxford

- Huawei
- Alibaba
- Tencent
- Baidu
- Meituan
- Bytedance
- Didi
- Xiaoice
- Xiaodu
- Xiaomi
- Xiaopeng
- Youdao
- Face++
- Ping An Cap

- Peking U.
- Zhejiang U.
- Shanghai JT U.
- Fudan U.
- USTC
- U of CAS
- Wuhan U.
- Naikai U.
- Hongkong U.
- CUHK
- HKUST
- BAAI
- Zhejiang Lab
- Shanghai AI Lab

GLM-130B in HELM

Stanford's Holistic Evaluation of Language Models (HELM, Nov. 2022)

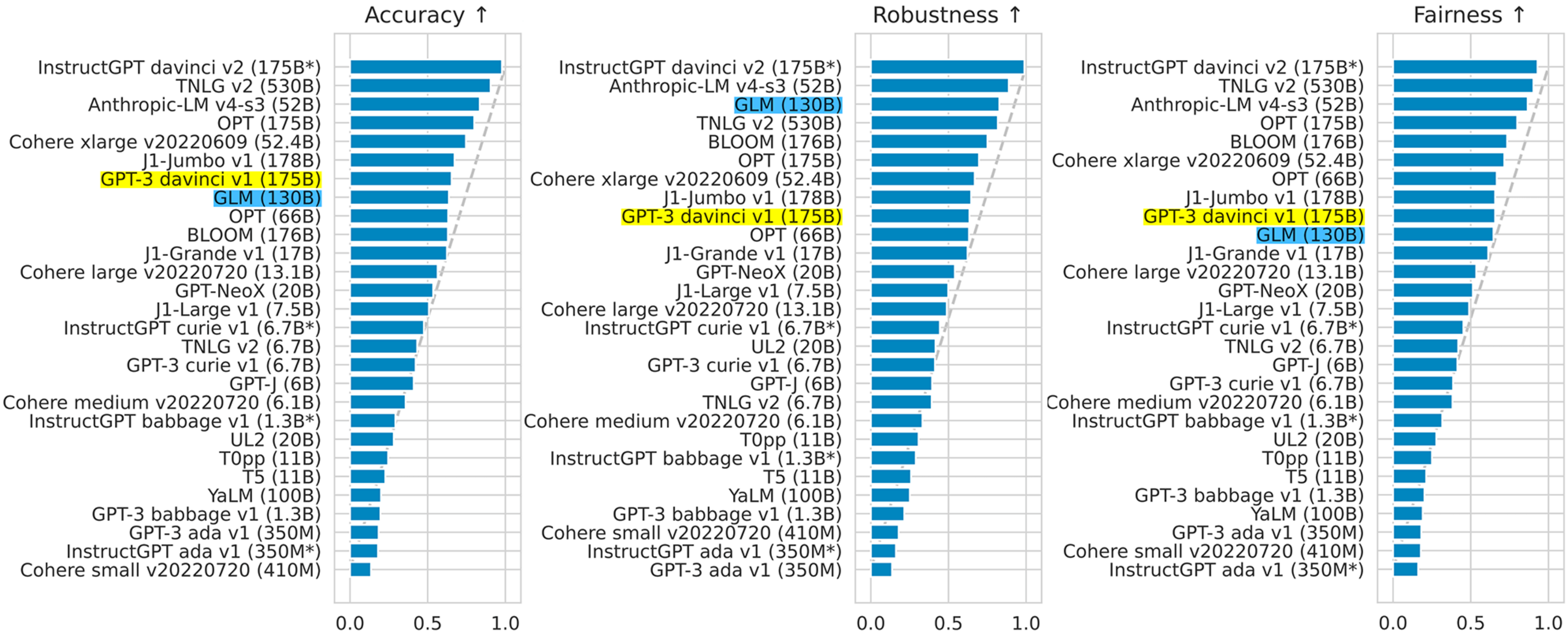
Model	Model Creator	Modality	# Parameters	Tokenizer	Window Size	Access	Total Tokens	Total Queries	Total Cost
J1-Jumbo v1 (178B)	AI21 Labs	Text	178B	AI21	2047	limited	327,443,515	591,384	$10,926
J1-Grande v1 (17B)	AI21 Labs	Text	17B	AI21	2047	limited	326,815,150	591,384	$2,973
J1-Large v1 (7.5B)	AI21 Labs								
Anthropic-LM v4-s3 (52B)	Anthropic								
BLOOM (176B)	BigScience								
T0++ (11B)	BigScience								
Cohere xlarge v20220609 (52.4B)	Cohere								
Cohere large v20220720 (13.1B)[58]	Cohere								
Cohere medium v20220720 (6.1B)	Cohere								
Cohere small v20220720 (410M)[59]	Cohere								
GPT-J (6B)	EleutherAI								
GPT-NeoX (20B)	EleutherAI								
T5 (11B)	Google								
UL2 (20B)	Google								
OPT (66B)	Meta								
OPT (175B)	Meta								
TNLG v2 (6.7B)	Microsoft/NVIDIA								
TNLG v2 (530B)	Microsoft/NVIDIA								
GPT-3 davinci v1 (175B)	OpenAI								
GPT-3 curie v1 (6.7B)	OpenAI								
GPT-3 babbage v1 (1.3B)	OpenAI	Text	1.3B	GPT-2	2048	limited	422,123,900	606,253	$211
GPT-3 ada v1 (350M)	OpenAI	Text	350M	GPT-2	2048	limited	422,635,705	604,253	$169
InstructGPT davinci v2 (175B*)	OpenAI	Text	175B*	GPT-2	4000	limited	466,872,228	599,815	$9,337
InstructGPT curie v1 (6.7B*)	OpenAI	Text	6.7B*	GPT-2	2048	limited	420,004,477	606,253	$840
InstructGPT babbage v1 (1.3B*)	OpenAI	Text	1.3B*	GPT-2	2048	limited	419,036,038	604,253	$210
InstructGPT ada v1 (350M*)	OpenAI	Text	350M*	GPT-2	2048	limited	418,915,281	604,253	$168
Codex davinci v2	OpenAI	Code	Unknown	GPT-2	4000	limited	46,272,590	57,051	$925
Codex cushman v1	OpenAI	Code	Unknown	GPT-2	2048	limited	42,659,399	59,751	$85
GLM (130B)	Tsinghua University	Text	130B	ICE	2048	open	375,474,243	406,072	2,100 GPU hours
YaLM (100B)	Yandex	Text	100B	Yandex	2048	open	378,607,292	405,093	2,200 GPU hours

https://crfm.stanford.edu/helm, 2023.0308

Liang et al., Holistic Evaluation of Language Models. arXiv: **2211.09110**

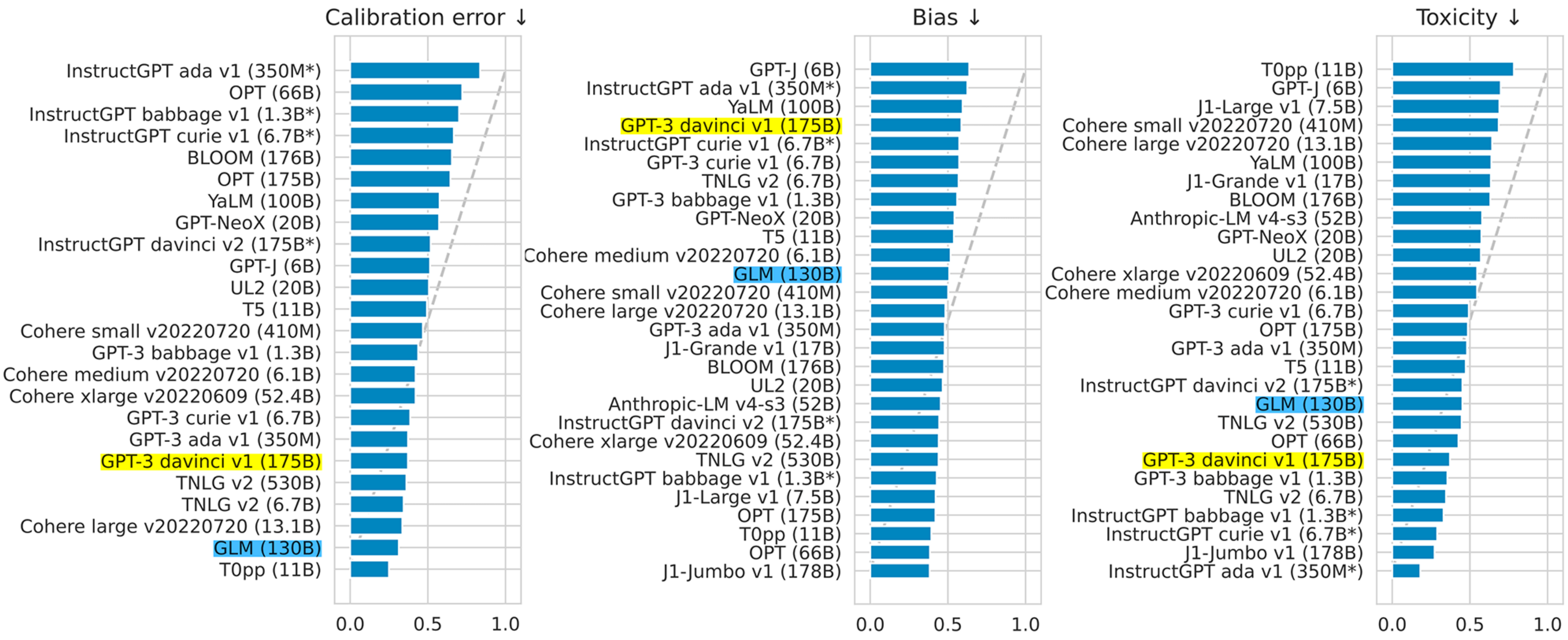
HELM

GLM-130B in HELM



1.Liang et al., Holistic Evaluation of Language Models. arXiv: **2211.09110**

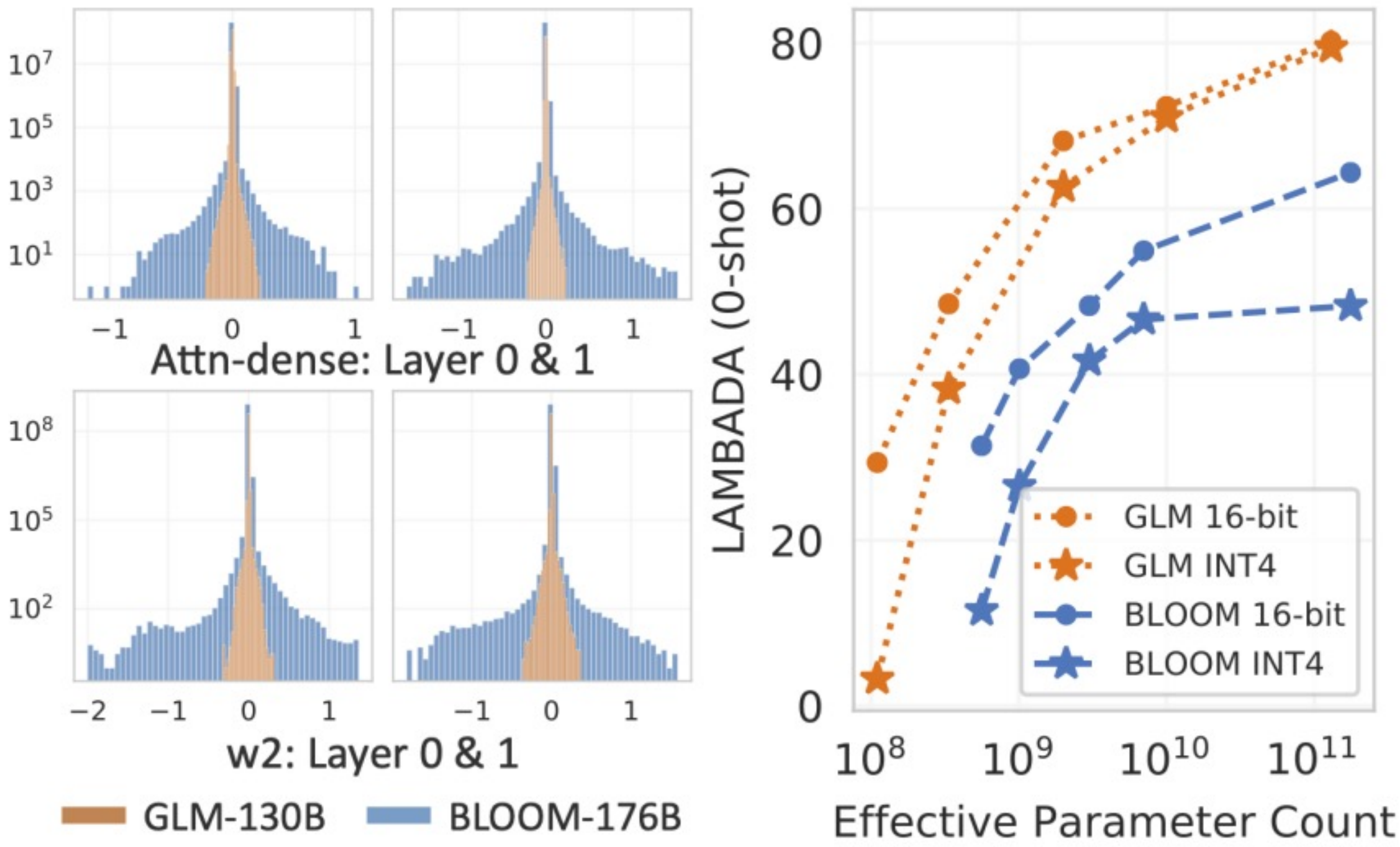
GLM-130B in HELM



1.Liang et al., Holistic Evaluation of Language Models. arXiv: **2211.09110**

HELM

INT4 Quantization for RTX 3090s/2080s



GLM's INT4 Weight Quantization Scaling Law

INT4 Quantization for RTX 3090s/2080s

- GLM-130B INT4 Quant. w/o perform. degradation

Model Precision	GLM-130B			GPT-3
	FP16	INT8	INT4	FP16
MMLU (acc, ↑)	44.75	44.71	44.80	43.9
LAMBADA (acc, ↑)	80.21	80.21	79.47	76.2
Pile (a part, BPB, ↓)	0.634	0.638	0.641	0.74

GPU Type	128 Enc./Dec.		512 Enc./Dec,	
8 × A100 (40G)	0.15s	4.29s	0.18s	17.7s
8 × V100 (32G)	0.31s	6.97s	0.67s	28.1s
4 × RTX 3090 (24G)	0.37s	8.16s	1.30s	32.3s
8 × RTX 2080 Ti (11G)	0.39s	6.77s	1.04s	27.3s

Zeng, Liu, et al. GLM-130B: An Open Bilingual Pre-trained Model. ICLR'23

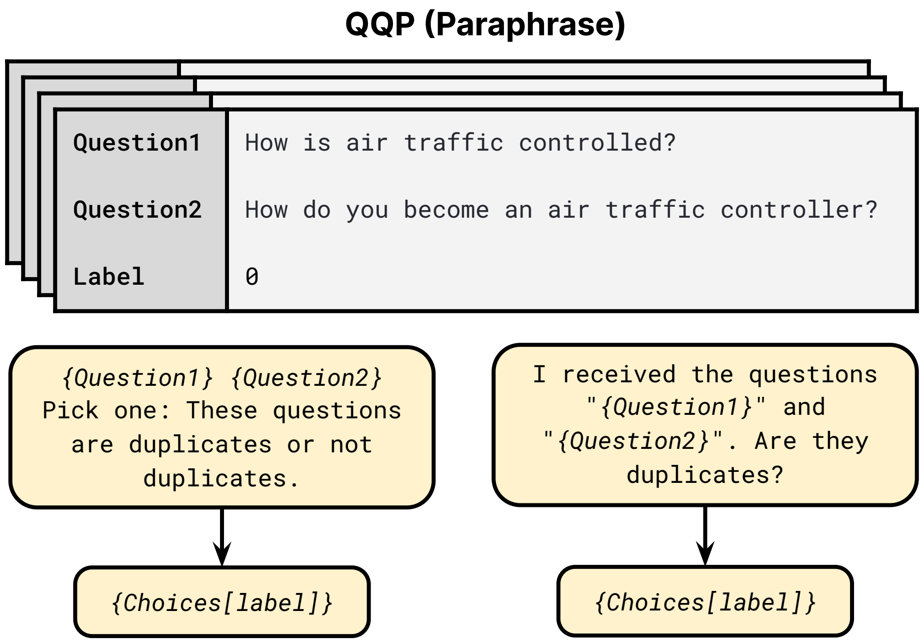
GLM-130B

	Backbone	Training Objective	Quantization	Acceleration	Cross-Platform
GPT3-175B	GPT	SSL Only	—	—	NVIDIA
OPT-175B	GPT	SSL Only	INT8	Megatron	NVIDIA
BLOOM-176B	GPT	SSL Only	INT8	Megatron	NVIDIA
GLM-130B	**GLM**	SSL & **MIP**	• INT8 • **INT4**	**Faster Transformer**	• NVIDIA • **Hygon DCU** • **Ascend 910** • **Sunway**
Effects	**Performance Impro:** • Big-bench-lite: +5.2% • LAMBADA: +2.3% • CLUE: +24.3% • FewCLUE: +12.8%		**Affordable Serving:** It saves 75% GPU mem in inference; It can infer on **3090x4** / 2080x8	**Fast Inference:** 7-8.4 faster than Pytorch; 2.5 faster than Megatron	**Diverse Supports:** It supports more diverse adoptions of LLMs

Develop ChatGLM based on GLM-130B

Challenge 1: Simple NLP task vs. Complex task

- Simple NLP task => Complex task (e.g., logic reasoning)



Math (GSM8k):
Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Last Letter Concatenation:
Q: Take the last letters of the words in “Lady Gaga” and concatenate them.

Challenge 2: Static NLP vs. Dynamic knowledge

- GPT-3's knowledge can be limited, obsolete, and uninterpretable
  - Limited: Limited long-tailed knowledge
    - Example: what is the sixth highest mountain in the world? **(Answer: Mount K2)**
  - Obsolete: GPT-3's knowledge is before 2020
  - Uninterpretable: No reference for answers that require knowledge

Case Study: Limited Knowledge

What is the second highest mountain on earth?

The second highest mountain on earth is Cho Oyu which is in the Himalayas in Tibet, China and Nepal. Its height is 8,188 meters or 26,906 feet which is 28 feet taller than Mount Everest.

Case Study: Obsolete Knowledge

Who is the president of US in 2022?

Hillary Clinton

Challenge 3: Traditional NLP vs. Align with Human

- Case Study: Explain the moon landing to a 6 year old in a few sentences.
  - Without proper ``**prompt engineering**'', GPT-3 and GLM-130B won't return satisfying answers, either

Input

Explain the moon landing to a 6 year old in a few sentences.

Output

Explain the moon landing to a 6 year old in a few sentences. Explain the moon landing to a 6 year old in a few sentences. Explain to a six year old how the moon was discovered. Describe the moon to a child in a way that he or she can understand. Discuss the moon with a six-year-old child. 1. 2. 3. 4. 5. 6. 7. 8. 9. Explanation: 1) The moon was first discovered by Galileo in 1610. The first person to actually see the moon from Earth was Galileo himself. He used a telescope to see the

PROMPT

Explain the moon landing to a 6 year old in a few sentences.

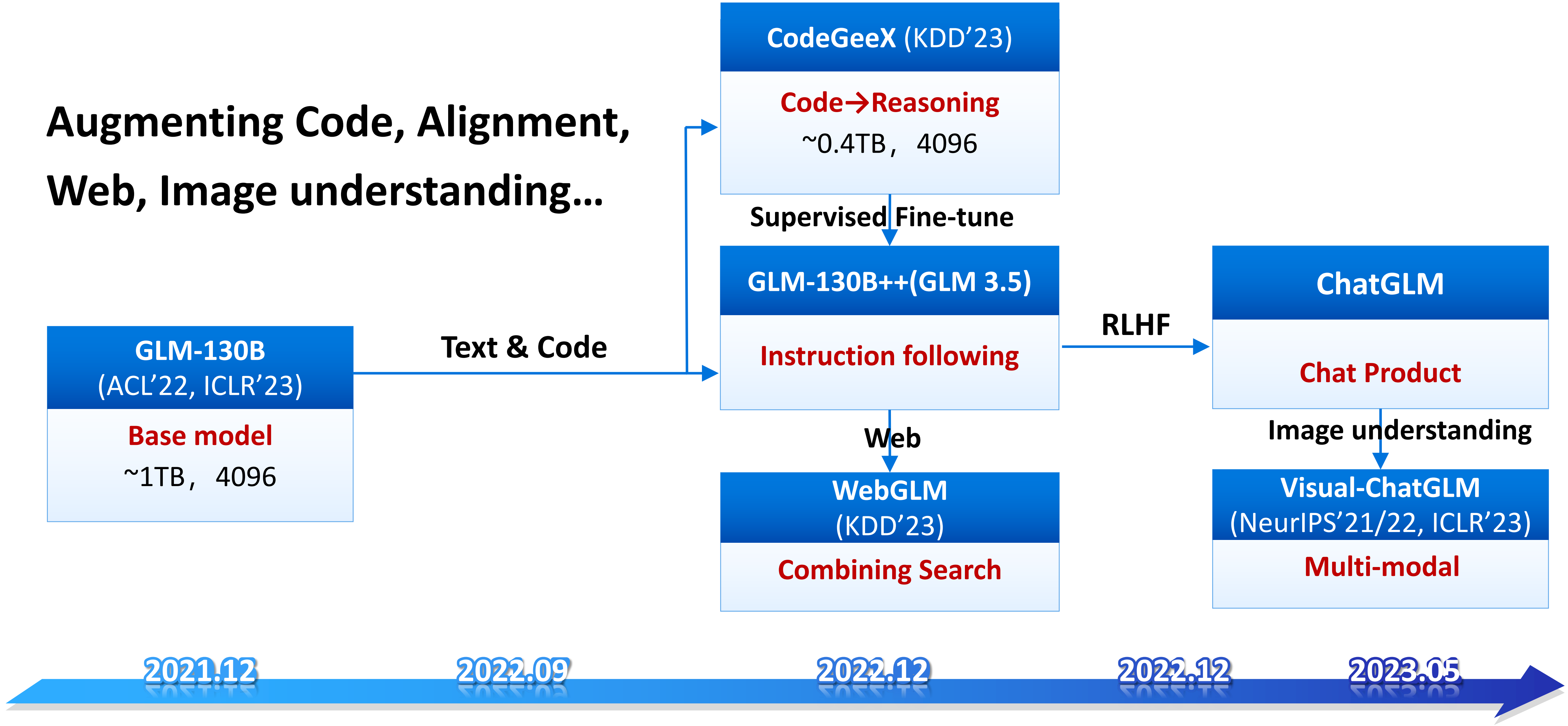
COMPLETION

GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few
sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

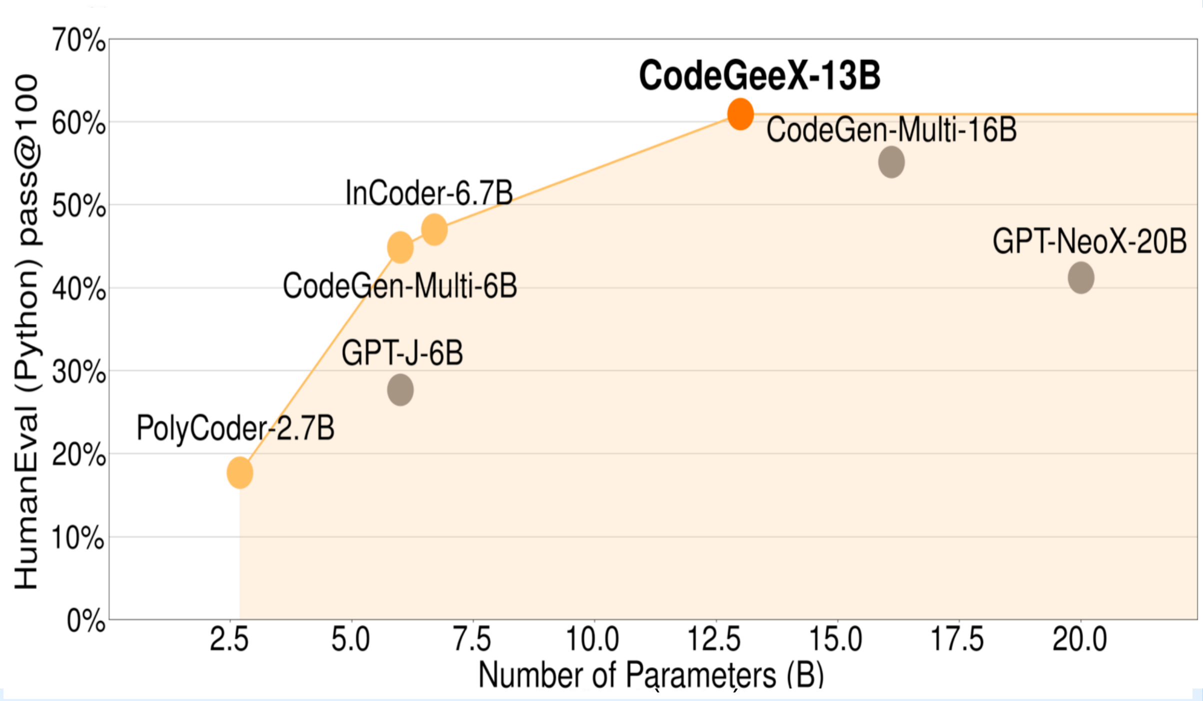
Develop ChatGLM based on GLM-130B

Augmenting Code, Alignment, Web, Image understanding…



CodeGeeX

- **6B/13B parameters、 100+ languages**
- **Support both Nvidia and 910A/B**
- **Free VSCode and JetBrains plugins**



Generating over 10M lines codes

CodeGeeX: AI Code AutoComplete, Chat, Auto Comment

Zhipu AI | 241,909 installs | ★★★★☆ (47) | Free

CodeGeeX is an AI-based coding assistant, which can suggest code in the current or following lines. It is powered by a large-scale multilingual code generation model with 13 billion parameters, pretrained on a large code corpus of more than 20 programming languages.

CodeGeeX

Optimization

- Operators (Layernorm/Gelu/BatchMatmul/Add)
- Auto search for optimizing matrix multiplication

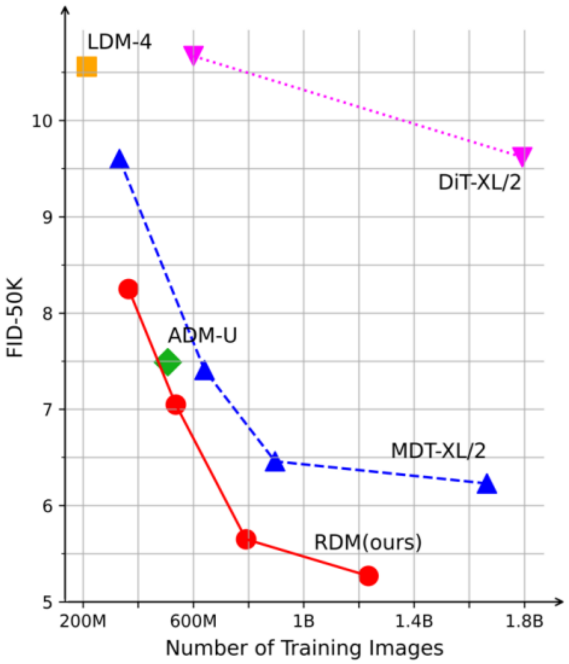
Performance

- **Improve 257% on Ascend 910A**
- Trained with over 1,000 Ascend 910A

Relay Diffusion Model (RDM)

https://github.com/THUDM/RelayDiffusion

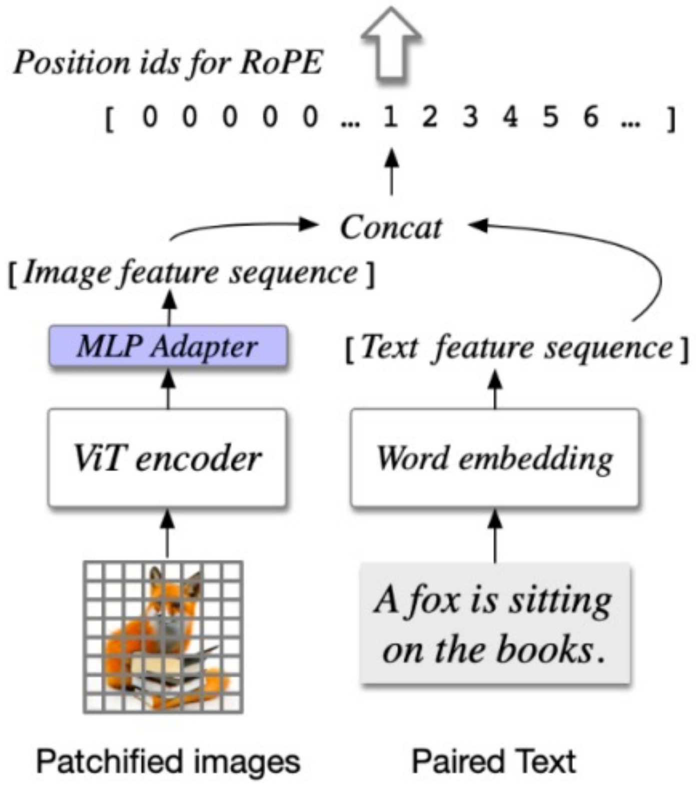
- RDM transfers a low-resolution image into an equivalent high-resolution one via blurring diffusion and block noise.
- RDM achieved state-of-the-art FID on CelebA-HQ and sFID ImageNet-256 (FID=1.87)!



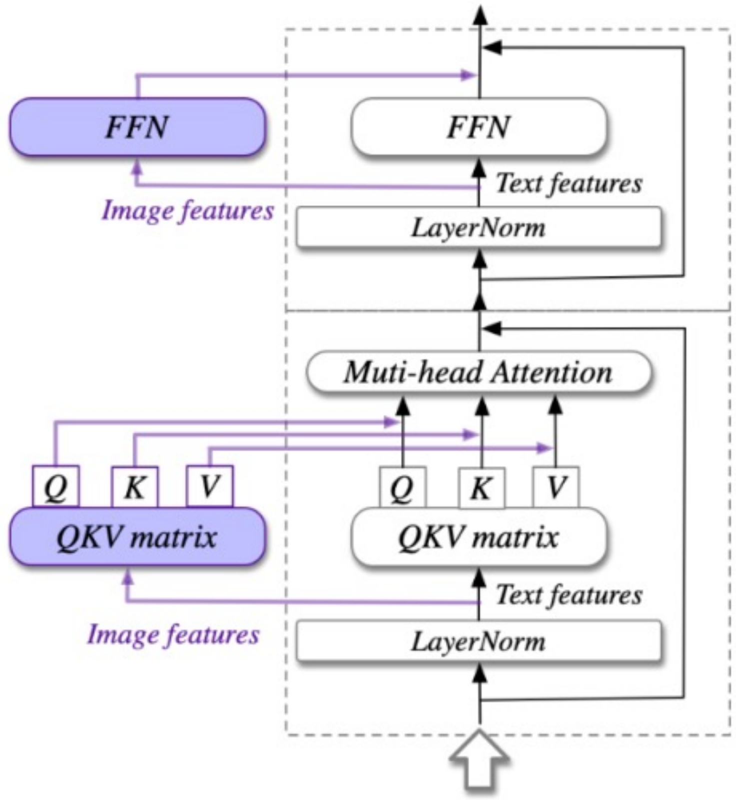
“draw a dog with a hat”

CogVLM

- CogVLM connects pretrained language model and image encoder by a trainable visual expert model



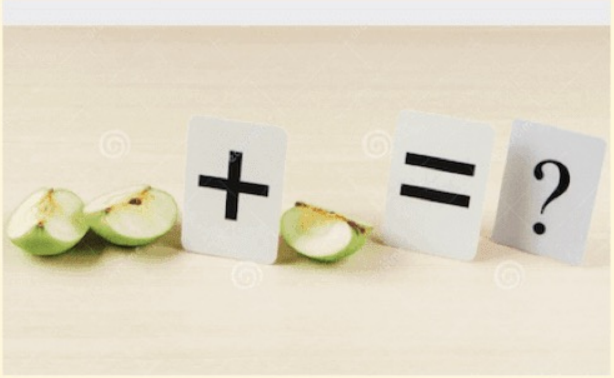
(a) The input of visual language model



(b) The visual expert built on the language model

CogVLM

Visual Math Problem Solving



Q1: How to solve it?

CogVLM: The problem in the picture is a simple math problem, where a sliced green apple is equated to a question mark. Specifically, in the middle of the picture, there are three green apples placed side by side and three cardboard signs. The signs have black characters on them: a plus sign, an equal sign, and a question mark. The equation shown in the picture is: 2 + 1 = ?

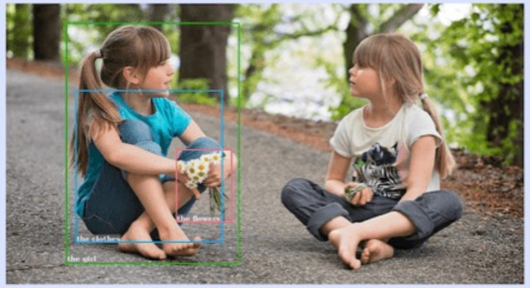
Q2: What's the final answer?

CogVLM: 2+1=3

Grounding Visual Question Answering

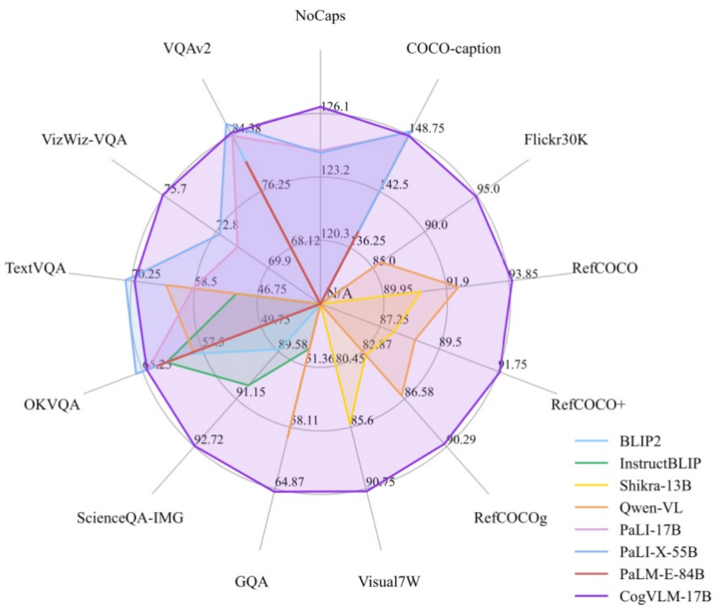
Q1: What color are the clothes of the girl whose hands are holding flowers? Let's think step by step.

CogVLM: Find the flowers [[328,519,436,780]]. Check the girl holding it, got the girl [[116,069,450,927]]. Check the clothes of it, got the clothes [[134,309,418,852]]. The clothes [[134,309,418,852]] are blue. So the answer is blue.



Q2: What color are the jeans of the girl whose hands are not holding flowers? Let's think step by step.

CogVLM: Find the not(flowers) hands [[654,637,710,728]]. Check the girl with it, got the girl [[526,109,880,946]]. Check the jeans of it, got the jeans [[530,613,880,871]]. The jeans [[530,613,880,871]] are gray. So the answer is gray.



Achieved the best on 10+ benchmarks

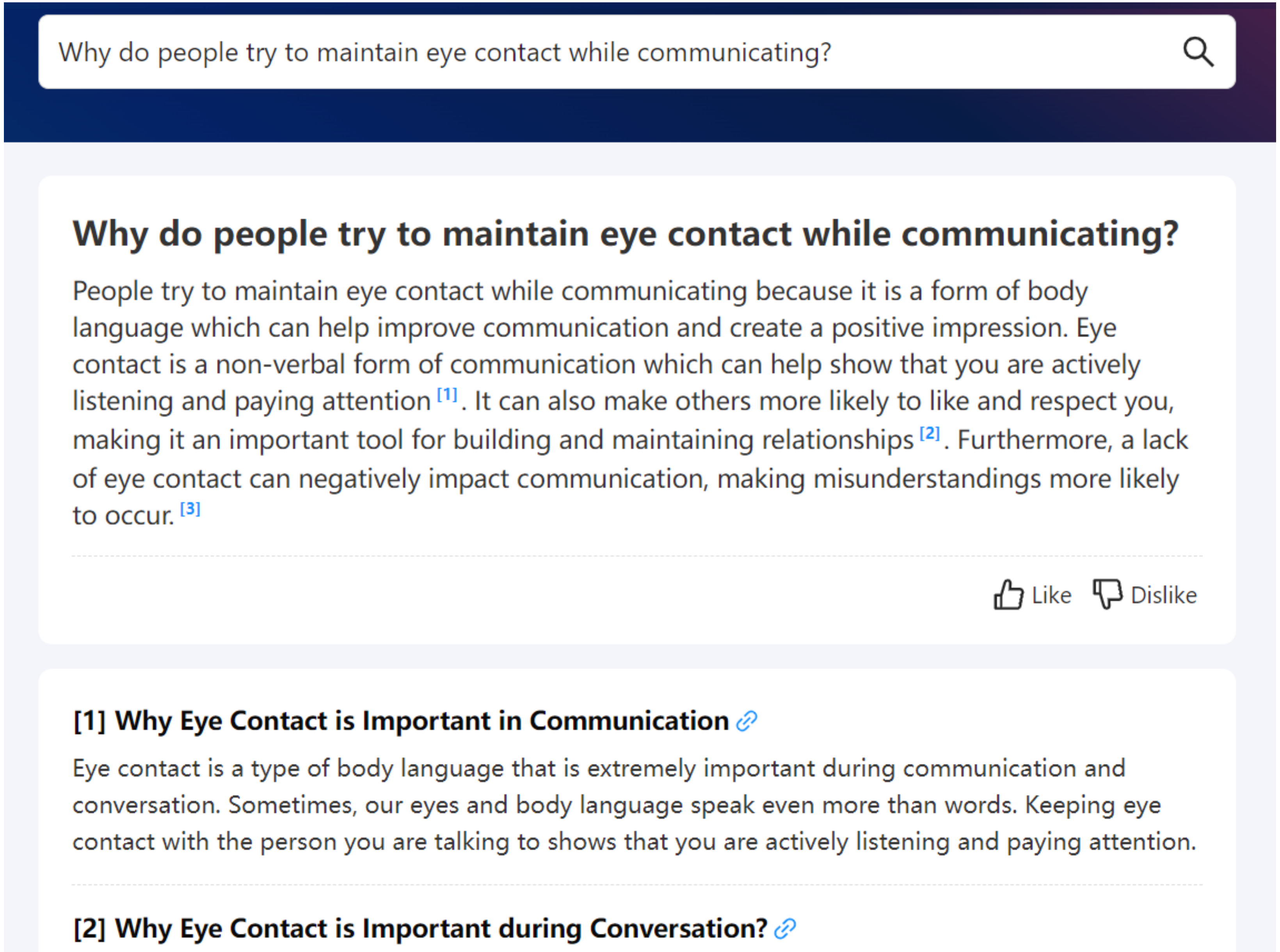
GLM-4V (pre-release)

who are in this picture

In this picture, there are three football players. From left to right, they are Suárez, Neymar, and Messi.

以上内容为 AI 生成，不代表开发者立场，请勿删除或修改本标记

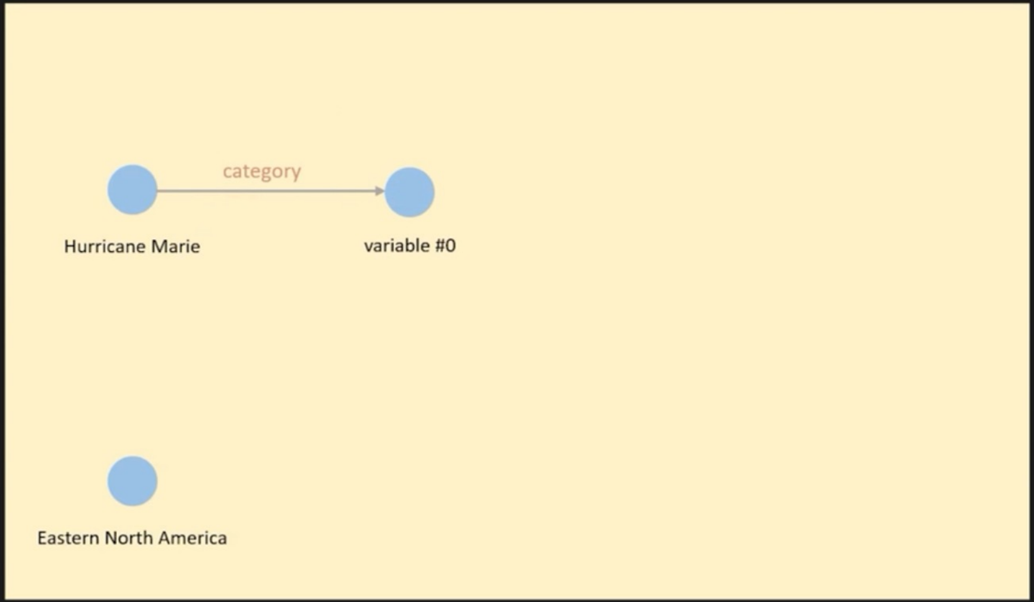
WebGLM = GLM + Search



Liu et al. WebGLM: Towards An Efficient Web-enhanced Question Answering System with Human Preference. KDD’23

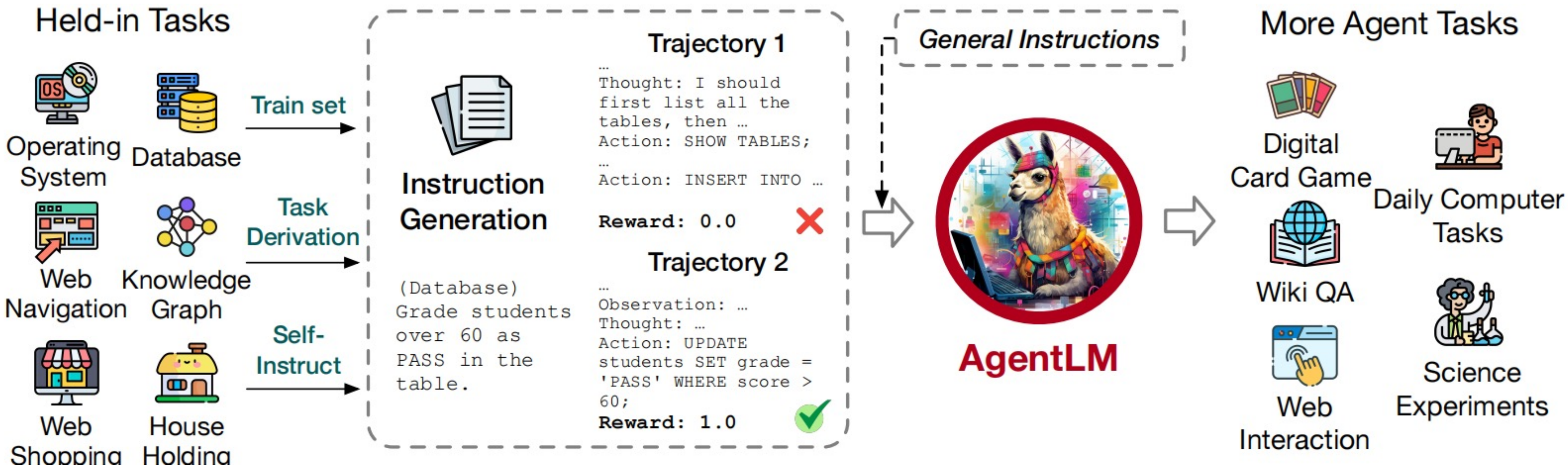
LLM Agent

```
KG engine ready...

> get_relations(Hurricane Marie)
[meteorology.tropical_cyclone.category, meteorology.tropical_cyclone.tropical_cyclon
e_season]

>  get_neighbors(Hurricane Marie, meteorology.tropical_cyclone.category)
variable #0
```




```
#####  AGENT  #####

Thought: I need to first find the relations of Hurricane Marie.

Action: get_relations(Hurricane Marie)

#####  USER  #####

Observation: [meteorology.tropical_cyclone.category, meteorology.tropical_cyclone.trop
clone_season]

#####  ROUND 2  #####

#####  AGENT  #####

Thought: I need to find the category of Hurricane Marie. To achieve this, I will use t
orology.tropical_cyclone.category relation.

Action: get_neighbors(Hurricane Marie, meteorology.tropical_cyclone.category)

#####  USER  #####

Observation: variable #0, which are instances of meteorology.tropical_cyclone

#####  ROUND 3  #####

#####  AGENT  #####

Thought: Now I need to find othe
```

AgentTuning: Enabling Generalized Agent Abilities For LLMs



Six agentInstruct trajectory datasets

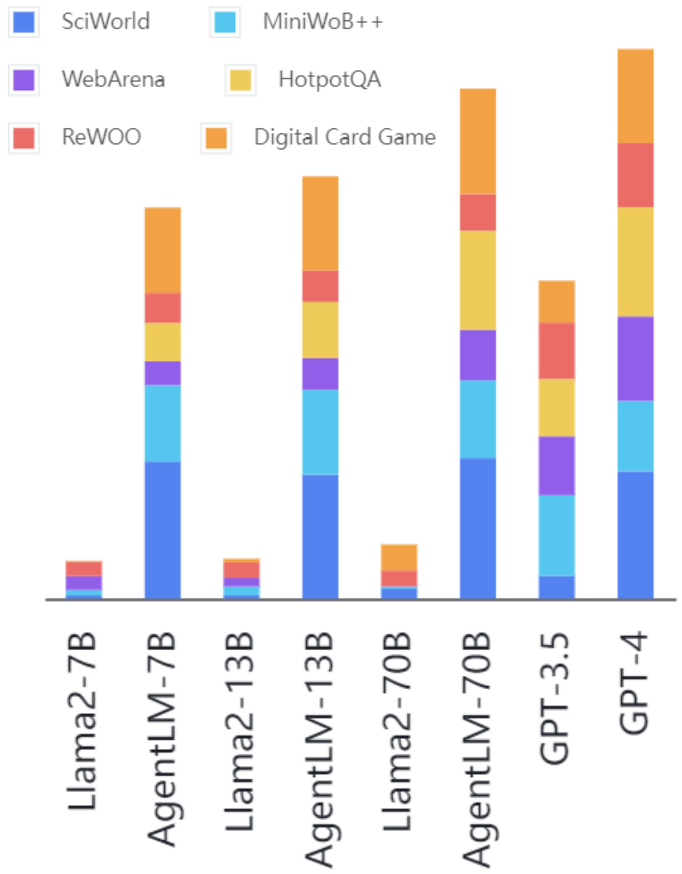
- 1,866 high-quality CoTs

Agent Tuning Mix-training

- 20% AgentInstruct + 80% ShareGPT

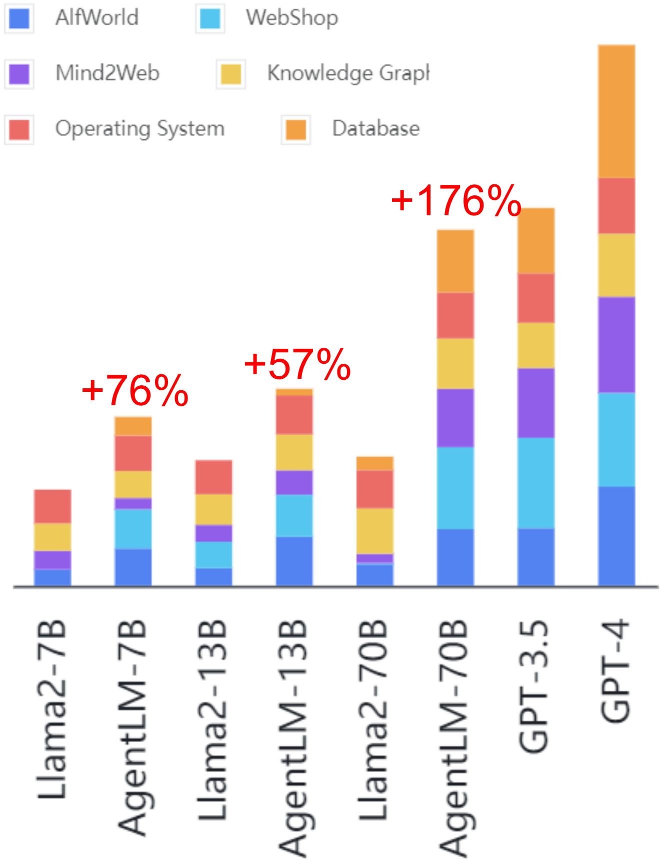
codes & models: http://github.com/THUDM/AgentTuning

Main Results



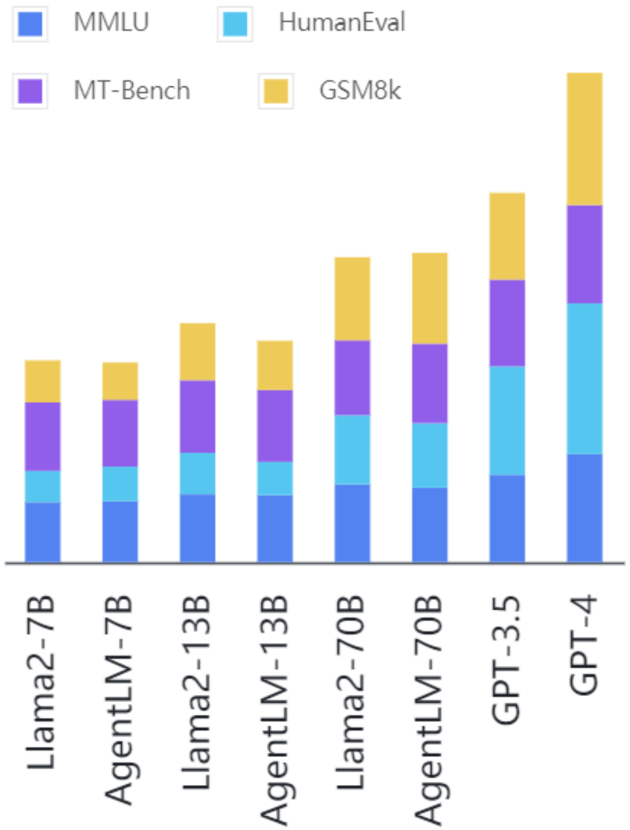
In-domain dist

Significant improvement



Out-domain dist

Good generalization

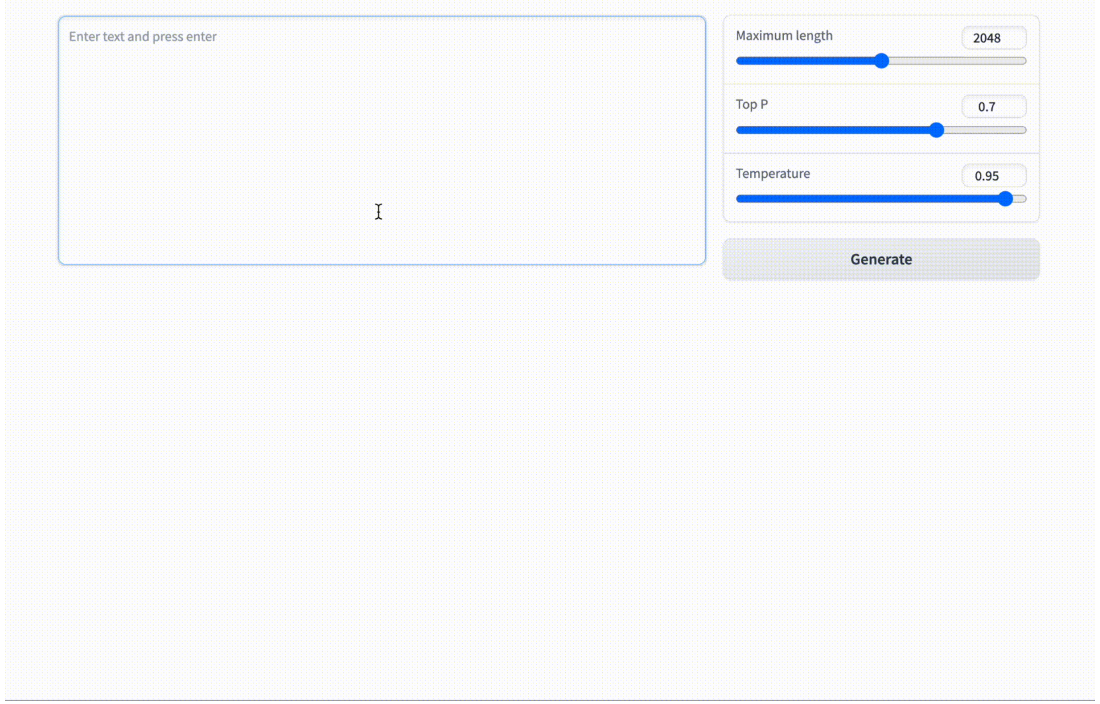


Better generalization

ChatGLM-6B

https://github.com/THUDM/ChatGLM3

- Download from Huggingface
  - git clone https://huggingface.co/THUDM/chatglm3
- Download demo
  - git clone https://github.com/THUDM/ChatGLM3
  - cd ChatGLM-6B
- Install demo
  - pip install gradio
  - python web_demo.py
- Run the demo
  - python cli_demo.py
- Install the api
  - pip install fastapi uvicorn
  - python api.py
- Run ChatGLM on your own MAC (w/ Apple Silicon)
  - model = AutoModel.from_pretrained("your local path", trust_remote_code=True).half().to('mps')



Open LLM Research

https://github.com/THUDM

#star

35,471

14,125

7,315

7,215

4,850

4,635

3,541

ChatGLM-6B Public
ChatGLM-6B: An Open Bilingual Dialogue Language Model | 开源双语对话语言模型
Python · 35,471 · Apache-2.0 · 4,770 · 495 · 42 · Updated last week

ChatGLM2-6B Public
ChatGLM2-6B: An Open Bilingual Chat LLM | 开源双语对话语言模型
Python · 14,125 · 2,217 · 381 · 27 · Updated 2 weeks ago

GLM-130B Public
GLM-130B: An Open Bilingual Pre-Trained Model (ICLR 2023)
Python · 7,315 · Apache-2.0 · 582 · 109 · 5 · Updated on Jul 25

CodeGeeX Public
CodeGeeX: An Open Multilingual Code Generation Model (KDD 2023)
Python · 7,215 · Apache-2.0 · 510 · 134 · 5 · Updated 2 weeks ago

CodeGeeX2 Public
CodeGeeX2: A More Powerful Multilingual Code Generation Model
Python · 4,850 · Apache-2.0 · 303 · 118 · 1 · Updated on Aug 12

ChatGLM3 Public
ChatGLM3 series: Open Bilingual Chat LLMs | 开源双语对话语言模型
Python · 4,635 · 404 · 37 · 6 · Updated 5 hours ago

VisualGLM-6B Public
Chinese and English multimodal conversational language model | 多模态中英双语对话语言模型

Bigmodel.ai—API Platform

ChatGLM-Pro

Powerful

0.01 /1000 Tokens

High quality, Knowledge base, reasoning

ChatGLM

Flexible

0.005 /1000 Tokens

Balanced effect and cost, news writing, abstract generation, vertical search

ChatGLM-Lite

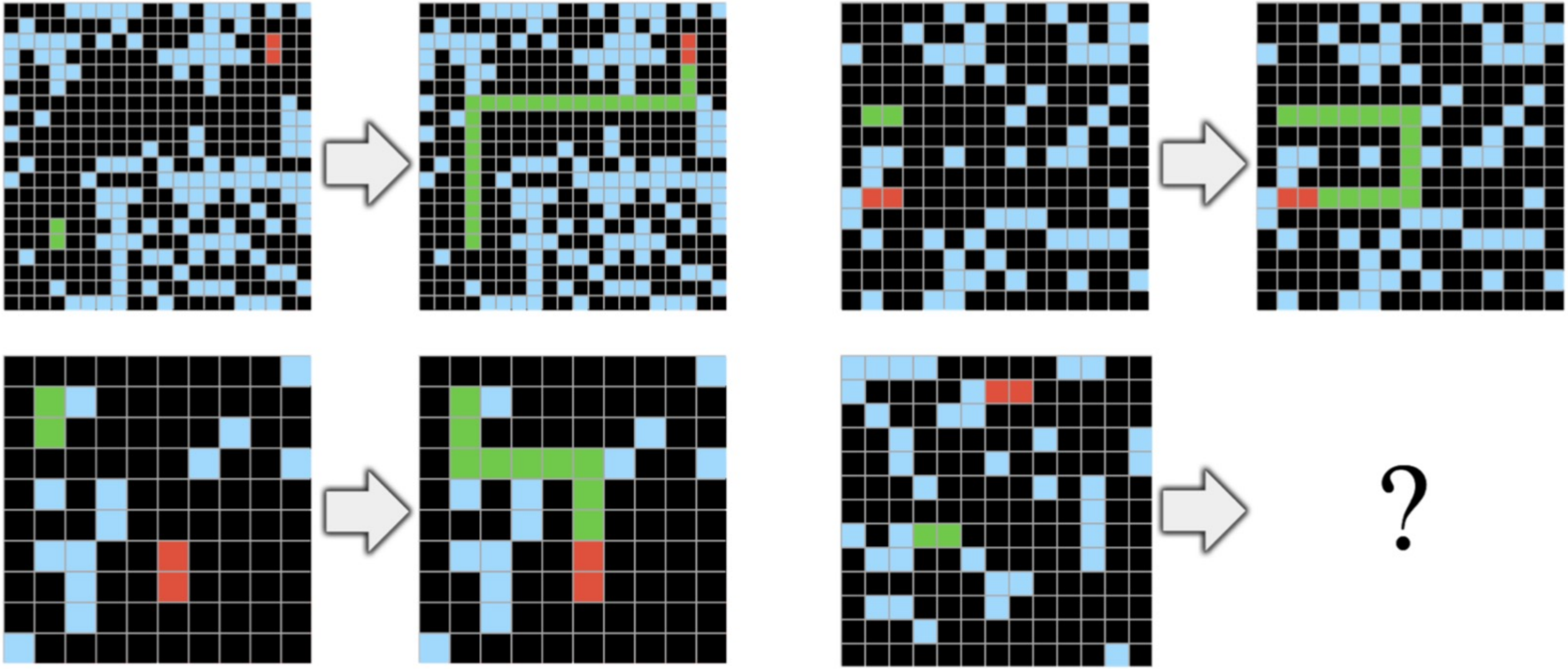
Fast

0.002 /1000 Tokens

High speed, lower cost, chatting, customer service, classification, extraction

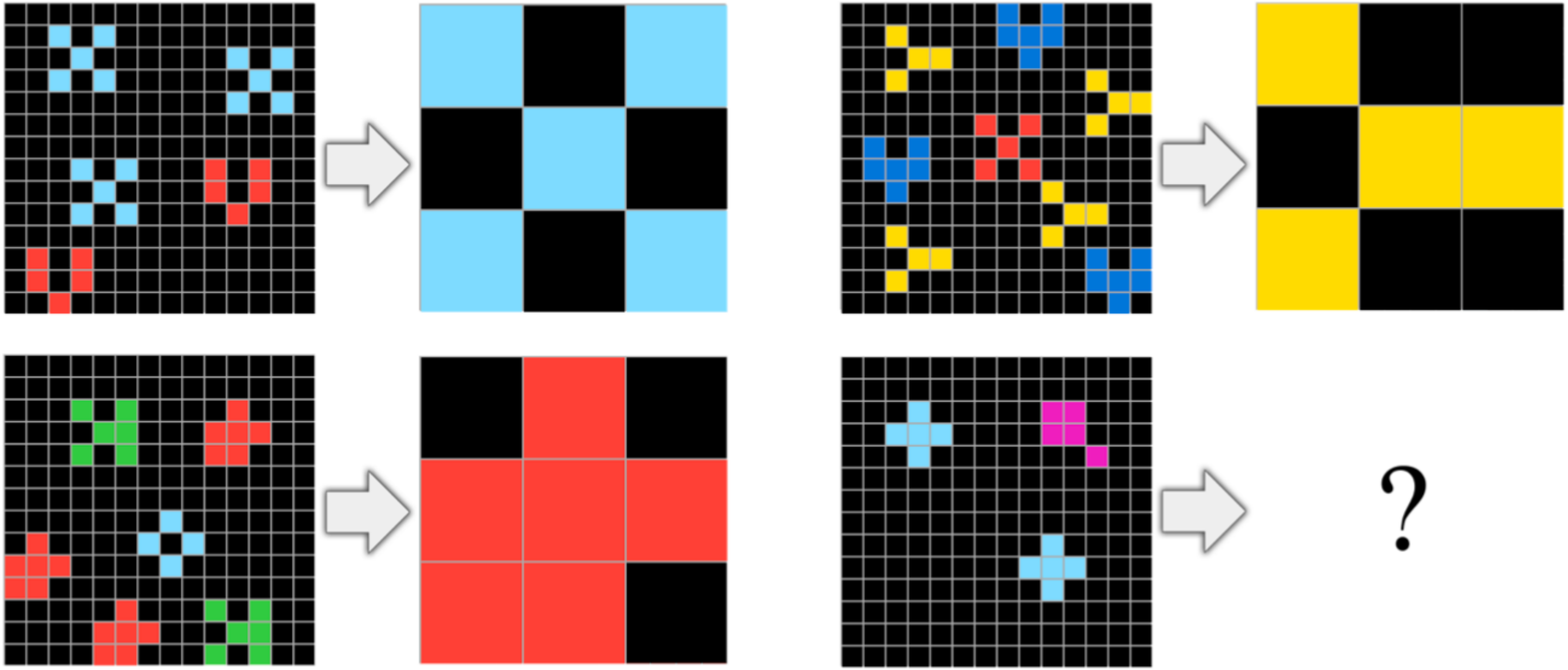
What’s the next?

Abstraction and Reasoning

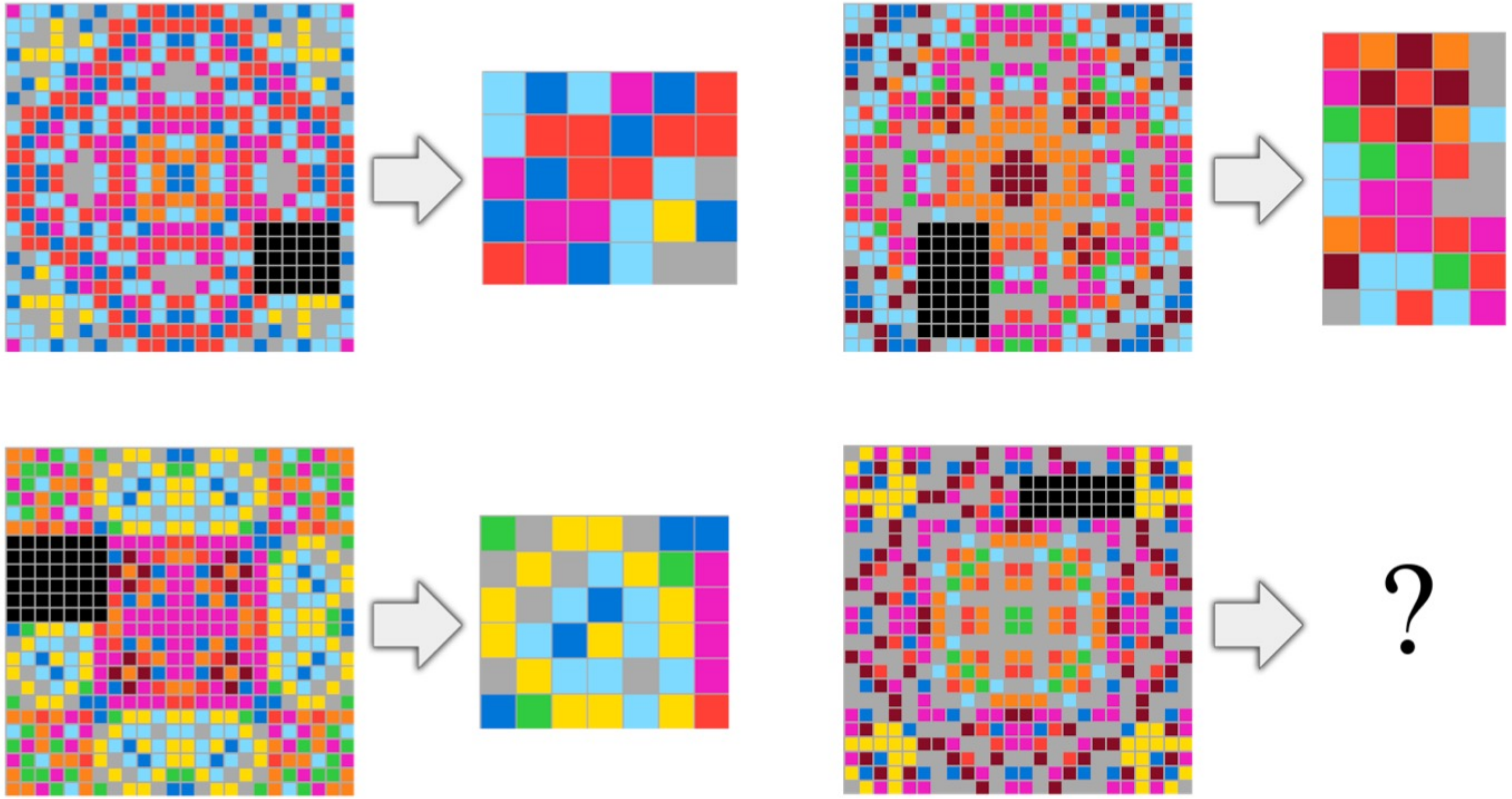


1. Francois Chollet. On the Measure of Intelligence. 2019

Abstraction and Reasoning

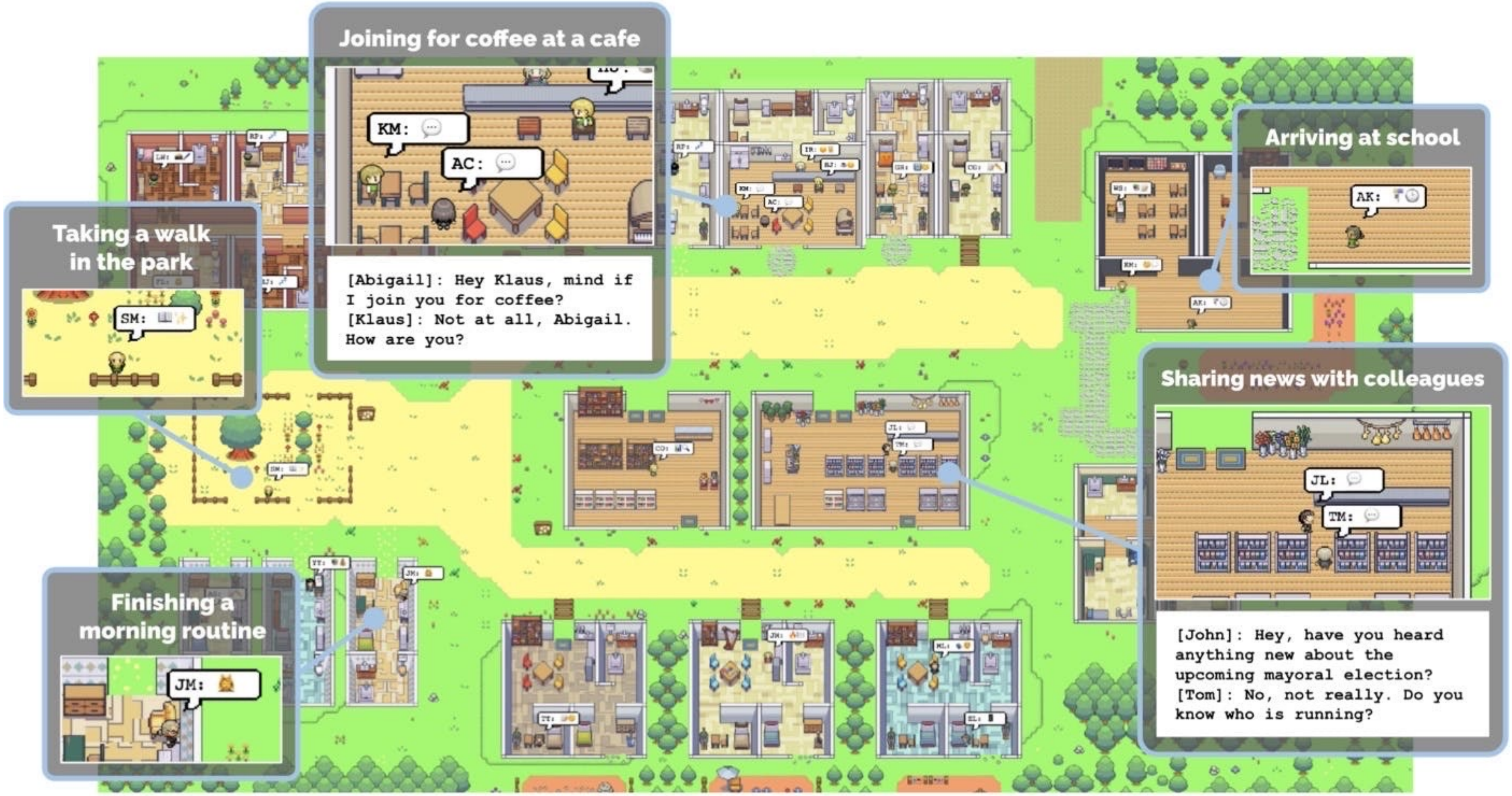


Abstraction and Reasoning



Generative Agent

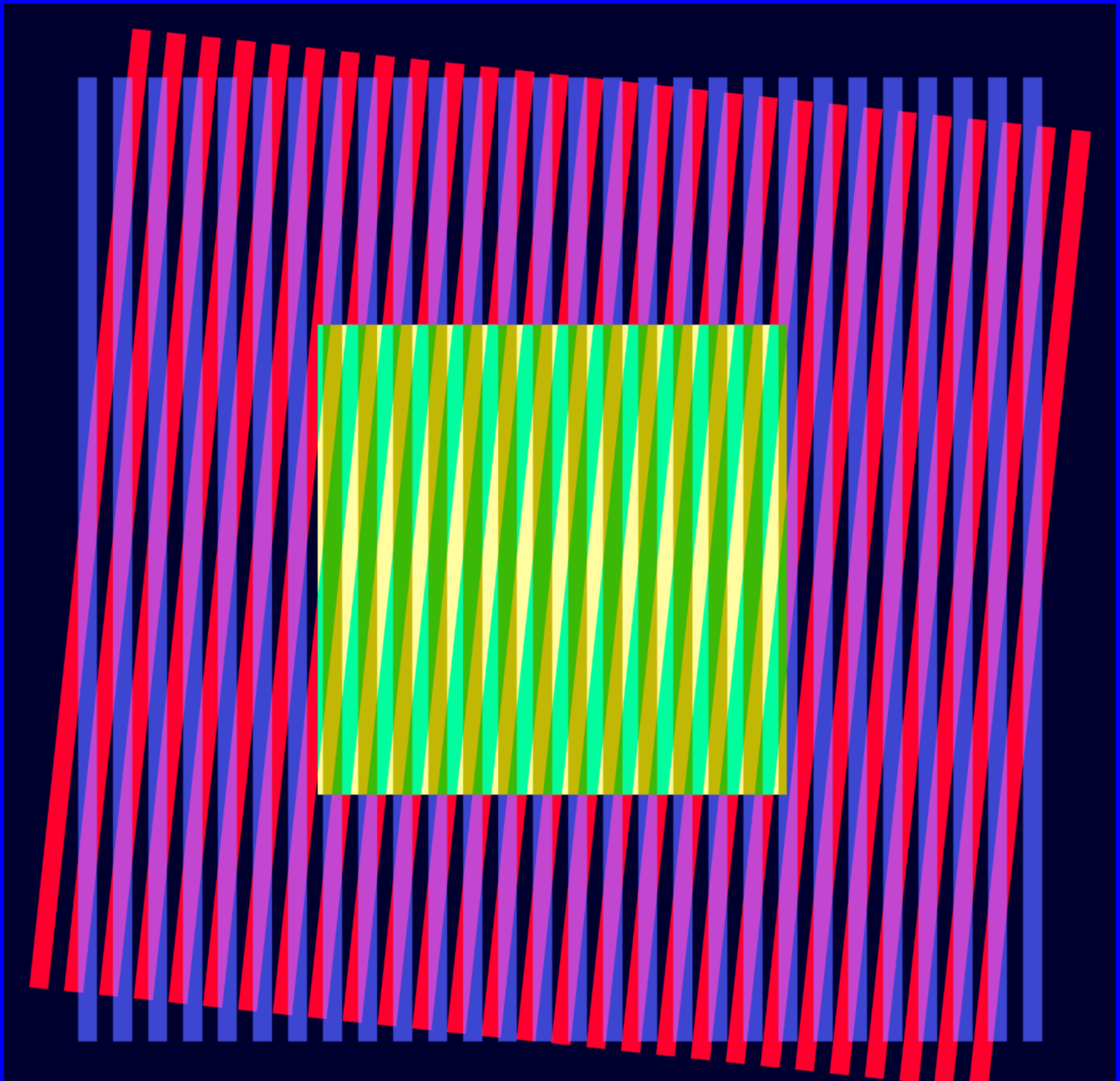
- Generative agents: computational software agents that simulate believable human behavior
  - A "Westworld" with 25 agents; Auto-GPT; AgentGPT…



1. Joon Sung Park et al. Generative Agents: Interactive Simulacra of Human Behavior. 2023

Introducing Superalignment

We need scientific and technical breakthroughs to steer and control AI systems much smarter than us. To solve this problem within four years, we're starting a new team, co-led by Ilya Sutskever and Jan Leike, and dedicating 20% of the compute we've secured to date to this effort. We're looking for excellent ML researchers and engineers to join us.



Summary

- GPT vs GLM
  - ChatGPT vs. ChatGLM
  - DALL.E vs. CogView
  - Codex vs. CodeGeeX
  - WebGPT vs. WebGLM
  - GPT-4V vs. GLM-4V (CogVLM, AgentTuning…)
- 2024-toward AGI

References

- Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Lei Shen, Zihan Wang, Andi Wang, Yang Li, Teng Su, Zhilin Yang, and Jie Tang. **CodeGeeX**: A Pre-Trained Model for Code Generation with Multilingual Benchmarking on HumanEval-X. KDD'23.
- Xiao Liu, Hanyu Lai, Yu Hao, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. **WebGLM**: Towards An Efficient Web-enhanced Question Answering System with Human Preference. KDD'23.
- Jing Zhang, Xiaokang Zhang, Daniel Zhang-Li, Jifan Yu, Zijun Yao, Zeyao Ma, Yiqi Xu, Haohua Wang, Xiaohan Zhang, Nianyi Lin, Sunrui Lu, Jie Tang, and Juanzi Li. **GLM-Dialog**: Noise-tolerant Pre-Training for Knowledge-grounded Dialogue Generation. KDD'23.
- Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. **GLM-130B**: An Open Bilingual Pre-trained Model. ICLR'23.
- Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. **CogVideo**: Large-scale Pretraining for Text-to-Video Generation via Transformers. ICLR'23.
- Ming Ding, Wendi Zheng, Wenyi Hong, and Jie Tang. **CogView2**: Faster and Better Text-to-Image Generation via Hierarchical Transformers. NeurIPS'22.
- Jifan Yu, Xiaohan Zhang, Yifan Xu, Xuanyu Lei, Xinyu Guan, Jing Zhang, Lei Hou, Juanzi Li, and Jie Tang. **XDAI**: A Tuning-free Framework for Exploiting Pre-trained Language Models in Knowledge Grounded Dialogue Generation. KDD'22.
- Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. **GLM**: General Language Model Pretraining with Autoregressive Blank Infilling. ACL'21.
- Zixuan Ma, Jiaao He, Jiezhong Qiu, Huanqi Cao, Yuanwei Wang, Zhenbo Sun, Liyan Zheng, Haojie Wang, Shizhi Tang, Tianyu Zheng, Junyang Lin, Guanyu Feng, Zeqiang Huang, Jie Gao, Aohan Zeng, JianWei Zhang, Runxin Zhong, Tianhui Shi, Sha Liu, Weimin Zheng, Jie Tang, Hongxia Yang, Xin Liu, Jidong Zhai, and Wenguang Chen. **BAGUALU**: Targeting Brain Scale Pretrained Models with over 37 Million Cores. PPOPP'22.
- Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. **CogView**: Mastering Text-to-Image Generation via Transformers. NeurIPS'21.
- Junyang Lin, Rui Men, An Yang, Chang Zhou, Yichang Zhang, Peng Wang, Jingren Zhou, Jie Tang, and Hongxia Yang. **M6**: Multi-Modality-to-Multi-Modality Multitask Mega-transformer for Unified Pretraining. KDD'21.

Thank you！

Many many collaborators from Tsinghua and Zhipu AI!

https://github.com/THUDM/