LLMBox: A Comprehensive Library for Large Language Models

Tianyi Tang[1*], **Yiwen Hu**[1*],
Bingqian Li[1†], **Wenyang Luo**[1†], **Zijing Qin**[3†], **Haoxiang Sun**[2†], **Jiapeng Wang**[1†],
Shiyi Xu[1], **Xiaoxue Cheng**[1], **Geyang Guo**[1], **Han Peng**[1], **Bowen Zheng**[1],
Yiru Tang[1], **Yingqian Min**[1], **Yushuo Chen**[1], **Jie Chen**[1], **Yuanqian Zhao**[1],
Luran Ding[1], **Yuhao Wang**[1], **Zican Dong**[1], **Chunxuan Xia**[1],
Junyi Li[1], **Kun Zhou**[2], **Wayne Xin Zhao**[1 ✉], **Ji-Rong Wen**[1,2],
[1] Gaoling School of Artificial Intelligence, Renmin University of China
[2] School of Information, Renmin University of China
[3] School of Computer Science and Technology, Xidian University
steventianyitang@outlook.com huyiwenwen@foxmail.com batmanfly@gmail.com

Abstract

To facilitate the research on large language models (LLMs), this paper presents a comprehensive and unified library, **LLMBox**, to ease the development, use, and evaluation of LLMs. This library is featured with three main merits: (1) *a unified data interface* that supports the flexible implementation of various training strategies, (2) *a comprehensive evaluation* that covers extensive tasks, datasets, and models, and (3) *more practical consideration*, especially on user-friendliness and efficiency. With our library, users can easily reproduce existing methods, train new models, and conduct comprehensive performance comparisons. To rigorously test LLMBox, we conduct extensive experiments in a diverse coverage of evaluation settings, and experimental results demonstrate the effectiveness and efficiency of our library in supporting various implementations related to LLMs. The detailed introduction and usage guidance can be found at https://github.com/RUCAIBox/LLMBox.

1 Introduction

Recent years have witnessed the rapid progress of large language models (LLMs) (Zhao et al., 2023). In the research community, great efforts have been devoted to the release of well-trained LLMs, the design of special tuning and inference methods, and the conduct of systematic capacity evaluation. However, the reproducibility and fair comparison of existing studies should still be emphasized, since they are mostly developed in different ways or frameworks. Without the standardized and unified implementation, it would take substantial efforts to reproduce or improve upon existing research work.

Considering the above issue, in this paper, we present a comprehensive library, called **LLMBox**, for easing the development, use, and evaluation of LLMs. In particular, our library focuses on building a comprehensive and unified framework (including training, inference, and evaluation) for better supporting LLM-based research and applications. Although there are already several open-source libraries for LLMs (Kwon et al., 2023; Gao et al., 2023a; hiyouga, 2023), they typically focus on a certain or some stage(s) of LLMs (either pre-training or fine-tuning) or conduct the training pipeline of LLMs in a separate way. Moreover, they can seldom support comprehensive and unified evaluation of various LLMs.

In order to better facilitate research on LLMs, LLMBox introduces a series of new features for the library design, which can be summarized into three major aspects below:

• *Unified data interface.* We design a unified data interface to encapsulate different formats of training data, including both plain texts and instruction data. With this interface, LLMBox can flexibly support the implementation of various strategies, such as dynamic mixture proportion (Xie et al., 2023) and combined training with pre-training and instruction data (Zeng et al., 2022). Furthermore, we extensively support mainstream training methodologies, including parameter-efficient tuning (*e.g.,* LoRA (Hu et al., 2022)) and alignment tuning (*e.g.,* PPO (Schulman et al., 2017)).

• *Comprehensive evaluation.* To support a comprehensive comparison of LLMs' performance, our library encompasses 18 downstream tasks alongside 56 datasets. Beyond the common benchmarks such as MMLU (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021), our framework also extends the support for probing LLMs' advanced

* Co-leading the project.
† Equal Contribution. Ordered by name.
✉ Corresponding author.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 388–399
August 11-16, 2024 ©2024 Association for Computational Linguistics

capabilities: human alignment, hallucination detection, instruction following, *etc*. Furthermore, LLMBox integrates a variety of publicly available LLMs and commercial APIs, offering a convenient platform for holistic evaluation.

• *More practical considerations.* To be user-friendly, LLMBox is designed to provide an easy-to-use pipeline, enabling users to quickly start with minimal commands. We introduce a *GPU calculator* to help users determine the minimum GPU resources necessary for training. To be efficient, we propose a novel *prefix caching* strategy for inference and a *packing* strategy for training. Remarkably, given the LLaMA (7B) model, our library can perform inference on the entire MMLU benchmark within six minutes on a single A800 GPU and completes instruction tuning with 52K instances on eight A800 GPUs in ten minutes.

An additional feature is that LLMBox is closely aligned with our prior survey paper on LLMs (Zhao et al., 2023). This is particularly useful for beginners, enabling the learning of basic knowledge and practice of LLMs through integrating the survey paper and the associated library.

In what follows, we will first introduce the training framework of our library in Section 2, then detail the utilization and evaluation parts in Section 3, and showcase how to use our library in Section 4. After that, we will conduct the experiments to verify the reliability of our LLMBox in Section 5, and conclude the paper in Section 6.

2 Training

The training process is a crucial step for the development of LLMs. However, it typically needs massive detailed designs considering both efficiency and effectiveness, and also often faces intractable problems when adapting into new domains or meeting special needs. To facilitate easy training of LLMs, we integrate various training methods and resources in our library, to unify and simplify their usage. Besides, we provide suggestions for GPU usage tailored to different training requirements.

2.1 LLM Training

In our LLMBox, we develop a unified architecture to encapsulate important training methods in developing LLMs, and implement efficient training strategies to support training on limited computing resource. The overall framework of LLMBox is illustrated in Figure 1.

Key Training Methods. In our LLMBox, we integrate massive functionalities to support the following four training processes:

• *Pre-training.* Our LLMBox supports pre-training LLMs from scratch or continually pre-training using corpora in specific languages or specialized domains. For continually pre-training, LLMBox supports expanding the vocabulary to facilitate the adaptation of LLMs to new fields.

• *Instruction tuning.* LLMBox integrates ten commonly-used datasets for supporting instruction-tuning, covering NLP task (*e.g.,* FLAN v2 (Chung et al., 2022)), daily chat (*e.g.,* ShareGPT (Eccleston, 2023)), and synthetic datasets (*e.g.,* Alpaca-52K (Taori et al., 2023)). Additionally, we integrate three methods to synthesize or rewrite instructions, namely Self-Instruct (Wang et al., 2023a), Evol-Instruct (Xu et al., 2023), and topic diversifying (YuLan-Team, 2023). Based on the above datasets, we specially design unified dataset class, which can automatically preprocess these datasets into a unified format for training LLMs, and provide flexible interfaces for users to adjust the settings about the data (*e.g.,* data mixture proportion).

• *Human alignment.* To enhance the alignment of LLMs with human values, we incorporate both the widely-used RLHF method PPO (Schulman et al., 2017) and non-RL approach DPO (Rafailov et al., 2023). Besides, LLMBox also integrates several preference datasets, including HH-RLHF (Bai et al., 2022) and SHP (Ethayarajh et al., 2022).

Efficient Training Strategies. We also integrate several widely-used efficient training strategies or libraries, to support training LLMs with limited computing resources.

• *LoRA and QLoRA.* LLMBox integrates the lightweight module LoRA (Hu et al., 2022) to facilitate the different training methods of LLMs in resource-constrained environments. We also encapsulate QLoRA (Dettmers et al., 2023) in LLMBox, which performs quantization on LoRA for further reducing its used GPU memory.

• *DeepSpeed.* Our LLMBox library is based on the distributed training library DeepSpeed (Rasley et al., 2020), which includes a range of training optimization strategies for efficient training LLMs, including zero redundancy optimizer (ZeRO) (Rajbhandari et al., 2020), gradient checkpointing (Chen et al., 2016), FlashAttention (Dao et al., 2022), *etc.*

• *Packing.* We implement the packing strategy (Raffel et al., 2020; Touvron et al., 2023b)

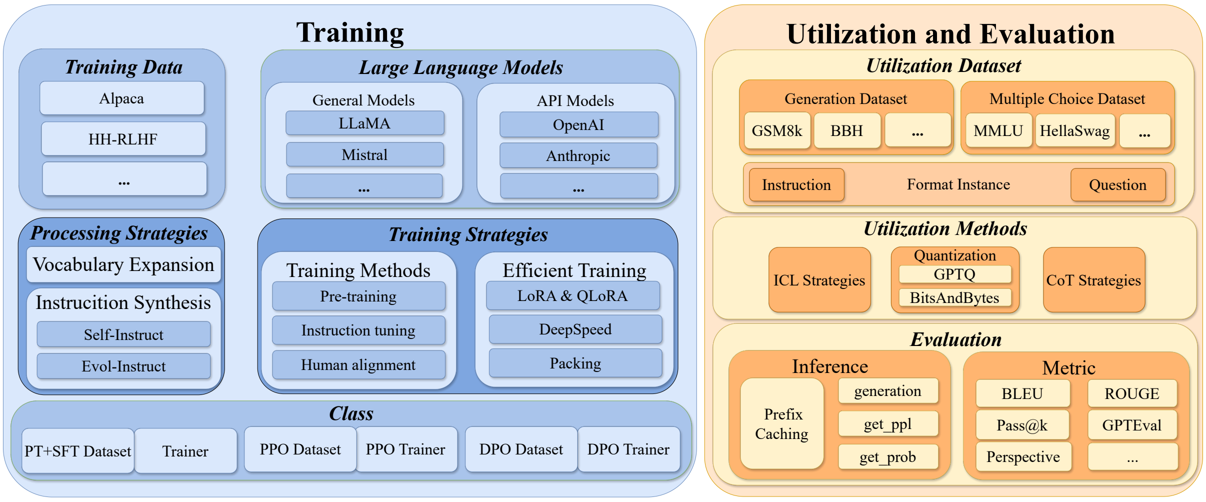

Figure 1: The overall framework of our LLMBox, supporting the training, utilization and evaluation of LLMs.

to enhance training efficiency. During pre-training, we concatenate all tokens into a long sentence and then split it to multiple sentences with the max length. For instruction-tuning, we concatenate all instructions as a long multi-turn conversation, and then break it into multiple conversations approaching to the maximum length constraint. Through minimizing paddings, we can optimize memory efficiency while maintaining model performance.

2.2 Training Suggestions

In practice, it is necessary for users to estimate the hardware requirements for training LLMs. Based on our LLMBox, we meticulously analyze GPU memory consumption throughout the model training process, by fully considering the impacts of parameters, gradients, optimizer states, and activation value (Rajbhandari et al., 2020; Ren et al., 2021; Korthikanti et al., 2023). We further introduce a “GPU memory calculator” to aid users in determining the minimal GPU requirements across LLMs with different parameter scales.

By merging the above strategies to reach efficiency[1], the memory consumption of each GPU can be roughly estimated by the equation:

$$\frac{16p}{n} + (12 + 2l)bsh + 12bsv, \tag{1}$$

where p represents the total number of parameters, and n, l, b, s, h, v stand for the number of GPUs, number of layers, batch size, sequence length, hidden size, and vocabulary size, respectively. Taking the training of LLaMA-2 (7B) ($l = 32, s = 4096, h = 4096, v = 32000$) as an example, we employ two A100 (80G) GPUs ($n = 2$) with a batch size of $b = 8$. By using Eq. 1 with the above configuration, we can estimate an approximate GPU memory usage of 71.42GB per unit. As shown in Table 1, we extrapolate the minimum GPU requirements using Eq. 1 for different model sizes across varying training settings, to help users for selecting proper GPU resources. For other special training settings, we invite users to utilize the calculator available on our library[2].

	DDP	**ZeRO-3**	**LoRA**	**QLoRA**
1.3B	1 A100 1 A6000	1 A100 1 A6000	1 A100 1 A6000	1 A100 1 A6000
2.7B	1 A100 N/A	1 A100 2 A6000	1 A100 1 A6000	1 A100 1 A6000
6.7B	N/A N/A	2 A100 3 A6000	1 A100 1 A6000	1 A100 1 A6000
13B	N/A N/A	3 A100 5 A6000	1 A100 1 A6000	1 A100 1 A6000
30B	N/A N/A	8 A100 12 A6000	1 A100 2 A6000	1 A100 1 A6000
70B	N/A N/A	16 A100 26 A6000	2 A100 4 A6000	1 A100 2 A6000

Table 1: Minimum GPU requirements for A100 (80G) and A6000 (48G) when training models with different sizes under four situations. N/A denotes DDP cannot be applied for such large models.

[1]For the training settings, we utilize data parallelism, ZeRO-3, gradient checkpointing, and FlashAttention.

[2]https://github.com/RUCAIBox/LLMBox/blob/main/training/gpu_calculator.py

3 Utilization and Evaluation

After training, we can develop suitable prompting strategies to use LLMs and assess their effectiveness. Users can reuse existing models, APIs or the models trained by LLMBox. The framework of our utilization pipeline is depicted in Figure 1.

3.1 Utilization Methods

We include quantization deployment strategies for using LLMs alongside two prompting methods: in-context learning (ICL) and chain-of-thought (CoT).

• *Quantization.* To enhance memory efficiency during inference, LLMBox incorporates two quantization techniques: bitsandbytes (Dettmers et al., 2022) and GPTQ (Frantar et al., 2023). Both methods facilitate 8-bit and 4-bit quantization and GPTQ additionally supports 3-bit quantization.

• *In-context learning.* We design a unified dataset class to organize diverse examples for few-shot learning. Furthermore, we implement three advanced ICL strategies, including KATE for example selection (Liu et al., 2022), GlobalE for example order arrange (Lu et al., 2022), and APE for instruction designing (Zhou et al., 2023c).

• *Chain-of-thought.* Moreover, LLMBox incorporates several CoT prompting methods, such as program-aided (PAL) CoT (Gao et al., 2023b) and least-to-most CoT (Zhou et al., 2023a). We develop a flexible framework to facilitate self-consistency (Wang et al., 2023a) and repeated sampling (Nijkamp et al., 2023), which are beneficial for tasks involving mathematics and coding.

3.2 Evaluation Methods

In LLMBox, we implement the evaluation of LLM performance through three distinct mechanisms:

• *Free-form generation:* This is the basic evaluation method for generative LLMs and is applicable across all tasks. Models are required to generate responses to queries in an auto-regressive manner. We integrate common decoding strategies, including greedy search, temperature sampling, top-p sampling, repetition penalties, *etc*.

• *Completion perplexity:* This method is widely adopted for assessing multi-choice tasks in base LLMs. It involves comparing the perplexity (PPL) of each completion conditioned on the context and choose the one with the lowest average PPL. Additionally, we incorporate the use of normalized PPL as introduced in GPT-3 (Brown et al., 2020).

• *Option probability:* Similar to the multi-choice formats in human examination, we feed a context with all the options to LLMs and require them to select the option letter (*e.g.,* A). This approach is commonly utilized in chat-based models.

Significantly, we introduce *prefix caching* mechanism that caches the hidden states of common prefix texts to speed up the inference process. This strategy is organized at two levels: (1) we store the states of few-shot examples and compute them just once for all instances, *e.g.,* 5-shot examples in MMLU (Hendrycks et al., 2021) and 8-shot examples in GSM8K (Cobbe et al., 2021); (2) we cache the states of identical contexts of different options when calculating completion perplexity. The effectiveness of this method is verified in Section 5.2.

3.3 Supported Models

We integrate a variety of LLMs to keep pace with the swift advancements in this field. Given that LLMBox is based on the Transformers library (Wolf et al., 2020), it is compatible with a vast majority of publicly available models. We list some included models as follows:

• *General models:* LLaMA (Touvron et al., 2023a) and Mistral (Jiang et al., 2023);

• *Chinese models:* Qwen (Bai et al., 2023) and Baichuan (Yang et al., 2023);

• *Multilingual models:* BLOOM (Le Scao et al., 2022);

• *Chat models:* LLaMA-2 Chat (Touvron et al., 2023b) and Vicuna (Chiang et al., 2023);

• *Code models:* CodeGen (Nijkamp et al., 2023) and StarCoder (Li et al., 2023c);

• *Mathematical models:* Llemma (Azerbayev et al., 2024) and DeepSeekMath (Shao et al., 2024).

We also incorporate commercial APIs including OpenAI[3] and Anthropic Claude[4].

3.4 Supported Tasks

Currently, LLMBox integrates 18 diverse tasks and corresponding 56 datasets with hundreds of subsets. The broad range of supported datasets within LLMBox enables to evaluate various models. For instance, users can employ English benchmarks, language modeling, and knowledge reasoning datasets for evaluating foundational pre-trained LLMs. In the case of chat-based models, users can utilize datasets focused on open-ended dialogue, human alignment, and instruction following. We list some included tasks and datasets as follows:

[3] https://openai.com/
[4] https://www.anthropic.com/

• *English benchmarks:* MMLU (Hendrycks et al., 2021) and BBH (Srivastava et al., 2023);

• *Chinese benchmarks:* CMMLU (Li et al., 2023a) and C-Eval (Huang et al., 2023);

• *Multilingual benchmarks:* TyDi QA (Clark et al., 2020) and MGSM (Shi et al., 2023);

• *Language modeling:* LAMBADA (Paperno et al., 2016);

• *Open-ended dialogue:* MT-Bench (Zheng et al., 2023) and AlpacaEval (Li et al., 2023d);

• *Machine translation:* general translation task in WMT[5] of every year and Flores-200 (Costa-jussà et al., 2022); 8

• *Text summarization:* CNN/Daily Mail (See et al., 2017) and XSum (Narayan et al., 2018);

• *Code synthesis:* HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021);

• *Closed-book question answering:* Natural Questions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017);

• *Reading comprehension:* SQuAD 2.0 (Rajpurkar et al., 2018) and RACE (Lai et al., 2017);

• *Knowledge reasoning:* HellaSwag (Zellers et al., 2019) and ARC (Clark et al., 2018);

• *Symbolic reasoning:* Tables of Penguins (Herzig et al., 2020) and Colored Objects (Srivastava et al., 2023);

• *Mathematical reasoning:* GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021);

• *Human Alignment:* TruthfulQA (Lin et al., 2022) and CrowS Pairs (Nangia et al., 2020);

• *Hallucination detection:* HaluEval (Li et al., 2023b);

• *Instruction following:* IFEval (Zhou et al., 2023b);

• *Environment Interaction:* ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022);

• *Tool Manipulation:* Gorilla (Patil et al., 2023).

4 Library Usage

In this section, we present the application of our library across four distinct research scenarios, illustrated through example code snippets.

Continually Pre-Training Language-Specific Models. As introduced in Section 2, we facilitate the continual pre-training of existing English-based LLMs for quick acquisition of new languages. Figure 2 (a) illustrates the process of tuning a Chinese LLM from LLaMA-2. Users are required only to prepare Chinese plain texts, such as those from Wikipedia, into a file named `chinese.txt`. Subsequently, LLMBox integrates new Chinese tokens into the vocabulary and trains the model.

[5]https://www2.statmt.org/

• *(a) Continually pre-training Chinese LLM:*

```
python merge_tokenizer.py --input chinese.txt
torchrun --nproc_per_node=8 train.py \
    --model Llama-2-7b --dataset chinese.txt
```

• *(b) Training medical LLM:*

```
torchrun --nproc_per_node=8 train.py \
    --model Llama-2-7b \
    --dataset_ratio 0.3 0.5 0.2 \
    --dataset pubmed.txt medmcqa.json sharegpt.json
```

• *(c) Evaluating davinci-002 on HellaSwag:*

```
python inference.py -m davinci-002 -d hellaswag
```

• *(d) Evaluating Gemma on MMLU:*

```
python inference.py -m gemma-7b -d mmlu -shots 5
```

• *(e) Evaluating Phi-2 on GSM8k using self-consistency and 4-bit quantization:*

```
python inference.py -m microsoft/phi-2 -d gsm8k \
    -shots 8 --sample_num 100 --load_in_4bit
```

• *(f) Designing prompting methods for a new dataset:*

```
def NewDataset(GenerationDataset):
    def load_dataset(self):
        self.exam_data = load(self.dataset, "exam")
        self.eval_data = load(self.dataset, "eval")
    def format_instance(self, instance):
        src, tgt = func(instance, self.exam_data)
        return dict(source=src, target=tgt)
    def reference(self):
        return [i["answer"] for i in self.eval_data]
```

Figure 2: Usage examples of our LLMBox library on six representative tasks.

Adapting LLMs to Specialized Domains. LLMBox facilitates the adaptation of LLMs to various specialized domains via instruction tuning, covering domains such as medicine, law, and finance. We present a script in Figure 2 (b) to train a medical LLM. We implement a convenient dataset mixture approach to sample instances from raw medical texts, medical instruction data, and general conversation data. This enables users to adjust the proportion to make a balance between medical knowledge, medical tasks, and conversational skills, thereby crafting an effective medical assistant.

Comprehensively Evaluating LLMs. We cover a broad range of tasks and various models within LLMBox to implement comprehensive evaluation. As illustrated in Figure 2 (c), (d), and (e), we present three exemplary command lines. Users are only required to designate the model and dataset names via the `-m` and `-d` options to achieve an efficient and accurate assessment of model performance. Furthermore, LLMBox supports multiple utilization methods, such as in-context learning (`-shots`), self-consistency (`--sample_num`), and quantitation (`--load_in_4bit`).

LLaMA-2		MMLU	BBH	HumanEval	NQs	HellaSwag	ARC-C	WinoGrande	BoolQ	GSM8K
7B	Paper	45.3	32.6	12.8	25.7	77.2	45.9	69.2	77.4	14.6
	LLMBox	46.5	33.2	13.6	25.5	75.6	49.6	69.6	78.5	14.6
70B	Paper	68.9	51.2	29.9	39.5	85.3	57.4	80.2	85.0	56.8
	LLMBox	69.5	54.8	29.2	40.3	83.3	57.8	80.7	85.6	56.6

Table 2: The results of different tasks on LLaMA-2 (7B) and (70B).

Proportion FLAN / Alpaca	MMLU	Alpaca-Eval
100 / 0	50.6	15.0
50 / 50	50.5	44.4
0 / 100	47.5	47.2
LLaMA-2 (7B)	46.5	23.0

Table 3: The performance of base LLaMA-2 (7B) and instruction tuned results using different data mixture.

Models	HellaSwag	MMLU	GSM8K
GPT-NeoX (20B)	71.4	26.4	7.1
OPT (66B)	73.5	27.3	2.2
BLOOM (7.1B)	61.1	26.0	4.2
LLaMA-2 (70B)	83.4	69.5	56.7
Pythia (12B)	67.2	25.1	4.6
MPT (30B)	79.8	45.4	21.5
Phi-2 (2.7B)	73.1	57.7	55.5
Mistral (7B)	80.2	63.8	43.6
Falcon (40B)	82.5	56.4	27.1
Gemma (7B)	79.2	65.3	52.3

Table 4: The results of different English LLMs using our developed LLMBox.

Designing Novel Prompting Methods. Since the implementation of each dataset in LLMBox is unified, it offers the flexibility to add new datasets or design various prompting methods without affecting other modules. Figure 2 (f) overviews the design of our `Dataset` class. When adding a new dataset, users are only required to implement three functions: `load_dataset` to load evaluation and example datasets; `format_instance` to format each instance with instruction or few-shot examples; and `reference` to define the ground truth. In the core function `format_instance`, users can develop innovative prompting methods tailored for each evaluation instance using example datasets.

5 Experiment

In the section, we conduct extensive experiments to verify the effectiveness and efficiency.

5.1 Effectiveness Evaluation

The essential attribute of an open-source library is its ability to reproduce results effectively. To confirm this, we choose several representative training and utilization scenarios for testing the outcomes derived from LLMBox.

Training results. We train LLaMA-2 (Touvron et al., 2023b) with the mixture of instruction tuning data FLAN (Chung et al., 2022) and Alpaca-52K (Taori et al., 2023) and evaluate its performance. We adjust the proportions of these datasets and assess the impact on performance using the MMLU benchmark (Hendrycks et al., 2021) and the chat-oriented AlpacaEval (Dubois et al., 2023). The experiments are conducted with a batch size of 128 and a constant learning rate of 1×10^{-5}. The model undergoes training for a total of 1200 steps, and we report the peak performance observed on the evaluation datasets. The results in Table 3 indicate that FLAN improves the model's performance on NLP tasks, whereas Alpaca-52K significantly enhances its performance in daily chat. Moreover, when mixing both instruction datasets yields a balanced improvement across both tasks, aligning with findings from prior research (Wang et al., 2023b).

Utilization results. Firstly, we examine the performance of LLaMA-2 (Touvron et al., 2023b) across various supported tasks. We totally evaluate nine tasks, including MMLU (5-shot, accuracy) (Hendrycks et al., 2021), BBH (3-shot, accuracy) (Srivastava et al., 2023), HumanEval (0-shot, pass1) (Chen et al., 2021), Natural Questions (NQs, 5-shot, EM) (Kwiatkowski et al., 2019), HellaSwag (0-shot, accuracy) (Zellers et al., 2019), ARC-Chanllge (ARC-C, 0-shot, accuracy) (Clark et al., 2018), WinoGrande (0-shot, accuracy) (Sakaguchi et al., 2021), BoolQ (0-shot, accuracy) (Clark et al., 2019), and GSM8K (8-shot, accuracy) (Cobbe et al., 2021). The results in Table 2 demonstrates that our LLMBox library faithfully reproduces the results reported in their original papers. Furthermore, we verify the performance of LLMBox across a variety of models. We utilize HellaSwag, MMLU, and GSM8K to evaluate the per-

Models	HellaSwag	C-Eval	GSM8K
ChatGLM-3 (6B)	63.6	53.0	48.5
C-LLaMA-2 (13B)	76.4	41.8	18.6
InternLM-2 (20B)	82.5	69.5	74.4
Baichuan-2 (13B)	74.7	59.2	42.8
Qwen-1.5 (72B)	83.8	83.5	78.2
Aquila-2 (34B)	78.8	98.6	2.0
Deepseek (67B)	83.4	65.9	64.1
Yi (34B)	83.2	81.4	5.4

Table 5: The experimental results of different Chinese LLMs and APIs using our developed LLMBox. C-LLaMA-2 is short for Chinese-LLaMA-2.

formance of ten English LLMs, including GPT-NeoX (Black et al., 2022), OPT (Zhang et al., 2022), BLOOM (Le Scao et al., 2022), LLaMA-2 (Touvron et al., 2023b), Pythia (Biderman et al., 2023), MPT (Team, 2023b), Phi-2 (Javaheripi et al., 2023), Mistral (Jiang et al., 2023), Falcon (Almazrouei et al., 2023), Gemma (Google, 2024). We employ HellaSwag, C-Eval (Huang et al., 2023), and GSM8K to evaluate the performance of eight Chinese LLMs, including ChatGLM3 (Zeng et al., 2022), Chinese-LLaMA-2 (Cui et al., 2023), InternLM-2 (Team, 2023a), Baichuan-2 (Baichuan, 2023), Qwen-1.5 (Bai et al., 2023), Aquila-2 (BAAI, 2023), Deepseek (DeepSeek-AI, 2024), Yi (Young et al., 2024). The results of these evaluations are detailed in Tables 4 and 5. We can find that our LLMBox is also compatible with various English and Chinese LLMs.

5.2 Efficiency Evaluation

The implementation efficiency is also a key factor to deploy LLMs. In addition to accurately reproducing results, we have optimized LLMBox for training and utilization efficiency. From the results in Table 6, it is evident that our prefix caching approach substantially decreases the inference time compared to the traditional Transformers implementation. As the number of examples increases (from 5-shot setting in MMLU to 8-shot setting in GSM8K), the efficiency gains from our method become increasingly pronounced. Remarkably, with the application of our prefix caching technique to the MMLU benchmark, LLMBox requires merely six minutes to process over ten thousand instances, achieving a 60% reduction in processing time compared to the vLLM toolkit. In the future, we aim to incorporate this prefix caching strategy into vLLM to further enhance the inference efficiency.

Strategies	HellaSwag	MMLU	GSM8K
Transformers	5.5	18.5	130.5
Transformers+PC	6.1	6.0	23.3
vLLM	6.6	14.9	3.6

Table 6: The execution time of different implementation methods on LLaMA-2 (7B) using one A800 (80G) GPU. PC is short for the proposed novel prefix caching mechanism in our developed LLMBox.

6 Conclusion

This paper presented **LLMBox**, a comprehensive library for conducting research on training, utilizing, and evaluating large language models. For training, we designed a unified data interface to support the implementation of various training strategies. For utilization and evaluation, we implemented typical approaches to use LLMs (including quantization, ICL, and CoT prompting), covered 18 tasks and 56 datasets, and included a number of popular open-sourced LLMs and closed-source APIs. We further conducted extensive experiments to verify the effectiveness and efficiency of LLMBox. Our library provides a unified framework to compare, reproduce, and develop LLMs and supporting methods for academic purposes, which would be of important value to promote the research on LLMs.

Acknowledgement

This work was partially supported by National Natural Science Foundation of China under Grant No. 62222215, Beijing Natural Science Foundation under Grant No.4222027 and L233008. Xin Zhao is the corresponding author.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. Falcon-40B: an open large language model with state-of-the-art performance.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

BAAI. 2023. Aquila2.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Baichuan. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 2397–2430. PMLR.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pages 16344–16359. Curran Associates, Inc.

DeepSeek-AI. 2024. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. In *International Conference on Learning Representations*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.

Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy S Liang, and Tatsunori B Hashimoto. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 30039–30069. Curran Associates, Inc.

Dom Eccleston. 2023. Sharegpt. `https://sharegpt.com/`.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023a. A framework for few-shot language model evaluation.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023b. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.

Team Google. 2024. Gemma.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

hiyouga. 2023. Llama factory. `https://github.com/hiyouga/LLaMA-Factory`.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, jiayi lei, Yao Fu, Maosong Sun, and Junxian He. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pages 62991–63010. Curran Associates, Inc.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Reducing activation recomputation in large transformer models. *Proceedings of Machine Learning and Systems*, 5.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023a. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023b. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

Raymond Li, Loubna Ben allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia LI, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Joel Lamy-Poirier, Joao Monteiro, Nicolas Gontier, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Ben Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason T Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Urvashi Bhattacharyya, Wenhao Yu, Sasha Luccioni, Paulo Villegas, Fedor Zhdanov, Tony Lee, Nadav Timor, Jennifer Ding, Claire S Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro Von Werra, and Harm de Vries. 2023c. Starcoder: may the source be with you! *Transactions on Machine Learning Research*. Reproducibility Certification.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023d. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2023. Codegen: An open large language model for code with multi-turn program synthesis. In *The Eleventh International Conference on Learning Representations*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2023. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. 2021. {Zero-offload}: Democratizing {billion-scale} model training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pages 551–564.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. {ALFW}orld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

InternLM Team. 2023a. Internlm: A multilingual language model with progressively enhanced capabilities. `https://github.com/InternLM/InternLM`.

MosaicML NLP Team. 2023b. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-05-05.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023b. How far can camels go? exploring the state of instruction tuning on open resources. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin

Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems*, volume 36, pages 69798–69818. Curran Associates, Inc.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2023. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757. Curran Associates, Inc.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

YuLan-Team. 2023. Yulan-chat: An open-source bilingual chatbot. `https://github.com/RUC-GSAI/YuLan-Chat`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023a. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023c. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.