A Comprehensive Overview of Large Language Models

Humza Naveed[1], Asad Ullah Khan[1,*], Shi Qiu[2,*], Muhammad Saqib[3,4,*], Saeed Anwar[5,6], Muhammad Usman[5,6], Naveed Akhtar[7], Nick Barnes[8], Ajmal Mian[9]

[1]University of Engineering and Technology (UET), Lahore, Pakistan
[2]The Chinese University of Hong Kong (CUHK), HKSAR, China
[3]University of Technology Sydney (UTS), Sydney, Australia
[4]Commonwealth Scientific and Industrial Research Organisation (CSIRO), Sydney, Australia
[5]King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia
[6]SDAIA-KFUPM Joint Research Center for Artificial Intelligence (JRCAI), Dhahran, Saudi Arabia
[7]The University of Melbourne (UoM), Melbourne, Australia
[8]Australian National University (ANU), Canberra, Australia
[9]The University of Western Australia (UWA), Perth, Australia

arXiv:2307.06435v6 [cs.CL] 23 Nov 2023

***Abstract*—**
Large Language Models (LLMs) have recently demonstrated remarkable capabilities in natural language processing tasks and beyond. This success of LLMs has led to a large influx of research contributions in this direction. These works encompass diverse topics such as architectural innovations, better training strategies, context length improvements, fine-tuning, multi-modal LLMs, robotics, datasets, benchmarking, efficiency, and more. With the rapid development of techniques and regular breakthroughs in LLM research, it has become considerably challenging to perceive the bigger picture of the advances in this direction. Considering the rapidly emerging plethora of literature on LLMs, it is imperative that the research community is able to benefit from a concise yet comprehensive overview of the recent developments in this field. This article provides an overview of the existing literature on a broad range of LLM-related concepts. Our self-contained comprehensive overview of LLMs discusses relevant background concepts along with covering the advanced topics at the frontier of research in LLMs. This review article is intended to not only provide a systematic survey but also a quick comprehensive reference for the researchers and practitioners to draw insights from extensive informative summaries of the existing works to advance the LLM research.

***Index Terms*—**
Large Language Models, LLMs, chatGPT, Augmented LLMs, Multimodal LLMs, LLM training, LLM Benchmarking

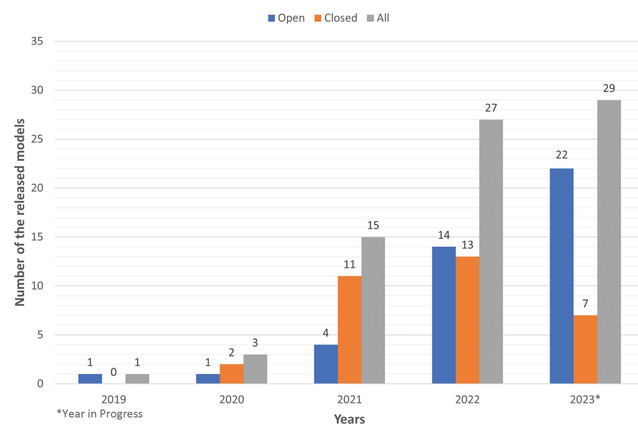

Fig. 1: The trends in the number of LLM models introduced over the years.

I. Introduction

Language plays a fundamental role in facilitating communication and self-expression for humans, and their interaction with machines. The need for generalized models stems from the growing demand for machines to handle complex language tasks, including translation, summarization, information retrieval, conversational interactions, etc. Recently, significant breakthroughs have been witnessed in language models, primarily attributed to transformers [1], increased computational capabilities, and the availability of large-scale training data. These developments have brought about a revolutionary transformation by enabling the creation of LLMs that can approximate human-level performance on various tasks [2], [3]. Large Language Models (LLMs) have emerged as cutting-edge artificial intelligence systems that can process and generate text with coherent communication [4], and generalize to multiple tasks [5], [6].
The historical progress in natural language processing (NLP) evolved from statistical to neural language modeling and then from pre-trained language models (PLMs) to LLMs. While conventional language modeling (LM) trains task-specific

* is for equal contribution
Contact e-mail: humza_naveed@yahoo.com
Email: humza_naveed@yahoo.com, aukhanee@gmail.com, shiqiu@cse.cuhk.edu.hk, muhammad.saqib@data61.csiro.au, saeed.anwar@kfupm.edu.sa, muhammad.usman@kfupm.edu.sa, naveed.akhtar1@unimelb.edu.au, nick.barnes@anu.edu.au, ajmal.mian@uwa.edu.au
Repo: https://github.com/humza909/LLM_Survey.git

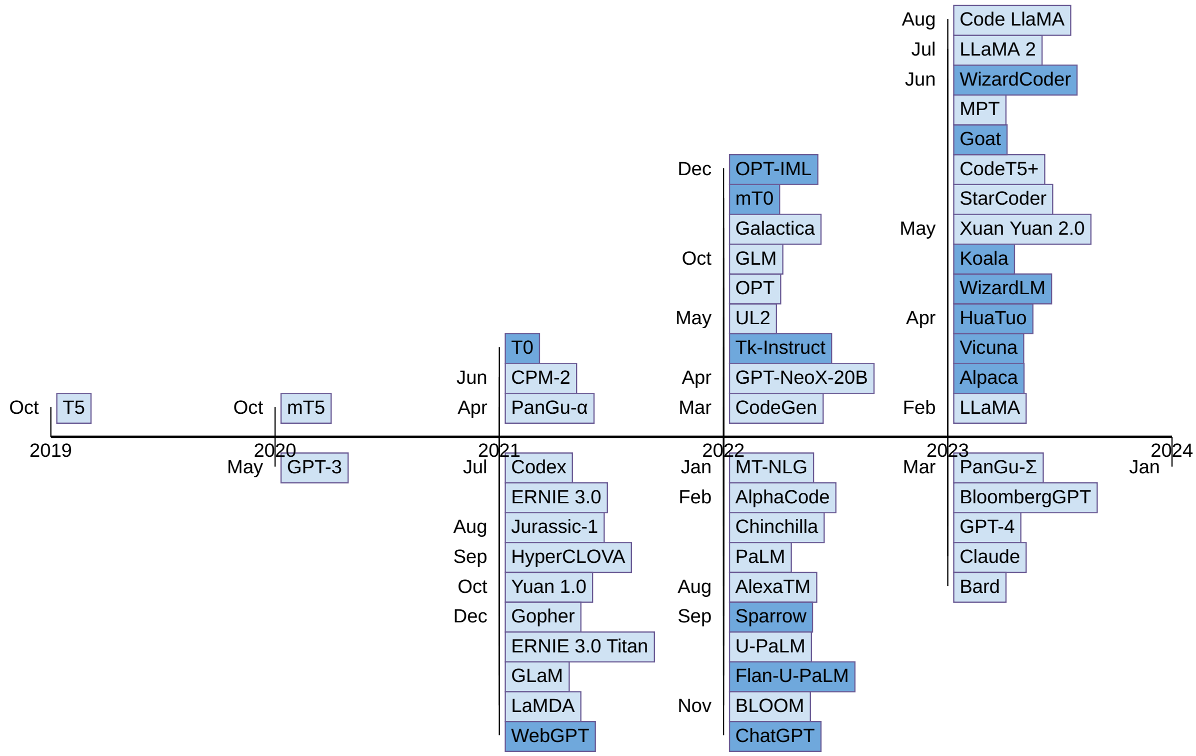

Fig. 2: Chronological display of LLM releases: light blue rectangles represent 'pre-trained' models, while dark rectangles correspond to 'instruction-tuned' models. Models on the upper half signify open-source availability, whereas those on the bottom half are closed-source. The chart illustrates the increasing trend towards instruction-tuned models and open-source models, highlighting the evolving landscape and trends in natural language processing research.

models in supervised settings, PLMs are trained in a self-supervised setting on a large corpus of text [7], [8], [9] with the aim to learn generic representation shareable among various NLP tasks. After fine-tuning for downstream tasks, PLMs surpass the performance gains of traditional language modeling (LM). The larger PLMs bring more performance gains, which has led to the transitioning of PLMs to LLMs by significantly increasing model parameters (tens to hundreds of billions) [10] and training dataset (many GBs and TBs) [10], [11]. Following this development, numerous LLMs have been proposed in the literature [10], [11], [12], [6], [13], [14], [15]. An increasing trend in the number of released LLMs and names of a few significant LLMs proposed over the years are shown in Fig 1 and Fig 2, respectively.

The early work on LLMs, such as T5 [10] and mT5 [11] employed transfer learning until GPT-3 [6] showing LLMs are zero-shot transferable to downstream tasks without fine-tuning. LLMs accurately respond to task queries when prompted with task descriptions and examples. However, pre-trained LLMs fail to follow user intent and perform worse in zero-shot settings than in few-shot. Fine-tuning them with task instructions data [16], [17], [18], [19] and aligning with human preferences [20], [21] enhances generalization to unseen tasks, improving zero-shot performance significantly and reducing misaligned behavior.

Additional to better generalization and domain adaptation, LLMs appear to have emergent abilities, such as reasoning, planning, decision-making, in-context learning, answering in zero-shot settings, etc. These abilities are known to be acquired by them due to their gigantic scale even when the pre-trained LLMs are not trained specifically to possess these attributes [22], [23], [24]. Such abilities have led LLMs widely adopted in diverse settings including, multi-modal, robotics, tool manipulation, question answering, autonomous agents, etc. Various improvements have also been suggested in these areas either by task-specific training [25], [26], [27], [28], [29], [30], [31] or better prompting [32].

The LLMs abilities to solve diverse tasks with human-level performance come at a cost of slow training and inference, extensive hardware requirements, and higher running costs. Such requirements have limited their adoption and opened up opportunities to devise better architectures [15], [33], [34], [35] and training strategies [36], [37], [21], [38], [39], [40], [41]. Parameter efficient tuning [38], [41], [40], pruning, quantization, knowledge distillation, and context length interpolation [42], [43], [44], [45] among others are some of the methods widely studied for efficient LLM utilization.

Due to the success of LLMs on a wide variety of tasks, the research literature has recently experienced a large influx of LLM-related contributions. Researchers have organized the

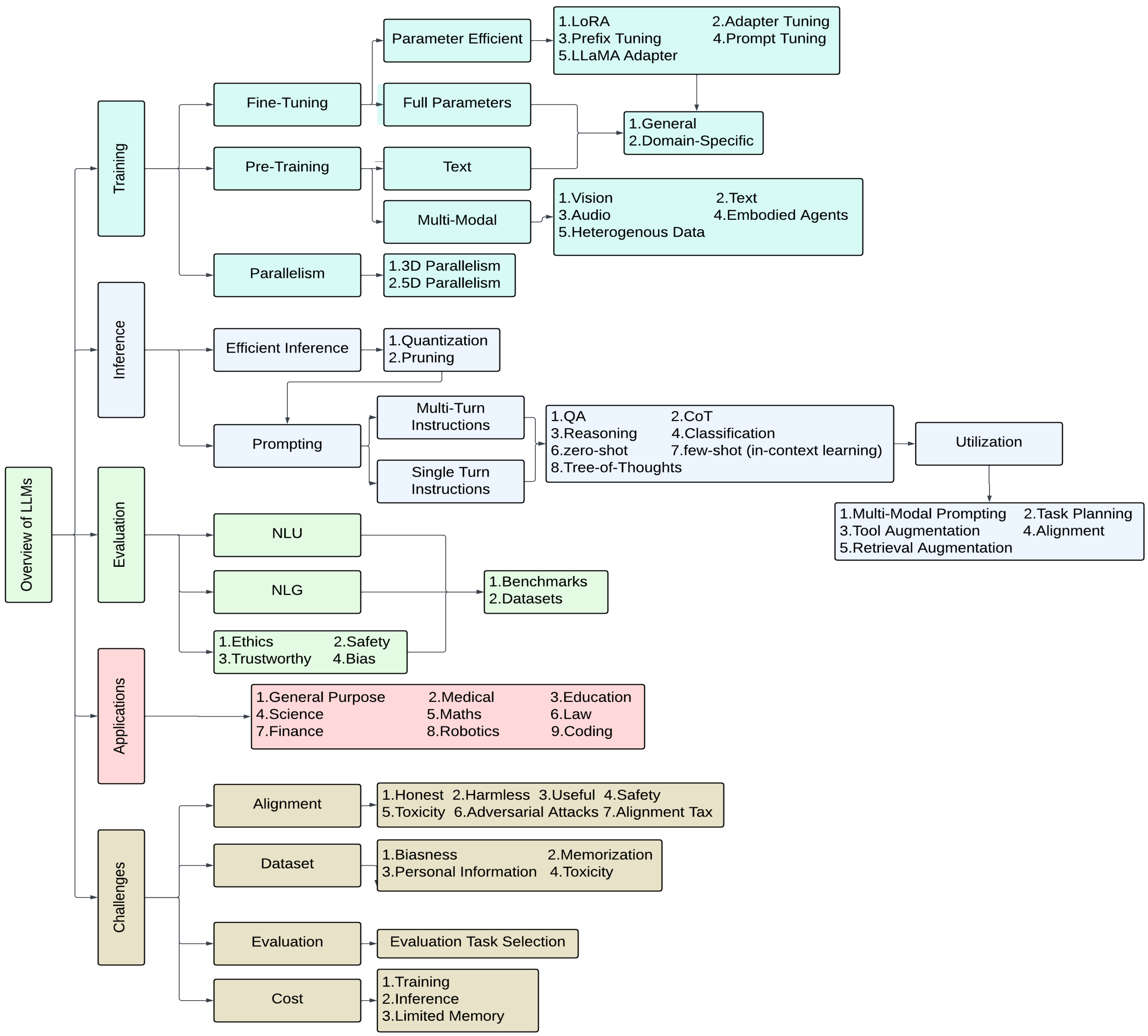

Fig. 3: A broader overview of LLMs, dividing LLMs into five branches: 1. Training 2. Inference 3. Evaluation 4. Applications 5. Challenges

LLMs literature in surveys [46], [47], [48], [49], and topic-specific surveys in [50], [51], [52], [53], [54]. In contrast to these surveys, our contribution focuses on providing a comprehensive yet concise overview of the general direction of LLM research. This article summarizes architectural and training details of pre-trained LLMs and delves deeper into the details of concepts like fine-tuning, multi-modal LLMs, robotics, augmented LLMs, datasets, evaluation, and others to provide a self-contained comprehensive overview. Our key contributions are summarized as follows.

- We present a survey on the developments in LLM research with the specific aim of providing a concise yet comprehensive overview of the direction.
- We present extensive summaries of pre-trained models that include fine-grained details of architecture and training details.
- Besides paying special attention to the chronological order of LLMs throughout the article, we also summarize major findings of the popular contributions and provide detailed discussion on the key design and development aspects of LLMs to help practitioners to effectively leverage this technology.
- In this self-contained article, we cover a range of concepts to comprehend the general direction of LLMs comprehensively, including background, pre-training, fine-tuning, robotics, multi-modal LLMs, augmented LLMs, datasets, evaluation, etc.

We loosely follow the existing terminologies to ensure providing a more standardized outlook of this research direction. For instance, following [46], our survey discusses pre-trained

LLMs with 10B parameters or more. We refer the readers interested in smaller pre-trained models to [47], [48], [49]. The organization of this paper is as follows. Section II discusses the background of LLMs. Section III focuses on LLMs overview, architectures, training pipelines and strategies, and utilization in different aspects. Section IV presents the key findings derived from each LLM. Section V highlights the configuration and parameters that play a crucial role in the functioning of these models. Summary and discussions are presented in section VII. The LLM training and evaluation, datasets and benchmarks are discussed in section VI, followed by challenges and future directions and conclusion in sections VIII and IX, respectively.

II. Background

We provide the relevant background to understand the fundamentals related to LLMs in this section. Aligned with our objective of providing a comprehensive overview of this direction, this section offers a comprehensive yet concise outline of the basic concepts. We focus more on the intuitive aspects and refer the readers interested in details to the original works.

A. Tokenization

LLMs are trained on text to predict text, and similar to other natural language processing systems, they use tokenization [55] as the essential preprocessing step. It aims to parse the text into non-decomposing units called tokens. Tokens can be characters, subwords [56], symbols [57], or words, depending on the size and type of the model. Some of the commonly used tokenization schemes in LLMs are briefed here. Readers are encouraged to refer to [58] for a detailed survey.

1. WordPiece [59]: It was introduced in [59] as a novel text segmentation technique for Japanese and Korean languages to improve the language model for voice search systems. WordPiece selects tokens that increase the likelihood of an n-gram-based language model trained on the vocabulary composed of tokens.

2. BPE [57]: Byte Pair Encoding (BPE) has its origin in compression algorithms. It is an iterative process of generating tokens where pairs of adjacent *symbols* are replaced by a new symbol, and the occurrences of the most occurring symbols in the input text are merged.

3. UnigramLM [56]: In this tokenization, a simple unigram language model (LM) is trained using an initial vocabulary of *subword* units. The vocabulary is pruned iteratively by removing the lowest probability items from the list, which are the worst performing on the unigram LM.

B. Attention

Attention, particularly *selective attention*, has been widely studied under perception, psychophysics, and psychology. Selective attention can be conceived as "the programming by the O of which stimuli will be processed or encoded and in what order this will occur" [60]. While this definition has its roots in visual perception, it has uncanny similarities with the recently formulated *attention* [61], [62] (which stimuli will be processed) and *positional encoding* (in what order this will occur) [62] in LLMs. We discuss both in sections II-C and II-D, respectively.

C. Attention in LLMs

The attention mechanism computes a representation of the input sequences by relating different positions (*tokens*) of these sequences. There are various approaches to calculating and implementing attention, out of which some famous types are given below.

1. Self-Attention [62]: The self-attention is also known as intra-attention since all the queries, keys, and values come from the same block (encoder or decoder). The self-attention layer connects all the sequence positions with $O(1)$ space complexity which is highly desirable for learning long-range dependencies in the input.

2. Cross Attention: In encoder-decoder architectures, the outputs of the encoder blocks act as the queries to the intermediate representation of the decoder, which provides the keys and values to calculate a representation of the decoder conditioned on the encoder. This attention is called cross-attention.

3. Full Attention: The naive implementation of calculating self-attention is known as full attention.

4. Sparse Attention [63]: The self-attention has a time complexity of $O(n^2)$, which becomes prohibitive when scaling the LLMs to large context windows. An approximation to the self-attention was proposed in [63], which greatly enhanced the capacity of GPT series LLMs to process a greater number of input tokens in a reasonable time.

5. Flash Attention [64]: The bottleneck for calculating the attention using GPUs lies in the memory access rather than the computational speed. Flash Attention uses the classical input tiling approach to process the blocks of the input in GPU on-chip SRAM rather than doing IO for every token from the High Bandwith Memory (HBM). An extension of this approach to sparse attention follows the speed gains of the full attention implementation. This trick allows even greater context-length windows in the LLMs as compared to those LLMs with sparse attention.

D. Encoding Positions

The *attention* modules do not consider the order of processing by design. Transformer [62] introduced "positional encodings" to feed information about the position of the tokens in input sequences. Several variants of positional encoding have been proposed [65], [66]. Interestingly, a recent study [67] suggests that adding this information may not matter for the state-of-the-art decoder-only Transformers.

1. Absolute: This is the most straightforward approach to adding the sequence order information by assigning a unique identifier to each position of the sequence before passing it to the attention module.

2. Relative: To pass the information on the relative dependencies of different tokens appearing at different locations in the sequence, a relative positional encoding is calculated by some kind of learning. Two famous types of relative encodings are:

Alibi: *[65]* In this approach, a scalar bias is subtracted from the attention score calculated using two tokens which increases with the distance between the positions of the tokens. This learned approach effectively favors using recent tokens for attention.

RoPE: Keys, queries, and values are all vectors in the LLMs. RoPE [66] involves the rotation of the query and key representations at an angle proportional to their absolute positions of the tokens in the input sequence. This step results in a relative positional encoding scheme which decays with the distance between the tokens.

E. Activation Functions

The activation functions serve a crucial role in the curve-fitting abilities of the neural networks, as proved in [68]. The modern activation functions used in LLMs are different from the earlier squashing functions but are critical to the success of LLMs. We discuss these activation functions in this section.

1. ReLU [69]: Rectified linear unit (ReLU) is defined as

$$ReLU(x) = max(0, x) \tag{1}$$

2. GeLU [70]: Gaussian Error Linear Unit (GeLU) is the combination of ReLU, dropout [71] and zoneout [72]. It is the most widely used activation function in contemporary LLM literature.

3. GLU variants [73]: Gated Linear Unit [74] is a neural network layer that is an element-wise product ($\otimes$) of a linear transformation and a sigmoid transformed (σ) linear projection of the input given as

$$GLU(x, W, V, b, c) = (xW + b) \otimes \sigma(xV + c), \tag{2}$$

where X is the input of layer and l, W, b, V and c are learned parameters.

GLU was modified in [73] to evaluate the effect of different variations in the training and testing of transformers, resulting in better empirical results. Here are the different GLU variations introduced in [73] and used in LLMs.

$$\begin{aligned} ReGLU(x, W, V, b, c) &= max(0, xW + b)\otimes, \\ GEGLU(x, W, V, b, c) &= GELU(xW + b) \otimes (xV + c), \\ SwiGLU(x, W, V, b, c, \beta) &= Swish\beta(xW + b) \otimes (xV + c). \end{aligned}$$

F. Layer Normalization

Layer normalization leads to faster convergence and is a widely used component in transformers. In this section, we provide different normalization techniques widely used in LLM literature.

1. LayerNorm: Layer norm computes statistics over all the hidden units in a layer (l) as follows:

$$u^l = \frac{1}{n}\sum_i^n a_i^l \qquad \sigma^l = \sqrt{\frac{1}{n}\sum_i^n (a_i^l - u^l)^2}, \tag{3}$$

where n is the number of neurons in the layer l and a_i^l is the summed input of the i neuron in layer l. LayerNorm provides invariance to rescaling of the weights and re-centering of the distribution.

2. RMSNorm: [75] proposed that the invariance properties of LayerNorm are spurious, and we can achieve the same performance benefits as we get from LayerNorm by using a computationally efficient normalization technique that trades off re-centering invariance with speed. LayerNorm gives the normalized summed input to layer l as follows

$$\overline{a_i^l} = \frac{a_i^l - u^l}{\sigma} g_i^l \tag{4}$$

where g_i^l is the gain parameter. RMSNorm [75] modifies $\overline{a_i^l}$ as

$$\overline{a_i^l} = \frac{a_i^l}{\text{RMS}(\mathbf{a}^l)} g_i^l, \text{ where } \text{RMS}(\mathbf{a}^l) = \sqrt{\frac{1}{n}\sum_i^n (a_i^l)^2}. \tag{5}$$

3. Pre-Norm and Post-Norm: LLMs use transformer [62] architecture with some variations. The original implementation [62] used layer normalization after the residual connection, commonly called post-LN, concerning the order of *Multihead attention – Residual – LN*. There is another order of the normalization, referred to as pre-LN [76] due to the position of the normalization step before the self-attention layer as in *LN – Multihead attention – Residual*. Pre-LN is known to provide more stability in the training [77].

4. DeepNorm: While pre-LN has certain benefits over post-LN training, pre-LN training has an unwanted effect on the gradients [77]. The earlier layers have larger gradients than those at the bottom. DeepNorm [78] mitigates these adverse effects on the gradients. It is given as

$$\mathbf{x}^{l_f} = LN(\alpha \mathbf{x}^{l_p} + G^{l_p}(\mathbf{x}^{l_p}, \theta^{l_p}), \tag{6}$$

where α is a constant and θ^{l_p} represents the parameters of layer l_p. These parameters are scaled by another constant β. Both of these constants depend only on the architecture.

G. Distributed LLM Training

This section describes distributed LLM training approaches briefly. More details are available in [13], [37], [79], [80].

1. Data Parallelism: Data parallelism replicates the model on multiple devices where data in a batch gets divided across devices. At the end of each training iteration weights are synchronized across all devices.

2. Tensor Parallelism: Tensor parallelism shards a tensor computation across devices. It is also known as horizontal parallelism or intra-layer model parallelism.

3. Pipeline Parallelism: Pipeline parallelism shards model layers across different devices. This is also known as vertical parallelism.

4. Model Parallelism: A combination of tensor and pipeline parallelism is known as model parallelism.

5. 3D Parallelism: A combination of data, tensor, and model parallelism is known as 3D parallelism.

6. Optimizer Parallelism: Optimizer parallelism also known as zero redundancy optimizer [37] implements optimizer state partitioning, gradient partitioning, and parameter partitioning across devices to reduce memory consumption while keeping the communication costs as low as possible.

H. Libraries

Some commonly used libraries for LLMs training are: 1) Transformers [81], 2) DeepSpeed [36], 3) Megatron-LM [79], 4) JAX [82], 5) Colossal-AI [83], 6) BMTrain [80], 7) FastMoE [84], and frameworks are 1) MindSpore [85], 2) PyTorch [86], 3) Tensorflow [87], 4) MXNet [88].

I. Data PreProcessing

This section briefly summarizes data preprocessing techniques used in LLMs training.

1. Quality Filtering: For better results, training data quality is essential. Some approaches to filtering data are: 1) classifier-based and 2) heuristics-based. Classifier-based approaches train a classifier on high-quality data and predict the quality of text for filtering, whereas heuristics-based employ some rules for filtering like language, metrics, statistics, and keywords.

2. Data Deduplication: Duplicated data can affect model performance and increase data memorization; therefore, to train LLMs, data deduplication is one of the preprocessing steps. This can be performed at multiple levels, like sentences, documents, and datasets.

3. Privacy Reduction: Most of the training data for LLMs is collected through web sources. This data contains private information; therefore, many LLMs employ heuristics-based methods to filter information such as names, addresses, and phone numbers to avoid learning personal information.

J. Architectures

Here we discuss the variants of the transformer architectures at a higher level which arise due to the difference in the application of the attention and the connection of transformer blocks. An illustration of attention patterns of these architectures is shown in Figure 4.

1. Encoder Decoder: Transformers were originally designed as sequence transduction models and followed other prevalent model architectures for machine translation systems. They selected encoder-decoder architecture to train human language translation tasks. This architecture is adopted by [10], [89]. In this architectural scheme, an encoder encodes the input sequences to variable length context vectors, which are then passed to the decoder to maximize a joint objective of minimizing the gap between predicted token labels and the actual target token labels.

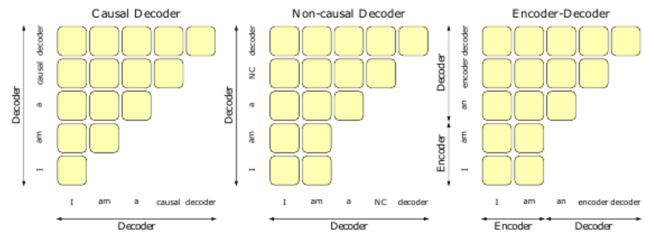

Fig. 4: An example of attention patterns in language models, image is taken from [91].

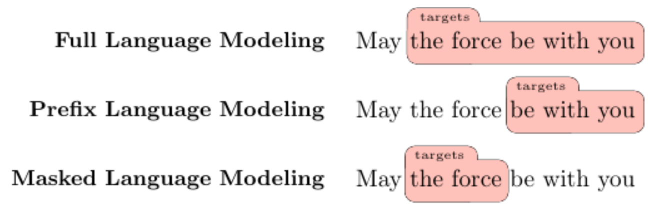

Fig. 5: An example of language model training objectives, image from [91].

2. Causal Decoder: The underlying objective of an LLM is to predict the next token based on the input sequence. While additional information from the encoder binds the prediction strongly to the context, it is found in practice that the LLMs can perform well in the absence of encoder [90], relying only on the decoder. Similar to the original encoder-decoder architecture's decoder block, this decoder restricts the flow of information backward, i.e., the predicted token t_k only depends on the tokens preceded by and up to t_{k-1}. This is the most widely used variant in the state-of-the-art LLMs.

3. Prefix Decoder: The causal masked attention is reasonable in the encoder-decoder architectures where the encoder can attend to all the tokens in the sentence from every position using self-attention. This means that the encoder can also attend to tokens t_{k+1} to t_n in addition to the tokens from t_1 to t_{k-1} while calculating the representation for t_k. But when we drop the encoder and only keep the decoder, we also lose this flexibility in attention. A variation in the decoder-only architectures is by changing the mask from strictly causal to fully visible on a portion of the input sequence, as shown in Figure 4. The Prefix decoder is also known as non-causal decoder architecture.

K. Pre-Training Objectives

This section describes LLMs pre-training objectives. For more details see the paper [91].

1. Full Language Modeling: An autoregressive language modeling objective where the model is asked to predict future tokens given the previous tokens, an example is shown in Figure 5.

2. Prefix Language Modeling: A non-causal training objective, where a prefix is chosen randomly and only remaining target tokens are used to calculate the loss. An example is shown in Figure 5.

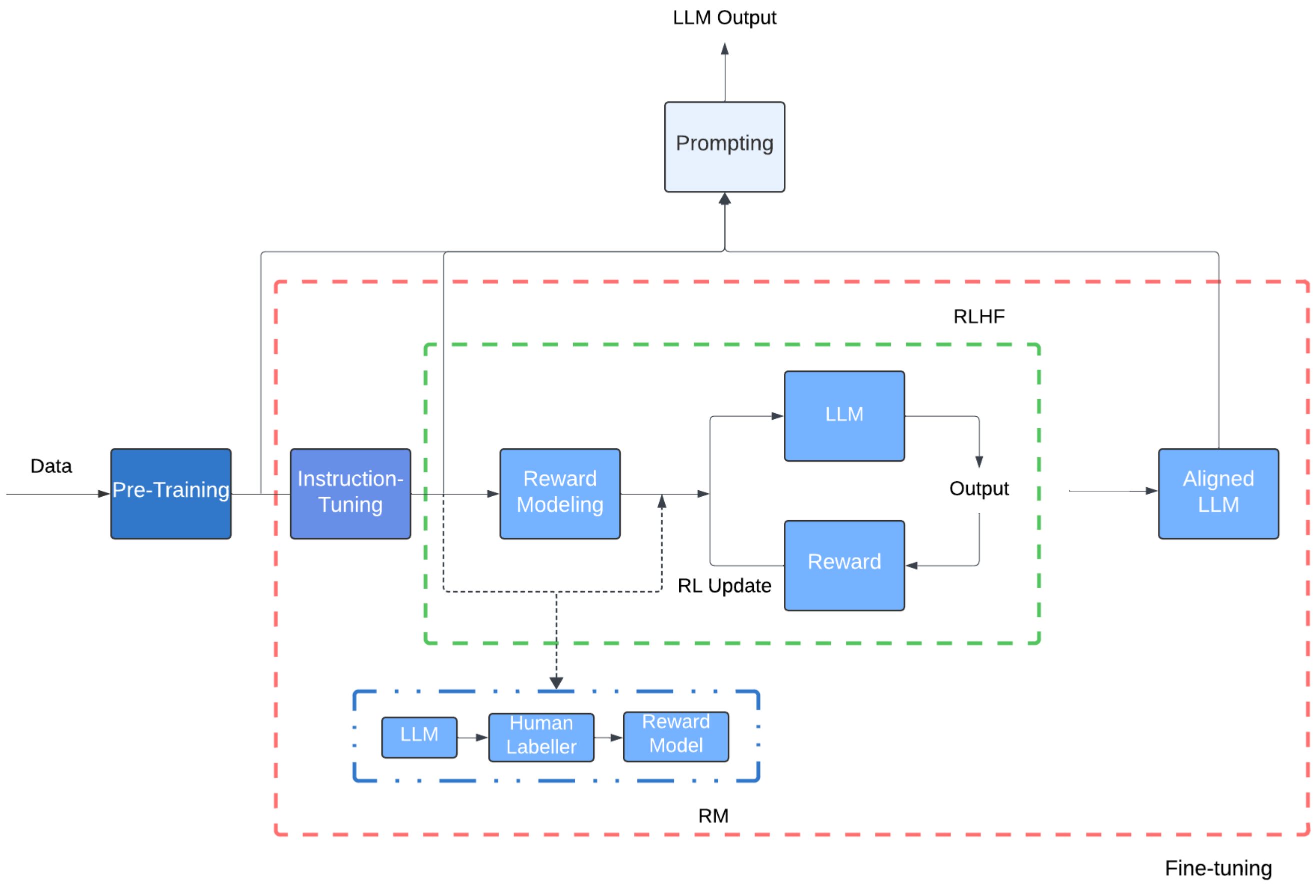

Fig. 6: A basic flow diagram depicting various stages of LLMs from pre-training to prompting/utilization. Prompting LLMs to generate responses is possible at different training stages like pre-training, instruction-tuning, or alignment tuning.

3. Masked Language Modeling: In this training objective, tokens or spans (a sequence of tokens) are masked randomly and the model is asked to predict masked tokens given the past and future context. An example is shown in Figure 5.

4. Unified Language Modeling: Unified language modeling [92] is a combination of causal, non-causal, and masked language training objectives. Here in masked language modeling, the attention is not bidirectional but unidirectional, attending either left-to-right or right-to-left context.

L. Model Adaptation

This section discusses the fundamentals of LLMs adaptation stages, from pre-training to fine-tuning for downstream tasks and utilization. An example of different training stages and inference in LLMs is shown in Figure 6. In this paper, we refer alignment-tuning to aligning with human preferences, while occasionally the literature uses the term alignment for different purposes.

1. Pre-Training: In the very first stage, the model is trained in a self-supervised manner on a large corpus to predict the next tokens given the input. The design choices of LLMs vary from encoder-decoder to decoder-only architectures with different building blocks and loss functions in sections II-F, II-E, II-K.

2. Fine-Tuning: There are different styles to fine-tune an LLM. This section briefly discusses fine-tuning approaches.

Transfer Learning: The pre-trained LLMs perform well for various tasks [6], [15]. But to improve the performance for a downstream task, pre-trained models are fine-tuned with the task-specific data [10], [11], known as transfer learning.

Instruction-tuning: To enable a model to respond to user queries effectively, the pre-trained model is fine-tuned on instruction formatted data i.e., instruction and an input-output pair. Instructions generally comprise multi-task data in plain natural language, guiding the model to respond according to the prompt and the input. This type of fine-tuning improves zero-shot generalization and downstream task performance. Details on formatting instruction data and its various styles are available in [16], [46], [93].

Alignment-tuning: LLMs are prone to generate false, biased, and harmful text. To make them helpful, honest, and harmless models are aligned using human feedback. Alignment involves asking LLMs to generate unexpected responses and then updating their parameters to avoid such responses [20], [21], [94].

It ensures LLMs operate according to human intentions and values. A model is defined to be an "aligned" model if the model fulfills three criteria of helpful, honest, and harmless or "HHH" [95].
Researchers employ reinforcement learning with human feedback (RLHF) [96] for model alignment. In RLHF, a fine-tuned model on demonstrations is further trained with reward modeling (RM) and reinforcement learning (RL), shown in Figure 6. Below we briefly discuss RM and RL pipelines in RLHF.
Reward modeling: trains a model to rank generated responses according to human preferences using a classification objective. To train the classifier humans annotate LLMs generated responses based on HHH criteria.
Reinforcement learning: in combination with the reward model is used for alignment in the next stage. The previously trained reward model ranks LLM-generated responses into preferred vs. dispreferred, which is used to align the model with proximal policy optimization (PPO). This process repeats iteratively until convergence.
Parameter-Efficient Tuning: LLMs require bigger memory and computing for training. To train them using fewer resources, researchers suggested various parameter-efficient fine-tuning techniques by updating few parameters, either by adding new parameters to the model or the existing ones. Some of the commonly used methods are discussed below.
Prompt Tuning: [40], [97] adds trainable prompt token embeddings as prefixes or free-style to the input token embeddings. During fine-tuning only these embedding parameters are trained for the downstream task while keeping the rest of the weights frozen.
Prefix Tuning: [41] adds task-specific trainable prefix vectors to the transformer layers, where only prefix parameters are fine-tuned, and the rest of the model stays frozen. The input sequence tokens can attend prefixes acting as virtual tokens.
Adapter Tuning: module is an encoder-decoder architecture that is placed either sequential or parallel to the attention and feed-forward layers in the transformer block [98], [38], [39]. Only these layers are fine-tuned, and the rest of the model is kept frozen.

3. Prompting/Utilization: Prompting is a method to query trained LLMs for generating responses, as illustrated in Figure 6. LLMs can be prompted in various prompt setups, where they can be adapted to the instructions without fine-tuning and in other cases with fine-tuning on data containing different prompt styles [16], [99], [100]. A good guide on prompt engineering is available at [32]. Below, we will discuss various widely used prompt setups.
Zero-Shot Prompting: LLMs are zero-shot learners and capable of answering queries never seen before. This style of prompting requires LLMs to answer user questions without seeing any examples in the prompt.
In-context Learning: Also known as few-shot learning, here, multiple input-output demonstration pairs are shown to the model to generate the desired response. This adaptation style is also called few-shot learning. A discussion on formatting in-context learning (ICL) templates is available in [50], [46], [18], [16].
Reasoning in LLMs: LLMs are zero-shot reasoners and can be provoked to generate answers to logical problems, task planning, critical thinking, etc. with reasoning. Generating reasons is possible only by using different prompting styles, whereas to improve LLMs further on reasoning tasks many methods [16], [93] train them on reasoning datasets. We discuss various prompting techniques for reasoning below.
Chain-of-Thought (CoT): A special case of prompting where demonstrations contain reasoning information aggregated with inputs and outputs so that the model generates outcomes with step-by-step reasoning. More details on CoT prompts are available in [51], [101], [99].
Self-Consistency: Improves CoT performance by generating multiple responses and selecting the most frequent answer [102].
Tree-of-Thought (ToT): Explores multiple reasoning paths with possibilities to look ahead and backtrack for problem-solving [103].
Single-Turn Instructions: In this prompting setup, LLMs are queried only once with all the relevant information in the prompt. LLMs generate responses by understanding the context either in a zero-shot or few-shot setting.
Multi-Turn Instructions: Solving a complex task requires multiple interactions with LLMs, where feedback and responses from the other tools are given as input to the LLM for the next rounds. This style of using LLMs in the loop is common in autonomous agents.

III. Large Language Models

This section reviews LLMs, briefly describing their architectures, training objectives, pipelines, datasets, and fine-tuning details.

A. Pre-Trained LLMs

Here, we provide summaries of various well-known pre-trained LLMs with significant discoveries, changing the course of research and development in NLP. These LLMs have considerably improved the performance in NLU and NLG domains, and are widely fine-tuned for downstream tasks.

1. General Purpose:

1.1 T5 [10]: An encoder-decoder model employing a unified text-to-text training for all NLP problems, shown in Figure 7. T5 places layer normalization outside the residual path in a conventional transformer model [62]. It uses masked language modeling as a pre-training objective where spans (consecutive tokens) are replaced with a single mask instead of separate masks for each token. This type of masking speeds up the training as it produces shorter sequences. After pre-training, the model is fine-tuned using adapter layers [98] for downstream tasks.

1.2 GPT-3 [6]: The GPT-3 architecture is same as the GPT-2 [5] but with dense and sparse attention in transformer layers similar to the Sparse Transformer [63]. It shows that large models can train on larger batch sizes with a lower learning rate; in order to decide the batch size during training, GPT-3 uses the gradient noise scale as in [104]. Overall, GPT-3 increases model parameters to 175B showing that the

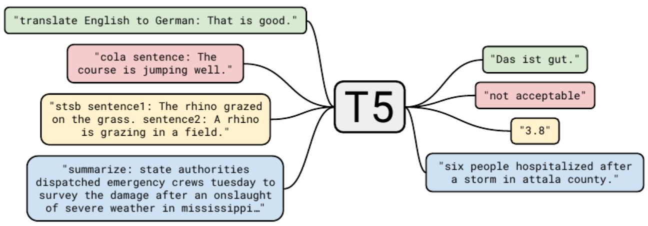

Fig. 7: Unified text-to-text training example, source image from [10].

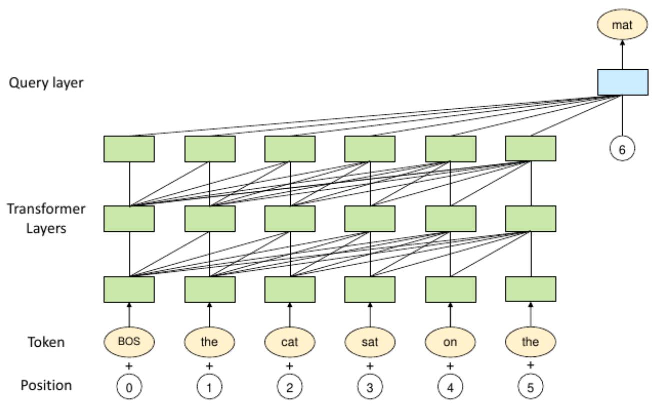

Fig. 8: The image is the article of [105], showing an example of PanGu-α architecture.

performance of large language models improves with the scale and is competitive with the fine-tuned models.

1.3 mT5 [11]: A multilingual T5 model [10] trained on the mC4 dataset with 101 languages. The dataset is extracted from the public common crawl scrape. The model uses a larger vocab size of 250,000 to cover multiple languages. To avoid over-fitting or under-fitting for a language, mT5 employs a data sampling procedure to select samples from all languages. The paper suggests using a small amount of pre-training datasets, including all languages when fine-tuning for a task using English language data. This allows the model to generate correct non-English outputs.

1.4 PanGu-α [105]: An autoregressive model that has a query layer at the end of standard transformer layers, example shown in Figure 8, with aim to predict next token. Its structure is similar to the transformer layer but with an additional embedding for the next position in the attention mechanism, given in Eq. 7.

$$a = p_n W_h^q W_h^k T H_L^T \tag{7}$$

1.5 CPM-2 [12]: Cost-efficient Pre-trained language Models (CPM-2) pre-trains bilingual (English and Chinese) 11B and 198B mixture-of-experts (MoE) models on the WuDaoCorpus [106] dataset. The tokenization process removes "_" white space tokens in the sentencepiece tokenizer. The models are trained with knowledge inheritance, starting with only the Chinese language in the first stage and then adding English and Chinese data. This trained model gets duplicated multiple times to initialize the 198B MoE model. Moreover, to use the model for downstream tasks, CPM-2 experimented with both complete fine-tuning and prompt fine-tuning as in [107] where only prompt-related parameters are updated by inserting prompts at various positions, front, middle, and back. CPM-2 also proposes INFMOE, a memory-efficient framework with a strategy to dynamically offload parameters to the CPU for inference at a 100B scale. It overlaps data movement with inference computation for lower inference time.

1.6 ERNIE 3.0 [108]: ERNIE 3.0 takes inspiration from multi-task learning to build a modular architecture using Transformer-XL [109] as the backbone. The universal representation module is shared by all the tasks, which serve as the basic block for task-specific representation modules, which are all trained jointly for natural language understanding, natural language generation, and knowledge extraction. This LLM is primarily focused on the Chinese language, claims to train on the largest Chinese text corpora for LLM training, and achieved state-of-the-art in 54 Chinese NLP tasks.

1.7 Jurassic-1 [110]: A pair of auto-regressive language models, including a 7B-parameter J1-Large model and a 178B-parameter J1-Jumbo model. The training vocabulary of Jurassic-1 comprise word pieces, complete words, and multi-word expressions without any word boundaries, where possible out-of-vocabulary instances are interpreted as Unicode bytes. Compared to the GPT-3 counterparts, the Jurassic-1 models apply a more balanced depth-to-width self-attention architecture [111] and an improved tokenizer for a faster prediction based on broader resources, achieving a comparable performance in zero-shot learning tasks and a superior performance in few-shot learning tasks given the ability to feed more examples as a prompt.

1.8 HyperCLOVA [112]: A Korean language model with GPT-3 architecture.

1.9 Yuan 1.0 [113]: Trained on a Chinese corpus with 5TB of high-quality text collected from the Internet. A Massive Data Filtering System (MDFS) built on Spark is developed to process the raw data via coarse and fine filtering techniques. To speed up the training of Yuan 1.0 with the aim of saving energy expenses and carbon emissions, various factors that improve the performance of distributed training are incorporated in architecture and training like increasing the number of hidden size improves pipeline and tensor parallelism performance, larger micro batches improve pipeline parallelism performance, and higher global batch size improve data parallelism performance. In practice, the Yuan 1.0 model performs well on text classification, Winograd Schema, natural language inference, and reading comprehension tasks.

1.10 Gopher [114]: The Gopher family of models ranges from 44M to 280B parameters in size to study the effect of *scale* on the LLMs performance. The 280B model beats GPT-3 [6], Jurrasic-1 [110], MT-NLG [115], and others on 81% of the evaluated tasks.

1.11 ERNIE 3.0 TITAN [35]: ERNIE 3.0 Titan extends ERNIE 3.0 by training a larger model with 26x the number of parameters of the latter. This bigger model outperformed other state-of-the-art models in 68 NLP tasks. LLMs produce text with incorrect facts. In order to have control of the generated text with factual consistency, ERNIE 3.0 Titan adds another task, *Credible and Controllable Generations*, to its multi-task learning setup. It introduces additional self-supervised adversarial and controllable language modeling losses to the pre-training step, which enables ERNIE 3.0 Titan to beat other LLMs in their manually selected Factual QA task set evaluations.

1.12 GPT-NeoX-20B [116]: An auto-regressive model that largely follows GPT-3 with a few deviations in architecture design, trained on the Pile dataset without any data deduplication. GPT-NeoX has parallel attention and feed-forward layers in a transformer block, given in Eq. 8, that increases throughput by 15%. It uses rotary positional embedding [66], applying it to only 25% of embedding vector dimension as in [117]. This reduces the computation without performance degradation. Opposite to GPT-3, which uses dense and sparse layers, GPT-NeoX-20B uses only dense layers. The hyperparameter tuning at this scale is difficult; therefore, the model chooses hyperparameters from the method [6] and interpolates values between 13B and 175B models for the 20B model. The model training is distributed among GPUs using both tensor and pipeline parallelism.

$$x + Attn(LN_1(x)) + FF(LN_2(x)) \tag{8}$$

1.13 OPT [14]: It is a clone of GPT-3, developed with the intention to open-source a model that replicates GPT-3 performance. Training of OPT employs dynamic loss scaling [118] and restarts from an earlier checkpoint with a lower learning rate whenever loss divergence is observed. Overall, the performance of OPT-175B models is comparable to the GPT3-175B model.

1.14 BLOOM [13]: A causal decoder model trained on ROOTS corpus with the aim of open-sourcing an LLM. The architecture of BLOOM is shown in Figure 9, with differences like ALiBi positional embedding, an additional normalization layer after the embedding layer as suggested by the bitsandbytes[1] library. These changes stabilize training with improved downstream performance.

1.15 GLaM [119]: Generalist Language Model (GLaM) represents a family of language models using a sparsely activated decoder-only mixture-of-experts (MoE) structure [120], [121]. To gain more model capacity while reducing computation, the experts are sparsely activated where only the best two experts are used to process each input token. The largest GLaM model, GLaM (64B/64E), is about 7× larger than GPT-3 [6], while only a part of the parameters is activated per input token. The largest GLaM (64B/64E) model achieves better overall results as compared to GPT-3 while consuming only one-third of GPT-3's training energy.

1.16 MT-NLG [115]: A 530B causal decoder based on GPT-2 architecture that is roughly 3× GPT-3 model parameters. MT-NLG is trained on filtered high-quality data collected from various public datasets and blends various types of datasets in a single batch, which beats GPT-3 on a number of evaluations.

[1]https://github.com/TimDettmers/bitsandbytes

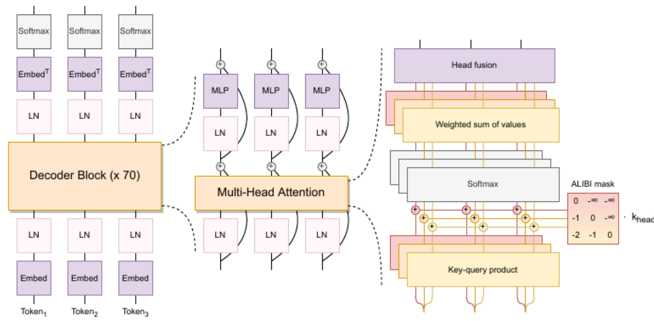

Fig. 9: The BLOOM architecture example sourced from [13].

1.17 Chinchilla [122]: A causal decoder trained on the same dataset as the Gopher [114] but with a little different data sampling distribution (sampled from MassiveText). The model architecture is similar to the one used for Gopher, with the exception of AdamW optimizer instead of Adam. Chinchilla identifies the relationship that model size should be doubled for every doubling of training tokens. Over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens are trained to get the estimates for compute-optimal training under a given budget. The authors train a 70B model with the same compute budget as Gopher (280B) but with 4 times more data. It outperforms Gopher [114], GPT-3 [6], and others on various downstream tasks, after fine-tuning.

1.18 AlexaTM [123]: An encoder-decoder model, where encoder weights and decoder embeddings are initialized with a pre-trained encoder to speedup training. The encoder stays frozen for initial 100k steps and later unfreezed for end-to-end training. The model is trained on a combination of denoising and causal language modeling (CLM) objectives, concatenating $[CLM]$ token at the beginning for mode switching. During training, the CLM task is applied for 20% of the time, which improves the in-context learning performance.

1.19 PaLM [15]: A causal decoder with parallel attention and feed-forward layers similar to Eq. 8, speeding up training 15 times faster. Additional changes to the conventional transformer model include SwiGLU activation, RoPE embeddings, multi-query attention that saves computation cost during decoding, and shared input-output embeddings. During training, loss spiking was observed, and to fix it, model training was restarted from a 100 steps earlier checkpoint by skipping 200-500 batches around the spike. Moreover, the model was found to memorize around 2.4% of the training data at the 540B model scale, whereas this number was lower for smaller models.

PaLM-2 *[124]:* A smaller multi-lingual variant of PaLM, trained for larger iterations on a better quality dataset. The PaLM-2 shows significant improvements over PaLM, while reducing training and inference costs due to its smaller size. To lessen toxicity and memorization, it appends special tokens with a fraction of pre-training data, which shows reduction in generating harmful responses.

1.20 U-PaLM [125]: This method trains PaLM for 0.1% additional compute with UL2 (also named as UL2Restore) objective [89] using the same dataset and outperforms baseline significantly on various NLP tasks, including zero-shot, few-shot, commonsense reasoning, CoT, etc. Training with UL2R involves converting a causal decoder PaLM to a non-causal decoder PaLM and employing 50% sequential denoising, 25% regular denoising, and 25% extreme denoising loss functions.

1.21 UL2 [89]: An encoder-decoder architecture trained using a mixture of denoisers (MoD) objectives. Denoisers include 1) R-Denoiser: a regular span masking, 2) S-Denoiser: which corrupts consecutive tokens of a large sequence and 3) X-Denoiser: which corrupts a large number of tokens randomly. During pre-training, UL2 includes a denoiser token from R, S, X to represent a denoising setup. It helps improve fine-tuning performance for downstream tasks that bind the task to one of the upstream training modes. This MoD style of training outperforms the T5 model on many benchmarks.

1.22 GLM-130B [33]: GLM-130B is a bilingual (English and Chinese) model trained using an auto-regressive mask infilling pre-training objective similar to the GLM [126]. This training style makes the model bidirectional as compared to GPT-3, which is unidirectional. Opposite to the GLM, the training of GLM-130B includes a small amount of multi-task instruction pre-training data (5% of the total data) along with the self-supervised mask infilling. To stabilize the training, it applies embedding layer gradient shrink.

1.23 LLaMA [127], [21]: A set of decoder-only language models varying from 7B to 70B parameters. LLaMA models series is the most famous among the community for parameter-efficient and instruction tuning.

LLaMA-1 *[127]:* Implements efficient causal attention [128] by not storing and computing masked attention weights and key/query scores. Another optimization is reducing number of activations recomputed in backward pass, as in [129].

LLaMA-2 *[21]:* This work is more focused towards fine-tuning a safer and better LLaMA-2-Chat model for dialogue generation. The pre-trained model has 40% more training data with a larger context length and grouped-query attention.

1.24 PanGu-Σ [130]: An autoregressive model with parameters copied from PanGu-α and extended to a trillion scale with Random Routed Experts (RRE), the architectural diagram is shown in Figure 10. RRE is similar to the MoE architecture, with distinctions at the second level, where tokens are randomly routed to experts in a domain instead of using a learnable gating method. The model has bottom layers densely activated and shared across all domains, whereas top layers are sparsely activated according to the domain. This training style allows extracting task-specific models and reduces catastrophic forgetting effects in case of continual learning.

2. Coding:

2.1 CodeGen [131]: CodeGen has similar architecture to the PaLM [15], i.e., parallel attention, MLP layers, and RoPE embeddings. The model is trained on both natural language and programming language data sequentially (trained on the first dataset, then the second and so on) on the following datasets 1) PILE, 2) BIGQUERY and 3) BIGPYTHON. CodeGen proposed a multi-step approach to synthesizing code. The purpose is to simplify the generation of long sequences where the previous prompt and generated code are given as input with the next prompt to generate the next code sequence. CodeGen opensource a Multi-Turn Programming Benchmark (MTPB) to evaluate multi-step program synthesis.

2.2 Codex [132]: This LLM is trained on a subset of public Python Github repositories to generate code from docstrings. Computer programming is an iterative process where the programs are often debugged and updated before fulfilling the requirements. Similarly to this, Codex generates 100 versions of a program by repetitive sampling for a given description, which produces a working solution for 77.5% of the problems passing unit tests. Its powerful version powers Github Copilot[2].

2.3 AlphaCode [133]: A set of large language models, ranging from 300M to 41B parameters, designed for competition-level code generation tasks. It uses the multi-query attention [134] to reduce memory and cache costs. Since competitive programming problems highly require deep reasoning and an understanding of complex natural language algorithms, the AlphaCode models are pre-trained on filtered GitHub code in popular languages and then fine-tuned on a new competitive programming dataset named CodeContests. The CodeContests dataset mainly contains problems, solutions, and test cases collected from the Codeforces platform[3]. The pre-training employs standard language modeling objectives, while GOLD [135] with tempering [136] serves as the training objective for the fine-tuning on CodeContests data. To evaluate the performance of AlphaCode, simulated programming competitions are hosted on the Codeforces platform: overall, AlphaCode ranks at the top 54.3% among over 5000 competitors, where its Codeforces rating is within the top 28% of recently participated users.

2.4 CodeT5+ [34]: CodeT5+ is based on CodeT5 [137], with shallow encoder and deep decoder, trained in multiple stages initially unimodal data (code) and later bimodal data (text-code pairs). Each training stage has different training objectives and activates different model blocks encoder, decoder, or both according to the task. The unimodal pre-training includes span denoising and CLM objectives, whereas bimodal pre-training objectives contain contrastive learning, matching, and CLM for text-code pairs. CodeT5+ adds special tokens with the text to enable task modes, for example, $[CLS]$ for contrastive loss, $[Match]$ for text-code matching, etc.

2.5 StarCoder [138]: A decoder-only model with SantaCoder architecture, employing Flash attention to scale up the context length to 8k. The StarCoder trains an encoder to filter names, emails, and other personal data from the training data. Its fine-tuned variant outperforms PaLM, LLaMA, and LAMDA on HumanEval and MBPP benchmarks.

3. Scientific Knowledge:

3.1 Galactica [139]: A large curated corpus of human scientific knowledge with 48 million papers, textbooks, lecture notes, millions of compounds and proteins, scientific websites, encyclopedias, and more are trained using metaseq library3, which is built on PyTorch and fairscale [140]. The model wraps reasoning datasets with $< work >$ token to provide

[2]https://github.com/features/copilot

[3]https://codeforces.com/

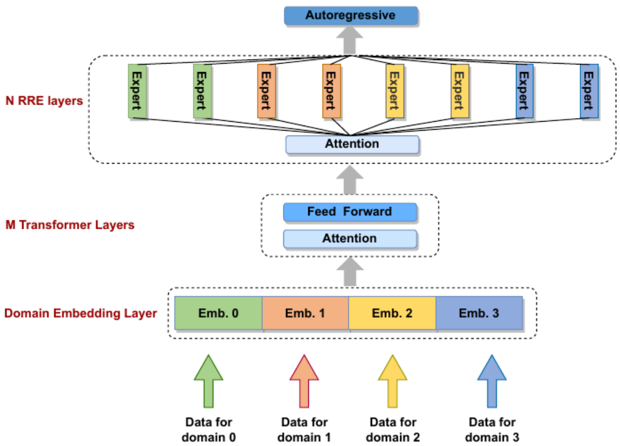

Fig. 10: This example illustrates the PanGu-$\sum$ architecture, as depicted in the image sourced from [130].

step-by-step reasoning context to the model, which has been shown to improve the performance on reasoning tasks.

4. Dialog:

4.1 LaMDA [141]: A decoder-only model pre-trained on public dialog data, public dialog utterances, and public web documents, where more than 90% of the pre-training data is in English. LaMDA is trained with the objective of producing responses that exhibit high levels of quality, safety, and groundedness. To achieve this, discriminative and generative fine-tuning techniques are incorporated to enhance the model's safety and quality aspects. As a result, the LaMDA models can be utilized as a general language model performing various tasks.

5. Finance:

5.1 BloombergGPT [142]: A non-causal decoder model trained using both financial ("FINPILE" from the Bloomberg archive) and general-purpose datasets. The model's architecture is similar to the BLOOM [13] and OPT [14]. It allocates 50B parameters to different blocks of the model using the approach [143]. For effective training, BloombergGPT packs documents together with $< |endoftext| >$ to use maximum sequence length, use warmup batch size starting from 1024 to 2048, and manually reduces the learning rate multiple times during the training.

5.2 Xuan Yuan 2.0 [144]: A Chinese financial chat model with BLOOM's [13] architecture trained on a combination of general purpose, financial, general purpose instructions, and financial institutions datasets. Xuan Yuan 2.0 combined the pre-training and fine-tuning stages to avoid catastrophic forgetting.

B. Fine-Tuned LLMs

Pre-trained LLMs have excellent generalization abilities to unseen tasks. However, because they are generally trained with the objective of next token prediction, LLMs have limited capacity to follow user intent and are prone to generate unethical, toxic or inaccurate responses [20]. For their effective utilization, LLMs are fine-tuned to follow instructions [16], [17], [93] and generate safe responses [20], which also results

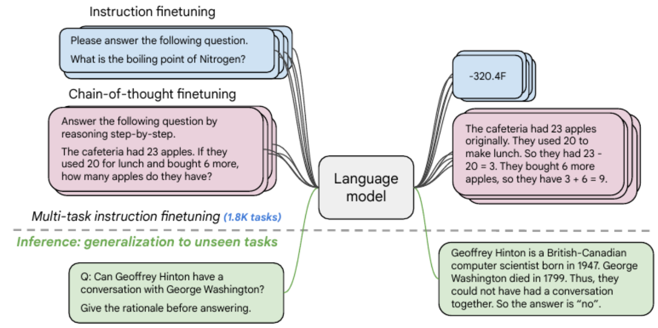

Fig. 11: An example image shows an instance of the Flan training paradigm, taken from [16].

in increasing zero-shot, few-shot, and cross-task generalization [93], [16], [18], with minimal compute increment, e.g., 0.2% of the total pre-training for PaLM 540B [16].
We review various fine-tuned LLMs and strategies for effective fine-tuning in this section.

1. Instruction-Tuning with Manually Created Datasets: Numerous hand-crafted instruction-tuning datasets with different design choices are proposed in the literature to instruction-tune LLMs. The performance of fine-tuned LLMs depends on multiple factors, such as dataset, instruction diversity, prompting templates, model size, and training objectives. Keeping this in view, diverse fine-tuned models have emerged in the literature using manually created datasets. The models T0 [17] and mT0 (multi-lingual) [146] employ templates to convert existing datasets into prompt datasets. They have shown improvements in generalization to zero-shot and held-out tasks. Tk-Instruct [18] fine-tuned the T5 model with in-context instructions to study generalization on unseen tasks when given in-context instructions during test time. The model outperformed Instruct-GPT, despite being smaller in size, i.e., 11B parameters as compared to 175B of GPT-3.
Increasing Tasks and Prompt Setups: Zero-shot and few-shot performance improves significantly by expanding task collection and prompt styles. OPT-IML [93] and Flan [16] curated larger 2k and 1.8k task datasets, respectively. While increasing task size alone is not enough, OPT-IML and Flan add more prompting setups in their datasets, zero-shot, few-shot, and CoT. In continuation, CoT Collection [99] fine-tunes Flan-T5 further on 1.88M CoT samples. Another method [100] uses symbolic tasks with tasks in T0, Flan, etc.

2. Instruction-Tuning with LLMs Generated Datasets: Generating an instruction-tuning dataset requires carefully writing instructions and input-output pairs, which are often written by humans, smaller in size, and less diverse. To overcome this, self-instruct [19] proposed an approach to prompt available LLMs to generate instruction-tuning datasets. Self-instruct outperformed models trained on manually created dataset SUPER-NATURALINSTRUCTIONS (a dataset with 1600+ tasks) [18] by 33%. It starts with a seed of 175 tasks, 1 instruction, and 1 sample per task and iteratively generates new instructions (52k) and instances (82k input-output pairs) using GPT-3 [6]. Contrary to this, Dynosaur [147] uses the

TABLE I: Noteworthy findings and insights from *pre-trained* Large Language Model.

Models	Findings & Insights
T5	• Encoder and decoder with shared parameters perform equivalently when parameters are not shared • Fine-tuning model layers (adapter layers) work better than the conventional way of training on only classification layers
GPT-3	• Few-shot performance of LLMs is better than the zero-shot, suggesting that LLMs are meta-learners
mT5	• Large multi-lingual models perform equivalently to single language models on downstream tasks. However, smaller multi-lingual models perform worse
PanGu-α	• LLMs are good at a few shot capabilities
CPM-2	• Prompt fine-tuning requires updating very few parameters while achieving performance comparable to full model fine-tuning • Prompt fine-tuning takes more time to converge as compared to full model fine-tuning • Inserting prompt tokens in-between sentences can allow the model to understand relations between sentences and long sequences • In an analysis, CPM-2 finds that prompts work as a provider (additional context) and aggregator (aggregate information with the input text) for the model
Codex	• This LLM focuses on code evaluations and introduces a novel way of selecting the best code samples. • The results indicate it is possible to accurately select code samples using heuristic ranking in lieu of a detailed evaluation of each sample, which may not be feasible or feasible in some situations.
ERNIE 3.0	• ERNIE 3.0 shows that a modular LLM architecture with a universal representation module and task-specific representation module helps in finetuning phase. • Optimizing the parameters of a task-specific representation network during the fine-tuning phase is an efficient way to take advantage of the powerful pretrained model.
Jurassic-1	• The performance of an LLM is highly related to the network size. • To improve runtime performance, more operations can be performed in parallel (width) rather than sequentially (depth). • To efficiently represent and fit more text in the same context length, the model uses a larger vocabulary to train a SentencePiece tokenizer without restricting it to word boundaries. This tokenizer improvement can further benefit few-shot learning tasks.
HyperCLOVA	• By employing prompt-based tuning, the performances of models can be improved, often surpassing those of state-of-the-art models when the backward gradients of inputs are accessible.
Yuan 1.0	• The model architecture that excels in pre-training and fine-tuning cases may exhibit contrasting behavior in zero-shot and few-shot learning.
Gopher	• Relative encodings enable models to be evaluated for longer sequences than those on which it was trained.
ERNIE 3.0 Titan	• This LLM builds on top of ERNIE 3.0 and add a self-supervised adversarial loss to distinguish whether a text is generated or the original one. • This distinction ability between real and generate text improves the LLM's performance as compared to ERNIE 3.0.
GPT-NeoX-20B	• Parallel attention + FF layers speed-up training 15% with the same performance as with cascaded layers • Initializing feed-forward output layers before residuals with scheme in [145] avoids activations from growing with increasing depth and width • Training on Pile outperforms GPT-3 on five-shot
OPT	• Restart training from an earlier checkpoint with a lower learning rate if loss diverges • Model is prone to generate repetitive text and stuck in a loop
BLOOM	• None
Galactica	• Galactica's performance has continued to improve across validation set, in-domain, and out-of-domain benchmarks, even with multiple repetitions of the corpus, which is superior to existing research on LLMs. • A working memory token approach can achieve strong performance over existing methods on mathematical MMLU and MATH benchmarks. It sets a new state-of-the-art on several downstream tasks such as PubMedQA (77.6%) and MedMCQA dev (52.9%).
GLaM	• The feed-forward component of each Transformer layer can be replaced with a mixture-of-experts (MoE) module consisting of a set of independent feed-forward networks (*i.e.*, the 'experts'). By sparsely activating these experts, the model capacity can be maintained while much computation is saved. • By leveraging sparsity, we can make significant strides toward developing high-quality NLP models while simultaneously reducing energy consumption. Consequently, MoE emerges as a robust candidate for future scaling endeavors. • The model trained on filtered data shows consistently better performances on both NLG and NLU tasks, where the effect of filtering is more significant on the former tasks. • Filtered pretraining corpora plays a crucial role in the generation capability of LLMs, especially for the downstream tasks. • The scaling of GLaM MoE models can be achieved by increasing the size or number of experts in the MoE layer. Given a fixed budget of computation, more experts contribute to better predictions.
LaMDA	• The model can be fine-tuned to learn to call different external information resources and tools.
MT-NLG	• None.
AlphaCode	• For higher effectiveness and efficiency, a transformer model can be asymmetrically constructed with a shallower encoder and a deeper decoder. • To achieve better performances, it is necessary to employ strategies such as massively scaling up sampling, followed by the filtering and clustering of samples into a compact set. • The utilization of novel sampling-efficient transformer architectures designed to facilitate large-scale sampling is crucial. • Simplifying problem descriptions can effectively improve the model's performance.

Table Continued on Next Page

Models	Findings & Insights
Chinchilla	• The experiments that culminated in the development of Chinchilla determined that for optimal computation during training, the model size and the number of training tokens should be scaled proportionately: for each doubling of the model size, the number of training tokens should be doubled as well.
PaLM	• English-centric models produce better translations when translating to English as compared to non-English • Generalized models can have equivalent performance for language translation to specialized small models • Larger models have a higher percentage of training data memorization • Performance has not yet saturated even at 540B scale, which means larger models are likely to perform better
AlexaTM	• Compared to commonly used Decoder-only Transformer models, seq2seq architecture is more suitable for training generative LLMs given stronger bidirectional attention to the context. • An extra Causal Language Modeling (CLM) task can be added to benefit the model with a more efficient in-context learning, especially for few-shot learning tasks. • The key to training powerful seq2seq-based LLMs lies in mixed pre-training, rather than additional multitask training. • Placing layernorms at the beginning of each transformer layer can improve the training stability of large models.
U-PaLM	• Training with a mixture of denoisers outperforms PaLM when trained further for a few more FLOPs • Training with a mixture of denoisers improves the infilling ability and open-ended text generation diversity
UL2	• Mode switching training enables better performance on downstream tasks • CoT prompting outperforms standard prompting for UL2
GLM-130B	• Pre-training data with a small proportion of multi-task instruction data improves the overall model performance
CodeGen	• Multi-step prompting for code synthesis leads to a better user intent understanding and code generation
LLaMA	• LLaMA is open-source and can be fine-tuned or continually pre-trained to develop new models or instruction-based tools. • A few optimizations are proposed to improve the training efficiency of LLaMA, such as efficient implementation of multi-head self-attention and a reduced amount of activations during back-propagation. • Training exclusively on public data can also achieve state-of-the-art performance. • A constant performance improvement is gained when scaling the model. • Smaller models can also realize good performances using more training data and time.
PanGu-Σ	• Sparse models provide the benefits of large models at a lower computation cost • Randomly Routed Experts reduces catastrophic forgetting effects which in turn is essential for continual learning • Randomly Routed Experts allow extracting a domain-specific sub-model in deployment which is cost-efficient while maintaining a performance similar to the original
BloombergGPT	• Pre-training with general-purpose and task-specific data improves task performance without hurting other model capabilities
XuanYuan 2.0	• Combining pre-training and fine-tuning stages in single training avoids catastrophic forgetting
CodeT5+	• Causal LM is crucial for a model's generation capability in encoder-decoder architectures • Multiple training objectives like span corruption, Causal LM, matching, etc complement each other for better performance
StarCoder	• HHH prompt by Anthropic allows the model to follow instructions without fine-tuning
LLaMA-2	• Model trained on unfiltered data is more toxic but may perform better on downstream tasks after fine-tuning • Model trained on unfiltered data requires fewer samples for safety alignment
PaLM-2	• Data quality is important to train better models • Model and data size should be scaled with 1:1 proportions • Smaller models trained for larger iterations outperform larger models

meta-data of datasets on Huggingface to prompt LLMs to generate multiple task instruction-tuning datasets.

LLaMA Tuned: Various models in literature instruction-tune LLaMA [148] with GPT-3 [6] or GPT-4 [149] generated datasets. Among these, Alpaca [150], Vicuna [151], and LLaMA-GPT-4 [152] are a few general-purpose fine-tuned models, where Alpaca is trained on 52k samples from text-davinci-003, Vicuna on 70k samples from ShareGPT.com, and LLaMA-GPT-4 by re-creating Alpaca instructions from GPT-4. Goat [153] fine-tunes LLaMA for arithmetic tasks (1 million samples) by generating data from ChatGPT and outperforms GPT-4, PaLM, BLOOM, OPT, etc, attributing its success to the LLaMA's consistent tokenization of numbers. HuaTuo [154] is a medical knowledge model, fine-tuned with a generated QA dataset of 8k instructions.

Complex Instructions: Evol-Instruct [155], [156] prompts LLMs to convert given instructions into a more complex set. The instructions are iteratively evolved with re-writing instructions in complex wording and creating new instructions. With this style of automated instruction generation, WizardLM [155] (fine-tuned LLaMA on 250k instructions), outperforms Vicuna and Alpaca, and WizardCoder [156] (fine-tuned StarCoder) beats Claude-Plus, Bard, and others.

3. Aligning with Human Preferences: Incorporating human preferences into LLMs presents a significant advantage in mitigating undesirable behaviors and ensuring accurate outputs. The initial work on alignment, such as InstructGPT [20] aligns GPT-3 using a 3-step approach, instruction-tuning, reward modeling, and fine-tuning with reinforcement learning (RL). The supervised fine-tuned GPT-3 on demonstrations is queried to generate responses, which human labelers rank according to human values, and a reward model is trained on the ranked data. Lastly, the GPT-3 is trained with proximal policy optimization (PPO) using rewards on the generated data from the reward model.

TABLE II: Key insights and findings from the study of *instruction-tuned* Large Language Models.

Models	Findings & Insights
T0	• Multi-task prompting enables zero-shot generalization and outperforms baselines • Even a single prompt per dataset task is enough to improve performance
WebGPT	• The answer quality of LLMs can be further improved with human feedback. • To aid the model in effectively filtering and utilizing relevant information, human labelers play a crucial role in answering questions regarding the usefulness of the retrieved documents. • Interacting a fine-tuned language model with a text-based web-browsing environment can improve end-to-end retrieval and synthesis via imitation learning and reinforcement learning. • Generating answers with references can make labelers easily judge the factual accuracy of answers.
Tk-INSTRUCT	• Instruction tuning leads to a stronger generalization of unseen tasks • More tasks improve generalization whereas only increasing task instances does not help • Supervised trained models are better than generalized models • Models pre-trained with instructions and examples perform well for different types of inputs
mT0 and BLOOMZ	• Instruction tuning enables zero-shot generalization to the tasks never seen before • Multi-lingual training leads to even better zero-shot generalization for both English and non-English • Training on machine-translated prompts improves performance for held-out tasks with non-English prompts • English only fine-tuning on multilingual pre-trained language model is enough to generalize to other pre-trained language tasks
OPT-IML	• Task size sampling to create a batch with most of the task examples is important for better performance • Only example proportional sampling is not enough, training datasets/benchmarks should also be proportional for better generalization/performance • Fully held-out and partially supervised tasks performance improves by scaling tasks or categories whereas fully supervised tasks have no effect • Including small amounts i.e. 5% of pretraining data during fine-tuning is effective • Only 1% reasoning data improves the performance, adding more deteriorates performance • Adding dialogue data makes the performance worse
Flan	• Finetuning with CoT improves performance on held-out tasks • Fine-tuning along with CoT data improves reasoning abilities • CoT tuning improves zero-shot reasoning • Performance improves with more tasks • Instruction fine-tuning improves usability which otherwise is challenging for pre-trained models • Improving the model's performance with instruction tuning is compute-efficient • Multitask prompting enables zero-shot generalization abilities in LLM
Sparrow	• The judgments of labelers and the alignments with defined rules can help the model generate better responses. • Good dialogue goals can be broken down into detailed natural language rules for the agent and the raters. • The combination of reinforcement learning (RL) with reranking yields optimal performance in terms of preference win rates and resilience against adversarial probing.
WizardCoder	• Fine-tuning with re-written instruction-tuning data into a complex set improves the performance significantly
LLaMA-2-Chat	• Model learns to write safe responses with fine-tuning on safe demonstrations, while additional RLHF step further improves model safety and make it less prone to jailbreak attacks
LIMA	• Less high quality data is enough for fine-tuned model generalization

LLaMA 2-Chat [21] improves alignment by dividing reward modeling into helpfulness and safety rewards and using rejection sampling in addition to PPO. The initial four versions of LLaMA 2-Chat are fine-tuned with rejection sampling and then with PPO on top of rejection sampling.

Aligning with Supported Evidence: This style of alignment allows the model to generate responses with proofs and facts, reduces hallucination, and assists humans more effectively, which increases trust in the model's output. Similar to the RLHF training style, a reward model is trained to rank generated responses containing web citations in answers to questions, which is later used to train the model, as in GopherCite [157], WebGPT [158], and Sparrow [159]. The ranking model in Sparrow [159] is divided into two branches, preference reward and rule reward, where human annotators adversarial probe the model to break a rule. These two rewards together rank a response to train with RL.

Aligning Directly with SFT: The PPO in the RLHF pipeline is complex, memory-intensive, and unstable, requiring multiple models, reward, value, policy, and reference models. Avoiding this sophisticated alignment pipeline is possible by incorporating minimal changes in the supervised fine-tuning (SFT) pipeline as in [160], [161], [162], with better or comparable performance to PPO. Direct preference optimization (DPO) [160] trains a model directly on the human-preferred responses to maximize the likelihood of preferred against unpreferred responses, with per-sample importance weight. Reward ranked fine-tuning RAFT [161] fine-tunes the model on ranked responses by the reward model. Preference ranking optimization (PRO) [163] and RRHF [162] penalize the model to rank responses with human preferences and supervised loss. On the other hand, chain-of-hindsight (CoH) [164] provides feedback to the model in language rather than reward, to learn good versus bad responses.

Aligning with Synthetic Feedback: Aligning LLMs with human feedback is slow and costly. The literature suggests a semi-automated process to align LLMs by prompting LLMs to generate helpful, honest, and ethical responses to the queries, and fine-tuning using the newly created dataset. Constitutional AI [165] replaces human feedback in RLHF with AI, calling it RL from AI feedback (RLAIF). AlpacaFarm [166] designs prompts to imitate human feedback using LLMs APIs. Opposite to constitutional AI, AlpacaFarm injects noise in feedback to replicate human mistakes. Self-Align [94] prompts the LLM with ICL examples, instructing the LLM about what the response should contain to be considered useful and ethical. The same LLM is later fine-tuned with the new dataset.

Aligning with Prompts: LLMs can be steered with prompts to generate desirable responses without training [167], [168]. The self-correction prompting in [168] concatenates instructions and CoT with questions, guiding the model to answer its instruction following strategy to ensure moral safety before the actual answer. This strategy is shown to reduce the harm in generated responses significantly.

Red-Teaming/Jailbreaking/Adversarial Attacks: LLMs exhibit harmful behaviors, hallucinations, leaking personal information, and other shortcomings through adversarial probing. The models are susceptible to generating harmful responses even though they are aligned for safety [169], [170]. Red-teaming is a common approach to address illicit outputs, where the LLMs are prompted to generate harmful outputs [170], [171]. The dataset collected through red-teaming is used to fine-tune models for safety. While red-teaming largely relies on human annotators, another work [172] red-team LLMs to find prompts that lead to harmful outputs of other LLMs.

4. Continue Pre-Training: Although fine-tuning boosts a model's performance, it leads to catastrophic forgetting of previously learned information. Concatenating fine-tuning data with a few randomly selected pre-training samples in every iteration avoids network forgetting [173], [144]. This is also effective in adapting LLMs for cases where fine-tuning data is small and the original capacity is to be maintained. Prompt-based continued pre-training (PCP) [174] trains the model with text and instructions related to tasks and then finally instruction-tunes the model for downstream tasks.

5. Sample Efficiency: While fine-tuning data is generally many-fold smaller than the pre-training data, it still has to be large enough for acceptable performance [16], [93], [18] and requires proportional computing resources. To study the effects on performance with less data, existing literature [175], [176] finds that the models trained on lesser data can outperform models trained with more data. In [175], 25% of the total downstream data is found enough for state-of-the-art performance. Selecting coreset-based 0.5% of the total instruction-tuning data improves the model performance by 2% in [176], as compared to the complete data tuning. Less is more for alignment (LIMA) [177] uses only 1000 carefully created demonstrations to fine-tune the model and has achieved comparable performance to GPT-4.

C. Increasing Context Window

LLMs are trained with limited context windows due to expensive attention and high memory requirements. A model trained on limited sequence lengths fails to generalize to unseen lengths at inference time [178], [45]. Alternatively, LLMs with ALiBi [65] positional encodings can perform zero-shot length extrapolation. However, ALiBi has less expressive power [66] and inferior performance on multiple benchmarks [42], and many LLMs use RoPE positional embedding that is unable to perform zero-shot extrapolation. A larger context length has benefits such as a better understanding of longer documents, more samples in in-context learning, execution of bigger reasoning processes, etc. Expanding context length during fine-tuning is slow, inefficient, and computationally expensive [45]. Therefore, researchers employ various context window extrapolation techniques discussed below.

Position Interpolation: Rather than extrapolating, [45] shows that interpolating position encodings within the pre-trained context window are more effective. The work demonstrates that only 1000 steps of fine-tuning are enough to achieve better results on larger windows without performance loss compared to the original context size. Giraffe [42] uses power scaling in RoPE, and YaRN [43] proposed NTK-aware interpolation.

Efficient Attention Mechanism: Dense global attention is one of the major constraints in training larger context window LLMs. Using efficient attention variants, such as local, sparse, and dilated attention, reduces the computation cost significantly. LongT5 [44] proposes transient global attention (TGlobal), applying attention to local and global tokens (windowing token averaging). The model replaces attention in T5 [10] with TGlobal attention, pre-trains the model on 4098 sequence length, fine-tunes on larger window sizes, as large as 16k, and improves task performance with longer inputs. This shows the extrapolation ability of TGlobal attention with only fine-tuning. COLT5 [179] uses two branches, one with lightweight and the other with heavyweight attention and feed-forward layers. All tokens are processed from the lightweight branch, and only important tokens are routed to the heavyweight branch. LongNet [180] replaces standard attention with dilated attention, expanding sequence length to 1 billion tokens. LongLoRA [181] proposes shift-short attention, used during fine-tuning to reduce dense attention costs, while the model during inference can use dense attention and achieve similar performance as full attention fine-tuning.

Extrapolation without Training: LM-Infinite [178] and parallel context windows (PCW) [182] show length extrapolation is possible using pre-trained LLMs. LM-Infinite suggested Λ-shaped attention applied within the original context window limits. Likewise, PCW chunks larger inputs into the pre-trained context lengths and applies the same positional encodings to each chunk.

D. Robotics

LLMs have been rapidly adopted across various domains in the scientific community due to their multipurpose capabilities [46]. In robotics research, the LLMs have very promising applications as well, such as enhancing human-robot interaction [28], [183], [184], [185], task planning [186], [187], [188], navigation [189], [190], and learning [191], [192]. They can enable robots to understand and generate natural language, aiding in instruction following, data annotation, and collaborative problem-solving. They can facilitate continuous learning by allowing robots to access and integrate information from a wide range of sources. This can help robots acquire new skills, adapt to changes, and refine their performance based on real-time data.

LLMs have also started assisting in simulating environments for testing and offer potential for innovative research in robotics, despite challenges like bias mitigation and integration complexity. The work in [193] focuses on personalizing robot household cleanup tasks. By combining language-based planning and perception with LLMs, such that having users provide object placement examples, which the LLM summarizes to generate generalized preferences, they show that robots can generalize user preferences from a few examples. An embodied LLM is introduced in [26], which employs a Transformer-based language model where sensor inputs are embedded alongside language tokens, enabling joint processing to enhance decision-making in real-world scenarios. The model is trained end-to-end for various embodied tasks, achieving positive transfer from diverse training across language and vision domains. LLMs have also been explored as zero-shot human models for enhancing human-robot interaction.

The study in [28] demonstrates that LLMs, trained on vast text data, can serve as effective human models for certain HRI tasks, achieving predictive performance comparable to specialized machine-learning models. However, limitations were identified, such as sensitivity to prompts and difficulties with spatial/numerical reasoning. In another study [194], the authors enable LLMs to reason over sources of natural language feedback, forming an "inner monologue" that enhances their ability to process and plan actions in robotic control scenarios. They combine LLMs with various forms of textual feedback, allowing the LLMs to incorporate conclusions into their decision-making process for improving the execution of user instructions in different domains, including simulated and real-world robotic tasks involving tabletop rearrangement and mobile manipulation. All of these studies employ LLMs as the core mechanism for assimilating everyday intuitive knowledge into the functionality of robotic systems.

Planning: LLMs are increasingly integral in robotics, particularly for strategic planning [186], [195], [196]. Their proficiency in processing and generating natural language is crucial for enhancing human-robot interaction and enabling robots to understand and execute complex tasks based on verbal instructions. LLMs also play a key role in task planning, a higher-level cognitive process involving the determination of sequential actions needed to achieve specific goals. This proficiency is crucial across a spectrum of applications, from autonomous manufacturing processes to household chores, where the ability to understand and execute multi-step instructions is of paramount significance.

Manipulation: In the area of manipulation [197], [198], [199], [200], LLMs enhance a robot's dexterity and adaptability, excelling in tasks like object recognition, grasping, and collaboration. They analyze visual and spatial information to determine the most effective approach to interact with objects, proving invaluable in operations requiring precision and flexibility, such as surgical procedures or assembly line tasks. They also enable the integration of sensor inputs and linguistic cues in an embodied framework, enhancing decision-making in real-world scenarios. It enhances the model's performance across various embodied tasks by allowing it to gather insights and generalize from diverse training data spanning language and vision domains.

Navigation: LLMs have revolutionized the navigation in robotics [201], [202], [203], [204], offering significant potential to enhance a robot's ability to navigate complex environments with precision and adaptability. Motion planning [189], in particular, stands out as a critical domain where LLMs have shown remarkable promise, excelling in generating feasible paths and trajectories for robots, accounting for intricate environmental details. This ability proves particularly valuable in scenarios requiring precise and dynamically adaptable navigation, as observed in environments like warehouses, transport and healthcare facilities, and smart residences. LLMs have also played a key role in localization and mapping, which are foundational components for successful robot navigation. They empower robots to determine their precise position within an environment while concurrently constructing or updating a spatial representation of their surroundings. This capability is crucial for tasks demanding spatial awareness, including autonomous exploration, search and rescue missions, and the operations of mobile robots. They have also contributed significantly to the proficiency of collision-free navigation within the environment while accounting for obstacles and dynamic alterations, playing an important role in scenarios where robots are tasked with traversing predefined paths with accuracy and reliability, as seen in the operations of automated guided vehicles (AGVs) and delivery robots (e.g., SADRs – pedestrian sized robots that deliver items to customers without the involvement of a delivery person).

E. Multimodal LLMs

Inspired by the success of LLMs in natural language processing applications, an increasing number of research works are now facilitating LLMs to perceive different modalities of information like image [205], [206], [207], video [208], [209], [210], audio [211], [210], [212], *etc.* Multimodal LLMs (MLLMs) present substantial benefits compared to standard LLMs that process only text. By incorporating information from various modalities, MLLMs can achieve a deeper understanding of context, leading to more intelligent responses infused with a variety of expressions. Importantly, MLLMs align closely with human perceptual experiences, leveraging the synergistic nature of our multisensory inputs to form a comprehensive understanding of the world [212], [26]. Coupled with a user-friendly interface, MLLMs can offer intuitive, flexible, and adaptable interactions, allowing users to engage with intelligent assistants through a spectrum of input methods. According to the ways of constructing models, current MLLMs can be generally divided into three streams:

pre-training, fine-tuning, and prompting. In this section, we will discuss more details of these main streams, as well as the important application of MLLMs in visual reasoning.

Pre-training: This stream of MLLMs intends to support different modalities using unified end-to-end models. For instance, Flamingo [205] applies gated cross-attention to fuse vision and language modalities, which are collected from pre-trained and frozen visual encoder and LLM, respectively. Moreover, BLIP-2 [206] proposes a two-stage strategy to pre-train a Querying Transformer (Q-Former) for the alignment between vision and language modalities: in the first stage, vision-language representation learning is bootstrapped from a frozen visual encoder; and in the second stage, a frozen LLM bootstraps vision-to-language generative learning for zero-shot image-to-text generation. Similarly, MiniGPT-4 [213] also deploys pre-trained and frozen ViT [214], Q-Former and Vicuna LLM [151], while only a linear projection layer needs to be trained for vision and language modalities alignment.

Fine-tuning: Derived from instruction tuning [16] for NLP tasks [20], [16], [93], researchers are now fine-tuning pre-trained LLMs using multimodal instructions. Following this method, LLMs can be easily and effectively extended as multimodal chatbots [213], [207], [29] and multimodal task solvers [215], [30], [216]. The key issue of this stream of MLLMs is to collect multimodal instruction-following data for fine-tuning [217]. To address this issue, the solutions of benchmark adaptation [215], [218], [219], self-instruction [19], [31], [220], and hybrid composition [221], [216] are employed, respectively. To mitigate the gap between the original language modality and additional modalities, the learnable interface is introduced to connect different modalities from frozen pre-trained models. Particularly, the learnable interface is expected to work in a parameter-efficient tuning manner: *e.g.*, LLaMA-Adapter [222] applies an efficient transformer-based adapter module for training, and LaVIN [221] dynamically learns the multimodal feature weights using a mixture-of-modality adapter. Different from the learnable interface, the expert models can directly convert multimodalities into language: *e.g.*, VideoChat-Text [208] incorporates Whisper [223], a speech recognition expert model, to generate the captions of given videos for the understanding of following LLMs.

Prompting: Different from the fine-tuning technique that directly updates the model parameters given task-specific datasets, the prompting technique provides certain context, examples, or instructions to the model, fulfilling specialized tasks without changing the model parameters. Since prompting can significantly reduce the need for large-scale multimodal data, this technique is widely used to construct MLLMs. Particularly, to solve multimodal Chain of Thought (CoT) problems [101], LLMs are prompted to generate both the reasoning process and the answer given multimodal inputs [224]. On this front, different learning paradigms are exploited in practice: for example, Multimodal-CoT [224] involves two stages of rationale generation and answer inference, where the input of the second stage is a combination of the original input and the output of the first stage; and CoT-PT [225] applies both prompt tuning and specific visual bias to generate a chain of reasoning implicitly. In addition to CoT problems, LLMs can also be prompted with multimodal descriptions and tools, effectively dividing complex tasks into sub-tasks [226], [227].

Visual Reasoning Application: Recent visual reasoning systems [228], [229], [230], [231] tend to apply LLMs for better visual information analysis and visual-language integration. Different from previous works [232], [233] that rely on limited VQA datasets and small-scale neural networks, current LLM-aided methods offer benefits of stronger generalization ability, emergent ability, and interactivity [217]. To realize visual reasoning with the help of LLMs, prompting and fine-tuning techniques can also be utilized: for example, PointClip V2 [229] applies LLMs to generate 3D-specific prompts, which are encoded as textual features and then combined with visual features for 3D recognition; and GPT4Tools [31] employs LoRA [234] to fine-tune LLMs following tool-related instructions. Serving as a controller [231], decision maker [235], or semantics refiner [228], [236], LLMs significantly facilitates the progress of visual reasoning research.

F. Augmented LLMs

LLMs are capable of learning from the examples concatenated with the input, known as context augmentation, in-context learning (ICL), or few-shot prompting. They show excellent generalization to unseen tasks with few-shot prompting, enabling LLMs to answer queries beyond the capacity acquired during training [6], [51]. These emergent abilities allow for adapting the model without fine-tuning - a costly process. Aside from this, hallucination, producing inaccurate, unsafe or factually incorrect responses, is common for LLMs, which is avoided by augmenting contextual data. While the user can provide in-context samples in the query [50], [32], here we specifically refer to the methods that access external storage programmatically, calling them augmented LLMs.

The literature suggests various external memory designs to augment LLMs, long-term [237], [238], [239], [240], short-term [241], symbolic [242], and non-symbolic [243], [244]. The memory can be maintained in different formats such as documents, vectors, or databases. A few systems maintain intermediate memory representations to retain information across multiple iterations [240], [238], while others extract important information from the datasets and save it in memory for recall [245]. The memory read and write operations are performed either with or without LLMs cooperation [238], [246], [240], [247], acting as a feedback signal in [241]. We discuss different types of augmented LLMs below.

1. Retrieval Augmented LLMs: LLMs may have limited memory and outdated information, leading to inaccurate responses. Retrieving relevant information from external up-to-date storage enables the LLMs to accurately answer with references and utilize more information. With retrieval augmentation, smaller models have been shown to perform at par with larger models. For instance, the 11B model can become competitive to 540B PaLM in [25] and 7.5B to 280B Gopher in [239]. Retrieval augmented language modeling (RALM) has two major components, shown in Figure 12, namely: 1) retriever and 2) language model. In RALM, the retriever plays a crucial role in driving LLM response, where

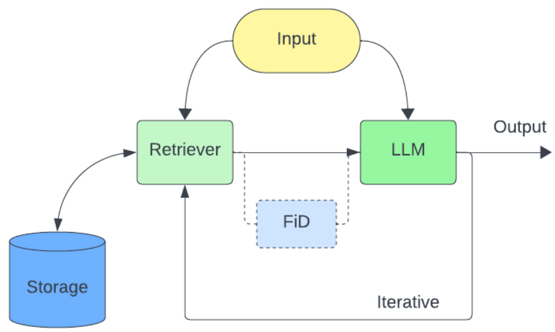

Fig. 12: A flow diagram of Retrieval Augmented LLMs. The retriever extracts a similar context to the input and forwards it to the LLM either in simple language or encoded through Fusion-in-Decoder (FiD). Depending on the task, retrieval and generation may repeat multiple times.

incorrect information can steer LLMs to false behavior. This leads to the development of various methods to retrieve accurate information and fuse with the query for better performance.

Zero-Shot Retrieval Augmentation: This kind of augmentation keeps the original LLM architecture and weights unchanged and uses BM25 [248], nearest neighbors, or frozen pre-trained models like Bert [7] as a retriever. The retrieved information is provided as input to the model for response generation, shown to improve performance over LLMs without retrieval [244], [249]. In some scenarios, multiple retrieval iterations are required to complete the task. The output generated in the first iteration is forwarded to the retriever to fetch similar documents. Forward-looking active retrieval (FLARE) [243] initially generates the response and corrects the output by retrieving relevant documents if the response contains low-confidence tokens. Similarly, RepoCoder [250] fetches code snippets recursively for code completion.

Training with Retrieval Augmentation: To reduce failures in retrieval augmentation generation (RAG), researchers train or fine-tune retrievers and LLMs with a retrieval augmentation pipeline. We discuss the literature below based on their focus on the respective training processes of the pipeline.

Training LLM: Retrieval-enhanced transformer (RETRO) [239] shows pre-training smaller LLMs with RAG pipeline outperforms larger LLMs, such as GPT-3 trained without RAG. RETRO uses a 2-trillion token subset of MassiveText as a database. The retrieval pipeline divides the input query into subsets and retrieves relevant chunks from the database for each subset, encoded together with input intermediate representations for generating tokens. It uses cross-chunked attention to attend to previous chunks auto-regressively. A study on RETRO [251] shows models pre-trained without RAG but fine-tuned using RAG lack the performance gains obtained by pre-training with RAG.

Training Retriever: Quality of responses generated by LLMs is highly dependent on the in-context examples. Therefore, [252], [253], [254], [255] train retrievers to retrieve accurate few-shot samples while keeping the LLM frozen for generation. Retrieved samples are ranked to build ground-truth data to train retrievers with contrastive learning in [252], [254]. RoBERTa is trained for downstream tasks in [253] for ICL samples retrieval. REPLUG [255] trains the retriever with supervised signals from the frozen LLM-generated outputs.

Training Retriever and LLM: Further benefits are achieved by training both the retriever and the model in [25], [256], [257]. In this case, the error propagates back to the retriever, updating both the language model and the retriever. While masked language modeling (MLM) is a common pre-training objective [25], [257], retrieval pre-trained transformer (RPT) [256] used document chunk prediction as a pre-training objective for long text modeling.

Encoded Context Augmentation: Concatenating retrieved documents with the query becomes infeasible as the sequence length and sample size grow. Encoding the context and fusing it with the decoder (Fusion-in-Decoder) using cross-attention makes it possible to augment more samples without increasing computation costs significantly [258], [239], [256], [25].

Web Augmented: Locally stored memory, but external to LLM, has limited information. However, a large amount of information is available on the internet, which gets updated regularly. Rather than storing information locally, various methods retrieve query-related context through a web search and forward it to LLMs [259], [260], [158].

2. *Tool Augmented LLMs:* While RAG relies on the retriever to provide context to the LLM to answer queries, tool augmented LLMs capitalize on the reasoning abilities of LLMs to iteratively plan by dividing tasks into sub-tasks, selecting necessary tools, and taking actions to complete the task [261], [262], [263], [27]. A generic pipeline of tool-augmented LLMs is shown in Figure 13, where different modules in Figure 13 are selected in a loop until the task completion.

Zero-Shot Tool Augmentation: LLMs in-context learning and reasoning abilities enable them to interact with tools without training. Automatic reasoning and tool-use (ART) [263] builds a task library with demonstrations of reasoning steps and calling external tools. It retrieves similar task examples and provides the context to the LLM for inference. Aside from this, [264] shows tool documentation is enough to teach LLMs to use tools without demonstrations. RestGPT [265] integrates LLMs with RESTful APIs by decomposing tasks into planning and API selection steps. The API selector understands the API documentation to select a suitable API for the task and plan the execution. ToolkenGPT [266] uses tools as tokens by concatenating tool embeddings with other token embeddings. During inference, the LLM generates the tool tokens representing the tool call, stops text generation, and restarts using the tool execution output.

Training with Tool Augmentation: LLMs are trained to interact with diverse tools, enhancing planning abilities to overcome the limitations of zero-shot tool augmentation [267], [27], [268], [269]. Gorilla [267] instruction-tunes LLaMA with information retrieval from API documentation. It uses self-

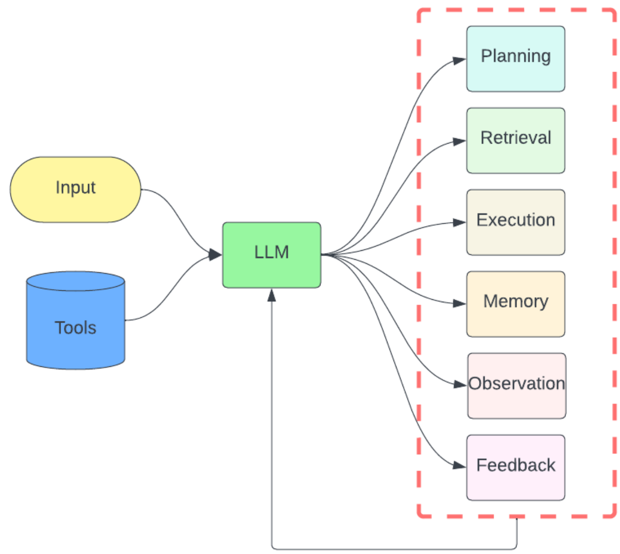

Fig. 13: A basic flow diagram of tool augmented LLMs. Given an input and a set of available tools, the model generates a plan to complete the task. The tool augmented LLMs utilize different modules iteratively, such as retriever, tool execution, read-write to memory, feedback, etc., depending on the task.

instruct [19] data generation pipeline with GPT-4 by providing in-context examples retrieved from API documentation. Tool augmented language model (TALM) [27] fine-tunes T5 [10] for tool use with a self-play approach, where it iteratively completes tool manipulation tasks and includes them back in the training set. ToolLLM [269] collects 16k APIs from RapidAPI. It samples APIs from the list to generate an instruction-tuning dataset using ChatGPT in single-tool and multi-tool scenarios. For high-quality datasets, ToolLLM suggested a depth-first search-based decision tree (DFSDT) method to generate ground-truths with diverse reasoning and planning.

Multimodal Tool Augmentation: The compositional reasoning capacity of LLMs allows them to manipulate tools in multimodal settings [261], [262], [270]. Following the pipeline shown in Figure 13, the LLM outlines a plan, generally executing in a sequence: Plan $\rightarrow$ Tool selection $\rightarrow$ Execute $\rightarrow$ Inspect $\rightarrow$ Generate, to respond to the user query. Here, the database of tools is rich in modalities, including text, images, etc. Many of the multimodal tool augmentation systems employ multimodal LLMs [271], [272], [270], [262], while others utilize single modality LLMs and generate a plan on using different modality tools to solve multimodal queries [273].

IV. Findings & Insights

Training a billion-scale model is difficult as compared to a smaller model. LLMs are prone to various instabilities during training, such as hardware failure and instability. Other than this, LLMs exhibit different behaviors such as emergent abilities, improved zero-shot, few-shot, and reasoning abilities. Researchers report these essential details in their papers for results reproduction and field progress. We identify critical information in Table I and II such as architecture, training strategies, and pipelines that improve LLMs' performance or other abilities acquired because of changes mentioned in section III.

V. Model Configurations

We provide different statistics of pre-trained and instruction-tuned models in this section. This includes information such as publication venue, license type, model creators, steps trained, parallelism, etc in Table III and Table IV. Architecture details of pre-trained LLMs are available in Table V. Providing these details for instruction-tuned models is unnecessary because it fine-tunes pre-trained models for instruction datasets. Hence, architectural details are the same as the baselines. Moreover, optimization settings for various LLMs are available in Table VI and Table VII. We do not include details on precision, warmup, and weight decay in Table VII. Neither of these details are important as others to mention for instruction-tuned models nor provided by the papers.

VI. Datasets and Evaluation

Generating training and evaluation datasets is expensive because of the large-scale data demand of LLMs. Hence, datasets for training and benchmarking these models are topics of key importance. In Fig. 14, we show the distribution of the existing datasets for various NLP tasks. We restrict our distribution to only the most important tasks in the literature by including tasks with at least 20 datasets. LLMs can directly benefit from these datasets for training and evaluation. A summary of the training and evaluation datasets commonly used by LLMs is provided next.

A. Training Datasets

The performance of LLMs largely depends on the training data's quality, size, and diversity. Preparing training datasets of high quality at a large scale is laborious. Researchers have suggested various pre-training and fine-tuning datasets to enhance LLMs capabilities. We summarize these efforts in Table VIII. While numerous training datasets are available in the literature, we cover the most widely used ones in our summary.

B. Evaluation Datasets and Tasks

The evaluation of LLMs is important in gauging their proficiency and limitations. This process measures the model's ability to comprehend, generate, and interact with human language across a spectrum of tasks. Evaluating a language model (LM) is divided into two broader categories: 1) natural language understanding (NLU) and 2) natural language generation (NLG). It is emphasized that tasks in NLU and NLG are softly categorized and are often used interchangeably in the literature.

Natural Language Understanding: This task measures the language understanding capacity of LMs. It encompasses

TABLE III: Summary of pre-trained LLMs (>10B). Only the LLMs discussed individually in the previous sections are summarized. "Data/Tokens" is the model's pre-training data which is either the number of tokens or data size. "Data Cleaning" indicates whether the data cleaning is performed or not. This includes heuristics (Heur), deduplication (Dedup), quality filtering (QF), and privacy filtering (PF), "Cost" is the calculated training cost obtained by multiplying the GPUs/TPUs hourly rate with the number of GPUs and the training time. The actual cost may vary due to many reasons such as using in-house GPUs or getting a discounted rate, re-training, number of employees working on the problem, etc. "Training Parallelism" indicates distributed training using data parallelism (D), tensor parallelism (T), pipeline parallelism (P), model parallelism (M), optimizer parallelism (OP), and rematerialization (R), where for "Library" column, "DS" is a short form for Deep Speed. In column "Commercial Use", we assumed a model is for non-commercial purposes if its license is not available.

Models	Publication Venue	License Type	Model Creators	Purpose	No. of Params	Commercial Use	Steps Trained	Data/ Tokens	Data Cleaning	No. of Processing Units	Processing Unit Type	Training Time	Calculated Train. Cost	Training Parallelism	Library
T5 [10]	JMLR'20	Apache-2.0	Google	General	11B	✓	1M	1T	Heur+Dedup	1024	TPU v3	-	-	D+M	Mesh TensorFlow
GPT-3 [6]	NeurIPS'20	-	OpenAI	General	175B	×	-	300B	Dedup+QF	-	V100	-	-	M	-
mT5 [11]	NAACL'21	Apache-2.0	Google	General	13B	✓	1M	1T	-	-	-	-	-	-	-
PanGu-α [105]	arXiv'21	Apache-2.0	Huawei	General	200B	✓	260k	1.1TB	Heur+Dedup	2048	Ascend 910	-	-	D+OP+P+O+R	MindSpore
CPM-2 [12]	AI Open'21	MIT	Tsinghua	General	198B	✓	1M	2.6TB	Dedup	-	-	-	-	D+M	JAXFormer
Codex [132]	arXiv'21	-	OpenAI	Coding	12B	×	-	100B	Heur	-	-	-	-	-	-
ERNIE 3.0 [108]	arXiv'21	-	Baidu	General	10B	×	120k*	375B	Heur+Dedup	384	V100	-	-	M*	PaddlePaddle
Jurassic-1 [110]	White-Paper'21	Apache-2.0	AI21	General	178B	✓	-	300B	-	800	GPU	-	-	D+M+P	Megatron+DS
HyperCLOVA [112]	EMNLP'21	-	Naver	General	82B	×	-	300B	Clf+Dedup+PF	1024	A100	321h	1.32 Mil	M	Megatron
Yuan 1.0 [113]	arXiv'21	Apache-2.0	-	General	245B	✓	26k*	180B	Heur+Clf+Dedup	2128	GPU	-	-	D+T+P	-
Gopher [114]	arXiv'21	-	Google	General	280B	×	-	300B	QF+Dedup	4096	TPU v3	920h	13.19 Mil	D+M	JAX+Haiku
ERNIE 3.0 Titan [35]	arXiv'21	-	Baidu	General	260B	×	-	300B	Heur+Dedup	-	Ascend 910	-	-	D+M+P+D*	PaddlePaddle
GPT-NeoX-20B [274]	BigScience'22	Apache-2.0	EleutherAI	General	20B	✓	150k	825GB	None	96	40G A100	-	-	M	Megatron+DS+PyTorch
OPT [14]	arXiv'22	MIT	Meta	General	175B	✓	150k	180B	Dedup	992	80G A100	-	-	D+T	Megatron
BLOOM [13]	arXiv'22	RAIL-1.0	BigScience	General	176B	✓	-	366B	Dedup+PR	384	80G A100	2520h	3.87 Mil	D+T+P	Megatron+DS
Galactica [139]	arXiv'22	Apache-2.0	Meta	Science	120B	×	225k	106B	Dedup	128	80GB A100	-	-	-	Metaseq
GLaM [119]	ICML'22	-	Google	General	1.2T	×	600k*	600B	Clf	1024	TPU v4	-	-	M	GSPMD
LaMDA [141]	arXiv'22	-	Google	Dialog	137B	×	3M	2.81T	Filtered	1024	TPU v3	1384h	4.96 Mil	D+M	Lingvo
MT-NLG [115]	arXiv'22	Apache-v2.0	MS.+Nvidia	General	530B	×	-	270B	-	4480	80G A100	-	-	D+T+P	Megatron+DS
AlphaCode [133]	Science'22	Apache-v2.0	Google	Coding	41B	✓	205k	967B	Heur+Dedup	-	TPU v4	-	-	M	JAX+Haiku
Chinchilla [122]	arXiv'22	-	Google	General	70B	×	-	1.4T	QF+Dedup	-	TPUv4	-	-	-	JAX+Haiku
PaLM [15]	arXiv'22	-	Google	General	540B	×	255k	780B	Heur	6144	TPU v4	-	-	D+M	JAX+T5X
AlexaTM [123]	arXiv'22	Apache v2.0	Amazon	General	20B	×	500k	1.1T	Filtered	128	A100	2880h	1.47 Mil	M	DS
U-PaLM [125]	arXiv'22	-	Google	General	540B	×	20k	-	-	512	TPU v4	120h	0.25 Mil	-	-
UL2 [89]	ICLR'23	Apache-2.0	Google	General	20B	✓	2M	1T	-	512	TPU v4	-	-	M	JAX+T5X
GLM [33]	ICLR'23	Apache-2.0	Multiple	General	130B	×	-	400B	-	768	40G A100	1440h	3.37 Mil	M	-
CodeGen [131]	ICLR'23	Apache-2.0	Salesforce	Coding	16B	✓	650k	577B	Heur+Dedup	-	TPU v4	-	-	D+M	JAXFormer
LLaMA [127]	arXiv'23	-	Meta	General	65B	×	350k	1.4T	Clf+Heur+Dedup	2048	80G A100	504h	4.12 Mil	D+M	xFormers
PanGuΣ [130]	arXiv'23	-	Huawei	General	1.085T	×	-	329B	-	512	Ascend 910	2400h	-	D+OP+P+O+R	MindSpore
BloombergGPT [142]	arXiv23	-	Bloomberg	Finance	50B	×	139k	569B	Dedup	512	40G A100	1272h	1.97 Mil	M	PyTorch
Xuan Yuan 2.0 [144]	arXiv23	RAIL-1.0	Du Xiaoman	Finance	176B	✓	-	366B	Filtered	80GB	A100	-	-	P	DS
CodeT5+ [34]	arXiv'23	BSD-3	Salesforce	Coding	16B	✓	110k	51.5B	Dedup	16	40G A100	-	-	-	DS
StarCoder [138]	arXiv'23	OpenRAIL-M	BigCode	Coding	15.5B	✓	250k	1T	Dedup+QF+PF	512	80G A100	624h	1.28 Mil	D+T+P	Megatron-LM
LLaMA-2 [21]	arXiv'23	LLaMA-2.0	Meta	General	70B	✓	500k	2T	Minimal Filtering	-	80G A100	1.7Mh	-	-	-
PaLM-2 [124]	arXiv'23	-	Google	General	-	×	-	-	Ddedup+PF+QF	-	-	-	-	-	-

TABLE IV: Summary of instruction tuned LLMs (>10B). All abbreviations are the same as Table III. Entries in "Data/Tokens" starting with "S-" represents the number of training samples.

Models	Publication Venue	License Type	Model Creators	Purpose	No. of Params	Commercial Use	Pre-trained Models	Steps Trained	Data/ Tokens	No. of Processing Units	Processing Unit Type	Train. Time	Calculated Train. Cost	Train. Parallelism	Library
WebGPT [158]	arXiv'21	-	OpenAI	General	175B	×	GPT-3	-	-	-	-	-	-	-	-
T0 [17]	ICLR'22	Apache-2.0	BigScience	General	11B	✓	T5	-	250B	512	TPU v3	270h	0.48 Mil	-	-
Tk-Instruct [18]	EMNLP'22	MIT	AI2+	General	11B	✓	T5	1000	-	256	TPU v3	4h	0.0036 Mil	-	Google T5
OPT-IML [93]	arXiv'22	-	Meta	General	175B	×	OPT	8k	2B	128	40G A100	-	-	D+T	Megatron
Flan-U-PaLM [16]	ICLR'22	Apache-2.0	Google	General	540B	✓	U-PaLM	30k	-	512	TPU v4	-	-	-	JAX+T5X
mT0 [146]	ACL'23	Apache-2.0	HuggingFace+	General	13B	✓	mT5	-	-	-	-	-	-	-	-
Sparrow [159]	arXiv'22	-	Google	Dialog	70B	×	Chinchilla	-	-	64	TPU v3	-	-	M	-
WizardCoder [156]	arXiv'23	Apache-2.0	HK Bapt.	Coding	15B	×	StarCoder	200	S-78k	-	-	-	-	-	-
Alpaca [150]	Github'23	Apache-2.0	Stanford	General	13B	✓	LLaMA	3-Epoch	S-52k	8	80G A100	3h	600	FSDP	PyTorch
Vicuna [151]	Github'23	Apache-2.0	LMSYS	General	13B	✓	LLaMA	3-Epoch	S-125k	-	-	-	-	FSDP	PyTorch
LIMA [177]	arXiv'23	-	Meta+	General	65B	-	LLaMA	15-Epoch	S-1000	-	-	-	-	-	-
Koala [275]	Github'23	Apache-2.0	UC-Berkley	General	13B	×	LLaMA	2-Epoch	S-472k	8	A100	6h	100	-	JAX/FLAX

multiple tasks, including sentiment analysis, text classification, natural language inference (NLI), question answering (QA), commonsense reasoning (CR), mathematical reasoning (MR), reading comprehension (RC), etc.

Natural Language Generation: This task assesses the language generation capabilities of LLMs by understanding the provided input context. It includes tasks such as summarization, sentence completion, machine translation (MT), dialogue generation, etc.

Numerous datasets are proposed for each task, evaluating LLMs against different characteristics. To provide an overview of evaluation datasets, we briefly discuss a few famous datasets within each category and offer a comprehensive list of datasets in Table IX. Moreover, we show a detailed overview of the training datasets and evaluation tasks and benchmarks used by various pre-trained LLMs in Table X and fine-tuned LLMs in Table XI. We also compare the top-performing LLMs in various NLP tasks in Table XII.

1. Multi-task:

1.1 MMLU [282]: A benchmark that measures the knowledge acquired by models during pretraining and evaluates models in zero-shot and few-shot settings across 57 subjects, testing both world knowledge and problem-solving ability.

1.2 SuperGLUE [2]: A more challenging and diverse successor to the GLUE [284] benchmark, SuperGLUE includes a variety of language understanding tasks, such as question answering, natural language inference, and coreference

TABLE V: Architecture details of LLMs. Here, "PE" is the positional embedding, "nL" is the number of layers, "nH" is the number of attention heads, "HS" is the size of hidden states.

Models	Type	Training Objective	Attention	Vocab	Tokenizer	Norm	PE	Activation	Bias	nL	nH	HS
T5 (11B)	Enc-Dec	Span Corruption	Standard	32k	SentencePiece	Pre-RMS	Relative	ReLU	×	24	128	1024
GPT3 (175B)	Causal-Dec	Next Token	Dense+Sparse	-	-	Layer	Learned	GeLU	✓	96	96	12288
mT5 (13B)	Enc-Dec	Span Corruption	Standard	250k	SentencePiece	Pre-RMS	Relative	ReLU	-	-	-	-
PanGu-α (200B)	Causal-Dec	Next Token	Standard	40k	BPE	Layer	-	-	-	64	128	16384
CPM-2 (198B)	Enc-Dec	Span Corruption	Standard	250k	SentencePiece	Pre-RMS	Relative	ReLU	-	24	64	-
Codex (12B)	Causal-Dec	Next Token	Standard	-	BPE+	Pre-Layer	Learned	GeLU	-	96	96	12288
ERNIE 3.0 (10B)	Causal-Dec	Next Token	Standard	-	WordPiece	Post-Layer	Relative	GeLU	-	48	64	4096
Jurassic-1 (178B)	Causal-Dec	Next Token	Standard	256k	SentencePiece*	Pre-Layer	Learned	GeLU	✓	76	96	13824
HyperCLOVA (82B)	Causal-Dec	Next Token	Dense+Sparse	-	BPE*	Pre-Layer	Learned	GeLU	-	64	80	10240
Yuan 1.0 (245B)	Causal-Dec	Next Token	Standard	-	-	-	-	-	-	76	-	16384
Gopher (280B)	Causal-Dec	Next Token	Standard	32k	SentencePiece	Pre-RMS	Relative	GeLU	✓	80	128	16384
ERNIE 3.0 Titan (260B)	Causal-Dec	Next Token	Standard	-	WordPiece	Post-Layer	Relative	GeLU	-	48	192	12288
GPT-NeoX-20B	Causal-Dec	Next Token	Parallel	50k	BPE	Layer	Rotary	GeLU	✓	44	64	-
OPT (175B)	Causal-Dec	Next Token	Standard	-	BPE	-	-	ReLU	✓	96	96	-
BLOOM (176B)	Causal-Dec	Next Token	Standard	250k	BPE	Layer	ALiBi	GeLU	✓	70	112	14336
Galactica (120B)	Causal-Dec	Next Token	Standard	50k	BPE+custom	Layer	Learned	GeLU	×	96	80	10240
GLaM (1.2T)	MoE-Dec	Next Token	Standard	256k	SentencePiece	Layer	Relative	GeLU	✓	64	128	32768
LaMDA (137B)	Causal-Dec	Next Token	Standard	32k	BPE	Layer	Relative	GeGLU	-	64	128	8192
MT-NLG (530B)	Causal-Dec	Next Token	Standard	50k	BPE	Pre-Layer	Learned	GeLU	✓	105	128	20480
AlphaCode (41B)	Enc-Dec	Next Token	Multi-query	8k	SentencePiece	-	-	-	-	64	128	6144
Chinchilla (70B)	Causal-Dec	Next Token	Standard	32k	SentencePiece-NFKC	Pre-RMS	Relative	GeLU	✓	80	64	8192
PaLM (540B)	Causal-Dec	Next Token	Parallel+Multi-query	256k	SentencePiece	Layer	RoPE	SwiGLU	×	118	48	18432
AlexaTM (20B)	Enc-Dec	Denoising	Standard	150k	SentencePiece	Pre-Layer	Learned	GeLU	✓	78	32	4096
Sparrow (70B)	Causal-Dec	Pref.&Rule RM	-	32k	SentencePiece-NFKC	Pre-RMS	Relative	GeLU	✓	16*	64	8192
U-PaLM (540B)	Non-Causal-Dec	MoD	Parallel+Multi-query	256k	SentencePiece	Layer	RoPE	SwiGLU	×	118	48	18432
UL2 (20B)	Enc-Dec	MoD	Standard	32k	SentencePiece	-	-	-	-	64	16	4096
GLM (130B)	Non-Causal-Dec	AR Blank Infilling	Standard	130k	SentencePiece	Deep	RoPE	GeGLU	✓	70	96	12288
CodeGen (16B)	Causal-Dec	Next Token	Parallel	-	BPE	Layer	RoPE	-	-	34	24	-
LLaMA (65B)	Causal-Dec	Next Token	Standard	32k	BPE	Pre-RMS	RoPE	SwiGLU	-	80	64	8192
PanGu-Σ (1085B)	Causal-Dec	Next Token	Standard	-	BPE	Fused Layer	-	FastGeLU	-	40	40	5120
BloombergGPT (50B)	Causal-Dec	Next Token	Standard	131k	Unigram	Layer	ALiBi	GeLU	✓	70	40	7680
Xuan Yuan 2.0 (176B)	Causal-Dec	Next Token	Self	250k	BPE	Layer	ALiBi	GeLU	✓	70	112	14336
CodeT5+ (16B)	Enc-Dec	SC+NT+Cont.+Match	Standard	-	Code-Specific	-	-	-	-	-	-	-
StarCoder (15.5B)	Causal-Dec	FIM	Multi-query	49k	BPE	-	Learned	-	-	40	48	6144
LLaMA (70B)	Causal-Dec	Next Token	Grouped-query	32k	BPE	Pre-RMS	RoPE	SwiGLUE	-	-	-	-
PaLM-2	-	MoD	Parallel	-	-	-	-	-	-	-	-	-

TABLE VI: Summary of optimization settings used for pre-trained LLMs. The values for weight decay, gradient clipping, and dropout are 0.1, 1.0, and 0.1, respectively, for most of the LLMs.

Models	Batch Size	Sequence Length	LR	Warmup	LR Decay	Optimizers			Precision			Weight Decay	Grad Clip	Dropout
						AdaFactor	Adam	AdamW	FP16	BF16	Mixed			
T5 (11B)	2^{11}	512	0.01	×	inverse square root	✓			-	-	-	-	-	✓
GPT3 (175B)	32K	-	6e-5	✓	cosine		✓		✓			✓	✓	-
mT5 (13B)	1024	1024	0.01	-	inverse square root	✓			-	-	-	-	-	✓
PanGu-α (200B)	-	1024	2e-5	-	-	-	-	-	-	✓	-	-	-	-
CPM-2 (198B)	1024	1024	0.001	-	-	✓			-	-	-	-	-	✓
Codex (12B)	-	-	6e-5	✓	cosine		✓		✓			✓	-	-
ERNIE 3.0 (12B)	6144	512	1e-4	✓	linear		✓		-	-	-	✓	-	-
Jurassic-1 (178B)	3.2M	2048	6e-5	✓	cosine		✓		✓			✓	✓	-
HyperCLOVA (82B)	1024	-	6e-5	-	cosine			✓	-	-	-	✓	-	-
Yuan 1.0 (245B)	<10M	2048	1.6e-4	✓	cosine decay to 10%		✓		-	-	-	✓	-	-
Gopher (280B)	3M	2048	4e-5	✓	cosine decay to 10%		✓			✓		-	✓	-
ERNIE 3.0 Titan (260B)	-	512	1e-4	✓	linear		✓		✓			✓	✓	-
GPT-NeoX-20B	1538	2048	0.97e-5	✓	cosine			✓	✓			✓	✓	×
OPT (175B)	2M	2048	1.2e-4	-	linear			✓	✓			✓	✓	✓
BLOOM (176B)	2048	2048	6e-5	✓	cosine		✓			✓		✓	✓	×
Galactica (120B)	2M	2048	7e-6	✓	linear decay to 10%			✓	-	-	-	✓	✓	✓
GLaM (1.2T)	1M	1024	0.01	-	inverse square root	✓				FP32 + ✓		-	✓	×
LaMDA (137B)	256K	-	-	-	-	-	-	-	-	-	-	-	-	-
MT-NLG (530B)	1920	2048	5e-5	✓	cosine decay to 10%		✓			✓		✓	✓	-
AlphaCode (41B)	2048	1536+768	1e-4	✓	cosine decay to 10%			✓		✓		✓	✓	-
Chinchilla (70B)	1.5M	2048	1e-4	✓	cosine decay to 10%			✓		✓		-	-	-
PaLM (540B)	2048	2048	0.01	-	inverse square root	✓			-	-	-	✓	✓	×
AlexaTM (20B)	2M	1024	1e-4	-	linear decay to 5%		✓			✓		✓	-	✓
U-PaLM (540B)	32	2048	1e-4	-	cosine	✓			-	-	-	-	-	-
UL2 (20B)	1024	1024	-	-	inverse square root	-	-	-	-	-	-	×	-	-
GLM (130B)	4224	2048	8e-5	✓	cosine			✓	✓			✓	✓	✓
CodeGen (16B)	2M	2048	5e-5	✓	cosine		✓		-	-	-	✓	✓	-
LLaMA (65B)	4M Tokens	2048	1.5e-4	✓	cosine decay to 10%			✓	-	-	-	✓	✓	-
PanGu-Σ (1.085T)	512	1024	2e-5	✓	-		✓				✓	-	-	-
BloombergGPT (50B)	2048	2048	6e-5	✓	cosine			✓			✓	✓	✓	×
Xuan Yuan 2.0 (176B)	2048	2048	6e-5	✓	cosine		✓		✓			✓	✓	-
CodeT5+ (16B)	2048	1024	2e-4	-	linear			✓			✓	✓	-	-
StarCoder (15.5B)	512	8k	3e-4	✓	cosine		✓			✓		✓	-	-
LLaMA-2 (70B)	4M Tokens	4k	1.5e-4	✓	cosine			✓		✓		✓	✓	-

TABLE VII: Summary of optimization settings used for instruction-tuned LLMs. Values for gradient clipping and dropout are the same as the pre-trained models, while no model uses weight decay for instruction tuning.

Models	Batch Size	Sequence Length	LR	Warmup	LR_Decay	Optimizers AdaFactor	Adam	AdamW	Grad Clip	Dropout
WebGPT (175B)	BC:512, RM:32	-	6e-5	-	-		✓		-	-
T0 (11B)	1024	1280	1e-3	-	-	✓			-	✓
Tk-Instruct (11B)	1024	-	1e-5	-	constant	-	-	-	-	-
OPT-IML (175B)	128	2048	5e-5	×	linear		✓		✓	✓
Flan-U-PaLM (540B)	32	-	1e-3	-	constant	✓			-	✓
Sparrow (70B)	RM: 8+16, RL:16	-	2e-6	✓	cosine decay to 10%	✓			✓	×
WizardCoder (15B)	512	2048	2e-5	✓	cosine	-	-	-	-	-
Alpaca (13B)	128	512	1e-5	✓	cosine	-	-	✓	✓	×
Vicuna (13B)	128	-2048	2e-5	✓	cosine			✓	-	×
LIMA (65B)	32	2048	1e-5	×	linear			✓	-	✓

TABLE VIII: Details of various well-known pre-training and fine-tuning datasets. Here, alignment means aligning with human preferences.

Dataset	Type	Size/Samples	Tasks	Source	Creation	Comments
C4 [10]	Pretrain	806GB	-	Common Crawl	Automated	A clean, multilingual dataset with billions of tokens
mC4 [11]	Pretrain	38.49TB	-	Common Crawl	Automated	A multilingual extension of the C4 dataset, mC4 identifies over 100 languages using cld3 from 71 monthly web scrapes of Common Crawl.
PILE [276]	Pretrain	825GB	-	Common Crawl, PubMed Central, OpenWebText2, ArXiv, GitHub, Books3, and others	Automated	A massive dataset comprised of 22 constituent sub-datasets
ROOTs [277]	Pretrain	1.61TB	-	498 Hugging Face datasets	Automated	46 natural and 13 programming languages
MassiveText [114]	Pretrain	10.5TB	-	MassiveWeb, Books, News, Wikipedia, Github, C4	Automated	99% of the data is in English
Wikipedia [278]	Pretrain	-	-	Wikipedia	Automated	Dump of wikipedia
RedPajama [279]	Pretrain	5TB	-	CommonCrawl, C4, Wikipedia, Github, Books, StackExchange	Automated	Open-source replica of LLaMA dataset
PushShift.io Reddit	Pretrain	21.1GB	-	Reddit	Automated	Submissions and comments on Reddit from 2005 to 2019
BigPython [131]	Pretrain	5.5TB	Coding	GitHub	Automated	-
Pool of Prompt (P3) [17]	Instructions	12M	62	PromptSource	Manual	A Subset of PromptSource, created from 177 datasets including summarization, QA, classification, etc.
xP3 [146]	Instructions	81M	71	P3+Multilingual datasets	Manual	Extending P3 to total 46 languages
Super-NaturalInstructions (SNI) [18]	Instructions	12.4M	1616	Multiple datasets	Manual	Extending P3 with additional multi-lingual datasets, total 46 languages
Flan [16]	Instructions	15M	1836	Muffin+T0-SF+NIV2	Manual	Total 60 languages
OPT-IML [93]	Instructions	18.1M	1667	-	Manual	-
Self-Instruct [19]	Instructions	82k	175	-	Automated	Generated 52k instructions with 82k samples from 175 seed tasks using GPT-3
Alpaca [150]	Instructions	52k	-	-	Automated	Employed self-instruct method to generate data from text-davinci-003
Vicuna [151]	Instructions	125k	-	ShareGPT	Automated	Conversations shared by users on ShareGPT using public APIs
LLaMA-GPT-4 [152]	Instructions	52k	-	Alpaca	Automated	Recreated Alpaca dataset with GPT-4 in English and Chinese
Unnatural Instructions [280]	Instructions	68k	-	15-Seeds (SNI)	Automated	-
LIMA [177]	Instructions	1k	-	Multiple datasets	Manual	Carefully created samples to test performance with fine-tuning on less data
Anthropic-HH-RLHF [281]	Alignment	142k	-	-	Manual	
Anthropic-HH-RLHF-2 [170]	Alignment	39k	-	-	Manual	

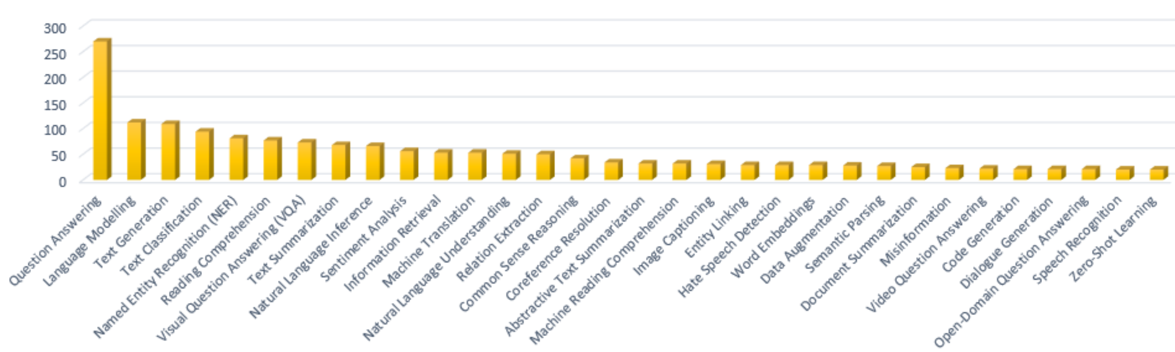

Fig. 14: A distribution of datasets proposed for different NLP tasks. We include only the tasks for which at least 20 datasets have already been proposed.

TABLE IX: Categorized evaluation datasets used in evaluating LLMs.

Type	Datasets/Benchmarks
Multi-Task	MMLU [282], SuperGLUE [2], BIG-bench [283], GLUE [284], BBH [283], CUGE [285], ZeroCLUE [286], FewCLUE [287], Blended Skill Talk [288], HELM [289], KLUE-STS [290]
Language Understanding	CoQA [291], WiC [292], Wikitext103 [293], PG19 [294], LCQMC [295], QQP [296], WinoGender [297], CB [298], FinRE [299], SanWen [300], AFQMC [286], BQ Corpus [301], CNSS [302], CKBQA 13 [303], CLUENER [286], Weibo [304], AQuA [305], OntoNotes [306], HeadQA [307], Twitter Dataset [308]
Story Cloze and Sentence Completion	StoryCloze [309], LAMBADA [310], LCSTS [311], AdGen [312], E2E [313], CHID [314], CHID-FC [287]
Physical Knowledge and World Understanding	PIQA [315], TriviaQA [316], ARC [317], ARC-Easy [317], ARC-Challenge [317], PROST [318], OpenBookQA [319], WebNLG [320], DogWhistle Insider & Outsider [321]
Contextual Language Understanding	RACE [322], RACE-Middle [322], RACE-High [322], QuAC [323], StrategyQA [324], Quiz Bowl [325], cMedQA [326], cMedQA2 [327], MATINF-QA [328]
Commonsense Reasoning	WinoGrande [329], HellaSwag [330], COPA [331], WSC [332], CSQA [333], SIQA [334], C^3 [335], CLUEWSC2020 [286], CLUEWSC [286], CLUEWSC-FC [287], ReCoRD [336]
Reading Comprehension	SQuAD [337], BoolQ [338], SQUADv2 [339], DROP [340], RTE [341], WebQA [342], CMRC2017 [343], CMRC2018 [344], CMRC2019 [345], COTE-BD [346], COTE-DP [346], COTE-MFW [346], MultiRC [347], Natural Questions [348], CNSE [302], DRCD [349], DuReader [350], $\text{Dureader}_{\text{robust}}$ [351], DuReader-QG [350], SciQ [352], Sogou-log [353], $\text{Dureader}_{\text{robust}}$-QG [351], QA4MRE [354], KorQuAD 1.0 [355], CAIL2018-Task1 & Task2 [356]
Mathematical Reasoning	MATH [357], Math23k [358], GSM8K [359], MathQA [360], MGSM [361], MultiArith [362], ASDiv [363], MAWPS [364], SVAMP [365]
Problem Solving	HumanEval [366], DS-1000 [367], MBPP [368], APPS [357], CodeContests [133]
Natural Language Inference & Logical Reasoning	ANLI [369], MNLI-m [370], MNLI-mm [370],QNLI [337], WNLI [332], OCNLI [286], CMNLI [286], ANLI R1 [369], ANLI R2 [369], ANLI R3 [369], HANS [371], OCNLI-FC [287], LogiQA [372], StrategyQA [324]
Cross-Lingual Understanding	MLQA [373], XNLI [374], PAWS-X [375], XSum [376], XCOPA [377], XWinograd [378], TyDiQA-GoldP [379], MLSum [380]
Truthfulness and Fact Checking	TruthfulQA [381], MultiFC [382], Fact Checking on Fever [383]
Biases and Ethics in AI	ETHOS [384], StereoSet [385], BBQ [386], Winobias [387], CrowS-Pairs [388]
Toxicity	RealToxicityPrompts [389], CivilComments toxicity classification [390]
Language Translation	WMT [391], WMT20 [392], WMT20-enzh [392], EPRSTMT [287], CCPM [393]
Scientific Knowledge	AminoProbe [139], BioLAMA [139], Chemical Reactions [139], Galaxy Clusters [139], Mineral Groups [139]
Dialogue	Wizard of Wikipedia [394], Empathetic Dialogues [395], DPC-generated [122] dialogues, ConvAI2 [396], KdConv [397]
Topic Classification	TNEWS-FC [287], YNAT [290], KLUE-TC [290], CSL [286], CSL-FC [287], IFLYTEK [398]

resolution. It is designed to provide a rigorous test of language understanding and requires significant progress in areas like sample-efficient, transfer, multitasking, and unsupervised or self-supervised learning.

1.3 BIG-bench [283]: The BIG-bench (Behavior of Intelligent Generative Models Benchmark) is a large-scale benchmark designed to test the abilities of LLMs across a wide range of tasks, including reasoning, creativity, ethics, and understanding of specific domains.

1.4 GLUE [284]: The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating, and analyzing natural language understanding systems. It includes a variety of tasks that test a wide range of linguistic phenomena, making it a comprehensive tool for evaluating language understanding in AI.

2. Language Understanding:

2.1 WinoGrande [329]: A large-scale dataset inspired by the original Winograd [332] Schema Challenge tests models on their ability to resolve pronoun ambiguity and encourages the development of models that understand the broad context in natural language text.

2.2 CoQA [291]: A conversational question-answering dataset, CoQA challenges models with questions that rely on conversation history and require free-form text answers. Its diverse content from seven domains makes it a rigorous test for models' ability to handle a wide range of topics and conversational contexts.

2.3 WiC [292]: This dataset assesses a model's ability to discern word meanings based on context, aiding in tasks related to Word Sense Disambiguation.

2.4 Wikitext103 [293]: With over 100 million tokens from Wikipedia's top articles, this dataset is a rich resource for tasks that require understanding long-term dependencies, such as language modeling and translation.

2.5 PG19 [294]: This is a digital library of diverse books from Project Gutenberg. It's specifically designed to facilitate research in unsupervised learning and language modeling, with a special focus on long-form content.

2.6 C4 [10]: A clean, multilingual dataset, C4 offers billions of tokens from web-crawled data. It's a comprehensive resource for training advanced Transformer models on various languages.

2.7 LCQMC [295]: The Large-scale Chinese Question Matching Corpus (LCQMC) is a dataset for evaluating the performance of models in semantic matching tasks. It contains pairs of questions in Chinese and their matching status, making it a valuable resource for research in Chinese language understanding.

3. Story Cloze and Sentence Completion:

3.1 StoryCloze [309]: It introduces a new "StoryCloze Test", a commonsense reasoning framework for evaluating story understanding, generation, and script learning. It considers a model's ability to understand and generate coherent and sensible stories.

TABLE X: An illustration of training datasets and evaluation tasks employed by pre-trained LLMs. Here, "QA" is question-answering, "Clf" is classification, "NLI" is natural language inference, "MT" is machine translation, "RC" is reading comprehension, "CR" is commonsense reasoning, "MR" is mathematical reasoning, "Mem." is memorization.

Models	**Training Dataset**	**Benchmark**			**QA**	**Clf**	**NLI**	**MT**	**Cloze/ Completion**	**RC**	**CR**	**MR**	**Coding**	**Truthful/ Bias/ Toxicity/ Mem.**
		BIG-bench	MMLU	Super GLUE										
T5	C4 [10]			✓	✓		✓	✓	✓	✓	✓	✓		
GPT-3	Common Crawl, WebText, Books Corpora, Wikipedia			✓	✓			✓	✓	✓				✓
mT5	mC4 [11]				✓		✓	✓						
PanGu-α	1.1TB Chinese Text Corpus				✓		✓		✓	✓	✓			
CPM-2	WuDaoCorpus [106]									✓		✓		
Codex	54 million public repositories from Github												✓	
ERNIE-3.0	Chinese text corpora, Baidu Search, Web text, QA-long, QA-short, Poetry and Couplet Domain-specific data from medical, law, and financial area Baidu knowledge graph with more than 50 million facts			✓	✓	✓	✓	✓	✓	✓		✓		
Jurassic-1	Wikipedia, OWT, Books, C4, Pile [276], arXiv, GitHub				✓		✓		✓	✓				
HyperCLOVA	Korean blogs, Community sites, News, KiN Korean Wikipedia, Wikipedia (English and Japanese), Modu-Corpus: Messenger, News, Spoken and written language corpus, Web corpus							✓						
Yuan 1.0	Common Crawl, SogouT, Sogou News, Baidu Baike, Wikipedia, Books				✓	✓	✓			✓				
Gopher	subsets of MassiveWeb Books, C4, News, GitHub and Wikipedia samples from MassiveText	✓	✓	✓	✓						✓	✓		✓
ERNIE-3.0 TITAN	Same as ERNIE 3.0 and ERNIE 3.0 adversarial dataset, ERNIE 3.0 controllable dataset				✓	✓	✓		✓	✓				
GPT-NeoX-20B	Pile [276]			✓	✓		✓		✓		✓	✓		
OPT	RoBERTa [399], Pile [276], PushShift.io Reddit [400]				✓	✓					✓			✓
BLOOM	ROOTs [13]			✓			✓	✓	✓				✓	✓
Galactica	arXiv, PMC, Semantic Scholar, Wikipedia, StackExchange, LibreText, Open Textbooks, RefSeq Genome, OEIS, LIPID MAPS, NASAExoplanet, Common Crawl, ScientificCC, AcademicCC, GitHub repositories Khan Problems, GSM8K, OneSmallStep	✓	✓		✓							✓		✓
GLaM	Filtered Webpages, Social media conversations Wikipedia, Forums, Books, News				✓		✓		✓	✓	✓			
LaMDA	Infiniset : Public documents, Dialogs, Utterances													✓
MT-NLG	Two snapshots of Common Crawl and Books3, OpenWebText2, Stack Exchange, PubMed Abstracts, Wikipedia, PG-19 [242], BookCorpus2, NIH ExPorter, Pile, CC-Stories, RealNews						✓		✓	✓	✓			✓
AlphaCode	Selected GitHub repositories, CodeContests: Codeforces, Description2Code, CodeNet												✓	
Chinchilla	MassiveWeb, MassiveText Books, C4, News, GitHub, Wikipedia	✓	✓		✓					✓	✓			✓
PaLM	webpages, books, Wikipedia, news, articles, source code, social media conversations	✓			✓			✓			✓		✓	✓
AlexaTM	Wikipedia, mC4			✓			✓	✓			✓			✓
U-PaLM	Same as PaLM	✓		✓	✓		✓		✓	✓	✓			
UL2	-			✓	✓	✓	✓					✓		✓
GLM-130B	-	✓	✓						✓					
CodeGen	Pile, BigQuery, BigPython												✓	
LLaMA	CommonCrawl, C4, Github, Wikipedia, Books, arXiv, StackExchange		✓		✓					✓	✓	✓	✓	✓
PanGu-Σ	WuDaoCorpora, CLUE, Pile, C4, Python code				✓	✓	✓	✓	✓				✓	
BloombergGPT	inPile, Pile, C4, Wikipedia	✓	✓				✓		✓	✓	✓			✓
CodeT5+	CodeSearchNet, Github Code											✓	✓	
StarCoder	The Stack v1.2		✓									✓	✓	✓
LLaMA-2	✓	✓		✓						✓	✓	✓	✓	
PaLM-2	Web documents, Code, Books, Maths, Conversation			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

TABLE XI: An illustration of training datasets and evaluation benchmarks used in fine-tuned LLMs. "SNI" is a short of Super-NaturalInsturctions.

Models	Training Dataset	BIG-bench	MMLU	BBH	RAFT	FLAN	SNI	PromptSource	TyDiQA	HumanEval	MBPP	Truthful/ Bias/ Toxicity
T0	Pool of Prompts	✓										
WebGPT	ELI5 [401], ELI5 fact-check [158], TriviaQA [316], ARC-Challenge [317], ARC-Easy [317], Hand-written data, Demonstrations of humans, Comparisons between model-generated answers											✓
Tk-INSTRUCT	SNI [18]						✓					
mT0	xP3 [146]											
OPT-IML	PromptSource [17], FLAN [16], SNI [402], UnifiedSKG [403], CrossFit [404], ExMix [405], T5 [10], Reasoning		✓	✓	✓	✓	✓	✓				
Flan	Muffin, T0-SF, NIv2, CoT		✓	✓					✓			
WizardCoder	Code Alpaca									✓	✓	

3.2 LAMBADA [310]: This dataset evaluates contextual text understanding through a word prediction task. Models must predict the last word of a passage, which is easy for humans when given the whole passage, but not when given only the last sentence.

4. Physical Knowledge and World Understanding:

4.1 PIQA [315]: A dataset that probes the physical knowledge of models, aiming to understand how well they are learning about the real world.

4.2 TriviaQA [316]: A dataset that tests models on reading comprehension and open domain question answering (QA) tasks, with a focus on Information Retrieval (IR)-style QA.

4.3 ARC [317]: A larger version of the ARC-Challenge, this dataset contains both easy and challenging grade-school level, multiple-choice science questions. It's a comprehensive test of a model's ability to understand and answer complex questions.

4.4 ARC-Easy [317]: A subset of the ARC dataset, ARC-Easy, contains questions that are answered correctly by either a retrieval-based algorithm or a word co-occurrence algorithm. It's a great starting point for models beginning to explore advanced question-answering.

4.5 ARC-Challenge [317]: A rigorous question-answering dataset, ARC-Challenge includes complex, grade-school level questions that demand reasoning beyond simple retrieval, testing the true comprehension capabilities of models.

5. Contextual Language Understanding:

5.1 RACE [322]: The RACE is a reading comprehension dataset collected from English examinations in China, which benchmarks AI models for understanding and answering questions on long and complex passages, simulating the challenge of a real-world examination.

5.2 RACE-Middle [322]: Another subset of the RACE [322] dataset, RACE-Middle, contains middle school-level English exam questions. It offers a slightly less challenging but academically oriented evaluation of a model's comprehension skills.

5.3 RACE-High [322]: A subset of the RACE [322] dataset, RACE-High consists of high school-level English exam questions. It is designed to evaluate the comprehension ability of models in a more academic and challenging context.

5.4 QuAC [323]: This dataset simulates an information-seeking dialog between students and teachers using hidden Wikipedia text. It introduces unique challenges not found in machine comprehension datasets, making it a valuable resource for advancing dialog systems.

6. Commonsense Reasoning:

6.1 HellaSwag [330]: A dataset that challenges models to pick the best ending to a context uses Adversarial Filtering to create a 'Goldilocks' zone of complexity, where generated text is absurd to humans but often misclassified by models.

6.2 COPA [377]: This dataset evaluates a model's progress in open-domain commonsense causal reasoning. Each question comprises a premise and two alternatives, and the model must select the more plausible alternative, testing a model's ability to understand and reason about cause and effect.

6.3 WSC [332]: The Winograd Schema Challenge (WSC) is a reading comprehension task in which a system must resolve references in a text, often requiring world knowledge and reasoning about the text.

6.4 CSQA [333]: The CommonsenseQA is a question-answering dataset that requires commonsense knowledge to answer the ability of AI models to understand and answer questions that require commonsense reasoning.

7. Reading Comprehension:

7.1 BoolQ [338]: A dataset derived from Google search queries, BoolQ challenges models to answer binary (yes/no) questions. The questions are naturally occurring and are paired with a paragraph from a Wikipedia article containing the answer. It's a test of reading comprehension and reasoning.

7.2 SQUADv2 [339]: The Stanford Question Answering Dataset (SQuAD) [337] is a collection of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text from the corresponding reading passage. SQuADv2 combines the original SQuAD1.1 dataset with over 50,000 unanswerable questions. The aim is to evaluate a model's ability to understand and answer questions based on a given context and to determine when a question is unanswerable.

7.3 DROP [340]: DROP, or Discrete Reasoning Over the content of Paragraphs, is designed to test a model's ability to understand a wide variety of reading phenomena. It encourages comprehensive and reliable evaluation of reading comprehension capabilities.

7.4 RTE [341]: The Recognizing Textual Entailment (RTE) datasets come from a series of annual competitions on textual entailment, predicting whether a given sentence logically follows from another and evaluating a model's understanding of logical relationships in a text.

7.5 WebQA [342]: A dataset for open-domain question answering, WebQA offers a large collection of web-based question-answer pairs. It is designed to assess the ability of AI models to understand and answer questions based on web content.

7.6 CMRC2018 [344]: This dataset is a test of Chinese language models' ability to reason comprehensively and is designed with a challenging span-extraction format that pushes the boundaries of machine performance.

8. Mathematical Reasoning:

8.1 MATH [357]: This dataset is a platform for evaluating the mathematical problem-solving abilities of AI models. It contains a diverse set of math problems, ranging from arithmetic to calculus, and is designed to test the model's ability to understand and solve complex mathematical problems.

8.2 Math23k [358]: This one challenges a model's ability to understand and solve mathematical word problems. It contains 23,000 Chinese arithmetic word problems that require models to perform reasoning and computation based on the problem description.

8.3 GSM8K [359]: A dataset of diverse grade school math word problems, testing a model's ability to perform multi-step mathematical reasoning.

9. Problem Solving and Logical Reasoning:

9.1 ANLI [369]: A large-scale dataset designed to test the robustness of machine learning models in Natural Language Inference (NLI) is created through an iterative, adversarial process where humans try to generate examples that models cannot correctly classify.

9.2 HumanEval [366]: A dataset for the problem-solving ability of AI models, which includes a diverse set of tasks that require various cognitive abilities, makes it a comprehensive tool for assessing general intelligence in AI.

9.3 StrategyQA [324]: A question-answering dataset that requires reasoning over multiple pieces of evidence to evaluate the strategic reasoning ability of AI models, pushing the boundaries of what machines can understand and answer.

10. Cross-Lingual Understanding:

10.1 XNLI [374]: A cross-lingual benchmark, XNLI extends the MultiNLI [406] corpus to 15 languages, including low-resource ones like Urdu. It tests models on cross-lingual sentence understanding, with 112,500 annotated pairs across three categories: entailment, contradiction, and neutral.

10.2 PAWS-X [375]: PAWS-X, or Cross-lingual Paraphrase Adversaries from Word Scrambling, is a multilingual version of the PAWS [407] dataset for paraphrase identification. It includes examples in seven languages and is designed to evaluate the performance of cross-lingual paraphrase identification models.

11. Truthfulness:

11.1 Truthful-QA [381]: A unique benchmark that measures a language model's truthfulness when generating answers. The dataset includes questions across various categories like health, law, and politics, some designed to test the model against common human misconceptions.

12. Biases and Ethics in AI:

12.1 ETHOS [384]: ETHOS is a hate speech detection dataset built from YouTube and Reddit comments. It's a tool in the fight against online hate speech, offering binary and multi-label variants for robust content moderation.

12.2 StereoSet [385]: StereoSet is a comprehensive dataset designed to measure and evaluate the presence of stereotypical biases in language models. It focuses on four key domains: gender, profession, race, and religion. Contrasting stereotypical bias against language modeling ability provides a valuable tool for understanding and mitigating biases in large language models.

VII. Summary and Discussion

A. Architecture

Due to the gigantic scale of LLMs, minor changes in architecture and training strategies have a big impact on performance and stability. Here, we summarize key architectural modules used in various LLMs, leading to better performance, reduced training time and memory, and better training stability.

Layer Normalization is found to have a significant effect on the performance and training stability of LLMs. Pre-norm, that is normalizing inputs rather than outputs, is more common among LLMs stabilizing the training [6], [127], [105]. BLOOM [13] and AlexaTM [123] utilize an additional layer normalization before embedding layer to stabilize the training of large-scale models, while the model's zero-shot generalization ability can be negatively impacted [13]. However, another study [33] finds that pre-norm degrades fine-tuned model performance as compared to post-norm, and there are no stability benefits of pre-norm beyond the 100B scale. Therefore, GLM-130B [33] used deep-norm which is a variant of post-norm for better downstream task performance after fine-tuning.

Positional Encoding effect performance and training stability of LLMs like other building blocks of a model. BLOOM [13] finds ALiBi outperforming learned and rotary positional encodings. Contrary to this, GLM-130B [33] identifies rotary positional encoding better than ALiBi. So, there is no conclusion in literature about the positional encodings yet.

Parallel Attention where attention and feed-forward layers are parallel to each other rather than sequential in transformer block has shown to reduce training time by 15%. There is no evidence of performance drop due to this change in literature and used by the models PaLM [15], GPT-NeoX [116], and CodeGen [131].

Multi-Query Attention has shared key and value attention heads in a transformer block while query attention heads are projected as usual. This reduces memory usage and speeds up sampling in autoregressive decoding. No performance degradation has been observed with this change and makes the training efficient allowing larger batch sizes. Multi-query attention is used in [15], [133].

Mixture of Experts allows easily scaling model to trillion of parameters [130], [119]. Only a few experts are activated

TABLE XII: Performance comparison of top performing LLMs across various NLU and NLG tasks. Here, "N-Shots" indicate the number of example prompts provided to the model during the evaluation, representing its capability in few-shot or zero-shot learning settings, "f" represents the fine-tuned version, and "B" represents the benchmark.

Task	Dataset/Benchmark	Model	Model Size	N-Shots	Score
Multi-Task	BIG-bench (B)	Chinchilla	70B	5-shot	65.1
		Gopher	280B	5-shot	53.97
		PaLM	540B	5-shot	53.7
	MMLU (B)	GPT-4	-	5-shot	86.4
		Flan-PaLM-2$_{(f)}$	Large	5-shot	81.2
		PaLM-2	Large	5-shot	78.3
Language Understanding	SuperGLUE (B)	ERNIE 3.0	12B	-	90.6
		PaLM$_{(f)}$	540B	-	90.4
		T5	11B	-	88.9
Story Comprehension and Generation	HellaSwag	GPT-4	-	10-shot	95.3
		PaLM-2	Large	one shot	86.8
		LLaMA-2	70B	zero shot	85.3
	StoryCloze	GPT3	175B	few shot	87.7
		PaLM-2	Large	one shot	87.4
		OPT	175B	-	79.82
Physical Knowledge and World Understanding	PIQA	PaLM-2	Large	one shot	85.0
		LLaMa	65B	zero shot	82.8
		MT-NLG	530B	zero shot	81.99
	TriviaQA	PaLM-2	Large	one shot	86.1
		LLaMA-2	70B	one shot	85.0
		PaLM	540B	one shot	81.4
Contextual Language Understanding	LAMBADA	PaLM	540B	few shot	89.7
		MT-NLG	530B	few shot	87.15
		PaLM-2	Large	one shot	86.9
Commonsense Reasoning	WinoGrande	GPT-4	-	5-shot	87.5
		PaLM-2	Large	one shot	83.0
		PaLM	540B	zero shot	81.1
	SIQA	LLaMA	65B	zero shot	52.3
		Chinchilla	70B	zero shot	51.3
		Gopher	280B	zero shot	50.6
Reading Comprehension	BoolQ	PaLM$_{(f)}$	540B	-	92.2
		T5	11B	-	91.2
		PaLM-2	Large	one shot	90.9
Truthfulness	Truthful-QA	LLaMA	65B	-	57

during the computation making them compute-efficient. The performance of MoE models is better than the dense models for the same amount of data and requires less computation during fine-tuning to achieve performance similar to the dense models as discussed in [119]. MoE architectures are less prone to catastrophic forgetting, therefore are more suited for continual learning [130]. Extracting smaller sub-models for downstream tasks is possible without losing any performance, making MoE architecture hardware-friendly [130].

Sparse vs Dense Activated GPT-3 [6] uses sparse transformers [63] whereas GLaM [119] and PanGu-$\sum$ [130] use MoE [120] architecture to lower computational costs and increase the model size and capacity. According to the literature, sparse modules do not degrade the model's performance [63]. However, more experiments are required to verify this statement.

B. Training Strategies

Training models at a huge scale require some tricks to reduce training costs, avoid loss divergence and achieve better performance. We summarize and discuss some of these key tricks used in different LLMs.

Mixed Precision is a famous method for LLMs to reduce memory usage and improve training efficiency. In mixed precision, forward and backward passes are performed in FP16 format whereas optimizer states and master weights are kept in FP32 format [408]. A drawback associated with this format change is training instability due to a smaller value range resulting in loss spikes [33]. An alternative to FP16 is BF16 which has a comparatively larger range and performs some precision-sensitive operations like gradient accumulation and softmax in FP32 [13]. BF16 has better performance and training stability but uses more memory and is supported on specific hardware, for example, A100 GPUs. Therefore, its adoption in LLMs is limited.

Training Instability is a common issue in LLMs where loss divergence or spiking is observed multiple times during training. This happens in the presence of gradient clipping [15]. To mitigate this problem, many approaches suggest restarting training from an earlier checkpoint [15], [33], [119], skipping

200-500 earlier data batches at the point of divergence in [15] and re-shuffling batches in [119]. The embedding layer gradient shrink proves to further stabilize the training as its gradient norm is significantly larger than the other layers [33]. Another suggestion to improve training stability for larger models is not to use **biases** in dense and norm layers as in [15].

Weight Initialization plays a significant role in model convergence and training stability. GPT-NeoX [116] initializes feed-forward layers before residuals with $\frac{2}{L\sqrt{d}}$ as in [145] and other layers with small initialization scheme [409]. This avoids activations growing exponentially with the increasing depth. MT-NLG [115] found higher variance for weight initialization leads to unstable training, hence validating small initialization scheme [409]. Various models perform random weight initialization which can cause bad initialization, Galactica [139] suggests a longer warmup to negate the effect.

Learning Rate is important for stable training. It is suggested to use a lower value [13], [15], [125] with warmup and decay (cosine or linear). Usually, the learning rate is within the range $1e^{-4}$ to $8e^{-4}$. Moreover, MT-NLG (530B) [115] and GPT-NeoX (20B) [116] suggest interpolating learning rates based on the model size using the GPT-3 [6] models ranging between 13B and 175B. This avoids tuning the learning rate hyperparameter.

Training Parallelism 3D parallelism, a combination of data, pipeline and tensor parallelism, is the most utilized training parallelism approach in LLMs [33], [15], [14], [13], [115], [113], [110]. In addition to the 3D parallelism, BLOOM [13] uses zero optimizer [37] to shard optimizer states. PanGu-α [105] and PanGu-Σ [130] go beyond the 3D parallelism and apply 5D parallelism which additionally contains optimizer parallelism and rematerialization.

Mode Switching adds task-related tokens at the beginning of the text during training. These tokens refer to the natural language understanding and natural language generation tasks which are shown to improve the downstream task performance in [89], [125], [123]. During fine-tuning and inference, tokens are appended based on the downstream tasks.

Controllable Text Generation Generating credible and controlled text from a pre-trained model is challenging. GPT-3 [6] and other LLMs use in-context learning to control generated text. While in-context learning helps in controlling the generated text, ERNIE 3.0 Titan [35] suggests using adversarial loss to rank its generated text for credibility and soft prompts such as genre, topic, keywords, sentiment, and length for better control on generated text.

C. Pre-Training vs Instruction Tuning

While pre-training is important for the generalization of LLMs, instruction-tuning improves the performance of LLMs further and makes them useable. Therefore, it is suggested to perform instruction fine-tuning of pre-trained LLMs to use them effectively [16], [18], [20], [93], [158].

D. Supervised Models vs Generalized Models

Although generalized models are capable of performing diverse tasks with good performance they have not yet outperformed models trained in supervised settings. The supervised trained models are still state-of-the-art in various NLP tasks by a large margin as shown in [6], [15], [18].

E. Zero-Shot vs Few-Shot

LLMs perform well in zero-shot and few-shot settings. But the performance difference between zero-shot and few-shot is large for pre-trained models [6], [15], naming LLMs as meta-learners [6]. LLMs zero-shot evaluations underperform unsupervised methods in neural machine translation [6]. The literature shows pre-training is not enough for good zero-shot performance [15], [16]. To improve the zero-shot performance the literature suggests using instruction fine-tuning that improves the zero-shot performance significantly and outperforms baselines. Instruction fine-tuning has also been shown to improve zero-shot generalization to unseen tasks. Another model Flan-PaLM [16] unlocks zero-shot reasoning with CoT training.

F. Encoder vs Decoder vs Encoder-Decoder

Traditionally, these architectures perform well for different tasks, for example, encoder-only for NLU tasks, decoder-only for NLG, and encoder-decoder for sequence2sequence modeling. Encoder-only models are famous for smaller models such as Bert [7], RoBERTa [399], etc, whereas LLMs are either decoder-only [6], [116], [13] or encoder-decoder [10], [11], [123]. While decoder-only models are good at NLG tasks, various LLMs, PaLM [15], OPT [14], GPT-3 [6], BLOOM [13], LLaMA [148], are decoder-only models with significant performance gains on both NLU and NLG tasks. In contradiction to this, T5 [10] and UL2 [89] identify encoder-decoder models out-performing decoder-only models. In another study, PaLM [15] finds increasing the size of decoder-only models can reduce the performance gap between decoder-only and encoder-decoder architectures.

Although decoder-only architectures have become a trend for LLMs, many recently proposed approaches [89], [123] use mode-switching tokens in text with encoder-decoder architectures to enable task-specific modes. Similarly, CodeT5+ [34] uses an encoder-decoder architecture with multiple training objectives for different tasks, activating the encoder, decoder, or both according to the tasks. These variations in architecture and training objectives allow a model to perform well in different settings. Because of this dynamic configuration, the future of LLMs can be attributed to encoder-decoder architectures.

VIII. Challenges and Future Directions

LLMs such as GPT-4 and its predecessors have significantly advanced natural language processing. Nevertheless, they also bring along a set of challenges. The computational cost, adversarial robustness, and interpretability are among the technical challenges that are intrinsic to these models. Furthermore, as these models are scaled up to handle more complex tasks or to operate in more complex or dynamic environments, new challenges in scalability, privacy, and real-time processing emerge. On the frontier of foundational research, integrating multi-modality and the effectiveness of transfer learning are being keenly explored. Additionally, the continuous learning aspect of these models, which aims to have models that can adapt to new information over time, presents a fresh set of challenges. These challenges not only underscore the technical intricacies involved but also highlight the broader impact and the future trajectory of LLMs in real-world applications. The following sections delve into these challenges, shedding light on the ongoing and potential efforts to address them.

Computational Cost: Training LLMs requires extensive computational resources, which increases production costs and raises environmental concerns due to substantial energy consumption during large-scale training. Improved performance occurs as computational resources increase, but the rate of improvement gradually decreases when both the model and dataset size remain fixed, following the power law of diminishing returns [410].

Bias and Fairness: LLMs can inherit and amplify societal biases in their training data. These biases can manifest in the model's outputs, leading to potential ethical and fairness issues [411].

Overfitting: Although LLMs possess substantial learning capabilities, they are susceptible to overfitting noisy and peculiar patterns within their extensive training data. Consequently, this may cause them to generate illogical responses [412]. The debate about Memorization vs. Generalization in LLMs is about finding the right balance. Memorization allows the model to remember specific details from its training data, ensuring it can provide accurate answers to precise questions. However, generalization enables the model to make inferences and produce responses for inputs it hasn't seen before, which is essential for handling various real-world tasks. Striking the right balance is the challenge: too much memorization can lead to overfitting, making the model inflexible and struggling with new inputs [413].

Economic and Research Inequality: The high cost of training and deploying LLMs may make their development concentrated within well-funded organizations, potentially worsening economic and research inequalities in AI [414].

Reasoning and Planning: Some reasoning and planning tasks, even as seemingly simple as common-sense planning, which humans find easy, remain well beyond the current capabilities of LLMs evaluated using an assessment framework. This isn't entirely unexpected, considering that LLMs primarily generate text completions based on likelihood and offer no solid guarantees in terms of reasoning abilities [415].

Hallucinations: LLMs exhibit "hallucinations," where they generate responses that, while sounding plausible, are incorrect or don't align with the provided information [416]. The hallucination can be categorized into three categories.

- Input-conflicting hallucination, wherein LLMs produce content that diverges from the input given by users.
- Context-conflicting hallucination, where LLMs generate content that contradicts information they have generated earlier.
- Fact-conflicting hallucination involves LLM's generation of content that does not align with established world knowledge.

Prompt Engineering: Prompts serve as inputs to LLMs, and their syntax and semantics play a crucial role in determining the model's output. The prompt variations, sometimes counter-intuitive to humans, can result in significant changes in model output and are addressed through prompt engineering, which involves designing natural language queries to guide LLMs responses effectively [417], [32].

Limited Knowledge: Information acquired during pretraining is limited and may become obsolete after some time. Re-training the model using updated data is costly. To generate factually accurate responses people use retrieval augmentation pipeline [244]. However, pre-trained models are not trained with retrieval augmentation generation (RAG) [6], [21], hence, adapting the training pipeline is necessary [239], [25].

Safety and Controllability: Using LLMs comes with the risk of generating harmful, misleading, or inappropriate content, whether by accident or when given specific prompts. Ensuring these models are safely utilized is a significant concern [418].

Multi-Modality: Multi-modal learning, where LLMs are trained on diverse data like text, images, and videos, aims to create models with richer understanding but faces challenges in data alignment, fusion strategies, and higher computational demands.

Catastrophic Forgetting: LLMs are often pre-trained on large datasets and then fine-tuned on domain-specific data, reducing training resources but facing issues like domain adaptation and catastrophic forgetting, which hinders the retention of original knowledge when learning new tasks.

Adversarial Robustness: Large Language Models (LLMs) have shown great capabilities in various tasks but are vulnerable to adversarial attacks, where slight, deliberate input alterations can mislead them. Especially with models like BERT, adversarial fine-tuning can enhance robustness, although it sometimes compromises generalization [419]. As LLMs integrate more into complex systems, examining their security properties becomes crucial, given the emerging field of adversarial attacks on LLMs within trustworthy ML [420]. This vulnerability is notable in safety-critical domains, necessitating robust adversarial evaluation tools to ensure LLM reliability [421].

Interpretability and Explainability: The "black-box" nature of LLMs poses challenges in understanding their decision-making, which is crucial for broader acceptance and trust, especially in sensitive domains. Despite their advanced capabilities, the lack of insight into their operation limits their effectiveness and trustworthiness [422], [423]. Efforts are being made to make LLMs more explainable to promote user trust and to ensure responsible AI usage. Understanding the logic behind LLMs' responses is essential for fostering trust and ensuring they align with human values and legal standards.

Privacy Concerns: Privacy concerns in Large Language Models (LLMs) have escalated with their growth in complexity and size, particularly around data sharing and potential misuse. There is a risk of malicious content creation, filter bypass, and data privacy issues, especially in e-commerce, where protecting customer privacy is crucial. If models are trained on private data, additional concerns arise if such models are made publicly available. LLMs tend to memorize phrases from their training sets, which an adversary could exploit to extract sensitive data, posing a threat to personal privacy [424], [425].

Real-Time Processing: Real-time processing in Large Language Models (LLMs) is pivotal for various applications, especially with the rising popularity of mobile AI applications and concerns regarding information security and privacy. However, LLMs often have hundreds of layers and millions of parameters, which impede real-time processing due to the high computational demands and limited weight storage on hardware platforms, particularly in edge computing environments [426]. While certain efforts like MobileBERT aim to reduce memory requirements, they still face substantial execution overhead due to the large number of model layers, leading to high inference latency.

Long-Term Dependencies: Large Language Models (LLMs) have shown considerable progress in understanding and generating text, yet they often struggle with preserving context and handling long-term dependencies, particularly in complex, multi-turn conversations or long documents. This limitation can lead to incoherent or irrelevant responses.

Hardware Acceleration: The growth of LLMs presents significant hardware challenges due to the increasing computational and memory demands associated with training and deploying these models. GPUs have played a crucial role in meeting the hardware requirements for training LLMs, with the networking industry also evolving to optimize hardware for training workloads. However, the growing size of LLMs, which has been outpacing hardware progress, makes model inference increasingly costly. Model quantization is a promising approach to bridge the widening gap between LLM size and hardware capacity [427]. Although specialized hardware acceleration like GPUs or TPUs can significantly reduce the computational cost, making real-time applications more feasible, they may not fully resolve all limitations, necessitating further advancements in hardware technology.

Regulatory and Ethical Frameworks: The rapid advancements in artificial intelligence have given rise to sophisticated Large Language Models (LLMs) like OpenAI's GPT-4 [149] and Google's Bard. These developments underscore the imperative for regulatory oversight to manage the ethical and social challenges accompanying LLMs' widespread use [428]. For instance, LLMs can generate content that can be used positively or negatively, emphasizing the need for proactive ethical frameworks and policy measures to guide their responsible use and assign accountability for their outputs [429]. Auditing is identified as a promising governance mechanism to ensure that AI systems, including LLMs, are designed and deployed ethically, legally, and technically robust [430].

IX. Conclusion

This article has reviewed various LLMs, discussing their interesting aspects. It contributes to summarizing significant findings in the existing literature and provides a detailed analysis of the design aspects of LLMs, including architectures, datasets, and training pipelines. We identified crucial architectural components and training strategies employed by different LLMs. These aspects are presented as summaries and discussions throughout the article. Moreover, we have discussed the performance differences of LLMs in zero-shot and few-shot settings, explored the impact of fine-tuning, and compared supervised and generalized models and encoder vs decoder vs encoder-decoder architectures. A comprehensive review of LLMs in robotics, multi-modal LLMs, augmented LLMs, datasets, and evaluation is also provided. This article is anticipated to serve as a valuable resource for researchers, offering insights into the recent advancements in LLMs and providing fundamental concepts and details to develop better LLMs.

X. Versioning

We keep track of the versions of this paper we release as the content updates.

Version 1.0: We covered 30 pre-trained models and 6 instruction-tuned models, including their overview, findings, training, and evaluation datasets, and discussed important architectural and training tricks by various LLMs.

Version 2.0: Further pre-trained LLMs added along with discussion on on self-instruct LLMs. Categorized LLMs according to the application, provided descriptions of widely used evaluation datasets, added a section on robotics, and extended discussion in section VII. Tables have been updated.

Version 3.0: Added sections on Alignment tuning and multimodal LLMs. A performance comparison table on various benchmarks and datasets. Added LLaMA-2 and PaLM-2.

Version 4.0: Tables on training and evaluation datasets, a subsection on increasing context window, and minor improvements.

Version 5.0: Added sections on augmented LLMs and challenges and future directions.

Version 6.0: Minor improvements in abstract, introduction, and conclusion.

Note: If you find any mistakes, or have issues and conflicts with the writing in this paper, please email us. We welcome suggestions to improve this paper.

References

[1] A. Chernyavskiy, D. Ilvovsky, and P. Nakov, "Transformers:"the end of history" for natural language processing?" in *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part III 21*. Springer, 2021, pp. 677–693. 1

[2] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems," *Advances in neural information processing systems*, vol. 32, 2019. 1, 21, 24

[3] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu *et al.*, "Towards a human-like open-domain chatbot," *arXiv preprint arXiv:2001.09977*, 2020. 1

[4] B. A. y Arcas, "Do large language models understand us?" *Daedalus*, vol. 151, no. 2, pp. 183–197, 2022. 1

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019. 1, 8

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020. 1, 2, 7, 8, 9, 10, 12, 14, 18, 21, 27, 28, 29, 30

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. 2, 19, 29

[8] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL-HLT*. Association for Computational Linguistics, 2018, pp. 2227–2237. 2

[9] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019. 2

[10] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020. 2, 6, 7, 8, 9, 16, 20, 21, 23, 24, 25, 26, 29

[11] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," *arXiv preprint arXiv:2010.11934*, 2020. 2, 7, 9, 21, 23, 25, 29

[12] Z. Zhang, Y. Gu, X. Han, S. Chen, C. Xiao, Z. Sun, Y. Yao, F. Qi, J. Guan, P. Ke *et al.*, "Cpm-2: Large-scale cost-effective pre-trained language models," *AI Open*, vol. 2, pp. 216–224, 2021. 2, 9, 21

[13] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé *et al.*, "Bloom: A 176b-parameter open-access multilingual language model," *arXiv preprint arXiv:2211.05100*, 2022. 2, 5, 10, 12, 21, 25, 27, 28, 29

[14] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin *et al.*, "Opt: Open pre-trained transformer language models," *arXiv preprint arXiv:2205.01068*, 2022. 2, 10, 12, 21, 29

[15] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022. 2, 7, 10, 11, 21, 27, 28, 29

[16] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, "Scaling instruction-finetuned language models," *arXiv preprint arXiv:2210.11416*, 2022. 2, 7, 8, 12, 16, 18, 21, 23, 26, 29

[17] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja *et al.*, "Multitask prompted training enables zero-shot task generalization," *arXiv preprint arXiv:2110.08207*, 2021. 2, 12, 21, 23, 26

[18] Y. Wang, S. Mishra, P. Alipoormolabashi, Y. Kordi, A. Mirzaei, A. Naik, A. Ashok, A. S. Dhanasekaran, A. Arunkumar, D. Stap *et al.*, "Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 5085–5109. 2, 8, 12, 16, 21, 23, 26, 29

[19] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language model with self generated instructions," *arXiv preprint arXiv:2212.10560*, 2022. 2, 12, 18, 20, 23

[20] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022. 2, 7, 12, 14, 18, 29

[21] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023. 2, 7, 11, 15, 21, 30

[22] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022. 2

[23] T. Webb, K. J. Holyoak, and H. Lu, "Emergent analogical reasoning in large language models," *Nature Human Behaviour*, vol. 7, no. 9, pp. 1526–1541, 2023. 2

[24] D. A. Boiko, R. MacKnight, and G. Gomes, "Emergent autonomous scientific research capabilities of large language models," *arXiv preprint arXiv:2304.05332*, 2023. 2

[25] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave, "Few-shot learning with retrieval augmented language models," *arXiv preprint arXiv:2208.03299*, 2022. 2, 18, 19, 30

[26] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023. 2, 17

[27] A. Parisi, Y. Zhao, and N. Fiedel, "Talm: Tool augmented language models," *arXiv preprint arXiv:2205.12255*, 2022. 2, 19, 20

[28] B. Zhang and H. Soh, "Large language models as zero-shot human models for human-robot interaction," *arXiv preprint arXiv:2303.03548*, 2023. 2, 17

[29] Q. Ye, H. Xu, G. Xu, J. Ye, M. Yan, Y. Zhou, J. Wang, A. Hu, P. Shi, Y. Shi *et al.*, "mplug-owl: Modularization empowers large language models with multimodality," *arXiv preprint arXiv:2304.14178*, 2023. 2, 18

[30] W. Wang, Z. Chen, X. Chen, J. Wu, X. Zhu, G. Zeng, P. Luo, T. Lu, J. Zhou, Y. Qiao *et al.*, "Visionllm: Large language model is also an open-ended decoder for vision-centric tasks," *arXiv preprint arXiv:2305.11175*, 2023. 2, 18

[31] R. Yang, L. Song, Y. Li, S. Zhao, Y. Ge, X. Li, and Y. Shan, "Gpt4tools: Teaching large language model to use tools via self-instruction," *arXiv preprint arXiv:2305.18752*, 2023. 2, 18

[32] E. Saravia, "Prompt Engineering Guide," *https://github.com/dair-ai/Prompt-Engineering-Guide*, 12 2022. 2, 8, 18, 30

[33] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia *et al.*, "Glm-130b: An open bilingual pre-trained model," *arXiv preprint arXiv:2210.02414*, 2022. 2, 11, 21, 27, 28, 29

[34] Y. Wang, H. Le, A. D. Gotmare, N. D. Bui, J. Li, and S. C. Hoi, "Codet5+: Open code large language models for code understanding and generation," *arXiv preprint arXiv:2305.07922*, 2023. 2, 11, 21, 29

[35] S. Wang, Y. Sun, Y. Xiang, Z. Wu, S. Ding, W. Gong, S. Feng, J. Shang, Y. Zhao, C. Pang *et al.*, "Ernie 3.0 titan: Exploring larger-scale knowledge enhanced pre-training for language understanding and generation," *arXiv preprint arXiv:2112.12731*, 2021. 2, 9, 21, 29

[36] J. Rasley, S. Rajbhandari, O. Ruwase, and Y. He, "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*, 2020, pp. 3505–3506. 2, 6

[37] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, "Zero: Memory optimizations toward training trillion parameter models," in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2020, pp. 1–16. 2, 5, 6, 29

[38] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "Towards a unified view of parameter-efficient transfer learning," *arXiv preprint arXiv:2110.04366*, 2021. 2, 8

[39] Z. Hu, Y. Lan, L. Wang, W. Xu, E.-P. Lim, R. K.-W. Lee, L. Bing, and S. Poria, "Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models," *arXiv preprint arXiv:2304.01933*, 2023. 2, 8

[40] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021. 2, 8

[41] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," *arXiv preprint arXiv:2101.00190*, 2021. 2, 8

[42] A. Pal, D. Karkhanis, M. Roberts, S. Dooley, A. Sundararajan, and S. Naidu, "Giraffe: Adventures in expanding context lengths in llms," *arXiv preprint arXiv:2308.10882*, 2023. 2, 16

[43] B. Peng, J. Quesnelle, H. Fan, and E. Shippole, "Yarn: Efficient context window extension of large language models," *arXiv preprint arXiv:2309.00071*, 2023. 2, 16

[44] M. Guo, J. Ainslie, D. Uthus, S. Ontanon, J. Ni, Y.-H. Sung, and Y. Yang, "Longt5: Efficient text-to-text transformer for long sequences," *arXiv preprint arXiv:2112.07916*, 2021. 2, 16

[45] S. Chen, S. Wong, L. Chen, and Y. Tian, "Extending context window of large language models via positional interpolation," *arXiv preprint arXiv:2306.15595*, 2023. 2, 16

[46] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023. 3, 7, 8, 16

[47] U. Naseem, I. Razzak, S. K. Khan, and M. Prasad, "A comprehensive survey on word representation models: From classical to state-of-the-art word representation language models," *Transactions on Asian and Low-Resource Language Information Processing*, vol. 20, no. 5, pp. 1–35, 2021. 3, 4

[48] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heinz, and D. Roth, "Recent advances in natural language processing via large pre-trained language models: A survey," *arXiv preprint arXiv:2111.01243*, 2021. 3, 4

[49] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He *et al.*, "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt," *arXiv preprint arXiv:2302.09419*, 2023. 3, 4

[50] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, and Z. Sui, "A survey for in-context learning," *arXiv preprint arXiv:2301.00234*, 2022. 3, 8, 18

[51] J. Huang and K. C.-C. Chang, "Towards reasoning in large language models: A survey," *arXiv preprint arXiv:2212.10403*, 2022. 3, 8, 18

[52] Y. Wang, W. Zhong, L. Li, F. Mi, X. Zeng, W. Huang, L. Shang, X. Jiang, and Q. Liu, "Aligning large language models with human: A survey," *arXiv preprint arXiv:2307.12966*, 2023. 3

[53] X. Zhu, J. Li, Y. Liu, C. Ma, and W. Wang, "A survey on model compression for large language models," *arXiv preprint arXiv:2308.07633*, 2023. 3

[54] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A survey on multimodal large language models," *arXiv preprint arXiv:2306.13549*, 2023. 3

[55] J. J. Webster and C. Kit, "Tokenization as the initial phase in nlp," in *COLING 1992 volume 4: The 14th international conference on computational linguistics*, 1992. 4

[56] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 66–75. 4

[57] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1715–1725. 4

[58] S. J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Gallé, A. Raja, C. Si, W. Y. Lee, B. Sagot *et al.*, "Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp," *arXiv preprint arXiv:2112.10508*, 2021. 4

[59] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2012, pp. 5149–5152. 4

[60] C. W. Eriksen and J. E. Hoffman, "Some characteristics of selective attention in visual perception determined by vocal reaction time," *Perception & Psychophysics*, vol. 11, no. 2, pp. 169–171, 1972. 4

[61] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. 4

[62] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. 4, 5, 8

[63] R. Child, S. Gray, A. Radford, and I. Sutskever, "Generating long sequences with sparse transformers," *arXiv preprint arXiv:1904.10509*, 2019. 4, 8, 28

[64] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 344–16 359, 2022. 4

[65] O. Press, N. Smith, and M. Lewis, "Train short, test long: Attention with linear biases enables input length extrapolation," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=R8sQPpGCv0 4, 5, 16

[66] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *arXiv preprint arXiv:2104.09864*, 2021. 4, 5, 10, 16

[67] A. Kazemnejad, I. Padhi, K. N. Ramamurthy, P. Das, and S. Reddy, "The impact of positional encoding on length generalization in transformers," *arXiv preprint arXiv:2305.19466*, 2023. 4

[68] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989. 5

[69] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814. 5

[70] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016. 5

[71] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014. 5

[72] D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N. R. Ke, A. Goyal, Y. Bengio, A. Courville, and C. Pal, "Zoneout: Regularizing rnns by randomly preserving hidden activations," *arXiv preprint arXiv:1606.01305*, 2016. 5

[73] N. Shazeer, "Glu variants improve transformer," *arXiv preprint arXiv:2002.05202*, 2020. 5

[74] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *International conference on machine learning*. PMLR, 2017, pp. 933–941. 5

[75] B. Zhang and R. Sennrich, "Root mean square layer normalization," *Advances in Neural Information Processing Systems*, vol. 32, 2019. 5

[76] A. Baevski and M. Auli, "Adaptive input representations for neural language modeling," *arXiv preprint arXiv:1809.10853*, 2018. 5

[77] S. Shleifer, J. Weston, and M. Ott, "Normformer: Improved transformer pretraining with extra normalization," *arXiv preprint arXiv:2110.09456*, 2021. 5

[78] H. Wang, S. Ma, L. Dong, S. Huang, D. Zhang, and F. Wei, "Deepnet: Scaling transformers to 1,000 layers," *arXiv preprint arXiv:2203.00555*, 2022. 5

[79] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-lm: Training multi-billion parameter language models using model parallelism," *arXiv preprint arXiv:1909.08053*, 2019. 5, 6

[80] ""bmtrain: Efficient training for big models."." [Online]. Available: https://github.com/OpenBMB/BMTrain 5, 6

[81] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020, pp. 38–45. 6

[82] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne *et al.*, "Jax: composable transformations of python+ numpy programs," 2018. 6

[83] S. Li, J. Fang, Z. Bian, H. Liu, Y. Liu, H. Huang, B. Wang, and Y. You, "Colossal-ai: A unified deep learning system for large-scale parallel training," *arXiv preprint arXiv:2110.14883*, 2021. 6

[84] J. He, J. Qiu, A. Zeng, Z. Yang, J. Zhai, and J. Tang, "Fastmoe: A fast mixture-of-expert training system," *arXiv preprint arXiv:2103.13262*, 2021. 6

[85] L. Huawei Technologies Co., "Huawei mindspore ai development framework," in *Artificial Intelligence Technology*. Springer, 2022, pp. 137–162. 6

[86] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019. 6

[87] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *Osdi*, vol. 16, no. 2016. Savannah, GA, USA, 2016, pp. 265–283. 6

[88] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv preprint arXiv:1512.01274*, 2015. 6

[89] Y. Tay, M. Dehghani, V. Q. Tran, X. Garcia, J. Wei, X. Wang, H. W. Chung, D. Bahri, T. Schuster, S. Zheng *et al.*, "Ul2: Unifying language learning paradigms," in *The Eleventh International Conference on Learning Representations*, 2022. 6, 11, 21, 29

[90] P. J. Liu*, M. Saleh*, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating wikipedia by summarizing long sequences," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=Hyg0vbWC- 6

[91] T. Wang, A. Roberts, D. Hesslow, T. Le Scao, H. W. Chung, I. Beltagy, J. Launay, and C. Raffel, "What language model architecture and pretraining objective works best for zero-shot generalization?" in *International Conference on Machine Learning*. PMLR, 2022, pp. 22 964–22 984. 6

[92] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon, "Unified language model pre-training for natural language understanding and generation," *Advances in neural information processing systems*, vol. 32, 2019. 7

[93] S. Iyer, X. V. Lin, R. Pasunuru, T. Mihaylov, D. Simig, P. Yu, K. Shuster, T. Wang, Q. Liu, P. S. Koura *et al.*, "Opt-iml: Scaling language model instruction meta learning through the lens of generalization," *arXiv preprint arXiv:2212.12017*, 2022. 7, 8, 12, 16, 18, 21, 23, 29

[94] Z. Sun, Y. Shen, Q. Zhou, H. Zhang, Z. Chen, D. Cox, Y. Yang, and C. Gan, "Principle-driven self-alignment of language models from scratch with minimal human supervision," *arXiv preprint arXiv:2305.03047*, 2023. 7, 16

[95] A. Askell, Y. Bai, A. Chen, D. Drain, D. Ganguli, T. Henighan, A. Jones, N. Joseph, B. Mann, N. DasSarma *et al.*, "A general language assistant as a laboratory for alignment," *arXiv preprint arXiv:2112.00861*, 2021. 8

[96] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, "Fine-tuning language models from human preferences," *arXiv preprint arXiv:1909.08593*, 2019. 8

[97] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang, "Gpt understands, too," *arXiv preprint arXiv:2103.10385*, 2021. 8

[98] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2790–2799. 8

[99] S. Kim, S. J. Joo, D. Kim, J. Jang, S. Ye, J. Shin, and M. Seo, "The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning," *arXiv preprint arXiv:2305.14045*, 2023. 8, 12

[100] Q. Liu, F. Zhou, Z. Jiang, L. Dou, and M. Lin, "From zero to hero: Examining the power of symbolic tasks in instruction tuning," *arXiv preprint arXiv:2304.07995*, 2023. 8, 12

[101] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022. 8, 18

[102] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022. 8

[103] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *arXiv preprint arXiv:2305.10601*, 2023. 8

[104] S. McCandlish, J. Kaplan, D. Amodei, and O. D. Team, "An empirical model of large-batch training," *arXiv preprint arXiv:1812.06162*, 2018. 8

[105] W. Zeng, X. Ren, T. Su, H. Wang, Y. Liao, Z. Wang, X. Jiang, Z. Yang, K. Wang, X. Zhang *et al.*, "Pangu-α : Large-scale autoregressive pretrained chinese language models with auto-parallel computation," *arXiv preprint arXiv:2104.12369*, 2021. 9, 21, 27, 29

[106] S. Yuan, H. Zhao, Z. Du, M. Ding, X. Liu, Y. Cen, X. Zou, Z. Yang, and J. Tang, "Wudaocorpora: A super large-scale chinese corpora for pre-training language models," *AI Open*, vol. 2, pp. 65–68, 2021. 9, 25

[107] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021. 9

[108] Y. Sun, S. Wang, S. Feng, S. Ding, C. Pang, J. Shang, J. Liu, X. Chen, Y. Zhao, Y. Lu *et al.*, "Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation," *arXiv preprint arXiv:2107.02137*, 2021. 9, 21

[109] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019. 9

[110] O. Lieber, O. Sharir, B. Lenz, and Y. Shoham, "Jurassic-1: Technical details and evaluation," *White Paper. AI21 Labs*, vol. 1, 2021. 9, 21, 29

[111] Y. Levine, N. Wies, O. Sharir, H. Bata, and A. Shashua, "Limits to depth efficiencies of self-attention," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 640–22 651, 2020. 9

[112] B. Kim, H. Kim, S.-W. Lee, G. Lee, D. Kwak, D. H. Jeon, S. Park, S. Kim, S. Kim, D. Seo *et al.*, "What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers," *arXiv preprint arXiv:2109.04650*, 2021. 9, 21

[113] S. Wu, X. Zhao, T. Yu, R. Zhang, C. Shen, H. Liu, F. Li, H. Zhu, J. Luo, L. Xu *et al.*, "Yuan 1.0: Large-scale pre-trained language model in zero-shot and few-shot learning," *arXiv preprint arXiv:2110.04725*, 2021. 9, 21, 29

[114] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young *et al.*, "Scaling language models: Methods, analysis & insights from training gopher," *arXiv preprint arXiv:2112.11446*, 2021. 9, 10, 21, 23

[115] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti *et al.*, "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model," *arXiv preprint arXiv:2201.11990*, 2022. 9, 10, 21, 29

[116] S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonell, J. Phang *et al.*, "Gpt-neox-20b: An open-source autoregressive language model," *arXiv preprint arXiv:2204.06745*, 2022. 10, 27, 29

[117] W. Ben and K. Aran, "Gpt-j-6b: A 6 billion parameter autoregressive language model," 2021. 10

[118] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh *et al.*, "Mixed precision training," *arXiv preprint arXiv:1710.03740*, 2017. 10

[119] N. Du, Y. Huang, A. M. Dai, S. Tong, D. Lepikhin, Y. Xu, M. Krikun, Y. Zhou, A. W. Yu, O. Firat *et al.*, "Glam: Efficient scaling of language models with mixture-of-experts," in *International Conference on Machine Learning*. PMLR, 2022, pp. 5547–5569. 10, 21, 27, 28, 29

[120] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017. 10, 28

[121] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *The Journal of Machine Learning Research*, vol. 23, no. 1, pp. 5232–5270, 2022. 10

[122] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark *et al.*, "Training compute-optimal large language models," *arXiv preprint arXiv:2203.15556*, 2022. 10, 21, 24

[123] S. Soltan, S. Ananthakrishnan, J. FitzGerald, R. Gupta, W. Hamza, H. Khan, C. Peris, S. Rawls, A. Rosenbaum, A. Rumshisky *et al.*, "Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model," *arXiv preprint arXiv:2208.01448*, 2022. 10, 21, 27, 29

[124] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen *et al.*, "Palm 2 technical report," *arXiv preprint arXiv:2305.10403*, 2023. 10, 21

[125] Y. Tay, J. Wei, H. W. Chung, V. Q. Tran, D. R. So, S. Shakeri, X. Garcia, H. S. Zheng, J. Rao, A. Chowdhery *et al.*, "Transcending scaling laws with 0.1% extra compute," *arXiv preprint arXiv:2210.11399*, 2022. 10, 21, 29

[126] Z. Du, Y. Qian, X. Liu, M. Ding, J. Qiu, Z. Yang, and J. Tang, "Glm: General language model pretraining with autoregressive blank infilling," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 320–335. 11

[127] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023. 11, 21, 27

[128] M. N. Rabe and C. Staats, "Self-attention does not need o(n^2) memory," *arXiv preprint arXiv:2112.05682*, 2021. 11
[129] V. A. Korthikanti, J. Casper, S. Lym, L. McAfee, M. Andersch, M. Shoeybi, and B. Catanzaro, "Reducing activation recomputation in large transformer models," *Proceedings of Machine Learning and Systems*, vol. 5, 2023. 11
[130] X. Ren, P. Zhou, X. Meng, X. Huang, Y. Wang, W. Wang, P. Li, X. Zhang, A. Podolskiy, G. Arshinov *et al.*, "Pangu-$\sum$: Towards trillion parameter language model with sparse heterogeneous computing," *arXiv preprint arXiv:2303.10845*, 2023. 11, 12, 21, 27, 28, 29
[131] E. Nijkamp, B. Pang, H. Hayashi, L. Tu, H. Wang, Y. Zhou, S. Savarese, and C. Xiong, "Codegen: An open large language model for code with multi-turn program synthesis," *arXiv preprint arXiv:2203.13474*, 2022. 11, 21, 23, 27
[132] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021. 11, 21
[133] Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. Dal Lago *et al.*, "Competition-level code generation with alphacode," *Science*, vol. 378, no. 6624, pp. 1092–1097, 2022. 11, 21, 24, 27
[134] N. Shazeer, "Fast transformer decoding: One write-head is all you need," *arXiv preprint arXiv:1911.02150*, 2019. 11
[135] R. Y. Pang and H. He, "Text generation by learning from demonstrations," *arXiv preprint arXiv:2009.07839*, 2020. 11
[136] R. Dabre and A. Fujita, "Softmax tempering for training neural machine translation models," *arXiv preprint arXiv:2009.09372*, 2020. 11
[137] Y. Wang, W. Wang, S. Joty, and S. C. Hoi, "Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation," *arXiv preprint arXiv:2109.00859*, 2021. 11
[138] R. Li, L. B. Allal, Y. Zi, N. Muennighoff, D. Kocetkov, C. Mou, M. Marone, C. Akiki, J. Li, J. Chim *et al.*, "Starcoder: may the source be with you!" *arXiv preprint arXiv:2305.06161*, 2023. 11, 21
[139] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic, "Galactica: A large language model for science," *arXiv preprint arXiv:2211.09085*, 2022. 11, 21, 24, 29
[140] FairScale authors, "Fairscale: A general purpose modular pytorch library for high performance and large scale training," https://github.com/facebookresearch/fairscale, 2021. 11
[141] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du *et al.*, "Lamda: Language models for dialog applications," *arXiv preprint arXiv:2201.08239*, 2022. 12, 21
[142] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, "Bloomberggpt: A large language model for finance," *arXiv preprint arXiv:2303.17564*, 2023. 12, 21
[143] Y. Levine, N. Wies, O. Sharir, H. Bata, and A. Shashua, "Limits to depth efficiencies of self-attention," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 640–22 651, 2020. 12
[144] X. Zhang, Q. Yang, and D. Xu, "Xuanyuan 2.0: A large chinese financial chat model with hundreds of billions parameters," *arXiv preprint arXiv:2305.12002*, 2023. 12, 16, 21
[145] W. Ben, "Mesh-transformer-jax: Model-parallel implementation of transformer language model with jax," 2021. 13, 29
[146] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf *et al.*, "Crosslingual generalization through multitask finetuning," *arXiv preprint arXiv:2211.01786*, 2022. 12, 21, 23, 26
[147] D. Yin, X. Liu, F. Yin, M. Zhong, H. Bansal, J. Han, and K.-W. Chang, "Dynosaur: A dynamic growth paradigm for instruction-tuning data curation," *arXiv preprint arXiv:2305.14327*, 2023. 12
[148] P. Gao, J. Han, R. Zhang, Z. Lin, S. Geng, A. Zhou, W. Zhang, P. Lu, C. He, X. Yue *et al.*, "Llama-adapter v2: Parameter-efficient visual instruction model," *arXiv preprint arXiv:2304.15010*, 2023. 14, 29
[149] "Openai. gpt-4 technical report," 2023. 14, 31
[150] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," https://github.com/tatsu-lab/stanford_alpaca, 2023. 14, 21, 23
[151] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," March 2023. [Online]. Available: https://lmsys.org/blog/2023-03-30-vicuna/ 14, 18, 21, 23
[152] B. Peng, C. Li, P. He, M. Galley, and J. Gao, "Instruction tuning with gpt-4," *arXiv preprint arXiv:2304.03277*, 2023. 14, 23
[153] T. Liu and B. K. H. Low, "Goat: Fine-tuned llama outperforms gpt-4 on arithmetic tasks," *arXiv preprint arXiv:2305.14201*, 2023. 14
[154] H. Wang, C. Liu, N. Xi, Z. Qiang, S. Zhao, B. Qin, and T. Liu, "Huatuo: Tuning llama model with chinese medical knowledge," *arXiv preprint arXiv:2304.06975*, 2023. 14
[155] C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao, and D. Jiang, "Wizardlm: Empowering large language models to follow complex instructions," *arXiv preprint arXiv:2304.12244*, 2023. 14
[156] Z. Luo, C. Xu, P. Zhao, Q. Sun, X. Geng, W. Hu, C. Tao, J. Ma, Q. Lin, and D. Jiang, "Wizardcoder: Empowering code large language models with evol-instruct," *arXiv preprint arXiv:2306.08568*, 2023. 14, 21
[157] J. Menick, M. Trebacz, V. Mikulik, J. Aslanides, F. Song, M. Chadwick, M. Glaese, S. Young, L. Campbell-Gillingham, G. Irving *et al.*, "Teaching language models to support answers with verified quotes," *arXiv preprint arXiv:2203.11147*, 2022. 15
[158] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders *et al.*, "Webgpt: Browser-assisted question-answering with human feedback," *arXiv preprint arXiv:2112.09332*, 2021. 15, 19, 21, 26, 29
[159] A. Glaese, N. McAleese, M. Trębacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker *et al.*, "Improving alignment of dialogue agents via targeted human judgements," *arXiv preprint arXiv:2209.14375*, 2022. 15, 21
[160] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *arXiv preprint arXiv:2305.18290*, 2023. 15
[161] H. Dong, W. Xiong, D. Goyal, R. Pan, S. Diao, J. Zhang, K. Shum, and T. Zhang, "Raft: Reward ranked finetuning for generative foundation model alignment," *arXiv preprint arXiv:2304.06767*, 2023. 15
[162] Z. Yuan, H. Yuan, C. Tan, W. Wang, S. Huang, and F. Huang, "Rrhf: Rank responses to align language models with human feedback without tears," *arXiv preprint arXiv:2304.05302*, 2023. 15
[163] F. Song, B. Yu, M. Li, H. Yu, F. Huang, Y. Li, and H. Wang, "Preference ranking optimization for human alignment," *arXiv preprint arXiv:2306.17492*, 2023. 15
[164] H. Liu, C. Sferrazza, and P. Abbeel, "Languages are rewards: Hindsight finetuning using human feedback," *arXiv preprint arXiv:2302.02676*, 2023. 15
[165] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon *et al.*, "Constitutional ai: Harmlessness from ai feedback," *arXiv preprint arXiv:2212.08073*, 2022. 16
[166] Y. Dubois, X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. B. Hashimoto, "Alpacafarm: A simulation framework for methods that learn from human feedback," *arXiv preprint arXiv:2305.14387*, 2023. 16
[167] C. Si, Z. Gan, Z. Yang, S. Wang, J. Wang, J. Boyd-Graber, and L. Wang, "Prompting gpt-3 to be reliable," *arXiv preprint arXiv:2210.09150*, 2022. 16
[168] D. Ganguli, A. Askell, N. Schiefer, T. Liao, K. Lukošiūtė, A. Chen, A. Goldie, A. Mirhoseini, C. Olsson, D. Hernandez *et al.*, "The capacity for moral self-correction in large language models," *arXiv preprint arXiv:2302.07459*, 2023. 16
[169] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How does llm safety training fail?" *arXiv preprint arXiv:2307.02483*, 2023. 16
[170] D. Ganguli, L. Lovitt, J. Kernion, A. Askell, Y. Bai, S. Kadavath, B. Mann, E. Perez, N. Schiefer, K. Ndousse *et al.*, "Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned," *arXiv preprint arXiv:2209.07858*, 2022. 16, 23
[171] S. Casper, J. Lin, J. Kwon, G. Culp, and D. Hadfield-Menell, "Explore, establish, exploit: Red teaming language models from scratch," *arXiv preprint arXiv:2306.09442*, 2023. 16
[172] E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, and G. Irving, "Red teaming language models with language models," *arXiv preprint arXiv:2202.03286*, 2022. 16
[173] T. Scialom, T. Chakrabarty, and S. Muresan, "Fine-tuned language models are continual learners," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 6107–6122. 16
[174] Z. Shi and A. Lipani, "Don't stop pretraining? make prompt-based fine-tuning powerful learner," *arXiv preprint arXiv:2305.01711*, 2023. 16
[175] H. Gupta, S. A. Sawant, S. Mishra, M. Nakamura, A. Mitra, S. Mashetty, and C. Baral, "Instruction tuned models are quick learners," *arXiv preprint arXiv:2306.05539*, 2023. 16

[176] H. Chen, Y. Zhang, Q. Zhang, H. Yang, X. Hu, X. Ma, Y. Yanggong, and J. Zhao, "Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning," *arXiv preprint arXiv:2305.09246*, 2023. 16
[177] C. Zhou, P. Liu, P. Xu, S. Iyer, J. Sun, Y. Mao, X. Ma, A. Efrat, P. Yu, L. Yu *et al.*, "Lima: Less is more for alignment," *arXiv preprint arXiv:2305.11206*, 2023. 16, 21, 23
[178] C. Han, Q. Wang, W. Xiong, Y. Chen, H. Ji, and S. Wang, "Lm-infinite: Simple on-the-fly length generalization for large language models," *arXiv preprint arXiv:2308.16137*, 2023. 16
[179] J. Ainslie, T. Lei, M. de Jong, S. Ontañón, S. Brahma, Y. Zemlyanskiy, D. Uthus, M. Guo, J. Lee-Thorp, Y. Tay *et al.*, "Colt5: Faster long-range transformers with conditional computation," *arXiv preprint arXiv:2303.09752*, 2023. 16
[180] J. Ding, S. Ma, L. Dong, X. Zhang, S. Huang, W. Wang, and F. Wei, "Longnet: Scaling transformers to 1,000,000,000 tokens," *arXiv preprint arXiv:2307.02486*, 2023. 16
[181] Y. Chen, S. Qian, H. Tang, X. Lai, Z. Liu, S. Han, and J. Jia, "Longlora: Efficient fine-tuning of long-context large language models," *arXiv preprint arXiv:2309.12307*, 2023. 16
[182] N. Ratner, Y. Levine, Y. Belinkov, O. Ram, I. Magar, O. Abend, E. Karpas, A. Shashua, K. Leyton-Brown, and Y. Shoham, "Parallel context windows for large language models," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 6383–6402. 16
[183] A. Lykov and D. Tsetserukou, "Llm-brain: Ai-driven fast generation of robot behaviour tree based on large language model," *arXiv preprint arXiv:2305.19352*, 2023. 17
[184] E. Billing, J. Rosén, and M. Lamb, "Language models for human-robot interaction," in *ACM/IEEE International Conference on Human-Robot Interaction, March 13–16, 2023, Stockholm, Sweden*. ACM Digital Library, 2023, pp. 905–906. 17
[185] Y. Ye, H. You, and J. Du, "Improved trust in human-robot collaboration with chatgpt," *IEEE Access*, 2023. 17
[186] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530. 17
[187] Y. Zhen, S. Bi, L. Xing-tong, P. Wei-qin, S. Hai-peng, C. Zi-rui, and F. Yi-shu, "Robot task planning based on large language model representing knowledge with directed graph structures," *arXiv preprint arXiv:2306.05171*, 2023. 17
[188] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International Conference on Machine Learning*. PMLR, 2022, pp. 9118–9147. 17
[189] Y. Ding, X. Zhang, C. Paxton, and S. Zhang, "Task and motion planning with large language models for object rearrangement," *arXiv preprint arXiv:2303.06247*, 2023. 17
[190] ——, "Leveraging commonsense knowledge from large language models for task and motion planning," in *RSS 2023 Workshop on Learning for Task and Motion Planning*, 2023. 17
[191] Y. Ge, W. Hua, J. Ji, J. Tan, S. Xu, and Y. Zhang, "Openagi: When llm meets domain experts," *arXiv preprint arXiv:2304.04370*, 2023. 17
[192] T. Zhong, Y. Wei, L. Yang, Z. Wu, Z. Liu, X. Wei, W. Li, J. Yao, C. Ma, X. Li *et al.*, "Chatabl: Abductive learning via natural language interaction with chatgpt," *arXiv preprint arXiv:2304.11107*, 2023. 17
[193] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser, "Tidybot: Personalized robot assistance with large language models," *arXiv preprint arXiv:2305.05658*, 2023. 17
[194] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, T. Jackson, N. Brown, L. Luu, S. Levine, K. Hausman, and brian ichter, "Inner monologue: Embodied reasoning through planning with language models," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=3R3Pz5i0tye 17
[195] S. S. Kannan, V. L. Venkatesh, and B.-C. Min, "Smart-llm: Smart multi-agent robot task planning using large language models," *arXiv preprint arXiv:2309.10062*, 2023. 17
[196] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: program generation for situated robot task planning using large language models," *Autonomous Robots*, pp. 1–14, 2023. 17
[197] C. Jin, W. Tan, J. Yang, B. Liu, R. Song, L. Wang, and J. Fu, "Alphablock: Embodied finetuning for vision-language reasoning in robot manipulation," *arXiv preprint arXiv:2305.18898*, 2023. 17
[198] G. Chalvatzaki, A. Younes, D. Nandha, A. T. Le, L. F. Ribeiro, and I. Gurevych, "Learning to reason over scene graphs: a case study of finetuning gpt-2 into a robot language model for grounded task planning," *Frontiers in Robotics and AI*, vol. 10, p. 1221739, 2023. 17
[199] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," *arXiv preprint arXiv:2307.14535*, 2023. 17
[200] Z. Mandi, S. Jain, and S. Song, "Roco: Dialectic multi-robot collaboration with large language models," *arXiv preprint arXiv:2307.04738*, 2023. 17
[201] A. Rajvanshi, K. Sikka, X. Lin, B. Lee, H.-P. Chiu, and A. Velasquez, "Saynav: Grounding large language models for dynamic planning to navigation in new environments," *arXiv preprint arXiv:2309.04077*, 2023. 17
[202] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, "Llm-planner: Few-shot grounded planning for embodied agents with large language models," *arXiv preprint arXiv:2212.04088*, 2022. 17
[203] V. S. Dorbala, J. F. Mullen Jr, and D. Manocha, "Can an embodied agent find your" cat-shaped mug"? llm-based zero-shot object navigation," *arXiv preprint arXiv:2303.03480*, 2023. 17
[204] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 608–10 615. 17
[205] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds *et al.*, "Flamingo: a visual language model for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23 716–23 736, 2022. 17, 18
[206] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," *arXiv preprint arXiv:2301.12597*, 2023. 17, 18
[207] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *arXiv preprint arXiv:2304.08485*, 2023. 17, 18
[208] K. Li, Y. He, Y. Wang, Y. Li, W. Wang, P. Luo, Y. Wang, L. Wang, and Y. Qiao, "Videochat: Chat-centric video understanding," *arXiv preprint arXiv:2305.06355*, 2023. 17, 18
[209] M. Maaz, H. Rasheed, S. Khan, and F. S. Khan, "Video-chatgpt: Towards detailed video understanding via large vision and language models," *arXiv preprint arXiv:2306.05424*, 2023. 17
[210] H. Zhang, X. Li, and L. Bing, "Video-llama: An instruction-tuned audio-visual language model for video understanding," *arXiv preprint arXiv:2306.02858*, 2023. 17
[211] X. Mei, C. Meng, H. Liu, Q. Kong, T. Ko, C. Zhao, M. D. Plumbley, Y. Zou, and W. Wang, "Wavcaps: A chatgpt-assisted weakly-labelled audio captioning dataset for audio-language multimodal research," *arXiv preprint arXiv:2303.17395*, 2023. 17
[212] C. Lyu, M. Wu, L. Wang, X. Huang, B. Liu, Z. Du, S. Shi, and Z. Tu, "Macaw-llm: Multi-modal language modeling with image, audio, video, and text integration," *arXiv preprint arXiv:2306.09093*, 2023. 17
[213] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "Minigpt-4: Enhancing vision-language understanding with advanced large language models," *arXiv preprint arXiv:2304.10592*, 2023. 18
[214] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020. 18
[215] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. Hoi, "Instructblip: Towards general-purpose vision-language models with instruction tuning," *arXiv preprint arXiv:2305.06500*, 2023. 18
[216] Z. Xu, Y. Shen, and L. Huang, "Multiinstruct: Improving multi-modal zero-shot learning via instruction tuning," *arXiv preprint arXiv:2212.10773*, 2022. 18
[217] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A survey on multimodal large language models," *arXiv preprint arXiv:2306.13549*, 2023. 18
[218] Z. Zhao, L. Guo, T. Yue, S. Chen, S. Shao, X. Zhu, Z. Yuan, and J. Liu, "Chatbridge: Bridging modalities with large language model as a language catalyst," *arXiv preprint arXiv:2305.16103*, 2023. 18
[219] L. Li, Y. Yin, S. Li, L. Chen, P. Wang, S. Ren, M. Li, Y. Yang, J. Xu, X. Sun *et al.*, "M3 it: A large-scale dataset towards multi-modal mul-

tilingual instruction tuning," *arXiv preprint arXiv:2306.04387*, 2023. 18
[220] R. Pi, J. Gao, S. Diao, R. Pan, H. Dong, J. Zhang, L. Yao, J. Han, H. Xu, and L. K. T. Zhang, "Detgpt: Detect what you need via reasoning," *arXiv preprint arXiv:2305.14167*, 2023. 18
[221] G. Luo, Y. Zhou, T. Ren, S. Chen, X. Sun, and R. Ji, "Cheap and quick: Efficient vision-language instruction tuning for large language models," *arXiv preprint arXiv:2305.15023*, 2023. 18
[222] R. Zhang, J. Han, A. Zhou, X. Hu, S. Yan, P. Lu, H. Li, P. Gao, and Y. Qiao, "Llama-adapter: Efficient fine-tuning of language models with zero-init attention," *arXiv preprint arXiv:2303.16199*, 2023. 18
[223] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518. 18
[224] Z. Zhang, A. Zhang, M. Li, H. Zhao, G. Karypis, and A. Smola, "Multimodal chain-of-thought reasoning in language models," *arXiv preprint arXiv:2302.00923*, 2023. 18
[225] J. Ge, H. Luo, S. Qian, Y. Gan, J. Fu, and S. Zhan, "Chain of thought prompt tuning in vision language models," *arXiv preprint arXiv:2304.07919*, 2023. 18
[226] C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, and N. Duan, "Visual chatgpt: Talking, drawing and editing with visual foundation models," *arXiv preprint arXiv:2303.04671*, 2023. 18
[227] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, and L. Wang, "Mm-react: Prompting chatgpt for multimodal reasoning and action," *arXiv preprint arXiv:2303.11381*, 2023. 18
[228] T. Wang, J. Zhang, J. Fei, Y. Ge, H. Zheng, Y. Tang, Z. Li, M. Gao, S. Zhao, Y. Shan *et al.*, "Caption anything: Interactive image description with diverse multimodal controls," *arXiv preprint arXiv:2305.02677*, 2023. 18
[229] X. Zhu, R. Zhang, B. He, Z. Zeng, S. Zhang, and P. Gao, "Pointclip v2: Adapting clip for powerful 3d open-world learning," *arXiv preprint arXiv:2211.11682*, 2022. 18
[230] P. Lu, B. Peng, H. Cheng, M. Galley, K.-W. Chang, Y. N. Wu, S.-C. Zhu, and J. Gao, "Chameleon: Plug-and-play compositional reasoning with large language models," *arXiv preprint arXiv:2304.09842*, 2023. 18
[231] T. Gupta and A. Kembhavi, "Visual programming: Compositional visual reasoning without training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 953–14 962. 18
[232] P. Gao, Z. Jiang, H. You, P. Lu, S. C. Hoi, X. Wang, and H. Li, "Dynamic fusion with intra-and inter-modality attention flow for visual question answering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6639–6648. 18
[233] Z. Yu, J. Yu, Y. Cui, D. Tao, and Q. Tian, "Deep modular co-attention networks for visual question answering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6281–6290. 18
[234] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021. 18
[235] H. You, R. Sun, Z. Wang, L. Chen, G. Wang, H. A. Ayyubi, K.-W. Chang, and S.-F. Chang, "Idealgpt: Iteratively decomposing vision and language reasoning via large language models," *arXiv preprint arXiv:2305.14985*, 2023. 18
[236] R. Zhang, X. Hu, B. Li, S. Huang, H. Deng, Y. Qiao, P. Gao, and H. Li, "Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 15 211–15 222. 18
[237] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, and F. Wei, "Augmenting language models with long-term memory," *arXiv preprint arXiv:2306.07174*, 2023. 18
[238] X. Xu, Z. Gou, W. Wu, Z.-Y. Niu, H. Wu, H. Wang, and S. Wang, "Long time no see! open-domain conversation with long-term persona memory," *arXiv preprint arXiv:2203.05797*, 2022. 18
[239] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark *et al.*, "Improving language models by retrieving from trillions of tokens," in *International conference on machine learning*. PMLR, 2022, pp. 2206–2240. 18, 19, 30
[240] W. Zhong, L. Guo, Q. Gao, and Y. Wang, "Memorybank: Enhancing large language models with long-term memory," *arXiv preprint arXiv:2305.10250*, 2023. 18
[241] N. Shinn, F. Cassano, B. Labash, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," *arXiv preprint arXiv:2303.11366*, vol. 14, 2023. 18
[242] C. Hu, J. Fu, C. Du, S. Luo, J. Zhao, and H. Zhao, "Chatdb: Augmenting llms with databases as their symbolic memory," *arXiv preprint arXiv:2306.03901*, 2023. 18
[243] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," *arXiv preprint arXiv:2305.06983*, 2023. 18, 19
[244] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," *arXiv preprint arXiv:2302.00083*, 2023. 18, 19, 30
[245] X. Li and X. Qiu, "Mot: Pre-thinking and recalling enable chatgpt to self-improve with memory-of-thoughts," *arXiv preprint arXiv:2305.05181*, 2023. 18
[246] D. Schuurmans, "Memory augmented large language models are computationally universal," *arXiv preprint arXiv:2301.04589*, 2023. 18
[247] A. Modarressi, A. Imani, M. Fayyaz, and H. Schütze, "Ret-llm: Towards a general read-write memory for large language models," *arXiv preprint arXiv:2305.14322*, 2023. 18
[248] S. Robertson, H. Zaragoza *et al.*, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009. 19
[249] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, and D. Zhou, "Rationale-augmented ensembles in language models," *arXiv preprint arXiv:2207.00747*, 2022. 19
[250] F. Zhang, B. Chen, Y. Zhang, J. Liu, D. Zan, Y. Mao, J.-G. Lou, and W. Chen, "Repocoder: Repository-level code completion through iterative retrieval and generation," *arXiv preprint arXiv:2303.12570*, 2023. 19
[251] B. Wang, W. Ping, P. Xu, L. McAfee, Z. Liu, M. Shoeybi, Y. Dong, O. Kuchaiev, B. Li, C. Xiao *et al.*, "Shall we pretrain autoregressive language models with retrieval? a comprehensive study," *arXiv preprint arXiv:2304.06762*, 2023. 19
[252] L. Wang, N. Yang, and F. Wei, "Learning to retrieve in-context examples for large language models," *arXiv preprint arXiv:2307.07164*, 2023. 19
[253] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?" *arXiv preprint arXiv:2101.06804*, 2021. 19
[254] O. Rubin, J. Herzig, and J. Berant, "Learning to retrieve prompts for in-context learning," *arXiv preprint arXiv:2112.08633*, 2021. 19
[255] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W.-t. Yih, "Replug: Retrieval-augmented black-box language models," *arXiv preprint arXiv:2301.12652*, 2023. 19
[256] O. Rubin and J. Berant, "Long-range language modeling with self-retrieval," *arXiv preprint arXiv:2306.13421*, 2023. 19
[257] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "Retrieval augmented language model pre-training," in *International conference on machine learning*. PMLR, 2020, pp. 3929–3938. 19
[258] S. Hofstätter, J. Chen, K. Raman, and H. Zamani, "Fid-light: Efficient and effective retrieval-augmented text generation," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 1437–1447. 19
[259] M. Komeili, K. Shuster, and J. Weston, "Internet-augmented dialogue generation," *arXiv preprint arXiv:2107.07566*, 2021. 19
[260] A. Lazaridou, E. Gribovskaya, W. Stokowiec, and N. Grigorev, "Internet-augmented language models through few-shot prompting for open-domain question answering," *arXiv preprint arXiv:2203.05115*, 2022. 19
[261] D. Gao, L. Ji, L. Zhou, K. Q. Lin, J. Chen, Z. Fan, and M. Z. Shou, "Assistgpt: A general multi-modal assistant that can plan, execute, inspect, and learn," *arXiv preprint arXiv:2306.08640*, 2023. 19, 20
[262] P. Lu, B. Peng, H. Cheng, M. Galley, K.-W. Chang, Y. N. Wu, S.-C. Zhu, and J. Gao, "Chameleon: Plug-and-play compositional reasoning with large language models," *arXiv preprint arXiv:2304.09842*, 2023. 19, 20
[263] B. Paranjape, S. Lundberg, S. Singh, H. Hajishirzi, L. Zettlemoyer, and M. T. Ribeiro, "Art: Automatic multi-step reasoning and tool-use for large language models," *arXiv preprint arXiv:2303.09014*, 2023. 19
[264] C.-Y. Hsieh, S.-A. Chen, C.-L. Li, Y. Fujii, A. Ratner, C.-Y. Lee, R. Krishna, and T. Pfister, "Tool documentation enables zero-shot tool-usage with large language models," *arXiv preprint arXiv:2308.00675*, 2023. 19
[265] Y. Song, W. Xiong, D. Zhu, C. Li, K. Wang, Y. Tian, and S. Li, "Rest-gpt: Connecting large language models with real-world applications via restful apis," *arXiv preprint arXiv:2306.06624*, 2023. 19

[266] S. Hao, T. Liu, Z. Wang, and Z. Hu, "Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings," *arXiv preprint arXiv:2305.11554*, 2023. 19
[267] S. G. Patil, T. Zhang, X. Wang, and J. E. Gonzalez, "Gorilla: Large language model connected with massive apis," *arXiv preprint arXiv:2305.15334*, 2023. 19
[268] Q. Xu, F. Hong, B. Li, C. Hu, Z. Chen, and J. Zhang, "On the tool manipulation capability of open-source large language models," *arXiv preprint arXiv:2305.16504*, 2023. 19
[269] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian *et al.*, "Toolllm: Facilitating large language models to master 16000+ real-world apis," *arXiv preprint arXiv:2307.16789*, 2023. 19, 20
[270] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface," *arXiv preprint arXiv:2303.17580*, 2023. 20
[271] R. Yang, L. Song, Y. Li, S. Zhao, Y. Ge, X. Li, and Y. Shan, "Gpt4tools: Teaching large language model to use tools via self-instruction," *arXiv preprint arXiv:2305.18752*, 2023. 20
[272] Y. Liang, C. Wu, T. Song, W. Wu, Y. Xia, Y. Liu, Y. Ou, S. Lu, L. Ji, S. Mao *et al.*, "Taskmatrix. ai: Completing tasks by connecting foundation models with millions of apis," *arXiv preprint arXiv:2303.16434*, 2023. 20
[273] D. Surís, S. Menon, and C. Vondrick, "Vipergpt: Visual inference via python execution for reasoning," *arXiv preprint arXiv:2303.08128*, 2023. 20
[274] S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonell, J. Phang *et al.*, "Gpt-neox-20b: An open-source autoregressive language model," *arXiv preprint arXiv:2204.06745*, 2022. 21
[275] X. Geng, A. Gudibande, H. Liu, E. Wallace, P. Abbeel, S. Levine, and D. Song, "Koala: A dialogue model for academic research," Blog post, April 2023. [Online]. Available: https://bair.berkeley.edu/blog/2023/04/03/koala/ 21
[276] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima *et al.*, "The pile: An 800gb dataset of diverse text for language modeling," *arXiv preprint arXiv:2101.00027*, 2020. 23, 25
[277] H. Laurençon, L. Saulnier, T. Wang, C. Akiki, A. Villanova del Moral, T. Le Scao, L. Von Werra, C. Mou, E. González Ponferrada, H. Nguyen *et al.*, "The bigscience roots corpus: A 1.6 tb composite multilingual dataset," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 809–31 826, 2022. 23
[278] "Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Main_Page 23
[279] T. Computer, "Redpajama: An open source recipe to reproduce llama training dataset," Apr. 2023. [Online]. Available: https://github.com/togethercomputer/RedPajama-Data 23
[280] O. Honovich, T. Scialom, O. Levy, and T. Schick, "Unnatural instructions: Tuning language models with (almost) no human labor," *arXiv preprint arXiv:2212.09689*, 2022. 23
[281] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback," *arXiv preprint arXiv:2204.05862*, 2022. 23
[282] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," *arXiv preprint arXiv:2009.03300*, 2020. 21, 24
[283] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso *et al.*, "Beyond the imitation game: Quantifying and extrapolating the capabilities of language models," *arXiv preprint arXiv:2206.04615*, 2022. 24
[284] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018. 21, 24
[285] Y. Yao, Q. Dong, J. Guan, B. Cao, Z. Zhang, C. Xiao, X. Wang, F. Qi, J. Bao, J. Nie *et al.*, "Cuge: A chinese language understanding and generation evaluation benchmark," *arXiv preprint arXiv:2112.13610*, 2021. 24
[286] L. Xu, H. Hu, X. Zhang, L. Li, C. Cao, Y. Li, Y. Xu, K. Sun, D. Yu, C. Yu *et al.*, "Clue: A chinese language understanding evaluation benchmark," *arXiv preprint arXiv:2004.05986*, 2020. 24
[287] L. Xu, X. Lu, C. Yuan, X. Zhang, H. Xu, H. Yuan, G. Wei, X. Pan, X. Tian, L. Qin *et al.*, "Fewclue: A chinese few-shot learning evaluation benchmark," *arXiv preprint arXiv:2107.07498*, 2021. 24
[288] E. M. Smith, M. Williamson, K. Shuster, J. Weston, and Y.-L. Boureau, "Can you put it all together: Evaluating conversational agents' ability to blend skills," *arXiv preprint arXiv:2004.08449*, 2020. 24
[289] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar *et al.*, "Holistic evaluation of language models," *arXiv preprint arXiv:2211.09110*, 2022. 24
[290] S. Park, J. Moon, S. Kim, W. I. Cho, J. Han, J. Park, C. Song, J. Kim, Y. Song, T. Oh *et al.*, "Klue: Korean language understanding evaluation," *arXiv preprint arXiv:2105.09680*, 2021. 24
[291] S. Reddy, D. Chen, and C. D. Manning, "Coqa: A conversational question answering challenge," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 249–266, 2019. 24
[292] M. T. Pilehvar and J. Camacho-Collados, "Wic: 10,000 example pairs for evaluating context-sensitive representations," *arXiv preprint arXiv:1808.09121*, vol. 6, 2018. 24
[293] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016. 24
[294] J. W. Rae, A. Potapenko, S. M. Jayakumar, and T. P. Lillicrap, "Compressive transformers for long-range sequence modelling," *arXiv preprint arXiv:1911.05507*, 2019. 24
[295] X. Liu, Q. Chen, C. Deng, H. Zeng, J. Chen, D. Li, and B. Tang, "Lcqmc: A large-scale chinese question matching corpus," in *Proceedings of the 27th international conference on computational linguistics*, 2018, pp. 1952–1962. 24
[296] S. Iyer, N. Dandekar, and K. Csernai, "First quora dataset release: Question pairs," https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs. 24
[297] R. Rudinger, J. Naradowsky, B. Leonard, and B. Van Durme, "Gender bias in coreference resolution," *arXiv preprint arXiv:1804.09301*, 2018. 24
[298] M.-C. De Marneffe, M. Simons, and J. Tonhauser, "The commitmentbank: Investigating projection in naturally occurring discourse," in *proceedings of Sinn und Bedeutung*, vol. 23, no. 2, 2019, pp. 107–124. 24
[299] Z. Li, N. Ding, Z. Liu, H. Zheng, and Y. Shen, "Chinese relation extraction with multi-grained information and external linguistic knowledge," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4377–4386. 24
[300] J. Xu, J. Wen, X. Sun, and Q. Su, "A discourse-level named entity recognition and relation extraction dataset for chinese literature text," *arXiv preprint arXiv:1711.07010*, 2017. 24
[301] J. Chen, Q. Chen, X. Liu, H. Yang, D. Lu, and B. Tang, "The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification," in *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2018, pp. 4946–4951. 24
[302] B. Liu, D. Niu, H. Wei, J. Lin, Y. He, K. Lai, and Y. Xu, "Matching article pairs with graphical decomposition and convolutions," *arXiv preprint arXiv:1802.07459*, 2018. 24
[303] P. Li, W. Li, Z. He, X. Wang, Y. Cao, J. Zhou, and W. Xu, "Dataset and neural recurrent sequence labeling model for open-domain factoid question answering," *arXiv preprint arXiv:1607.06275*, 2016. 24
[304] N. Peng and M. Dredze, "Named entity recognition for chinese social media with jointly trained embeddings," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 548–554. 24
[305] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom, "Program induction by rationale generation: Learning to solve and explain algebraic word problems," *arXiv preprint arXiv:1705.04146*, 2017. 24
[306] R. Weischedel, S. Pradhan, L. Ramshaw, M. Palmer, N. Xue, M. Marcus, A. Taylor, C. Greenberg, E. Hovy, R. Belvin *et al.*, "Ontonotes release 4.0," *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*, 2011. 24
[307] D. Vilares and C. Gómez-Rodríguez, "Head-qa: A healthcare dataset for complex reasoning," *arXiv preprint arXiv:1906.04701*, 2019. 24
[308] S. L. Blodgett, L. Green, and B. O'Connor, "Demographic dialectal variation in social media: A case study of african-american english," *arXiv preprint arXiv:1608.08868*, 2016. 24
[309] N. Mostafazadeh, N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli, and J. Allen, "A corpus and evaluation framework for deeper understanding of commonsense stories," *arXiv preprint arXiv:1604.01696*, 2016. 24
[310] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández, "The lambada dataset: Word prediction requiring a broad discourse context," *arXiv preprint arXiv:1606.06031*, 2016. 24, 26

[311] B. Hu, Q. Chen, and F. Zhu, "Lcsts: A large scale chinese short text summarization dataset," *arXiv preprint arXiv:1506.05865*, 2015. 24
[312] Z. Shao, M. Huang, J. Wen, W. Xu, and X. Zhu, "Long and diverse text generation with planning-based hierarchical variational model," *arXiv preprint arXiv:1908.06605*, 2019. 24
[313] J. Novikova, O. Dušek, and V. Rieser, "The e2e dataset: New challenges for end-to-end generation," *arXiv preprint arXiv:1706.09254*, 2017. 24
[314] C. Zheng, M. Huang, and A. Sun, "Chid: A large-scale chinese idiom dataset for cloze test," *arXiv preprint arXiv:1906.01265*, 2019. 24
[315] Y. Bisk, R. Zellers, J. Gao, Y. Choi *et al.*, "Piqa: Reasoning about physical commonsense in natural language," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 7432–7439. 24, 26
[316] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension," *arXiv preprint arXiv:1705.03551*, 2017. 24, 26
[317] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018. 24, 26
[318] S. Aroca-Ouellette, C. Paik, A. Roncone, and K. Kann, "Prost: Physical reasoning of objects through space and time," *arXiv preprint arXiv:2106.03634*, 2021. 24
[319] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," *arXiv preprint arXiv:1809.02789*, 2018. 24
[320] T. C. Ferreira, C. Gardent, N. Ilinykh, C. Van Der Lee, S. Mille, D. Moussallem, and A. Shimorina, "The 2020 bilingual, bi-directional webnlg+ shared task overview and evaluation results (webnlg+ 2020)," in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, 2020. 24
[321] C. Xu, W. Zhou, T. Ge, K. Xu, J. McAuley, and F. Wei, "Blow the dog whistle: A chinese dataset for cant understanding with common sense and world knowledge," *arXiv preprint arXiv:2104.02704*, 2021. 24
[322] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, "Race: Large-scale reading comprehension dataset from examinations," *arXiv preprint arXiv:1704.04683*, 2017. 24, 26
[323] E. Choi, H. He, M. Iyyer, M. Yatskar, W.-t. Yih, Y. Choi, P. Liang, and L. Zettlemoyer, "Quac: Question answering in context," *arXiv preprint arXiv:1808.07036*, 2018. 24, 26
[324] M. Geva, D. Khashabi, E. Segal, T. Khot, D. Roth, and J. Berant, "Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 346–361, 2021. 24, 27
[325] J. Boyd-Graber, B. Satinoff, H. He, and H. Daumé III, "Besting the quiz master: Crowdsourcing incremental classification games," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, pp. 1290–1301. 24
[326] S. Zhang, X. Zhang, H. Wang, J. Cheng, P. Li, and Z. Ding, "Chinese medical question answer matching using end-to-end character-level multi-scale cnns," *Applied Sciences*, vol. 7, no. 8, p. 767, 2017. 24
[327] S. Zhang, X. Zhang, H. Wang, L. Guo, and S. Liu, "Multi-scale attentive interaction networks for chinese medical question answer selection," *IEEE Access*, vol. 6, pp. 74 061–74 071, 2018. 24
[328] C. Xu, J. Pei, H. Wu, Y. Liu, and C. Li, "Matinf: A jointly labeled large-scale dataset for classification, question answering and summarization," *arXiv preprint arXiv:2004.12302*, 2020. 24
[329] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, "Winogrande: An adversarial winograd schema challenge at scale," *Communications of the ACM*, vol. 64, no. 9, pp. 99–106, 2021. 24
[330] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?" *arXiv preprint arXiv:1905.07830*, 2019. 24, 26
[331] M. Roemmele, C. A. Bejan, and A. S. Gordon, "Choice of plausible alternatives: An evaluation of commonsense causal reasoning." in *AAAI spring symposium: logical formalizations of commonsense reasoning*, 2011, pp. 90–95. 24
[332] H. Levesque, E. Davis, and L. Morgenstern, "The winograd schema challenge," in *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012. 24, 26
[333] A. Talmor, J. Herzig, N. Lourie, and J. Berant, "Commonsenseqa: A question answering challenge targeting commonsense knowledge," *arXiv preprint arXiv:1811.00937*, 2018. 24, 26
[334] M. Sap, H. Rashkin, D. Chen, R. LeBras, and Y. Choi, "Socialiqa: Commonsense reasoning about social interactions," *arXiv preprint arXiv:1904.09728*, 2019. 24
[335] K. Sun, D. Yu, D. Yu, and C. Cardie, "Investigating prior knowledge for challenging chinese machine reading comprehension," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 141–155, 2020. 24
[336] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. Van Durme, "Record: Bridging the gap between human and machine commonsense reading comprehension," *arXiv preprint arXiv:1810.12885*, 2018. 24
[337] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016. 24, 26
[338] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," *arXiv preprint arXiv:1905.10044*, 2019. 24, 26
[339] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," *arXiv preprint arXiv:1806.03822*, 2018. 24, 26
[340] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, "Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs," *arXiv preprint arXiv:1903.00161*, 2019. 24, 26
[341] I. Dagan, O. Glickman, and B. Magnini, "The pascal recognising textual entailment challenge," in *Machine learning challenges workshop*. Springer, 2005, pp. 177–190. 24, 27
[342] Y. Chang, M. Narang, H. Suzuki, G. Cao, J. Gao, and Y. Bisk, "Webqa: Multihop and multimodal qa," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 495–16 504. 24, 27
[343] Y. Cui, T. Liu, Z. Chen, W. Ma, S. Wang, and G. Hu, "Dataset for the first evaluation on chinese machine reading comprehension," *arXiv preprint arXiv:1709.08299*, 2017. 24
[344] Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. Ma, S. Wang, and G. Hu, "A span-extraction dataset for chinese machine reading comprehension," *arXiv preprint arXiv:1810.07366*, 2018. 24, 27
[345] Y. Cui, T. Liu, Z. Yang, Z. Chen, W. Ma, W. Che, S. Wang, and G. Hu, "A sentence cloze dataset for chinese machine reading comprehension," *arXiv preprint arXiv:2004.03116*, 2020. 24
[346] Y. Li, T. Liu, D. Li, Q. Li, J. Shi, and Y. Wang, "Character-based bilstm-crf incorporating pos and dictionaries for chinese opinion target extraction," in *Asian Conference on Machine Learning*. PMLR, 2018, pp. 518–533. 24
[347] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 252–262. 24
[348] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee *et al.*, "Natural questions: a benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 453–466, 2019. 24
[349] C. C. Shao, T. Liu, Y. Lai, Y. Tseng, and S. Tsai, "Drcd: A chinese machine reading comprehension dataset," *arXiv preprint arXiv:1806.00920*, 2018. 24
[350] W. He, K. Liu, J. Liu, Y. Lyu, S. Zhao, X. Xiao, Y. Liu, Y. Wang, H. Wu, Q. She *et al.*, "Dureader: a chinese machine reading comprehension dataset from real-world applications," *arXiv preprint arXiv:1711.05073*, 2017. 24
[351] H. Tang, J. Liu, H. Li, Y. Hong, H. Wu, and H. Wang, "Dureaderrobust: A chinese dataset towards evaluating the robustness of machine reading comprehension models," *arXiv preprint arXiv:2004.11142*, 2020. 24
[352] J. Welbl, N. F. Liu, and M. Gardner, "Crowdsourcing multiple choice science questions," *arXiv preprint arXiv:1707.06209*, 2017. 24
[353] C. Xiong, Z. Dai, J. Callan, Z. Liu, and R. Power, "End-to-end neural ad-hoc ranking with kernel pooling," in *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, 2017, pp. 55–64. 24
[354] A. Peñas, E. Hovy, P. Forner, Á. Rodrigo, R. Sutcliffe, and R. Morante, "Qa4mre 2011-2013: Overview of question answering for machine reading evaluation," in *Information Access Evaluation. Multilinguality, Multimodality, and Visualization: 4th International Conference of the CLEF Initiative, CLEF 2013, Valencia, Spain, September 23-26, 2013. Proceedings 4*. Springer, 2013, pp. 303–320. 24
[355] S. Lim, M. Kim, and J. Lee, "Korquad1. 0: Korean qa dataset for machine reading comprehension," *arXiv preprint arXiv:1909.07005*, 2019. 24

[356] C. Xiao, H. Zhong, Z. Guo, C. Tu, Z. Liu, M. Sun, Y. Feng, X. Han, Z. Hu, H. Wang *et al.*, "Cail2018: A large-scale legal dataset for judgment prediction," *arXiv preprint arXiv:1807.02478*, 2018. 24

[357] D. Hendrycks, S. Basart, S. Kadavath, M. Mazeika, A. Arora, E. Guo, C. Burns, S. Puranik, H. He, D. Song *et al.*, "Measuring coding challenge competence with apps," *arXiv preprint arXiv:2105.09938*, 2021. 24, 27

[358] Y. Wang, X. Liu, and S. Shi, "Deep neural solver for math word problems," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 845–854. 24, 27

[359] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano *et al.*, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021. 24, 27

[360] J. Austin, A. Odena, M. I. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. J. Cai, M. Terry, Q. V. Le, and C. Sutton, "Program synthesis with large language models," *CoRR*, vol. abs/2108.07732, 2021. 24

[361] F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou *et al.*, "Language models are multilingual chain-of-thought reasoners," *arXiv preprint arXiv:2210.03057*, 2022. 24

[362] S. Roy and D. Roth, "Solving general arithmetic word problems," *arXiv preprint arXiv:1608.01413*, 2016. 24

[363] S.-Y. Miao, C.-C. Liang, and K.-Y. Su, "A diverse corpus for evaluating and developing english math word problem solvers," *arXiv preprint arXiv:2106.15772*, 2021. 24

[364] R. Koncel-Kedziorski, S. Roy, A. Amini, N. Kushman, and H. Hajishirzi, "Mawps: A math word problem repository," in *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1152–1157. 24

[365] A. Patel, S. Bhattamishra, and N. Goyal, "Are nlp models really able to solve simple math word problems?" *arXiv preprint arXiv:2103.07191*, 2021. 24

[366] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021. 24, 27

[367] Y. Lai, C. Li, Y. Wang, T. Zhang, R. Zhong, L. Zettlemoyer, W.-t. Yih, D. Fried, S. Wang, and T. Yu, "Ds-1000: A natural and reliable benchmark for data science code generation," in *International Conference on Machine Learning*. PMLR, 2023, pp. 18 319–18 345. 24

[368] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le *et al.*, "Program synthesis with large language models," *arXiv preprint arXiv:2108.07732*, 2021. 24

[369] Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, and D. Kiela, "Adversarial nli: A new benchmark for natural language understanding," *arXiv preprint arXiv:1910.14599*, 2019. 24, 27

[370] A. Williams, N. Nangia, and S. R. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," *arXiv preprint arXiv:1704.05426*, 2017. 24

[371] R. T. McCoy, E. Pavlick, and T. Linzen, "Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference," *arXiv preprint arXiv:1902.01007*, 2019. 24

[372] J. Liu, L. Cui, H. Liu, D. Huang, Y. Wang, and Y. Zhang, "Logiqa: A challenge dataset for machine reading comprehension with logical reasoning," *arXiv preprint arXiv:2007.08124*, 2020. 24

[373] P. Lewis, B. Oğuz, R. Rinott, S. Riedel, and H. Schwenk, "Mlqa: Evaluating cross-lingual extractive question answering," *arXiv preprint arXiv:1910.07475*, 2019. 24

[374] A. Conneau, G. Lample, R. Rinott, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov, "Xnli: Evaluating cross-lingual sentence representations," *arXiv preprint arXiv:1809.05053*, 2018. 24, 27

[375] Y. Yang, Y. Zhang, C. Tar, and J. Baldridge, "Paws-x: A cross-lingual adversarial dataset for paraphrase identification," *arXiv preprint arXiv:1908.11828*, 2019. 24, 27

[376] S. Narayan, S. B. Cohen, and M. Lapata, "Don't give me the details, just the summary!" *Topic-Aware Convolutional Neural Networks for Extreme Summarization. ArXiv, abs*, 1808. 24

[377] E. M. Ponti, G. Glavaš, O. Majewska, Q. Liu, I. Vulić, and A. Korhonen, "Xcopa: A multilingual dataset for causal commonsense reasoning," *arXiv preprint arXiv:2005.00333*, 2020. 24, 26

[378] A. Tikhonov and M. Ryabinin, "It's all in the heads: Using attention heads as a baseline for cross-lingual transfer in commonsense reasoning," *arXiv preprint arXiv:2106.12066*, 2021. 24

[379] J. H. Clark, E. Choi, M. Collins, D. Garrette, T. Kwiatkowski, V. Nikolaev, and J. Palomaki, "Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 454–470, 2020. 24

[380] T. Scialom, P.-A. Dray, S. Lamprier, B. Piwowarski, and J. Staiano, "Mlsum: The multilingual summarization corpus," *arXiv preprint arXiv:2004.14900*, 2020. 24

[381] S. Lin, J. Hilton, and O. Evans, "Truthfulqa: Measuring how models mimic human falsehoods," *arXiv preprint arXiv:2109.07958*, 2021. 24, 27

[382] I. Augenstein, C. Lioma, D. Wang, L. C. Lima, C. Hansen, C. Hansen, and J. G. Simonsen, "Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims," *arXiv preprint arXiv:1909.03242*, 2019. 24

[383] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, "Fever: a large-scale dataset for fact extraction and verification," *arXiv preprint arXiv:1803.05355*, 2018. 24

[384] I. Mollas, Z. Chrysopoulou, S. Karlos, and G. Tsoumakas, "Ethos: an online hate speech detection dataset," *arXiv preprint arXiv:2006.08328*, 2020. 24, 27

[385] M. Nadeem, A. Bethke, and S. Reddy, "Stereoset: Measuring stereotypical bias in pretrained language models," *arXiv preprint arXiv:2004.09456*, 2020. 24, 27

[386] A. Parrish, A. Chen, N. Nangia, V. Padmakumar, J. Phang, J. Thompson, P. M. Htut, and S. R. Bowman, "Bbq: A hand-built bias benchmark for question answering," *arXiv preprint arXiv:2110.08193*, 2021. 24

[387] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K.-W. Chang, "Gender bias in coreference resolution: Evaluation and debiasing methods," *arXiv preprint arXiv:1804.06876*, 2018. 24

[388] N. Nangia, C. Vania, R. Bhalerao, and S. R. Bowman, "Crows-pairs: A challenge dataset for measuring social biases in masked language models," *arXiv preprint arXiv:2010.00133*, 2020. 24

[389] S. Gehman, S. Gururangan, M. Sap, Y. Choi, and N. A. Smith, "Realtoxicityprompts: Evaluating neural toxic degeneration in language models," *arXiv preprint arXiv:2009.11462*, 2020. 24

[390] D. Borkan, L. Dixon, J. Sorensen, N. Thain, and L. Vasserman, "Nuanced metrics for measuring unintended bias with real data for text classification," in *Companion proceedings of the 2019 world wide web conference*, 2019, pp. 491–500. 24

[391] O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, V. Logacheva, C. Monz *et al.*, "Findings of the 2016 conference on machine translation," in *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, 2016, pp. 131–198. 24

[392] B. Loïc, B. Magdalena, B. Ondřej, F. Christian, G. Yvette, G. Roman, H. Barry, H. Matthias, J. Eric, K. Tom *et al.*, "Findings of the 2020 conference on machine translation (wmt20)," in *Proceedings of the Fifth Conference on Machine Translation*. Association for Computational Linguistics,, 2020, pp. 1–55. 24

[393] W. Li, F. Qi, M. Sun, X. Yi, and J. Zhang, "Ccpm: A chinese classical poetry matching dataset," *arXiv preprint arXiv:2106.01979*, 2021. 24

[394] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, "Wizard of wikipedia: Knowledge-powered conversational agents," *arXiv preprint arXiv:1811.01241*, 2018. 24

[395] H. Rashkin, E. M. Smith, M. Li, and Y.-L. Boureau, "Towards empathetic open-domain conversation models: A new benchmark and dataset," *arXiv preprint arXiv:1811.00207*, 2018. 24

[396] E. Dinan, V. Logacheva, V. Malykh, A. Miller, K. Shuster, J. Urbanek, D. Kiela, A. Szlam, I. Serban, R. Lowe *et al.*, "The second conversational intelligence challenge (convai2)," in *The NeurIPS'18 Competition: From Machine Learning to Intelligent Conversations*. Springer, 2020, pp. 187–208. 24

[397] H. Zhou, C. Zheng, K. Huang, M. Huang, and X. Zhu, "Kdconv: A chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation," *arXiv preprint arXiv:2004.04100*, 2020. 24

[398] L. CO, "Iflytek: a multiple categories chinese text classifier. competition official website," 2019. 24

[399] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019. 25, 29

[400] J. Baumgartner, S. Zannettou, B. Keegan, M. Squire, and J. Blackburn, "The pushshift reddit dataset," in *Proceedings of the international AAAI conference on web and social media*, vol. 14, 2020, pp. 830–839. 25

[401] A. Fan, Y. Jernite, E. Perez, D. Grangier, J. Weston, and M. Auli, "Eli5: Long form question answering," *arXiv preprint arXiv:1907.09190*, 2019. 26

[402] Y. Wang, S. Mishra, P. Alipoormolabashi, Y. Kordi, A. Mirzaei, A. Arunkumar, A. Ashok, A. S. Dhanasekaran, A. Naik, D. Stap *et al.*, "Benchmarking generalization via in-context instructions on 1,600+ language tasks," *arXiv preprint arXiv:2204.07705*, 2022. 26

[403] T. Xie, C. H. Wu, P. Shi, R. Zhong, T. Scholak, M. Yasunaga, C.-S. Wu, M. Zhong, P. Yin, S. I. Wang *et al.*, "Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models," *arXiv preprint arXiv:2201.05966*, 2022. 26

[404] Q. Ye, B. Y. Lin, and X. Ren, "Crossfit: A few-shot learning challenge for cross-task generalization in nlp," *arXiv preprint arXiv:2104.08835*, 2021. 26

[405] V. Aribandi, Y. Tay, T. Schuster, J. Rao, H. S. Zheng, S. V. Mehta, H. Zhuang, V. Q. Tran, D. Bahri, J. Ni *et al.*, "Ext5: Towards extreme multi-task scaling for transfer learning," *arXiv preprint arXiv:2111.10952*, 2021. 26

[406] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1112–1122. [Online]. Available: https://aclanthology.org/N18-1101 27

[407] Y. Zhang, J. Baldridge, and L. He, "PAWS: Paraphrase adversaries from word scrambling," 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 1298–1308. [Online]. Available: https://aclanthology.org/N19-1131 27

[408] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh *et al.*, "Mixed precision training," *arXiv preprint arXiv:1710.03740*, 2017. 28

[409] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," *CoRR*, vol. abs/1910.05895, 2019. 29

[410] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019. 30

[411] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 2021, pp. 610–623. 30

[412] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021. 30

[413] M. Tänzer, S. Ruder, and M. Rei, "Memorisation versus generalisation in pre-trained language models," *arXiv preprint arXiv:2105.00828*, 2021. 30

[414] S. M. West, M. Whittaker, and K. Crawford, "Discriminating systems," *AI Now*, pp. 1–33, 2019. 30

[415] K. Valmeekam, A. Olmo, S. Sreedharan, and S. Kambhampati, "Large language models still can't plan (a benchmark for llms on planning and reasoning about change)," *arXiv preprint arXiv:2206.10498*, 2022. 30

[416] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen *et al.*, "Siren's song in the ai ocean: A survey on hallucination in large language models," *arXiv preprint arXiv:2309.01219*, 2023. 30

[417] A. Webson and E. Pavlick, "Do prompt-based models really understand the meaning of their prompts?" *arXiv preprint arXiv:2109.01247*, 2021. 30

[418] O. Shaikh, H. Zhang, W. Held, M. Bernstein, and D. Yang, "On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning," *arXiv preprint arXiv:2212.08061*, 2022. 30

[419] X. Liu, H. Cheng, P. He, W. Chen, Y. Wang, H. Poon, and J. Gao, "Adversarial training for large neural language models," ArXiv, April 2020. [Online]. Available: https://www.microsoft.com/en-us/research/publication/adversarial-training-for-large-neural-language-models/ 30

[420] E. Shayegani, M. A. A. Mamun, Y. Fu, P. Zaree, Y. Dong, and N. Abu-Ghazaleh, "Survey of vulnerabilities in large language models revealed by adversarial attacks," 2023. 30

[421] X. Xu, K. Kong, N. Liu, L. Cui, D. Wang, J. Zhang, and M. Kankanhalli, "An llm can fool itself: A prompt-based adversarial attack," 2023. 30

[422] H. Zhao, H. Chen, F. Yang, N. Liu, H. Deng, H. Cai, S. Wang, D. Yin, and M. Du, "Explainability for large language models: A survey," 2023. 30

[423] S. Huang, S. Mamidanna, S. Jangam, Y. Zhou, and L. H. Gilpin, "Can large language models explain themselves? a study of llm-generated self-explanations," 2023. 30

[424] H. Brown, K. Lee, F. Mireshghallah, R. Shokri, and F. Tramèr, "What does it mean for a language model to preserve privacy?" in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022, pp. 2280–2292. 31

[425] R. Plant, V. Giuffrida, and D. Gkatzia, "You are what you write: Preserving privacy in the era of large language models," *arXiv preprint arXiv:2204.09391*, 2022. 31

[426] W. Niu, Z. Kong, G. Yuan, W. Jiang, J. Guan, C. Ding, P. Zhao, S. Liu, B. Ren, and Y. Wang, "Real-time execution of large-scale language models on mobile," 2020. 31

[427] C. Guo, J. Tang, W. Hu, J. Leng, C. Zhang, F. Yang, Y. Liu, M. Guo, and Y. Zhu, "Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization," in *Proceedings of the 50th Annual International Symposium on Computer Architecture*, 2023, pp. 1–15. 31

[428] B. Meskó and E. J. Topol, "The imperative for regulatory oversight of large language models (or generative ai) in healthcare," *npj Digital Medicine*, vol. 6, no. 1, p. 120, 2023. 31

[429] J. Zhang, X. Ji, Z. Zhao, X. Hei, and K.-K. R. Choo, "Ethical considerations and policy implications for large language models: Guiding responsible development and deployment," *arXiv preprint arXiv:2308.02678*, 2023. 31

[430] J. Mökander, J. Schuett, H. R. Kirk, and L. Floridi, "Auditing large language models: a three-layered approach," *AI and Ethics*, pp. 1–31, 2023. 31