AI and Ethics (2025) 5:1955–1971
https://doi.org/10.1007/s43681-024-00583-7

REVIEW

History, development, and principles of large language models: an introductory survey

Zichong Wang[1] · Zhibo Chu[1] · Thang Viet Doan[1] · Shiwen Ni[2] · Min Yang[2] · Wenbin Zhang[1]

Received: 31 July 2024 / Accepted: 14 September 2024 / Published online: 14 October 2024

Abstract
Language models serve as a cornerstone in natural language processing, utilizing mathematical methods to generalize language laws and knowledge for prediction and generation. Over extensive research spanning decades, language modeling has progressed from initial statistical language models to the contemporary landscape of large language models (LLMs). Notably, the swift evolution of LLMs has reached the ability to process, understand, and generate human-level text. Nevertheless, despite the significant advantages that LLMs offer in improving both work and personal lives, the limited understanding among general practitioners about the background and principles of these models hampers their full potential. Notably, most LLM reviews focus on specific aspects and utilize specialized language, posing a challenge for practitioners lacking relevant background knowledge. In light of this, this survey aims to present a comprehensible overview of LLMs to assist a broader audience. It strives to facilitate a comprehensive understanding by exploring the historical background of language models and tracing their evolution over time. The survey further investigates the factors influencing the development of LLMs, emphasizing key contributions. Additionally, it concentrates on elucidating the underlying principles of LLMs, equipping audiences with essential theoretical knowledge. The survey also highlights the limitations of existing work and points out promising future directions.

Keywords Large language model · Natural language processing · Artificial intelligence · Language model

1 Introduction

Language stands as humanity's most potent tool, enabling the expression of thoughts and emotions and facilitating communication with others [1, 2]. However, machines lack the intrinsic ability to comprehend and communicate in human language, necessitating powerful Artificial Intelligence (AI) algorithms to acquire such capabilities. The ultimate aim of AI research is the creation of machines capable of reading, writing, and conversing like humans [3–5]. Natural language processing (NLP) emerges as the AI branch dedicated to realizing this goal. At the core of NLP are language models (LMs), tasked with predicting or generating the probability or likelihood of linguistic units (e.g., words, phrases, sentences) based on context. The evolutionary stages of LMs unfold chronologically from the initial statistical language models (SLMs) to subsequent neural language models (NLMs), progressing to pre-trained language models (PLMs), and reaching the current state of large language models (LLMs). Specifically, SLMs employ simple probability distributions to model word sequences, while NLMs utilize neural networks to grasp complex patterns and representations of language. PLMs leverage large-scale corpora and self-supervised learning to capture general linguistic knowledge, and LLMs extend PLMs by incorporating massive data, computation, and algorithms, resulting in more expressive, sophisticated, and adaptable LMs [6, 7].

In recent years, significant advancements in NLP have been achieved with LLMs, such as the generative pre-trained transformer (GPT) series of models [8–11]. Extensively

✉ Wenbin Zhang
wenbin.zhang@fiu.edu

Zichong Wang
ziwang@fiu.edu

Thang Viet Doan
tdoan011@fiu.edu

[1] Florida International University, 11200, SW 8th Street, Miami, FL 33199, USA

[2] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong, China

trained on textual data, these models demonstrate the ability to generate human-level texts and perform language-based tasks with exceptional precision. Despite these advancements having greatly benefited various aspects of work and life, LLMs might not be widely understood by practitioners, particularly those without a background in NLP. To provide practitioners with a fundamental understanding of LLMs, this review introduces it across six dimensions: history, development, principles, applications, drawbacks and future directions. This survey distinguishes itself from other reviews on PLMs or LLMs [4, 12–14], which often concentrate on specific aspects using technical language; this survey aims to elucidate LLMs and their principles in a more accessible manner, intending to maximize its full potential.

To achieve a comprehensive survey of LLMs, we employed a meticulous review process guided by a structured search strategy. We searched databases such as IEEE Xplore, ACM Digital Library, Scopus, and Google Scholar using keywords including "large language models", "large language models survey", and "large language models for *X*" where *X* includes software engineering, drug discovery, finance, medical, legal, and education. Our review focused on publications from major conferences and journals about NLP and LLMs' applications, including EMNLP [15–18], ACL [19–22], ICLR [23–30], ICML [31–34], ICSE [35, 36], ICAIF [37, 38], ICDM [39–41], BEA [42], Financial Innovation [43], PLOS Digital Health [44, 45], Applied Sciences [46], Interspeech [47–49], Journal of Machine Learning Research [50–53], Journal of AI [54], etc., with an emphasis on studies from the past decade. Additionally, we reviewed references from selected papers and cross-checked our findings against existing LLMs surveys [4, 55–57]. Our process involved identifying relevant studies and applying screening criteria based on LLMs relevance, citation count, and the quality of the publication venue. Papers with minimal contributions or lower relevance were excluded. The remaining studies were then evaluated for methodological rigor and clarity of conclusions. Key data from these studies were extracted and synthesized to highlight trends, gaps, and emerging themes in LLM research. This synthesis offers a comprehensive overview of current advancements, ongoing challenges, and future research directions, providing valuable insights for both academic and industrial applications.

The subsequent sections of this review are structured as follows: Sect. 2 details the history of LLMs and the contributing factors behind their rapid growth, covering the evolution of LMs, increased data diversity, computational advancements, and algorithmic innovations. Section 3 provides an overview of the principles of LLMs, using the GPT family of models as an accessible example to enhance understanding of LLMs' principles. Additionally, we present a comparison of state-of-the-art models across various types to assist readers in selecting the most suitable model for their specific applications. Moving on, Sect. 4 explores the wide-ranging applications of LLMs across various industries and professional domains. Section 5 critically examines the drawbacks of current state-of-the-art LLMs, and finally, Sect. 6 concludes the survey.

2 History and development of large language models

In this section, we will explore the history of the LMs and analyze the developmental dynamics and influences shaping LLMs.

2.1 History of LLMs

To better understand LLMs, this section will follow the developmental stages of LMs and introduce SLMs, NLMs, PLMs, and LLMs. Figure 1 provides a visual map of the history of the development of the LMs.

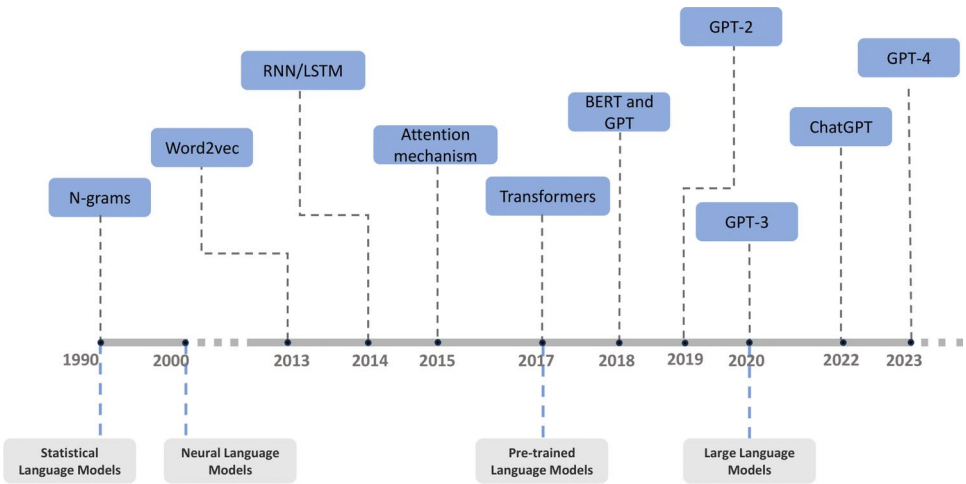

Fig. 1 History and development of language models

- Statistical Language Models: SLMs [49, 58, 59] originated in the 1990s as mathematical models addressing contextually relevant properties of natural language from a probabilistic statistical perspective. The essence of statistical language modeling lies in ascertaining the probability of a sentence occurring within a text. Considering *S* as the sentence "I am very happy", $P(\omega_i)$ signifies the probability of the *i*-th word in the sentence: ω_1 as "I", ω_2 as "am", ω_3 as "very", and ω_4 as "happy". Now, the objective is to ascertain the likelihood of *S* appearing in the text, denoted as *P*(*S*):

$$P(S) = P(\omega_1, \omega_2, \omega_3, \omega_4) = P(I, am, very, happy) \tag{1}$$

To calculate this probability, the conditional probability can be employed:

$$P(I, am, very, happy) = P(I) \cdot P(am \mid I) \cdot P(very \mid I, am) \cdot P(happy \mid I, am, very) \tag{2}$$

where *P*(*I*) represents the probability of the word "I" appearing and *P*(*am*|*I*) stands for the probability of "am" appearing given that "I" has appeared. When we multiply *P*(*am*|*I*) by *P*(*I*), it fulfills the condition of "I" appearing in *P*(*am*|*I*), resulting in the probability of "I am" appearing together as $P(I, am) = P(I) \cdot P(am|I)$. Now, the question arises: how do we calculate the conditional probability of the occurrence of each word? The answer lies in Maximum Likelihood Estimation, enabling us to estimate probabilities by substituting them with frequencies when the sample size is sufficiently large, given by:

$$P(w_i|w_1 w_2 \cdots w_{i-1}) = \frac{P(w_1 \cdots w_{i-1} w_i)}{P(w_1 w_2 \cdots w_{i-1})} = \frac{\mathbf{C}(w_1 w_2 \cdots w_i)}{\mathbf{C}(w_1 w_2 \cdots w_{i-1})} \tag{3}$$

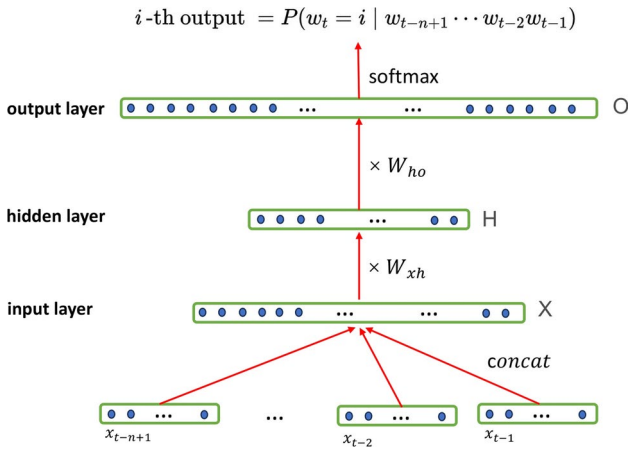

Fig. 2 Neural language model

where $\mathbf{C}(\cdot)$ represents the count of occurrences of the subsequence in the training set. Using this formula, we can calculate the likelihood of each word as the *i*-th word given the preceding $i-1$ words. Then, we select the *i*-th word by choosing the word associated with the highest probability. The previous discussion assumes that the *n*-th word is related to the initial $n-1$ words, consistent with our intuition and commonly referred to as the n-gram model. In order to efficiently compute conditional probabilities, it is necessary to pre-compute and save *C*(*X*) required for the conditional probability computation, where *X* is a sentence of length *n*. The number of possible sentences *X* grows exponentially with the size of the vocabulary. For instance, with 1000 different words, there exist 1000^n potential sequences of length *n*. However, excessively large values of *n* pose storage limitations. Typically, *n* is confined to 2 or 3, causing each word to relate to only its first 1 or 2 preceding words, ultimately leading to a reduction in the model's accuracy.

- Neural Language Models: NLMs [47, 48, 60] leverage neural networks to predict the probabilities of subsequent words within sequences. They effectively handle longer sequences and mitigate the limitations associated with small *n* in SLMs. Before delving into neural networks, let's grasp the concept of word vectors. Humans effortlessly comprehend word meanings. For instance, "cat" and "dog" share closer semantic connections than "cat" and "tree" since they both represent animals. But how does a computer accomplish this? Computers operate using binary code—0 s and 1 s—so human language needs translation into binary. Word vectors can accomplish this, which are numerical representations of human language, where each word corresponds to a distinct vector. These vectors usually possess fixed dimensions and can simulate word relationships through the angles between them. An intriguing example is the angle between the word vectors for "cat" and "dog" which is smaller than the angle between the word vector for "cat" and "tree". Another illustrative example is "Messi − Argentina + Portugal = Cristiano Ronaldo". Word2Vec [61] is a widely recognized tool for computing word vectors. Its function involves converting words into dense numerical representations, making words with similar semantics closer together in the vector space. The efficacy of Word2Vec is notably influenced by its utilization of neural networks, which mirrors the structure and functionality of the human brain, comprising interconnected simple units known as neurons organized into different layers. Figure 2 provides a visualization of a simple NLM structure, composed of input, hidden, and output layers. Within this structure, each word vector $x_t \in \mathbb{R}^m$, the matrix $W_{xh} \in \mathbb{R}^{m(n-1)\times h}$, and the matrix

$W_{ho} \in \mathbb{R}^{h \times |V|}$, where h signifies the number of neurons in the hidden layer, and V is the vocabulary containing all words recognized by the model. NLMs operate akin to the n-gram concept, assuming that the probability of each word depends only on its previous n-1 words. The first layer-the input layer-concatenates the word vectors of n-1 words, forming a vector $X \in \mathbb{R}^{m \times (n-1)}$. Subsequently, the second layer-the hidden layer-derives the output H by applying a non-linear activation function, like sigmoid or tanh, to the matrix product of X and W_{xh}. Following this, the third layer-the output layer-aims to forecast the subsequent word based on the hidden layer's output. With$|V|$ neurons within the output layer, the resulting output vector $O \in \mathbb{R}^{|V|}$ is computed by multiplying H and W_{ho}. This resultant vector then undergoes the Softmax function, producing a vector O' containing the probability value assigned to each word. The Softmax function, concerning the output of the i-th neuron, is defined as follows:

$$O'_i = \text{Softmax}\left(O'_i\right) = \frac{\exp\left(O_i\right)}{\sum_{j=1}^{|V|} \exp\left(O_j\right)}, \tag{4}$$

where O'_i denotes the output value of the i-th node, and$|V|$ represents the count of output nodes, corresponding to the classification categories. Utilizing the Softmax function allows the transformation of output values from multiclass classification into a probability distribution, ensuring a sum of 1 within the range [0, 1].

- Pre-trained Language Models: PLMs undergo initial training using an extensive volume of unlabeled text, enabling them to grasp fundamental language structures such as vocabulary, syntax, semantics, and logic—a phase termed pre-training. Subsequently, this comprehensive LM can be applied to various NLP tasks like machine translation, text summarization, and question-answering systems. To optimize its performance, models need to be trained a second time on a smaller dataset customized for a specific downstream task—a phase known as fine-tuning. This is the "pre-training and fine-tuning" learning paradigm. We can use a visual example to understand the "pre-training and fine-tuning", as follows: in martial arts novels, a person who wants to become a martial arts master needs to have a solid foundation of internal martial arts, which can be acquired by training on a large variety of techniques. Then, the person can learn a specific skill, such as a sword or a palm strike, and master it quickly and effectively by applying internal martial arts principles. A large number of studies on PLMs have been built on this paradigm, which introduces different architectures [52, 62] (e.g., GPT-2 [9] and Bert [62]).
- Large Language Models: LLMs are trained on massive text corpora with tens of billions (or more) of parameters, such as GPT-3 [10], GPT-4 [11], PaLM [51], and LLaMA [63]. The goal of LLMs is to enable machines to understand human commands and adherence to human values. Their hallmark lies in the consolidation of two stages: initial pre-training on a vast general-purpose corpus followed by alignment with human values, rather than transitioning to a different domain. LLMs exhibit remarkable adaptability compared to PLMs, transitioning from specialized to general-purpose models. The substantial increase in model size, dataset volume, and computational prowess has resulted in significant enhancements across various tasks and unveiled remarkable capabilities absent in smaller models. For example, GPT-3 has the capability to leverage contextual information, a functionality that GPT-2 [9] lacks. This means that when GPT-3 is prompted with task-related examples, it utilizes them to enhance its problem-solving capabilities. The number of parameters for LLMs typically exceeds a hundred billion, and the training data is usually in the range of a few hundred GB to a few TB. A concrete example is that there are multiple versions of the GPT-2 model, with the largest version having 1.5 billion parameters and using 40GB of text data for training. In contrast, the largest version of the GPT-3 model has 175 billion parameters and uses 570GB of text data for training. This example illustrates the significant discrepancy between LLMs and PLMs concerning parameter count and training data volume.

2.2 Factors propelling large language models

In the past few years, various factors have played a significant role in the swift advancement of LLMs:

- Data Diversity: Data diversity has emerged as a crucial catalyst for the development of LLMs. Recent years have witnessed a surge in the availability and diversity of extensive internet-based data sources, furnishing these models with training data containing rich linguistic and worldly insights. The quality and origins of this data significantly influence the efficacy and capacity of LLMs. Earlier studies trained language models on a single domain of text, such as news articles [64], Wikipedia [30], or fiction books [65]. However, the proliferation of vast and heterogeneous text corpora across the web-including blogs, social media, forums, and reviews, among others-has spurred researchers to pivot towards training

models on multi-domain texts. This shift enhances the model's aptitude for generalization and multitasking. An exemplary instance is Meta AI's open-source large language model, LLaMA [63], trained on publicly accessible datasets like CommonCrawl,[1] C4 [53], Github,[2] Wikipedia,[3] Books [66], ArXiv,[4] and StackExchange.[5]

- Computational Advancement: LLMs are neural network models with a large number of parameters. With the progression of deep learning (DL) technology in recent years, the scale and effectiveness of LLMs have notably escalated. Simultaneously, they present serious challenges due to their substantial computational demands [67]. Computational power signifies a computer system's capability to execute computational tasks, often quantified in terms of floating-point operations per second. According to OpenAI calculations,[6] since 2012, the amount of computation used for global AI training has shown exponential growth, doubling every 3.43 months on average, and now the amount of computation has expanded 300,000 times, far outpacing the rate of computational power growth. Fortunately, hardware innovators like Nvidia continually invent specialized devices such as GPUs, TPUs, and NPUs with amplified computational prowess, LLMs can be trained quickly. For instance, the launch of GPT-3 by OpenAI in May 2020 [10] underscores this reliance on robust hardware. Training GPT-3 on a single NVIDIA Tesla V100 GPU would require an estimated 355 years. Conversely, leveraging 1024 $\times$ A100 GPUs, researchers estimated the potential to train GPT-3 in as little as 34 days. This exemplifies the indispensability of potent hardware in facilitating the development and training of LLMs.
- Algorithmic Innovation: Algorithmic innovation stands as the pivotal driving force behind LLMs, shaping their structure and functionality. From the initial rule- and statistics-based approach to the later DL-based approach, the algorithms of LMs have continuously evolved and improved. Presently, all LLMs are based on the transformer architecture [68], which employs a self-attention mechanism (see discussion in Sect. 3). This architecture offers distinct advantages over traditional recurrent [48] and convolutional neural networks [69], enabling fully parallel computation and adeptly capturing long-distance dependencies. Moreover, there exist numerous iterations and enhancements of the transformer, such as Transformer-XL [70], XLNet [71], ALBERT [72], each targeting specific aspects for optimization. These variations aim to enhance the attention mechanism, refine pre-training objectives, and curtail parameter counts, thereby advancing the transformer architecture in diverse dimensions.

[1] https://commoncrawl.org/get-started.

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

[3] https://huggingface.co/datasets/wikipedia.

[4] https://arxiv.org/.

[5] https://stackexchange.com/.

[6] https://openai.com/research/ai-and-compute.

3 Principles and taxonomy of large language models

This section strives to clarify the fundamental principles and taxonomy of LLMs, intending to provide general practitioners with a more profound understanding of the operational mechanisms employed by these models and help readers choose the most suitable LLMs for their specific applications.

Fundamental to LLMs is the utilization of deep neural networks to grasp the statistical patterns and semantic nuances of language, enabling the understanding and generation of language. The GPT series, a notable example among these models, incorporates the transformer architecture [68] and utilizes autoregression to predict consecutive words. The following discussion will concentrate on the GPT-3 [10] as an exemplar, providing insights into multiple facets that elucidate the principles of LLMs, as illustrated in Fig. 3.

- Input: The GPT model architecture takes a sequence of symbol representations $(x_1, \ldots, x_n)$ as input. This sequence comprises n words, also known as tokens, with GPT-3 explicitly defining the input sequence length as 2048 words.
- Encoding: The significance of encoding words into vectors is heightened by the fact that the machine learning model operates on numeric inputs. Within the input embedding layer, the initial step involves constructing a comprehensive vocabulary that encompasses all words and assigning each word a unique position within this established vocabulary. For instance, "a" could be designated as 0, "an" as 1, and so forth. In the case of GPT-3, its vocabulary encompasses 50,257 words. Following this, every word can be transformed into a one-hot vector of 50,257 dimensions, where only a single element has the value of one while the rest are zeros. For example, "a" would be represented as [1,0,0,0,...], and "an" as [0,1,0,0,0,...]. This process leads to the encoding of the input sequence into a matrix denoted as $I_E \in \mathbb{R}^{2048\times 50{,}257}$.
- Embedding: The encoding process above transforms every input word into a one-hot vector of 50,257 dimensions. However, this vector exhibits high

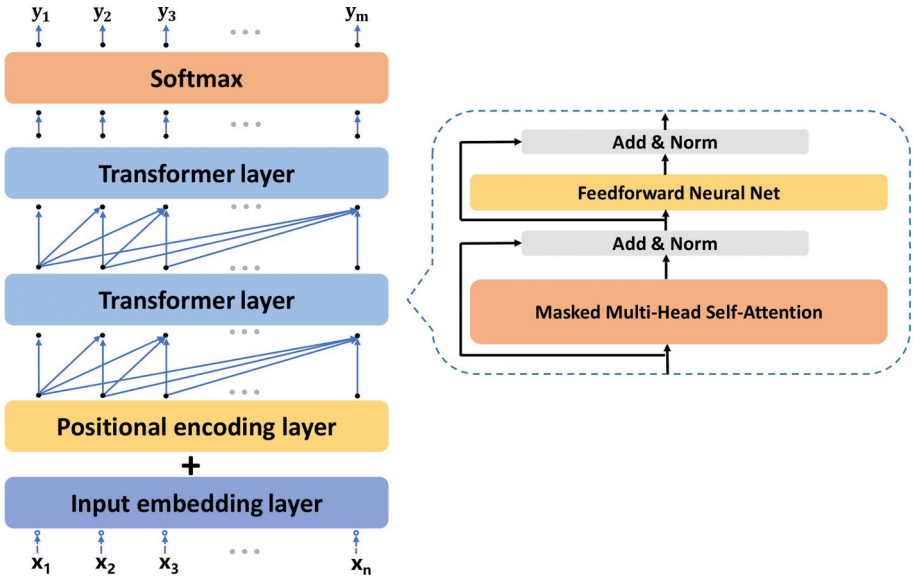

Fig. 3 The GPT-3 model architecture. Note that only two transformer layers are illustrated for simplicity and illustrative purposes, and the actual model consists of 12 transformer layers

sparsity, primarily consisting of zeros, leading to significant space inefficiency. To address this drawback, the subsequent step in the input embedding layer employs an embedding matrix, denoted as W_E, to condense these 50,257-dimensional input vectors into shorter numeric vectors of length n (n is set to 12,288 for GPT-3). This method condenses information about word meanings into a more compact space, effectively reducing the vector's length. Specifically, the transformation is represented as: $X_{WordEmbedding} = I_E \times W_E$, where $X_{WordEmbedding} \in \mathbb{R}^{2048\times12,288}$, $I_E \in \mathbb{R}^{2048\times50257}$ and $W_E \in \mathbb{R}^{50,257\times12,288}$.

- Positional Encoding: The position and order of elements in a sequence are very important for understanding and generating sequences. Demonstrated through the sentences, "He is a good person and does not do bad things" and "He is a bad person and does not do good things", a slight alteration in word order—the transition from "good" to "bad"—results in a significant change in the sentence's semantic meaning. Within the architectural framework of the GPT model, the positional encoding layer serves as a complement to the input embedding layer. This augmentation facilitates the transformer layer in comprehending both the positional and sequential cues within the data. GPT-3 employs the following implementation for positional encodings:

$$PE(\text{ pos },\ 2\text{i}) = \sin\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right) \tag{5}$$

$$PE(\text{ pos },\ 2\text{i}+1) = \cos\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right) \tag{6}$$

- where pos is the position of the current word within the sentence (e.g., in the sentence "A survey of Large Language Model", pos_A=0, pos_{Large}=3), d_{model} is the length of the embedding vector 12,288 and i is the dimension. However, why delve into such intricate computations for positional encodings? Let's explore a more straightforward approach to conceptualize it. Initially, considering a text of length T, the most basic approach of positional encoding is counting. This method utilizes $[0, 1, 2 \ldots, T-1]$ as positional encodings for each word in the text. For instance, the positional encodings for the 2-th word would be $[1, 1, \ldots, 1]$, aligning in length with the embedding vector. However, this encoding strategy presents a notable drawback: within longer sentences, subsequent words hold considerably larger positional encoding values compared to the positional encodings of the initial words. As a result, this disparity inevitably introduces numerical biases in the merged features when combined with word embeddings. To address this issue, normalization becomes a viable consideration. The most straightforward normalization method involves dividing directly by T, resulting in positional encodings such as $[0, \frac{1}{T}, \ldots, \frac{T-1}{T}]$ (e.g., the 2nd word's positional encodings = $[\frac{1}{T}, \frac{1}{T}, \ldots, \frac{1}{T}]$). While this method restricts all positional encodings to the [0,1] range, it introduces a significant inconsistency: the positional values are contingent on the text's length, causing the positional distinctions between adjacent words in shorter texts to differ from those in longer texts. As illustrated in Fig. 4, the positional encodings discrepancy between adjacent word positions in sentence ① is 0.5, while in sentence ②, neighboring word positions are coded with a difference of 0.125. Consequently, neither of these methods proves to be satisfactory. To address this issue, Google

Fig. 4 Incorrect position encoding

devised an alternative approach involving trigonometric functions [68]. As a result, a positional encoding matrix denoted as $X_{PositionalEncoding} \in \mathbb{R}^{2048\times12,288}$.

- Input Matrix: The input sequence $(x_1, \ldots, x_n)$ undergoes processing through both the input embedding layer and the positional embedding layer to generate the input matrix. Specifically, input matrix $X =$ embedding matrix $X_{WordEmbedding} +$ positional encoding matrix $X_{PositionalEncoding}$, where $X \in \mathbb{R}^{2048\times12,288}, X_{WordEmbedding} \in \mathbb{R}^{2048\times12,288}$ and $X_{PositionalEncoding} \in \mathbb{R}^{2048\times12288}$.
- Masked Multi-Head Self-Attention: Masked multi-head self-attention involves combining a masking technique and a multi-head mechanism within the framework of self-attention. The following discussion commences with self-attention before delving into masked multi-head self-attention for enhanced comprehension. Self-Attention: Attention is a crucial cognitive function for human beings, allowing us to concentrate on significant aspects within a complex environment. Within the transformer layer, self-attention mirrors this cognitive functionality by forecasting which input words warrant attention for every output in the sequence, subsequently assessing their significance. For instance, when analyzing the input "I am very" and predicting the output "happy", the relative contributions of the three words "I am very" to the output "happy" might be quantified as 0.7, 0.1, and 0.2, respectively. The specific steps are as follows: Step 1: Initiation commences with the creation of three projection matrices $W_q \in \mathbb{R}^{12288\times128}$, $W_k \in \mathbb{R}^{12288\times128}$, and $W_v \in \mathbb{R}^{12,288\times128}$. These matrices are subsequently multiplied with the input matrix $X \in \mathbb{R}^{2048\times12,288}$ to obtain three different matrices Q, K, and V, which respectively represent the query, key, and value. Step 2: Following this, a matrix is derived using the formula $Softmax\ (Q \times K^T)$. This matrix then multiplied with V to obtain the output Y, represented as $Y = Softmax\ (Q \times K^T) \times V$, where $Q \in \mathbb{R}^{2048\times128}$, $K^T \in \mathbb{R}^{128\times2048}, V \in \mathbb{R}^{2048\times128}$, and $Y \in \mathbb{R}^{2048\times128}$. An illustrative analogy can be drawn to searching for a song within a music software—here, Q denotes the queried song's name, K encompasses all song names within the software database, and V includes the corresponding audio data of all songs in the software. Thus, Y signifies the audio data of the sought-after song. Masked Self-Attention: Masked self-attention is a modified version of self-attention designed to prevent the model from accessing future words during the prediction of the next word by incorporating a masking technique. For example, with the corpus "abcde", we can train GPT to predict the next word four times by feeding in "a", "ab", "abc" and "abcd" separately. However, this involves segmenting and inputting the sentence four times. The mask technique trick streamlines this process, enabling the model to receive the entire sentence at once and generate an output for each word. The challenge arises from the model peeking at subsequent words, which necessitates the use of masking. When predicting the next word of "a" GPT will only see "a", and when predicting the next word of "ab", it will only see "ab". It's essential to differentiate between self-attention (utilized in BERT [62]) and masked self-attention (utilized in GPT-3 [10]). Figure 5 provides a visual illustration. Multi-Head Self-Attention: Multi-head self-attention subdivides the self-attention mechanism into h independent heads to compute in parallel (for GPT-3, h is 96). Each head focuses on different aspects and connections within the sequence, concatenating their outcomes afterward. This approach facilitates the model to collectively concentrate on information derived from diverse representation subspaces, thereby enhancing its capacity for expression. Each head possesses its distinctive projection matrix, each with dimensions $d = d_{model}/h = 12288/96 = 128$. The outcome from each head is symbolized as Y_s, and when these individual head outputs are combined through concatenation ($96Y_s = 12,288 = 96 \times 128$), the resultant amalgamation is denoted as Y.
- Feedforward Neural Net: In addition to masked multi-head self-attention sub-layers, each of the transformer layers in the GPT model architecture contains a fully connected feedforward neural net. This neural network serves as a mechanism designed to non-linearly transform its inputs, thereby enhancing the model's overall expressive capabilities.
- Add & Norm: The add & norm modules are incorporated into each transformer layer on the GPT model architecture, where "add" represents the skip connection [73] and "norm" represents layer normalization [74]. The skip connection helps alleviate the issue of gradient vanishing in deep models, enabling the transformer

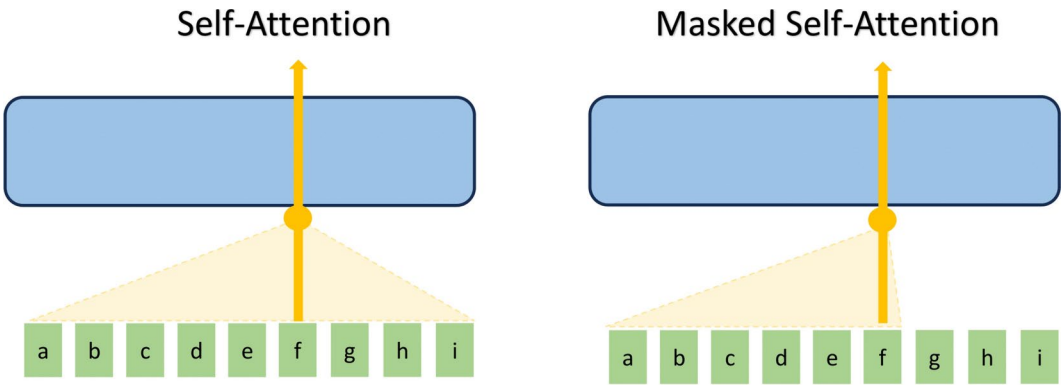

Fig. 5 Self-Attention and Musk Self-Attention: a normal self-attention block allows a position to attend to all words to its right, while masked self-attention prevents a position from attending to words that come after it in the sequence

layer to be stacked to a considerable depth. Additionally, layer normalization serves as a regularization strategy to prevent network overfitting. It calculates the mean and variance on each sample independently, without considering other data, thereby mitigating the impact of varying batch sizes.

- Decoding & Output: After 12 transformer layers, the input matrix $X \in \mathbb{R}^{2048\times12288}$ transforms into an information-rich matrix $Y \in \mathbb{R}^{2048\times12288}$, representing the final representation of the input sequence. To achieve the desired result, we employ $Softmax(Y \cdot W)$, where $W \in \mathbb{R}^{12288\times50257}$, with 50,257 as the vocabulary size. This process generates probabilities assigned to each word in the vocabulary, enabling the selection of the most likely output as GPT's prediction for each word.

To help readers choose the most suitable model for their specific applications, we further organize LLMs based on their structure. Specifically, we classify LLMs into three types: *encoder-only*, *encoder-decoder*, and *decoder-only* models. This taxonomy is based on the transformer architecture, the backbone of LLMs, comprising core components like the encoder, decoder, and their variations, which together enable their capabilities [68]. Specifically, encoders are designed to process and interpret input sequences, while decoders generate output sequences based on this interpretation [75]. In some models, both components are integrated into an encoder-decoder framework to handle a diverse set of tasks effectively [33]. Conversely, other models focus on specific applications by using either the encoder alone for understanding tasks or the decoder alone for text generation. The detail information about each type of models is as outlined below:

- Encoder-only models, such as BERT [62], primarily focus on understanding and interpreting input data rather than generating it. They excel in tasks that require strong contextual understanding, such as sentiment analysis, named entity recognition, and text classification. These models are highly effective for analyzing text but are less suitable for text generation tasks due to their limited generation capabilities and computational intensity when handling large texts.
- Encoder-decoder LLMs, like T5 [53], combine the strengths of both encoders and decoders, making them versatile and powerful for a range of applications. These models are particularly effective for tasks that require both text understanding and generation, such as machine translation, text summarization, and question answering. The encoder-decoder architecture provides a unified approach to these tasks, allowing for a comprehensive understanding and generation of text. However, the complexity of this dual architecture can result in more challenging training processes and slower inference times compared to other model types.
- Decoder-only LLMs, such as GPT-3 [10], are optimized for generating text, making them ideal for content creation, conversational AI, and creative writing. These models are known for their excellent text-generation capabilities and are effective in few-shot learning scenarios, where they can generate coherent and contextually relevant responses from minimal input. However, decoder-only models can face limitations in understanding context deeply and are often resource-intensive, requiring significant computational power to generate high-quality text outputs.With the proposed taxonomy, Table 1 provides a comparison of state-of-the-art LLMs across different types, with models in each category organized in ascending order of parameter size. This structure enables a systematic examination of key aspects such as architecture, parameter count, provider, release date, open-source status, tunability status, subsequently fine-tuning applied for the model (Adaptation), pre-training data scale, and evaluation methods in their

Table 1 Statistics of large language models in recent years

Type	Model	Size	Provider	Release	Opensourced	Tunability	Adaptation		Pre-train Data scale	Evaluation	
							IT	RLHF		ICL	CoT
Encoder-only	ALBERT [72]	0.012	Google AI	02/2020	✓	✓	–	–	16 GB	–	–
	DeBERTa [29]	0.1	Microsoft	06/2020	✓	✓	–	–	80 GB	–	–
	ELECTRA [77]	0.11	Google AI	03/2020	✓	✓	–	–	–	–	–
	BERT [62]	0.11	Google AI	10/2018	✓	✓	–	–	3.3 B words	-	–
	RoBERTa [78]	0.125	Meta AI	07/2019	✓	✓	–	–	160GB	-	–
	XLM-RoBERTa [79]	0.27	Meta AI	07/2020	✓	✓	–	–	2.5 TB	–	–
Encoder-decoder	ERNIE 3.0 [80]	10	Baidu	07/2021	×	✓	–	–	300B tokens	✓	–
	T5 [53]	11	Google AI	10/2019	✓	✓	–	–	1T tokens	✓	–
	T0 [26]	11	BigScience	10/2021	✓	✓	✓	-	–	✓	–
	Flan-T5 [50]	11	Google AI	10/2022	✓	✓	✓	–	–	✓	✓
	UL2 [27]	20	Google AI	05/2022	✓	✓	–	–	1 T tokens	✓	✓
	AlexaTM [81]	20	Amazon	08/2022	×	✓	–	–	1.3 T tokens	✓	✓
	AlphaCode [82]	41	Google AI	02/2022	×	×	–	–	967 B tokens	–	–
	Anthropic [83]	52	Anthropic	12/2021	×	✓	–	-	400B tokens	✓	–
	NLLB [84]	54.5	Meta AI	07/2022	✓	✓	–	–	–	✓	–
	LLaMa [63]	65	Meta AI	02/2023	✓	✓	–	–	1.4 T tokens	✓	–
	FLAN [25]	67	Google AI	09/2021	×	✓	✓	–	–	✓	–
	Sparrow [85]	70	Google AI	09/2022	×	✓	–	✓	–	✓	✓
	Chinchilla [86]	70	Google AI	03/2022	×	✓	–	–	1.4 T tokens	✓	–
	OPT-OML [87]	175	Meta AI	12/2022	✓	×	✓	–	–	✓	✓
	Gopher [88]	280	Google AI	12/2021	×	✓	–	–	300B tokens	✓	–
	PaLM [51]	540	Google AI	04/2022	✓	✓	–	–	780B tokens	✓	✓
	U-PaLM [18]	540	Google AI	10/2022	×	×	–	–	–	✓	✓
	GLaM [31]	1200	Google AI	12/2021	×	✓	–	–	280 B tokens	✓	–
Decoder-only	Codex [76]	12	OpenAI	07/2021	×	×	–	–	100B tokens	✓	–
	Pythia [32]	12	Eleuther AI	04/2023	✓	×	–	–	300B tokens	✓	–
	CodeGeeX [89]	13	Tsinghua U	09/2022	✓	✓	–	–	850B tokens	✓	–
	Skywork [90]	13	Kunlun Inc	10/2023	✓	✓	–	–	3.2T tokens	✓	–
	QWEN [91]	14	Alibaba	09/2023	✓	✓	✓	✓	3T tokens	✓	–
	CodeGen2 [92]	16	Salesforce AI	05/2023	✓	✓	–	–	400B tokens	✓	–
	GPT-NeoX-20B [93]	20	EleutherAI	04/2022	✓	×	–	–	825GB	✓	–
	Gemini 1.5 [94]	20	Google AI	02/2024	×	✓	–	–	–	✓	–
	LlaMa 2 [63]	70	Meta AI	07/2023	✓	✓	✓	✓	2T tokens	✓	–
	LlaMa 3 [95]	70	Meta AI	04/2024	✓	✓	✓	✓	15T tokens	✓	–
	HyperCLOVA [17]	82	Naiver	09/2021	×	✓	–	–	300B tokens	✓	–
	Galactia [96]	120	Meta AI	11/2022	✓	✓	–	–	106B tokens	✓	✓
	GLM [28]	130	Tsinghua U	10/2022	✓	✓	–	–	400B tokens	✓	–
	LaMDA [97]	137	Google AI	01/2022	×	✓	–	–	768B tokens	✓	–
	Jurassic-1 [98]	178	AI21 Labs	08/2021	×	✓	–	–	300B tokens	✓	–
	GPT-3 [10]	175	OpenAI	05/2020	×	×	–	–	300B tokens	✓	–
	WebGPT [99]	175	OpenAI	12/2021	×	×	–	–	–	✓	–
	InstructGPT [100]	175	OpenAI	03/2022	×	×	✓	✓	–	✓	–
	BLOOM [101]	176	BigScience	11/2022	✓	✓	–	–	366B tokens	✓	–
	BLOOMZ [20]	176	BigScience	11/2022	✓	×	✓	–	–	✓	–
	Llama 3.1 [102]	405	Meta AI	06/2024	✓	✓	✓	✓	15T tokens	✓	–
	GPT-4 [11]	450	OpenAI	03/2023	×	✓	✓	✓	1.76T	✓	✓
	MT-NLG [103]	530	Microsoft	01/2022	×	✓	–	–	967B tokens	✓	–
	Gemma [104]	2700	Google AI	02/2024	✓	✓	–	–	6T tokens	✓	–

IT instruction tuning, *RLHF* reinforcement learning with human feedback, *ICL* in-context learning, *COT* chain-of-thought

original papers. Analyzing these elements together provides a thorough understanding of the models' capabilities and differences. Specifically, the type of each model indicates its architecture, whereas the parameter size reflects both the model's complexity and the resources required for fine-tuning. Next, the provider describes the model's development context and goals, and the release date places the model within the timeline of technological advancements. Furthermore, open-source status indicates the model's accessibility for future research and adaptation, whereas tunability status indicates whether the model has been customized after release. Last but not least, the pre-training data scale shows the model's exposure to various types of information, whereas the evaluation methods assess its performance against benchmarks. As an example, Codex [76], a decoder-only model with 12 billion parameters created by OpenAI and released in July 2021, was pre-trained on 100 billion tokens and supports in-context learning. While it is customizable and open-source, Codex does not incorporate more advanced techniques such as chain-of-thought methods or reinforcement learning with human feedback, which limits its ability to handle certain complex tasks.

4 Applications of large language models

LLMs have been instrumental across diverse domains, and this section aims to illustrate exemplary applications of LLMs in various fields.

Software Engineering LLMs have demonstrated remarkable capabilities across various software engineering tasks such as defect prediction [35], code review [105], code generation [19], and analysis of software-related questions [106–108]. These models significantly aid in improving code quality, debugging, and reducing human involvement in many aspects of software development. For instance, Codex [76] is a pioneering model in this domain, utilizing a large generative pre-trained model with up to 12 billion parameters to aid developers by generating code, suggesting optimizations, and identifying and correcting errors. Building on this foundation, several other models have been developed to address various facets of coding and software development. Notable examples include DeepMind's AlphaCode [82], which specializes in generating code for programming competitions, and Meta's InCoder [23] and Code Llama [109], as well as Salesforce's CodeRL [110] and CodeGen [24, 92]. The BigCode project has introduced StarCoder [110], and OpenAI has made significant strides with its GPT [11] and ChatGPT series, enhanced through Reinforcement Learning from Human Feedback (RLHF). These developments expand the scope and potential of automated code generation, setting a new standard in software engineering that offers both opportunities and challenges for the industry.

Drug Discovery The field of drug discovery is characterized by substantial challenges and elevated costs. The utilization of computational methodologies, such as AI, DL, and quantum mechanical techniques, holds the promise of accelerating the identification of potential drug candidates and predicting their properties [111, 112]. Among recent advancements in this arena, the integration of LLMs, such as ChatGPT, emerges as a valuable addition, adept at generating and analyzing natural language texts [113]. LLMs contribute significantly to the initial phase of drug discovery, particularly in identifying suitable drug targets for diverse diseases [114, 115]. They offer preliminary insights into the structure and functionality of protein-based drug targets, subject to further validation by alternative methodologies. Furthermore, LLMs play a pivotal role in subsequent drug discovery steps, aiding in the design and screening of small molecules capable of interacting with these drug targets [113].

Finance LLMs are driving significant advancements in the finance industry, with significant financial applications including trading and portfolio management [116], financial risk modeling [117], financial text mining [38, 43], and financial advisory and customer services [118]. AI integration in financial advisory and customer services is an emerging and rapidly expanding field. According to [119], AI-driven chatbots already provide more than 37% of supporting functions in various e-service scenarios. Within the financial field, these chatbots are embraced as cost-effective alternatives to human customer service, a trend underscored in the "Chatbots in Consumer Finance" report.[7] With the advent of LLMs, more and more tasks that were previously considered difficult to handle have become possible, further expanding the potential applications of AI in the financial industry [37]. Notably, there are already some specialized LLMs like BloombergGPT [120], a 50-billion-parameter model excelling in financial tasks while maintaining strong performance in general LM benchmarks.

Medical LLMs are now at the forefront of medical AI with immense potential to improve the efficiency and effectiveness of clinical, educational, and research work [121]. One example is the performance of LLMs on the MedQA dataset [46], which serves as a widely used biomedical question-answering dataset consisting of the United States Medical Licensing Examination (USMLE)-style questions. In this dataset, ChatGPT attains human passing levels and

[7] https://www.consumerfinance.gov/data-research/research-reports/chatbots-in-consumer-finance/chatbots-in-consumer-finance.

[44]. Additionally, in a span of fewer than six months, Med-PaLM2 achieves proficiency levels close to human experts [122]. The strong performance of GPT-4 [11] and Med-PaLM2 in medical tests suggests that LLMs may be useful teaching tools for students currently attaining a lower level in such tests [121]. GPT-4's meta-prompt feature allows users to explicitly describe the desired role for the chatbot to take on during conversation; a useful example is a "Socratic tutor mode",[8] which encourages students to think for themselves by pitching questions at decreasing levels of difficulty until students can work out solutions to the fuller question at hand. This mode can foster critical thinking and problem-solving skills among students [121]. ChatGPT demonstrates superior performance in tasks not necessitating specialized knowledge or when such expertise is available in user prompts [123]. For example, consider discharge summaries as a case in point - they entail repetitive tasks involving information interpretation and compression, often requiring minimal specialized knowledge and user prompts [63].

Legal LLMs exhibit robust capabilities across various legal tasks, particularly in case prediction and generating legal texts. For instance, ChatGPT achieved a 50.3% accuracy rate on NCBE MBE practice exams, significantly surpassing the 25% random guess level, and achieved passing scores in both Evidence and Torts [124]. The integration of ChatGPT holds the potential to reshape the entire legal industry. An Australian law firm experimented with ChatGPT to draft a statement of claim based on the 1992 Mabo case, yielding results akin to those of a first-year lawyer[9]. [125] investigation explored the possibility that ChatGPT could replace litigators by evaluating its drafting and research capabilities. While recommending ChatGPT as a complement to litigators rather than a complete replacement, this suggestion stemmed from ChatGPT's limitations in meeting the precise accuracy and timeliness demands of legal texts, especially in ensuring continuous access to updated legal research. More recently, advancements such as DeLi-legal[10] addressed this issue by integrating with external law databases to enhance retrieval and minimize hallucination. Meanwhile, LLMs in the legal field have emerged in recent months such as ChatLaw [70].

Education With the rapid advancement of AI technology, its applications in education are expanding, fundamentally transforming teaching and learning methods. Researchers have recognized the remarkable capabilities of LLMs, such as ChatGPT, and have explored their substantial potential to impact various educational settings. These models can be used in diverse ways, such as simulating students to help train teachers [126, 127] and acting as virtual instructors to provide personalized instruction to students [16, 128, 129]. LLMs can also function as personal tutors, offering real-time feedback [130] and tailored evaluations and suggestions based on individual learning progress [54, 131]. Furthermore, they can enhance the overall learning experience by improving student engagement and promoting greater autonomy in learning [42, 132]. In addition to supporting individualized learning, LLMs can address broader educational challenges, such as the low teacher-student ratio, by supplementing human instructors and providing additional support to students [133]. The integration of LLMs into education presents a unique opportunity to personalize and scale educational resources, making quality education more accessible to a broader range of learners.

5 Drawbacks and future directions of large language models

While LLMs offer significant advantages in improving both professional and personal aspects of life, it is essential to acknowledge their accompanying drawbacks. In the subsequent discussion, we will identify and explore some of these limitations inherent in current LLMs, which concurrently provide insights into potential directions for future advancements.

Privacy LLMs are widely used across various sectors, but they also present significant privacy concerns [56, 134]. One major issue is the risk of data leakage, where LLMs may inadvertently memorize and reproduce sensitive or personal information from their training datasets, potentially leading to privacy breaches. This risk is particularly severe in domains like healthcare and finance, where exposing private data can have serious consequences [135]. Additionally, the training datasets used for LLMs are often not fully anonymized, meaning personal identifiers and other sensitive information could be embedded within the model [136]. This increases the risk of violating privacy regulations, such as the General Data Protection Regulation [137]. Recent developments have further complicated the privacy landscape, as tools like FraudGPT and WormGPT have emerged. These tools, designed for cybercrime, highlight the misuse potential of LLMs by generating fraudulent emails, suggesting malicious link placements, and facilitating cyberattacks. FraudGPT [138, 139] and WormGPT [140] were trained on datasets focused on malware and fraud, enabling cybercriminals to conduct sophisticated attacks such as Business Email Compromise (BEC). This illustrates the urgent need for robust privacy safeguards and regulatory frameworks to

[8] https://sites.highlands.edu/tutorial-center/tutor-resources/online-tutor-training/module-4/the-socratic-method/.

[9] How ChatGPT and other new AI tools are being used by lawyers, architects and coders - ABC News.

[10] https://data.delilegal.com/lawQuestion.

prevent the misuse of LLMs and protect sensitive information from being exploited maliciously.

Fairness LLMs are widely employed for decision-making across various domains, impacting individuals and society. However, these decisions might exhibit bias against marginalized populations. For instance, when utilized in screening resumes for programming positions, LLMs might exhibit a bias favoring male candidates, demonstrating evident gender discrimination [141–144]. This bias stems from the training of LLMs on extensive and unstructured Internet data, where they inherit stereotypes, misrepresentations, derogatory language, and exclusionary behaviors. These aspects disproportionately affect already disadvantaged and marginalized groups [15, 22, 145]. The commonly employed strategy to mitigate bias in LLMs is to remove biased data from the training dataset. However, this strategy does not entirely eliminate bias and often diminishes the effectiveness of language modeling [146]. Saxena et al. [39] discussed that the root causes of these problems lie not only in technology but also in socially acceptable definitions of fairness and meaningful interventions to ensure the long-term well-being of all groups. Researchers should consider the needs of disadvantaged groups from the outset and design these technologies proactively, rather than simply reacting to the biases present in their designed systems [40, 147, 148]. The documented bias in LLMs, as highlighted by studies like [45, 149, 150], underscores the imperative for future efforts in this direction.

Safety LLMs have diverse applications in industries such as finance and healthcare. However, these applications also present significant security challenges. LLMs may generate outputs that diverge from the provided context, user input, or factual knowledge, a phenomenon referred to as "hallucination" [21, 57]. This undermines their reliability in real-world scenarios, emphasizing the vital necessity for precision, particularly in critical domains like medicine [151, 152]. As a primary way to avoid these problems, well-aligned LLMs can be developed by including humans in the training loop and using reinforcement learning from human feedback (RLHF) [66, 100]. To improve model safety, it is also important to include safety-related cues in the RLHF process, as shown in GPT-4 [11]. However, RLHF relies heavily on high-quality human feedback data from professional annotators, which is costly and time-consuming. As a result, enhancing the RLHF framework is essential to alleviate the burden on human annotators. Additionally, exploring more efficient annotation methods with assured data quality, such as LLMs, can assist in the annotation process and enhance overall safety.

Intellectual Property LLMs possess the capability to create content resembling human-generated material, potentially encroaching upon users' intellectual property rights. These models rely on training and optimizing using users' textual data, which may encompass personal information, proprietary knowledge, patented technology, and more. Notably, AI-generated code exemplifies this issue. Code-generation models trained on open-source repositories can produce programs resembling existing ones, potentially disregarding relevant licenses [34]. For instance, legal action has been taken against Microsoft, GitHub, and OpenAI, alleging copyright infringement due to Copilot reproducing licensed code without adherence to licensing terms[11].[12] Additionally, there are concerns about potential copyright infringements with LLM-generated images and texts [153, 154], as highlighted in recent lawsuits by a group of novelists against OpenAI in July 2023 for allegedly using their books without permission to train models [155], and by The New York Times against OpenAI and Microsoft in December 2023 for scraping articles to train their generative models without consent [156]. Consequently, there's a pressing need for heightened attention and regulation concerning intellectual property rights concerning LLMs. This is essential to safeguard the legitimate rights and interests of original creators, users, and the public while fostering and respecting the innovation and advancement of LLMs technology [157].

6 Conclusions

In recent times, Large Language Models (LLMs) have garnered significant attention from the fields of science, industry, society, and beyond. However, despite the substantial advantages that LLMs offer in enhancing both professional and personal aspects of life, a limited understanding among general practitioners regarding the background and principles of these models impedes their full potential. In light of this, the survey at hand systematically delves into the evolution of LLMs, presenting key concepts and techniques for a comprehensive understanding of these models. It introduces a detailed taxonomy of LLMs and compares state-of-the-art models to assist readers in selecting the most suitable model for their specific needs. Additionally, the survey highlights the limitations of current LLMs and identifies promising areas for future research. Notably, setting itself apart from other surveys on LLMs, this study prioritizes empowering practitioners, irrespective of their background knowledge, to ensure their proficient utilization in both scientific research and daily tasks.

[11] https://www.reuters.com/technology/microsoft-attracting-users-its-code-writing-generative-ai-software-2023-01-25/.

[12] https://www.reuters.com/legal/litigation/openai-microsoft-want-court-toss-lawsuit-accusing-them-abusing-open-source-code-2023-01-27/.

References

1. Pinker, S.: The Language Instinct: How the Mind Creates. Harper Collins Language, New York (1994)
2. Turing, A.M., Geirsson, H., Losonsky, M.: Computing machinery and intelligence. Artif. Intell. Crit. Concepts **2**(236), 19 (2000)
3. Dwivedi, Y.K., Kshetri, N., Hughes, L., Slade, E.L., Jeyaraj, A., Kar, A.K., Baabdullah, A.M., Koohang, A., Raghavan, V., Ahuja, M., et al.: "so what if chatgpt wrote it?" multidisciplinary perspectives on opportunities, challenges and implications of generative conversational ai for research, practice and policy. Int. J. Inf. Manage. **71**, 102642 (2023)
4. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al.: A survey of large language models (2023). arXiv preprint arXiv:2303.18223
5. Jin, H., Wei, W., Wang, X., Zhang, W., Wu, Y.: Rethinking Learning Rate Tuning in the Era of Large Language Models, pp. 112–121 (2023). IEEE
6. Fu, Y., Peng, H., Khot, T.: How does gpt obtain its ability? Tracing emergent abilities of language models to their sources. Yao Fu's Notion (2022)
7. Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al.: Emergent abilities of large language models. Trans. Mach. Learn. Res. (2022)
8. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training (2018)
9. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)
10. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Adv. Neural. Inf. Process. Syst. **33**, 1877–1901 (2020)
11. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report (2023). arXiv preprint arXiv:2303.08774
12. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. ACM Comput. Surv. **55**(9), 1–35 (2023)
13. Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Yao, Y., Zhang, A., Zhang, L., et al.: Pre-trained models: past, present and future. AI Open **2**, 225–250 (2021)
14. Shanahan, M.: Talking about large language models. Commun. ACM **67**(2), 68–79 (2024)
15. Dodge, J., Sap, M., Marasović, A., Agnew, W., Ilharco, G., Groeneveld, D., Mitchell, M., Gardner, M.: 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 (EMNLP), pp. 1286–1305 (2021)
16. Sonkar, S., Liu, N., Mallick, D., Baraniuk, R.: Class: A design framework for building intelligent tutoring systems based on learning science principles. In: Findings of the Association for Computational Linguistics: EMNLP 2023, pp. 1941–1961 (2023)
17. Kim, B., Kim, H., Lee, S.-W., Lee, G., Kwak, D., Hyeon, J.D., Park, S., Kim, S., Kim, S., Seo, D., et al.: What changes can large-scale language models bring? intensive study on hyperclova: billions-scale Korean generative pretrained transformers. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3405–3424 (2021)
18. Tay, Y., Wei, J., Chung, H., Tran, V., So, D., Shakeri, S., Garcia, X., Zheng, S., Rao, J., Chowdhery, A., et al.: Transcending scaling laws with 0.1% extra compute. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1471–1486 (2023)
19. Ahmad, W., Chakraborty, S., Ray, B., Chang, K.: Unified pre-training for program understanding and generation. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2021)
20. Muennighoff, N., Wang, T., Sutawika, L., Roberts, A., Biderman, S., Le Scao, T., Bari, M.S., Shen, S., Yong, Z.X., Schoelkopf, H., et al.: Crosslingual generalization through multitask finetuning. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 15991–16111 (2023)
21. Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., et al.: A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. In: Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 675–718 (2023)
22. Sheng, E., Chang, K.-W., Natarajan, P., Peng, N.: Societal biases in language generation: progress and challenges. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 4275–4293 (2021)
23. Fried, D., Aghajanyan, A., Lin, J., Wang, S., Wallace, E., Shi, F., Zhong, R., Yih, S., Zettlemoyer, L., Lewis, M.: Incoder: a generative model for code infilling and synthesis. In: The Eleventh International Conference on Learning Representations
24. Nijkamp, E., Pang, B., Hayashi, H., Tu, L., Wang, H., Zhou, Y., Savarese, S., Xiong, C.: Codegen: an open large language model for code with multi-turn program synthesis. In: The Eleventh International Conference on Learning Representations (2022)
25. Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A.W., Lester, B., Du, N., Dai, A.M., Le, Q.V.: Finetuned language models are zero-shot learners. In: International Conference on Learning Representations (2021)
26. Sanh, V., Webson, A., Raffel, C., Bach, S.H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Le Scao, T., Raja, A., et al.: Multitask prompted training enables zero-shot task generalization. In: ICLR 2022-Tenth International Conference on Learning Representations (2022)
27. Tay, Y., Dehghani, M., Tran, V.Q., Garcia, X., Wei, J., Wang, X., Chung, H.W., Bahri, D., Schuster, T., Zheng, S., et al.: Ul2: Unifying language learning paradigms. In: The Eleventh International Conference on Learning Representations
28. Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., et al.: Glm-130b: an open bilingual pre-trained model. In: The Eleventh International Conference on Learning Representations
29. He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention. In: International Conference on Learning Representations
30. Merity, S., Xiong, C., Bradbury, J., Socher, R.: Pointer sentinel mixture models (2022)
31. Du, N., Huang, Y., Dai, A.M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A.W., Firat, O., et al.: Glam: efficient scaling of language models with mixture-of-experts. In: International Conference on Machine Learning, pp. 5547–5569 (2022). PMLR
32. Biderman, S., Schoelkopf, H., Anthony, Q.G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M.A., Purohit, S., Prashanth,

U.S., Raff, E., et al.: Pythia: A suite for analyzing large language models across training and scaling. In: International Conference on Machine Learning, pp. 2397–2430 (2023). PMLR
33. Wang, T., Roberts, A., Hesslow, D., Le Scao, T., Chung, H.W., Beltagy, I., Launay, J., Raffel, C.: What language model architecture and pretraining objective works best for zero-shot generalization? In: International Conference on Machine Learning, pp. 22964–22984 (2022). PMLR
34. Yu, Z., Wu, Y., Zhang, N., Wang, C., Vorobeychik, Y., Xiao, C.: Codeipprompt: intellectual property infringement assessment of code language models. In: International Conference on Machine Learning, pp. 40373–40389 (2023). PMLR
35. Steenhoek, B., Rahman, M.M., Jiles, R., Le, W.: An empirical study of deep learning models for vulnerability detection. In: 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), pp. 2237–2248 (2023). IEEE
36. Yin, Z., Wang, Z., Zhang, W.: Improving fairness in machine learning software via counterfactual fairness thinking. In: Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings, pp. 420–421 (2024)
37. Li, Y., Wang, S., Ding, H., Chen, H.: Large language models in finance: a survey. In: Proceedings of the Fourth ACM International Conference on AI in Finance, pp. 374–382 (2023)
38. Pagliaro, C., Mehta, D., Shiao, H.-T., Wang, S., Xiong, L.: Investor behavior modeling by analyzing financial advisor notes: a machine learning perspective. In: Proceedings of the Second ACM International Conference on AI in Finance, pp. 1–8 (2021)
39. Saxena, N.A., Zhang, W., Shahabi, C.: Missed opportunities in fair ai. In: Proceedings of the 2023 SIAM International Conference on Data Mining (SDM), pp. 961–964 (2023). SIAM
40. Wang, Z., Narasimhan, G., Yao, X., Zhang, W.: Mitigating multisource biases in graph neural networks via real counterfactual samples. In: 2023 IEEE International Conference on Data Mining (ICDM), pp. 638–647 (2023). IEEE
41. Chinta, S.V., Fernandes, K., Cheng, N., Fernandez, J., Yazdani, S., Yin, Z., Wang, Z., Wang, X., Xu, W., Liu, J., et al.: Optimization and improvement of fake news detection using voting technique for societal benefit. In: 2023 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 1565–1574 (2023). IEEE
42. Xiao, C., Xu, S.X., Zhang, K., Wang, Y., Xia, L.: Evaluating reading comprehension exercises generated by llms: a showcase of chatgpt in education applications. In: Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023), pp. 610–625 (2023)
43. Gupta, A., Dengre, V., Kheruwala, H.A., Shah, M.: Comprehensive review of text-mining applications in finance. Financ. Innov. **6**, 1–25 (2020)
44. Kung, T.H., Cheatham, M., Medenilla, A., Sillos, C., De Leon, L., Elepaño, C., Madriaga, M., Aggabao, R., Diaz-Candido, G., Maningo, J., et al.: Performance of chatgpt on usmle: potential for ai-assisted medical education using large language models. PLoS Digit. Health **2**(2), 0000198 (2023)
45. Mozafari, M., Farahbakhsh, R., Crespi, N.: Hate speech detection and racial bias mitigation in social media based on bert model. PLoS ONE **15**(8), 0237861 (2020)
46. Jin, D., Pan, E., Oufattole, N., Weng, W.-H., Fang, H., Szolovits, P.: What disease does this patient have? a large-scale open domain question answering dataset from medical exams. Appl. Sci. **11**(14), 6421 (2021)
47. Kombrink, S., Mikolov, T., Karafiát, M., Burget, L.: Recurrent neural network based language modeling in meeting recognition. Interspeech **11**, 2877–2880 (2011)
48. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech, vol. 2, pp. 1045–1048 (2010). Makuhari
49. Stolcke, A., et al.: Srilm—an extensible language modeling toolkit. Interspeech **2002**, 2002 (2002)
50. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., et al.: Scaling instruction-finetuned language models. J. Mach. Learn. Res. **25**(70), 1–53 (2024)
51. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., et al.: Palm: scaling language modeling with pathways. J. Mach. Learn. Res. **24**(240), 1–113 (2023)
52. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. J. Mach. Learn. Res. **23**(120), 1–39 (2022)
53. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
54. Baidoo-Anu, D., Ansah, L.O.: Education in the era of generative artificial intelligence (ai): understanding the potential benefits of chatgpt in promoting teaching and learning. J. AI **7**(1), 52–62 (2023)
55. Chen, Z.Z., Ma, J., Zhang, X., Hao, N., Yan, A., Nourbakhsh, A., Yang, X., McAuley, J., Petzold, L., Wang, W.Y.: A survey on large language models for critical societal domains: finance, healthcare, and law (2024). arXiv preprint arXiv:2405.01769
56. Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., Zhang, Y.: A survey on large language model (llm) security and privacy: the good, the bad, and the ugly. High-Confiden. Comput. 100211 (2024)
57. Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al.: A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions (2023). arXiv preprint arXiv:2311.05232
58. Jelinek, F.: Statistical Methods for Speech Recognition. MIT Press, Cambridge (1998)
59. Rosenfeld, R.: Two decades of statistical language modeling: where do we go from here? Proc. IEEE **88**(8), 1270–1278 (2000)
60. Bengio, Y., Ducharme, R., Vincent, P.: A neural probabilistic language model. Adv. Neural Inf. Process. Syst. **13** (2000)
61. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural Inf. Process. Syst. **26** (2013)
62. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018). arXiv preprint arXiv:1810.04805
63. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Roziere, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models (2023). arXiv preprint arXiv:2302.13971
64. Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., Wu, Y.: Exploring the limits of language modeling (2016). arXiv preprint arXiv:1602.02410
65. Kiros, R., Zhu, Y., Salakhutdinov, R.R., Zemel, R., Urtasun, R., Torralba, A., Fidler, S.: Skip-thought vectors. Adv. Neural Inf. Process. Syst. **28** (2015)
66. Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al.: The pile: an 800gb dataset of diverse text for language modeling (2020). arXiv preprint arXiv:2101.00027
67. Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D.: Scaling laws for neural language models (2020). arXiv preprint arXiv:2001.08361

68. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
69. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
70. Cui, J., Li, Z., Yan, Y., Chen, B., Yuan, L.: Chatlaw: open-source legal large language model with integrated external knowledge bases (2023). arXiv preprint arXiv:2306.16092
71. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. Adv. Neural Inf. Process. Syst. **32** (2019)
72. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite bert for self-supervised learning of language representations (2019). arXiv preprint arXiv:1909.11942
73. Kaiming, H., Xiangyu, Z., Shaoqing, R., Jian, S., et al.: Deep residual learning for image recognition. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. **34**, 770–778 (2016)
74. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization (2016). arXiv preprint arXiv:1607.06450
75. Wang, C., Li, M., Smola, A.J.: Language models with transformers (2019). arXiv preprint arXiv:1904.09408
76. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H.P.D.O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al.: Evaluating large language models trained on code (2021). arXiv preprint arXiv:2107.03374
77. Clark, K.: Electra: pre-training text encoders as discriminators rather than generators (2020). arXiv preprint arXiv:2003.10555
78. Zhuang, L., Wayne, L., Ya, S., Jun, Z.: A robustly optimized bert pre-training approach with post-training. In: Proceedings of the 20th Chinese National Conference on Computational Linguistics, pp. 1218–1227 (2021)
79. Conneau, A.: Unsupervised cross-lingual representation learning at scale (2019). arXiv preprint arXiv:1911.02116
80. Sun, Y., Wang, S., Feng, S., Ding, S., Pang, C., Shang, J., Liu, J., Chen, X., Zhao, Y., Lu, Y., et al.: Ernie 3.0: large-scale knowledge enhanced pre-training for language understanding and generation (2021). arXiv preprint arXiv:2107.02137
81. Soltan, S., Ananthakrishnan, S., FitzGerald, J., Gupta, R., Hamza, W., Khan, H., Peris, C., Rawls, S., Rosenbaum, A., Rumshisky, A., et al.: Alexatm 20b: few-shot learning using a large-scale multilingual seq2seq model (2022). arXiv preprint arXiv:2208.01448
82. Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., et al.: Competition-level code generation with alphacode. Science **378**(6624), 1092–1097 (2022)
83. Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al.: A general language assistant as a laboratory for alignment (2021). arXiv preprint arXiv:2112.00861
84. Costa-jussà, M.R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., et al.: No language left behind: Scaling human-centered machine translation (2022). arXiv preprint arXiv:2207.04672
85. Glaese, A., McAleese, N., Trębacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., et al.: Improving alignment of dialogue agents via targeted human judgements (2022). arXiv preprint arXiv:2209.14375
86. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Las Casas, D., Hendricks, L.A., Welbl, J., Clark, A., et al.: Training compute-optimal large language models. In: Proceedings of the 36th International Conference on Neural Information Processing Systems, pp. 30016–30030 (2022)
87. Iyer, S., Lin, X.V., Pasunuru, R., Mihaylov, T., Simig, D., Yu, P., Shuster, K., Wang, T., Liu, Q., Koura, P.S., et al.: Opt-iml: scaling language model instruction meta learning through the lens of generalization (2022). arXiv preprint arXiv:2212.12017
88. Rae, J.W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al.: Scaling language models: methods, analysis & insights from training gopher (2021). arXiv preprint arXiv:2112.11446
89. Zheng, Q., Xia, X., Zou, X., Dong, Y., Wang, S., Xue, Y., Wang, Z., Shen, L., Wang, A., Li, Y., et al.: Codegeex: a pre-trained model for code generation with multilingual evaluations on humaneval-x (2023). arXiv preprint arXiv:2303.17568
90. Wei, T., Zhao, L., Zhang, L., Zhu, B., Wang, L., Yang, H., Li, B., Cheng, C., Lü, W., Hu, R., et al.: Skywork: a more open bilingual foundation model (2023). arXiv preprint arXiv:2310.19341
91. Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al.: Qwen technical report (2023). arXiv preprint arXiv:2309.16609
92. Nijkamp, E., Hayashi, H., Xiong, C., Savarese, S., Zhou, Y.: Codegen2: Lessons for training llms on programming and natural languages (2023). arXiv preprint arXiv:2305.02309
93. Black, S., Biderman, S., Hallahan, E., Anthony, Q.G., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., et al.: Gpt-neox-20b: an open-source autoregressive language model. In: Challenges { \ &} Perspectives in Creating Large Language Models
94. Reid, M., Savinov, N., Teplyashin, D., Lepikhin, D., Lillicrap, T., Alayrac, J.-b., Soricut, R., Lazaridou, A., Firat, O., Schrittwieser, J., et al.: Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context (2024). arXiv preprint arXiv:2403.05530
95. Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al.: The llama 3 herd of models (2024). arXiv preprint arXiv:2407.21783
96. Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., Stojnic, R.: Galactica: A large language model for science (2022). arXiv preprint arXiv:2211.09085
97. Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al.: Lamda: Language models for dialog applications (2022). arXiv preprint arXiv:2201.08239
98. Lieber, O., Sharir, O., Lenz, B., Shoham, Y.: Jurassic-1: Technical details and evaluation. White Paper. AI21 Labs **1**(9) (2021)
99. Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al.: Webgpt: browser-assisted question-answering with human feedback (2021). arXiv preprint arXiv:2112.09332
100. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al.: Training language models to follow instructions with human feedback. Adv. Neural. Inf. Process. Syst. **35**, 27730–27744 (2022)
101. Le Scao, T., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A.S., Yvon, F., Gallé, M., et al.: Bloom: a 176b-parameter open-access multilingual language model (2023)
102. Vavekanand, R., Sam, K.: Llama 3.1: an in-depth analysis of the next-generation large language model
103. Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., et al.: Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model (2022). arXiv preprint arXiv:2201.11990
104. Team, G., Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M.S., Love, J., et al.: Gemma: open models based on gemini research and technology (2024). arXiv preprint arXiv:2403.08295
105. Li, Z., Lu, S., Guo, D., Duan, N., Jannu, S., Jenks, G., Majumder, D., Green, J., Svyatkovskiy, A., Fu, S., et al.: Automating code

review activities by large-scale pre-training. In: Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 1035–1047 (2022)
106. He, J., Zhou, X., Xu, B., Zhang, T., Kim, K., Yang, Z., Thung, F., Irsan, I.C., Lo, D.: Representation learning for stack overflow posts: how far are we? ACM Trans. Softw. Eng. Methodol. **33**(3), 1–24 (2024)
107. He, J., Xu, B., Yang, Z., Han, D., Yang, C., Lo, D.: Ptm4tag: sharpening tag recommendation of stack overflow posts with pre-trained models. In: Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, pp. 1–11 (2022)
108. Yang, C., Xu, B., Thung, F., Shi, Y., Zhang, T., Yang, Z., Zhou, X., Shi, J., He, J., Han, D., et al.: Answer summarization for technical queries: benchmark and new approach. In: Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering, pp. 1–13 (2022)
109. Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X.E., Adi, Y., Liu, J., Remez, T., Rapin, J., et al.: Code llama: open foundation models for code (2023). arXiv preprint arXiv:2308.12950
110. Le, H., Wang, Y., Gotmare, A.D., Savarese, S., Hoi, S.C.H.: Coderl: mastering code generation through pretrained models and deep reinforcement learning. Adv. Neural. Inf. Process. Syst. **35**, 21314–21328 (2022)
111. Sadybekov, A.V., Katritch, V.: Computational approaches streamlining drug discovery. Nature **616**(7958), 673–685 (2023)
112. Gorgulla, C., Jayaraj, A., Fackeldey, K., Arthanari, H.: Emerging frontiers in virtual drug discovery: from quantum mechanical methods to deep learning approaches. Curr. Opin. Chem. Biol. **69**, 102156 (2022)
113. Savage, N.: Drug discovery companies are customizing chatgpt: here's how. Nat. Biotechnol. **41**(5), 585–586 (2023)
114. Haley, B., Roudnicky, F.: Functional genomics for cancer drug target discovery. Cancer Cell **38**(1), 31–43 (2020)
115. Paananen, J., Fortino, V.: An omics perspective on drug target discovery platforms. Brief. Bioinform. **21**(6), 1937–1953 (2020)
116. Zhang, Z., Zohren, S., Roberts, S.: Deep learning for portfolio optimization. J. Financ. Data Sci. (2020)
117. Mashrur, A., Luo, W., Zaidi, N.A., Robles-Kelly, A.: Machine learning for financial risk management: a survey. Ieee Access **8**, 203203–203223 (2020)
118. Shah, A., Raj, P., Pushpam Kumar, S.P., Asha, H.: Finaid, a financial advisor application using ai
119. Misischia, C.V., Poecze, F., Strauss, C.: Chatbots in customer service: their relevance and impact on service quality. Procedia Comput. Sci. **201**, 421–428 (2022)
120. Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., Mann, G.: Bloomberggpt: a large language model for finance (2023). arXiv preprint arXiv:2303.17564
121. Thirunavukarasu, A.J., Ting, D.S.J., Elangovan, K., Gutierrez, L., Tan, T.F., Ting, D.S.W.: Large language models in medicine. Nat. Med. **29**(8), 1930–1940 (2023)
122. Singhal, K., Tu, T., Gottweis, J., Sayres, R., Wulczyn, E., Hou, L., Clark, K., Pfohl, S., Cole-Lewis, H., Neal, D., et al.: Towards expert-level medical question answering with large language models (2023). arXiv preprint arXiv:2305.09617
123. Arora, A., Arora, A.: The promise of large language models in health care. Lancet **401**(10377), 641 (2023)
124. Bommarito II, M., Katz, D.M.: Gpt takes the bar exam (2022). arXiv preprint arXiv:2212.14402
125. Iu, K.Y., Wong, V.M.-Y.: Chatgpt by openai: the end of litigation lawyers? Available at SSRN 4339839 (2023)
126. Lee, U., Lee, S., Koh, J., Jeong, Y., Jung, H., Byun, G., Lee, Y., Moon, J., Lim, J., Kim, H.: Generative Agent for Teacher Training: Designing Educational Problem-Solving Simulations with Large Language Model-based Agents for Pre-Service Teachers. NeurIPS
127. Markel, J.M., Opferman, S.G., Landay, J.A., Piech, C.: Gpteach: Interactive ta training with gpt-based students. In: Proceedings of the Tenth Acm Conference on Learning@ Scale, pp. 226–236 (2023)
128. Tu, S., Zhang, Z., Yu, J., Li, C., Zhang, S., Yao, Z., Hou, L., Li, J.: Littlemu: Deploying an online virtual teaching assistant via heterogeneous sources integration and chain of teach prompts. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 4843–4849 (2023)
129. Chen, Y., Ding, N., Zheng, H.-T., Liu, Z., Sun, M., Zhou, B.: Empowering private tutoring by chaining large language models (2023). arXiv preprint arXiv:2309.08112
130. Zentner, A.: Applied innovation: artificial intelligence in higher education. Available at SSRN 4314180 (2022)
131. Zhang, B.: Preparing educators and students for chatgpt and ai technology in higher education. ResearchGate (2023)
132. Dwivedi, Y.K., Kshetri, N., Hughes, L., Slade, E.L., Jeyaraj, A., Kar, A.K., Baabdullah, A.M., Koohang, A., Raghavan, V., Ahuja, M., et al.: Opinion paper:"so what if chatgpt wrote it?" multidisciplinary perspectives on opportunities, challenges and implications of generative conversational ai for research, practice and policy. Int. J. Inf. Manage. **71**, 102642 (2023)
133. Chen, Y., Jensen, S., Albert, L.J., Gupta, S., Lee, T.: Artificial intelligence (ai) student assistants in the classroom: designing chatbots to support student success. Inf. Syst. Front. **25**(1), 161–182 (2023)
134. Yan, B., Li, K., Xu, M., Dong, Y., Zhang, Y., Ren, Z., Cheng, X.: On protecting the data privacy of large language models (llms): a survey (2024). arXiv preprint arXiv:2403.05156
135. Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al.: Extracting training data from large language models. In: 30th USENIX Security Symposium (USENIX Security 21), pp. 2633–2650 (2021)
136. Fredrikson, M., Jha, S., Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1322–1333 (2015)
137. Leboukh, F., Aduku, E.B., Ali, O.: Balancing chatgpt and data protection in Germany: challenges and opportunities for policy makers. J. Polit. Ethics New Technol. AI **2**(1), 35166–35166 (2023)
138. Falade, P.V.: Decoding the threat landscape: Chatgpt, fraudgpt, and wormgpt in social engineering attacks (2023). arXiv preprint arXiv:2310.05595
139. Amos, Z.: What is fraudgpt? (2023)
140. Delley, D.: Wormgpt—the generative ai tool cybercriminals are using to launch business email compromise attacks. SlashNext. Retrieved August 24, 2023 (2023)
141. Chu, Z., Wang, Z., Zhang, W.: Fairness in large language models: a taxonomic survey. ACM SIGKDD Explor. Newsl. **2024**, 34–48 (2024)
142. Doan, T.V., Wang, Z., Nguyen, M.N., Zhang, W.: Fairness in large language models in three hours. In: Proceedings of the 33rd ACM International Conference on Information & Knowledge Management (Boise, USA) (2024)
143. Doan, T., Chu, Z., Wang, Z., Zhang, W.: Fairness definitions in language models explained (2024). arXiv preprint arXiv:2407.18454
144. Zhang, W.: Ai fairness in practice: paradigm, challenges, and prospects. Ai Mag. (2024)
145. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: can language models be too big? In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, pp. 610–623 (2021)

146. Meade, N., Poole-Dayan, E., Reddy, S.: An empirical survey of the effectiveness of debiasing techniques for pre-trained language models (2021). arXiv preprint arXiv:2110.08527
147. Gallegos, I.O., Rossi, R.A., Barrow, J., Tanjim, M.M., Kim, S., Dernoncourt, F., Yu, T., Zhang, R., Ahmed, N.K.: Bias and fairness in large language models: a survey. Comput. Linguist. 1–79 (2024)
148. Wang, Z., Chu, Z., Blanco, R., Chen, Z., Chen, S.-C., Zhang, W.: Advancing graph counterfactual fairness through fair representation learning. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases (2024). Springer Nature Switzerland
149. Blodgett, S.L., O'Connor, B.: Racial disparity in natural language processing: a case study of social media African-American English (2017). arXiv preprint arXiv:1707.00061
150. Mei, K., Fereidooni, S., Caliskan, A.: Bias against 93 stigmatized groups in masked language models and downstream sentiment classification tasks. In: Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency, pp. 1699–1710 (2023)
151. Dash, D., Thapa, R., Banda, J.M., Swaminathan, A., Cheatham, M., Kashyap, M., Kotecha, N., Chen, J.H., Gombar, S., Downing, L., et al.: Evaluation of gpt-3.5 and gpt-4 for supporting real-world information needs in healthcare delivery (2023). arXiv preprint arXiv:2304.13714
152. Pal, A., Umapathi, L.K., Sankarasubbu, M.: Med-halt: Medical domain hallucination test for large language models. In: Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL), pp. 314–334 (2023)
153. Dzuong, J., Wang, Z., Zhang, W.: Uncertain boundaries: multidisciplinary approaches to copyright issues in generative ai (2024). arXiv preprint arXiv:2404.08221
154. Yazdani, S., Saxena, N., Wang, Z., Wu, Y., Zhang, W.: A comprehensive survey of image and video generative ai: Recent advances, variants, and applications (2024)
155. Small, Z.: Sarah silverman sues openai and meta over copyright infringement. The New York Times (2023)
156. Stempel, J.: NY Times sues openai, Microsoft for infringing copyrighted works... Thomson Reuters Corporation (2023). https://www.reuters.com/legal/transactional/ny-times-sues-openai-microsoft-infringing-copyrighted-work-2023-12-27/
157. Li, Z., Wang, C., Wang, S., Gao, C.: Protecting intellectual property of large language model-based code generation apis via watermarks. In: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, pp. 2336–2350 (2023)