COMFYFLOW: Benchmarking LLMs for AIGC Workflow Generation

Zhenran Xu[1], **Yiyu Wang**[2], **Yunxin Li**[1], **Muyang Ye**[3], **Xue Yang**[2], **Kai Chen**[4], **Longyue Wang**[2], **Weihua Luo**[2], **Baotian Hu**[1*], **Min Zhang**[1]
[1]Harbin Institute of Technology (Shenzhen) [2]Alibaba Group
[3]Zhejiang University [4]Huaqiao University
xuzhenran@stu.hit.edu.cn,wangyiyu18@mails.ucas.ac.cn
{shali.yx,wanglongyue.wly,weihua.luowh}@alibaba-inc.com
{liyx,hubaotian,zhangmin2021}@hit.edu.cn

Abstract

Large language models (LLMs) have shown promising advancements in tackling human-level tasks, wherein generating workflows for collaborative AI systems remains a critical and challenging step. To explore this frontier, we introduce COMFYFLOW, a comprehensive benchmark to evaluate current LLMs' ability to generate executable and instruction-following AIGC workflows in ComfyUI. The dataset includes 400 diverse visual generation tasks across 20 categories, supported by 10K training examples constructed from knowledge bases, which contain detailed annotations for 2,480 nodes and 3,298 workflows. We establish a systematic evaluation protocol that quantifies performance across multiple dimensions, ranging from basic format validity to multi-level hallucination rates. Our extensive evaluations show that: 1) COMFYFLOW presents a substantial challenge even for top-tier proprietary LLMs such as GPT-5.1 and the Claude series; 2) Open-source models achieve new state-of-the-art results after post-training, yet struggle with long-horizon planning as the number of nodes increases; 3) Different post-training strategies offer complementary benefits in following instructions and mitigating hallucinations. By establishing both a challenging benchmark and a principled evaluation scheme, COMFYFLOW lays the foundation for developing more intelligent and reliable collaborative AI systems[1].

1 Introduction

The recent evolution of AI is defined by the growing importance of collaborative AI systems, which integrate multiple models and tools to work as a whole collaborative system (Xue et al., 2025). There have been multiple successful applications where multiple large language models (LLMs) with specialized capabilities utilize workflows to work collaboratively to achieve more effective solutions (Xu et al., 2023; Guo et al., 2024; Xi et al., 2023; Li et al., 2025b). Despite their promising performance, the workflows in these systems are manually crafted, i.e., the collaboration structure and model prompts are pre-determined and static (Zhang et al., 2025). As LLMs drive significant advancements in tackling reasoning and planning tasks, our work aims to explore: *Are LLMs capable of automated workflow generation akin to a human expert?* This capability represents a pivotal step in advancing from OpenAI's "Reasoners" (Level 2) toward "Organizational AI" (Level 5), the final and most sophisticated stage in OpenAI's five-level taxonomy of AGI, where systems must autonomously coordinate complex operations to approach human-level intelligence (OpenAI, 2023).

In this work, we explore this question in ComfyUI, a trending open-source AIGC platform. As shown in Figure 1, ComfyUI supports a vast array of generative tools, allowing users to treat functional components as distinct nodes and connect them to build complex workflows for highly customized artwork (Xu et al., 2025b). However, constructing these workflows from scratch is non-trivial, demanding substantial domain expertise and planning ability. While recent works have explored applying LLMs to AIGC workflow generation, three critical limitations remain: (1) **Limited task scope**: Existing works have limited coverage of visual generation tasks, mainly focusing on text-to-image generation (Gal et al., 2024; Sobania et al., 2024) while neglecting other important AIGC tasks such as image editing, video and 3D generation. (2) **Coarse-grained evaluation**: Current benchmark rely solely on pass rates, ignoring hallucinations in nodes, parameters, and graph structures that prevent valid workflow execution. (3) **Lack of high-quality open-source data**: Previous studies mainly provide test sets for evaluation, lacking comprehensive knowledge bases (KBs) or

*Corresponding author.
[1]GitHub: AIDC-AI/ComfyUI-Copilot

Findings of the Association for Computational Linguistics: ACL 2026, pages 2903–2916
July 2-7, 2026

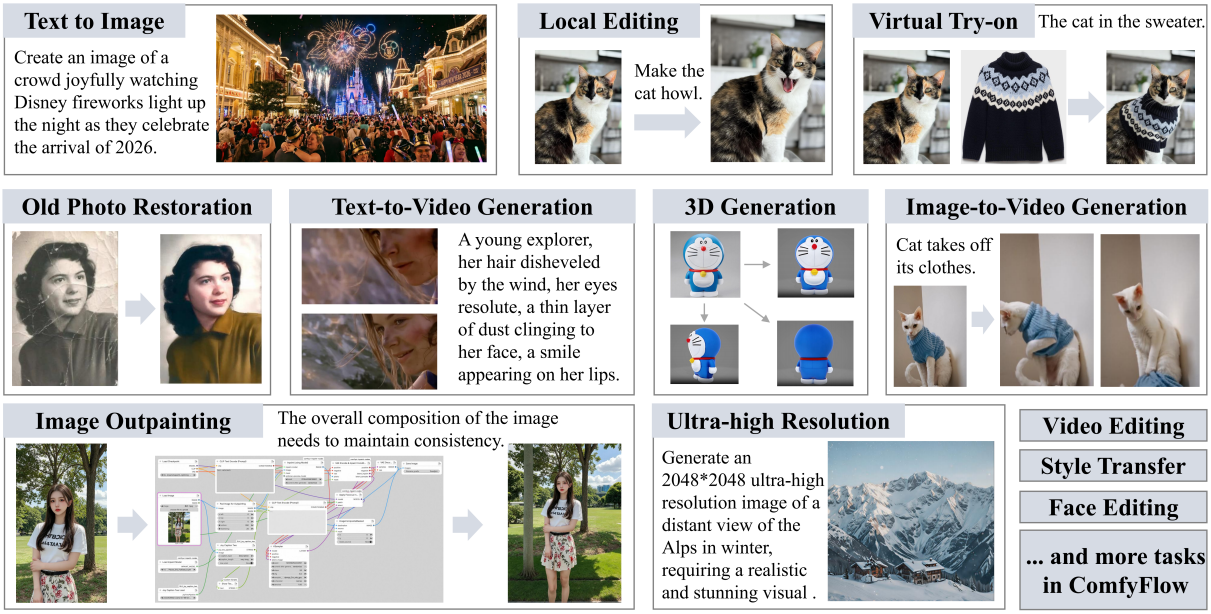


Figure 1: **Examples of ComfyFlow workflows and their execution outputs.** For the image outpainting task, we also provide a visualization of the ComfyUI workflow on canvas.

detailed workflow annotations, which are essential for training advanced models to push the boundary further. The above limitations motivate our work.

To address the above issues, we present ComfyFlow, a comprehensive dataset for AIGC workflow generation with the following features:

- **Comprehensive task coverage**: As shown in Figure 1, ComfyFlow comprises 400 diverse tasks spanning 20 types of AIGC challenges with varying difficulty levels, covering image, video, and 3D generation. It is supported by 10K training data constructed from KBs, featuring detailed annotations for 2,480 nodes and 3,298 workflows. To the best of our knowledge, ComfyFlow is the first project to open-source the large-scale workflow KB and constructed post-training dataset.

- **Strict quality control**: Starting with raw data collected from the ComfyUI community, we conduct rigorous validation of graph structures and execution success, followed by human verification. This process has filtered 27K original workflows down to 3.3K high-quality examples.

- **Fine-grained quantitative evaluation**: We establish a protocol to accurately quantify workflow generation capabilities, evaluating format validity, multi-level hallucinations (parameter, node, and graph levels), and instruction following based on subgraph and subsequence matching.

Following Ye et al. (2025), we reformulate the workflow generation task as domain-specific code generation. Given a task instruction and candidate nodes, LLMs are required to generate a code representation of the workflow, which can be seamlessly converted into a Directed Acyclic Graphs (DAGs) on the ComfyUI canvas. We evaluate 13 proprietary and open-source LLMs on ComfyFlow, including the latest Deepseek V3.2 and GPT-5.1. Our analysis reveals that while current LLMs can generate format-compliant workflows (with GPT-4o achieving 96% format validity), their ability to generate instruction-following, hallucination-free workflows remains limited; even GPT-5.1 achieves a pass rate of only 56%. Although open-source models initially lag behind proprietary ones, they achieve new state-of-the-art (SOTA) results on ComfyFlow through supervised fine-tuning (SFT) or reinforcement learning (RL) on the training set. Furthermore, we find that different post-training strategies offer complementary benefits: SFT excels at instruction following, while RL with fine-grained rewards is superior in mitigating hallucinations and improving robustness. Finally, despite these promising gains, SOTA LLMs still exhibit notable performance degradation as the number of nodes increases (e.g., GPT-4o drops from 91% pass rate on easy tasks to 35% on hard tasks), highlighting the persistent challenge of long-horizon planning.

To sum up, we summarize our main contribu-

tions as follows:

- We propose ComfyFlow, a high-quality, comprehensive AIGC workflow generation dataset with wide task coverage, accompanied by a large-scale workflow and node knowledge base.
- We introduce a fine-grained evaluation protocol for workflow generation, ranging from fundamental format validity to hallucinations at the node, graph, and parameter levels.
- Experimental results demonstrate that ComfyFlow presents a significant challenge for current SOTA models. We further leverage the training data to explore various post-training methods and reward designs, paving the way for more intelligent collaborative AI systems.

2 Related Work

2.1 General Workflow Generation

A significant trend in recent research is the development of collaborative AI systems that orchestrate multiple models and tools within structured workflows to achieve superior performance (Xue et al., 2025). Through such workflows, multiple models can collaborate effectively, demonstrating collective intelligence across various domains. For example, MetaGPT (Hong et al., 2024) and ChatDev (Qian et al., 2023) assemble teams of models with specialized programmer roles to solve complex coding tasks via predefined workflows. Similar applications include multi-LLMs as judge (Chan et al., 2024; Chen et al., 2024), ultra-long literary translation (Wu et al., 2024a,b), and long video generation (Xu et al., 2025a; Li et al., 2024b; Shi et al., 2025). However, these manually crafted workflows are often static, time-consuming to develop, and dependent on domain expertise, limiting their flexibility and scalability.

To address this challenge, recent studies have focused on training models or developing language agents to automatically generate workflows (Shang et al., 2025; Niu et al., 2025; Hu et al., 2025; Xu et al., 2024). For example, WorkflowLLM (Fan et al., 2025) uses data-centric fine-tuning to improve API orchestration, while AFlow (Zhang et al., 2025) employs Monte Carlo Tree Search (MCTS) to explore the expansive action space, iteratively refining workflows to solve math problems. These advancements reflect a shift toward managing complex, realistic demands. In this context, the ComfyUI platform (comfyanonymous, 2023) stands out as a particularly challenging testbed in this context, due to the large number of components involved in AI art creation workflows and the intricate connections between different modules with diverse functionalities (Xue et al., 2025; Guo et al., 2025).

2.2 AIGC Workflow Generation

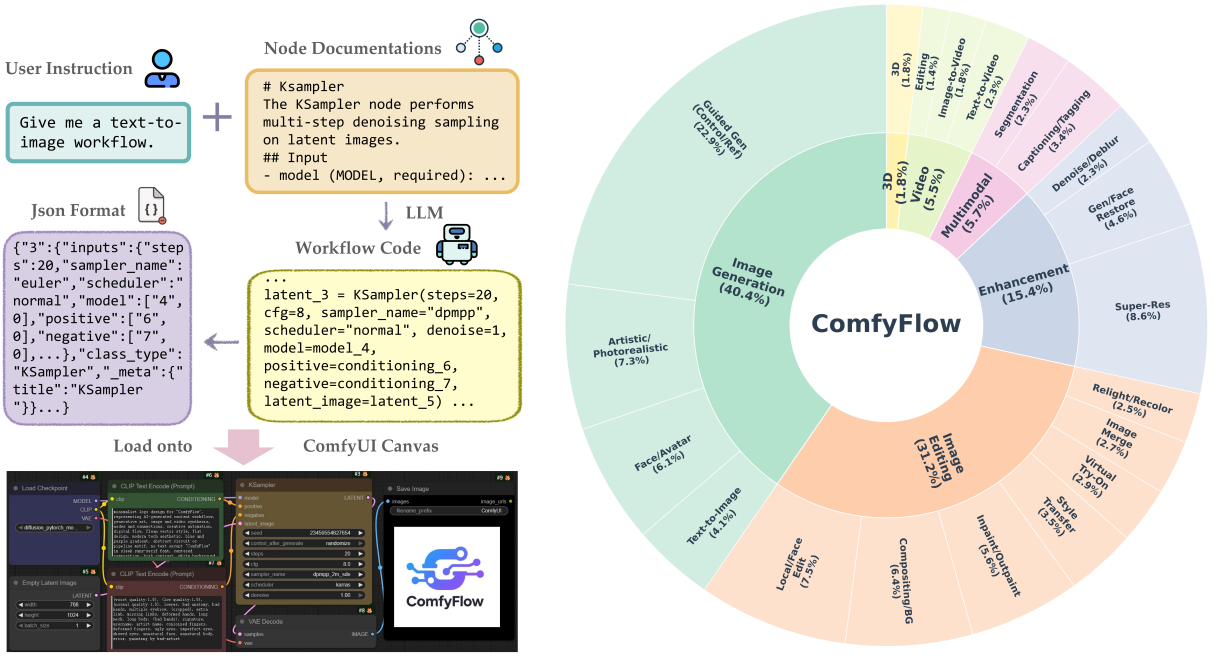
Recent advances in AIGC have transitioned from end-to-end diffusion models (Dhariwal and Nichol, 2021; Wang et al., 2025; Podell et al., 2023; Li et al., 2024a, 2025a) to more sophisticated workflows on open-source platforms such as ComfyUI. This modularity allows users to construct workflows by connecting specific blocks, such as LLMs for prompt refinement, LoRA modules for artistic style transfer, or super-resolution blocks for finer detail (Hu et al., 2021; Mañas et al., 2024; Ning et al., 2021; Chen et al., 2025). However, proper workflow design and node selection require significant domain expertise (Xu et al., 2025b), creating a strong demand for automated workflow generation.

AIGC workflow generation has emerged as a trending research topic in multimodal content synthesis. For example, ComfyBench (Xue et al., 2025) proposes a multi-agent framework with memory, planning and action modules for step-by-step workflow design. Despite the advancements, the field remains in its early stages with three key limitations. Firstly, some existing methods are limited to text-to-image generation (Sobania et al., 2024; Gal et al., 2024), restricting applicability to broader multimodal tasks. Secondly, generated workflows often suffer from structural inconsistencies and hallucinated components (Huang et al., 2025; Guo et al., 2024), yet current benchmarks lack fine-grained metrics to evaluate these specific errors. Thirdly, previous studies lack comprehensive KBs or detailed workflow annotations. For example, the open-source ComfyBench only provides 20 annotated workflows. To address these gaps, our work expands the scope to a diverse range of AIGC tasks, sets a fine-grained evaluation scheme, and leverages the constructed KBs to explore LLM post-training for automated workflow generation.

3 ComfyFlow

3.1 Overview

We present ComfyFlow, a comprehensive dataset for AIGC workflow generation. Following previous works (Ye et al., 2025; Fan et al., 2025), we formulate workflow generation as a domain-specific



(a) Problem Illustration

(b) COMFYFLOW Task Distribution

Figure 2: **(Left)** Example of a workflow generation task. Given an AIGC task description and node documentation, the goal is to generate a code-formatted workflow. **(Right)** The sunburst chart shows 6 categories and 20 subtypes in COMFYFLOW, where the sector area represents the proportion of each type.

code generation task. Specifically, as shown in Figure 2 (a), given an AIGC task description and a set of candidate nodes with their documentation, the goal is to generate a code representation of an instruction-following, executable workflow.

To facilitate the effective training and evaluation of LLMs, COMFYFLOW comprises a training set of 10,677 entries. Each entry pairs a natural language task description with its corresponding ComfyUI workflow in the code format. We have meticulously categorized the workflows in COMFYFLOW into 6 primary types and 20 subtypes (detailed in Appendix A). Figure 2 (b) shows the type distribution. Additionally, we provide a curated test set containing 400 entries. As demonstrated in Table 1, COMFYFLOW provides the most extensive task support and the largest training corpus to date.

3.2 Data Construction

We construct comprehensive KBs for ComfyUI workflows and nodes, sourcing data from popular generative resource platforms, ComfyUI-related GitHub repositories, and the official ComfyUI website. Given the significant noise in raw community data, we conduct a rigorous cleaning process to filter and standardize the data.

Node KB From an initial collection of 40K nodes, we first perform exact-match deduplication. Nodes lacking defined input or output parameters are discarded, leaving 6,825 unique nodes. For nodes missing structured documentation, we utilize Claude 3.5 to generate detailed descriptions by analyzing their source code from GitHub. The resulting documentation is standardized to include the node type, functional description, and the meanings of each input/output parameter. Furthermore, we augment each entry with a usage code snippet derived from the workflow KB. Our preliminary study indicates that such examples are vital for ensuring models generate workflows with correct parameter configurations. Here we open-source the detailed documentation for the 2,480 nodes utilized in our workflow KB. An example of a Node KB entry is provided in Appendix B.

Workflow KB As shown in Figure 2 (a), workflows are originally stored as JSON files on the ComfyUI canvas. We develop a parser to extract a Directed Acyclic Graph (DAG) from these files, which we then convert into a sequence of Python-like function calls in topological order. We also implement the reverse parser, enabling seamless conversion between JSON and code while preserving compatibility with ComfyUI.

Dataset	Approach	Task Types						Dataset Statistics	
		T2I	IE	IR	3DG	VG	VE	Instructions	Workflows
Training	ComfyGen (Gal et al., 2024)	✓	×	×	×	×	×	500	310
	ComfyFlow (Ours)	✓	✓	✓	✓	✓	✓	**10,677**	**2,898**
Evaluation	ComfyGen (Gal et al., 2024)	✓	×	×	×	×	×	**500**	-
	ComfyBench (Xue et al., 2025)	✓	✓	✓	×	✓	×	200	20
	ComfyFlow (Ours)	✓	✓	✓	✓	✓	✓	400	**400**

Table 1: Comparison of our ComfyFlow and other AIGC workflow generation benchmarks. Task categories: Text-to-Image (T2I), Image Editing (IE), Image Enhancement and Restoration (IR), 3D Generation (3DG), Video Generation (VG), and Video Editing (VE).

We begin with a raw collection of 27K workflows and apply a multi-stage filtering and refinement pipeline. First, every workflow is imported into the ComfyUI canvas via Selenium[2], and any example that cannot be transformed into the API format is discarded. This step ensures that all remaining 19K workflows adhere to basic execution requirements. Second, we remove exact duplicates, reducing the count to 5,909. Third, we verify the integrity of the bidirectional transformation between JSON and code, retaining only workflows that support both conversions. We then refine the remaining entries by removing auxiliary or irrelevant components, such as "Anything Anywhere" nodes, which leaves 4,077 workflows. Finally, we exclude workflows containing nodes not present in our 6.8K-node KB. This cleaning process results in a preliminary set of 3,395 workflows, with an average of 20 nodes per workflow.

Since community-contributed workflows often focus more on installation than functionality, many lack clear descriptions. To bridge this gap, we utilize the multimodal reasoning ability of GPT-4o. We prompt the model with a combination of community-provided text and accompanying preview images, which typically illustrate workflow outcomes, to generate functional descriptions. This stage also serves as a safety filter to remove NSFW content, resulting in a final collection of 3,298 high-quality workflows. Therefore, each entry in our workflow KB includes a JSON representation, a code representation, and a functional description.

Training and Evaluation Set Each entry in the workflow KB is represented as a pair $(desc, c)$, where $desc$ denotes the functional description and c denotes its corresponding code representation. Let $\mathcal{V}^g$ represent the set of nodes in the ground-truth workflow, and $\mathcal{V}^{\text{KB}}$ denote the full set of nodes in the KB. we construct a candidate node set $\mathcal{V}^{\text{cand}}$ by combining the ground-truth nodes $\mathcal{V}^g$ with a randomly sampled set of distractor nodes $\mathcal{V}^{\text{random}}$, where $\mathcal{V}^{\text{random}} \subset \mathcal{V}^{\text{KB}} \setminus \mathcal{V}^g$.

We partition the dataset into 2,898 workflows for training and 400 for testing. Based on the workflow description $desc$, we generate diverse user instructions $query$ using a suite of large language models, including Qwen-Max, Claude 3.5, Gemini 2.5 Pro, and GPT-4o. This process yields 10,677 training samples and 400 testing samples. Each sample consists of the user instruction $query$, the candidate node set $\mathcal{V}^{\text{cand}}$, and the target code representation c. The prompt for evaluation is detailed in Appendix C.

To ensure the validity of the evaluation set, five experienced ComfyUI developers perform rigorous human verification. Annotators upload and execute each workflow on the ComfyUI canvas to validate both functional correctness and semantic alignment with the user instructions. Each workflow is labeled twice, and the inter-annotator agreement is 93.5%, indicating high consistency in decisions. In cases of execution failure, developers debug the workflow by installing missing nodes and models, or adjusting the graph structure until the output is fully executable and instruction-compliant.

3.3 Evaluation Metrics

We establish a fine-grained evaluation protocol for workflow generation that assesses quality across multiple dimensions, ranging from basic format validity to hallucinations at the node, graph, and parameter levels. **Format Validity** rate serves as an initial check of the syntactic and structural correctness of the generated workflows. Specifically, it verifies whether all node names referenced in the workflow exist and whether the resulting structure constitutes a valid graph.

Following the evaluation protocol in WorFE-

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

val (Qiao et al., 2025), we quantitatively assess the alignment of the predicted and gold workflows to measure instruction following. Specifically, we identify the longest node chain via Longest Increasing Subsequence and the largest workflow subgraph via Maximum Common Induced Subgraph. Based on these matches, we compute and report **node-level and graph-level F1 scores**.

Finally, observations from our pilot study indicate that generated workflows often suffer from structural inconsistencies and hallucinated components that impede successful execution. To address these issues, we propose these following metrics:

- **Unique Connectivity**: The generated workflow is a single DAG without disconnected subgraphs.
- **Invalid Terminal Node**: The terminal node contains output parameters, indicating an incomplete workflow.
- **Undefined Variable**: There are undefined variables in the code representation, which prevents edges from correctly connecting nodes.
- **Illegal Parameters**: Input parameters provided to a node are not defined in its documentation (i.e., redundant or hallucinated inputs).
- **Missing Parameters**: Required input parameters for a node are absent, making the node nonfunctional.

We consolidate these metrics into an overall **Hallucination Pass Rate**. A generated workflow passes this evaluation only if it satisfies all the aforementioned criteria: it must form a single, well-connected DAG with valid, complete, and correctly specified parameters.

4 Experiments

4.1 Comparing Models

To comprehensively evaluate the AIGC workflow generation capabilities of current LLMs, we assess 7 widely used commercial models: GPT-4o (*gpt-4o-2024-11-20*), GPT-4.1 (*gpt-4.1-2025-04-14*), GPT-5.1 (*gpt-5.1-2025-11-13*), Claude 3.5 (*claude-3-5-sonnet-20241022*), Claude 3.7 (*claude-3-7-sonnet-20250219*), and Claude 4 (*claude-sonnet-4-20250514*). Additionally, we evaluate 6 state-of-the-art open-source LLMs, including the recent Deepseek V3.2 (DeepSeek-AI, 2025), Kimi-K2-Instruct (Team, 2025a), Llama-3.1-8B (Team, 2024), and the Qwen3 series models ranging from 8B to 32B parameters (Team, 2025b). Notably, the Qwen3 series supports both thinking and non-thinking modes. We further conduct SFT and RL on Llama-3.1-8B, Qwen3-8B, and Qwen3-14B, and report their post-training results.

4.2 Implementation Details

We evaluate all open-source models using the LlamaFactory (Zheng et al., 2024) framework. For models above 70B parameters, we utilize vLLM (Kwon et al., 2023) to accelerate inference. The decoding temperature is set to 1. The SFT training stage is conducted on 8×80G NVIDIA A100 GPUs for 1 epoch, with a learning rate of 1e-5 and a batch size of 1 per GPU.

During RL training, we employ a fine-grained rule-metric hybrid reward, covering format, graph structure, node and parameter fidelity, and node selection correctness. Details on the reward design are provided in Appendix D. Based on this hybrid reward, we train the models using the Group Relative Policy Optimization (GRPO) algorithm (Shao et al., 2024). We utilize 8×80G NVIDIA A100 GPUs to train for 300 steps, with a learning rate of 1e-6 and a global batch size of 64. The maximum context length is set to 32K tokens for both training and inference.

4.3 Main Results

Proprietary LLMs. Table 2 presents the performance of current LLMs on ComfyFlow. Top-tier proprietary models demonstrate strong capabilities in generating format-compliant workflows, exemplified by GPT-4o, which achieves a high format validity rate of 96%. However, even the strongest models fail to produce high-fidelity workflows. Although GPT-4o leads the proprietary category, its graph-level F1 score reaches only 0.37. Similarly, Gemini-2.5-Pro, despite being the top performer in hallucination avoidance, achieves a pass rate of only 0.72. *Therefore, ComfyFlow poses a substantial challenge for SOTA LLMs in generating instruction-following and executable workflows.*

Open-Source LLMs without post-training exhibit significantly poorer performance than commercial LLMs across all metrics. Models such as Llama-3.1-8B fail to adhere to correct formats, scoring a negligible 5% in format validity. Therefore, their functional metrics are severely impacted, with graph-level F1 scores limited to 0.21 for the Qwen3 series and nearly zero for Llama. *This indi-*

Models	Format Validity	Node F1	Graph F1	Halluc. Pass	Unique Conn.	Invalid Node	Undef. Variable	Illegal Param.	Miss. Param.
Proprietary Models									
GPT-4o	**0.96**	**0.55**	**0.37**	0.66	**0.87**	**0.26**	0.17	**0.04**	0.05
GPT-4.1	0.87	0.51	**0.37**	0.65	0.79	0.30	0.21	0.13	0.13
GPT-5.1	0.75	0.44	0.30	0.56	0.68	0.29	0.29	0.24	0.25
Claude 3.5	0.73	0.43	0.29	0.46	0.65	0.47	0.35	0.28	0.36
Claude 3.7	0.74	0.44	0.31	0.51	0.68	0.46	0.33	0.26	0.27
Claude 4	0.88	0.50	0.35	0.67	0.83	0.31	**0.16**	0.12	0.12
Gemini-2.5-Pro	0.81	0.48	0.35	**0.72**	0.76	**0.26**	0.25	0.20	0.21
Open-source Models									
Llama-3.1-8B	0.05	0.04	0.03	0.02	0.03	0.98	0.97	0.95	0.95
Llama-3.1-8B-SFT	0.90	0.55	0.46	0.65	0.80	0.33	0.11	0.11	0.11
Llama-3.1-8B-RL	**0.98**	0.53	0.37	0.57	0.75	0.42	**0.02**	**0.03**	**0.03**
Qwen3-8B	0.46	0.28	0.21	0.22	0.32	0.72	0.71	0.55	0.57
Qwen3-8B-Think	0.48	0.30	0.22	0.27	0.42	0.61	0.65	0.52	0.50
Qwen3-8B-SFT	0.87	0.53	0.45	0.65	0.78	0.33	0.16	0.14	0.13
Qwen3-8B-RL	0.89	0.51	0.35	0.72	0.83	0.27	0.13	0.12	0.12
Qwen3-14B	0.45	0.29	0.21	0.24	0.39	0.66	0.70	0.56	0.58
Qwen3-14B-Think	0.64	0.32	0.28	0.32	0.56	0.50	0.56	0.36	0.50
Qwen3-14B-SFT	0.93	**0.57**	**0.48**	0.76	0.87	0.21	0.08	0.09	0.08
Qwen3-14B-RL	0.96	0.56	0.40	**0.83**	**0.91**	**0.16**	0.06	0.04	0.05
Qwen3-32B	0.43	0.27	0.21	0.20	0.34	0.73	0.70	0.57	0.58
Qwen3-32B-Think	0.61	0.38	0.28	0.33	0.57	0.48	0.49	0.40	0.61
Deepseek V3.2	0.30	0.21	0.16	0.19	0.24	0.79	0.77	0.71	0.72
Kimi-K2-Instruct	0.51	0.32	0.24	0.25	0.40	0.64	0.62	0.49	0.62

Table 2: Overall results on the ComfyFlow benchmark. Higher values indicate better performance for Format Validity, Node-level F1, Graph-level F1, Hallucination Pass Rate, and Unique Connectivity, while lower values are desirable for error metrics (Invalid Terminal Node, Undefined Variable, Illegal Parameters, and Missing Parameters).

cates that open-source models lack the prior knowledge to navigate the complex node-link definitions required for valid workflow construction.

Thinking Mode. As the Qwen3 series supports inference in both thinking and non-thinking modes, we evaluate performance under both settings. We observe that the non-thinking versions across the 8B, 14B, and 32B scales perform consistently poorly, yielding graph-level F1 scores of 0.21 and format validity below 46%. Switching to thinking mode consistently lifts performance. For example, Qwen3-14B-Think demonstrates an absolute gain of 19 points in format validity and 8 points in hallucination pass rate. However, scaling model size in thinking mode yields no significant gains: increasing from 14B to 32B results in no improvement in graph-level F1 or pass rates. *We conclude that while scaling model parameters yields limited benefits, test-time scaling via long chain-of-thought enhances workflow generation performance.*

Post-Training. Different post-training strategies yield complementary effects, creating a trade-off between instruction adherence and format validity. SFT excels at instruction following, with SFT variants consistently achieving the highest F1 scores. Notably, Qwen3-14B-SFT reaches a graph-level F1 of 0.48, significantly surpassing its RL counterpart and even GPT-4o. Conversely, RL proves superior at lifting format validity and reducing hallucinations. Qwen3-14B-RL achieves the highest hallucination pass rate across all models, while Llama-3.1-8B-RL attains a near-perfect format validity of 98%. *This indicates that SFT memorizes node chains and subgraphs, whereas RL ensures high format compliance and low hallucination rates through rule-based rewards.*

Fine-grained Hallucination Analysis. Breaking down the hallucination metrics reveals a clear hierarchy in the difficulty of error types. Across all model categories, structural errors, specifically Invalid Terminal Nodes, prove to be the most persistent bottleneck compared to parameter errors. For example, even the top-performing proprietary model, GPT-4o, exhibits an Invalid Node rate of 26%, which is significantly higher than its Illegal or Missing Parameter rates. This trend is more pronounced in post-trained open-source models. While Llama-3.1-8B-RL successfully minimizes Undefined Variables to 2%, its Invalid Terminal Node rate remains high at 42%. *This suggests that while LLMs can effectively learn local parameter and variable constraints through post-training, they struggle with the global topological planning required to use valid termination nodes.*

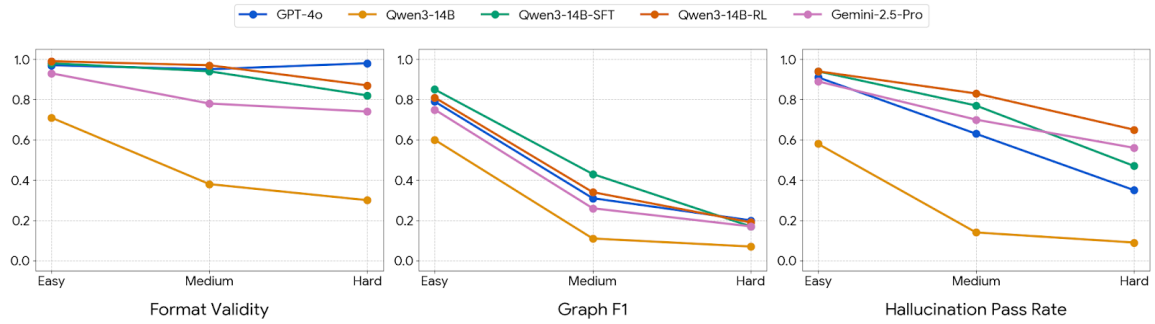


Figure 3: Performance comparison of GPT-4o, Gemini-2.5-Pro and Qwen3-14B variants across three difficulty levels. Full results are reported in Table 5 in Appendix.

Model	Format	Node F1	Graph F1	Halluc. Pass
Qwen3-8B	0.46	0.28	0.21	0.22
Qwen3-8B-SFT	0.87	**0.53**	**0.45**	0.65
Qwen3-8B-RL	**0.92**	**0.53**	0.34	**0.77**
- w/o Node Fidelity	0.91	0.52	0.36	0.72
- w/o Param Fidelity	0.81	0.49	0.34	0.67
Qwen3-8B SFT+RL	0.86	**0.53**	0.43	0.73

Table 3: Ablation study of different RL settings and reward configurations.

4.4 Analysis

Difficulty To assess model performance across varying levels of complexity, we categorize the workflows based on the number of nodes. We split the dataset into three groups using the 33rd and 66th percentiles of the node distribution as thresholds: Easy ($\leq$ 10 nodes), Medium (10–25 nodes), and Hard ($>$ 25 nodes). As shown in Figure 3, while performance across all metrics generally degrades as complexity increases, the severity of this decline varies by model and metric. The degradation in format validity is smaller compared to other two metrics. GPT-4o keeps a high format validity, all above 95% at all subsets. However, in hallucination pass rate, GPT-4o achieves a dominant 91% on easy tasks but suffers a sharp decline to 35% on hard tasks. Qwen3-14B-RL shows remarkable resilience, dropping only 29 percentage points (94% $\rightarrow$ 65%). *In conclusion, long-horizon planning in the hard subset remains a significant challenge for SOTA LLMs, and RL appears to handle this complexity more effectively.*

RL Settings We experiment with 3 different settings of RL and the results are in Table 3. Based on the reward used in Table 2, we conduct an ablation study with the node and parameter fidelity rewards (i.e. R_{node} and R_{param} in Appendix D) to demonstrate the effects of reward granularity. While removing the node fidelity reward results in no evident changes other than to the hallucination pass rate, removing the parameter fidelity reward leads to a significant drop in format validity and pass rate. This indicates that penalizing parameter hallucinations is critical for model reliability.

We further experiment with RL training initialized on the SFT checkpoint (SFT+RL). While the SFT and RL variants each possess distinct strengths regarding instruction following and hallucination pass rate respectively (as discussed in Sec. 4.3), the hybrid SFT+RL approach balances these trade-offs in Table 3, recovering most of graph-level F1 performance while maintaining a high pass rate. *In conclusion, given a large workflow KB and training set, there are many ways for future post-training exploration, such as training recipes, reward design, and RL algorithms.* More analysis on model robustness is provided in Appendix E.

5 Conclusion

In this paper, we propose ComfyFlow, a comprehensive and high-quality dataset for automated AIGC workflow generation across 20 diverse task types. ComfyFlow provides a large-scale knowledge base and 10K training examples, supported by a fine-grained evaluation protocol. Our experiments demonstrate that even top-tier models like GPT-4o face significant challenges, achieving a pass rate of only 35% on hard subset requiring long-horizon planning. However, by leveraging our open-source training data for post-training, smaller models can achieve new SOTA results, with RL effectively mitigating hallucinations and ensuring format. By providing high-quality data and an evaluation suite, ComfyFlow paves the way for future work to transition from human-defined workflows toward intelligent collaborative AI systems.

Limitations

Although ComfyFlow provides high-quality workflow data with extensive AIGC task coverage and a well-established evaluation scheme, current methods to AIGC workflow generation are limited to the ComfyUI platform. Future work could incorporate additional platforms, such as OpenAI Agent Builder, Dify, and Coze, to improve the generalization of the training recipe. Furthermore, workflow generation should ideally operate in a multi-turn mode; incorporating external feedback (e.g., execution quality or debugging information) could significantly refine the process. Therefore, building an agent system based on post-trained specialized models could be a promising direction for future exploration.

Ethical Considerations

We strictly exclude Personally Identifiable Information (PII) and NSFW content from our training and evaluation datasets. Our knowledge bases are exclusively curated from public documentation, synthetically generated, or derived from open-source platforms under permissive licenses. We advocate for the responsible deployment of our models and datasets, specifically guarding against misuse such as the creation of deepfakes or deceptive content. To foster transparency and reproducibility, we plan to release our training data and model checkpoints under open licenses, aiming to facilitate future research in this emerging field.

Acknowledgments

We thank the reviewers for their efforts in improving our paper. This work was supported by National Natural Science Foundation of China (Grant No. 62422603 and No. 625B2061), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2024B0101050003), and Shenzhen Science and Technology Program (Grant No. ZDSYS20230626091203008).

References

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2024. Chateval: Towards better LLM-based evaluators through multi-agent debate. In *The Twelfth International Conference on Learning Representations*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*.

Yulin Chen, Haoran Li, Yuan Sui, Yufei He, Yue Liu, Yangqiu Song, and Bryan Hooi. 2025. Can indirect prompt injection attacks be detected and removed? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18189–18206, Vienna, Austria. Association for Computational Linguistics.

comfyanonymous. 2023. Comfyui. `https://github.com/comfyanonymous/ComfyUI`.

DeepSeek-AI. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc.

Shengda Fan, Xin Cong, Yuepeng Fu, Zhong Zhang, Shuyan Zhang, Yuanwei Liu, Yesai Wu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2025. WorkflowLLM: Enhancing workflow orchestration capability of large language models. In *The Thirteenth International Conference on Learning Representations*.

Rinon Gal, Adi Haviv, Yuval Alaluf, Amit H. Bermano, Daniel Cohen-Or, and Gal Chechik. 2024. Comfygen: Prompt-adaptive workflows for text-to-image generation. *Preprint*, arXiv:2410.01731.

Litao Guo, Xinli Xu, Luozhou Wang, Jiantao Lin, Jinsong Zhou, Zixin Zhang, Bolan Su, and Ying-Cong Chen. 2025. Comfymind: Toward general-purpose generation via tree-based planning and reactive feedback. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *Preprint*, arXiv:2402.01680.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2021. Lora: Low-rank adaptation of

large language models. In *International Conference on Learning Representations*.

Shengran Hu, Cong Lu, and Jeff Clune. 2025. Automated design of agentic systems. In *The Thirteenth International Conference on Learning Representations*.

Oucheng Huang, Yuhang Ma, Zeng Zhao, Mingrui Wu, Jiayi Ji, Rongsheng Zhang, Zhipeng Hu, Xiaoshuai Sun, and Rongrong Ji. 2025. Comfygpt: A self-optimizing multi-agent system for comprehensive comfyui workflow generation. *Preprint*, arXiv:2503.17671.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Yunxin Li, Xinyu Chen, Shenyuan Jiang, Haoyuan Shi, Zhenyu Liu, Xuanyu Zhang, Nanhao Deng, Zhenran Xu, Yicheng Ma, Meishan Zhang, Baotian Hu, and Min Zhang. 2025a. Uni-moe-2.0-omni: Scaling language-centric omnimodal large model with advanced moe, training and data. *Preprint*, arXiv:2511.12609.

Yunxin Li, Shenyuan Jiang, Baotian Hu, Longyue Wang, Wanqi Zhong, Wenhan Luo, Lin Ma, and Min Zhang. 2024a. Uni-moe: Scaling unified multimodal llms with mixture of experts. *Preprint*, arXiv:2405.11273.

Yunxin Li, Zhenyu Liu, Zitao Li, Xuanyu Zhang, Zhenran Xu, Xinyu Chen, Haoyuan Shi, Shenyuan Jiang, Xintong Wang, Jifang Wang, Shouzheng Huang, Xinping Zhao, Borui Jiang, Lanqing Hong, Longyue Wang, Zhuotao Tian, Baoxing Huai, Wenhan Luo, Weihua Luo, and 3 others. 2025b. Perception, reason, think, and plan: A survey on large multimodal reasoning models. *Preprint*, arXiv:2505.04921.

Yunxin Li, Haoyuan Shi, Baotian Hu, Longyue Wang, Jiashun Zhu, Jinyi Xu, Zhen Zhao, and Min Zhang. 2024b. Anim-director: A large multimodal model powered agent for controllable animation video generation. In *SIGGRAPH Asia 2024 Conference Papers*, SA '24, New York, NY, USA. Association for Computing Machinery.

Oscar Mañas, Pietro Astolfi, Melissa Hall, Candace Ross, Jack Urbanek, Adina Williams, Aishwarya Agrawal, Adriana Romero-Soriano, and Michal Drozdzal. 2024. Improving text-to-image consistency via automatic prompt optimization. *Preprint*, arXiv:2403.17804.

Qian Ning, Weisheng Dong, Guangming Shi, Leida Li, and Xin Li. 2021. Accurate and lightweight image super-resolution with model-guided deep unfolding network. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):240–252.

Boye Niu, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu. 2025. Flow: Modularized agentic workflow automation. In *The Thirteenth International Conference on Learning Representations*.

OpenAI. 2023. Planning for AGI and beyond. Accessed: 2026-01-02.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2023. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *Preprint*, arXiv:2307.01952.

Chen Qian, Xin Cong, Wei Liu, Cheng Yang, Weize Chen, Yusheng Su, Yufan Dang, Jiahao Li, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. *Preprint*, arXiv:2307.07924.

Shuofei Qiao, Runnan Fang, Zhisong Qiu, Xiaobin Wang, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2025. Benchmarking agentic workflow generation. *Preprint*, arXiv:2410.07869.

Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. 2025. Agentsquare: Automatic LLM agent search in modular design space. In *The Thirteenth International Conference on Learning Representations*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Haoyuan Shi, Yunxin Li, Xinyu Chen, Longyue Wang, Baotian Hu, and Min Zhang. 2025. Animaker: Multi-agent animated storytelling with mcts-driven clip generation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, SA Conference Papers '25, New York, NY, USA. Association for Computing Machinery.

Dominik Sobania, Martin Briesch, and Franz Rothlauf. 2024. Comfygi: Automatic improvement of image generation workflows. *Preprint*, arXiv:2411.14193.

Kimi Team. 2025a. Kimi k2: Open agentic intelligence. *Preprint*, arXiv:2507.20534.

Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Qwen Team. 2025b. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jifang Wang, Xue Yang, Longyue Wang, Zhenran Xu, Yiyu Wang, Yaowei Wang, Weihua Luo, Kaifu Zhang, Baotian Hu, and Min Zhang. 2025. A unified agentic framework for evaluating conditional image generation. In *Proceedings of the 63rd Annual Meeting of*

the Association for Computational Linguistics (Volume 1: Long Papers), pages 12626–12646, Vienna, Austria. Association for Computational Linguistics.

Minghao Wu, Jiahao Xu, and Longyue Wang. 2024a. TransAgents: Build your translation company with language agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 131–141, Miami, Florida, USA. Association for Computational Linguistics.

Minghao Wu, Yulin Yuan, Gholamreza Haffari, and Longyue Wang. 2024b. (perhaps) beyond human translation: Harnessing multi-agent collaboration for translating ultra-long literary texts. *Preprint*, arXiv:2405.11804.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 10 others. 2023. The rise and potential of large language model based agents: A survey. *Preprint*, arXiv:2309.07864.

Jia Xu, Weilin Du, Xiao Liu, and Xuejun Li. 2024. Llm4workflow: An llm-based automated workflow model generation tool. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, ASE '24, page 2394–2398, New York, NY, USA. Association for Computing Machinery.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. 2023. Towards reasoning in large language models via multi-agent peer review collaboration. *Preprint*, arXiv:2311.08152.

Zhenran Xu, Longyue Wang, Jifang Wang, Zhouyi Li, Senbao Shi, Xue Yang, Yiyu Wang, Baotian Hu, Jun Yu, and Min Zhang. 2025a. Filmagent: A multi-agent framework for end-to-end film automation in virtual 3d spaces. *Preprint*, arXiv:2501.12909.

Zhenran Xu, Xue Yang, Yiyu Wang, Qingli Hu, Zijiao Wu, Longyue Wang, Weihua Luo, Kaifu Zhang, Baotian Hu, and Min Zhang. 2025b. ComfyUI-copilot: An intelligent assistant for automated workflow development. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 632–643, Vienna, Austria. Association for Computational Linguistics.

Xiangyuan Xue, Zeyu Lu, Di Huang, Zidong Wang, Wanli Ouyang, and Lei Bai. 2025. Comfybench: Benchmarking llm-based agents in comfyui for autonomously designing collaborative ai systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24614–24624.

Rui Ye, Shuo Tang, Rui Ge, Yaxin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. 2025. MAS-GPT: Training LLMs to build LLM-based multi-agent systems. In *Forty-second International Conference on Machine Learning*.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2025. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

A Task Types

We leverage GPT-4o to classify the workflows into 6 types and 20 subtypes based on their functional descriptions. Each workflow is assigned one to two subtype labels. Following previous work (Guo et al., 2025), the type taxonomy is in Table 4.

Task	Subtask
Image Generation	Guided Generation (Control/Reference) Artistic/Stylistic, Photorealistic Face/Portrait & Avatar Generation Text-to-Image Generation
Image Editing	Local & Face Editing Background Replacement Inpainting/Outpainting Style/Look Transfer Product/Fashion Virtual Try-On Image Merge Relighting/Recoloring
Enhancement & Restoration	Super-Resolution/Upscaling General & Face Enhancement Denoising/DeBlur
Multimodal Analysis	Captioning/Tagging Vision-Language Segmentation
Video Gen. & Editing	Text-to-Video/Animation Generation Image-to-Video/Animation Generation Video Editing/Inpaint/Outpaint
3D Generation	3D & Geometry

Table 4: Taxonomy of the task types.

B Example of Node Documentation

Here we present an example of node documentation. The documentation for the commonly-used *Image Batch* node is as follows.

ImageBatch Documentation

ImageBatch (function name: ImageBatch(...))
The ImageBatch node is designed for combining two images into a single batch. If the dimensions of the images do not match, it automatically rescales the second image to match the first one's dimensions before combining them.
Input
- **`image1`** (`IMAGE`, required): The first image to be combined into the batch. It serves as the reference for the dimensions to which the second image will be adjusted if necessary.
- **`image2`** (`IMAGE`, required): The second image to be combined into the batch. It is automatically rescaled to match the dimensions of the first image if they differ.
Output
- **`IMAGE`** (`IMAGE`): The combined batch of images, with the second image rescaled to match the first one's dimensions if needed.
Example

```
IMAGE_3 = ImageBatch (image1 = IMAGE_1, image2 = IMAGE_2) # ImageBatch
```

C Prompt Design

Here we present the prompt utilized during evaluation. The prompt consists of the user instruction $query$ and the documentations of candidate node set $\mathcal{V}^{\text{cand}}$.

D Reward Design

We propose a fine-grained rule-metric hybrid reward R_{final}, including output format reward R_{format}, structure reward R_{DAG}, node fidelity R_{node}, parameter fidelity R_{param}, precision reward R_{correct}.

Format reward (R_{format}). This reward verifies whether the output response adheres to the expected structure, including the chosen nodes enclosed within the "<selected_nodes>...</selected_nodes>". The final workflow code is wrapped within "<workflow>...</workflow>". A format score of 0 is given if all required tags are present and their contents can be successfully extracted. Otherwise, a penalty is applied. The format reward function R_{format} is defined as follows:

$$R_{\text{format}} = \begin{cases} 0 & \text{if all required fields appear} \\ -1 & \text{otherwise} \end{cases} \tag{1}$$

Structure reward (R_{DAG}). This reward assesses whether the parsed "<workflow>" section of the generated output forms a valid DAG. A penalty is assigned if the structure is not a valid DAG.

$$R_{\text{DAG}} = \begin{cases} 0 & \text{if the structure is a valid DAG} \\ -1 & \text{otherwise} \end{cases} \tag{2}$$

Node fidelity reward (R_{node}). This component penalizes the inclusion of hallucinated or inconsistent nodes during node selection and workflow

Models	Easy			Medium			Hard		
	Format	Graph F1	Halluc.	Format	Graph F1	Halluc.	Format	Graph F1	Halluc.
Proprietary Models									
GPT-4o	**0.97**	**0.79**	**0.91**	**0.95**	**0.31**	0.63	**0.98**	**0.20**	0.35
GPT-4.1	**0.97**	0.77	**0.91**	0.85	0.29	0.63	0.79	0.18	0.36
Claude 3.5	0.85	0.63	0.76	0.67	0.22	0.42	0.72	0.16	0.18
Claude 3.7	0.83	0.64	0.77	0.70	0.23	0.46	0.76	0.16	0.35
Claude 4	0.94	0.71	0.89	0.87	0.26	0.66	0.85	0.17	0.42
Gemini-2.5-Pro	0.93	0.75	0.89	0.78	0.26	**0.70**	0.74	0.17	**0.56**
Open Source Models									
Llama-3.1-8B	0.15	0.11	0.09	0.02	0.01	0.00	0.01	0.00	0.00
Llama-3.1-8B-SFT	0.97	0.78	0.86	0.92	0.41	0.67	0.76	0.16	0.27
Llama-3.1-8B-RL	**1.00**	0.67	0.77	**0.97**	0.35	0.53	**0.96**	0.18	0.38
Qwen3-8B	0.79	0.60	0.62	0.39	0.11	0.12	0.23	0.04	0.02
Qwen3-8B-SFT	0.96	0.81	0.90	0.88	0.40	0.65	0.73	0.15	0.31
Qwen3-8B-RL	0.97	0.77	0.92	0.89	0.28	0.71	0.76	0.16	0.44
Qwen3-14B	0.71	0.60	0.58	0.38	0.11	0.14	0.30	0.07	0.09
Qwen3-14B-SFT	0.98	**0.85**	**0.94**	0.94	**0.43**	0.77	0.82	0.17	0.47
Qwen3-14B-RL	0.99	0.81	**0.94**	**0.97**	0.34	**0.83**	0.87	**0.19**	**0.65**

Table 5: Performance comparison of proprietary and open-source models across Easy, Medium, and Hard difficulty levels. Metrics reported are Format Validity, Graph F1, and Hallucination Pass Rate. Best scores in each category are marked in bold.

Evaluation Prompt

{query}

Knowledge Base
Below is the metadata for the nodes that may be used in the workflow:
{node_documentation}

Format
1. **Node Selection**: First, list the function names of the selected nodes as a Python List [str] (e.g., ["CR_Overlay_Text", ...]), and enclose them within `<selected_nodes>` tags.
2. **Workflow Construction**: Next, present the workflow as Python code, where each line should invoke a node function with appropriately named input and output parameters. Enclose the code within `<designed_workflow>` tags.

Please provide your response in the format specified above.

generation. A penalty of -1 is applied in either of the following cases:

- **Invalid Nodes**: Any node listed in the predicted "<selected_nodes>" block $\mathcal{V}^p$ is not present in the provided candidate nodes $\mathcal{V}^{\text{cand}}$.
- **Inconsistent Nodes**: The set of nodes listed in "<selected_nodes>" block does not exactly match the set of nodes parsed from the '<workflow>" block.

$$\mathcal{R}_{\text{node}} = \begin{cases} -1, & \text{if invalid or inconsistent nodes are detected} \\ 0, & \text{otherwise} \end{cases}$$

Parameter fidelity reward (R_{param}). This component penalizes the inclusion of hallucinated parameters in the code representation of workflows. A penalty of -1 is applied if illegal or missing input parameters are detected as detailed in Sec 3.3:

$$R_{\text{param}} = \begin{cases} -1 & \text{if illegal param. } \vee \text{ missing param.} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Node Selection Accuracy (R_{correct}). This reward measures the overlap between generated node set $\mathcal{V}^p$ and the ground truth node set $\mathcal{V}^g$, reflecting the model's ability to select the correct nodes for the workflow.

$$R_{\text{correct}} = \frac{|\mathcal{V}^p \cap \mathcal{V}^g|}{|\mathcal{V}^g|} - 1 \tag{4}$$

Combining the above rewards The total reward R_{final} aggregates all the individual reward components using a veto-based mechanism. If any of R_{format}, R_{DAG}, R_{node} or R_{param} is -1, indicating a fundamental error in output format, structural integrity, or fidelity, the total reward is immediately

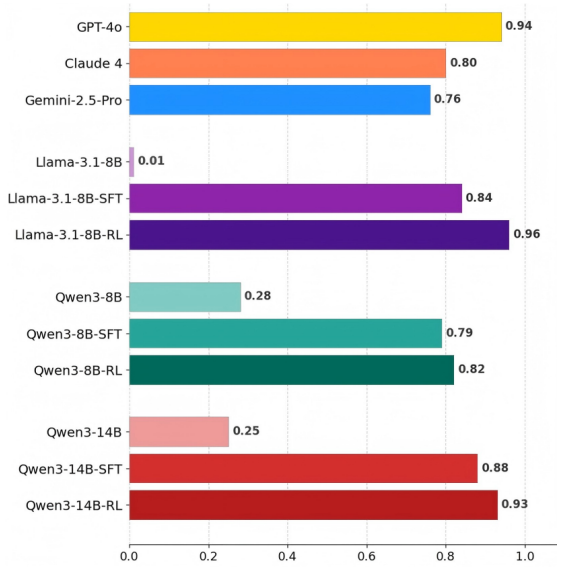


Figure 4: Comparison of robustness scores.

set to -1, preventing any positive reward for an invalid output. Otherwise, the total reward is computed based on the node selection accuracy.

$$R_{\text{final}} = \begin{cases} -1, & \text{if } R_{\text{format}} = -1 \text{ or } R_{\text{DAG}} = -1 \\ & \text{or } R_{\text{node}} = -1 \text{or } R_{\text{param}} = -1 \\ \dfrac{3 + R_{\text{correct}}}{3.0}, & \text{otherwise} \end{cases}$$

E Robustness Analysis

To preliminarily evaluate the robustness of workflow generation, for each sample in the evaluation set, we modify the set of distractor nodes $\mathcal{V}^{\text{random}}$ to form a new set of candidate nodes $\mathcal{V}^{\text{cand}}$. We calculate robustness based on format validity: for each task, if the output format remains valid both before and after this modification, the model is considered robust for this task. Figure 4 displays the experimental results. Among closed-source models, GPT-4o achieves a high robustness score of 94%. While open-source models initially lag behind, post-training bridges this gap. Both SFT and RL provide substantial gains in robustness. Notably, RL consistently outperforms SFT; for example, Llama-3.1-8B-RL achieves a score of 96%, surpassing even GPT-4o. *This demonstrates that applying RL elevates model robustness.*