A Literature Survey on Open Source Large Language Models

Sanjay Kukreja
Student, SP Jain School of Global Management
sanjay.ds18dba008@spjain.org

Tarun Kumar
Program Manager, eClerx Services Ltd.
tk7ua1678@gmail.com

Amit Purohit
Sr. Process Manager, eClerx Services Ltd.
amit.iitgn@gmail.com

Dr. Abhijit Dasgupta
Assistant Professor and Director, SP Jain School of Global Management
abhijit.dasgupta@spjain.org

Dr. Debashis Guha
Associate Professor and Director, SP Jain School of Global Management
debashis.guha@spjain.org

ABSTRACT

Since the 1950s, post the Turing test, humans have been striving hard to make machines learn the art of mastering linguistic intelligence. Language being a complex and intricate tool of expression used by humans, poses a large number of challenges for AI enabled algorithms to grasp its understanding in entirety. Over the past few years, a chain of efforts have been made to make machines understand linguistic intricacies. Small scale models such as BERT and pre-trained language models (PLMs) have demonstrated strong capabilities in understanding and solving various language based tasks. Over the period of years, it is also observed that by increasing the parameters scale to larger size, large language models show a significant improvement in performance and showcase abilities to understand context. For the PLMs of a humongous size i.e in the tune of tens or hundreds of billions of parameters, and to understand the large parametric scales, the scientific community introduced the term LLMs - large language models. The whole world witnessed the launch and quick adoption of ChatGPT, an AI chatbot built on LLMs. As the usage of AI algorithms changes the way the scientific community, society and industry works, it is imperative to review the advances of LLMs. Since 2022, almost daily nearly a dozen LLMs are released. These LLMs are categorized as open and closed source. This paper aims to focus on major aspects of open source LLMs - pre-training covering data collection and pre-processing, model architecture and training. We will select open source models released in June, July and August 2023 with training parameters greater than 70 billion parameters and provide a comprehensive survey on the mentioned aspects. As new models are released on daily / weekly basis in the LLM space, in order to keep the survey concise and targeted to important models, we chose to select time-box of 3 months and a large parameter range of 70 billion in our literature survey. We will also cover historical evolution of LLMs and list open items for future directions.

ICCMB 2024, January 12–14, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-1665-2/24/01
https://doi.org/10.1145/3647782.3647803

CCS CONCEPTS

• **Computing methodologies** → Artificial intelligence; Natural language processing; Natural language generation.

KEYWORDS

Open Source LLMs, Large Language Models, Generative AI

ACM Reference Format:
Sanjay Kukreja, Tarun Kumar, Amit Purohit, Dr. Abhijit Dasgupta, and Dr. Debashis Guha. 2024. A Literature Survey on Open Source Large Language Models. In *2024 7th International Conference on Computers in Management and Business (ICCMB 2024), January 12–14, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3647782.3647803

1 INTRODUCTION

Language is the light of the mind - - - John Stuart Mill. The ability to express and communicate with each other universally lies in the magic of language and its semantics. Since the human race has evolved, language has evolved across cultures and demographics. Every human, right from their childhood, attempts to grasp linguistic capabilities for communicating in the form of language. There has been long standing research by the scientific community to enable machines to communicate, understand, read, write and have linguistic capabilities just like humans. Language modeling (LM) is one of the approaches towards overcoming the challenge of imparting linguistic capabilities to machines. This paper covers developments in language modeling extensively. One of the aims of LM is to understand in-context language by predicting the probabilities of future / missing tokens [1]. The four major developmental stages of LM are shown in Table 1.

The term large language models (LLMs) is coined by the research community to distinguish large sized PLMs [4-7]. Large Language Models (LLMs) are Transformer language models that contain hundreds of billions (or more) of parameters, which are trained on massive text data [4]. A novel application of LLMs is ChatGPT by Open AI. There has been an astronomical rise in the research papers and materials that mention ChatGPT at arXIv [1]. With the advent of ChatGPT - 4 and its multifaceted applications across various business and technical domains, it is pegged as the initial days of how AGI will look in the future [10]. ChatGPT is disrupting the way output of traditional search engines is consumed, leading companies like Microsoft to launch Co-Pilot that is used with its search engine offering Bing. In the area of computer vision, researchers are now training LLMs to be capable of handling visual

Table 1: Four major developmental stages of LM

Model Name	Brief description
Statistical Language Models (SLM)	Statistical Language Models (SLMs) are computational models used in natural language processing (NLP) and machine learning to understand and generate human language text [2]. They rely on statistical probability distributions to model the relationships between words in a given language.
Neural Language Models (NLM)	Neural Language Models are a type of language model that uses neural networks, particularly deep learning models, to understand and generate human language. These models employ techniques such as word embeddings, recurrent neural networks (RNNs), or transformer architectures to capture complex language patterns and relationships [3]. NLMs have become the foundation for various NLP tasks, including machine translation, text generation, and sentiment analysis, due to their ability to handle context and semantics effectively.
Pre-trained Language Model (PLM)	A Pre-trained Language Model (PLM) is a type of machine learning model that has been trained on a vast amount of text data to learn the patterns, structure, and relationships within natural language. These models are trained on diverse and extensive text corpora, often consisting of internet text, books, articles, and other sources, to capture a broad understanding of language.
Large Language Models (LLM)	An LLM is a type of artificial intelligence model, typically based on deep learning techniques, that is designed to understand and generate human language text [8]. These models are characterized by having a large number of parameters, which enables them to capture complex language patterns and relationships.

/ audio information as input and output as both text and audio / visual information. Platforms such as Midjourney are already capable of AI generated images leading. All the progress made by the evolution of LLMs and ChatGPT also poses several challenges. These challenges are not limited to lack of basic principles used for development of LLMs, training of such LLMs, usage of public / private data for training and the cost associated with training of LLMs. LLMs are costly to train making it difficult for the research community to train LLMs and majority of the training of LLMs is industry led; the training data collection and cleaning methodologies used for training are never made public. It is important to study how and when LLMs gain such capabilities [9]. LLMs can produce toxic, inaccurate, fictitious and harmful results. It requires effective and efficient control approaches to eliminating the potential risk of the use of LLMs [10]. This paper covers working of LLMs, how open source LLMs are balancing the order vis-à-vis closed source LLMs, latest open source LLMs launched in 2022-2023 with key distinctions. Further the survey covers 3 leading open source LLMs namely Falcon, BLOOM and Llama-2 and covers the data collection & pre-processing, model architecture and training aspects. A summarized view of the applications of LLMs in several representative domains. items for future directions of the work are listed at the end.

2 HOW DOES A LLM WORK?

Large Language Models (LLMs) can be classified in three broad parts based on the underlying architecture namely transformer based, RNN driven and other novel architecture designs. In order to read text and generate language based output like humans, many of the leading LLMs use transformed based models. Vaswani et al. published a paper, "Attention is All You Need," in 2017 and provided more insights about the transformer models based on attention mechanism. With the help of an attention mechanism, a LLM can understand the context of a group of words, sentences or a paragraph instead of understanding a given word in isolation. This enables the model to provide relevant and better outputs like a human being. The transformer model works based on an encoder - decoder architecture [11]. For text processing, one needs to first tokenize a sequence of words, provided as input. Such tokens are encoded in numeric format and transformed into embedding. The tokens are represented as Vector-space representations which retain the meaning of the sequence of words. Further, the encoder converts the embedding of all such tokens in a context vector [12].

3 OPEN AND CLOSE SOURCE LLMS

As the name suggests, the term open source means something that is available to the public and is open for modification as its core design is accessible publicly. In the context of software engineering, the term open source is used to designate a particular approach to create software programs. Open source software programs and initiatives are now used globally to promote transparency and community oriented software development. On the other hand, closed source is something that is not available to the public. Closed source software is proprietary software and follows controlled protocols for any modification. In the world of LLMs, we have open source LLMs and closed source LLMs. Open source LLMs are freely accessible, open for modification and distribution. The scientific community can contribute to the improvement of such modes. This fosters innovation, sharing of knowledge and collaborative development of the LLMs. Closed source LLMs are developed, updated, distributed and trained by one or a consortium of large organizations. The underlying code is proprietary in nature and not accessible to the public. Such LLMs are developed in the form of license based software products that require subscriptions for usage. Key difference

Table 2: Difference between open and closed source models based on key aspects

Aspects	Open Source LLMs	Closed Source LLMs
Accessibility	Available to public	Not available to public
Customization	Possible	Not possible
Community Contributions	Highly dependent on efforts and contributions of citizen developers	Not applicable
Licensing & Costs	None	Depends on utilization

between open and closed source models based on key aspects follow as showing in Table 2.

4 OPEN SOURCE LLMS: BALANCING THE ORDER

The training cost of a large language model is humongous in nature. With closed LLMs, corporates and large business houses are spending millions of dollars on daily basis for training and up-keep of such models. The training data used, its implications, underlying risks of repurposing of data and placement of output is unclear and less transparent. This leads to a potent risk of biased recommendations or fabricated LLM output that may be used in direct or indirect favor of the model training conglomerate. With a new LLM released almost every week of 2023, it is imperative to balance the order of such powerful technology with open source LLMs. Apart from being accessible to the scientific and tech community, open source models have a lot more to offer to the emerging space of LLMs. Key aspect are covered below.

Transparency: For enterprises lacking in-house machine learning expertise, the solution lies in the utilization of open source Large Language Models (LLMs). These LLMs grant organizations the capacity to maintain full control over their data, regardless of whether they choose to operate within the cloud or on their own premises. This control serves as a safeguard against the potential hazards associated with data leakage or unauthorized access. The advantages of employing an open source LLM extend to its inherent transparency, offering insights into its operational mechanisms, architecture, training data, and methodologies, as well as its utilization. The ability to scrutinize the underlying code and gain visibility into the algorithms employed instills a higher level of trust for enterprises. Furthermore, it aids in facilitating audits and ensures that ethical and legal standards are upheld. Additionally, the efficient optimization of an open source LLM can lead to reduced latency and improved overall performance.

Cost Savings: With emerging tech companies and businesses as well as corporates in developing countries, look forward to embracing LLMs as part of their success journey, cost of training and model building is a huge blocker to do the same. In the grand scheme of things, open source LLMs are often considerably more cost-effective than their proprietary counterparts due to the absence of licensing fees. Nonetheless, the overall expenses associated with running an LLM encompass the infrastructure costs, whether in the cloud or on-premises, which usually entail a substantial initial deployment expenditure.

Enablement of new businesses as open source LLMs are transparent in providing training data, model architecture and approach towards performance, it is now becoming a key ingredient in developing businesses of tech companies that are emerging in nature or wanting to launch themselves in the AI space. Growth of businesses such as chat bot set-up, code generation, content generation, image / video generation and content repurposing are fueled by open source LLMs. It is observed that large financial organizations and banks in developing countries are harnessing open source LLMs to create better customer experience and improved target marketing.

Less Risk, More Trust: Data security and cross border data transfer protocols are key pillars of success to any global business. Businesses across sectors are spending millions to ensure zero data breaches and neutralize / monitor cyber-attacks on a daily basis. This becomes applicable to the world of LLMs. Closed LLMs are less transparent in terms of divulging training data and repurposing of outputs generated while usage poses a threat to businesses that thrive on data marketing and dealing in business processing of confidential data. Applications / models using Open AI APIs for output generation are skeptical of continuity and data sharing protocols. This gives open source LLMs a leading edge as open source models can be hosted on a private / hybrid cloud keeping data ownership with the business and data operations while handling confidential data can be seamlessly covered with trust and continual support.

5 RISE OF OPEN SOURCE LLMS

The world has witnessed a surge in open source LLMs in the last 9 months. Open Source LLMs are critical to the quick adoption of AI in everyday life. The openness and flexibility of open-source Large Language Models (LLMs) provide benefits compared to closed-source LLMs that are often opaque in explaining how they arrived at their conclusions. Open source LLMs enable a large section of citizen developers to participate in the community development of LLMs. Currently open source LLMs are at a nascent stage, but picking pace more quickly than anticipated. Some of the notable Open source LLMs of the year 2023 follow.

5.1 Mistral - 7B | Developed by: Mistral AI

Mistral 7B is introduced as a fundamental model, underscoring its pivotal role as a cornerstone in the field of natural language processing. It forms an integral part of the continually evolving landscape of large language models. Despite its relatively compact size when compared to larger models, Mistral 7B impressively demonstrates its prowess across a wide spectrum of tasks related to natural language understanding and generation. It excels in various domains, including text summarization, text classification, text completion, and code completion. What sets Mistral 7B apart is its distinctive architectural approach. The model employs an innovative design,

which, while not fully detailed, suggests a unique design philosophy in contrast to well-established models such as GPT-3. It leverages Grouped-query attention (GQA) to enhance inference speed and employs Sliding Window Attention (SWA) to efficiently manage longer sequences at a reduced computational expense [13]. This innovative approach yields exceptionally high performance for a model of its relatively modest size.

5.2 Deci LM | Developed by: Deci Inc, Israel

DeciLM's exclusive decoder-only transformer architecture incorporates a distinctive implementation of variable Grouped-Query Attention (GQA). In contrast to other transformer models, such as Llama 2 70B, that maintain consistent attention groups per transformer layer, DeciLM introduces variation in the number of attention groups, keys, and values across different transformer layers. To the best of our knowledge, DeciLM represents the inaugural language model in which the transformer layers are not exact structural replicas of one another. The architecture of DeciLM was crafted using Deci's proprietary Neural Architecture Search (NAS) engine, known as AutoNAC. Traditional NAS methods, while promising, tend to be computationally demanding. AutoNAC confronts this challenge by automating the search process in a computationally efficient manner. This engine has played a crucial role in generating a diverse range of highly efficient foundational models, including the cutting-edge object detection model YOLO-NAS, the text-to-image model DeciDiffusion, and the code generation LLM, DeciCoder. In the case of DeciLM, AutoNAC played a pivotal role in selecting the optimal GQA group parameters for each individual transformer layer within the model.

5.3 Falcon | Developed by: Technology Innovation Institute (TII), UAE

Fundamentally, the Falcon LLM stands out as a language model renowned for its impressive performance and remarkable scalability. Its training was executed using a vast web dataset named RefinedWeb, which was supplemented with carefully curated sources. Notably, the model's unique attribute of multi-query attention greatly bolsters its scalability, enabling it to efficiently handle extensive tasks. However, the uniqueness of the Falcon LLM extends beyond this point. It was meticulously crafted using custom tools and a distinctive data pipeline, ensuring the quality of the training data. Falcon 40B's multilingual capabilities extend to English, German, Spanish, and French, and it also demonstrates basic proficiency in several other languages, such as Italian, Portuguese, Polish, Dutch, Romanian, Czech, and Swedish. The Falcon LLM family consists of two variations: Falcon-40B and Falcon-7B.

5.4 BLOOM | Developed by: HuggingFace, EleutherAI, and other 250+ institutions

In contrast to traditional models, BLOOM utilizes 46 natural world languages and 13 programming languages in its operation. Its distinctive architecture permits extensive fine-tuning for a diverse range of downstream tasks. BLOOM's objective is to enhance worldwide scientific progress by promoting interdisciplinary research and providing scholars with the means to leverage its capabilities across various applications, thus democratizing AI-powered research and innovation. This groundbreaking model has the potential to redefine the landscape of scientific exploration and cooperation.

5.5 LLAMA 2 | Developed by: Meta Inc & Microsoft

Llama 2 underwent training using a wide variety of internet text and image data. Its architectural framework incorporates ideas from its predecessor, Llama 1, and enhances upon them. Llama 2 can be employed for various tasks, including generating comprehensive responses to both textual and image inputs, enabling interactive storytelling, providing answers to questions based on images, and more. Additionally, it holds promise for applications in content creation, research, and entertainment.

5.6 Qwen - 14 B | Developed by: Alibaba

It is the most potent open-source model considering its compact size. Also, it holds the record for the most extensive training, with 3 trillion tokens. Available in five distinct variations: Base, Chat, Code, Math and Vision. Similar to numerous other multilingual Large Language Models (LLMs), specific language tokens had to be incorporated into their tokenizer.

5.7 OPT - 175 B | Developed by: Meta AI

The Open Pre-trained Transformer (OPT) model represents a substantial achievement from Meta AI, contributing to the democratization of large-scale language models. Its most formidable iteration, OPT-175B, stands out with an immense parameter count of 175 billion. OPT underwent training on unannotated text data that primarily comprises English sentences, equipping it with the capability to comprehend and produce human-like text across diverse domains.

5.8 GPT-NeoX-20B | Developed by: EleutherAI

GPT-NeoX-20B, created by EleutherAI, stands as one of the leading open-source large language models with a parameter count of 20 billion. Its training data source is the Pile dataset, an open-source collection of 886 gigabytes of language modeling data divided into 22 smaller datasets. The Pile dataset encompasses a wide array of text origins, including books, Wikipedia, GitHub, and Reddit. This model is constructed based on the GPT-3 architecture but introduces advancements such as synchronous data parallelism and gradient check pointing, among others. GPT-NeoX-20B employs autoregressive language modeling, predicting the next word in a text to improve comprehension and generate coherent responses.

5.9 GPT-J | Developed by: EleutherAI

GPT-J, another advanced language model developed by EleutherAI, has been trained on the extensive Pile dataset. It stands out as an autoregressive, decoder-only transformer model, primarily tailored for addressing natural language processing tasks. Despite its remarkable capabilities, GPT-J comprises 6 billion parameters, which is significantly more compact compared to GPT-3's 175 billion parameters. GPT-J is fundamentally rooted in the GPT-2 architecture, with a notable distinction lying in the incorporation of parallel decoders. This architectural enhancement allows GPT-J to process

multiple tokens or text segments concurrently during training. Consequently, GPT-J leverages distributed computing resources, such as multiple GPUs or TPUs, to expedite the model training process.

5.10 Baichuan - 13 B | Developed by: Baichuan Inc

Baichuan Inc., a pioneering search engine company in China, has introduced an open-source large language model called Baichuan-13B, positioning itself as a competitor to OpenAI. Boasting a substantial model size of 13 billion parameters, Baichuan-13B aims to provide businesses and researchers with advanced English and Chinese AI language processing and generation capabilities. The model's pre-training dataset encompasses a vast 1.3 trillion tokens. Baichuan-13B empowers users with functionalities such as text generation, summarization, translation, and more. This endeavor follows Baichuan's previous success with Baichuan-7B and aligns perfectly with the company's overarching mission to democratize generative AI, making it more accessible and practical for a wider audience.

5.11 CodeGen | Developed by: Salesforce AI research

CodeGen, a remarkable achievement developed by researchers at Salesforce AI Research, builds upon the GPT-3.5 architecture's foundation. This groundbreaking model is available in various sizes, ranging from 350 million to an impressive 16 billion parameters. The training data for CodeGen comprises a diverse array of programming languages and frameworks, including code snippets sourced from GitHub and Stack Overflow. This extensive dataset plays a pivotal role in equipping CodeGen with a profound understanding of programming concepts and establishing connections between code and natural language. As a result, the model can generate precise and dependable code solutions when provided with straightforward English prompts. CodeGen has garnered significant attention for its potential to streamline software development processes and elevate developer efficiency.

5.12 BERT | Developed by: Google AI

BERT, which stands for "Bidirectional Encoder Representations from Transformers," emerged from the efforts of Google AI researchers. This powerful model boasts an expansive parameter count of up to 340 million and has undergone training on a diverse dataset containing a staggering 3.3 billion words, sourced from resources like BookCorpus and Wikipedia. What sets BERT apart is its groundbreaking approach to context comprehension. Unlike its predecessors that processed text in a linear fashion, BERT reads sentences in bidirectional manner, simultaneously assimilating nuanced contextual relationships. Throughout its training, BERT employs word masking, learning to predict these masked words, a technique that fosters a profound grasp of context. This innovation has had a transformative impact on various Natural Language Processing (NLP) tasks, consistently achieving state-of-the-art results in domains such as question answering and sentiment analysis, among others.

5.13 T5 | Developed by Google AI

T5, which stands for Text-To-Text Transfer Transformer, is a versatile pre-trained language model created by Google AI researchers. Built upon the Transformer architecture, T5 is specifically engineered to tackle a broad spectrum of natural language processing tasks through a unified "text-to-text" framework. It offers a range of model sizes, spanning from small to extra-large, with the largest configuration boasting an impressive 11 billion parameters. T5's training process drew from the vast and diverse Colossal Clean Crawled Corpus (C4) dataset, encompassing multiple languages, including English, German, French, and Romanian. What sets T5 apart is its innovative approach to task handling, as it reformulates various tasks into a consistent text-to-text format. This approach allows T5 to produce results across a multitude of applications, including translation, summarization, classification, and more, by treating each task as a text-generation challenge.

In this paper, we will now take up open source LLMs as our area of study and focus on Falcon, LLAMA 2 and BLOOM, as they are one of the most enterprise grade open source LLMs. As part of our literature survey, we will focus on four important aspects namely data collection & pre-processing. Model architecture, model training.

Pre-training serves as the foundation for the proficiency of Large Language Models (LLMs). Through extensive pre-training on vast textual datasets, LLMs acquire fundamental language comprehension and generation skills [14, 15]. The magnitude and quality of the pre-training dataset play a pivotal role in empowering LLMs with robust capabilities. Additionally, the effective pre-training of LLMs necessitates well-crafted model structures, acceleration strategies, and optimization methods. In the subsequent sections, we delve into the processes of data collection and processing, introduce commonly utilized model architectures, model training techniques, model utilization and evaluation.

6 DATA COLLECTION AND PRE-PROCESSING

6.1 FALCON

6.1.1 Data Sources. Falcon 40B underwent training on an extensive dataset comprising 1 trillion tokens sourced from REDEFINEDWEB, an internet corpus carefully filtered for quality, in addition to incorporating supplementary curated datasets [39]. Rather than amassing disparate curated sources, TII (The training institute) elevated the quality of the web data by implementing large-scale deduplication and rigorous filtering processes [39]. This approach culminated in the creation of a top-tier training dataset, which significantly contributes to the exceptional performance exhibited by Falcon models. It is demonstrated that web data only can result in modes outperforming both private and publicly available corpora and challenge current dominant views about data quality [39].

6.1.2 Data Pre-processing. MDR (MacroData Refinement), a comprehensive pipeline designed for the extensive filtering and deduplication of web data sourced from CommonCrawl is introduced as data processing techniques for Falcon LLM. Through the utilization of MDR, on REFINEDWEB, an English pre-training dataset encompassing an impressive five trillion tokens, exclusively derived from web data was achieved [39]. Data pre-processing also capitalizes on

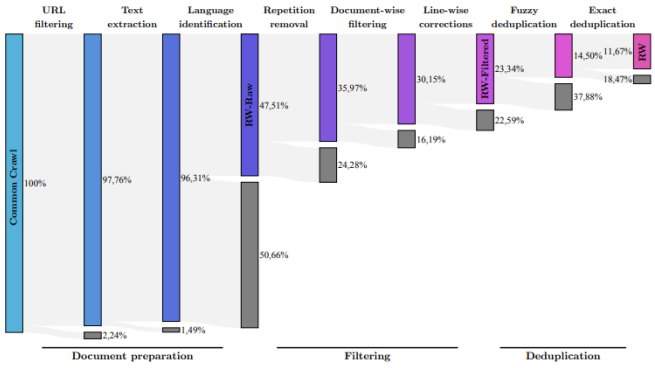

Figure 1: Macrodata refinement stages [39]

rigorous filtering and stringent deduplication techniques to elevate the overall quality of web data, resulting in a corpus that attains the same level of quality as desired.

Deduplication: Taking inspiration from the research conducted by Lee et al. in 2022 [40], which underscored the significance of deduplication in the context of large language models, an uncompromising deduplication process was established. This approach encompasses a combination of both exact and fuzzy deduplication techniques, executed with stringent parameters that result in removal rates surpassing those reported by other models [39].

URL filtering: In order to prevent the introduction of additional unwanted biases into the model, as cautioned by Dodge et al. in 2021 and noted by Welbl et al. in 2021 [41] employing machine learning-based filtering methods except for language identification was avoided. URL filtering based on straightforward rules and heuristics to recognize and exclude adult content was employed. Other pre-processing techniques include document wise fileting, repetition removal, exact and fuzzy deduplication. Figure 1 represents macrodata refinement stages covering document preparation, filtering and deduplication.

6.2 BLOOM

6.2.1 Data Sources. BLOOM was trained on the ROOTS corpus [16], which is a comprehensive amalgamation of 498 Hugging Face datasets [17]. This corpus is vast, containing 1.61 terabytes of text that spans 46 natural languages and 13 programming languages. For more details on the data set refer Lhoest et al., 2021 [17]. The data's diversity is evident in its detailed list showcasing every language, its linguistic genus, family, and macroarea. The motivation behind such a diverse and extensive dataset was to bridge the gap between developers and the users of the technology. There has been a historical disconnect, especially in the curation of datasets for large-scale machine learning projects. Often, the focus has been on accumulating "high-quality" data at minimal costs, sometimes at the expense of marginalized populations. The BigScience workshop aimed to rectify this by emphasizing human involvement, local expertise, and language expertise in the data curation process. Detailed social construction of The BigScience workshop and understanding coverage of inclusivity in training the large language model are available at Akiki et al. (2022) [18].

6.2.2 Data Pre-processing. After identifying the data sources, the next step was to process this data. This involved several steps to ensure the data's quality and relevance. Initially, the data was obtained from various sources, including NLP datasets, archives, and websites. Once gathered, a significant challenge was to filter out non-natural language content, such as SEO pages, spam, or preprocessing errors. The goal was to ensure the text was genuinely aligned with the human chain of thoughts [27] and is meant to be written by humans and for humans [19].

This filtering was done using quality indicators, which were adapted to each source's specific needs. After filtering, the data underwent deduplication to remove near-identical documents [19]. Additionally, any personal identifiable information was redacted to ensure privacy, especially from sources like OSCAR, version 21.09 corresponding to the February 2021 snapshot of the Common Crawl [20] which presented higher privacy risks. OSCAR ended with 38% of the Corpus [19].

6.3 LLAMA 2

6.3.1 Data Sources. Llama 2 is an updated version of Llama 1 [29], trained on a new mix of publicly available data. The data pipeline creation of Corpus is shown in 2 2. Primary sources included English CommonCrawl, C4, GitHub, Wikipedia, Gutenberg, Books3, ArXiv, and Stack Exchange [30]. The size of the pre-training corpus was increased by 40%, which doubled the context length of the model, and adopted grouped-query attention [31].

6.3.2 Data Pre-processing. Most of the data pre-processing techniques were adapted from Llama 1 [29]. The data underwent robust cleaning processes. The training corpus was up-sampled from the most factual sources to enhance the model's knowledge and reduce hallucinations. The total data consisted of 2 trillion tokens, which was chosen for its balance between performance and cost [30]. The data did not include any from Meta's products or services. Efforts were made to exclude data from sites known to contain significant personal information [30].

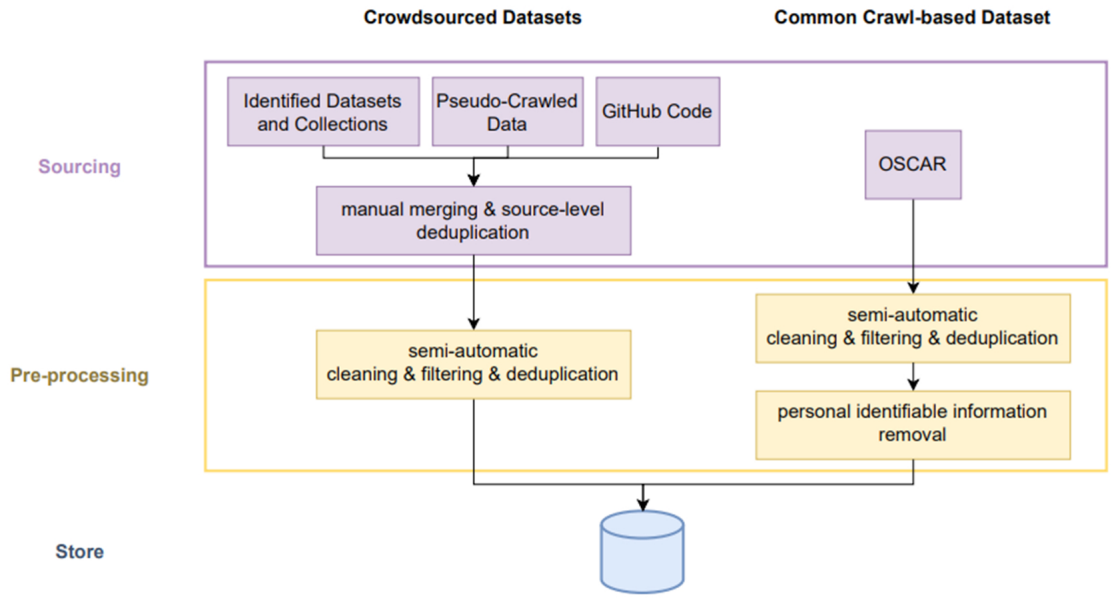

Figure 2: Data pipeline creation of Corpus [19]

7 MODEL ARCHITECTURE

7.1 FALCON

Furthermore, the model incorporates various improvements, such as rotary positional embeddings [35], similar to LLAMA - 2. Additionally, Falcon also integrates multi-query attention, contributing significantly to its remarkable performance. Multi-query attention (MQA), which utilizes only a single key-value head, brings about a substantial acceleration in decoder inference [42]. In a departure from the typical practice of employing static matrices to retain positional values, the model encodes absolute positional information of tokens using a rotation matrix. This unique approach to positional embeddings, known as rotational positional embeddings, endows Falcon-40B with enhanced flexibility in handling sequences of tokens. Research indicates that rotational positional embeddings surpass other methods across various architectural designs [35]. Furthermore, the computational overhead associated with this technique remains largely inconsequential, even when applied to every layer within the LLM.

A pioneering feature within the Falcon LLM model architecture is the introduction of flash attention, an innovative attention algorithm that excels in both speed and memory efficiency for Transformers. Flash Attention achieves this by minimizing the volume of memory reads/writes between the GPU's high bandwidth memory (HBM) and on-chip SRAM, accomplished through tiling [39]. The result is an expedited Transformer training process, accommodating extended contextual information, ultimately culminating in the creation of superior models and heightened performance across a diverse array of tasks [25].

7.2 BLOOM

The BLOOM model's architecture is a blend of existing and innovative practices. While the design space for architectures is vast, BLOOM strikes a balance, focusing on scalable models with public tool support. The BLOOM model evaluated encoder-decoder and decoder-only architectures with different pre training objectives [21]. Causal decoder-only models performed best after pre-training [26]. The model's evaluation emphasizes zero-shot generalization across diverse tasks.

The BLOOM model adopted two architectural deviations: ALiBi Positional Embeddings and Embedding LayerNorm [19]. This approach ensures a robust and efficient model, ready to tackle a wide range of language tasks without compromising on training stability and performance. Following considerations were taken for model configurations.

Design Methodology: The BLOOM model's design is a middle ground between replicating existing models and adopting new practices [19]. The focus is on model families that scale well and are supported by public tools.

Experimental Design for Ablations: The model's evaluation is based on zero-shot generalization, measuring performance on diverse tasks. Ablation experiments were conducted using smaller models to determine the best configurations [19].

Out-of-scope Architectures: The BLOOM model did not consider mixture-of-experts (MoE) or state-space models due to various reasons, including lack of GPU-based codebases and underperformance in natural language tasks [19].

7.3 LLAMA 2

The Llama 2 models largely adopted the pre-training setting and architecture from Llama 1 [30]. The model's architectural foundation

is based on the transformer architecture [32] with several notable enhancements and changes such as increased context length and grouped query attention (GQA). Additionally, pre-normalization techniques namely RMSNorm [33] and SwiGLU activation functions [34], rotatory positional embeddings [35] were used for better performance.

8 MODEL TRAINING

8.1 FALCON

In the world of machine learning, merely collecting and processing data isn't enough. It is crucial to understand how the model is trained on this humongous data. Using multitask prompted fine-tuning, a sophisticated technique that refines a pre trained language model with a mixture of different tasks specified through natural language prompts is used [28].

The training process was executed on the AWS SageMaker platform, leveraging a specialized distributed training codebase and harnessing the computational prowess of 384 A100 40GB GPUs. TTI invested a period of two months in training the 60-layer deep large language model containing 1 trillion tokens. The model's training occurred on AWS SageMaker, making use of a robust ensemble of 384 A100 40GB GPUs [39]. This training endeavor was undertaken in December 2022 and was guided by the following hyper parameters [39].

Precision: bfloat16
Optimizer: AdamW
Learning rate: 1.85e-4
Weight decay: 1e-1
Z-loss: 1e-4
Batch size: 1152

The enhanced architecture of Falcon-40B allowed for a significant reduction in the computational resources required for training, amounting to only 75% of what GPT-3 necessitated [39]. Moreover, the model exhibits superior efficiency during token prediction in production, consuming a mere 20% of the processing power compared to GPT-3 [39].

8.2 BLOOM

The T0 model, a brainchild of the BigScience initiative, stands as a testament to the effectiveness of this approach. T0 showcased that models, when fine-tuned using this method, exhibit superior zero-shot task generalization abilities, outperforming even larger models that haven't undergone such fine-tuning [19]. Riding on this success, BLOOM was further enhanced using this technique. It was trained on a subset of the Public Pool of Prompts (P3), a rich collection of prompts designed for various existing and open-source English natural language datasets. Recognizing the need for multilingual capabilities, this collection was expanded, giving birth to xP3, which covered an impressive 46 languages and 16 tasks [19]. The overarching goal was clear: to equip BLOOM with unparalleled multilingual zero-shot task generalization capabilities, culminating in the development of BLOOMZ.

BLOOM was trained using two model training techniques namely: Multi-task fine tuning and Contrastive fine tuning [19]. Multi-task fine tuning: BLOOMZ models, which were fine-tuned, retained the same architectural hyper parameters as the original

Language	ISO-639-3	catalog-ref	Genus	Family	Macroarea	Size in Bytes
Akan	aka	ak	Kwa	Niger-Congo	Africa	70,1554
Arabic	arb	ar	Semitic	Afro-Asiatic	Eurasia	74,854,900,600
Assamese	asm	as	Indic	Indo-European	Eurasia	291,522,098
Bambara	bam	bm	Western Mande	Mande	Africa	391,747
Basque	eus	eu	Basque	Basque	Eurasia	2,360,470,848
Bengali	ben	bn	Indic	Indo-European	Eurasia	18,606,823,104
Catalan	cat	ca	Romance	Indo-European	Eurasia	17,792,493,289
Chichewa	nya	ny	Bantoid	Niger-Congo	Africa	1,187,405
chiShona	sna	sn	Bantoid	Niger-Congo	Africa	6,638,639
Chitumbuka	tum	tum	Bantoid	Niger-Congo	Africa	170,360
English	eng	en	Germanic	Indo-European	Eurasia	484,953,009,124
Fon	fon	fon	Kwa	Niger-Congo	Africa	2,478,546
French	fra	fr	Romance	Indo-European	Eurasia	208,242,620,434
Gujarati	guj	gu	Indic	Indo-European	Eurasia	1,199,986,460
Hindi	hin	hi	Indic	Indo-European	Eurasia	24,622,119,985
Igbo	ibo	ig	Igboid	Niger-Congo	Africa	14078,521
Indonesian	ind	id	Malayo-Sumbawan	Austronesian	Papunesia	19,972,325,222
isiXhosa	xho	xh	Bantoid	Niger-Congo	Africa	14,304,074
isiZulu	zul	zu	Bantoid	Niger-Congo	Africa	8,511,561
Kannada	kan	kn	Southern Dravidian	Dravidian	Eurasia	2,098,453,560
Kikuyu	kik	ki	Bantoid	Niger-Congo	Africa	359,615
Kinyarwanda	kin	rw	Bantoid	Niger-Congo	Africa	40,428,299
Kirundi	run	rn	Bantoid	Niger-Congo	Africa	3,272,550
Lingala	lin	ln	Bantoid	Niger-Congo	Africa	1,650,804
Luganda	lug	lg	Bantoid	Niger-Congo	Africa	4,568,367
Malayalam	mal	ml	Southern Dravidian	Dravidian	Eurasia	3,662,571,498
Marathi	mar	mr	Indic	Indo-European	Eurasia	1,775,483,122
Nepali	nep	ne	Indic	Indo-European	Eurasia	2,551,307,393
Northern Sotho	nso	nso	Bantoid	Niger-Congo	Africa	1,764,506
Odia	ori	or	Indic	Indo-European	Eurasia	1,157,100,133
Portuguese	por	pt	Romance	Indo-European	Eurasia	79,277,543,375
Punjabi	pan	pa	Indic	Indo-European	Eurasia	1,572,109,752
Sesotho	sot	st	Bantoid	Niger-Congo	Africa	751,034
Setswana	tsn	tn	Bantoid	Niger-Congo	Africa	1,502,200
Simplified Chinese	—	zhs	Chinese	Sino-Tibetan	Eurasia	261,019,433,892
Spanish	spa	es	Romance	Indo-European	Eurasia	175,098,365,045
Swahili	swh	sw	Bantoid	Niger-Congo	Africa	236,482,543
Tamil	tam	ta	Southern Dravidian	Dravidian	Eurasia	7,989,206,220
Telugu	tel	te	South-Central Dravidian	Dravidian	Eurasia	2993407,159
Traditional Chinese	—	zht	Chinese	Sino-Tibetan	Eurasia	762,489,150
Twi	twi	tw	Kwa	Niger-Congo	Africa	1,265,041
Urdu	urd	ur	Indic	Indo-European	Eurasia	2,781,329,959
Vietnamese	vie	vi	Viet-Muong	Austro-Asiatic	Eurasia	43,709,279,959
Wolof	wol	wo	Wolof	Niger-Congo	Africa	3,606,973
Xitsonga	tso	ts	Bantoid	Niger-Congo	Africa	707,634
Yoruba	yor	yo	Defoid	Niger-Congo	Africa	89,695,835
Programming Languages	—	—	—	—		174,700,245,772

Figure 3: ROOT Corpus - Linguistic mark up [19]

BLOOM models. The fine tuning hyper parameters were inspired by models like T0 and FLAN [19]. Learning rates were adjusted by doubling the minimum rate of the pre-trained model and rounding. Global batch sizes were increased for smaller variants to boost throughput. The models underwent fine tuning for 13 billion tokens, but the optimal checkpoint was selected based on a separate validation set. Performance plateaued after 1 – 6 billion tokens of fine tuning [19].

Contrastive fine tuning: It was applied to the 1.3 and 7.1 billion parameter BLOOM models using the SGPT Bi-Encoder recipe [22]. This was done to train models that produce high-quality text embeddings [19]. Two models, SGPT-BLOOM-7.1B-msmarco24 and SGPT-BLOOM-1.7B-nli25, were created for multilingual information retrieval and semantic textual similarity, respectively [19]. These models also demonstrated versatility in other embedding tasks like bitext mining, re-ranking, or feature extraction.

The 3 3 shows ROOT Corpus – Linguistic mark up. The BLOOM model was trained across six size variants, with hyper parameters detailed in referenced tables.

The architectural and training hyper parameters were derived from experimental results and past research on LLMs as shown in Figure 4 . The depth and width of the BLOOM model, especially for non-176B models, were influenced by existing literature [23] but adjusted to fit the training setup. The model's embedding parameter sizes were increased due to its larger multilingual vocabulary. During the 104B parameter scale development, various optimization techniques were tested, including different values of Adam β

Hyperparameter (↓)	BLOOM-560M	BLOOM-1.1B	BLOOM-1.7B	BLOOM-3B	BLOOM-7.1B	BLOOM
Architecture hyperparameters						
Parameters	559M	1,065M	1,722M	3,003M	7,069M	176,247M
Precision			`float16`			`bfloat16`
Layers	24	24	24	30	30	70
Hidden dim.	1024	1536	2048	2560	4096	14336
Attention heads	16	16	16	32	32	112
Vocab size			250,680			250,680
Sequence length			2048			2048
Activation			`GELU`			`GELU`
Position emb.			`Alibi`			`Alibi`
Tied emb.			`True`			`True`
Pretraining hyperparameters						
Global Batch Size	256	256	512	512	512	2048
Learning rate	3.0e-4	2.5e-4	2e-4	1.6e-4	1.2e-4	6e-5
Total tokens			341B			366B
Warmup tokens			375M			375M
Decay tokens			410B			410B
Decay style			`cosine`			`cosine`
Min. learning rate			1e-5			6e-6
Adam (β_1, β_2)			(0.9, 0.95)			(0.9, 0.95)
Weight decay			1e-1			1e-1
Gradient clipping			1.0			1.0
Multitask finetuning hyperparameters						
Global Batch Size	1024	1024	2048	2048	2048	2048
Learning rate	2.0e-5	2.0e-5	2.0e-5	2.0e-5	2.0e-5	2.0e-5
Total tokens			13B			13B
Warmup tokens			0			0
Decay style			`constant`			`constant`
Weight decay			1e-4			1e-4

Figure 4: Hyper parameters: Pre training; Multi-task fine tuning and Architecture [19]

	Training Data	Params	Context Length	GQA	Tokens	LR
Llama 1	*See Touvron et al. (2023)*	7B	2k	✗	1.0T	3.0×10^{-4}
		13B	2k	✗	1.0T	3.0×10^{-4}
		33B	2k	✗	1.4T	1.5×10^{-4}
		65B	2k	✗	1.4T	1.5×10^{-4}
Llama 2	*A new mix of publicly available online data*	7B	4k	✗	2.0T	3.0×10^{-4}
		13B	4k	✗	2.0T	3.0×10^{-4}
		34B	4k	✓	2.0T	1.5×10^{-4}
		70B	4k	✓	2.0T	1.5×10^{-4}

Figure 5: Training data summary of LLAMA family of models [31]

parameters, weight decay, and gradient clipping [19]. However, these did not yield significant improvements. A cosine learning rate decay schedule was used over 410B tokens, with a warm up phase for 375M tokens. Weight decay, gradient clipping, and no dropout were applied [24]. The training aimed to cover the equivalent of the ROOTS dataset's 341 billion tokens, but larger models were trained for an additional 25 billion tokens based on revised scaling laws [19].

8.3 LLAMA 2

8.3.1 Hyper parameters. AdamW optimizer [36] was used for training purposes with $\beta 1 = 0.9$, $\beta 2 = 0.95$, eps = $10-5$ [31]. A cosine learning rate schedule with warm up of 2000 steps along with decay final learning rate down to 10% of the peak learning rate [31]. Weight decay of 0.1 and gradient clipping of 1.0 was used [31].

8.3.2 Tokenizer. LLAMA 2 used the same tokenizer as LLMA 1 as shown in Figure 5. Byte pair encoding (BPE) [37] was used along with SentencePiece [38]. The total vocabulary size is 32k tokens [31]. The tokenizer used was the same as Llama 1, employing a byte pair encoding (BPE) algorithm. The vocabulary size was 32k tokens [31].

9 APPLICATIONS

The utility of LLMs is spanned across multiple domains including healthcare, education, law, finance, and scientific research. In healthcare, LLMs have been used for tasks like medical advice consultation and biology information extraction. However, concerns about misinformation and patient privacy persist. In education, LLMs have shown promise in assisting with standardized tests and writing, but issues like plagiarism and bias are challenges. In law, LLMs have been applied for legal document analysis and judgment prediction, but concerns about copyright issues and bias remain. In finance, LLMs are used for tasks like financial sentiment analysis, but their application requires strict monitoring due to potential risks to financial markets. In scientific research, LLMs assist in

various stages from literature survey to data analysis, although their reliability needs further improvement.

10 CONCLUSIONS AND FUTURE DIRECTIONS

The architecture of LLMs, primarily based on the Transformer model, is discussed with a focus on its scalability and the challenges of 'catastrophic forgetting' during fine-tuning. Training LLMs is computationally expensive and sensitive to data quality, calling for more efficient pre-training approaches. In terms of utilization, the text emphasizes the role of 'prompting' as a cost-effective way to apply LLMs, although it has its limitations. Safety and alignment are significant concerns. LLMs can generate hallucinations or factually incorrect information and could be misused for generating harmful or biased content. Methods like Reinforcement Learning from Human Feedback (RLHF) are used to align LLMs better with human values, but these methods are not without their challenges, including the need for high-quality human feedback. It is interesting to observe the future trajectory for open source holds vis-a-vis licensed / closed LLMs as infrastructure and human capital required for training LLMs is expensive and scarce. Finally, the text suggests that LLMs are poised to impact a broad range of real-world applications, potentially leading to an ecosystem of LLM-empowered applications. However, it stresses that AI safety should be a primary concern in this development process. Overall, the survey serves as a detailed resource on the current state, challenges, and future prospects of leading open source LLMs released between July and September 2023.

11 DISCLAIMER

The primary objective of this paper is to provide a concise overview and analysis of existing evaluations conducted on large language models (LLMs). The findings and conclusions presented in each paper are the original contributions of their respective authors, with a particular focus on addressing ethical and bias-related concerns. The paper also explores potential side effects of LLMs, with the sole aim of promoting a deeper comprehension of these models. Furthermore, given the ongoing advancement of LLMs, especially through online services like Claude and ChatGPT, it is highly probable that they will continue to improve, potentially mitigating some of the limitations outlined in this paper while possibly introducing new ones. We encourage interested readers to utilize this survey as a foundational reference for future research and to conduct practical experiments using current LLM systems for evaluations. Lastly, the evaluation of LLMs remains an evolving field, and it's possible that we may have overlooked some recent papers or benchmarks. We welcome all constructive feedback and suggestions to enhance the quality of this survey.

REFERENCES

[1] Wayne Xin Zhao, Kun Zhou*, Junyi Li*, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie and Ji-Rong Wen, 2023, A Survey of Large Language Models, arXiv:2303.18223v11 [cs.CL], pg. 1

[2] J. Gao and C. Lin, 2004, "Introduction to the special issue on statistical language modeling," ACM Trans. Asian Lang. Inf. Process, vol. 3, no. 2, pp. 87–93

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, 2003, "A neural probabilistic language model," J. Mach. Learn. Res., vol. 3, pp. 1137–1155.

[4] M. Shanahan, 2022, "Talking about large language models," CoRR, vol. abs/2212.03551.

[5] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi, Q. Le, and D. Zhou, 2022, "Chain of thought prompting elicits reasoning in large language models," CoRR, vol.abs/2201.11903.

[6] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, 2022, "Training compute-optimal large language models," vol. abs/2203.15556.

[7] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic, 2022, "Galactica: A large language model for science," CoRR, vol. abs/2211.09085.

[8] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, 2023, "Sparks of artificial general intelligence: Early experiments with gpt-4," vol. abs/2303.12712.

[9] Y. Fu, H. Peng, and T. Khot, 2022, "How does gpt obtain its ability? tracing emergent abilities of language models to their sources," Yao Fu's Notion.

[10] OpenAI, 2023 "Gpt-4 technical report," OpenAI.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 2014, Neural machine translation by jointly learning to align and translate. CoRR, abs/1409.0473.

[12] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio,2014 Learning phrase representations using rnn encoder-decoder for statistical machine translation. CoRR, abs/1406.1078.

[13] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le, 2014, Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, pages 3104–3112.

[14] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. Mc- Candlish, A. Radford, I. Sutskever, and D. Amodei, 2020, "Language models are few-shot learners," in Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems.

[15] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi,S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, 2022, "Palm: Scaling language modeling with pathways," CoRR, vol. abs/2204.02311.

[16] Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada,Huu Nguyen, Jörg Frohberg, Mario Šaško, Quentin Lhoest, Angelina McMillan-Major, Gérard Dupont, Stella Biderman, Anna Rogers, Loubna Ben allal, Francesco De Toni, Giada Pistilli, Olivier Nguyen, Somaieh Nikpoor, Maraim Masoud, Pierre Colombo, Javier de la Rosa, Paulo Villegas, Tristan Thrush, Shayne Longpre, Sebastian Nagel, Leon Weber, Manuel Romero Muñoz, Jian Zhu, Daniel Van Strien, Zaid Alyafeai, Khalid Almubarak, Vu Minh Chien, Itziar Gonzalez-Dios, Aitor Soroa, Kyle Lo, Manan Dey, Pedro Ortiz Suarez, Aaron Gokaslan, Shamik Bose, David Ifeoluwa Adelani, Long Phan, Hieu Tran, Ian Yu, Suhas Pai, Jenny Chim, Violette Lepercq, Suzana Ilic, Margaret Mitchell, Sasha Luccioni, and Yacine Jernite, 2022. The BigScience ROOTS corpus: A 1.6TB composite multilingual dataset. In Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track. URL https://openreview.net/forum?id=UoEw6KigkUn.

[17] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf, 2021. Datasets: A community library for natural language processing In Proceedings of the 2021 Conference on Empirical Methods in Natural Language 54 BLOOM Processing: System Demonstrations, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-demo.21. URL https://aclanthology.org/2021.emnlp-demo.21.

[18] Christopher Akiki, Giada Pistilli, Margot Mieskes, Matthias Gallé, Thomas Wolf, Suzana Ilić, and Yacine Jernite, 2022. BigScience: A Case Study in the Social Construction of a Multilingual Large Language Model. URL https://arxiv.org/abs/2212.04960.

[19] BLOOM: A 176B-Parameter Open-Access Multilingual Language Model, 2022, arXiv:2211.05100v4 [cs.CL]

[20] Pedro Javier Ortiz Suárez, Benoˆıt Sagot, and Laurent Romary, 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In Piotr Ba´nski, Adrien Barbaresi, Hanno Biber, Evelyn Breiteneder, Simon Clematide, Marc Kupietz, Harald L¨ungen, and Caroline Iliadi, editors, Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7), pages 9 – 16, Cardiff, UK, Leibniz-Institut f¨ur Deutsche Sprache. doi: 10.14618/ids-pub-9021. URL http://nbn-resolving.de/urn:nbn:de:bsz:mh39-90215.

[21] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, *et al*, 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. arXiv preprint arXiv:2204.07705, 2022b.

[22] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, *et al*, 2022. Crosslingual generalization through multitask fine-tuning. arXiv preprint arXiv:2211.01786, 2022b.

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

[24] Ilya Loshchilov and Frank Hutter, 2016 SGDR: stochastic gradient descent with restarts. CoRR, abs/1608.03983, URL http://arxiv.org/abs/1608.03983.

[25] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, L. Li, and Z. Sui, 2023, "A survey for in-context learning," CoRR, vol. abs/2301.00234.

[26] W. X. Zhao, J. Liu, R. Ren, and J. Wen, 2022, "Dense text retrieval based on pretrained language models: A survey," CoRR, vol. abs/2211.14876.

[27] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi, Q. Le, and D. Zhou, 2022, "Chain of thought prompting elicits reasoning in large language models," CoRR, vol. abs/2201.11903.

[28] D. Zhou, N. Sch¨arli, L. Hou, J.Wei, N. Scales, X.Wang, D. Schuurmans, O. Bousquet, Q. Le, and E. H. Chi, 2022, "Least-to-most prompting enables complex reasoning in large language models," CoRR, vol. abs/2205.10625.

[29] Llama: Open and Efficient Foundation Language Models Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin Edouard Grave, Guillaume Lample, 2023, arXiv:2302.13971v1 [cs.CL].

[30] Llama 2: Open Foundation and Fine-Tuned Chat Models Hugo Touvron∗ Louis Martin†Kevin Stone†Peter Albert Amjad Almahairi Yasmine Babaei Nikolay Bashlykov Soumya Batra Prajjwal Bhargava Shruti Bhosale Dan Bikel Lukas Blecher Cristian Canton Ferrer Moya Chen Guillem Cucurull David Esiobu Jude Fernandes Jeremy Fu Wenyin Fu Brian Fuller Cynthia Gao Vedanuj Goswami Naman Goyal Anthony Hartshorn Saghar Hosseini Rui Hou Hakan Inan Marcin Kardas Viktor Kerkez Madian Khabsa Isabel Kloumann Artem Korenev Punit Singh Koura Marie-Anne Lachaux Thibaut Lavril Jenya Lee Diana Liskovich Yinghai Lu Yuning Mao Xavier Martinet Todor Mihaylov Pushkar Mishra Igor Molybog Yixin Nie Andrew Poulton Jeremy Reizenstein Rashi Rungta Kalyan Saladi Alan Schelten Ruan Silva Eric Michael Smith Ranjan Subramanian Xiaoqing Ellen Tan Binh Tang Ross Taylor Adina Williams Jian Xiang Kuan Puxin Xu Zheng Yan Iliyan Zarov Yuchen Zhang Angela Fan Melanie Kambadur Sharan Narang Aurelien Rodriguez Robert Stojnic Sergey Edunov Thomas Scialom, 2023 arXiv:2307.09288v2 [cs.CL].

[31] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai, 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, 2017. Attention is all you need.

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

[34] Noam Shazeer. Glu, 2020 variants improve transformer.

[35] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu, 2022. Roformer: Enhanced transformer with rotary position embedding.

[36] Ilya Loshchilov and Frank Hutter, 2017. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101.

[37] Rico Sennrich, Barry Haddow, and Alexandra Birch, 2016. Neural machine translation of rare words with subword units.

[38] Taku Kudo and John Richardson, 2018. Sentence piece: A simple and language independent subword tokenizer and detokenizer for neural text processing.

[39] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, Julien Launay, 2023. The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only

[40] Lee, K., Ippolito, D., Nystrom, A., Zhang, C., Eck, D., Callison-Burch, C., and Carlini, N., 2022, Deduplicating training data makes language models better. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 8424– 8445.

[41] Dodge, J., Sap, M., Marasovic, A., Agnew, W., Ilharco, G., ´ Groeneveld, D., Mitchell, M., and Gardner, M, 2021, Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 1286–1305.

[42] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, Sumit Sanghai, 2023, GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints