Scilight

Smart Chemical Engineering

https://www.sciltp.com/journals/sce

Review

Large Language Models for Automating Computational Fluid Dynamics (CFD): From Predictive Modeling and Optimization to Execution Scheduling

Pei-Zhong Ma [1], Guo-Dong Gai [2], Jian-Kun Li [3,*], Zheng-Hong Luo [4,*] and Li-Tao Zhu [1,*]

[1] College of Smart Energy, Shanghai Jiao Tong University, Shanghai 200240, China
[2] IMFT, Université de Toulouse, Toulouse INP, CNRS, 31400 Toulouse, France
[3] State Key Laboratory of Polyolefins and Catalysis, Shanghai Key Laboratory of Catalysis Technology for Polyolefins, Shanghai Research Institute of Chemical Industry Co., Ltd., No. 345 East Yunling Rd, Shanghai 200062, China
[4] Department of Chemical Engineering, School of Chemistry and Chemical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
* Correspondence: lijiankun@shhuayi.com (J.-K.L.); luozh@sjtu.edu.cn (Z.-H.L.); sjtu_zlt@sjtu.edu.cn (L.-T.Z.)

How To Cite: Ma, P.-Z.; Gai, G.-D.; Li, J.-K.; et al. Large Language Models for Automating Computational Fluid Dynamics (CFD): From Predictive Modeling and Optimization to Execution Scheduling. *Smart Chemical Engineering* **2026**, *2*(1), 3. https://doi.org/10.53941/sce.2026.100003

Received: 3 February 2026
Revised: 25 February 2026
Accepted: 5 March 2026
Published: 18 March 2026

Abstract: Artificial intelligence (AI), especially large language models (LLMs), are triggering an intelligent transformation in the field of computational fluid dynamics (CFD). LLMs can assist and even partially replace humans in executing complex fluid simulation tasks. Due to such opportunities, we systematically review recent progress and limitations of applying LLMs to the CFD. Specifically, three key applications of LLMs in CFD are surveyed, explored, and analyzed, including: (1) Predictive modeling for prediction of fluid behavior (such as real-time generation of flow field distributions) and automatic discovery of turbulence models (such as improvement of the k-ε model based on direct numerical simulation data, achieving implicit encoding of physical laws); (2) Optimization applications, including CFD model optimization (i.e., optimization of numerical model parameters and their sub-models, such as turbulence model parameters), hyperparameter optimization of machine learning models (such as learning rate, optimizer, number of neurons, and hidden layers involved in neural networks training), geometric structure optimization (such as topological optimization of fluid process equipment structures), and process operating parameters (such as inlet velocity, operating pressure and temperature); (3) Automated execution scheduling for full-process end-to-end automation for efficient CFD simulations, including automatic geometry generation, boundary condition configuration, meshing, and solver invocation. Meanwhile, we summarize and analyze key weaknesses, such as insufficient physical credibility and engineering applicability limitations. Finally, we suggest future directions to possibly address these challenges, including reducing the difficulty of obtaining domain data, continuously updating domain models, and improving model interpretability. The complementary relationship between human intelligence and AI (HI-AI) will promote CFD to better serve academic and engineering applications.

Keywords: computational fluid dynamics (CFD); large language models (LLMs); predictive modeling; parameter optimization; automated execution scheduling; intelligent agent

1. Introduction

In computational fluid dynamics (CFD), direct numerical simulation (DNS) [1,2], large eddy simulation (LES) [3], and Reynolds-averaged Navier-Stokes (RANS) [4] constitute the primary frameworks for flow solving and turbulence modeling, each balancing accuracy, cost, and applicability across engineering and scientific research [5]. While DNS offers high fidelity, its prohibitive computational cost restricts it to small scales [6]. LES extends spatiotemporal coverage yet struggles with high-Reynolds-number near-wall flows [6], and RANS, though widely adopted in engineering, often lacks accuracy in complex flows [7–9]. Accurate modeling of multiphase flows is critical for oil and gas transportation industry [10], chemical processes [11], and environmental management [12], aiming to resolve interactions among immiscible phases. Traditional Euler-Euler methods provide efficiency for macroscopic dispersed phases [13] but fail to capture microscopic interfacial behaviors like particle agglomeration. Conversely, Euler-Lagrange models offer detailed dispersed phase information [11] but incur significant costs as particle numbers rise. Specifically, CFD-DEM resolves fluid-solid interactions and particle collisions [14] but remains computationally burdensome for industrial-scale applications involving massive particle counts [15], such as blast furnace simulations [16]. Consequently, traditional numerical approaches face persistent hurdles: unresolved closure models in CFD-DEM/RANS coupling, excessive grid requirements for LES, and the impracticality of DNS for real-world engineering problems. These constraints hinder traditional approaches when tackling complex flow forecasting, multi-scale interactions, or real-time flow field visualization.

Machine learning (ML), particularly deep learning, has fundamentally reshaped predictive modeling in multiphase flows, evolving from pure data-driven approaches to physics-integrated frameworks [17]. Data-driven strategies have proven vital for closure modeling, where they extract complex flow features to correct traditional closure coefficients without requiring explicit functional forms. For example, Zhu et al. [18] utilized such methods to accurately predict mesoscale gas-solid drag and heat transfer coefficients, facilitating efficient coarse-grid CFD simulations. While early applications of data-driven modeling using machine learning methods such as convolutional neural networks (CNN) [19] and XGBoost [20] demonstrated efficacy in flow pattern identification [21–25] and parameter prediction [26–31], the field has increasingly shifted towards methods that embed physical constraints to overcome data scarcity and ensure physical consistency. For instance, physics-informed neural networks (PINNs) [32,33] exemplify this shift by integrating partial differential equation (PDE) residuals as regularization terms, enabling robust reconstruction of turbulent fields even with incomplete boundary conditions [33]. Furthermore, hybrid architectures like the DeepRANS framework [34] leverage CNNs to extract flow features, directly addressing traditional CFD bottlenecks such as high computational costs and limited interface resolution. Despite these advances, current ML models are often heavily reliant on task-specific high-fidelity data, limiting their generalizability and interpretability, a gap that emerging large language models (LLMs) aim to bridge through their reasoning and generative capabilities.

Parallel to predictive modeling, the optimization of CFD workflows has evolved from manual parameter tuning to intelligent, algorithm-driven frameworks. Traditionally, optimization focused on numerical solver settings (e.g., time steps, discretization schemes in FDM/FVM/FEM) and operational conditions (velocity, pressure) to balance efficiency and stability [35]. However, the complexity of modern fluid problems necessitates advanced search strategies beyond manual trial-and-error. The integration of AI has introduced hyperparameter optimization and metaheuristic algorithms as core components of the CFD workflow [36]. For example, particle swarm optimization (PSO) has been successfully deployed to enhance aerodynamic designs, such as optimizing lift coefficients for blown airfoils [37] and configuring Flettner rotors [38]. In chemical engineering, combining ML with optimization algorithms has streamlined the structural design of agitators to improve mixing and energy efficiency [39]. While these ML-aided optimization techniques significantly improve design efficiency, they inherit the black-box limitations of underlying models and often require extensive retraining for new scenarios. This underscores the need for more versatile, language-driven agents capable of understanding complex optimization goals and autonomously navigating the CFD solution space, a central theme of this review.

While traditional ML methods have demonstrated capabilities in predictive modeling and optimization in the context of CFD, their black-box nature and data dependency [39] limit scalability in complex fluid engineering scenarios. This gap is precisely where LLMs emerge as a transformative solution. In recent years, LLMs have provided a promising new pathway for CFD research with their unique capabilities in multi-source information integration, domain knowledge mining, and interpretability enhancement, as well as their user-friendly operation. Among them, generative LLMs have made significant progress in reasoning capabilities [40]. Such models, relying on continuous iteration and optimization, are driving developments in multidisciplinary fields [41], such as the energy system intelligence level boosting [42], pharmaceutical dissolution prediction [43], pharmaceutical powder technology development [44,45], material design [46,47], engineering simulations [47]. In the field of CFD, the

potential applications of LLMs are mainly reflected in predictive modeling and model discovery, closure model construction, and optimization. They can further support automated simulation processes and task scheduling, thereby providing possibilities to address bottlenecks in traditional CFD methods, such as computational costs, characterization of complex interface behaviors, and simulations of large-scale particle systems [48]. In the field of predictive modeling and closure model discovery, LLMs can be combined with deep learning models to extract effective features from high-fidelity data for rapid prediction of key flow quantities, such as modeling and predicting two-phase bubble flows [49]. Additionally, LLMs can integrate symbolic regression and knowledge extraction methods to deduce correction laws for turbulence model parameters from high-fidelity data, and generate analytical forms of turbulence closure terms [50,51]. For example, Zhang et al. [52] developed the AutoTurb framework, which uses LLM to automatically search for algebraic expressions to correct turbulence models in RANS equations, thereby improving the accuracy and generalization ability of separated flow predictions. In terms of optimization, LLMs can be used for tasks such as geometric shape optimization, drag reduction design, and parameter tuning. In this context, Zhang et al. [53] proposed a LLM-PSO framework, which combines LLMs with swarm intelligence/evolutionary strategies, and successfully applied it to parametric shape optimization of 2D airfoils and 3D axisymmetric bodies. However, to our knowledge, there are few reports on the aforementioned applications in fluid process engineering currently. In terms of automated execution and scheduling, traditional CFD simulations often rely heavily on expert experience for tasks ranging from operating condition setup to mesh generation, solver selection, and resource allocation. The barrier to entry limits their broader application in various scenarios. The development of LLMs provides supplementary pathways for automating CFD workflows. Through natural language interaction and rule/template constraints, LLMs can assist in key tasks such as solver selection, mesh strategy recommendations, and computational resource allocation, thereby lowering the barrier to use and improving overall efficiency. Looking ahead, with further integration of physics-informed machine learning and LLMs, as well as exploration of novel computational paradigms (e.g., quantum machine learning) in related problems, the application of LLMs in CFD is expected to advance toward higher reliability and stronger transferability.

Due to such emergence, we systematically review the applications and advances of LLMs in fluid mechanics research, analyzes the key challenges they face, and outlines future development directions. Specifically, we first elaborate on the foundational theoretical framework of LLMs, encompassing the Transformer architecture, attention mechanism, pre-training paradigms, prompt engineering, fine-tuning techniques, and retrieval-augmented generation (RAG) approaches, thereby establishing a theoretical foundation for subsequent application analysis. On this basis, an in-depth exploration is conducted on the application of CFD combined with LLMs in predictive modeling, model optimization, and automated execution. (1) In the context of predictive modeling, we gain insights into the applications of LLMs in flow field parameter prediction, turbulence closure model construction, and multi-physics coupling modeling, with a focus on analyzing the critical role of prompt engineering and fine-tuning techniques in improving prediction accuracy. (2) In optimization, the analysis is conducted from three key perspectives: numerical parameter optimization of CFD models, hyperparameter optimization of machine learning models, and operational condition and geometric structure optimization in fluid dynamics scenarios. (3) At the automation execution level, we highlight representative frameworks such as OpenFOAMGPT [54] and MetaOpenFOAM [55]. These methods leverage RAG technology and multi-agent collaboration mechanisms to achieve intelligence and automation in CFD workflows, including case configuration, solver control, and result analysis.

2. How Large Language Models Work

2.1. Core Architecture

LLMs, as a cutting-edge technology in the field of AI, are based on the deep integration of deep learning, probability statistics, and information theory as their core theoretical foundation. The basic architecture of modern large models is mainly based on Transformer [56], and its architectural principle is shown in Figure 1. The core strategy lies in the self-attention mechanism, which can effectively capture the dependencies between any positions in the sequence. Given the input embedding matrix $H \in \mathbb{R}^{n\times d}$, where n is the sequence length and d is the embedding dimension, the self-attention mechanism is calculated as follows:

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

Here, $Q = HW_Q$, $K = HW_K$, $V = HW_V$ represent the query, key and value matrices, while $W_Q, W_K, W_V \in \mathbb{R}^{d\times d_k}$ are learnable projection matrices, where d_k denotes the dimension of the key vectors. The multi-head

attention mechanism enhances the model's ability to capture information from different subspaces by performing parallel computations on multiple attention heads and then concatenating the results:

$$MultiHead(Q,K,V) = Concat(head_1,\ldots,head_h)W^O \tag{2}$$

$$head_i = Attention(HW_Q^i, HW_K^i, HW_V^i) \tag{3}$$

The Transformer architecture includes encoder-decoder structures (such as the original Transformer), encoder-only structures (such as BERT), and decoder-only structures (such as the GPT series). In particular, the GPT series employs causal self-attention masking to ensure that predictions depend only on historical information:

$$MaskedAttention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}} + M)V \tag{4}$$

Here, the mask matrix $M_{ij} = 0$ when $i \geq j$, otherwise $M_{ij} = -\infty$. This architectural design makes it particularly suitable for generative tasks, and in the field of CFD, it can effectively handle sequential prediction problems such as flow field sequence generation and turbulence modeling.

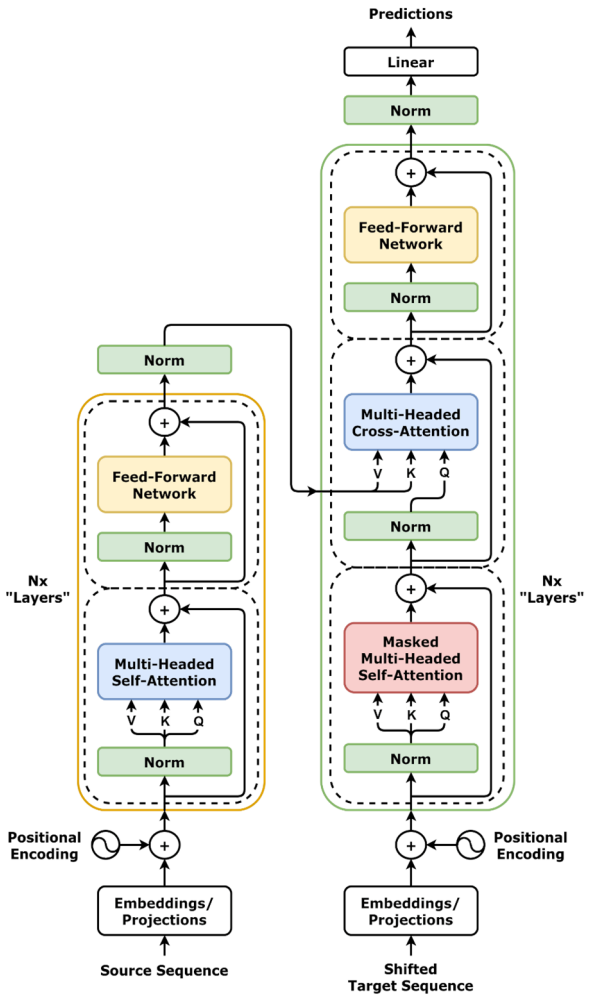

Figure 1. Diagram of the Transformer architecture. Image from Ref. [56].

2.2. Prompt Engineering

The aforementioned Transformer architecture lays the core foundation for LLMs, while the development of prompt engineering further enhances the model's capabilities in practical applications. The development of prompt engineering has simultaneously enhanced the capabilities of large language models. Initially, zero-shot prompting or few-shot prompting relied on carefully designed prompts or providing a small number of similar input-output examples to guide the model in completing tasks. Its core mathematical expression can be represented as:

$$\hat{y} = f_{\mathrm{LLM}}(x;\varepsilon) \tag{5}$$

In this context, x represents the input prompt, ε denotes the set of few-shot examples, and f_{LLM} stands for the mapping function of the large language model.

Park et al. [57] created a chemical markup language plugin, a specialized domain-specific language (DSL) tool that bridges the gap between prompt engineering theory and real-world use. Here, DSL is a programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain. Its declarative syntax streamlines how experimental documentation is handled. The plugin later expanded to incorporate logic and reasoning techniques like chain of thought (CoT) [58] and automatic chain of thought for LLMs. Unlike conventional prompting, these methods steer models toward step-by-step reasoning, producing responses that are not only more organized but also more insightful. The underlying mathematical framework is expressed as:

$$\hat{y} = f_{LLM}(x, z_1, z_2, \ldots, z_k) = \prod_{t=1}^{k+1} P(z_t \mid x, z_{<t}) \tag{6}$$

Here, $z_{k+1} = y$ is the final output, while $z_{<t} = \{z_1, \ldots, z_{t-1}\}$ denotes the sequence of reasoning steps leading up to it.

Self-consistency [59] addresses complex reasoning tasks with multiple valid paths by sampling diverse reasoning chains from the language model's decoder, then determining the most consistent final answer by marginalizing these sampled chains. Its core mathematical expression can be represented as:

$$\hat{y} = arg\ \max_{y}\ \sum_{i:y^{(i)}=y} P(z^{(i)} \mid x) \tag{7}$$

In this context, $z^{(i)}$ represents the i-th full step-by-step reasoning sequence.

Beyond these, innovative reasoning techniques like Tree of Thoughts [60], Graph of Thought, S2A Prompt, Threads of Thought, and Chain-of-Table Prompting have emerged. These approaches boost the logical reasoning and adaptability of LLMs, opening up avenues for applying prompt engineering to computational fluid dynamics in the future [61].

2.3. Fine-Tuning and RAG (Retrieval-Augmented Generation)

In addition to prompt engineering, fine-tuning and retrieval-augmented generation (RAG) techniques are also important means to optimize the performance of LLMs in specific domains. Fine-tuning is a powerful technique in AI that tailors pre-trained models for specialized tasks or fields, boosting their performance in real-world applications [62,63]. The idea is simple: tweak the model's parameters to minimize errors on a specific task. Starting with pre-trained parameters θ_0, the process aims to find the best-adjusted parameters θ^*, defined by the equation:

$$\theta^* = arg\ \min_{\theta} \mathcal{L}_{\text{task}}(\theta) = arg\ \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{CE}}\,(y_i, f_\theta(x_i)) \tag{8}$$

Here, $\mathcal{L}_{\text{CE}}$ measures prediction errors (cross-entropy loss), x_i is the input data, y_i its label, and f_θ the model's output.

LoRA [64]employs low-rank decomposition to modify weight matrices, drastically cutting down the trainable parameters needed. Given a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, The update form of LoRA is:

$$h = W_0 x + \Delta W x = W_0 x + \frac{a}{r} BAx \tag{9}$$

In this context, $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, $\frac{a}{r}$ is the scaling factor, and $r \ll \min(d, k)$. This represents a general framework for parameter-efficient fine-tuning:

$$h = W_0 x + \Delta W x \tag{10}$$

Here, $\Delta W = BA$. Through fine-tuning, researchers and engineers can further train and optimize these pre-trained models with relatively small amounts of domain-specific data, enabling them to efficiently and accurately solve specific engineering problems. This approach holds significant potential for development in the field of computational fluid dynamics.

LLMs may generate errors or "hallucinations", and minimizing these inaccuracies is crucial for reliable outputs. One effective approach is RAG [65], a technique that blends information retrieval with text generation to enhance output quality. Unlike traditional models that depend only on pre-trained knowledge, RAG integrates external data sources, overcoming the constraints of purely generative systems. The fundamental principle and architecture of RAG [66] typically involve a two-step process: retrieving relevant information based on the input query and generating text based on the query and the retrieved knowledge. Its mathematical expression is:

Retrieval phase: Given a query q, retrieve relevant documents from the document collection $\mathcal{D}$:

$$\mathcal{R}(q) = \{d \in \mathcal{D} \mid sim(E_q(q), E_d(d)) > \tau\} \quad (11)$$

Here, E_q and E_d are the encoders for the query and document respectively, $\mathrm{sim}(\cdot,\cdot)$ is the similarity function (e.g., cosine similarity), and τ is the similarity threshold.

Answer generation: Creating responses using queries and retrieved documents:

$$P(y \mid q, \mathcal{D}) = \sum_{d \in \mathcal{R}(q)} P(d \mid q) P(y \mid q, d) \quad (12)$$

Beyond RAG, techniques like Chain of Verification [67], Chain of Notes [68], Chain of Knowledge [69], and the react prompting method [70] help minimize hallucinations in LLMs.

3. How Large Language Models Are Revolutionizing CFD

3.1. LLM-assisted Predictive Modeling and Closure-Model Discovery

3.1.1. Modeling and Closure Model Discovery Using Prompt-Based Methods

Traditional tabular modeling methods generally operate on numerical feature values alone, neglecting column names, parameter semantics, and contextual information, thereby constraining their effectiveness in complex design spaces [71]. The paradigm of "new tasks without extensive training" has been widely adopted, as it does not depend on complex formats or code but instead relies on zero-shot text inputs or few-shot examples to convey instructions to AI models [72]. This approach lowers the barrier to entry, enabling even users without much programming expertise to accomplish tasks by interacting with AI models. Common natural language prompt types used across various industries include instructional, question-and-answer, role-playing, and zero-shot/few-shot prompts [73,74]. Users can directly provide instructions to LLMs or frame requests as questions, and they can also offer relevant modeling examples as references to assist researchers in developing feasible models. For instance, Zhang et al. proposed 'AutoTurb', a new framework that employs LLMs to autonomously identify algebraic corrections for the linear Reynolds stress model based on prompt strategies [52]. This framework utilized LLMs to create sub-expressions, enhancing exploration and diversity in evolutionary searches. The derived algebraic Reynolds stress model can be easily integrated into existing CFD codes and yields turbulence models with improved interpretability. Zhu et al. [75] prompted LLMs by structuring spatiotemporal information of the flow field, providing initial time-step state sequences of varying lengths as example prompts in contextual learning or few-shot prompts in few-shot learning scenarios, aiding LLMs in making physically plausible flow field predictions. Dinh et al. [76] designed a framework called μ-Fluidic-LLMs, as shown in Figure 2. The core of this method lies in text serialisation. It extracts the key parameters from tables, constructs contextual information, and converts these data into a processable serialized text format. This serialisation not only preserves the original numerical features but also incorporates column names and their semantic information, thereby providing the model with richer contextual information. The serialised text data is then fed into a pre-trained LLM to generate embeddings for input into machine learning models. Reported results showed that this method improves prediction accuracy and model generalization. When a DNN (Deep Neural Network) is combined with GPT-2, the prediction errors for droplet diameter and generation rate are reduced by approximately a factor of five and seven, respectively, compared to traditional ML models such as SVM and XGBoost, while flow pattern classification accuracy increases by over 4%. Additionally, the approach of integrating deep learning with large language models for microfluidic components opens new avenues for extending this approach to broader fluid dynamics applications. Looking ahead, emerging LLMs could potentially be combined for flow field prediction.

Unlike CNNs or PINNs that operate directly on numerical fields [77], LLMs excel in integrating heterogeneous inputs (e.g., natural language instructions, metadata, historical design logs), enabling higher-level abstraction and cross-modal reasoning. It should be clarified that in the μ-Fluidic-LLMs framework, the LLM serves not merely as a sophisticated text encoder for tabular data, but actively leverages its generative and reasoning capabilities to interpret parameter semantics, understand contextual relationships between variables, and generate meaningful predictions that go beyond simple numerical pattern recognition. This distinguishes it from conventional deep learning models like PINNs or CNNs, which primarily focus on learning numerical mappings without explicit semantic understanding.

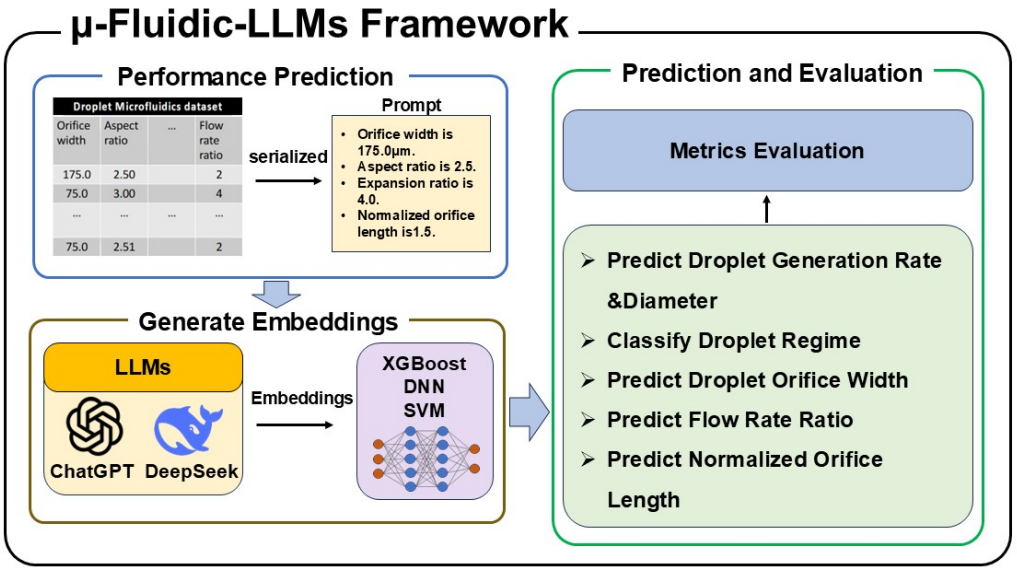

Figure 2. Workflow of the μ-Fluidic-LLMs framework. Adapted from Ref. [76].

Although natural language prompting methods have demonstrated advantages in CFD, certain limitations remain. For example, the manual dependence and the scalability of text serialization are among the major challenges [78,79], i.e., manually or semi-automatically converting large volumes of tabular data into natural language text requires additional processing and verification steps, which may introduce additional time and labor costs in large-scale industrial data scenarios. Moreover, choosing the right model and tuning hyperparameters still demands rigorous testing [80,81]. Different LLMs exhibit markedly different performance on specific tasks, often requiring iterative testing with downstream algorithms to optimize parameters. This process adds additional complexity to both research and real-world use. To overcome the shortcomings of natural language prompts, researchers are exploring techniques like text serialization [81], regularization [82], dataset expansion [83], and advanced optimization algorithms [84]. These approaches could boost efficiency and accuracy while minimizing errors. Another promising direction is refining prompt engineering to help LLMs better capture user intent. In technical fields like CFD, where precision and logical reasoning are critical, Chain-of-Thought (CoT) [58] prompting has shown potential. The key idea behind CoT is to mimic human-like step-by-step problem-solving, breaking down intricate challenges into smaller, logically connected steps. This approach not only boosts the accuracy of LLMs in multi-step reasoning tasks but also makes their outputs easier to understand. Its potential is particularly notable when applied to LLMs working in fluid mechanics. Beyond CoT, innovative prompting techniques like Graph of Thought [85], S2A prompts [86], Thread of Thought [87], and Chain-of-Table prompting [88] provide additional mechanisms for structuring and guiding computational reasoning in fluid mechanics.

3.1.2. Fine-Tuning AI Models for Enhanced CFD Predictions

Recently, models using the Transformer-based architecture have been used to simulate transient solid-liquid flow in stirred tanks [89] as well as the identification of multiphase flow regimes [90]. Fine-tuning and pre-training of models are also key methods to effectively leverage LLMs [91,92]. Pre-trained models learn broad general knowledge and features by training on massive datasets [93]. Fine-tuning these models has shown promise in fluid mechanics [94–96]. By adapting pre-trained AI models with just a small amount of problem-specific data, researchers can boost the model's performance for targeted fluid dynamics applications [97].

Fine-tuning large models specifically for CFD has emerged as an effective approach to solving CFD problems. Table 1 summarizes the use of large language models to assist in fluid dynamics modeling, flow field prediction, and manifold recognition. Dong et al. developed a domain-specific LLM [98] to automate computational fluid dynamics simulations. As shown in Figure 3, the model was trained on a custom-built OpenFOAM configuration dataset NL2FOAM and incorporated CoT annotations. By fine-tuning Qwen2.5-7B-Instruct, the model can directly convert natural language descriptions into executable CFD setups. A multi-agent system coordinates the entire process, autonomously validating inputs, generating configurations, running simulations, and correcting errors. The fine-tuning was performed using the Low-Rank Adaptation (LoRA) on the pre-constructed dataset, reducing computational costs while maintaining comparable performance. The fine-tuned model's performance in predicting physical quantities was also evaluated against general-purpose LLMs, in cases such as vorticity magnitude in cylinder wake, jet flow velocity components, and square bend velocity components.

Compared to open-source general models, the fine-tuned large models have achieved improved performance in both solution quality and operational efficiency [98]. Regarding the specialized CFD datasets for fine-tuning LLMs, high-quality datasets should include: (1) diverse flow scenarios with well-defined boundary conditions and physical parameters; (2) paired natural language descriptions with corresponding simulation setups/results; (3) expert annotations explaining the physical principles behind model selections. For example, both NL2FOAM [98] and FLUID-GPT [99] studies confirmed that incorporating industrial-grade complex cases significantly enhances model adaptability: the former constructs an instruction dataset covering real geometric topologies, variable boundary conditions, and solver configurations, enabling the model to generate OpenFOAM code suitable for complex engineering scenarios, breaking through the limitations of "toy cases" caused by synthetic data. The latter leverages high-fidelity CFD simulation-generated particle trajectories and erosion data, allowing the model to effectively capture nonlinear physical features such as unsteady and multiphase flows. To sum up, incorporating industrial-grade data encompassing extreme conditions, complex geometries, and real physical constraints into the fine-tuning process is a necessary approach to enhance the robustness and engineering applicability of large language models in CFD tasks.

Table 1. Intelligent-driven fluid dynamics modeling and prediction.

New Framework	Effect	Modeling and Prediction Results	Applicable CFD Scenarios	Pros and Cons
AI-Augmented Turbulence and Aerodynamic Modeling [100]	70% acceleration in CFD prediction, 96% prediction accuracy	Turbulence structure prediction, drag coefficient optimization, lift-to-drag ratio improvement	Aerodynamic optimization, aerospace design	Benefits: Faster processing without sacrificing precision Limitation: The model depends on high-quality CFD data during training
Fine-tuning LLM for CFD Automation [98]	88.7% accuracy, 82.6% first-attempt success rate	Translating natural language descriptions into OpenFOAM configurations, supporting 21 flow scenarios	CFD configuration automation, simulation process optimization	Benefits: High accuracy, low computational cost Limitations: Only works for incompressible flows and demands extensive training data
Autonomous Droplet Microfluidic Design Framework [76]	Design time reduced by 65%, design success rate 90%	Automated microfluidic device design and optimization supporting multiple droplet generation modes	Microfluidic system design, droplet generation optimization	Benefits: Cuts down design time and boosts efficiency Limitations: Performance could be constrained in advanced microfluidic applications
FLUID-LLM: Spatiotemporal-aware LLM for CFD [75]	Flow field prediction accuracy 92.3%, computation speedup 30%	Capturing spatiotemporal dependencies, predicting fluid behavior and turbulence characteristics	Spatiotemporal flow field prediction, turbulence simulation	Benefits: Models spatiotemporal relationships for more accurate forecasts Limitations: Demand extensive spatiotemporal data and come with significant computational expenses

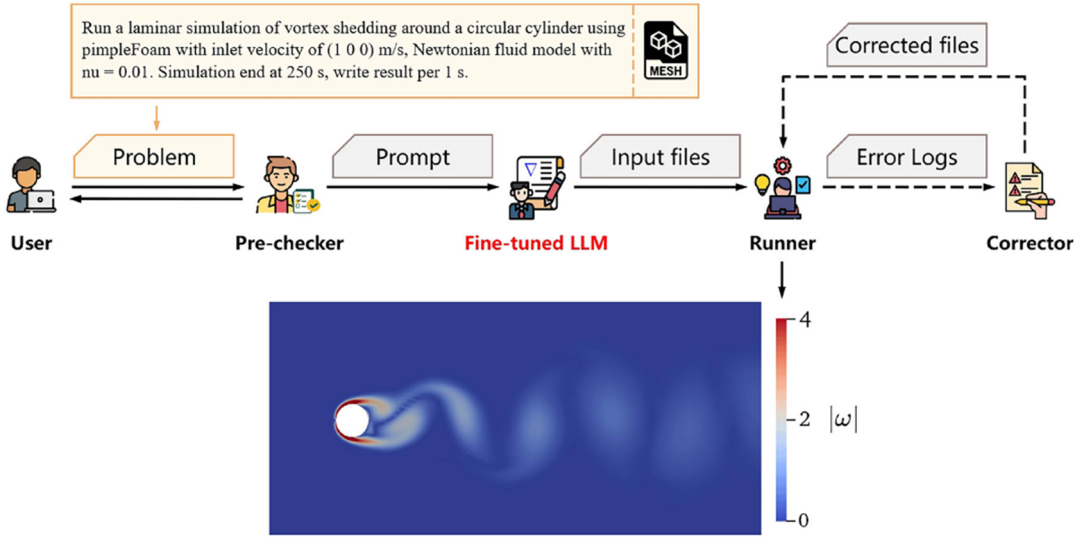

Figure 3. The process of autonomous verification of input, generation of configuration, running simulation, and error correction in a multi-agent system. Adapted from Ref. [98].

Compared to the baseline RAG approach, the above approach combines CoT annotations to enhance the reasoning capabilities of LLMs in specific domains. In a benchmark test composed of 21 different flow cases, this method demonstrated competitive performance, achieving a solution accuracy of 88.7% and a one-pass success rate as high as 82.6%. A comparison was also made between scenarios with and without chain-of-thought reasoning, revealing that models with chain-of-thought derivation exhibit improved accuracy and one-pass rates. Compared to conventional general-purpose models, this method generates parameter values with greater diversity and flexibility, achieving higher performance while requiring fewer error-correction iterations, higher computational efficiency, but at lower costs. Sahibzada et al. [100] proposed a "learn data first, then learn physics" approach for fine-tuning large language models. Although PINNs are utilized here instead of LLMs, the fine-tuning techniques in this method are still worth noting. First, a neural network was trained with high-resolution CFD data, enabling the model to learn fundamental patterns of flow fields (e.g., velocity fields, pressure distribution) and establish preliminary turbulence prediction capabilities. In the second step, a physics-informed loss function (combining data-driven loss and governing equation residuals) was introduced. By minimizing this function, the model retains the outcomes of data learning while enforcing consistency with the fundamental laws of fluid mechanics, thereby improving the physical consistency of predictions. The artificial intelligence model architecture here is based on PINNs, in which the Navier–Stokes equations, continuity equation, and turbulence transport equations are incorporated through physics-based constraints within the learning process. This approach helps ensure adherence to physical principles and can improve computational efficiency in certain modeling settings. After training, the AI model was integrated with traditional CFD solvers, forming an integrated framework. This innovative framework reduces the runtime of high-fidelity CFD simulations like LES from 48 h down to 7 h, an 85% time saving (or ×6.8 speed boost) while delivering nearly identical accuracy to LES (96% precision) [100]. Table 2 highlights how various AI-driven frameworks approach fluid dynamics modeling and prediction, comparing their key features.

Table 2. AI-powered CFD optimization techniques using LLMs.

Type of Optimization	Optimization Objective	Methods	Effects	Compared to Other Methods
Optimization of geometric parameters [53]	The geometric designs of aircraft wings, rocket nozzles, and industrial equipment like fluidized beds and bubble columns.	LLM-PSO framework, natural language prompts + iterative feedback, multi-objective balancing strategy	This method converges almost ten times faster than traditional reinforcement learning, efficiently navigating complex high-dimensional parameter spaces while striking a balance between competing objectives.	Lower computational costs, stronger capability in handling high-dimensional problems, and no need to convert multi-objective optimization into single-objective problems
Hyperparameter optimization for machine/deep learning models [101]	Hyperparameter configuration of deep/machine learning models	AgentHPO paradigm, LLM + Bayesian optimization, fine-tuning Code Llama to generate hyperparameter recommendations	Dynamic search space adaptation, generating accurate and efficient parameters for different network architectures, improving interpretability and user trust	High degree of automation, reduced manual intervention, simplified setup process, suitable for complex fluid dynamics models
AutoTurb: Automatic Algebraic Turbulence Model Discovery [52]	Automatic discovery of turbulence model structures and parameters	AutoTurb framework, LLM-driven symbolic regression multiscale verification mechanism, combined with evolutionary algorithm to constrain complexity	Breaking the dependence of traditional symbolic regression on predefined function libraries, capturing more fundamental physical laws, and leaping from "parameter optimization" to "model structure discovery"	Cutting labor expenses, combining first-principles modeling with data-driven approaches, delivering transparent mathematical formulas, and excelling in diverse real-world applications

Although fine-tuning techniques can improve the modeling efficiency and prediction accuracy of LLMs in fluid dynamics tasks, its process itself still faces challenges and limitations that urgently need to be addressed. First, fine-tuning heavily relies on high-quality, large-scale domain-specific annotated datasets, which are costly to acquire and clean, and their quality directly impacts model performance [102]. Second, fine-tuned models still exhibit black-box characteristics to some extent, making it difficult to trace the influence of weight updates on physical decision-making, resulting in insufficient interpretability of key outputs such as AI-generated turbulence models [103]. Additionally, under complex flow conditions, the decision logic of fine-tuned models is challenging to interpret and validate for physical consistency, possibly limiting their credibility in engineering practice [104,105]. At the technical strategy level, the applicability and performance trade-offs of different fine-tuning methods, such

as full-parameter fine-tuning, LoRA, and Adapter, remain unclear. Moreover, existing fine-tuning algorithms incur high computational costs, and data selection and sample optimization strategies require further refinement [106]. Future research should integrate explainable AI (XAI) techniques into the fine-tuning framework to enhance the transparency and trustworthiness of weight updates; systematically compare the applicability boundaries of full-parameter fine-tuning and parameter-efficient fine-tuning (PEFT) methods, develop intelligent data selection and curriculum learning strategies to reduce the need for annotated samples, and explore novel optimization algorithms and regularization techniques to lower computational costs while ensuring physical consistency constraints, thereby promoting the reliable deployment and widespread application of fine-tuning techniques in the CFD field.

3.2. Harnessing Large Language Models for Optimization

In traditional optimization methods, genetic algorithms search for optimal solutions by simulating processes such as natural selection, gene crossover, and mutation. As mentioned earlier, optimization problems in fluid dynamics can be divided into: optimization of numerical parameters in simulations, optimization of hyperparameters in AI models, and optimization of geometric parameters and operating conditions of fluid equipment. For example, fine-tuning the 3D blade geometry of a centrifugal pump impeller [107]. Neural network algorithms [108] can automatically adjust key settings such as time step size and the number of neurons in hidden layers. They have also been applied to XGBoost classifiers, achieving measurable performance improvements by optimizing multiple hyperparameters, including the learning rate [109].

Bayesian inference is a global optimization method based on Bayes' theorem. It approximates the objective function by constructing a probabilistic surrogate model and selects the next most "promising" evaluation point based on an acquisition function. It can be used to optimize the window area of buildings to improve ventilation performance [110]. Bayesian optimization [111] can also be applied to tune hyperparameters such as the learning rate to obtain optimal solutions with limited iterations. For LSTM neural network optimization [112], Bayesian optimization can be used to adjust hyperparameters such as the number of hidden layers and the number of neurons per layer, improving model prediction accuracy with RMSE or MAE as the objective function.

Traditional optimization methods such as genetic algorithms and Bayesian optimization, although effective in fluid dynamics parameter optimization and machine learning hyperparameter tuning, increasingly reveal their limitations as problem complexity grows. Large language model (LLM)-driven optimization techniques offer new potential to address these challenges, leveraging advantages such as dynamic kernel function design, efficient handling of high-dimensional and discrete parameters, and potential improvements in interpretability, thereby providing opportunities to overcome the shortcomings of traditional methods [113,114]. Table 2 lists several typical examples of the main applications of LLMs in optimization strategies.

3.2.1. Fine-Tuning Geometric Parameters with LLMs

When facing complex shape optimization problems, traditional genetic algorithms have been widely applied to the geometric design optimization of fluid-related components, such as aircraft wings, rocket nozzles, as well as process engineering equipment like fluidized beds, fixed beds, and bubble columns. For optimizing aircraft wings, these methods search for the optimal configuration with minimal drag or maximal lift in the design space by adjusting geometric shape parameters. For example, parameterizing blade geometry using Bézier curves and combining CFD with genetic algorithms can yield improved design solutions [115,116]. Yet genetic algorithms have drawbacks—they converge relatively slowly, struggle with complex high-dimensional problems, and require expert intervention to fine-tune parameters. To address these limitations, LLMs have been explored as a supplement, providing a possible way to overcome these limitations. Zhang et al. [53] proposed an LLM-PSO framework for parametric shape optimization in fluid mechanics, as shown in Figure 4. Drawing inspiration from evolutionary strategies, the framework utilizes a LLM to determine the optimal shape for parameterized engineering design, achieving more efficient and intelligent design optimization in fields such as fluid dynamics. It also provides a generalizable method applicable to shape parameter optimization in fluid mechanics. Specifically, researchers first provide natural language prompts to the LLM, specifying the optimization objective, explaining boundary conditions, and displaying existing optimization records. The LLM then generates the next generation of shape parameters based on this information. Next, the researchers submit these parameters to a solver for computation and generate performance evaluation data, which is recorded and resubmitted to the LLM to initiate a new iteration. In comparisons with existing optimization algorithms, LLM-PSO demonstrates faster convergence. In 2D airfoil optimization, its convergence speed is nearly an order of magnitude faster than reinforcement learning algorithms [53]. Moreover, in 3D axisymmetric body optimization, LLM-PSO shows improved early-stage performance compared with genetic algorithms and achieves lower average drag. Compared

to genetic algorithms, the LLM-based approach reduces computational costs, accelerates convergence, and relies less on specialized expertise [53].

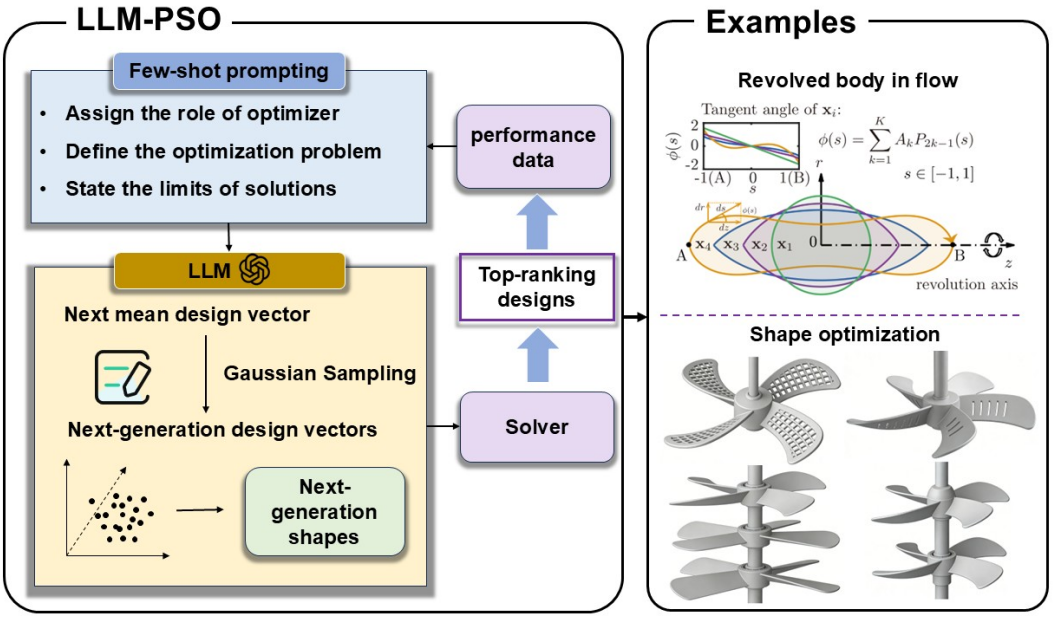

Figure 4. How the LLM-PSO framework works. Adapted from Ref. [53].

In addition, geometric parameters of process engineering equipment such as fluidized beds and bubble columns, including height, diameter, and the shape and position of internal structures (e.g., baffles, distributors), directly affect key characteristics like the two-phase mixing behaviors, mass transfer efficiency, fluidization state of the bed, and particle residence time. LLMs can understand objectives such as mass transfer efficiency and pressure drop through natural language processing and generate corresponding combinations of geometric parameters. The optimization case of the dual circulating fluidized bed solar reactor mentioned by Milanese et al. [117] illustrates that LLMs can generate optimized geometric parameter combinations based on factors such as particle circulation rate and reactor dimensions, thereby improving overall efficiency. Compared to genetic algorithms, LLMs are more effective at exploring high-dimensional parameter spaces and avoiding the curse of dimensionality.

Multi-objective optimization represents a promising capability of LLMs. Fluid shape optimization often involves multiple conflicting objectives, and LLMs can be prompted to account for multiple objectives simultaneously and generate parameter combinations that balance them. Liu et al. proposed a large language model-based multi-objective evolutionary algorithm [118], which uses prompt engineering to employ the LLM as a black-box search component within the optimization loop to directly handle multi-objective optimization problems, achieving performance comparable to traditional MOEAs. This capability is particularly beneficial for optimizing geometric parameters of process equipment such as fluidized beds and bubble columns, helping to achieve reasonable trade-offs among multiple competing performance metrics.

3.2.2. How LLMs Fine-Tune Hyperparameters in AI Models

The hyperparameter settings and tuning of machine learning models have a decisive impact on the overall prediction accuracy and generalization performance of the models. However, traditional hyperparameter optimization methods generally suffer from relatively low efficiency, complex settings, and lower interoperability [101]. Take heuristic tuning for example, it tends to be less automated and delivers underwhelming results, particularly with complex algorithms. Enter LLM agents: these AI-driven tools automate hyperparameter tuning, boosting algorithm performance [119]. Zhang et al. [120] introduced a workaround—using dataset and model descriptions as prompts to let LLMs suggest initial hyperparameter settings, then refining them iteratively based on performance feedback. Even with tight computational limits, LLMs rival or surpass conventional methods like random search and Bayesian optimization in benchmark tests. This workaround not only speeds up tuning but also cuts down on the need for human tinkering. Liu et al. [101] introduced a novel paradigm named AgentHPO, as shown in Figure 5. This paradigm employed LLM agents to autonomously process task information, conduct experiments targeting specific hyperparameters, and iteratively optimize parameters based on historical trial results. For example, when optimizing hyperparameters for a multilayer perceptron (MLP) model, AgentHPO

achieved performance comparable to traditional HPO tools like Optuna and HyperOpt in 50 trials, but with only 25 trials, demonstrating its advantage in experimental efficiency. Building on earlier methods, Mahammadli and Ertekin [121] introduced the SLLMBO framework, a hybrid approach that merges LLMs with Tree-structured Parzen Estimator sampling. This innovation allows dynamic adaptation of search spaces and deeper exploration of parameter configurations, addressing key shortcomings in both standalone LLM and conventional Bayesian optimization techniques. Separately, Kochnev et al. [122] tested a fine-tuned Code Llama model for hyperparameter tuning. By applying parameter-efficient adaptation methods, their modified LLM could suggest precise hyperparameters for diverse neural network designs, outperforming traditional approaches in both accuracy and efficiency. To sum up, such intelligent agent models could reduce the number of experiments, simplify the setup process, and enhance interpretability and user trust, providing a feasible solution for optimizing hyperparameters in complex domains such as fluid dynamics models.

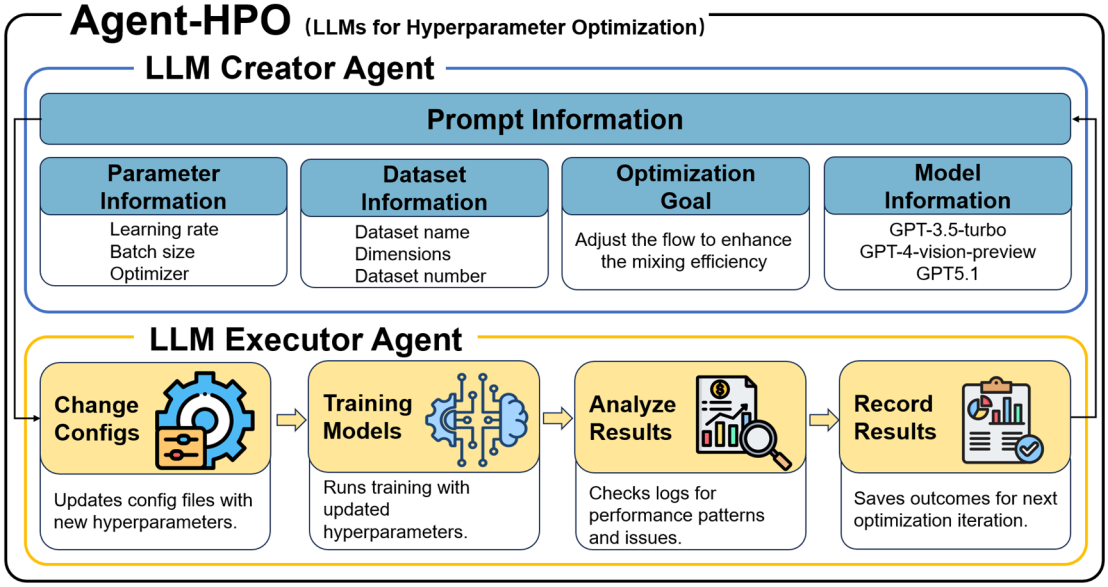

Figure 5. How the AgentHPO framework operates. Adapted from Ref. [101].

3.2.3. Using Large Language Models to Improve CFD Simulations

Beyond tuning hyperparameters and shape parameters, LLMs can also refine numerical model components like turbulence models. Optimization of flow-field dynamics has benefited from advances in deep learning, as illustrated by the automated engineering design system proposed by Chen et al. [123]. They combined deep neural networks with active learning. But a closer look reveals these methods still operate within a "parameter space": they tweak preset model configurations rather than reimagining the physics. While the model reduces the required data samples from 8000 to 625 compared to traditional approaches, their black-box design possibly limits physical interpretability. Without explicit mathematical representation, these models struggle to adapt when flow conditions or geometries shift significantly. As a result, these limitations restrict reliability in engineering applications and impede deeper understanding of turbulence mechanisms. In contrast, Zhang et al. [52] introduced the "AutoTurb" framework, which introduces a fundamentally different approach, shifting from tweaking parameters to autonomously uncovering model architectures, as illustrated in Figure 6. By harnessing large language models, this approach overcomes the limitations of conventional symbolic regression, which relies on predefined mathematical functions and operators, elevating the process from mere "data fitting" to true "knowledge creation." What sets AutoTurb apart is its dual focus: it not only adjusts Reynolds stress directly but also uses RANS outputs (like velocity fields) as training benchmarks. This multi-layered validation improves mathematical consistency and physical interpretability of the results. The symbolic expressions generated by AutoTurb are not only mathematically executable but also human-readable, enabling engineers to inspect, verify, and optimize the discovered turbulence closure terms. The symbolic expressions generated by AutoTurb are not only mathematically executable but also human-readable, enabling engineers to inspect, verify, and optimize the discovered turbulence closure terms. This stands in stark contrast to the opaque weight matrices of standard neural networks. While PINNs embed physical knowledge through loss functions, their internal representations remain uninterpretable; in contrast, AutoTurb produces explicit algebraic forms that can be integrated into traditional solvers and subjected to dimensional analysis and physical plausibility checks.

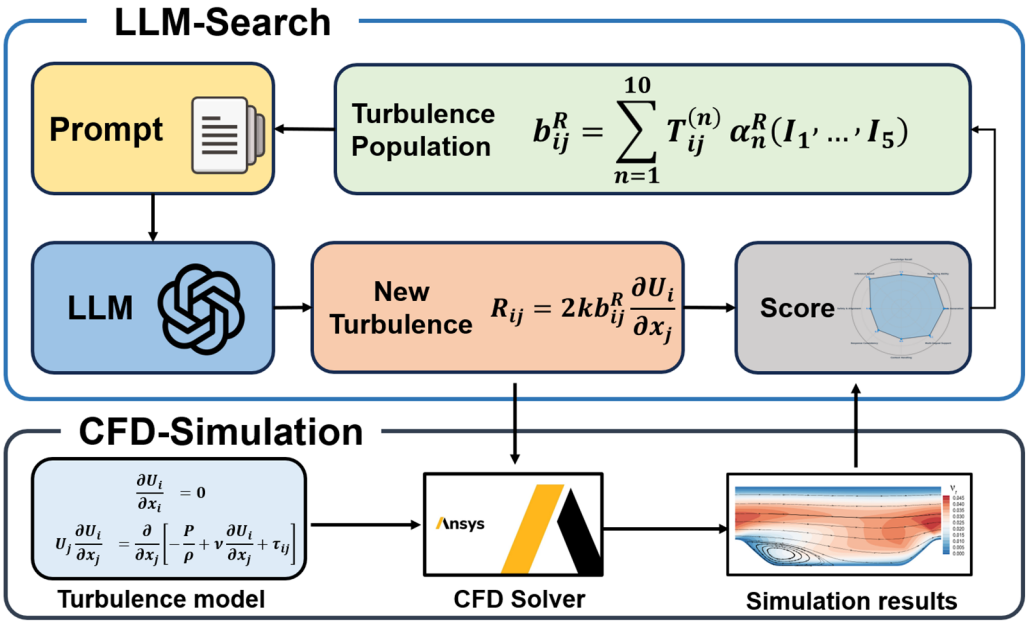

Figure 6. The AutoTurb framework. Adapted from Ref. [52].

The AutoTurb framework's ability to generate symbolic expressions represents a significant advancement in terms of scientific interpretability compared to standard black-box neural networks. While traditional neural networks performing turbulence modeling would output numerical predictions without revealing the underlying mathematical relationships, AutoTurb produces human-readable algebraic equations that can be directly analyzed, validated against physical principles, and integrated into existing CFD workflows. This transparency not only builds trust among domain experts but also enables iterative refinement based on physical insights, addressing a core limitation of conventional AI approaches in scientific computing.

Moreover, the AutoTurb framework reveals a new path for the integration of AI and physical sciences: LLMs do not replace traditional CFD methods but can support model development by assisting in the exploration and formulation of data-informed closures for complex flows. By integrating evolutionary algorithms and constraining function complexity and numerical convergence, this method achieves a balance between expressive power and physical constraints, avoiding overly complex models while retaining sufficient innovative potential. In validation, the models discovered by AutoTurb successfully generalized to flow cases at different Reynolds numbers and geometries. This advancement suggests that future scientific computing may no longer strictly distinguish between "first-principles" and "data-driven" methods but instead move toward a new paradigm of fusion and complementarity, where AI serves as an "extension of scientific intuition", helping scientists break through cognitive boundaries and discover physical laws hidden behind complex data. By leveraging LLMs to generate parameters and refining them through conventional solvers, it delivers better results than traditional methods. Unlike standard neural networks, it cuts down on manual effort, streamlines optimization, and introduces a collaborative performance review process between AI-agent and human experts. However, some gaps remain, such as unexplored hyperparameters and untested designs. Future research is suggested to focus on integrating advanced algorithms with LLM to boost its performance, or automating multi-physics modeling.

3.3. Automating CFD with Large Language Models

In addition to developing specialized LLMs for the field of fluid dynamics, general-purpose open-source models can also be integrated with CFD software to automate the computational workflow. This enables the automatic generation of executable CFD configurations, which can then be seamlessly input into solvers to complete numerical computation tasks. Integrating open-source LLMs with CFD solvers is currently an efficient approach to solving complex fluid dynamics problems [48]. This method not only simplifies the modeling and parameterization processes in traditional CFD problems but also enhances the prediction accuracy and solving efficiency of fluid dynamics issues by leveraging the capabilities of LLMs in natural language understanding and complex reasoning [124]. Simultaneously, incorporating prompt engineering, fine-tuning, and RAG methods into a new intelligent agent focused on fluid dynamics research enables efficient CFD automation. To contextualize these developments, Table 3 provides an overview of the different CFD agents and highlights the key distinctions among their frameworks.

Table 3. Comparison of CFD agent frameworks.

Name of the Framework	CoT	RAG	Preprocessing	Performance
OpenFOAMGPT	Not mentioned	Yes	Yes	Capable of handling complex simulations, and identifying geometric types to generate blockMesh
OpenFOAMGPT 2.0	Not mentioned	Yes	Yes	The system can not only respond to user queries, but also automatically complete the entire processing chain, and can smoothly generate all necessary configuration files.
ChatCFD	Not mentioned	Yes	Yes	The system can efficiently run various turbulence models, performing comparably to DeepSeek-R1 in terms of computational resource consumption and cost control.
MetaOpenFOAM	Not mentioned	Yes	Yes	The incorporation of RAG technology enables the system to both improve task completion efficiency and reduce the need for manual intervention.
MetaOpenFOAM 2.0	QDCOT, ICOT	Yes	Yes	Performance is improved through COT and RAG, with additional post-processing capabilities such as data processing and visualization.
OptMetaOpenFOAM	QDCOT, ICOT	Yes	Yes	It can be used for sensitivity analysis and parameter optimization.
FLUID-GPT	Not mentioned	Not mentioned	Not mentioned	Lower MSE loss, higher R^2, faster convergence speed can be achieved, and it can also be used to predict particle trajectories.
AutoCFD	Not mentioned	Yes	Yes	Solution accuracy: 88.7% First-attempt success rate: 82.6% Average correction iterations: 2.6 times

In particular, Chu's group [63] proposed OpenFOAMGPT, an excellent example of an AI assistant powered by an LLM enhanced with RAG. This specialized tool is designed for OpenFOAM-based CFD simulations and features a chain-of-thought reasoning capability to improve its problem-solving approach. The system operates by first receiving a user query, which is combined with a system prompt and, when necessary, augmented using RAG before being passed to the large language model. The LLM then feeds the input to the OpenFOAM agent. If the detection system identifies a failure, the workflow repeats. This approach allows typical test scenarios to be completed with low token usage. Nevertheless, human supervision remains essential to ensure the accuracy of the engineering process. As large models evolve, future developments are suggested to reduce token costs and further enhance reasoning capabilities. Building on this work, Feng et al. [125] introduced OpenFOAMGPT 2.0. Compared to the previous single large-model architecture, the new framework incorporates preprocessing agents, prompt generation agents, and postprocessing agents, truly achieving end-to-end automation and delivering higher success rates and reproducibility. The new version accepts pre-generated mesh files, maintaining compatibility with professional meshing tools. Additionally, during postprocessing and result analysis, it automatically generates Python scripts for data processing and visualization, significantly lowering the barrier for postprocessing. Similarly, Chen et al. [55] also developed a CFD automation framework called MetaOpenFOAM based on LLMs using OpenFOAM, as shown in Figure 7. In this process, RAG technology was employed to search OpenFOAM tutorials and databases. The difference from OpenFOAMGPT lies in MetaOpenFOAM's approach of breaking down user queries into sub-questions and integrating a multi-agent system built on MetaGPT [126] with RAG technology based on Langchain21. Experimental validation across eight tests demonstrated an average pass rate of 85% and an executability score of 3.6, indicating good performance. Additionally, the cost was significantly lower than manual labor, reducing the barrier to entry and providing a feasible reference for CFD automation. To assess the contributions of the RAG module and the multi-agent system, an ablation study was conducted, revealing a drop in both average pass rate and executability. Following MetaOpenFOAM, MetaOpenFOAM 2.0 was subsequently released, expanding on the first generation by adding post-processing features like data processing and visualization, making the software more comprehensive. Given the complexity, high error rate, and specialized nature of CFD tasks, it is necessary to develop a dedicated CoT that integrates problem decomposition, iterative verification and optimization, and data augmentation to serve the CFD field. Version 2.0 incorporates multiple forms of CoT [127], including Question Decomposition CoT (QDCOT) [128], Iterative Verification and Optimization CoT (ICOT) [129], data augmentation [130], and sampling [59], enhancing the framework's logical reasoning capabilities and improving usability for non-experts. Using CoT to augment LLMs has become a primary method for post-training optimization [58].

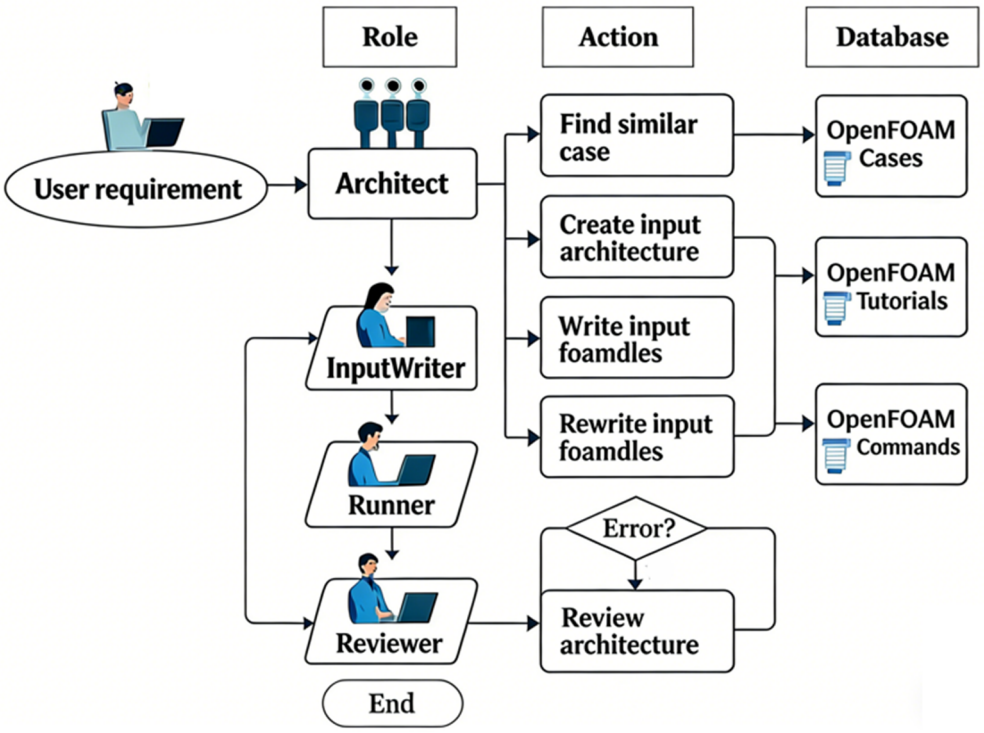

Figure 7. Overview of the MetaOpenFOAM Workflow Adapted from Ref. [55].

Beyond MetaOpenFOAM and OpenFOAMGPT, researchers have developed OptMetaOpenFOAM [131], an advanced version of MetaOpenFOAM 2.0 tailored for sensitivity studies and parameter optimization. This tool combines LLM-powered reasoning with external analysis tools to create a seamless, natural language-controlled CFD workflow. Another more recent agent is ChatCFD, a system that automates CFD processes from start to finish. Using the DeepSeek-R1 and DeepSeek-V3 language models, a multi-agent framework, and OpenFOAM expertise, ChatCFD can interpret diverse inputs like academic papers and mesh files. When tested on 205 standard cases, 110 modified scenarios, and 2 real-world examples, it achieved an 82.1% success rate, surpassing both MetaOpenFOAM and Foam-Agent [132] in head-to-head comparisons. Researchers also proposed FLUID-GPT [99], a data-driven surrogate model for automated prediction. It employs a hybrid deep learning architecture that leverages the self-attention mechanism and autoregressive capabilities of the GPT-2 Transformer model. With just five initial condition parameters (e.g., particle size, inlet velocity/pressure), it can efficiently and accurately predict complex particle trajectory time series (reducing error by 54% and speeding up training by 70% compared to traditional LSTM/BiLSTM models). The predicted trajectory data is then fed into a 3D CNN to automatically extract spatial features and generate the final surface erosion distribution map. This approach bypasses the tedious grid generation, iterative solving, and post-processing steps in traditional CFD through "end-to-end" machine learning inference, potentially enabling rapid automated prediction of particle erosion in industrial-scale equipment. The research also highlights how turbulence model choices and physical interactions affect outcomes, while noting the current constraints of LLMs in managing intricate CFD setups.

While advancements have been achieved, the CFD agent still grapples with challenges like limited open-source LLM options, inadequate incorporation of physical constraints, and the risk of LLM-produced configurations being physically unrealistic. For example, Yang et al. [133] proposed a method that can be used in conjunction with the latest LLMs (such as GPT-5, PaLM2, Gemini1.5, LLaMA3, etc.) to evaluate differences in overall performance. They combined the latest DeepSeek-R1 to establish a closed-loop, iterative workflow. The advantage of DeepSeek-R1 lies in its potentially stronger reasoning capabilities in physics and mathematics compared to commercial models like ChatGPT and Claude. Additionally, DeepSeek-R1 is open-source, offering greater freedom in prompt design and model deployment. Engineers use natural language to introduce their identity, problems, and requirements to the large model, then verify the multiple solutions provided and select the optimal one. During this process, engineers can continuously provide additional data or impose constraints, such as normalization, through interactive interfaces to further refine the model. Beyond integrating with cutting-edge open-source large models, a comparative validation mechanism is established to perform formal verification of relevant parameters, ensuring compliance with objective physical laws.

4. Conclusions and Future Perspectives

4.1. Conclusions

This review investigates, explores and analyzes current research progress, challenges, and opportunities of applying LLMs in CFD. The core technical foundation of LLMs is rooted in the Transformer architecture, whose self-attention mechanism efficiently captures long-range spatiotemporal dependencies in flow field sequences through dynamic weighting, providing a new computational paradigm for modeling complex fluid dynamics problems. The realization of this technical advantage relies on the deep integration of three synergistic methods, i.e., (i) Prompt Engineering guides LLMs in logical reasoning and problem decomposition through structured instruction design, (ii) fine-tuning techniques enable the model to deeply adapt to the professional knowledge base of the CFD domain (e.g., fluid mechanics equations, boundary condition specifications), and (iii) the retrieval-augmented generation method suppresses the "hallucination" phenomenon during model generation to some extent by integrating external knowledge sources such as the OpenFOAM documentation library and the open-source CFD database, thereby improving the physical credibility and engineering applicability of prediction results.

In the context of predictive modeling and closure model discovery, the LLM-aided intelligent prediction agents, such as a AutoTurb framework, leverage the symbolic reasoning capabilities of LLMs to automatically discover turbulence closure models, overcoming the inherent constraints of traditional symbolic regression methods on predefined function libraries (such as polynomial and exponential function sets), and achieving more flexible and physically self-consistent model construction. Furthermore, the method of fine-tuning LLMs combined with physical constraints (such as weak-form constraints of the Navier-Stokes equations) improves the accuracy of multiphase flow pattern classification, effectively addressing the physical consistency shortcomings of traditional data-driven models.

In the field of optimization, the LLM-driven optimization agents, such as LLM-PSO and AutoTurb, achieve intelligent collaborative optimization of geometric parameters (such as airfoil profiles, fluidized bed internal structures) and hyperparameters (such as neural network depth, XGBoost learning rate) through a natural language interaction interface. Compared to traditional genetic algorithms and Bayesian optimization, this kind of framework exhibits faster convergence while reducing reliance on CFD expert knowledge, enabling non-specialist users to efficiently complete complex optimization tasks.

At the level of automated execution, CFD frameworks based on RAG and multi-agent collaboration (such as OpenFOAMGPT and MetaOpenFOAM enable end-to-end automated generation of complete CFD configuration files from natural language instructions). The systems achieve this through division of labor among agents (such as parsing instructions and invoking solvers). It can be expected that advanced and improved versions of existing agents, and even more CFD agents, will be reported.

4.2. Opportunities and Challenges

Despite the advantages above, current research still faces challenges include but not limited to: (1) Potential lack of physical credibility may lead to boundary conditions or turbulence models violating fluid dynamics principles; (2) Insufficient integration of domain knowledge manifests as incomplete coverage of RAG knowledge bases and semantic retrieval mismatches (e.g., "cavity flow" being incorrectly retrieved as "airfoil flow"; boundary condition conflicts; turbulence model misuse); (3) Lack of standardized technical pathways results in irreproducible outcomes (due to mixed use of frameworks like prompt engineering/RAG/fine-tuning) and an absence of universal evaluation metrics; (4) Possible limitations in engineering applicability make it difficult for LLMs to handle practical demands such as 3D geometry generation or fluid-structure interaction; (5) Missing hyperparameter and model adaptation is evident in prompt templates failing for complex tasks and insufficient fine-tuning data (e.g., FLUID-GPT relying solely on particle trajectory data); (6) Absence of human-machine collaboration mechanisms forces manual intervention for verification (e.g., "iterative validation" in MetaOpenFOAM 2.0) and prevents autonomous solver invocation. (7) The issue of hallucinations still persists to some extent, such as fabricating non-existent numerical methods or parameters, leading to unreliable simulation results. These challenges fundamentally stem from the conflict between LLMs' general language capabilities and CFD's specialized physical constraints, directly hindering their reliable application across the entire engineering workflow. These challenges urgently require the construction of physics-embedded RAG, standardized evaluation systems, and human intelligence-AI (HI-AI) collaborative workflows. Future research may focus on the following directions to promote the deeper integration of LLMs with CFD applications:

(1) Building a CFD-specific knowledge graph, integrating OpenFOAM documentation, journal papers, and experimental databases, and generating structured training data through automated crawlers and semantic

annotation. Promoting a shareable, collaborative data crowdsourcing platform to encourage research institutions and enterprises to share anonymized CFD cases, establishing an open-source domain dataset.

(2) Developing an incremental learning mechanism to dynamically fine-tune LLMs based on new literature in fluid mechanics (e.g., discovery of turbulence models), avoiding "model obsolescence." Building a cross-scale model library, training specialized sub-models for different flow scenarios (e.g., microfluidics, high-Reynolds-number turbulence) to achieve "on-demand invocation."

(3) Embedding physical constraints involves incorporating residuals of the N-S equations, mass conservation, and other constraints as regularization terms in the loss function during the fine-tuning phase. However, it may be difficult for a CFD agent to strictly enforce dozens of tightly coupled physical constraints, possibly falling into the "runnable code trap". Therefore, more effort is necessary to solve the aforementioned problems. One can also integrate RAG to retrieve physics manuals and simulation results, forming a "prompt-generate-verify" closed loop. Embedding decision paths in LLM outputs generates traceable physical explanations. Developing visualization tools provides CFD engineers with interactive interfaces to display the physical basis of LLM-generated parameters.

The LLM-powered CFD framework shows promise in tackling varied flow patterns and intricate fluid dynamics challenges, making the technology more accessible and boosting modeling efficiency. Yet, LLM-CFD is at a critical transition period from toy cases to engineering applications, with current gaps remain in its ability to make independent decisions, handle complex shapes, and adapt to different real-world conditions. This approach highlights the growing synergy between scientific computing and AI, paving the way for smarter, faster, and easier-to-use CFD tools. In the next 3-5 years, through multi-agent collaboration, deep integration of domain knowledge, hybrid computing of HPC and AI, and enhanced interpretability, LLM-CFD has a potential to become a standard tool in the CFD field, significantly lowering the barrier to entry, accelerating innovation iteration, and propelling fluid dynamics research and industrial applications into a new paradigm. It should be noted that LLM-CFD is currently gaining increasing attention primarily in the computational aerodynamics field. However, chemical and energy process engineering often involves complex fluid flows, particularly multiphase flows, where relevant reports are still scarce. This scarcity stems from several possible hurdles: the extreme complexity of interfacial physics involving multiple phases with vastly different properties, the lack of standardized problem descriptions in natural language for these niche industrial processes, the limited availability of high-quality training data due to proprietary nature of industrial CFD simulations, and the challenge of encoding domain-specific constraints (e.g., thermodynamic equilibrium, reaction kinetics) into LLM prompts or fine-tuning datasets. These challenges require more research effort to be invested in the development of CFD agents for multiphase flow process equipment.

Last but not least, the analysis and summary of the above challenges and opportunities may not be entirely comprehensive and adequate, but it is hoped to serve as a catalyst to promote the faster and better development of the LLM-CFD field.

Author Contributions

P.-Z.M.: Conceptualization, Investigation, Visualization, Writing—original draft, Writing—review & editing; G.-D.G.: Investigation, Writing—original draft, Writing—review & editing; J.-K.L.: Supervision, Founding acquisition, Writing—review & editing; Z.-H.L.: Supervision, Writing—review & editing; L.-T.Z.: Conceptualization, Investigation, Founding acquisition, Supervision, Project administration, Resources, Writing—original draft, Writing—review & editing. All authors have read and agreed to the published version of the manuscript.

Funding

This work was supported by the National Natural Science Foundation of China (No. 22578267), and the project of State Key Laboratory of Polyolefins and Catalysis (grant No. SKLZX-2025-07).

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

This study is a review paper and does not generate new original data. All analyses are based on published literature, and relevant references have been adequately cited in the text.

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors used AI to improve language clarity and readability. After using this tool, the authors reviewed and edited the content as needed and take responsibility for the content of the publication.

References

1. Poinsot, T.; Candel, S.; Trouvé, A. Applications of Direct Numerical Simulation to Premixed Turbulent Combustion. *Prog. Energy Combust. Sci.* **1995**, *21*, 531–576. https://doi.org/10.1016/0360-1285(95)00011-9.
2. Xiong, Q.; Li, B.; Zhou, G.; et al. Large-Scale DNS of Gas–Solid Flows on Mole-8.5. *Chem. Eng. Sci.* **2012**, *71*, 422–430. https://doi.org/10.1016/j.ces.2011.10.059.
3. Fox, R.O. Large-Eddy-Simulation Tools for Multiphase Flows. *Annu. Rev. Fluid Mech.* **2012**, *44*, 47–76.
4. Chen, Z.; Deng, J. Data-Driven RANS Closures for Improving Mean Field Calculation of Separated Flows. *Front. Phys.* **2024**, *12*. https://doi.org/10.3389/fphy.2024.1347657.
5. Moin, P.; Mahesh, K. Direct Numerical Simulation: A Tool in Turbulence Research. *Annu. Rev. Fluid Mech.* **1998**, *30*, 539–578.
6. Menicali, L.; Grace, A.; Richter, D.H.; et al. A Physics-Informed Spatiotemporal Deep Learning Framework for Turbulent Systems. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2505.10919.
7. Li, H.; Su, X.; Yuan, X. Entropy Analysis of the Flat Tip Leakage Flow with Delayed Detached Eddy Simulation. *Entropy* **2018**, *21*, 21. https://doi.org/10.3390/e21010021.
8. Lin, D.; Xin, Y.; Su, X. Local Entropy Generation in Compressible Flow through a High Pressure Turbine with Delayed Detached Eddy Simulation. *Entropy* **2017**, *19*, 29. https://doi.org/10.3390/e19010029.
9. MacDougall, C.Y.; Piomelli, U.; Ambrogi, F. Evaluation of Turbulence Models in Unsteady Separation. *Fluids* **2023**, *8*, 273. https://doi.org/10.3390/fluids8100273.
10. Lima, L.E.M. Application of the One-Dimensional Drift-Flux Model for Numerical Simulation of Gas–Liquid Isothermal Flows in Vertical Pipes: A Mechanistic Approach Based on the Flow Pattern. *SN Appl. Sci.* **2020**, *2*, 658. https://doi.org/10.1007/s42452-020-2440-x.
11. Van Wachem, B.G.M.; Almstedt, A.E. Methods for Multiphase Computational Fluid Dynamics. *Chem. Eng. J.* **2003**, *96*, 81–98. https://doi.org/10.1016/j.cej.2003.08.025.
12. Cai, J.; Zhao, J.; Zhong, J.; et al. Microfluidic Experiments and Numerical Simulation Methods of Pore-Scale Multiphase Flow. *Capillarity* **2024**, *12*, 1–5. https://doi.org/10.46690/capi.2024.07.01.
13. Chen, X.; Wang, J. A Comparison of Two-Fluid Model, Dense Discrete Particle Model and CFD-DEM Method for Modeling Impinging Gas–Solid Flows. *Powder Technol.* **2014**, *254*, 94–102. https://doi.org/10.1016/j.powtec.2013.12.056.
14. Wachs, A. A DEM-DLM/FD Method for Direct Numerical Simulation of Particulate Flows: Sedimentation of Polygonal Isometric Particles in a Newtonian Fluid with Collisions. *Comput. Fluids* **2009**, *38*, 1608–1628. https://doi.org/10.1016/j.compfluid.2009.01.005.
15. Zhong, W.; Yu, A.; Liu, X.; et al. DEM/CFD-DEM Modelling of Non-Spherical Particulate Systems: Theoretical Developments and Applications. *Powder Technol.* **2016**, *302*, 108–152. https://doi.org/10.1016/j.powtec.2016.07.010.
16. Wang, S.; Shen, Y. CFD-DEM Modelling of Raceway Dynamics and Coke Combustion in an Ironmaking Blast Furnace. *Fuel* **2021**, *302*, 121167. https://doi.org/10.1016/j.fuel.2021.121167.
17. Zhu, L.-T.; Chen, X.-Z.; Ouyang, B.; et al. Review of Machine Learning for Hydrodynamics, Transport, and Reactions in Multiphase Flows and Reactors. *Ind. Eng. Chem. Res.* **2022**, *61*, 9901–9949. https://doi.org/10.1021/acs.iecr.2c01036.
18. Zhu, L.; Ouyang, B.; Lei, H.; et al. Conventional and Data-Driven Modeling of Filtered Drag, Heat Transfer, and Reaction Rate in Gas–Particle Flows. *AIChE J.* **2021**, *67*, e17299. https://doi.org/10.1002/aic.17299.
19. Mufti, B.; Perron, C.; Mavris, D.N. Nonlinear Reduced-Order Modeling of Compressible Flow Fields Using Deep Learning and Manifold Learning. *Phys. Fluids* **2025**, *37*, 036130. https://doi.org/10.1063/5.0253365.
20. Wang, D.; Guo, H.; Sun, Y.; et al. Prediction of Oil–Water Two-Phase Flow Patterns Based on Bayesian Optimisation of the XGBoost Algorithm. *Processes* **2024**, *12*, 1660. https://doi.org/10.3390/pr12081660.

21. Cui, Y.; Li, C.; Zhang, W.; et al. A Deep Learning-Based Image Processing Method for Bubble Detection, Segmentation, and Shape Reconstruction in High Gas Holdup Sub-Millimeter Bubbly Flows. *Chem. Eng. J.* **2022**, *449*, 137859. https://doi.org/10.1016/j.cej.2022.137859.
22. Zhang, D.; Ouyang, B.; Luo, Z.-H. Identification of Gas-Solid Flow Regimes Using Convolutional Neural Network Techniques. *Powder Technol.* **2024**, *442*, 119848. https://doi.org/10.1016/j.powtec.2024.119848.
23. Gómez-Camperos, J.A.; Diaz, C.M.R.; Hernandez-Cely, M.M.; et al. Dense-Gas/Liquid Two-Phase Flow Pattern Recognition Using Convolutional Neural Networks and Transfer Learning. *Int. J. Multiph. Flow* **2026**, *194*, 105479. https://doi.org/10.1016/j.ijmultiphaseflow.2025.105479.
24. Jiang, S.; Wu, K.; Francia, V.; et al. Machine Learning Assisted Experimental Characterization of Bubble Dynamics in Gas-Solid Fluidized Beds. *Ind. Eng. Chem. Res.* **2024**, *63*, 8819–8832. https://doi.org/10.1021/acs.iecr.4c00631.
25. Wang, Z.; Wang, Y.; Chen, P.; et al. A Study of Online Monitoring for Two-phase Flow Patterns in a Microchannel Based on Deep Learning. *AIChE J.* **2025**, *71*, e18785. https://doi.org/10.1002/aic.18785.
26. Wang, Y.; Li, Z.; Yuan, Z.; et al. Prediction of Turbulent Channel Flow Using Fourier Neural Operator-Based Machine-Learning Strategy. *Phys. Rev. Fluids* **2024**, *9*, 084604. https://doi.org/10.1103/PhysRevFluids.9.084604.
27. Lai, Y.; Peng, C.; Hu, W.; et al. Adaptive Optimization Random Forest for Pressure Prediction in Industrial Gas-Solid Fluidized Beds. *Powder Technol.* **2025**, *453*, 120607. https://doi.org/10.1016/j.powtec.2025.120607.
28. Xiao, H.; Oloruntoba, A.; Ke, X.; et al. Improving the Precision of Solids Velocity Measurement in Gas-Solid Fluidized Beds with a Hybrid Machine Learning Model. *Chem. Eng. Sci.* **2024**, *285*, 119579. https://doi.org/10.1016/j.ces.2023.119579.
29. Li, D.; Zhao, B.; Lu, S.; et al. A Data-Driven Method for Fast Predicting the Long-Term Hydrodynamics of Gas–Solid Flows: Optimized Dynamic Mode Decomposition with Control. *Phys. Fluids* **2024**, *36*, 103332. https://doi.org/10.1063/5.0232554.
30. Ranade, V.V.; Marchini, S.; Kipping, R.; et al. Estimation of Gas Hold-up in Bubble Columns Using Wall Pressure Fluctuations and Machine Learning. *Chem. Eng. J.* **2024**, *500*, 157078. https://doi.org/10.1016/j.cej.2024.157078.
31. Li, D.; Zhao, B.; Lu, S.; et al. Physics-Informed Dynamic Mode Decomposition for Short-Term and Long-Term Prediction of Gas-Solid Flows. *Chem. Eng. Sci.* **2024**, *289*, 119849. https://doi.org/10.1016/j.ces.2024.119849.
32. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; et al. Physics-Informed Machine Learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. https://doi.org/10.1038/s42254-021-00314-5.
33. Patel, Y.; Mons, V.; Marquet, O.; et al. Turbulence Model Augmented Physics Informed Neural Networks for Mean Flow Reconstruction. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2306.01065.
34. Bin, Y.; Huang, G.; Kunz, R.; et al. Constrained Re-Calibration of Reynolds-Averaged Navier-Stokes Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2310.09368.
35. Hu, T.; Qin, L.; Zhou, L.; et al. Analysis and Optimization of Methanol Production and Temperature Gradient in CO2 Hydrogenation Fixed Bed Reactors Using CFD and Bayesian Optimization. *Chem. Eng. Sci.* **2026**, *321*, 123030. https://doi.org/10.1016/j.ces.2025.123030.
36. Haake, J.; Oggian, T.; Utzig, J.; et al. Investigation of the Pressure Drop Increase in a Square Free-Vortex Cyclonic Separator Operating at Low Particle Concentration. *Powder Technol.* **2020**, *374*, 95–105. https://doi.org/10.1016/j.powtec.2020.07.008.
37. Xiang, S.; Wen, Q.; Wei, M.; et al. Optimization of the Double-Slot Blown Airfoil with Jet at the Leading and Trailing Edges of the Flap. *AIP Adv.* **2024**, *14*, 025341. https://doi.org/10.1063/5.0196505.
38. Zhang, R.; Huang, L.; Wang, K.; et al. Novel Optimized Layout for Flettner Rotors Based on Reuse of Wake Energy. *J. Clean. Prod.* **2024**, *443*, 140922. https://doi.org/10.1016/j.jclepro.2024.140922.
39. Kang, Z.; Feng, L.; Wang, J. Optimization of a Gas–Liquid Dual-Impeller Stirred Tank Based on Deep Learning with a Small Data Set from CFD Simulation. *Ind. Eng. Chem. Res.* **2024**, *63*, 843–855. https://doi.org/10.1021/acs.iecr.3c03561.
40. Touvron, H.; Martin, L.; Stone, K.; et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2307.09288.
41. Sriram, A.; Miller, B.K.; Chen, R.T.Q.; et al. FlowLLM: Flow Matching for Material Generation with Large Language Models as Base Distributions. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2410.23405.
42. Zhang, C.; Zhang, J.; Lu, J.; et al. Large Language Models Meet Energy Systems: Opportunities, Challenges, and Future Perspectives. *Appl. Energy* **2026**, *403*, 127076. https://doi.org/10.1016/j.apenergy.2025.127076.
43. Lin, L.; Zhou, X.; Yang, K.; et al. DeepSeek-LLM with Adaptive RAG for Pharmaceutical Dissolution Prediction. *Pharm. Res.* **2025**, *42*, 1821–1835. https://doi.org/10.1007/s11095-025-03932-1.
44. Lin, L.; Zhou, X.; Ding, Y.; et al. Generative Model Enhanced X-Ray Computed Tomography Imaging for Pharmaceutical Powder Microstructure Reconstruction and Evaluation. *AAPS PharmSciTech* **2025**, *27*, 37. https://doi.org/10.1208/s12249-025-03292-4.
45. Du, M.; Chen, Y.; Wang, Z.; et al. Large Language Models for Automatic Equation Discovery of Nonlinear Dynamics. *Phys. Fluids* **2024**, *36*, 097121. https://doi.org/10.1063/5.0224297.
46. Liu, N.; Jafarzadeh, S.; Lattimer, B.Y.; et al. Harnessing Large Language Models for Data-Scarce Learning of Polymer Properties. *Nat. Comput. Sci.* **2025**, *5*, 245–254. https://doi.org/10.1038/s43588-025-00768-y.

47. Guo, J.; Park, C.; Qian, D.; et al. Large Language Model-Empowered next-Generation Computer-Aided Engineering. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2509.11447.
48. Wang, W.; Xu, R.; Feng, J.; et al. A Status Quo Investigation of Large-Language Models for Cost-Effective Computational Fluid Dynamics Automation with OpenFOAMGPT. *Theor. Appl. Mech. Lett.* **2025**, *15*, 100623. https://doi.org/10.1016/j.taml.2025.100623.
49. Lin, L.; Liu, Y.; Liu, T.; et al. An SENet-Enhanced Transformer Approach for Multi-Condition Fast Forecasting of Two-Phase Bubble Flows. *Chem. Eng. J.* **2025**, *524*, 169259. https://doi.org/10.1016/j.cej.2025.169259.
50. Lorsung, C.; Farimani, A.B. Explain Like I'm Five: Using LLMs to Improve PDE Surrogate Models with Text. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2410.01137.
51. Bao, J.; Boullé, N.; Liu, T.J.B.; et al. Text-Trained LLMs Can Zero-Shot Extrapolate PDE Dynamics. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2509.06322.
52. Zhang, Y.; Zheng, K.; Liu, F.; et al. AutoTurb: Using Large Language Models for Automatic Algebraic Model Discovery of Turbulence Closure. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2410.10657.
53. Zhang, X.; Xu, Z.; Zhu, G.; et al. Using Large Language Models for Parametric Shape Optimization. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2412.08072.
54. Pandey, S.; Xu, R.; Wang, W.; et al. OpenFOAMGPT: A Retrieval-Augmented Large Language Model (LLM) Agent for OpenFOAM-Based Computational Fluid Dynamics. *Phys. Fluids* **2025**, *37*, 035120. https://doi.org/10.1063/5.0257555.
55. Chen, Y.; Zhu, X.; Zhou, H.; et al. Metaopenfoam: An LLM-Based Multi-Agent Framework for CFD. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2407.21320.
56. Vaswani, A.; Shazeer, N.; Parmar, N.; et al. Attention Is All You Need. *arXiv* **2023**. https://doi.org/10.48550/arXiv.1706.03762.
57. Park, N.H.; Manica, M.; Born, J.; et al. Artificial Intelligence Driven Design of Catalysts and Materials for Ring Opening Polymerization Using a Domain-Specific Language. *Nat. Commun.* **2023**, *14*, 3686. https://doi.org/10.1038/s41467-023-39396-3.
58. Wei, J.; Wang, X.; Schuurmans, D.; et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2201.11903.
59. Wang, X.; Wei, J.; Schuurmans, D.; et al. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2203.11171.
60. Yao, S.; Yu, D.; Zhao, J.; et al. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2305.10601.
61. Sahoo, P.; Singh, A.K.; Saha, S.; et al. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2402.07927.
62. Asad, F.; Shah, W. AI Ethics and Safety: Guiding Principles for Fine-Tuning Machine Learning Models. 2024. Available online: https://www.researchgate.net/publication/387517530_AI_Ethics_and_Safety_Guiding_Principles_for_Fine-Tuning_Machine_Learning_Models (accessed on 20 February 2026).
63. Pake, T.U.; Pachareney, U. Fine-Tuning Strategies for Transformers. In Proceedings of the 2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL), Bhimdatta, Nepal, 18–20 February 2025; pp. 670–675. https://doi.org/10.1109/ICSADL65848.2025.10933255.
64. Hu, E.J.; Shen, Y.; Wallis, P.; et al. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv* **2021**. https://doi.org/10.48550/arXiv.2106.09685.
65. Gao, Y.; Xiong, Y.; Gao, X.; et al. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2312.10997.
66. Shahade, A.K.; Deshmukh, P.V. Enhancing Natural Language Processing: A Comprehensive Review of Retrieval Augmented Generation. In Proceedings of the 2024 4th International Conference on Sustainable Expert Systems (ICSES), Lekhnath, Nepal, 15–17 October 2024; pp. 609–611. https://doi.org/10.1109/ICSES63445.2024.10763224.
67. Dhuliawala, S.; Komeili, M.; Xu, J.; et al. Chain-of-Verification Reduces Hallucination in Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*; Association for Computational Linguistics: Bangkok, Thailand, 2024; pp. 3563–3578. https://doi.org/10.18653/v1/2024.findings-acl.212.
68. Yu, W.; Zhang, H.; Pan, X.; et al. Chain-of-Note: Enhancing Robustness in Retrieval-Augmented Language Models. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2311.09210.
69. Li, X.; Zhao, R.; Chia, Y.K.; et al. Chain-of-Knowledge: Grounding Large Language Models via Dynamic Knowledge Adapting over Heterogeneous Sources. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2305.13269.
70. Yao, S.; Zhao, J.; Yu, D.; et al. ReAct: Synergizing Reasoning and Acting in Language Models. In Proceedings of the The Eleventh International Conference on Learning Representations ICLR 2023, Kigali, Rwanda, 1–5 May 2023.
71. Hosseini Boosari, S.S. Predicting the Dynamic Parameters of Multiphase Flow in CFD (Dam-Break Simulation) Using Artificial Intelligence-(Cascading Deployment). *Fluids* **2019**, *4*, 44. https://doi.org/10.3390/fluids4010044.

72. Vatsal, S.; Dubey, H. A Survey of Prompt Engineering Methods in Large Language Models for Different NLP Tasks. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2407.12994.
73. Sivarajkumar, S.; Kelley, M.; Samolyk-Mazzanti, A.; et al. An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing. *JMIR Med. Inform.* **2024**, *12*, e55318.
74. Saleem, S.; Asim, M.N.; Zulfiqar, S.; et al. The Evolution of Natural Language Processing: How Prompt Optimization and Language Models Are Shaping the Future. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2506.17700.
75. Zhu, M.; Bazaga, A.; Liò, P. FLUID-LLM: Learning Computational Fluid Dynamics with Spatiotemporal-Aware Large Language Models. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2406.04501.
76. Dinh, N.-D.; Nguyen, N.; Tong, R. Autonomous Droplet Microfluidic Design Framework with Large Language Models. *Res. Sq.* **2025**, *10*, 45801–45814. https://doi.org/10.21203/rs.3.rs-6282521/v1.
77. Okafor, C.E.; Iweriolor, S.; Ani, O.I.; et al. Advances in Machine Learning-Aided Design of Reinforced Polymer Composite and Hybrid Material Systems. *Hybrid Adv.* **2023**, *2*, 100026. https://doi.org/10.1016/j.hybadv.2023.100026.
78. Zhang, Z.; Li, A.; Wang, L.; et al. Big Data-Assisted Urban Governance: A Comprehensive System for Business Documents Classification of the Government Hotline. *Eng. Appl. Artif. Intell.* **2024**, *132*, 107997. https://doi.org/10.1016/j.engappai.2024.107997.
79. Elgeldawi, E.; Sayed, A.; Galal, A.R.; et al. Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis. *Informatics* **2021**, *8*, 79. https://doi.org/10.3390/informatics8040079.
80. Petruzzellis, F.; Testolin, A.; Sperduti, A. Benchmarking GPT-4 on Algorithmic Problems: A Systematic Evaluation of Prompting Strategies. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2402.17396.
81. Pryzant, R.; Iter, D.; Li, J.; et al. Automatic Prompt Optimization with 'Gradient Descent' and Beam Search. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2305.03495.
82. Li, M.; Wang, W.; Feng, F.; et al. Robust Prompt Optimization for Large Language Models Against Distribution Shifts. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–10 December 2023; pp. 1539–1554. https://doi.org/10.18653/v1/2023.emnlp-main.95.
83. Jaitly, S.; Shah, T.; Shugani, A.; et al. Towards Better Serialization of Tabular Data for Few-Shot Classification with Large Language Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2312.12464.
84. Cui, W.; Li, Z.; Sun, H.; et al. A Survey of Automatic Prompt Optimization with Instruction-Focused Heuristic-Based Search Algorithm. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2502.18746.
85. Besta, M.; Blach, N.; Kubicek, A.; et al. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proc. AAAI Conf. Artif. Intell.* **2024**, *38*, 17682–17690. https://doi.org/10.1609/aaai.v38i16.29720.
86. Yu, P.; Xu, J.; Weston, J.; et al. Distilling System 2 into System 1. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2407.06023.
87. Zhou, Y.; Geng, X.; Shen, T.; et al. Thread of Thought Unraveling Chaotic Contexts. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2311.08734.
88. Besta, M.; Memedi, F.; Zhang, Z.; et al. Demystifying Chains, Trees, and Graphs of Thoughts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *47*, 10967–10989. https://doi.org/10.1109/TPAMI.2025.3598188.
89. Jiang, Y.; Byrne, E.; Chen, X.; et al. A Dynamic PCA and Transformers Coupled Reduced-Order Model for Transient Solid-Liquid Flows in Stirred Tanks. *Chem. Eng. Process. Process Intensif.* **2026**, *219*, 110605. https://doi.org/10.1016/j.cep.2025.110605.
90. Yaqub, M.W.; Chen, X. Vision Transformers for Three-phase Flow Classifications with Data Augmentation through Generative Adversarial Networks. *AIChE J.* **2025**, *71*, e70002. https://doi.org/10.1002/aic.70002.
91. Zhang, W.-W.; Noack, B.R. Artificial Intelligence in Fluid Mechanics. *Acta Mech. Sin.* **2021**, *37*, 1715–1717. https://doi.org/10.1007/s10409-021-01154-3.
92. Zou, Z.; Xu, P.; Chen, Y.; et al. Application of Artificial Intelligence in Turbomachinery Aerodynamics: Progresses and Challenges. *Artif. Intell. Rev.* **2024**, *57*, 222. https://doi.org/10.1007/s10462-024-10867-3.
93. Suh, Y.; Chandramowlishwaran, A.; Won, Y. Recent Progress of Artificial Intelligence for Liquid-Vapor Phase Change Heat Transfer. *npj Comput. Mater.* **2024**, *10*, 65. https://doi.org/10.1038/s41524-024-01223-8.
94. Wang, W.; Chu, X. Optimized Flow Control Based on Automatic Differentiation in Compressible Turbulent Channel Flows. *J. Fluid Mech.* **2025**, *1011*, A1. https://doi.org/10.1017/jfm.2025.304.
95. Taira, K.; Rigas, G.; Fukami, K. Machine Learning in Fluid Dynamics: A Critical Assessment. *Phys. Rev. Fluids* **2025**, *10*, 090701. https://doi.org/10.1103/8t52-mtb9.
96. Adjailia, F.; Takac, M. Exploring the Application of Machine Learning in Computational Fluid Dynamics. In Proceedings of the 2023 IEEE World Conference on Applied Intelligence and Computing (AIC), Sonbhadra, India, 29–30 July 2023; pp. 51–57. https://doi.org/10.1109/AIC57670.2023.10263832.
97. Wang, G.; Cheng, S. Can Foundation Language Models Predict Fluid Dynamics? *Eng. Appl. Artif. Intell.* **2025**, *158*, 111427. https://doi.org/10.1016/j.engappai.2025.111427.

98. Dong, Z.; Lu, Z.; Yang, Y. Fine-Tuning a Large Language Model for Automating Computational Fluid Dynamics Simulations. *Theor. Appl. Mech. Lett.* **2025**, *15*, 100594. https://doi.org/10.1016/j.taml.2025.100594.
99. Yang, S.D.; Ali, Z.A.; Wong, B.M. FLUID-GPT (Fast Learning to Understand and Investigate Dynamics with a Generative Pre-Trained Transformer): Efficient Predictions of Particle Trajectories and Erosion. *Ind. Eng. Chem. Res.* **2023**, *62*, 15278–15289. https://doi.org/10.1021/acs.iecr.3c01639.
100. Sahibzada, S.; Malik, F.S.; Nasir, S.; et al. AI-Augmented Turbulence and Aerodynamic Modelling: Accelerating High-Fidelity CFD Simulations with Physics-Informed Neural Networks. *Int. J. Innov. Res. Comput. Sci. Technol.* **2025**, *13*, 91–97. https://doi.org/10.55524/ijircst.2025.13.1.14.
101. Liu, S.; Gao, C.; Li, Y. Large Language Model Agent for Hyper-Parameter Optimization. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2402.01881.
102. Dandamudi, S.R.P.; Sajja, J.; Khanna, A. Leveraging Artificial Intelligence for Data Networking and Cybersecurity in the United States. *Int. J. Innov. Res. Comput. Sci. Technol.* **2025**, *13*, 34–41. https://doi.org/10.55524/ijircst.2025.13.1.5.
103. Mehta, M.; Palade, V.; Chatterjee, I. (Eds.) *Explainable AI: Foundations, Methodologies and Applications*; Springer International Publishing: Berlin/Heidelberg, Germany, 2023. https://doi.org/10.1007/978-3-031-12807-3.
104. Uddin, M.; Irshad, M.S.; Kandhro, I.A.; et al. Generative AI Revolution in Cybersecurity: A Comprehensive Review of Threat Intelligence and Operations. *Artif. Intell. Rev.* **2025**, *58*, 236. https://doi.org/10.1007/s10462-025-11219-5.
105. Zainab, H.; Khan, A.R.A.; Khan, M.I.; et al. Ethical Considerations and Data Privacy Challenges in AI-Powered Healthcare Solutions for Cancer and Cardiovascular Diseases. *Glob. Trends Sci. Technol.* **2025**, *1*, 63–74. https://doi.org/10.70445/gtst.1.1.2025.63-74.
106. Agarwal, I.; Killamsetty, K.; Popa, L.; et al. DELIFT: Data Efficient Language Model Instruction Fine Tuning. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2411.04425.
107. Ekradi, K.; Madadi, A. Performance Improvement of a Transonic Centrifugal Compressor Impeller with Splitter Blade by Three-Dimensional Optimization. *Energy* **2020**, *201*, 117582. https://doi.org/10.1016/j.energy.2020.117582.
108. Hamici, H.; Kanan, A.; Al-hammuri, K. Optimized FIR Filter Using Genetic Algorithms: A Case Study of ECG Signals Filter Optimization. *BioMedInformatics* **2023**, *3*, 1197–1215. https://doi.org/10.3390/biomedinformatics3040071.
109. Ghatasheh, N.; Altaharwa, I.; Aldebei, K. Modified Genetic Algorithm for Feature Selection and Hyper Parameter Optimization: Case of XGBoost in Spam Prediction. *IEEE Access* **2022**, *10*, 84365–84383. https://doi.org/10.1109/ACCESS.2022.3196909.
110. Takabatake, T.; Yamamoto, M.; Hino, H. Algorithm for Searching Optimal Set Values of Absorption Chiller System Using Bayesian Optimization. *Sci. Technol. Built Environ.* **2022**, *28*, 188–199. https://doi.org/10.1080/23744731.2021.2005376.
111. Wang, X.; Jin, Y.; Schmitt, S.; et al. Recent Advances in Bayesian Optimization. *ACM Comput. Surv.* **2023**, *55*, 1–36. https://doi.org/10.1145/3582078.
112. Herrera Casanova, R.; Conde, A. Enhancement of LSTM Models Based on Data Pre-Processing and Optimization of Bayesian Hyperparameters for Day-Ahead Photovoltaic Generation Prediction. *Comput. Electr. Eng.* **2024**, *116*, 109162. https://doi.org/10.1016/j.compeleceng.2024.109162.
113. Wang, L.; Zhang, L.; He, G. Evaluations of Large Language Models in Computational Fluid Dynamics: Leveraging, Learning and Creating Knowledge. *Theor. Appl. Mech. Lett.* **2025**, *15*, 100597. https://doi.org/10.1016/j.taml.2025.100597.
114. Xu, Z.; Wang, L.; Wang, C.; et al. CFDAgent: A Language-Guided, Zero-Shot Multi-Agent System for Complex Flow Simulation. *Phys. Fluids* **2025**, *37*, 117124. https://doi.org/10.1063/5.0294696.
115. Kan, K.; Zhou, J.; Feng, J.; et al. Intelligent Optimization of Axial-Flow Pump Using Physics-Considering Machine Learning. *J. Comput. Des. Eng.* **2024**, *11*, 325–342. https://doi.org/10.1093/jcde/qwae013.
116. Li, Y.; Xu, J.; Li, F.; et al. Multi-Objective Optimization of Aerodynamic Performance and Internal Flow in Centrifugal Compressor Based on Neural Networks and Genetic Algorithms. *AIP Adv.* **2025**, *15*, 085105. https://doi.org/10.1063/5.0280922.
117. Milanese, M.; Colangelo, G.; Laforgia, D.; et al. Multi-Parameter Optimization of Double-Loop Fluidized Bed Solar Reactor for Thermochemical Fuel Production. *Energy* **2017**, *134*, 919–932. https://doi.org/10.1016/j.energy.2017.06.088.
118. Liu, F.; Lin, X.; Wang, Z.; et al. Large Language Model for Multi-Objective Evolutionary Optimization. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2310.12541.
119. Wang, W.; Peng, J.; Hu, M.; et al. LLM Agent for Hyper-Parameter Optimization. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2506.15167.
120. Zhang, M.R.; Desai, N.; Bae, J.; et al. Using Large Language Models for Hyperparameter Optimization. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2312.04528.
121. Mahammadli, K.; Ertekin, S. Sequential Large Language Model-Based Hyper-Parameter Optimization. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2410.20302.
122. Kochnev, R.; Goodarzi, A.T.; Bentyn, Z.A.; et al. Optuna vs Code Llama: Are LLMs a New Paradigm for Hyperparameter Tuning? *arXiv* **2025**. https://doi.org/10.48550/arXiv.2504.06006.

123. Chen, Y. Active Learning over DNN: Automated Engineering Design Optimization for Fluid Dynamics Based on Self-Simulated Dataset. *arXiv* **2020**. https://doi.org/10.48550/arXiv.2001.08075.
124. Amaral, C.A.; Oliveira, V.L.; Salazar, J.P.L.C.; et al. Quantum Machine Learning and Quantum-Inspired Methods Applied to Computational Fluid Dynamics: A Short Review. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2510.14099.
125. Feng, J.; Xu, R.; Chu, X. OpenFOAMGPT 2.0: End-to-End, Trustworthy Automation for Computational Fluid Dynamics. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2504.19338.
126. Hong, S.; Zhuge, M.; Chen, J.; et al. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2308.00352.
127. Chen, Y.; Zhu, X.; Zhou, H.; et al. MetaOpenFOAM 2.0: Large Language Model Driven Chain of Thought for Automating CFD Simulation and Post-Processing. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2502.00498.
128. Zhou, D.; Schärli, N.; Hou, L.; et al. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. *arXiv* **2023**. https://doi.org/10.48550/arXiv.2205.10625.
129. Paul, D.; West, R.; Bosselut, A.; et al. Making Reasoning Matter: Measuring and Improving Faithfulness of Chain-of-Thought Reasoning. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2402.13950.
130. Diao, S.; Wang, P.; Lin, Y.; et al. Active Prompting with Chain-of-Thought for Large Language Models. *arXiv* **2024**. https://doi.org/10.48550/arXiv.2302.12246.
131. Chen, Y.; Zhang, L.; Zhu, X.; et al. OptMetaOpenFOAM: Large Language Model Driven Chain of Thought for Sensitivity Analysis and Parameter Optimization Based on CFD. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2503.01273.
132. Fan, E.; Hu, K.; Wu, Z.; et al. ChatCFD: An LLM-Driven Agent for End-to-End CFD Automation with Domain-Specific Structured Reasoning. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2506.02019.
133. Yang, Z.; Bin, Y.; Shi, Y.; et al. Large Language Model Driven Development of Turbulence Models. *arXiv* **2025**. https://doi.org/10.48550/arXiv.2505.01681.