Large Language Models for Computer-Aided Design: A Survey

LICHENG ZHANG*, The University of Melbourne, Australia

BACH LE, The University of Melbourne, Australia

NAVEED AKHTAR, The University of Melbourne, Australia

SIEW-KEI LAM, Nanyang Technological University, Singapore

TUAN NGO, The University of Melbourne, Australia

Large Language Models (LLMs) have seen rapid advancements in recent years, with models like ChatGPT and DeepSeek, showcasing their remarkable capabilities across diverse domains. While substantial research has been conducted on LLMs in various fields, a comprehensive review focusing on their integration with Computer-Aided Design (CAD) remains notably absent. CAD is the industry standard for 3D modeling and plays a vital role in the design and development of products across different industries. As the complexity of modern designs increases, the potential for LLMs to enhance and streamline CAD workflows presents an exciting frontier. This article presents the first systematic survey exploring the intersection of LLMs and CAD. We begin by outlining the industrial significance of CAD, highlighting the need for AI-driven innovation. Next, we provide a detailed overview of the foundation of LLMs. We also examine both closed-source LLMs as well as publicly available models. The core of this review focuses on the various applications of LLMs in CAD, providing a taxonomy of six key areas where these models are making considerable impact. Finally, we propose several promising future directions for further advancements, which offer vast opportunities for innovation and are poised to shape the future of CAD technology. Github: https://github.com/lichengzhanguom/LLMs-CAD-Survey-Taxonomy

CCS Concepts: • **Applied computing → Computer-aided design**; • **Information systems → Language models**; • **Computing methodologies** → *Natural language processing*; *Computer vision tasks.*

Additional Key Words and Phrases: Computer-Aided Design, Large Language Models, Vision-Language Models, CAD Code Generation, Parametric CAD Generation

ACM Reference Format:

Licheng Zhang, Bach Le, Naveed Akhtar, Siew-Kei Lam, and Tuan Ngo. 2025. Large Language Models for Computer-Aided Design: A Survey. *ACM Comput. Surv.* 37, 4, Article 111 (May 2025), 31 pages. https://doi.org/XXXXXXX.XXXXXXX

*Corresponding author.

Authors' Contact Information: Licheng Zhang, The University of Melbourne, Parkville, Victoria, Australia, licheng.zhang@student.unimelb.edu.au; Bach Le, The University of Melbourne, Australia, bach.le@unimelb.edu.au; Naveed Akhtar, The University of Melbourne, Australia, naveed.akhtar1@unimelb.edu.au; Siew-Kei Lam, Nanyang Technological University, Singapore, ASSKLam@ntu.edu.sg; Tuan Ngo, The University of Melbourne, Australia, dtngo@unimelb.edu.au.

Manuscript submitted to ACM

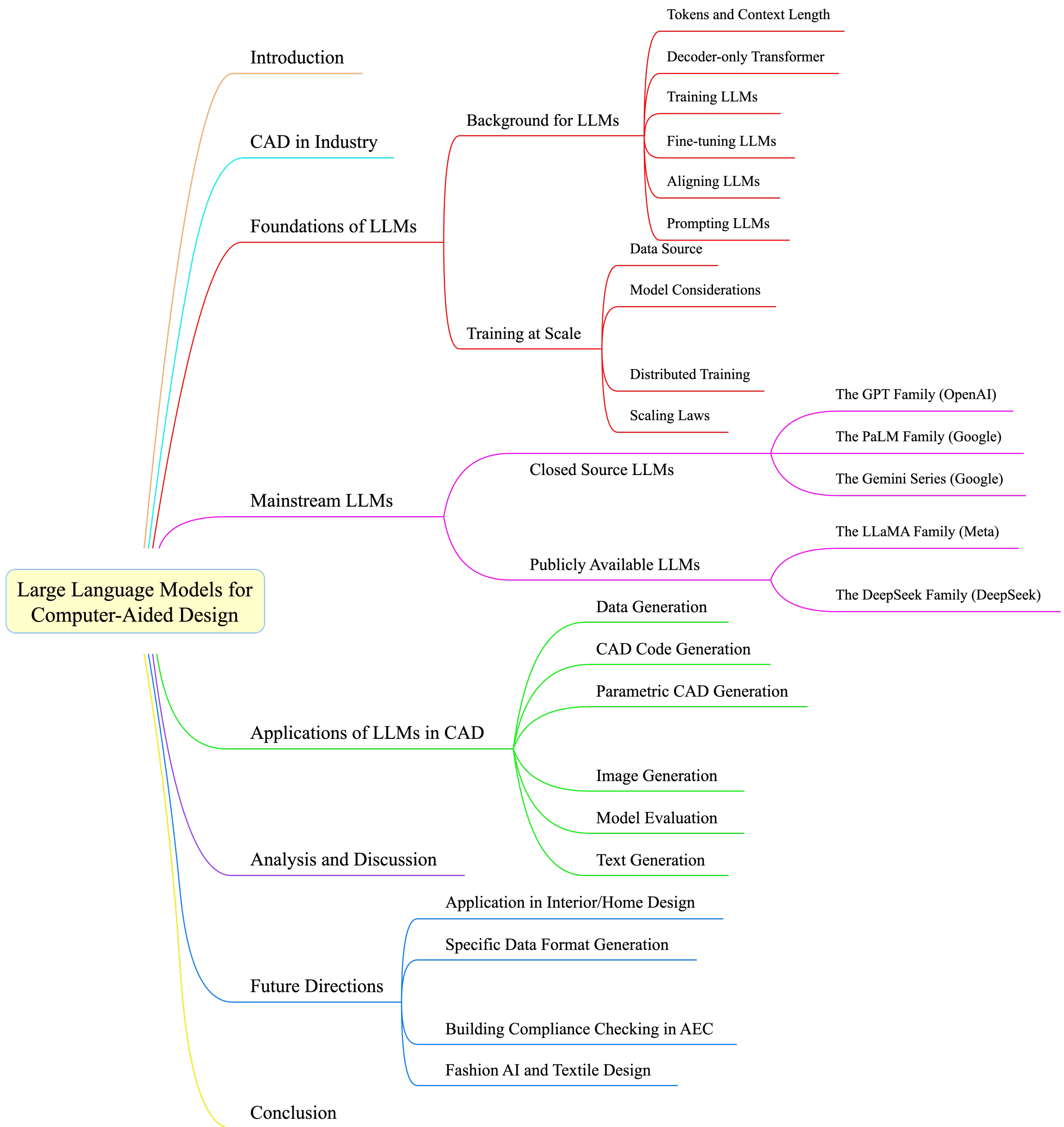

Fig. 1. The taxonomy of this review.

1 Introduction

Language is a fundamental aspect of human intelligence, as essential as vision. For decades, researchers have strived to endow machines with the ability to reason and communicate in natural language [212]. In recent years, this vision has come closer to reality with the advent of Large Language Models (LLMs), which have demonstrated remarkable progress across a wide range of tasks [154]. A key insight driving LLMs is that knowledge about the world can be effectively captured and represented through large-scale language modeling. This enables the construction of general-purpose models that can tackle a variety of problems via pre-training and task-specific fine-tuning, alignment, or prompting—bypassing the need for extensive domain-specific training from scratch [193].

LLMs are typically built upon the Transformer architecture [174], with a vastly larger number of parameters compared to earlier Pre-trained Language Models (PLMs). This increase in scale has resulted in significantly improved capabilities in language understanding and generation. The release of ChatGPT [147] marked a major milestone, showcasing unprecedented performance. Since then, many powerful LLMs have been introduced, including GPT-4 [2] and GPT-4V [125] by OpenAI; LLaMA [169], LLaMA-2 [170], and LLaMA-3 [49] by Meta; and Gemini [163], PaLM [31], PaLM-2 [7], and GLaM [42] by Google, among others. Parallel to these advancements in LLMs, Computer-Aided Design (CAD) has remained a cornerstone technology in engineering and industrial design. CAD refers to the use of computers to assist in the creation, modification, analysis, or optimization of a design [50, 113, 136, 146]. It is widely used to produce 2D drawings and 3D models of physical products, supporting applications in fields such as architecture, automotive design, manufacturing, and 3D printing. CAD enhances design precision, facilitates rapid iteration, and reduces both development time and cost. Moreover, CAD systems preserve essential design information—such as geometry, dimensions, and structural details—in standardized file formats, promoting reusability and collaboration.

Despite the growing popularity of LLMs and the critical importance of CAD in modern industry, no survey article to date has systematically explored the intersection of these two domains. While reviewing efforts have been made for various LLM applications [8, 53, 54, 57, 82, 115, 120, 154, 202, 212], a comprehensive survey of exploiting LLMs for CAD still remains missing.

LLMs have already demonstrated transformative potential in various generative and analytical tasks, including image synthesis [19, 80], text generation [87, 192], and code completion [29, 121]. This makes them a promising candidate for advancing CAD through automation, intelligent design assistance, and semantic understanding of design specifications. Hence, a dedicated review of this emerging research area is both timely and essential. It has the potential to offer invaluable insights for researchers and practitioners seeking to integrate LLMs into CAD workflows.

This article presents the first comprehensive survey of research at the intersection of LLMs and CAD. Our objective is to consolidate recent developments, identify trends, and highlight opportunities for future research in this direction. In addition to surveying LLM applications in CAD, we also extend the scope of prior LLM surveys by incorporating the latest generation of models. Specifically, our survey makes the following key contributions.

1. **Overview of CAD and its industrial significance:** We provide a context on the role of CAD across various industries to onboard researchers and practitioners in the LLM domain to CAD's industrial significance.
2. **Foundations of LLMs:** We review the related foundational concepts of LLMs in an accessible manner to bring CAD experts up to speed with the fundamental knowledge necessary to appreciate the contemporary developments related to LLMs.
3. **Review of state-of-the-art LLMs:** We provide a summary of the key recent developments in LLMs that are relevant to propel CAD developments with LLM augmentations.
4. **Applications of LLMs in CAD:** We systematically categorize and analyze the existing literature on LLMs in CAD-related tasks.
5. **Discussion and future directions:** We provide a detailed discussion on the limitations, challenges, and potential future pathways for integrating LLMs with CAD.

2 CAD in Industry

Computer-Aided Design involves the use of computers or workstations to assist in the creation and modification of designs, replacing traditional pencil drawings with precise digital sketches. CAD enhances designers' productivity, improves design quality, and streamlines communication through clear documentation. Notably, CAD tools can reduce design and prototyping time by 30% to 50%, particularly in iterative design processes [22].

CAD applications span a wide range of sectors, including automotive, aerospace and defense, architecture, manufacturing, electronics, healthcare, consumer goods, energy and utilities, fashion and textiles, entertainment and media, marine and shipbuilding, agriculture and heavy equipment, education and research, jewelry and art[1].

As noted in [3], CAD is especially important in engineering and manufacturing, buildings and construction, medicine and biotechnology, industrial design and consumer products, and media and entertainment. Similarly, [63] identifies its applications in architecture, electronics, shipbuilding, aerospace, textile industry, and education. From these sources, we can observe that CAD is extensively exploited across a variety of practical design related domains.

Taking architectural design as a case in point, CAD is used to model buildings and infrastructure with high precision. Architects and designers can generate detailed 3D models, explore alternative design options, develop construction plans, and collaborate effectively with structural engineers and other project stakeholders [3]. CAD also enables virtual beautification of building designs, helping to reflect realistic expectations visually [63].

The global 3D CAD software market reflects this widespread adoption. This market is projected to grow from USD 13.40 billion in 2025 to USD 24.23 billion by 2034[2]. This growth is largely driven by technological advancements. In 2023, North America led the global CAD software market, while the Asia-Pacific region was forecasted to experience significant growth through 2034. It is noteworthy that the on-premises deployment model accounted for the largest market share in 2023. The large enterprise segment is anticipated to expand significantly between 2025 and 2034. Moreover, the AEC (Architecture, Engineering, and Construction) segment dominated the CAD software market in 2023. These facts present a clear picture of industrial significance of CAD based design.

3 Foundations of LLMs

In this section, we discuss foundational concepts related to LLMs in an accessible manner. Our aim is to introduce the relevant terms and concepts to non-LLM experts to bridge the gap between LLM and CAD research communities. This section is not aimed at discussing the details of these concepts. Readers interested in technical details are referred to [120, 212].

3.1 Background for LLMs

3.1.1 Tokens and Context Length. In the context of LLMs, tokens are the basic units of text that the model reads, processes, and generates. Tokens are pieces of text, like words, subwords, word pieces, or characters depending on the tokenizer.

The context window, also known as context length, refers to the maximum number of tokens an LLM can see or process at once when generating or understanding text.

3.1.2 Decoder-only Transformer. The decoder-only Transformer architecture [174] is one of the most widely used structures for building LLMs. Its core structure consists of a stack of Transformer blocks [174], where each block includes

[1]https://www.linkedin.com/pulse/cad-industry-past-present-future-iseeq-ndppf/
[2]https://www.towardspackaging.com/insights/3d-cad-software-market-sizing?utm_source=chatgpt.com

two sub-layers: one for self-attention modeling and another for feed-forward network modeling. A softmax layer is placed on top of the final Transformer block to generate a probability distribution over the vocabulary. Self-attention, feed-forward networks (FFNs) and softmax layer are all standard components of contemporary neural networks. Although implementation details may vary, many LLMs share this general architecture. These models are referred to as "large" due to their significant width and depth [193].

3.1.3 Training LLMs. Training LLMs involves the standard neural network optimization process, typically using gradient descent algorithms [81]. Studies have shown that model performance improves as they are trained on larger datasets, and when they are scaled up in terms of architecture and computational capacity [70]. This insight has led to continued efforts to increase both the size of training data and model complexity, resulting in increasingly powerful contemporary LLMs.

3.1.4 Fine-tuning LLMs. LLMs are first pre-trained on a large corpus of textual data. This pre-training achieves general-purpose language understanding and generation capabilities. Commonly, this is followed by fine-tuning these models to solve specific natural language processing (NLP) tasks. This fine-tuning involves further training of the model on a limited data, which is available for the downstream task. However, fine-tuning only changes the model parameters slightly, thereby making the process of adaptation computationally less expensive. Since the fine-tuning process may also use instruction-following data, such fine-tuning is also known as instruction fine-tuning.

To enable instruction-following, datasets containing various instructions and corresponding responses are required. Scaling the number of such tasks for fine-tuning generally improves model performance [32]. Unlike pre-training, which may require billions or trillions of samples, fine-tuning can be performed with tens or hundreds of thousands of high-quality samples [25]. Fine-tuning plays a central role in enabling and enhancing this versatility, and ongoing research continues to improve fine-tuning techniques to make LLMs more efficient and effective.

3.1.5 Aligning LLMs. Alignment refers to the process of guiding LLMs to behave in accordance with human intentions and ethical standards. This often involves incorporating human feedback, labeled data, or explicitly defined preferences into the training process. Alignment is essential to ensure responsible and safe artificial intelligence (AI) behavior.

Typically, alignment follows two main steps after initial pre-training:

1. Supervised Fine-tuning (SFT): This step usually uses instruction-based data to further refine the model.
2. Reinforcement Learning from Human Feedback (RLHF) [128]: In this phase, alignment is treated as a reinforcement learning (RL) problem [151]. A reward model—representing the environment—evaluates outputs based on human feedback, while the LLM acts as the agent being optimized to maximize these rewards.

These techniques are critical to adapting LLMs for real-world applications, especially where safety, ethical behavior, and user alignment are essential.

3.1.6 Prompting LLMs. Prompting plays a key role in effective utilization of LLMs, as it requires no additional training or fine-tuning. LLMs are highly versatile once they are pre-trained on large-scale datasets, and careful prompting can strongly exploit this versatility. Instead of building task-specific systems, users can simply provide well-crafted prompts to perform a wide range of tasks. Consequently, prompt engineering [132] has become an active area of research within the NLP community. Owing to the benefits of prompting, LLMs are also known for their zero-shot learning—the ability to perform tasks they were not explicitly trained on.

Another relevant concept in this regard is in-context learning (ICL). In ICL, a well-trained LLM is provided an example of the input-output mapping in the prompt itself during the model inference stage. The model tries to understand the semantics behind this mapping and generalize it to the query in the prompt. The benefit of ICL is that it does not require further training or fine-tuning of the model on the new input-output mapping.

As well-known, LLMs have still been reported to face challenges in tasks that require arithmetic reasoning [201] and commonsense reasoning [64]. This shortcoming can be helped with ICL. Nevertheless, the 'reasoning' requirements of such tasks call for more sophisticated solutions. Hence, to incorporate reasoning capabilities, researchers use Chain-of-Thought (CoT) prompting, which encourages models to solve problems by breaking them down into a series of intermediate reasoning steps. This mimics human-like cognitive processes and has shown notable improvements, particularly in complex mathematical reasoning tasks [176].

There are three common forms of CoT prompting used in ICL:

- Zero-shot CoT prompting, where the model is prompted without examples but encouraged to generate intermediate reasoning steps.
- One-shot CoT prompting, where one example is provided in the prompt.
- Few-shot CoT prompting, where a small number of examples are included.

These prompting strategies enable LLMs to generalize better and solve more complex problems without modifying the model's internal parameters.

3.2 Training at Scale

3.2.1 Data Source. Data is a foundational element in training LLMs, and its importance in learning effective models cannot be overstated. While increasing the quantity of training data is essential, more data does not necessarily equate to better model performance. Several challenges must be addressed when sourcing the data:

- Data Quality: Raw data collected from diverse sources is often noisy or irrelevant. Therefore, filtering and cleaning processes are critical during data preparation. As demonstrated in [131], up to 90% of web-scraped data may need to be removed before training the model due to its adverse effect on the model.
- Data diversity: A robust dataset must encompass a wide variety of domains and languages to ensure generalization and reduce bias. Data sourced from a single or similar corpus can compromise the versatility of the model.
- Bias: Bias can arise from class imbalance or insufficient representation across languages and topics in the dataset. To mitigate bias, datasets are often needed to be explicitly balanced and diversified.
- Privacy concerns: When utilizing large-scale datasets, protecting sensitive information is a key concern. LLMs can be fine-tuned to detect and refuse prompts that might lead to data leakage [191].

Pre-training corpora to train LLMs at scale are typically divided into two categories: general data and specialized data [54, 212]. We summarize the common sources of these categories below.

- **General Data:**
  1. Webpages: The Internet offers a massive corpus of both high- and low-quality text. However, filtering is vital to remove spam, misinformation, or irrelevant content while preserving valuable sources.
  2. Conversational Text: Public dialogue datasets [14, 142] and social media platforms provide conversational data useful for training models in dialogue and response generation.
  3. Books: Long-form text from books supports the learning of linguistic richness, narrative flow, and long-term dependencies. Datasets like Books3 and BookCorpus2 (from the Pile [44]) are commonly used.

- **Specialized Data:**
  1. Multilingual Text: They enhance the multilingual capabilities of LLMs.
  2. Scientific Texts: They strengthen domain knowledge in technical or academic contexts, often sourced from arXiv, scientific textbooks, or math websites.
  3. Code: Programming data improves the model's ability to generate structured code and solve logic-based tasks. Common sources include Stack Exchange [196] and GitHub. Interestingly, formulating reasoning problems as code can also improve the accuracy of generated responses [105].

3.2.2 Model Considerations. Developing LLMs at scale often involves careful considerations related to the underlying neural architecture of the models. Key considerations in this regard include:

- Layer normalization and residual connections [194]: Most LLMs need to apply layer normalization inside residual blocks to enhance stability and trainability of their deep Transformer-based architectures, referred as pre-layer normalization (Pre-LN). Post-layer normalization (Post-LN) can achieve better performance in shallower models, but it may lead to training instability in deeper architectures due to issues like vanishing gradients [194]. Recent research has explored hybrid strategies to combine the benefits of both Pre-LN and Post-LN [75].
- Activation functions: Activation functions are responsible for incorporating non-linearity in the modeling process of neural networks. The choice of activation function is crucial, particularly for FFNs. While ReLU [48] is the prevailing standard, other options have also proven effective in the context of LLMs, e.g., Gaussian Error Linear Unit (GeLU) [55], used in GPT-3 [20] and BLOOM [83], Gated Linear Unit (GLU) [35], used in Gemma [165], and SwiGLU [153], adopted in models like PaLM [31] and LLaMA [169].
- Bias removal: Inclusion/removal of bias terms is another relevant design choice for LLM architectures. Some models, such as LLaMA [169] and Gemma [165], eliminate bias terms in components like layer normalization, FFNs and query-key-value transformations to improve training dynamics.

3.2.3 Distributed Training. Training LLMs at scale demands significant computational resources, typically provided by distributed systems. To that end, various forms of parallelism are employed, including; data parallelism, model parallelism, tensor parallelism and pipeline parallelism. For instance, LLaMA-3 (405 billion (B) parameters) [49] was trained on up to 16,000 H100 GPUs, with each server containing 8 GPUs. Communication across servers was managed using distributed parallelism.

3.2.4 Scaling Laws. Scaling laws [70] describe the empirical relationship between LLM performance and key training variables such as model size, dataset size, and computational budget. Research shows that performance consistently improves with increased data—even having the scale of trillions of tokens, and larger models. These scaling laws serve as a blueprint for LLM development, helping researchers make informed decisions about how to allocate resources effectively.

In summary, LLMs are advanced AI systems that can simulate human-like intelligence through large-scale learning [38]. By leveraging deep learning architectures and massive corpora of data, they learn complex patterns in text, enabling them to generate coherent, contextually appropriate responses and content [56, 88].

4 Mainstream LLMs

LLMs are a type of Transformer-based PLMs with tens to hundreds of billions of parameters. In this section, we categorize LLMs into closed-source and publicly available models. Within each category, we further group models by their families and present them according to their development timeline.

4.1 Closed Source LLMs

4.1.1 The GPT Family (OpenAI). The Generative Pre-trained Transformer (GPT) family consists of decoder-only Transformer models developed by OpenAI. This family of LMMs started with GPT-1 [138] and GPT-2 [139], and moved on to more advanced models, discussed below.

GPT-3 [20]: Often regarded as the first true LLM, GPT-3 is an autoregressive model with 175 billion parameters. It shares the same architecture as GPT-2 but introduces sparse attention and adopts gradient noise scaling [111] for better training. GPT-3 is significantly larger than earlier PLMs and exhibits emergent capabilities not seen in smaller models.

CODEX [26]: A descendant of GPT-3, Codex is fine-tuned on code corpora from GitHub and is optimized for generating code from natural language prompts.

WebGPT [119]: Another GPT-3 variant, WebGPT is trained to answer open-ended questions by browsing web pages, enhancing its ability to perform web-based information retrieval.

InstructGPT [128]: This version aligns the model with human intent by fine-tuning it using datasets containing human-written demonstrations and preferences. It shows improved truthfulness and reduced toxicity.

GPT-3.5: Built on GPT-3 and Codex, GPT-3.5 benefits from fine-tuning on code and improved instruction-following as well as RLHF. GPT-3.5 Turbo [15] is a faster, more cost-effective version optimized for chat.

ChatGPT [147]: A major milestone in LLMs, ChatGPT was fine-tuned with human preference data sourced from a wide range of texts (e.g., Wikipedia, books, websites, scientific papers, articles and news media [184]). It excels at delivering engaging, natural conversations [175] and completing diverse tasks such as translation, summarization, and Q&A [149].

GPT-4 [2]: GPT-4 is a multimodal model capable of processing both text and images. It delivers significantly improved performance on complex tasks and achieves human-level results on professional benchmarks. GPT-4 uses "predictable scaling" to optimize training efficiency, which improves performance in a measurable manner with respect to the required training resources. GPT-4 Turbo [150] is an enhanced version of GPT-4 with better capabilities, larger context length, and integrated tools such as vision, DALL·E 3, and text-to-speech.

GPT-4V [125]: GPT-4V focuses on safely deploying the visual capabilities of GPT-4, offering strong performance in a variety of vision tasks.

GPT-4o [62]: GPT-4o is an "omni" model that handles any combination of text, audio, image, and video as input and output. It brings powerful multimodal capabilities with an emphasis on audio.

OpenAI o1 and o3-mini [65, 127]:

- o1 uses a novel optimization technique and a tailored dataset, supporting visual reasoning.
- o3-mini is a lightweight model optimized for reasoning and cost-efficiency, though it does not support vision.

GPT-4.5 [126]: Building on the advancements of GPT-4o, GPT-4.5 offers a more natural interaction experience. It features a broader knowledge base, a stronger alignment with user intent, and improved emotional intelligence, allowing it to better understand and respond to emotional cues. Additionally, GPT-4.5 reduces the occurrence of hallucinations, ensuring more accurate and reliable responses.

4.1.2 The PaLM Family (Google). The PaLM (Pathways Language Model) series is developed by Google AI.

PaLM [31]: It is a 540B-parameter model trained on 780B tokens. It introduces several architectural enhancements like SwiGLU activation, multi-query attention, and shared input-output embeddings. PaLM shows state-of-the-art few-shot performance on a wide range of tasks.

PaLM-2 [7]: PaLM-2 is a smaller (340B-parameter) but more efficient model, pre-trained on 3.6 trillion (T) tokens. It delivers superior reasoning and multilingual capabilities while reducing training and inference costs.

Med-PaLM / Med-PaLM 2 [155, 156]: Med-PaLM is a domain-specific model fine-tuned for medical tasks. Med-PaLM 2 outperforms its predecessor using domain adaptation and ensemble techniques, achieving strong performance across clinical benchmarks and professional exams.

PaLM-E [40]: It is a multimodal version with 562B parameters (540B from PaLM and 22B from a vision Transformer [36]). It supports visual question answering, zero-shot multimodal CoT reasoning, few-shot prompting, OCR-free math reasoning, and multi-image reasoning.

U-PaLM [162]: U-PaLM improves model quality with minimal additional compute, achieving similar or better performance than PaLM at about half the computational budget.

Flan-PaLM / Flan-U-PaLM [32]: These versions focus on instruction tuning and CoT prompting. Flan-PaLM, with 540B parameters fine-tuned on 1.8K tasks, significantly outperforms the base PaLM model. Flan-U-PaLM further enhances the performance with advanced adaptation techniques like UL2R [162].

4.1.3 The Gemini Series (Google). Gemini is a new family of multimodal LLMs, succeeding LaMDA [168] and PaLM-2.

Gemini [163]: Gemini demonstrates strong capabilities across text, image, video, and audio understanding, with support for a 32K context length.

Gemini 1.5 [164]: Gemini 1.5 brings notable advancements, including a new model architecture, the use of mixture-of-experts (MoE), and training on a much larger dataset containing millions of tokens.

Gemini 2.0 [134]: Gemini 2.0 is a major update to its predecessor, improving both speed and performance compared to Gemini 1.5.

Gemini 2.5 [71]: Gemini 2.5 further enhances reasoning and coding capabilities, incorporating techniques like CoT prompting for more structured reasoning.

However, Gemma [165] and Gemma 2 [166] are lightweight, open-source versions of Gemini, providing powerful multimodal capabilities for free.

We summarize state-of-the-art closed-source LLMs in Table 1.

Since ChatGPT, released in November 2022, was the first LLM to demonstrate exceptional performance across a variety of tasks, we focus on models released from 2023 onward. Before ChatGPT, LLMs were generally less powerful, but following its release, models have grown powerful enough to compete with ChatGPT, driven by exponential growth in both model size and data. It is worth noticing that we only list representative LLMs in the table. For example, we list Gemini but omit Gemini 1.5 due to limited technical innovations. However, since PaLM and PaLM-2 represent distinct models, both are included.

4.2 Publicly Available LLMs

4.2.1 The LLaMA Family (Meta). The LLaMA family consists of a series of foundational language models developed by Meta.

Table 1. Summary of state-of-the-art closed source LLMs.

Model	#Param.	Tokens	Year	Entity	Key Feature
ChatGPT [147]	-	-	2022	OpenAI	Dialogue-optimized LLM via RLHF
GPT-4 [2]	-	-	2023	OpenAI	Multimodal support, stronger reasoning
GPT-4V [125]	-	-	2023	OpenAI	Vision-enabled GPT-4 variant
GPT-4o [62]	200B	-	2024	OpenAI	Omni-modal (text, image, audio, video)
OpenAI o3-mini [127]	3B	-	2025	OpenAI	Lightweight GPT-4 variant for mobile use
GPT-4.5 [126]	-	-	2025	OpenAI	Improved emotional intelligence
PaLM [31]	540B	780B	2023	Google	Dense decoder-only model
PaLM-2 [7]	340B	3.6T	2023	Google	Fine-tuned for multilingual reasoning
Med-PaLM [155]	540B	-	2023	Google	MedQA [68] benchmarked medical LLM
PaLM-E [40]	562B	-	2023	Google	Embodied multimodal reasoning
U-PaLM [162]	540B	-	2023	Google	Unified fine-tuning with few-shot and CoT
Flan-PaLM [32]	540B	-	2024	Google	Instruction-tuned PaLM variant
Med-PaLM 2 [156]	-	-	2025	Google	Safety-aligned medical LLM
Gemini [163]	-	-	2023	DeepMind	Multimodal, integrated with search
PanGu-Σ [141]	1085B	329B	2023	Huawei	China's largest LLM, multilingual
BloombergGPT [188]	50B	708B	2023	Bloomberg	Financial-domain LLM
PPLX[a]	70B	-	2023	Perplexity AI	Retrieval-augmented, mobile-ready LLM
Inflection-2.5[b]	2.5B	-	2024	Inflection AI	Personal assistant, dialogue-tuned
Claude 3 [10]	-	-	2024	Anthropic	Constitutional AI, safety-tuned
Grok 3[c]	-	-	2025	xAI	Twitter-integrated, humor-oriented chat

[a] https://grok.com/
[b] https://www.perplexity.ai/
[c] https://inflection.ai/blog/inflection-2-5

LLaMA [169]: The first model in the LLaMA family, LLaMA, ranges from 7B to 65B parameters and is pre-trained on trillions of tokens. It is built upon the Transformer architecture, similar to GPT-3 [18], with several architectural modifications, including:

- Adoption of the SwiGLU activation function instead of ReLU.
- Replacement of absolute positional embeddings with rotary positional embeddings.
- Use of root mean squared layer normalization rather than standard layer normalization.

LLaMA-2 [170]: LLaMA-2 includes both foundational models and chat-optimized versions, such as LLaMA-2 Chat. While the architecture remains largely the same as LLaMA, LLaMA-2 was trained on 40% more data. LLaMA-2 Chat further improves safety by fine-tuning on safe response samples and incorporating an additional RLHF step.

LLaMA-3 / 3.1 [49]: LLaMA-3 / 3.1 models are trained on a dataset seven times larger than that of LLaMA-2. LLaMA-3 comes in two sizes: 8B and 70B. With the largest model featuring 405B parameters, LLaMA-3.1 benefits from 15T tokens of training data and a significantly larger context window (128K compared to LLaMA-2's 8K). These improvements lead to its competitive performance, achieving results comparable to GPT-4 and GPT-4o.

LLaMA-4 [114]: LLaMA-4 introduces significant advancements by supporting text, images, audio, and video, enabling seamless multimodal reasoning and generation. This is a major shift from its predecessors, LLaMA-2 and LLaMA-3, which were primarily text-based. Leveraging a Mixture of Experts (MoE) architecture, LLaMA-4 efficiently scales to trillions of parameters, activating only a subset of experts during inference for optimal performance. Additionally, it supports an impressive context length of up to 10 million tokens, making it ideal for long-form tasks like multi-document summarization and complex dialogues.

In addition to the core LLaMA models, a variety of instruction-following models have been developed based on LLaMA or LLaMA-2, including: Alpaca [161], Vicuna [30], Guanaco [39], Koala [46], Mistral 7B [66], Code LLaMA [143], Gorilla [130], Giraffe [129], Vigogne [60], Tulu 65B [180], Long LLaMA [172], Stable Beluga 2 [106], Qwen [13], among others. Vicuna is particularly popular for its multimodal language modeling, giving rise to models like LLaVA [102], MiniGPT-4 [216], InstructBLIP [34], and PandaGPT [157].

4.2.2 The DeepSeek Family (DeepSeek). The DeepSeek series consist of LLMs developed by the Chinese AI company DeepSeek.

DeepSeek-Coder [52]: DeepSeek-Coder is the first model in the DeepSeek family, followed by DeepSeek-LLM [17], DeepSeek-MoE [33], and DeepSeek-Math [152]. DeepSeek-LLM investigates the scaling laws for LLMs to identify the optimal model size and training data scale. DeepSeek-MoE is an innovative MoE architecture specially designed towards ultimate expert specialization. DeepSeek-Math is a domain-specific language model that significantly outperforms the mathematical capabilities of open-source models and approaches the performance level of GPT-4 on academic benchmarks.

DeepSeek-VL [104]: DeepSeek-VL is an open-source Vision-Language Model designed for real-world applications that combine vision and language understanding.

DeepSeek-V2 [98]: DeepSeek-V2 uses multi-head latent attention to reduce inference costs by compressing the key-value cache into a latent vector. This method achieves 5.76 times faster inference throughput compared to the previous DeepSeek models.

DeepSeek-V3 [99]: DeepSeek-V3 is a stronger MoE model, featuring 671B parameters, with 37B activated per token. It is pre-trained on 14.8T high-quality, diverse tokens and undergoes SFT and RL stages to fully optimize its capabilities.

DeepSeek-R1 [51]: DeepSeek-R1-Zero is pre-trained with large-scale RL without the initial SFT stage. DeepSeek-R1 follows a multi-stage training process, including cold-start data before RL, and incorporating two RL stages to improve reasoning patterns and align with human preferences, along with two SFT stages to enhance reasoning and non-reasoning abilities.

We present an overview of state-of-the-art publicly available LLMs in Table 2, focusing on listing only the most representative LLMs of each type.

5 Applications of LLMs in CAD

This section presents the existing literature focusing on applications of LLMs in CAD. Due to the nascency of this emerging but critical research area, we are able to provide a comprehensive overview of the recent developments. These advances are organized by grouping the works according to their key application directions in the CAD domain.

5.1 Data Generation

LLMs have emerged as powerful tools for content generation, making data generation one of their primary applications in the field of CAD. This capability is particularly useful for creating synthetic datasets, fine-tuning models, and addressing the lack of annotated CAD data.

Yuan et al. [208] utilized GPT-4o [62] to generate datasets that map natural language descriptions and rendered view images to CAD operation sequences. This approach enabled the construction of textual descriptions corresponding

Table 2. Summary of state-of-the-art publicly available LLMs.

Model	#Params.	Tokens	Year	Entity	Key Feature
LLaMA [169]	65B	1.4T	2023	Meta	Foundation model for instruction tuning
LLaMA-2 [170]	70B	2T	2023	Meta	Improved safety, performance, training data
Code LLaMA [143]	70B	1T	2023	Meta	Code-specific extension of LLaMA
LIMA [215]	65B	-	2023	Meta	Fine-tuned with only 1K examples
Long LLaMA [172]	7B	1T	2023	Meta	Long-context with Focused Transformer [172]
LLaMA-3 [49]	70B	15T	2024	Meta	Supporting longer context, enhanced reasoning
LLaMA-4 [114]	109B	40T	2025	Meta	10 million context length, multimodal + MoE
DeepSeek-Coder [52]	33B	2T	2024	DeepSeek	Code synthesis and reasoning
DeepSeek-LLM [17]	67B	2T	2024	DeepSeek	General-purpose pre-trained
DeepSeek-MoE [33]	16B	2T	2024	DeepSeek	Sparse MoE
DeepSeek-Math [152]	7B	120B	2024	DeepSeek	Math benchmark specialist
DeepSeek-VL [104]	7B	2T	2024	DeepSeek	Vision-language aligned
DeepSeek-V2 [98]	236B	8.1T	2024	DeepSeek	High-performance general model
DeepSeek-V3 [99]	671B	14.8T	2024	DeepSeek	Flagship MoE LLM
DeepSeek-R1 [51]	70B	-	2025	DeepSeek	Reinforced instruction tuning
Stable Beluga 2 [106]	70B	-	2023	Stability AI	Instruction-tuned LLaMA-2
Stable LM 2 [16]	1.6B	2T	2024	Stability AI	Lightweight model
Giraffe [129]	13B	-	2023	Salesforce	Visual grounding and reasoning
InstructBLIP [34]	13B	-	2023	Salesforce	General-purpose vision-language model
CodeGen2 [123]	16B	400B	2023	Salesforce	Multi-language code generation
Koala [46]	13B	-	2023	Berkeley	Dialogue-tuned LLaMA on curated data
Gorilla [130]	7B	-	2024	Berkeley	API call-focused LLM
Orca [117]	13B	-	2023	Microsoft	Imitation learning from GPT-4
WizardLM [195]	13B	-	2024	Microsoft	Evol-Instruct fine-tuned
Alpaca [161]	7B	-	2023	Stanford	Instruction-tuned LLaMA on self-instruct
Vicuna [30]	13B	-	2023	LMSYS	Chatbot fine-tuned on ShareGPT [9]
Guanaco [39]	65B	-	2023	UW	QLoRA [39] fine-tuned Vicuna
Mistral [66]	7B	-	2023	Mistral AI	Sliding window attention, strong performance
Vigogne [60]	7B	-	2023	-	French fine-tuned Vicuna
Tulu [180]	65B	1.4T	2023	Allen AI	Instruction-tuned LLaMA
Qwen [13]	14B	3T	2023	Alibaba	Multilingual and code generation
LLaVA [102]	13B	-	2023	UIUC	Vision-language instruction tuning
PandaGPT [157]	13B	-	2023	-	Multimodal instruction-following
Pythia [18]	12B	300B	2023	EleutherAI	Transparent training for scientific benchmark
Baichuan2 [199]	13B	2.6T	2023	Baichuan	Strong multilingual and reasoning performance
FLM [93]	101B	311B	2023	-	Hybrid MoE + dense
Skywork [183]	13B	3.2T	2023	-	Multilingual, instruction-tuned
Falcon [131]	7.5B	600B	2023	TII	RefinedWeb dataset
Zephyr [171]	7.24B	800B	2023	HuggingFace	Aligned LLaMA-2 variant
StartCoder [89]	15.5	1T	2023	-	Code generation LLM
MPT [167]	7B	1T	2023	MosaicML	Commercially permissive LLM
XuanYuan 2.0 [210]	176B	-	2023	-	Financial sector
CodeT5+ [182]	16B	51.5B	2023	-	Code understanding + generation
BloomZ and mT0 [116]	176B	350B	2023	BigScience	Cross-lingual capabilities
Jamba [95]	52B	-	2024	AI21 Labs	Hybrid Transformer + MoE
Gemma [165]	7B	6T	2024	Google	Lightweight model with strong benchmarks
MiniGPT-4 [216]	13B	-	2024	-	Image-to-text multimodal capabilities
ChatGLM [47]	9B	10T	2024	Tsinghua	Bilingual chat, long-context
Llemma [11]	34B	50B	2024	-	Math and science focused
Command R+[d]	104B	-	2024	Cohere	RAG-enhanced instruction model for finance

[d] https://cohere.com/blog/command-r-plus-microsoft-azure

to CAD commands. Similarly, Xu et al. [197] employed the open-source model InternVL2-26B [28], by randomly selecting four view images of CAD models to generate high-quality textual captions. These captions were used to build a multimodal CAD dataset. Khan et al. [72] adopted a two-step generation process using open-source LLMs and Vision-Language Models (VLMs). First, they rendered multi-view images of individual parts and the final models, which were input to LLaVA-NeXT model [101] with predefined prompts to generate simplified object-level descriptions. Then, they used Mixtral-50B [67] to produce multi-level natural language instructions. Raw CAD sequences from DeepCAD [187] were preprocessed to replace meaningless keys with more descriptive terms, which, along with the simplified shape descriptions, served as input to Mixtral-50B. The result was a four-level hierarchy of textual instructions: abstract, simplified, generalized geometric, and detailed geometric descriptions, generated using k-shot prompting techniques [20].

In another recent work, Wang et al. [177] used a VLM to generate initial draft captions for rendered CAD images, which were then refined through human annotations to create a dataset, pairing these captions with CAD parametric sequences. In [178], Wang et al. employed GPT-4o [62] to classify and filter CAD models, and then used InstructGPT [181] to generate natural language descriptions. The resulting dataset contained CAD modeling sequences aligned with textual descriptions and rendered images from fixed viewpoints.

Other examples of using LLM for CAD data generation include [43], which developed BlendNet, a custom training dataset designed to capture diverse communication styles across three axes: 16 object categories, 8 instruction tones, and 5 length categories. Starting with 135 manually crafted seed instructions, the authors used self-instruct data distillation to expand to 50,000 samples. GPT-4o [62] was used to generate scripts from these instructions, which were executed to produce corresponding images. GPT-4o then served as a validator to assess the alignment between images and instructions, resulting in a final dataset of 2,000 human-verified and 6,000 model-validated samples. Similarly, Li et al. [90] also addressed the challenge of unavailable textual descriptions in existing CAD datasets. They applied the multimodal model CoCa [205] to generate captions for parametric CAD models in the DeepCAD dataset [187], creating a new, richly annotated dataset linking text to CAD models.

It is worth noting that current research in data generation primarily leverages the text generation capabilities of LLMs. To date, in the CAD field, there is no work that directly generates other forms of LLM-based data—such as images, point clouds, or other format of data—to construct new datasets in the CAD domain.

5.2 CAD Code Generation

Many LLMs, such as Codex [26], are pre-trained on vast code corpora, enabling them to generate code with a high degree of accuracy. Thus, a prominent application of LLMs in the CAD field is the generation of CAD code.

In this direction, Li et al. [91, 92] explored the use of multimodal LLMs for 3D CAD generation, leveraging various data formats, including textual descriptions, images, sketches, and ground truth 3D shapes. In this work, LLMs were employed to generate CAD programs from these inputs, which were subsequently parsed into 3D shapes. Two models were evaluated: GPT-4 [2] and GPT-4V [125]. To improve the quality of the generated CAD code, the authors developed a debugger that iteratively refined the generated code until it successfully executed. The results showed that GPT-4V outperformed GPT-4, especially when using text-only input. It also surpassed other input combinations (e.g., text + sketch, text + image, and text + sketch + image) in terms of parsing rate and intersection over union.

Sun et al. [158] recently conducted a fine-tuning experiment on LLMs, comparing them to the baseline GPT-4 [2] without fine-tuning. They used the dataset from [91] and implemented four different sampling strategies for fine-tuning data generation. The first strategy randomly selected 60 code templates, the second ensured diversity in code length, the third focused on the shortest lengths, and the fourth on the longest. Overall, the second strategy achieved the highest average parsing rate, while the third performed best in terms of intersection over union. All four fine-tuned LLMs outperformed the baseline GPT-4 in most cases. In [43], Du et al. developed BlenderLLM, a script generation model that underwent SFT and iterative self-improvement. They fine-tuned the Qwen2.5-Coder-7B-Instruct model [61] using the BlendNet dataset and created a filter to select high-quality data generated by the model. This approach allowed for iterative optimization through cycles of data generation and model training. On similar lines, Alrashedy et al. [4] introduced CADCodeVerify, a framework designed to iteratively verify and refine 3D objects generated from CAD code. CADCodeVerify employs a VLM to assess the generated object by answering a set of questions and correcting deviations. They also introduced a benchmark for CAD code generation, containing 200 prompts paired with expert-annotated scripting code. The framework's feedback loop involved two steps: generating question-answer pairs and producing feedback without human intervention. The study compared the performance of three LLMs: GPT-4 [2], Gemini [163], and Code LLaMA [143].

In [12], Badagabettu et al. presented Query2CAD, a framework that generated CAD designs by using LLMs to produce executable CAD macros. The process involved the user query being fed into a robust LLM, such as GPT-3.5 Turbo or GPT-4 Turbo, which generated a Python macro. In cases where errors occurred, the error messages and Python code were provided to the LLM, which then generated corrected, executable code. The framework also utilized BLIP2 [86] to generate captions for isometric views of CAD models, which were then passed onto the LLM to improve code generation through self-refinement loops. Yuan et al. [208] fine-tuned pre-trained LLMs to create their own LLM, called OpenECAD, which integrated the logical, visual, and coding abilities of VLMs. They rendered Boundary Representation (B-rep) data to generate 3D model images from three different views: default, orthographic, and transparent. For the base language models, they selected relatively small models like OpenELM [112], Gemma [165], and Phi-2 [1]. To endow OpenECAD with multimodal conversational capabilities, they leveraged training methods and datasets from LLaVA [102] and used GPT-4 [2] to generate multimodal image-language instruction data. The model was trained using the TinyLLaVA [214] framework and fine-tuned with the LoRA [58] method to enable the VLM to generate CAD code effectively.

In addition, several studies have explored the potential of LLMs across a variety of design and manufacturing tasks. For instance, Makatura et al. [107, 108] evaluated the performance of LLMs in diverse design domains, including 2D vector graphic design (using SVG and DXF formats), 3D parametric modeling (using Constructive Solid Geometry (CSG) and B-rep formats), and articulated robotics problems (using the Universal Robot Description Format (URDF) and general graph-based representations). Except for URDF, the team initially used GPT-4 [2] to generate code, which was then converted into the corresponding 3D models in the desired format. In [122], GPT-4 [2] was used to generate CAD code, employing a back-and-forth dialogue process. Errors in the generated code and resulting structure were fed back into GPT-4, allowing it to iteratively refine the code. The study demonstrated GPT-4's remarkable ability to generate unique solutions for constructing and debugging CAD code.

Mallis et al. [110] utilized GPT-4o [62] to generate CAD code actions expressed in Python. The generated code was executed in CAD software, and the output was concatenated with the input context before being fed back to GPT-4o for the next iterative step. Rukhovich et al. [144] proposed CAD-Recode, a solution for CAD reverse engineering. They fine-tuned an LLM to map input point clouds into CAD sketch-extrude sequences represented as Python code. The LLM used was Qwen2-1.5B [200], a relatively small model. CAD-Recode was augmented with a lightweight, trainable point cloud projector. The framework also utilized GPT-4o [62] to refactor the generated code in real time via interactive sliders.

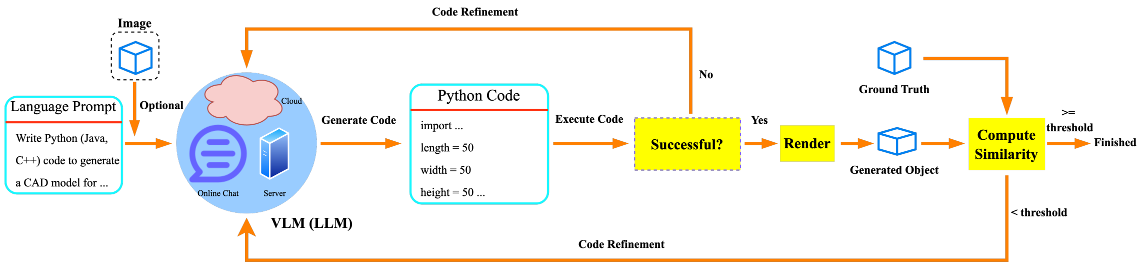

Fig. 2. Typical CAD code generation pipeline. A prompt and, optionally, an image are first input to a VLM or LLM. The VLM/LLM then generates the corresponding code, which is executed. If the code is not executable, refinement is performed until it is. Subsequently, the generated objects' computing similarity to the ground truth is evaluated. If the similarity is below a threshold, the code refinement process is repeated until the desired results are achieved.

Jones et al. [69] worked on generative CAD design with GPT-4o [62] (without fine-tuning) and a solver-aided, domain-specific language: AI design language (AIDL). The process involved prompting GPT-4o with detailed descriptions of AIDL, including its syntax and available geometry types. The LLM then generated a complete AIDL program prompted by manually designed example programs, which was executed to create the desired model. If errors occurred during execution, they were fed back to GPT-4o, which would correct them. Naik [118] demonstrated the use of LLMs for creating extraction queries in SQL for geometry tagging, streamlining the tool's functionality by allowing users to provide natural language input instead of complex code. Kienle et al. [74] designed QueryCAD, a deep learning-based question-answering system for CAD models. QueryCAD employed a code-writing LLM with in-context prompting to generate executable code for multi-view CAD part segmentation. It was integrated into MetaWizard, a robot program synthesis system [5], to automatically generate industrial robot programs based on natural language task specifications. The team primarily used GPT-4o [62] and compared its performance with other LLaMA models (e.g., LLaMA-3 13B, LLaMA-3.1 8B, and LLaMA-3.1 405B) [49].

In [124], Ocker et al. developed a VLM-based multi-agent system for CAD model generation. This system utilized multiple LLM-based agents to interpret sketches, images, and textual descriptions, generate CAD models, and verify the designs. The study proposed strategies for overcoming the well-known spatial reasoning limitations of VLMs, using GPT-4o [62] as the core LLM. In another recent work, Deng et al. [37] applied GPT-4 [2] to address three sub-tasks: parametric computation, instruction sequence construction, and coding model scripting. The process involved providing GPT-4 with design requirements, performing iterative design computations, and generating the necessary code by parsing a JSON file to formalize natural language instructions into executable commands.

CAD code generation remains one of the most crucial applications of LLMs in CAD. These generated code can take various forms, such as Python code, SQL code, CAD software-specific code, and even Java or C++ code. Users can generate the desired code formats based on their specific domain knowledge. Figure 2 illustrates the typical CAD code generation pipeline employed by most state-of-the-art approaches.

5.3 Parametric CAD Generation

Parametric CAD generation focuses on creating parametric data for CAD rather than code. In this application, LLMs generate parametric sequences, often formatted as JSON files.

In the direction of parametric CAD generation, Wu et al. [189, 190] utilized pre-trained LLMs to efficiently manipulate engineering sketches, integrating sketch primitive sequences and images for parametric CAD generation. This included CAD autocompletion, CAD autoconstraint, and image-conditioned generation. They employed CodeT5+ [182] as the text encoder and decoder for sketch primitive sequences. To assess the general LLM's capability in CAD autocompletion, they fine-tuned ChatGPT on three subsets, demonstrating that ChatGPT outperformed other baselines. Similarly, Li et al. [90] explored text-to-3D parametric CAD generative modeling. During training, both parametric CAD sequences and textual descriptions were used as inputs, and the model generated parametric CAD sequences. At the inference stage, only textual descriptions were provided to generate parametric CAD sequences.

In [197], Xu et al. proposed CAD-MLLM, a system designed to generate parametric CAD models conditioned on multimodal inputs such as text, images, and point clouds. For images, they rendered multi-view images from eight fixed perspectives. For point clouds, they randomly sampled points at different ratios and recorded their corresponding normal information. They fine-tuned the open-source LLM Vicuna-7B [30] and used LoRA [58] during training to minimize learnable parameters. You et al. [204] focused on reverse engineering 3D CAD models from images. They first used the GPT-4V [125] foundation model to predict a global discrete base structure, extracting semantic information from images. The model identified semantic parts from the image before generating CAD sequences for each part. They then built a Transformer model to predict continuous attribute values based on the discrete structure with semantics.

Zhang et al. [211] applied LLMs for controllable generation across various CAD construction hierarchies. They initially converted a CAD model into structured text. They then fine-tuned an LLM to develop a unified model for controllable CAD generation. During training, a hierarchy-aware field in the CAD text was masked, and LLMs were tasked with predicting the masked field. During inference, users could specify the part to modify by masking it and inputting it into the LLM for generating new CAD models. Wang et al. [178] proposed a CAD synthesis method using a multimodal LLM with enhanced spatial reasoning capabilities. They input either a single image or text and mapped the 3D space into 1D using a tokenization method to improve spatial reasoning. They employed LLaVA-1.5 7B [100] as their base model, with pre-trained Vicuna [30] as their foundation, which was built on LLaMA-2 [170]. The 3D CAD model was represented in JSON format, capturing key modeling commands and parameters in the order of CAD construction, based on the DeepCAD dataset [187].

Utilizing the LLaMA-3 8B Instruct [49] LLM as the backbone, Wang et al. [177] introduced a CAD-Fusion framework. It alternated between two training stages: the sequential learning stage, which trained LLMs using ground truth parametric sequences, and the visual feedback stage, which rewarded parametric sequences that rendered visually preferred objects and penalized those that did not. Makatura et al. [107, 108] created a design space defined by parametric design and the bounds of these parameters, encompassing a range of potential designs. When prompted with lower and upper bounds for parameters, LLMs suggested values based on typical proportions for the designed object. While the absolute scale was arbitrary, the proposed bounds were semantically reasonable and proportionate. In addition to creating design spaces, they also used GPT-4 [2] to generate XML-format data rather than code, focusing on pre-existing designs in formats like URDF.

Parametric CAD Generation is another key application of LLMs in the CAD field. Unlike CAD code generation, which relies on LLMs to generate executable code, parametric CAD generation uses LLMs to produce parametric sequences instead of code. While generating accurate and executable code can be challenging for state-of-the-art LLMs, parametric CAD generation is often preferred by researchers. Although LLMs still face difficulties in producing perfectly accurate parametric sequences, if the goal is to generate keys and values rather than fully executable sequences, many

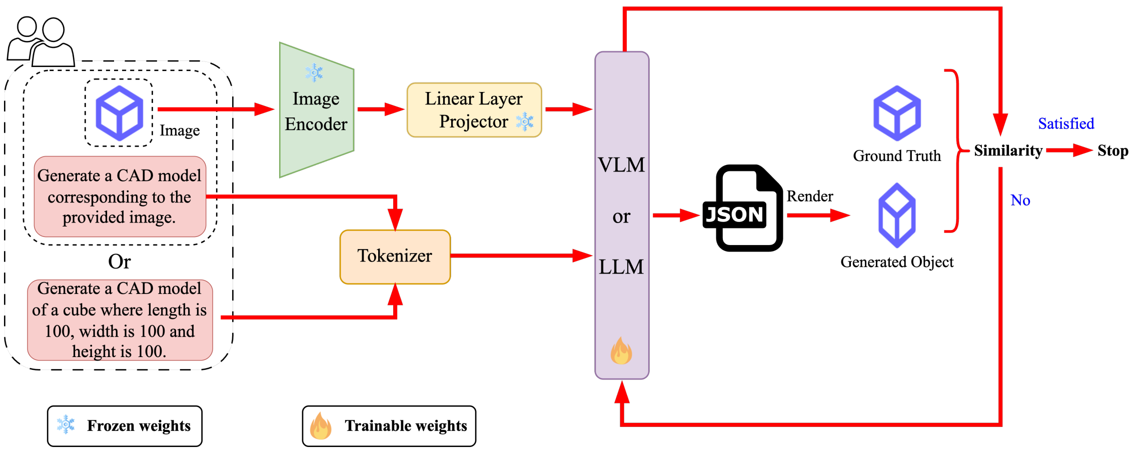

Fig. 3. A standard parametric CAD generation pipeline. A prompt, along with an optional image (converted into features by a frozen image encoder), is fed into a trainable VLM or LLM to generate parametric data. This data is then parsed to produce 3D CAD models. LLMs can also be employed to compute the similarity between the ground truth and the generated object.

CAD challenges can be alleviated. Figure 3 illustrates a standard parametric CAD generation pipeline used by most state-of-the-art methods in this direction.

5.4 Image Generation

Whereas generative technologies are becoming increasingly popular in visual modeling [173], the use of LLMs in image generation tasks related to CAD is still a widely open direction. Recently, Tang et al. [160] employed LangGraph[3], an LLM, to perform digital CAD drawing restoration and utilized Retrieval-Augmented Generation (RAG) [84] technology to incorporate engineering domain knowledge into the model. This is the only work we currently find that generates CAD drawings from an LLM in an end-to-end manner.

5.5 Model Evaluation

LLMs can also be used to compare two sources of data. For instance, ChatGPT can be used to estimate similarity between two textual descriptions—one generated by a model and the other being the ground truth. This ability of LLMs is also making its way into the CAD domain. For instance, Du et al. [43] developed an evaluation framework, CADBench, which utilized GPT-4o [62] for two complementary evaluation tasks: image-based evaluation, which assessed visual fidelity, and script-based evaluation, which compared objective attributes such as size, color, and material. In [12], the visual question-answering score between the user query and the generated isometric image was computed using Clip-FlanT5-XL [96]. Similarly, in [160], Tang et al. fed repaired drawings alongside ground truth drawings into GPT-4 [2] for automatic scoring along three dimensions: legibility, completeness, and tolerance.

Wang et al. [177] leveraged the strong visual understanding capabilities of VLMs to score visual objects, aiding in the construction of preference data, which would otherwise be costly and labor-intensive. The rendered CAD images, along with an instruction detailing the evaluation criteria—divided into three categories: shape quality, shape quantity, and distribution—were passed into the VLM, LLaVA-OV-Qwen2-7B [85], to compute the scores.

[3]https://www.langchain.com/built-with-langgraph

5.6 Text Generation

Text generation is one of the most common applications of LLMs. Unsurprisingly, many CAD-related works have also employed LLMs for generating text content. For example, Liu et al. [103] combined DALL-E [140], GPT-3 [20], and CLIP [137] within CAD software, enabling users to construct text and image prompts based on their modeling needs. Designers could use text-to-image AI to generate reference images, avoid design bias, prevent design mindset, and stimulate new design considerations. Users would input their design intentions, and GPT-3 would suggest prompts for them to select from. After choosing, the suggestions were rephrased, and DALL-E would generate the final results.

Kodnongbua et al. [79] employed ChatGPT to generate text prompts describing variations of a given model. Similarly, Yuan et al. [207] introduced a task for semantic commenting of CAD programs. In this task, LLMs segmented programs into code blocks, each corresponding to semantically meaningful parts, and assigned semantic labels to each block. They executed the program to generate a 3D shape and rendered images from ten representative viewpoints. Each image was then translated into a photorealistic image using ControlNet [209]. Using ChatGPT, the authors segmented the images into semantically meaningful parts and transformed every bounding box to a pixel-wise segment utilizing segment anything model [77], linking them to corresponding code blocks. They also explored how ChatGPT could comment on programs based on a given example program, finding that while it performed well at commenting similar programs, it struggled to generalize to new shapes and primitives.

In [204], You et al. employed a VLM to generate semantic part comments for their predicted global base structure. They used GPT-4V [125] to interpret an input image and decompose it into semantic parts. Mallis et al. [110] fed multimodal user requests into a VLM, which then outputted a response in the form of a natural language plan. Along similar lines, Rukhovich et al. [144] utilized GPT-4o [62] for CAD-specific question answering, allowing users to query the system for information related to CAD models. In another recent effort, Yu et al. [206] employed a locally built LLM to analyze users' requirements and generate initial design ideas. The LLM was used to establish structural requirements and guide how to model the scheme. They also employed finite element analysis to improve data and fine-tuned the LLM to generate easily understandable design improvement suggestions.

An investigation of VLMs for automatically recognizing manufacturing features in CAD design was conducted in [73]. The process began by converting a CAD file into three different image views, which, combined with prompts outlining the analysis criteria, served as inputs for five VLMs to identify and evaluate the features. The VLMs included both closed-source models such as GPT-4o [62], Claude-3.5-Sonnet, and Claude-3-Opus [10], as well as open-source models like MiniCPM-Llama3-V2.5 [59] and Llava-v1.6-mistral-7b [101]. A review along conceptually similar lines was also conducted by Sun et al. [159], who analyzed key developments in rule-based reasoning (RBR), case-based reasoning (CBR), and large AI models for advancing reusable design in CAD software, summarizing both their advantages and disadvantages. They proposed a hybrid framework combining RBR, CBR and LLMs along with knowledge graphs to enhance reusable CAD design.

Picard et al. [133] explored the capabilities of GPT-4V [125] and LLaVA 1.6 34B [101] in performing various engineering design tasks, which involved both visual and textual information. These tasks were categorized into four main areas: conceptual design, system-level and detailed design, manufacturing and inspection, and engineering education. Most of these tasks involved generating textual descriptions from images and text prompts. They also assessed GPT-4V's spatial reasoning abilities, concluding that while the model demonstrated some spatial reasoning capability, it was still limited when compared to human performance. In [179], Wang et al. treated 2D drawings as raster images and employed an image encoder to extract features. They utilized general-purpose language scripts to represent 3D parametric models and leveraged an LLM to autoregressively predict parametric sequences in text form. They developed CAD2Program by fine-tuning an open-source VLM, Mini-InternVL-1.5-2B [45], which used InternViT-300M [27] as the vision encoder and InternLM2-1.8B [21] as the language model. Relevant applications of LLMs in manufacturing were reviewed by Li et al. in [94], evaluating LLM use in various tasks through case studies and examples. While this survey also covered areas outside CAD, it discussed CAD-related tasks including data generation, text-grounded 3D content generation, initial design drafts, and idea generation in aerospace design.

Text generation is primarily associated with question answering. While this application may seem simple, it is essential to recognize that it is one of the key strengths of LLMs. By effectively leveraging the text and content generation capabilities of LLMs, we can achieve results that go beyond initial expectations, unlocking significant potential for innovative solutions.

6 Analysis and Discussion

From the aforementioned state-of-the-art works in Section 5, it can be concluded that nearly all approaches utilize LLMs to generate intermediate representations rather than directly outputting 3D CAD models or 2D CAD drawings. This is likely because directly generating accurate 3D or 2D CAD outputs remains a significant challenge for current models. Common intermediate formats include executable code—such as Python scripts, and parametric data—often structured as JSON files, which can then be parsed or executed to construct the final CAD models.

We also make another key observation that most state-of-the-art methods rely on multimodal inputs, making the use of multimodal LLMs increasingly prevalent. This highlights the potential of different input types for LLM-based automation. Among these input types, text and images are used most frequently. However, other data formats—such as point clouds and sketch sequences—are also explored, reflecting the growing diversity of input modalities being considered in the field.

We also record the frequency with which each LLM is used in the state-of-the-art works that we review. Figure 4 summarizes the number of times different LLMs have been employed in the reviewed contributions. As apparent, GPT-4o is the most frequently used model (11 instances), followed by GPT-4 (8 instances) and GPT-4V (4 instances)—all developed by OpenAI. This trend suggests that, despite being closed-source and potentially requiring payment, LLMs from the GPT family are still widely favored by the research community in CAD domain. A likely reason is their superior performance compared to the other types of LLMs, including many publicly available models.

We also analyze the distribution of reviewed works across different application types. Table 3 summarizes the number of studies attributed to each category. As shown, the majority of contributions focuses on CAD code generation (17 works), followed by text generation (12 works). This suggests that CAD code generation is currently the most prominent and attractive application of LLMs in the CAD domain, and it is expected to continue gaining traction among researchers. Text generation, being a core capability of LLMs, also remains a widely explored area within the research community.

We also analyze the datasets used in each state-of-the-art study. Table 4 presents the datasets employed across the reviewed works. Some datasets are sourced from industry data, while others are synthetically generated by the authors. Many of the industrial datasets are derived from publicly available datasets, such as DeepCAD, Onshape[4], ABC [78], and others.

Finally, we map each state-of-the-art work to its industrial context. This allows us to identify current active applications of LLMs for CAD, as well as the domains that are relatively underexplored. Table 5 presents the industries most

[4]http://onshape.com

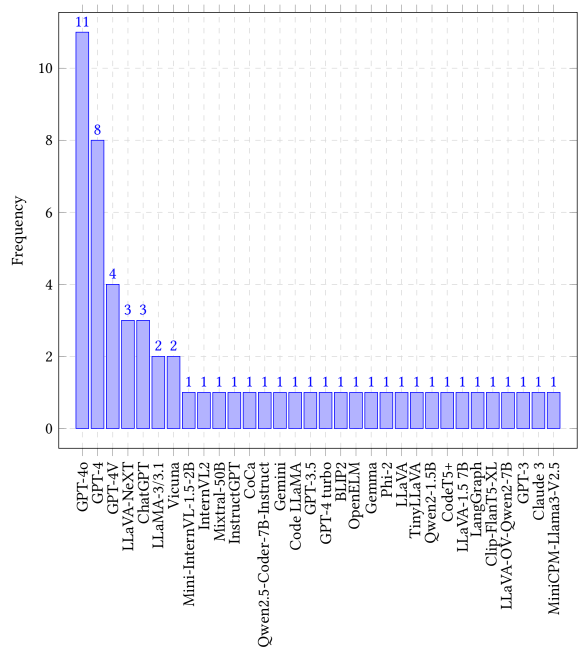

Fig. 4. Bar chart of LLM usage frequency in CAD-related research (sorted descending).

Table 3. Number of works attributed to various LLM applications in CAD.

Application	Works
Data Generation	7
CAD Code Generation	17
Parametric CAD Generation	9
Image Generation	1
Model Evaluation	4
Text Generation	12

commonly mentioned in the reviewed works. As shown in the table, the manufacturing industry currently attracts the most attention of the research community, whereas the shipbuilding industry has garnered relatively little focus. We also include the textile industry in the table which has no notable associated works. Nevertheless, we predict that future works will be associated to this industry due to the relevance of CAD to textile.

7 Future Directions

Leveraging LLMs in CAD domain is becoming increasingly popular. Our survey shows that the research community has already started exploiting the potential of LLMs to address different CAD challenges. However, this area of research is still in its nascency. Due to the practical applications of CAD and growing trend of automation across industries, we expect to see a much wider interest of the research community in this direction in the near future. Whereas we have already pointed towards the interesting explorations for future research at the intersection of LLMs and CAD

Table 4. A summary of the popular datasets used in the literature.

Dataset	Associated Works	Source Dataset	Type	Public
OpenECAD	[208]	DeepCAD [187]	Industry	✓
Omni-CAD	[197]	Onshape	Industry	✗
Text2CAD 1.0 & 1.1	[72]	DeepCAD	Industry	✓
Text-to-CAD	[177]	DeepCAD, SkexGen [198]	Industry	✗
CAD-GPT	[178]	DeepCAD	Industry	✗
BlendNet	[43]	-	Synthetic	✓
LLM4CAD	[91, 92, 158]	-	Synthetic	✗
CADPrompt	[4]	DeepCAD	Industry	✗
Query2CAD	[12]	-	Synthetic	✓
CAD-Assistant	[110]	SGPBench [135]	Industry	✗
CAD-Recode	[144]	-	Synthetic	✓
DeepCAD	[144, 211]	-	Industry	✓
Fusion360 [185]	[144]	-	Industry	✓
CC3D [109]	[144]	-	Industry	✓
QueryCAD	[74]	ABC [78]	Industry	✗
SketchGraphs [148]	[190]	-	Industry	✓
Text2CAD	[90]	DeepCAD	Industry	✗
Img2CAD	[204]	ShapeNet [24]	Synthetic	✗
ChatCAD	[160]	-	Synthetic	✗
ReparamCAD	[79]	-	Synthetic	✗
CADTalk	[207]	-	Synthetic & Industry	✓
CAD2Program	[179]	-	Industry	✗

throughout the article, we enumerate a few more potential directions in this section based on our literature review. We hope that the discussion below paves the way to more effective and well-directed future research.

7.1 Application in Interior / Home Design

A promising future direction for LLMs in CAD is home design, also known as interior design. A few LLM-based tools for home design have already started to emerge [23, 76]. For instance, HomeGPT[5] focuses on redesigning individual rooms with AI, making it ideal for completing home makeovers or room-by-room transformations. RoomGPT[6] is another tool specializing in interior design, similar to ChatGPT, but tailored for creating and visualizing interior design styles and layouts. Users can upload a picture, and these systems generate a “dream space”. In [23], I-Design utilized LLMs to transform text input into feasible scene graph designs, considering the relationships between objects. Littlefair et al. [97] demonstrated that LLMs could be combined with traditional optimization techniques to generate both functional and aesthetically pleasing interior designs. However, some of these tools remain closed source, and others do not focus on 3D CAD models. Consequently, home design represents a promising, but currently an underexplored area that is expected to attract more research attention in the future.

[5]https://www.homegpt.app/
[6]https://www.roomgpt.io/

Table 5. Mapping state-of-the-art research to relevant industries in CAD applications.

Industry	Associated Works
Automotive	[208], [197], [43], [108], [103], [79], [190], [159], [133], [94]
Architecture	[208], [197], [43], [107], [108], [190], [204], [211], [159], [133], [94]
Shipbuilding	[197]
Aerospace	[208], [197], [43], [107], [108], [118], [207], [159], [133], [94]
Textile	-
Manufacturing	[208], [197], [72], [177], [178], [43], [90], [91], [92], [158], [4], [12], [107], [108], [122], [110], [144], [69], [118], [74],n [124], [37], [190], [204], [211], [160], [103], [79], [206], [73], [133], [179], [94]
Medicine	[107], [108], [159], [133], [94]
Electronics	[107], [108], [159], [133], [94]
Consumer products	[190], [211], [103], [207], [133], [179], [94]
Entertainment	[107], [108]
Education	[133], [94]

7.2 Specific Data Format Generation

State-of-the-art LLMs are capable of generating a wide range of data formats, including text, image, audio, video, and code. As LLMs continue to evolve, future versions may expand their capabilities to generate even more complex data formats, such as point cloud and even 3D model. Consequently, the scope of LLM applications in data generation extends beyond merely producing text, captions, or textual descriptions. Future research could explore the potential of LLMs to generate these more specialized forms of data, paving the way for advancements in fields like 3D modeling, spatial data generation, and other emerging domains.

7.3 Building Compliance Checking in AEC

In the AEC industry, building designs must adhere to a wide range of requirements set forth by codes and standards [203]. Compliance with these regulations is essential to avoid legal issues, delays, and safety hazards [145]. Traditionally, ensuring compliance has been a manual process, requiring labor-intensive checks using 2D drawings and documents, which is time-consuming and costly. Additionally, building codes are complex and frequently updated, further complicating the process. As a result, the need for Automatic Compliance Checking (ACC) has emerged as a critical solution. ACC involves two main steps: understanding design requirements from textual descriptions and analyzing 3D CAD models or 2D drawings to verify that they meet these requirements. Despite its importance, a universal and scalable ACC system remains a challenge [6].

LLMs have already shown some promise in addressing this challenge within the AEC industry. For instance, Zheng and Fischer utilized GPT technologies to enhance natural language-based building information modeling (BIM) searches [213]. Additionally, Ying and Sacks [203] explored the use of LLMs for generic building compliance checking using BIM data. Du et al. [41] further leveraged LLMs to convert textual descriptions into code, generating editable BIM models and guiding iterative improvements in model quality. While these efforts focus on building infrastructure, there is a noticeable gap in research that directly applies LLMs to 3D CAD models. However, using LLMs to analyze 3D CAD models or 2D CAD drawings for automatic compliance checking holds significant potential, and further exploration in this area could greatly benefit the AEC industry.

7.4 Fashion AI and Textile Design

While much attention has been given to various applications of LLMs, fashion design, particularly in the context of textiles, remains largely unexplored. From Table 5, we observe that no state-of-the-art works have focused on leveraging LLMs for textile design. However, CAD plays a crucial role in fashion design, encompassing functions such as creative visualization, technical pattern development, and style information representation [186].

CAD in fashion design has streamlined the exchange of information across complex network of communication channels, reducing time, cost, and material usage, while also improving accuracy and garment quality. This is largely due to the digitization of design information, which facilitates communication and decision-making without relying on physical samples. Given these advantages, fashion design presents an exciting opportunity for the application of LLMs in generating 3D CAD models. The potential to combine LLMs with fashion CAD tools could open up new avenues for creativity, efficiency, and innovation in the fashion industry.

8 Conclusion

In this comprehensive review, we explored the intersection of LLMs and CAD, highlighting the transformative potential of LLMs in various CAD-related tasks. We began by introducing the significance of CAD in industry, emphasizing its broad applications across different sectors. We then delved into the foundational principles of LLMs in an accessible manner, covering their architecture, training methods, and fine-tuning processes that enable them to perform across diverse domains. We also presented an overview of state-of-the-art LLMs, focusing on both closed-source models like GPT and PaLM and publicly available models such as LLaMA and DeepSeek. Our review underscored the remarkable capabilities of these models in handling complex tasks related to CAD, from data generation to coding and parametric generation, image synthesis, model evaluation and text generation. These advancements open up new possibilities for automating design workflows, improving efficiency, and enhancing creativity in CAD processes. Additionally, we critically analyzed current research trends, pinpointing emerging areas and potential gaps in the existing literature. Finally, we outlined promising future directions, including exploring LLMs in home design, specific data format generation, building compliance checking in AEC, and fashion design and textile applications. These areas offer exciting opportunities for future research and innovation, potentially leading to significant breakthroughs in the integration of LLMs with CAD.

References

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. arXiv:2404.14219 [cs.CL]

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[3] STĂNCIOIU Alin and Elena Loredana STĂNCIOIU. 2024. The evolution and impact of cad in modern industries: a comprehensive review. *Fiability & Durability/Fiabilitate si Durabilitate* 33, 1 (2024).

[4] Kamel Alrashedy, Pradyumna Tambwekar, Zulfiqar Haider Zaidi, Megan Langwasser, Wei Xu, and Matthew Gombolay. 2024. Generating CAD Code with Vision-Language Models for 3D Designs. In *The Thirteenth International Conference on Learning Representations*.

[5] Benjamin Alt, Florian Stöckl, Silvan Müller, Christopher Braun, Julian Raible, Saad Alhasan, Oliver Rettig, Lukas Ringle, Darko Katic, Rainer Jäkel, et al. 2024. RoboGrind: Intuitive and Interactive Surface Treatment with Industrial Robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2140–2146.

[6] Robert Amor and Johannes Dimyadi. 2021. The promise of automated compliance checking. *Developments in the built environment* 5 (2021), 100039.

[7] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403* (2023).

[8] Yadagiri Annepaka and Partha Pakray. 2024. Large language models: A survey of their development, capabilities, and applications. *Knowledge and Information Systems* (2024), 1–56.

[9] anon8231489123. 2023. ShareGPT Vicuna Unfiltered Dataset. https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered.

[10] Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku. https://api.semanticscholar.org/CorpusID:268232499

[11] Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An Open Language Model for Mathematics. In *12th International Conference on Learning Representations, ICLR 2024*.

[12] Akshay Badagabettu, Sai Sravan Yarlagadda, and Amir Barati Farimani. 2024. Query2CAD: Generating CAD models using natural language queries. *CoRR* abs/2406.00144 (2024). arXiv:2406.00144

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

[14] Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *Proceedings of the international AAAI conference on web and social media*, Vol. 14. 830–839.

[15] Anas Belfathi, Nicolas Hernandez, and Laura Monceaux. 2023. Harnessing GPT-3.5-turbo for Rhetorical Role Prediction in Legal Cases. In *JURIX 2023-The 36th International Conference on Legal Knowledge and Information Systems*.

[16] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, et al. 2024. Stable lm 2 1.6 b technical report. *arXiv preprint arXiv:2402.17834* (2024).

[17] Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. 2024. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism. *arXiv e-prints* (2024), arXiv–2401.

[18] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*. 2397–2430.

[19] Stephen Brade, Bryan Wang, Mauricio Sousa, Sageev Oore, and Tovi Grossman. 2023. Promptify: Text-to-image generation through interactive prompt exploration with large language models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–14.

[20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[21] Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297* (2024).

[22] Bradley Camburn, Vimal Viswanathan, Julie Linsey, David Anderson, Daniel Jensen, Richard Crawford, Kevin Otto, and Kristin Wood. 2017. Design prototyping methods: state of the art in strategies, techniques, and guidelines. *Design Science* 3 (2017).

[23] Ata Çelen, Guo Han, Konrad Schindler, Luc Van Gool, Iro Armeni, Anton Obukhov, and Xi Wang. 2024. I-design: Personalized llm interior designer. *arXiv preprint arXiv:2404.02838* (2024).

[24] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015).

[25] Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. 2023. AlpaGasus: Training a Better Alpaca with Fewer Data. In *The Twelfth International Conference on Learning Representations*.

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

[27] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. 2024. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271* (2024).

[28] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24185–24198.

[29] Wen Cheng, Ke Sun, Xinyu Zhang, and Wei Wang. 2025. Security attacks on llm-based code completion tools. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 23669–23677.

[30] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. *https://lmsys.org/blog/2023-03-30-vicuna* 3, 5 (2023).

[31] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[32] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research* 25, 70 (2024), 1–53.

[33] Damai Dai, Chengqi Deng, Chenggang Zhao, Rx Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. 2024. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1280–1297.

[34] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. arXiv:2305.06500 [cs.CV]

[35] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *International conference on machine learning*. PMLR, 933–941.

[36] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. 2023. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*. 7480–7512.

[37] Haoxuan Deng, Samir Khan, and John Ahmet Erkoyuncu. 2024. An Investigation on Utilizing Large Language Model for Industrial Computer-Aided Design Automation. *Procedia CIRP* 128 (2024), 221–226. 34th CIRP Design Conference.

[38] Ismail Dergaa, Karim Chamari, Piotr Zmijewski, and Helmi Ben Saad. 2023. From human writing to artificial intelligence generated text: examining the prospects and potential threats of ChatGPT in academic writing. *Biology of sport* 40, 2 (2023), 615–622.

[39] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems* 36 (2023), 10088–10115.

[40] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. PaLM-E: An Embodied Multimodal Language Model. In *International Conference on Machine Learning*. 8469–8488.

[41] Changyu Du, Sebastian Esser, Stavros Nousias, and André Borrmann. 2024. Text2BIM: Generating Building Models Using a Large Language Model-based Multi-Agent Framework. *arXiv preprint arXiv:2408.08054* (2024).

[42] Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*. PMLR, 5547–5569.

[43] Yuhao Du, Shunian Chen, Wenbo Zan, Peizhao Li, Mingxuan Wang, Dingjie Song, Bo Li, Yan Hu, and Benyou Wang. 2024. BlenderLLM: Training Large Language Models for Computer-Aided Design with Self-improvement. *arXiv preprint arXiv:2412.14203* (2024).

[44] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027* (2020).

[45] Zhangwei Gao, Zhe Chen, Erfei Cui, Yiming Ren, Weiyun Wang, Jinguo Zhu, Hao Tian, Shenglong Ye, Junjun He, Xizhou Zhu, et al. 2024. Mini-InternVL: a flexible-transfer pocket multi-modal model with 5% parameters and 90% performance. *Visual Intelligence* 2, 1 (2024), 1–17.

[46] Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. Koala: A Dialogue Model for Academic Researc. *Blog post* (April 2023).

[47] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793* (2024).

[48] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 315–323.

[49] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[50] GP Gujarathi and Y-S Ma. 2011. Parametric CAD/CAE integration using a common data model. *Journal of Manufacturing Systems* 30, 3 (2011), 118–132.

[51] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

[52] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. 2024. DeepSeek-Coder: When the Large Language Model Meets Programming–The Rise of Code Intelligence. *arXiv preprint arXiv:2401.14196* (2024).

[53] Muhammad Usman Hadi, Qasem Al Tashi, Abbas Shah, Rizwan Qureshi, Amgad Muneer, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, et al. 2024. Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects. *Authorea Preprints* (2024).

[54] Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. 2023. A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints* 3 (2023).

[55] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* (2016).

[56] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*. 30016–30030.

[57] Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. 2024. Large language models for software engineering: A systematic literature review. *ACM Transactions on Software Engineering and Methodology* 33, 8 (2024), 1–79.

[58] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

[59] Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, et al. 2024. MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies. In *First Conference on Language Modeling*.

[60] Bofeng Huang. 2023. Vigogne: French Instruction-following and Chat Models. https://github.com/bofenghuang/vigogne.

[61] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, et al. 2024. Qwen2. 5-Coder Technical Report. *arXiv preprint arXiv:2409.12186* (2024).

[62] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[63] Peter P Ikubanni, Adekunle A Adeleke, Olayinka O Agboola, Chiebuka T Christopher, Boluwatife S Ademola, Joseph Okonkwo, Olanrewaju S Adesina, Peter O Omoniyi, and Esther T Akinlabi. 2022. Present and future impacts of computer-aided design/computer-aided manufacturing (CAD/CAM). *Journal Européen des Systèmes Automatisés* 55, 3 (2022), 349.

[64] Mete Ismayilzada, Debjit Paul, Syrielle Montariol, Mor Geva, and Antoine Bosselut. 2023. CRoW: Benchmarking Commonsense Reasoning in Real-World Tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 9785–9821.

[65] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720* (2024).

[66] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL]

[67] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088* (2024).

[68] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences* 11, 14 (2021), 6421.

[69] Benjamin T Jones, Felix Hähnlein, Zihan Zhang, Maaz Ahmad, Vladimir Kim, and Adriana Schulz. 2025. A Solver-Aided Hierarchical Language for LLM-Driven CAD Design. *arXiv preprint arXiv:2502.09819* (2025).

[70] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361* (2020).

[71] Koray Kavukcuoglu. 2025. Gemini 2.5: Our most intelligent AI model. *Google Blog* (2025).

[72] Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. 2024. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. *Advances in Neural Information Processing Systems* 37 (2024), 7552–7579.

[73] Muhammad Tayyab Khan, Lequn Chen, Ye Han Ng, Wenhe Feng, Nicholas Yew Jin Tan, and Seung Ki Moon. 2024. Leveraging Vision-Language Models for Manufacturing Feature Recognition in CAD Designs. *arXiv preprint arXiv:2411.02810* (2024).

[74] Claudius Kienle, Benjamin Alt, Darko Katic, and Rainer Jäkel. 2024. QueryCAD: Grounded Question Answering for CAD Models. *arXiv preprint arXiv:2409.08704* (2024).

[75] Jeonghoon Kim, Byeongchan Lee, Cheonbok Park, Yeontaek Oh, Beomjun Kim, Taehwan Yoo, Seongjin Shin, Dongyoon Han, Jinwoo Shin, and Kang Min Yoo. 2025. Peri-LN: Revisiting Layer Normalization in the Transformer Architecture. *arXiv preprint arXiv:2502.02732* (2025).

[76] Evan King, Haoxiang Yu, Sangsu Lee, and Christine Julien. 2023. "Get ready for a party": Exploring smarter smart spaces with help from large language models. *arXiv e-prints* (2023), arXiv–2303.

[77] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, et al. 2023. Segment Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 4015–4026.

[78] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. 2019. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9601–9611.

[79] Milin Kodnongbua, Benjamin Jones, Maaz Bin Safeer Ahmad, Vladimir Kim, and Adriana Schulz. 2023. ReparamCAD: Zero-shot CAD Re-Parameterization for Interactive Manipulation. In *SIGGRAPH Asia 2023 Conference Papers*.

[80] Jing Yu Koh, Daniel Fried, and Russ R Salakhutdinov. 2023. Generating images with multimodal language models. *Advances in Neural Information Processing Systems* 36 (2023), 21487–21506.
[81] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012).
[82] Pranjal Kumar. 2024. Large language models (LLMs): survey, technical frameworks, and future challenges. *Artificial Intelligence Review* 57, 10 (2024), 260.
[83] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.
[84] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33 (2020), 9459–9474.
[85] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024. LLaVA-OneVision: Easy Visual Task Transfer. arXiv:2408.03326 [cs.CV]
[86] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*. 19730–19742.
[87] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024. Pre-trained language models for text generation: A survey. *Comput. Surveys* 56, 9 (2024), 1–39.
[88] Pengfei Li, Min Zhang, Peijie Lin, Jian Wan, and Ming Jiang. 2022. Conditional embedding pre-training language model for image captioning. *Neural Processing Letters* 54, 6 (2022), 4987–5003.
[89] R Li, LB Allal, Y Zi, N Muennighoff, D Kocetkov, C Mou, M Marone, C Akiki, J Li, J Chim, et al. 2023. StarCoder: May the Source be With You! *Transactions on machine learning research* (2023).
[90] Xueyang Li, Yu Song, Yunzhong Lou, and Xiangdong Zhou. 2024. CAD Translator: An Effective Drive for Text to 3D Parametric Computer-Aided Design Generative Modeling. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 8461–8470.
[91] Xingang Li, Yuewan Sun, and Zhenghui Sha. 2024. LLM4CAD: Multi-Modal Large Language Models for 3D Computer-Aided Design Generation. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 88407. V006T06A015.
[92] Xingang Li, Yuewan Sun, and Zhenghui Sha. 2025. LLM4CAD: Multimodal Large Language Models for Three-Dimensional Computer-Aided Design Generation. *Journal of Computing and Information Science in Engineering* 25, 2 (2025).
[93] Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Xuying Meng, Siqi Fan, Peng Han, Jing Li, Li Du, Bowen Qin, et al. 2023. Flm-101b: An open llm and how to train it with $100 k budget. *arXiv preprint arXiv:2309.03852* (2023).
[94] Yiwei Li, Huaqin Zhao, Hanqi Jiang, Yi Pan, Zhengliang Liu, Zihao Wu, Peng Shu, Jie Tian, Tianze Yang, Shaochen Xu, et al. 2024. Large Language Models for Manufacturing. *arXiv e-prints* (2024), arXiv–2410.
[95] Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. 2024. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887* (2024).
[96] Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. 2024. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*. 366–384.
[97] Gabrielle Littlefair, Niladri Shekhar Dutt, and Niloy J Mitra. 2025. FlairGPT: Repurposing LLMs for interior designs. In *EUROGRAPHICS*.
[98] Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434* (2024).
[99] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. DeepSeek-V3 Technical Report. *arXiv e-prints* (2024), arXiv–2412.
[100] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved Baselines with Visual Instruction Tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 26296–26306.
[101] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. https://llava-vl.github.io/blog/2024-01-30-llava-next/
[102] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*.
[103] Vivian Liu, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. 2023. 3DALL-E: Integrating text-to-image AI in 3D design workflows. In *Proceedings of the 2023 ACM designing interactive systems conference*. 1955–1977.
[104] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. 2024. DeepSeek-VL: Towards Real-World Vision-Language Understanding. *arXiv preprint arXiv:2403.05525* (2024).
[105] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language Models of Code are Few-Shot Commonsense Learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 1384–1403.
[106] Dakota Mahan, Ryan Carlow, Louis Castricato, Nathan Cooper, and Christian Laforte. [Online]. Stable Beluga models. https://huggingface.co/stabilityai/StableBeluga2
[107] Liane Makatura, Michael Foshey, Bohan Wang, Felix Hähnlein, Pingchuan Ma, Bolei Deng, Megan Tjandrasuwita, Andrew Spielberg, Crystal Elaine Owens, Peter Yichen Chen, et al. 2023. How Can Large Language Models Help Humans in Design and Manufacturing? *CoRR* (2023).

[108] Liane Makatura, Michael Foshey, Bohan Wang, Felix Hähnlein, Pingchuan Ma, Bolei Deng, Megan Tjandrasuwita, Andrew Spielberg, Crystal Elaine Owens, Peter Yichen Chen, et al. 2024. Large Language Models for Design and Manufacturing. *An MIT Exploration of Generative AI* (mar 27 2024).

[109] Dimitrios Mallis, Ali Sk Aziz, Elona Dupont, Kseniya Cherenkova, Ahmet Serdar Karadeniz, Mohammad Sadil Khan, Anis Kacem, Gleb Gusev, and Djamila Aouada. 2023. SHARP Challenge 2023: Solving CAD History and pArameters Recovery from Point Clouds and 3D Scans. Overview, Datasets, Metrics, and Baselines.. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 1786–1795.

[110] Dimitrios Mallis, Ahmet Serdar Karadeniz, Sebastian Cavada, Danila Rukhovich, Niki Foteinopoulou, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2024. CAD-Assistant: Tool-Augmented VLLMs as Generic CAD Task Solvers? *arXiv e-prints* (2024), arXiv–2412.

[111] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. 2018. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162* (2018).

[112] Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, et al. 2024. Openelm: An efficient language model family with open training and inference framework. *arXiv preprint arXiv:2404.14619* (2024).

[113] Timo Meinders, IA Burchitz, MHA Bonte, and RA Lingbeek. 2008. Numerical product design: Springback prediction, compensation and optimization. *International Journal of Machine Tools and Manufacture* 48, 5 (2008), 499–514.

[114] Meta AI. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

[115] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large Language Models: A Survey. arXiv:2402.06196 [cs.CL]

[116] 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. 2023. Crosslingual Generalization through Multitask Finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 15991–16111.

[117] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707* (2023).

[118] Atharva Naik. 2024. Artificial Intelligence and Large Language Models in CAD. http://hdl.handle.net/20.500.12380/308899

[119] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332* (2021).

[120] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435* (2023).

[121] Mohamed Nejjar, Luca Zacharias, Fabian Stiehle, and Ingo Weber. 2025. Llms for science: Usage for code generation and data analysis. *Journal of Software: Evolution and Process* 37, 1 (2025), e2723.

[122] Matt D Nelson, Brady L Goenner, and Bruce K Gale. 2023. Utilizing ChatGPT to assist CAD design for microfluidic devices. *Lab on a Chip* 23, 17 (2023), 3778–3784.

[123] Erik Nijkamp, Hiroaki Hayashi, Caiming Xiong, Silvio Savarese, and Yingbo Zhou. 2023. CodeGen2: Lessons for Training LLMs on Programming and Natural Languages. *arXiv e-prints* (2023), arXiv–2305.

[124] Felix Ocker, Stefan Menzel, Ahmed Sadik, and Thiago Rios. 2025. From Idea to CAD: A Language Model-Driven Multi-Agent System for Collaborative Design. *arXiv preprint arXiv:2503.04417* (2025).

[125] OpenAI. 2023. GPT-4V(ision) System Card. https://cdn.openai.com/papers/GPTV_System_Card.pdf

[126] OpenAI. 2025. OpenAI GPT-4.5 System Card. https://cdn.openai.com/gpt-4-5-system-card-2272025.pdf

[127] OpenAI. 2025. Openai o3-mini system card. *OpenAI blog* (2025).

[128] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[129] Arka Pal, Deep Karkhanis, Manley Roberts, Samuel Dooley, Arvind Sundararajan, and Siddartha Naidu. 2023. Giraffe: Adventures in expanding context lengths in llms. *arXiv preprint arXiv:2308.10882* (2023).

[130] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2024. Gorilla: Large Language Model Connected with Massive APIs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[131] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data only. *Advances in Neural Information Processing Systems* 36 (2023), 79155–79172.

[132] Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *Advances in neural information processing systems* 34 (2021), 11054–11070.

[133] Cyril Picard, Kristen M Edwards, Anna C Doris, Brandon Man, Giorgio Giannone, Md Ferdous Alam, and Faez Ahmed. 2023. From concept to manufacturing: Evaluating vision-language models for engineering design. *arXiv preprint arXiv:2311.12668* (2023).

[134] Sundar Pichai, D Hassabis, and K Kavukcuoglu. 2024. Introducing Gemini 2.0: our new AI model for the agentic era. *Google Blog* (2024).

[135] Zeju Qiu, Weiyang Liu, Haiwen Feng, Zhen Liu, Tim Z Xiao, Katherine M Collins, Joshua B Tenenbaum, Adrian Weller, Michael J Black, and Bernhard Schölkopf. 2024. Can Large Language Models Understand Symbolic Graphics Programs? *arXiv preprint arXiv:2408.08313* (2024).

[136] Virgilio Quintana, Louis Rivest, Robert Pellerin, Frédérick Venne, and Fawzi Kheddouci. 2010. Will Model-based Definition replace engineering drawings throughout the product lifecycle? A global perspective from aerospace industry. *Computers in industry* 61, 5 (2010), 497–508.

[137] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. 8748–8763.

[138] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

[139] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[140] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv e-prints* (2022), arXiv–2204.

[141] Xiaozhe Ren, Pingyi Zhou, Xinfan Meng, Xinjing Huang, Yadao Wang, Weichao Wang, Pengfei Li, Xiaoda Zhang, Alexander Podolskiy, Grigory Arshinov, Andrey Bout, Irina Piontkovskaya, Jiansheng Wei, Xin Jiang, Teng Su, and Qun Liu. 2023. PanGu-Σ: Towards Trillion Parameter Language Model with Sparse Heterogeneous Computing. *arXiv preprint arXiv:2303.10845* (2023).

[142] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, et al. 2021. Recipes for Building an Open-Domain Chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 300–325.

[143] Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code Llama: Open Foundation Models for Code. *arXiv e-prints* (2023), arXiv–2308.

[144] Danila Rukhovich, Elona Dupont, Dimitrios Mallis, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2024. CAD-Recode: Reverse Engineering CAD Code from Point Clouds. *arXiv e-prints* (2024), arXiv–2412.

[145] Abdullahi Saka, Ridwan Taiwo, Nurudeen Saka, Babatunde Abiodun Salami, Saheed Ajayi, Kabiru Akande, and Hadi Kazemi. 2024. GPT models in construction industry: Opportunities, limitations, and a use case validation. *Developments in the Built Environment* 17 (2024), 100300.

[146] MMM Sarcar, K Mallikarjuna Rao, and K Lalit Narayan. 2008. *Computer aided design and manufacturing*.

[147] John Schulman, Barret Zoph, Christina Kim, Jacob Hilton, Jacob Menick, Jiayi Weng, Juan Felipe Ceron Uribe, Liam Fedus, Luke Metz, Michael Pokorny, et al. 2022. Introducing ChatGPT. *OpenAI blog* (2022).

[148] Ari Seff, Yaniv Ovadia, Wenda Zhou, and Ryan P Adams. 2020. Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design. *arXiv preprint arXiv:2007.08506* (2020).

[149] Wasswa Shafik. 2024. Introduction to ChatGPT. In *Advanced applications of generative AI and natural language processing models*. 1–25.

[150] Kimya Khakzad Shahandashti, Mithila Sivakumar, Mohammad Mahdi Mohajer, Alvine B Belle, Song Wang, and Timothy C Lethbridge. 2024. Evaluating the effectiveness of gpt-4 turbo in creating defeaters for assurance cases. *arXiv preprint arXiv:2401.17991* (2024).

[151] Ashish Kumar Shakya, Gopinatha Pillai, and Sohom Chakrabarty. 2023. Reinforcement learning algorithms: A brief survey. *Expert Systems with Applications* 231 (2023), 120495.

[152] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv e-prints* (2024), arXiv–2402.

[153] Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202* (2020).

[154] Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025* (2023).

[155] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2023. Large language models encode clinical knowledge. *Nature* 620, 7972 (2023), 172–180.

[156] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Mohamed Amin, Le Hou, Kevin Clark, Stephen R Pfohl, Heather Cole-Lewis, et al. 2025. Toward expert-level medical question answering with large language models. *Nature Medicine* (2025), 1–8.

[157] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. 2023. PandaGPT: One Model To Instruction-Follow Them All. In *Proceedings of the 1st Workshop on Taming Large Language Models: Controllability in the era of Interactive Assistants!* 11–23.

[158] Yuewan Sun, Xingang Li, and Zhenghui Sha. 2025. Large Language Models for Computer-Aided Design (LLM4CAD) Fine-Tuned: Dataset and Experiments. *Journal of Mechanical Design* (2025), 1–19.

[159] Yunlei Sun, Bingyi Yan, and Zhaotong Shao. 2024. Ai large models bring great opportunities to reusable design of cad software. *Computer Science and Information Systems* 00 (2024), 46–46.

[160] Jing Tang, Hongru Xiao, Xiang Li, Wei Wang, and Zeyu Gong. 2025. ChatCAD: An MLLM-Guided Framework for Zero-shot CAD Drawing Restoration. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1–5.

[161] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html* 3, 6 (2023), 7.

[162] Yi Tay, Jason Wei, Hyung Chung, Vinh Tran, David So, Siamak Shakeri, Xavier Garcia, Steven Zheng, Jinfeng Rao, Aakanksha Chowdhery, et al. 2023. Transcending Scaling Laws with 0.1% Extra Compute. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 1471–1486.

[163] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[164] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530* (2024).

[165] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295* (2024).

[166] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving Open Language Models at a Practical Size. *arXiv e-prints* (2024), arXiv–2408.

[167] MosaicML NLP Team. 2023. *Introducing MPT-7B: A New Standard for Open-Source, Commercially Usable LLMs.* https://www.databricks.com/blog/mpt-7b

[168] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239* (2022).

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

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

[171] Lewis Tunstall, Edward Emanuel Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro Von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct Distillation of LM Alignment. In *First Conference on Language Modeling*.

[172] Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2023. Focused transformer: Contrastive training for context scaling. *Advances in neural information processing systems* 36 (2023), 42661–42688.

[173] Anwaar Ulhaq and Naveed Akhtar. 2022. Efficient diffusion models for vision: A survey. *arXiv preprint arXiv:2210.09292* (2022).

[174] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[175] Charles Waghmare. 2023. Introduction to ChatGPT. In *Unleashing The Power of ChatGPT: A Real World Business Applications*. 1–26.

[176] Jianing Wang, Qiushi Sun, Xiang Li, and Ming Gao. 2024. Boosting Language Models Reasoning with Chain-of-Knowledge Prompting. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 4958–4981.

[177] Ruiyu Wang, Yu Yuan, Shizhao Sun, and Jiang Bian. 2025. Text-to-CAD Generation Through Infusing Visual Feedback in Large Language Models. *arXiv preprint arXiv:2501.19054* (2025).

[178] Siyu Wang, Cailian Chen, Xinyi Le, Qimin Xu, Lei Xu, Yanzhou Zhang, and Jie Yang. 2025. CAD-GPT: Synthesising CAD Construction Sequence with Spatial Reasoning-Enhanced Multimodal LLMs. In *AAAI*.

[179] Xilin Wang, Jia Zheng, Yuanchao Hu, Hao Zhu, Qian Yu, and Zihan Zhou. 2025. From 2D CAD Drawings to 3D Parametric Models: A Vision-Language Approach. In *AAAI*.

[180] Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems* 36 (2023), 74764–74786.

[181] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 13484–13508.

[182] Yue Wang, Hung Le, Akhilesh Gotmare, Nghi Bui, Junnan Li, and Steven Hoi. 2023. CodeT5+: Open Code Large Language Models for Code Understanding and Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 1069–1088.

[183] Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, et al. 2023. Skywork: A More Open Bilingual Foundation Model. *arXiv e-prints* (2023), arXiv–2310.

[184] Philip Welsby and Bernard MY Cheung. 2023. ChatGPT. 1047–1048 pages.

[185] Karl DD Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G Lambourne, Armando Solar-Lezama, and Wojciech Matusik. 2021. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–24.

[186] Jennifer Xiaopei Wu and Li Li. 2025. AI-driven computational creativity in fashion design: a review. *Textile Research Journal* 95, 5-6 (2025), 658–675.

[187] Rundi Wu, Chang Xiao, and Changxi Zheng. 2021. Deepcad: A deep generative network for computer-aided design models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6772–6782.

[188] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. BloombergGPT: A Large Language Model for Finance. *arXiv e-prints* (2023), arXiv–2303.

[189] Sifan Wu, Amir Khasahmadi, Mor Katz, Pradeep Kumar Jayaraman, Yewen Pu, Karl Willis, and Bang Liu. 2023. Cad-llm: Large language model for cad generation. In *NeurIPS 2023 Workshop on Machine Learning for Creativity and Design*.

[190] Sifan Wu, Amir Hosein Khasahmadi, Mor Katz, Pradeep Kumar Jayaraman, Yewen Pu, Karl Willis, and Bang Liu. 2024. Cadvlm: Bridging language and vision in the generation of parametric cad sketches. In *European Conference on Computer Vision*. Springer, 368–384.

[191] Xinwei Wu, Junzhuo Li, Minghui Xu, Weilong Dong, Shuangzhi Wu, Chao Bian, and Deyi Xiong. 2023. DEPN: Detecting and Editing Privacy Neurons in Pretrained Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2875–2886.

[192] Yonghui Wu. 2024. Large language model and text generation. In *Natural Language Processing in Biomedicine: A Practical Guide*. 265–297.

[193] Tong Xiao and Jingbo Zhu. 2025. Foundations of Large Language Models. *arXiv preprint arXiv:2501.09223* (2025).

[194] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International conference on machine learning*. 10524–10533.

[195] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering Large Pre-Trained Language Models to Follow Complex Instructions. In *The Twelfth International Conference on Learning Representations*.

[196] Frank F Xu, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn. 2022. A systematic evaluation of large language models of code. In *Proceedings of the 6th ACM SIGPLAN international symposium on machine programming*. 1–10.

[197] Jingwei Xu, Zibo Zhao, Chenyu Wang, Wen Liu, Yi Ma, and Shenghua Gao. 2025. CAD-MLLM: Unifying Multimodality-Conditioned CAD Generation With MLLM. arXiv:2411.04954 [cs.CV]

[198] Xiang Xu, Karl DD Willis, Joseph G Lambourne, Chin-Yi Cheng, Pradeep Kumar Jayaraman, and Yasutaka Furukawa. 2022. SkexGen: Autoregressive Generation of CAD Construction Sequences with Disentangled Codebooks. In *International Conference on Machine Learning*. 24698–24724.

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

[200] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 Technical Report. arXiv:2407.10671 [cs.CL]

[201] Xiaocheng Yang, Bingsen Chen, and Yik-Cheung Tam. 2024. Arithmetic Reasoning with LLM: Prolog Generation & Permutation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*. 699–710.

[202] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2024. A survey on multimodal large language models. *National science review* 11, 12 (2024), nwae403.

[203] Huaquan Ying and Rafael Sacks. 2024. From Automatic to Autonomous: A Large Language Model-driven Approach for Generic Building Compliance Checking. In *Proceedings of the CIB W78 Conference*.

[204] Yang You, Mikaela Angelina Uy, Jiaqi Han, Rahul Thomas, Haotong Zhang, Suya You, and Leonidas Guibas. 2024. Img2cad: Reverse engineering 3d cad models from images through vlm-assisted conditional factorization. *arXiv preprint arXiv:2408.01437* (2024).

[205] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. 2022. CoCa: Contrastive Captioners are Image-Text Foundation Models. *Transactions on Machine Learning Research* Aug 2022 (2022).

[206] Mengqiang Yu, Yujie Sheng, Zhongxu Wan, Shuyu Yang, Muping Sun, and Hongxia Cai. 2024. An intelligent interactive system based on LLM to combine CAD API development with FEA. In *2024 8th International Conference on Electrical, Mechanical and Computer Engineering (ICEMCE)*. 1749–1755.

[207] Haocheng Yuan, Jing Xu, Hao Pan, Adrien Bousseau, Niloy J. Mitra, and Changjian Li. 2024. CADTalk: An Algorithm and Benchmark for Semantic Commenting of CAD Programs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 3753–3762.

[208] Zhe Yuan, Jianqi Shi, and Yanhong Huang. 2024. OpenECAD: An efficient visual language model for editable 3D-CAD design. *Computers & Graphics* 124 (2024), 104048.

[209] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*. 3836–3847.

[210] Xuanyu Zhang and Qing Yang. 2023. Xuanyuan 2.0: A large chinese financial chat model with hundreds of billions parameters. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 4435–4439.

[211] Zhanwei Zhang, Shizhao Sun, Wenxiao Wang, Deng Cai, and Jiang Bian. 2025. FlexCAD: Unified and Versatile Controllable CAD Generation with Fine-tuned Large Language Models. In *The Thirteenth International Conference on Learning Representations*.

[212] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* 1, 2 (2023).

[213] Junwen Zheng and Martin Fischer. 2023. Dynamic prompt-based virtual assistant framework for BIM information search. *Automation in Construction* 155 (2023), 105067.

[214] Baichuan Zhou, Ying Hu, Xi Weng, Junlong Jia, Jie Luo, Xien Liu, Ji Wu, and Lei Huang. 2024. TinyLLaVA: A Framework of Small-scale Large Multimodal Models. arXiv:2402.14289 [cs.LG]

[215] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems* 36 (2023), 55006–55021.

[216] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2024. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *12th International Conference on Learning Representations, ICLR 2024*.