LAG: LLM agents for Leaderboard Auto Generation on Demanding

Anonymous ACL submission

Abstract

This paper introduces Leaderboard Auto Generation (LAG), a novel and well-organized framework for automatic generation of leaderboards on a given research topic in rapidly evolving fields like Artificial Intelligence (AI). Faced with a large number of AI papers updated daily, it becomes difficult for researchers to track every paper's proposed methods, experimental results, and settings, prompting the need for efficient automatic leaderboard construction. While large language models (LLMs) offer promise in automating this process, challenges such as multi-document summarization, leaderboard generation, and experiment fair comparison still remain under exploration. LAG solves these challenges through a systematic approach that involves the paper collection, experiment results extraction and integration, leaderboard generation, and quality evaluation. Our contributions include a comprehensive solution to the leaderboard construction problem, a reliable evaluation method, and experimental results showing the high quality of leaderboards.

1 Introduction

The explosive growth of scientific publications has created both unprecedented opportunities and significant challenges for researchers seeking to stay abreast of state-of-the-art methods (Bornmann et al., 2020; Wang et al., 2024; Şahinuç et al., 2024). Leaderboard platforms, such as NLP-progress[1] and Papers-With-Code[2] have become invaluable by offering comprehensive overviews of recent research developments, highlighting ongoing trends, and identifying future directions. However, the large amount of daily papers makes it increasingly difficult to update these leaderboards automatically and promptly. Figure 1 illustrates two pressing issues: First, the number of LLM-related articles submitted to arXiv has surged dramatically—from 2022 to 2025, with over 20,000 submissions in 2024 alone. Second, even as new methods continuously emerge, leaderboards, such as the one for Multi-hop Question Answering on the HotpotQA(Yang et al., 2018) dataset, remain stagnant, with the latest method dating back to 2023. These observations highlight a serious issue: the rapid accumulation of daily scientific publications often outpaces the capability of researchers to keep up with cutting-edge research and state-of-the-art methods, emphasizing the growing need for more efficient methods to generate the latest and useful leaderboards.

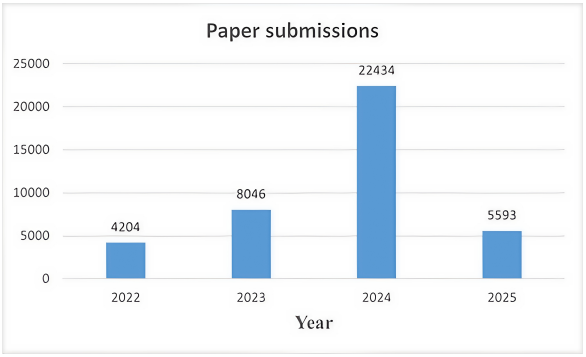
Prior efforts have attempted to address this gap. A line of work (Hou et al., 2019; Kardas et al., 2020) has proposed leaderboard construction methods that directly extract scientific entities from individual NLP papers, and construct a static leaderboard without updating and maintenance. Semi-supervised scientific NER, proposed by Li et al. (2023), focuses on extracting scientific entities from both tables and text. Şahinuç et al. (2024) introduce SCILEAD, a manually-curated Scientific Leaderboard dataset, including 27 leaderboards derived from 43 NLP papers. However, all previous methods have been limited to the extracted scientific entities and only give a static snapshot after extracting information from a narrow selections.

We consider using LLMs such as GPT-4 (Achiam et al., 2023), Qwen (Yang et al., 2024), and O1-preview which have demonstrated exceptional performance across diverse NLP tasks, especially in the long-context scenario (Chen et al., 2023a,b; Wang et al., 2023b) to automatically generate leaderboards based on given research topic.

Directly applying LLMs to this task still faces several key challenges. First, **Limited Paper Coverage**: It is challenging for human to search for all papers on a certain scientific topic, due to the overwhelming number of constantly emerging publications. Second, **Unfair Comparison**: Current studies do not consider fair experiment settings when

[1] https://nlpprogress.com/

[2] https://paperswithcode.com/



Leaderboard (Distractor Setting)

In the *distractor* setting, a question-answering system reads 10 paragraphs to provide an answer (Ans) to a question. They must also justify these answers with supporting facts (Sup).

	Model	Code	Ans		Sup		Joint	
			EM	F_1	EM	F_1	EM	F_1
1 Aug 7, 2023	Beam Retrieval (single model) *BUPT & Tencent* (Zhang, Zhang, Zhang, et al. 2023)		**72.69**	**85.04**	**66.25**	**90.09**	**50.53**	**77.54**
2 Jul 7, 2022	PipNet (single model) *Tencent Cloud Xiaowei*		72.26	84.86	63.71	89.41	48.76	76.95
3 Jun 27, 2022	Smoothing R3 (single model) *Fudan University & Huawei Poisson Lab* Rethinking Label Smoothing on Multi-hop Question Answering		72.07	84.34	65.44	89.55	49.73	76.69
4 Jan 28, 2022	FE2H on ALBERT (single model) *Nanjing University* From Easy to Hard: Two-stage Selector and Reader for Multi-hop Question Answering		71.89	84.44	64.98	89.14	50.04	76.54
5 May 16, 2022	R3 (single model) *Fudan University & Huawei Poisson Lab* Rethinking Label Smoothing on Multi-hop Question Answering		71.27	83.57	65.25	88.98	49.81	76.02

Figure 1: Left: Growth trend of paper submission on LLMs from 2022 to 2025-02. Right: An Example of a Multi-hop QA dataset leaderboard (HotpotQA Homepage), the latest method is still stuck in 2023.

making comparisons. For example, in NLP research, key experimental components, model size, train dataset size, and hyperparameter selection, vary significantly across publications, highlighting the need for automatic alignment. Finally, **Low Timeliness**: A leaderboard, which lacks regular updates and continuous maintenance, cannot provide researchers with sufficient useful information.

To this end, we introduce LAG, a novel agent framework for dynamically and automatically generating leaderboards. Figure 2 illustrates the framework of our method, which is organized into four stages: (1) **Paper Collection and split**: Initially, LAG automatically download all relevant LaTex code based on the given research topic from arXiv and filter out papers published before certain date and those unrelated to the topic, ensuring proper paper coverage and timeliness. (2) **Table Extraction and Classification**: We use LLMs to extract and classify experiment tables based on accompanying table descriptions. (3) **Table Unpacking and Integration**: LAG extracts the datasets, metrics, experiment settings, and experiment results from the tables in the form of a quintuple, including paper title. Experiment setting extraction is crucial for enabling fair comparisons across different baselines. (4) **Leaderboard Generation and Evaluation**: The extracted quintuples are recombined and re-ranked to form candidate leaderboards.

To evaluate the performance of LAG, we propose two key quality dimensions for assessment: (1) Topic-related Quality: paper coverage assessment, determining whether each quintuple in the LAG-generated leaderboards is related to the given topic. (2) Content Quality: We adopt the LLM-as-Judge method for leaderboard quality evaluation on four aspects, including Coverage, Structure, Latest, and Multiaspect. We also introduce human experts to manually evaluate LAG-generated leaderboards and compute the Pearson Correlation Coefficient between human- and LLMs-assigned scores.

Extensive experiments across different leaderboard lengths (5, 10, 15, and 20 items) show that LAG consistently achieves high topic-related and content quality scores. With 20 items, a LAG-generated Leaderboard represents 20 baselines for researchers, achieving 67.58% recall and 70.33% precision scores in topic-related quality. In content quality with 20 items, LAG achieves 4.12 coverage, 3.96 latest, 4.16 structure, and 4.08 multiaspect scores, approaching human performance (4.72 coverage, 4.68 latest, 4.34 structure, and 4.58 multiaspect scores). Although the manually created leaderboard achieves higher content quality, it is much more time-consuming than LAG, deeming the efficiency. With fewer items, LAG gets even higher performance, slightly lower than human performance. These results highlight the effectiveness of LAG, providing a reliable proxy for human judgment across varying leaderboard items. Furthermore, the Pearson correlation coefficient values indicate a moderate positive correlation between the human-assigned and LLM-assigned scores.

To our best knowledge, LAG is the first method to explore the potential of LLM agents for automatic leaderboard generation, proposing evaluation criteria that align with human preferences and offering valuable reference for future related research.

Related Work	Data Source	Experiment Settings	Multi Document	Dynamic
TDMS(Hou et al., 2019)	NProg.	✗	✗	✗
Axcell (Kardas et al., 2020)	PwC	✗	✗	✗
TELIN (Yang et al., 2022)	PwC	✗	✗	✗
ORKG(KABENAMUALU et al., 2023)	PwC	✗	✗	✗
LEGO (Singh et al., 2024)	PwC	✗	✗	✗
SciLead (Şahinuç et al., 2024)	NLP papers	✗	✓	✗
LAG (Ours)	arXiv	✓	✓	✓

Table 1: Comparison of related work and ours. **Data Source**: Source of leaderboards: NProg.: *NLP-progress*, PwC: *paperswithcode*. **Experiment Settings**: whether the experiment settings are extracted as part of leaderboards or not. **Multi Document**: whether the leaderboards are constructed from multiple papers or not. **Dynamic**: whether the generated leaderboards can be updated dynamically or not.

2 Related Work

LLM for Scientific Research. In the realm of LLMs, several studies have explored using LLMs for improving work efficiency in scientific research. Baek et al. (2024) and Yang et al. (2023) proposed a multi-agent-based scientific idea generation method to boost AI-related research. To evaluate the quality of LLM-generated ideas, Si et al. (2024) introduced a comprehensive human evaluation metric. Wang et al. (2023a) proposed SciMON, a method that uses LLMs for scientific literature retrieval. Wang et al. (2024) proposed an LAG to automatically generate scientific surveys based on the given research topic. The AI Scientist, Lu et al. (2024) introduced a fully automated, prompt-driven research pipeline. To make LLM-generated ideas more diverse and practical, Weng et al. (2024) proposed CycleResearcher, an iterative self-rewarding framework that allows the LLM to refine its ideas continuously, enhancing both diversity and practicality in research proposal generation. However, no previous research focused on the leaderboard generation for researchers to search, organize, and compare the state-of-the-art methods rapidly and fairly based on a certain research topic.

Leaderboard Construction. Table 1 illustrates the differences between the previous work and LAG. First of all, previous work builds leaderboards by using data sources such as NLP-progress or Papers-With-Code. However, these sources lack rigorous quality assurance, such as standardizing scientific entities across different leaderboards and ensuring complete coverage of relevant publications. Instead, we choose arXiv, which is a free distribution service and an open-access archive for nearly 2.4 million scholarly articles in different domains, providing a large amount of publications for researchers. Similar to our work, Hou et al. (2019), Kardas et al. (2020), and Singh et al. (2024) extract "Task", "Dataset", "Model" along with the experiment result entities as TDM triples to build a leaderboard. Yang et al. (2022) and KABENAMUALU et al. (2023) leverage the pre-defined TDM triples in an extraction process similar to Hou et al. (2019). Since these approaches require a pre-defined taxonomy of TDM triples, they are incompatible with realistic task definitions. In short, none of the previous work is adaptable to constantly emerging benchmarks driven by new research and innovation. Moreover, none of the studies extract the experiment settings as additional information to generate leaderboards, which results in a lack of fair comparison. In scientific research, experiment settings are important for educators or users to reproduce the experimental results claimed in scientific publications. In this work, we address the aforementioned problems. Specifically, we (1) dynamically download scientific publications and generate up-to-date leaderboards based on the given scientific topic and the specific date; (2) extract experiment settings as part of leaderboards for fair comparison; (3) apply a Multi-Agent-as-Judge to evaluate leaderboard quality.

3 Methods

Figure 2 depicts LAG, which consists of four stages: Paper Collection and Split, Table Extraction and Classification, Table Unpacking and Integration, and Leaderboard Generation and Evaluation. Each stage is meticulously designed to address specific challenges associated with leaderboard generation, thereby enhancing the efficiency and quality of the resulting leaderboards. The whole process is iterated several times (e.g., five times) to generate a high-quality leaderboard.

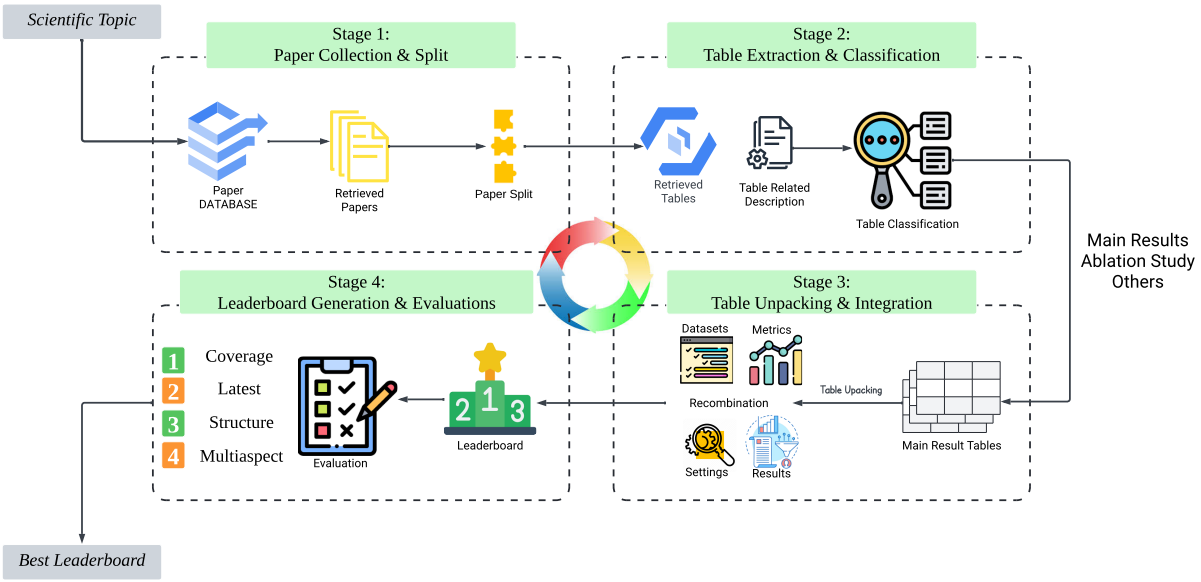


Figure 2: The LAG framework for leaderboard automatic generation. In Stage 1, we automatically crawl scientific papers from arXiv. In Stage 2, we retrieve, extract, and classify tables from the latex code. In Stage 3, we select the main results tables and extract datasets, metrics, results, and experiment settings from the main results table. In Stage 4, we generate Leaderboards from the selected results and evaluate the quality.

3.1 Paper Collection and Split

Utilizing the off-the-shelf tools [3], LAG first searches and retrieves a set of papers $P_{\text{init}} = \{P_1, P_2, ..., P_N\}$ from arXiv and downloads LaTeX code files related to a specific scientific research topic T. Then, we specify a certain date and filter out all papers published before the date. The filtering stage is important for ensuring that the generated leaderboards are grounded in the most relevant and recent research. Moreover, since the search tool just identifies only the keywords in the paper title and abstract, which can lead to a significant amount of noisy data, we also introduce a retrieval model to filter out papers that are irrelevant to the given topic and retrieve topic-related papers. The set of filtered papers $P_{\text{filtered}} = \{\text{Retrieval}\{P_1, P_2, ..., P_U\}\}$ is used to generate the leaderboards, ensuring comprehensive coverage of the topic and logical structure. Due to the extensive number of relevant papers retrieved and filtered during this stage, the total input length of P_{filtered} often exceeds the maximum input length of LLMs. Since most of the LaTex content is unproductive for generating leaderboards, we split the LaTeX code into several sections based on the structure of each paper. Most tables, table-related descriptions, experiment results, and experiment settings are located in the "Experiment" section, which contains the key information for generating leaderboards. Consequently, we select all "Experiment" sections as well as all tables $\{\text{Table}_1, \text{Table}_2, ..., \text{Table}_U\}$ and all table-related descriptions $\{D_1, D_2, ...D_U\}$, extracted from all papers, as input for the next stage.

3.2 Table Extraction and Classification

Typically, a scientific paper, such as those in the natural language processing domain, contains several types of tables, including "Main Results", "Ablation Study", and "Others". The "Main Results" tables are the most important tables in the paper, which illustrate the novelty, contributions, and effectiveness of the proposed methods or models by comparing the experiment results of the proposed method with other baselines. We utilize these tables for leaderboard generation. The "Ablation Study" tables examine the effect of damaging or removing certain components in a controlled setting to investigate all possible outcomes of system failure. The "Others" tables are the tables that illustrate the supplementary information of the experiments. For example, some tables illustrate the dataset statistics of the benchmark used in the experiments, while other tables list the results of "Case Study" and "Error Analysis". To address this, we propose an agent that uses the In-Context Learning method (Dong et al., 2022) to manually select one table from each of the three different types. The agent then prompts

[3] https://github.com/lukasschwab/arxiv.py

LLMs to classify the table types and only keeps the "Main Experiments" tables and their descriptions as the final input. The i_{th} table types can be described as: $\texttt{LLM}(\text{Table}_i, D_i; \text{Prompt}) \rightarrow \texttt{Table type}$.

In practice, the most intrinsic approach is to divide Stage 2 into the following sequential steps: (1) Extract all tables and their associated captions from the LaTeX code. (2) Classify the extracted tables according to predefined table types. (3) Extract metrics, performance values, and experimental settings related to the proposed model from tables categorized as "Main Results". Moreover, each of these three steps necessitates the use of LLM APIs, and repeated reference to certain table contents further exacerbates the substantial waste of tokens. To address this issue, we create the agent following the few-shot Chain of Thought (CoT) prompting process, enabling it to classify and extract information from identified "Main Results" tables in a single dialogue round. Specifically, in the requested JSON output, we additionally set the key points as follows: "number of tables (Int)", "classification of tables (Dict)" and "selected table's index (Int)".

3.3 Table Unpacking and Integration

Following the table extraction and classification phase, each table $Table_i$ is sent into the LLM to extract the core information. To build a useful and high-quality leaderboard, we define four types of scientific terms: Datasets, Metrics, Experiment Results, and Experiment Settings. For datasets, we use LLMs to count the frequency in all filtered papers P_{filtered} of each dataset under a certain research topic and retain the top-K (K=5) datasets with the highest frequency of occurrence in scientific papers. For the rest of the three scientific terms, we utilize LLMs to extract from given $Table_i$ with a related table description D_i. After scientific term extraction, we recombined them into a quintuple, including the paper title as the unique identification ID. Each paper can produce one quintuple and finally we get a raw leaderboard with M quintuples from M filtered papers. The raw leaderboard is reranked on the basis of the experiment results.

3.4 Leaderboard Generation and Evaluation

After we obtain K leaderboards based on top-K frequent datasets, the final stage involves a quality evaluation based on our pre-defined four criteria, which is shown in Table 4 in Appendix. Each leaderboard is assigned three scores based on "Coverage", "Latest" and "Structure". Since a research topic may contain several datasets, the "Multi-Aspect" is the average quality score that is used to evaluate the LLM-generated leaderboards for each dataset. The best leaderboard is chosen from N candidates. LLMs critically examine the leaderboards in several aspects. The final output of Leaderboard is $L_{\text{best}} = \text{Evaluate}(L_{\text{ca1}}, L_{\text{ca2}}, ..., L_{\text{caN}})$.

The methodology outlined here, from paper collection to leaderboard evaluation, ensures that LAG effectively addresses the complexities of leaderboard generation in the AI domain using advanced LLM agents. We provide Pseudo-code for easily understanding, which is shown in Algorithm 1.

Algorithm 1 Leaderboard Automatic Generation.

Input: Scientific topic T, open-access platform arXiv A
Output: Final refined and evaluated leaderboard L
 # Stage 1: Paper Collection and Document Split
Crawl topic T related N publications $P_{\text{init}} = \{P_1, ...P_N\} \leftarrow \text{Retrieve}(T, A)$
Filter out topic-unrelated and old papers, $P_{\text{filtered}} = \{P_1, ...P_M\} \leftarrow \text{Retrieve}(P_{\text{init}}, \text{date}, \text{topic})$
 # Stage 2: Table Extraction and Classification
for each Leaderboard iteration $i = 1$ to $Iters$ **do**
  Count frequency of all datasets and retain top-K datasets from U papers.
  for each dataset $j = 1$ to K **do**
    Split P_i, Extract U Tables $\{\text{Table}_1, ..., \text{Table}_U\}$ and table-related description $\{D_1, ...D_U\}$.
    Classify each table and keep "Main Results Table".
    for each main table and table description **do**
      Extract Paper title, Dataset, Metrics, Experiment Settings, and Experiment Results as quintuple.
    end for
    # Stage 3: Leaderboard Generation
    Recombine all quintuples and rank the quintuples by performance scores.
    Output the Candidate Leaderboard L_{ca}
  end for
end for
 # Stage 4: Quality Evaluation and Iteration
Evaluate and select the best leaderboard $L_{\text{best}} \leftarrow \text{Evaluate}(L_{ca1}, L_{ca2}, \ldots, L_{caN})$
Output: Refined and evaluated leaderboard L_{best}

4 Experiments

We designed experiments for LAG, aiming to answer four questions: RQ-1: Can LAG address the paper coverage issue and generate fair leaderboards by incorporating the latest baselines? RQ-2: Can LAG reduce time consumption? RQ-3: Is the evaluation consistent between LAG and human experts? RQ-4: Is each proposed component of LAG useful?

4.1 Experimental Setup

We evaluated LAG's performance by testing its ability to generate leaderboards for specific topics

Leaderboard Length (items)	Topic-related Quality Recall	Topic-related Quality Precision	Model	Speed$_{/s}$	Content Quality Coverage	Content Quality Latest	Content Quality Structure	Content Quality Multiaspect
5	$76.57_{\pm 11.65}$	$79.43_{\pm 8.86}$	Qwen2.5-7B	131.43	$3.60_{\pm 0.48}$	$3.46_{\pm 0.49}$	$3.18_{\pm 0.32}$	3.41
			Qwen2.5-14B	129.51	$4.23_{\pm 0.38}$	$4.14_{\pm 0.31}$	$3.68_{\pm 0.29}$	4.02
			GPT4-o	49.64	$4.52_{\pm 0.42}$	$4.70_{\pm 0.32}$	$4.32_{\pm 0.38}$	4.41
			O1-preview	79.67	$4.63_{\pm 0.48}$	$4.71_{\pm 0.71}$	$4.40_{\pm 0.33}$	4.58
			Human Writing	355	4.89	4.83	4.91	4.88
10	$75.19_{\pm 9.81}$	$80.05_{\pm 6.76}$	Qwen2.5-7B	156.41	$3.22_{\pm 0.48}$	$3.41_{\pm 0.49}$	$4.11_{\pm 0.39}$	3.57
			Qwen2.5-14B	163.54	$3.91_{\pm 0.48}$	$3.55_{\pm 0.49}$	$3.41_{\pm 0.39}$	4.61
			GPT4-o	88.96	$4.68_{\pm 0.39}$	$4.59_{\pm 0.33}$	$4.45_{\pm 0.41}$	4.56
			O1-preview	98.44	$4.40_{\pm 0.48}$	$4.46_{\pm 0.71}$	$4.31_{\pm 0.33}$	4.39
			Human Writing	612	4.81	4.72	4.65	4.72
15	$71.34_{\pm 8.39}$	$74.58_{\pm 7.35}$	Qwen2.5-7B	183.45	$3.11_{\pm 0.28}$	$3.23_{\pm 0.26}$	$3.15_{\pm 0.27}$	3.16
			Qwen2.5-14B	195.63	$3.68_{\pm 0.28}$	$3.32_{\pm 0.19}$	$3.18_{\pm 0.24}$	3.39
			GPT4-o	105.61	$4.47_{\pm 0.22}$	$4.16_{\pm 0.27}$	$4.32_{\pm 0.24}$	4.28
			O1-preview	109.33	$4.21_{\pm 0.48}$	$4.06_{\pm 0.21}$	$4.28_{\pm 0.31}$	4.18
			Human Writing	839	4.71	4.65	4.44	4.60
20	$67.58_{\pm 9.12}$	$70.33_{\pm 6.89}$	Qwen2.5-7B	196.33	$3.03_{\pm 0.25}$	$3.11_{\pm 0.31}$	$2.98_{\pm 0.25}$	3.16
			Qwen2.5-14B	208.64	$3.49_{\pm 0.34}$	$3.17_{\pm 0.26}$	$3.03_{\pm 0.28}$	3.39
			GPT4-o	120.52	$4.28_{\pm 0.28}$	$3.92_{\pm 0.22}$	$4.21_{\pm 0.25}$	4.13
			O1-preview	117.45	$4.12_{\pm 0.38}$	$3.96_{\pm 0.25}$	$4.16_{\pm 0.29}$	4.08
			Human Writing	1128	4.72	4.68	4.34	4.58

Table 2: Results of leaderboard quality generated by LLMs in the first iteration. **Leaderboard Length**: The number of items in the leaderboard. For example, a 5-item leaderboard contains 5 baselines. **Topic-related Quality**: The precision and recall of each paper in relation to its relevance to the topic. **Speed**: The average time required to generate a single leaderboard. **Content Quality**: The evaluation results of the leaderboard content.

across various complex settings.

4.1.1 Evaluation Metrics

We use two metrics to evaluate the quality (topic-related and leaderboard content) and speed of leaderboard generation, respectively, in response to the three challenges mentioned in the introduction.

(1) Topic-related Quality: The aforementioned arXiv crawler employs regular expression matching in the abstract section to identify papers related to specified topics. While this method is efficient, it is relatively rudimentary and cannot guarantee that all retrieved papers meet our requirements. The quality of these papers not only directly affects the final leaderboard, but low-quality candidate papers can also significantly prolong the time required for construction. Therefore, it is essential to evaluate the quality of the retrieved articles. We evaluate the quality of content from the following two aspects. **(i) Recall:** It measures whether all items in the generated leaderboard are related to the given research topic. **(ii) Precision:** It identifies irrelevant items, ensuring that the items in the generated leaderboards are pertinent and directly support the given research topic.

(2) Leaderboard Content Quality: The evaluation metric of leaderboard content quality includes four aspects. Each aspect is judged by LLMs according to a 5-point, calibrated by human experts. The evaluation criteria are listed in Table 4. **(i) Coverage**: Assess each paper represented on the LAG-generated leaderboards encapsulates all aspects of the topic. **(ii) Latest**: Test whether all papers represented on the LAG-generated leaderboards are latest. **(iii) Structure**: Evaluate the logical organization and determine whether LAG leaderboards are missing any items. **(iv) Multi-aspect**: Average score of the previous three criteria for LAG-generated leaderboards.

(3) Leaderboard Construction Speed: Manually building a leaderboard is a time-consuming and laborious task. This process can be divided into the following main components: T_r (search for papers on a specific topic), T_b (browse all retrieved articles and develop several highly frequent datasets), T_f (filter candidate articles based on the selected datasets), T_e (read and extract information), and T_c (the integration and construction time). And the total time consumption can be calculated as:

$$T_{\text{manual}} = T_r + T_b + T_f + T_e + T_c. \quad (1)$$

Given L denotes the length of the leaderboard, $N_{\text{retrieved}}$ number of retrieved articles, N_{filtered} number of articles retained, and P the proportion of valid articles with $P = \frac{N_{\text{filtered}}}{N_{\text{retrieved}}}$. We find that T_b and T_f are strongly correlated with leaderboard length L and the Topic-related quality:

$$\{T_b, T_f\} \propto \frac{L}{P} = \frac{L \cdot N_{\text{retrieved}}}{N_{\text{filtered}}}. \quad (2)$$

While T_r is relatively fixed, T_e and T_c usually only have a positive correlation with L.

For LAG, we barely account for all the invocation time of the agents' API calls. Compared to manual work, which often takes several days, LAG reduces the total time cost in the minute level. This is largely attributed to the task decomposition conducted in this paper, the division of labor and scheduling among agents, and the superior performance of the LLMs.

4.1.2 Baselines

We employ proprietary and open-source LLMs in our experiments and set the temperature to 0.7 for proprietary models. For proprietary models, we adopt GPT-4o (Achiam et al., 2023), and the O1-preview. For open-source LLMs, we adopt Qwen2.5-7B and Qwen2.5-14B (Yang et al., 2024). We provide a detailed illustration of our designed prompts for different stages in Appendix A.

4.2 Experiment Results

4.2.1 Performance Comparison (RQ-1)

Topic-related Quality Evaluation: Table 2 illustrates the Topic-related Quality LAG achieved a recall of 67.58% and a precision of 70.33% with 20 items, indicating that it successfully retrieved a large proportion of relevant papers while maintaining a low rate of irrelevant ones. This performance is crucial for ensuring that the generated leaderboards are both comprehensive and accurate. The high precision and recall scores illustrate that LAG could help solve the paper coverage problem.

Fair Comparison: To ensure fair comparison, LAG extracted all experiment settings as part of the LAG-generated leaderboards. We provide a detailed case study of LAG-generated leaderboards with experiment settings in Appendix B.

Content Quality Evaluation: Table 2 presents the results of leaderboard quality generated by LAG and the baselines. LAG consistently achieved high scores across all evaluation metrics, particularly in terms of Coverage and Latest, indicating its ability to include a wide range of relevant and recent papers. For example, at a leaderboard length of 20 items, LAG achieved a Coverage score of 4.12 and a Latest score of 3.96, approaching human performance (4.72 and 4.68, respectively). While manual leaderboards scored slightly higher in content quality, LAG significantly reduced the time required for leaderboard generation, demonstrating its efficiency.

Iteration Evaluation: To ensure the high-quality of LAG-generated leaderboards, we iterated the process to evaluate the performance change during the whole iteration. Figure 3 presents the effect of different iteration counts on the performance of LAG. The results show that increasing the number of iterations from 1 to 5 provides a significant improvement in structure quality and Coverage quality scores. However, the latest score remains at a relatively high level, which is because in stage 1 of LAG, the old papers are filtered out.

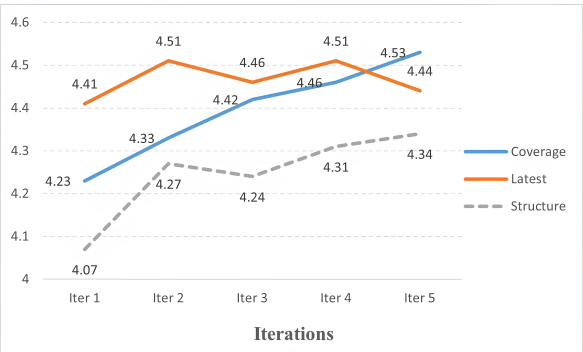


Figure 3: Impact of Iteration on LAG Performance.

Our experiments demonstrate that LAG is highly effective in generating high-quality, up-to-date leaderboards across various research topics. The framework's ability to dynamically update leaderboards and extract detailed experiment settings ensures a fair comparison between state-of-the-art baselines. While LAG's content quality scores are slightly lower than those of manually created leaderboards, its efficiency and scalability make it a valuable tool for researchers in rapidly evolving fields like AI and NLP.

4.2.2 Efficiency Analysis (RQ-2)

Construction Speed: LAG dramatically reduced the time required to generate leaderboards compared to manual methods. For instance, generating a 20-item leaderboard with LAG took approximately 120 seconds, while manual construction took over 18 minutes. This speed advantage makes LAG a practical tool for researchers who need up-to-date leaderboards in rapidly evolving fields. The high speed of LAG illustrates that LAG could help generate high-quality leaderboards timely.

4.2.3 Meta Evaluation (RQ-3)

To verify the consistency between our proposed LLM evaluation strategy and human evaluation, we conduct a correlation evaluation involving human experts and our automated evaluation method. Human experts judge pairs of generated leader-

Methods	Leaderboard Length (items)	Speed$_{/s}$	Content Quality			
			Coverage	Latest	Structure	Multiaspect
LAG *w/o* Table Classification	5	42.35	4.43	4.52	3.95	4.30
LAG *w/o* Refinement		43.58	4.41	4.46	4.05	4.31
LAG		49.64	4.52	4.70	4.32	4.51
LAG *w/o* Table Classification	10	81.37	4.31	4.36	4.01	4.33
LAG *w/o* Refinement		80.52	4.24	4.17	3.88	4.10
LAG		88.96	4.68	4.59	4.45	4.56
LAG *w/o* Table Classification	15	93.28	4.13	4.08	3.61	3.94
LAG *w/o* Refinement		91.32	4.19	4.13	3.72	4.01
LAG		105.61	4.47	4.16	4.32	4.28
LAG *w/o* Table Classification	20	105.31	3.92	3.88	3.51	3.77
LAG *w/o* Refinement		99.35	3.85	3.71	3.62	3.72
LAG		120.52	4.28	3.92	4.21	4.13

Table 3: Ablation Study for LAG with different components removed. We use the GPT-4o as the backbone of LAG.

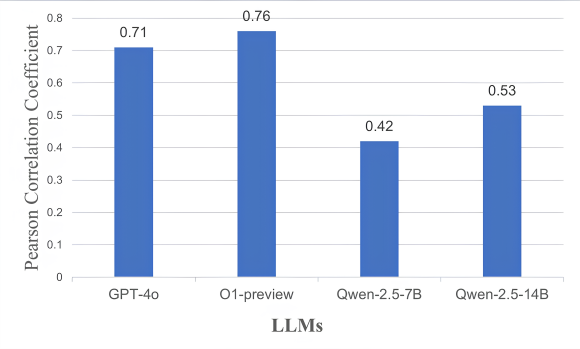


Figure 4: Pearson Correlation Coefficient values given by four LLMs and human experts. Note that the Pearson Correlation Coefficient is between -1 and 1, the larger value indicates more positive correlations.

boards to determine which one is superior. We compare the judgments made by our method against those made by human experts. Specifically, we provide experts with the same scoring criteria used in our evaluation for reference. The experts rank the 20 LAG-generated leaderboards, and we compare these rankings with those generated by the LLM using the Pearson Correlation Coefficient to measure consistency between human and LLM evaluations.

The results of this meta-evaluation are presented in Figure 4. The table shows the Pearson Correlation Coefficient values, indicating the degree of correlation between the rankings given by each LLM and the human experts. The Pearson correlation coefficient values indicate a strong positive correlation between the quality scores provided by the LLM and those given by human experts, with the O1-preview achieving the highest correlation at **0.76**. These results suggest that our evaluation method aligns well with human preferences, providing a reliable proxy for human evaluation.

4.2.4 Ablation Study (RQ-4)

To understand the contribution of each component in LAG, we conducted an ablation study by removing key components of LAG as follows: (1) LAG w/o Table Classification: We removed the table classification step, which led to a slight decrease in Structure and Multiaspect scores, indicating that classifying tables is essential for maintaining a logical and well-organized leaderboard. (2) LAG w/o Refinement: We disabled the Refinementing step, which resulted in a minor drop in Coverage and Latest scores, suggesting that Refinementing helps refine the leaderboard by ensuring that only the most relevant and recent papers are included. As shown in Table 3, the results of the ablation study confirm that each component of the LAG plays a crucial role in achieving the generation of the high-quality leaderboard.

5 Conclusion

We introduce LAG, a novel agent framework leveraging large language models to automatically generate the latest, and high-quality leaderboards based on given research topics. LAG addresses key challenges including paper coverage, fair comparison, and timeliness through a systematic approach involving paper collection and split, table extraction and classification, table unpacking and integration, and leaderboard generation and evaluation. Experiments showed that LAG can automatically generate new, high-quality leaderboards in a relatively short time and match human performance topic-related quality and content qulaity. This advancement offers a scalable and effective solution for synthesizing the latest leaderboards, providing a valuable tool for researchers in rapidly evolving fields like artificial intelligence.

Limitations

One limitation of LAG is its reliance on the quality of the retrieved papers. While our topic-related quality metrics are strong, there is still room for improvement in ensuring that all relevant papers are included. Future work could explore more sophisticated retrieval models to further enhance the coverage of the generated leaderboards. Another limitation is, a specific dataset may contain several evaluation metrics, and different papers may use different metrics to evaluate proposed models' performance, bringing challenges for leaderboard generation and baseline comparison.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2024. Researchagent: Iterative research idea generation over scientific literature with large language models. *ArXiv*, abs/2404.07738.

Lutz Bornmann, Robin Haunschild, and Ruediger Mutz. 2020. Growth rates of modern science: a latent piecewise growth curve approach to model publication numbers from established and new literature databases. *Humanities and Social Sciences Communications*, 8.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023a. Extending context window of large language models via positional interpolation. *ArXiv*, abs/2306.15595.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023b. Longlora: Efficient fine-tuning of long-context large language models. *ArXiv*, abs/2309.12307.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2022. A survey on in-context learning. In *Conference on Empirical Methods in Natural Language Processing*.

Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, and Debasis Ganguly. 2019. Identification of tasks, datasets, evaluation metrics, and numeric scores for scientific leaderboards construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5203–5213, Florence, Italy. Association for Computational Linguistics.

Salomon Kabongo KABENAMUALU, Jennifer D'Souza, and S. Auer. 2023. Orkg-leaderboards: a systematic workflow for mining leaderboards as a knowledge graph. *International Journal on Digital Libraries*, pages 1–14.

Marcin Kardas, Piotr Czapla, Pontus Stenetorp, Sebastian Ruder, Sebastian Riedel, Ross Taylor, and Robert Stojnic. 2020. AxCell: Automatic extraction of results from machine learning papers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8580–8594, Online. Association for Computational Linguistics.

Yuhan Li, Jian Wu, Zhiwei Yu, Börje F. Karlsson, Wei Shen, Manabu Okumura, and Chin-Yew Lin. 2023. All data on the table: Novel dataset and benchmark for cross-modality scientific information extraction.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob N. Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *ArXiv*, abs/2408.06292.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *ArXiv*, abs/2409.04109.

Shruti Singh, Shoaib Alam, Husain Malwat, and Mayank Singh. 2024. Legobench: Scientific leaderboard generation benchmark. In *Conference on Empirical Methods in Natural Language Processing*.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2023a. Scimon: Scientific inspiration machines optimized for novelty. In *Annual Meeting of the Association for Computational Linguistics*.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2023b. Augmenting language models with long-term memory. *ArXiv*, abs/2306.07174.

Yidong Wang, Qi Guo, Wenjin Yao, Hongbo Zhang, Xin Zhang, Zhen Wu, Meishan Zhang, Xinyu Dai, Min Zhang, Qingsong Wen, Wei Ye, Shikun Zhang, and Yue Zhang. 2024. Autosurvey: Large language models can automatically write surveys. *ArXiv*, abs/2406.10252.

Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. 2024. Cycleresearcher: Improving automated research via automated review. *ArXiv*, abs/2411.00816.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, and Shanghaoran Quan. 2024. Qwen2.5 technical report.

Sean Yang, Chris Tensmeyer, and Curtis Wigington. 2022. TELIN: Table entity LINker for extracting

leaderboards from machine learning publications. In *Proceedings of the first Workshop on Information Extraction from Scientific Publications*, pages 20–25, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and E. Cambria. 2023. Large language models for automated open-domain scientific hypotheses discovery. In *Annual Meeting of the Association for Computational Linguistics*.

Furkan Şahinuç, Thy Thy Tran, Yulia Grishina, Yufang Hou, Bei Chen, and Iryna Gurevych. 2024. Efficient performance tracking: Leveraging large language models for automated construction of scientific leaderboards. In *Conference on Empirical Methods in Natural Language Processing*.

A Example Prompts

The prompts of instructing LLMs in different stages of LAG are illustrated in Prompts 1, 2, and 3.

B Example LAG-generated Leaderboards

Figure 5 and 6 illustrate two examples generated by LAG. The papers in the first leaderboard are the latest methods of semi-supervised medical image segmentation on the LA dataset from March to December in 2024. The second leaderboard collects the most recent methods for image quality assessment conducted on the LIVE dataset from February 2022 to November 2024. To ensure that the table content is fully displayed, the "model & size" and "hyperparameters selection" within the experimental settings are presented beneath the paper titles.

First and foremost, when viewed holistically, both leaderboards with 20 entries, whether utilizing qwen2.5-14B or GPT-4o as the construction model, exhibit a notably high level of completeness. Upon specific analysis of the missing information, in Leaderboard 1, LAG failed to successfully extract the HD value from "Self-Paced Sample Selection for Barely-Supervised Medical Image Segmentation" (No. 11) because the metric was referred to as 95HD in the original text. Although our design accounts for such situations: our designed agent is required to extract metrics from both text and tables to avoid confusion caused by abbreviations. This design has successfully resolved most of the issues arising from abbreviations, but such errors still occur with a small probability. The absence of metrics in entries No. 13 and No. 20 is acceptable because the original text indeed lacks these metrics. The situation in Leaderboard 2 is similar; the only two missing items (No. 4 and No. 14) are also due to the absence of corresponding results in the original texts.

The higher missing rate in the 5-row leaderboard compared to the 20-row leaderboard for the LIVE dataset can be attributed to the following reasons: When only 5 papers are included, LAG extracts a larger number of metrics, including RMSE, mIoU, and mAcc. The missing values for these metrics are tolerable in a 5-row leaderboard. However, when expanding to a 20-row leaderboard, the excessive number of missing values forces LAG to discard these metrics to ensure that the leaderboard conveys meaningful information.

Secondly, regarding the experimental settings, we observe that in Leaderboard 1, the information on "model & size", "hyperparameters", and "training strategy" is both accurate and comprehensive. Notably, there is a consistent thread throughout the hyperparameters: the portion of labeled data. In

Semi-Supervised Medical Image Segmentation Leaderboard: LA dataset

Papers due: 2024 December

Latest 20 papers

No.	Model		Experimental Setting	Metrics			
	Model Name	Code	Training Strategy	Dice	Jaccard	95HD	ASD
1	**Uncertainty-Guided Cross Attention Ensemble Mean Teacher for Semi-supervised Medical Image Segmentation** UG-CEMT framework with V-Net backbone labeled data percentage of 20%, EWA decay rate of 0.99, dropout rate of 0.5, SAM radius of 0.5	GitHub	semi-supervised learning with uncertainty-guided consistency regularization	89.73	81.63	2.2	0.5
2	**Biologically-inspired Semi-supervised Semantic Segmentation for Biomedical Imaging** UNet-like architecture labeled data percentage of 20%	GitHub	two-stage semi-supervised approach	89.17	80.45	11.92	2.66
3	**GraphCL: Graph-based Clustering for Semi-Supervised Medical Image Segmentation** GraphCL with a 3D V-Net backbone labeled scans of 8 (10%), unlabeled scans of 72, alpha of 0.5, kappa of 0.01, tau of 2	-	Graph-based clustering with a teacher-student framework	90.24	82.31	6.42	1.71
4	**Leveraging CORAL-Correlation Consistency Network for Semi-Supervised Left Atrium MRI Segmentation** V-Net backbone labeled scans of 16, unlabeled scans of 64, batch size of 4, learning rate of 0.01, momentum of 0.9, weight decay of 0.0001	-	semi-supervised learning with CORAL-Correlation Consistency Network (CORN)	91.22	83.96	5.34	1.54
5	**Dual-Teacher Ensemble Models with Double-Copy-Paste for 3D Semi-Supervised Medical Image Segmentation** V-Net backbone labeled_ratio of 20%, similarity_threshold of 0.01, EMA_decay_rate of 0.99	GitHub	dual-teacher framework with staged selective ensemble and double-copy-paste strategy	91.82	84.92	5.11	1.5
6	**Affinity-Graph-Guided Contractive Learning for Pretext-Free Medical Image Segmentation with Minimal Annotation** Semi-AGCL framework Labeled of 5%, Unlabeled of 95%	-	Affinity-graph-guided semi-supervised contrastive learning	90.44	79.05	7.78	2.11
7	**Manifold-Aware Local Feature Modeling for Semi-Supervised Medical Image Segmentation** V-Net architecture alpha of 0.05	GitHub	semi-supervised learning with 10% labeled data	90.28	82.37	6.49	1.66
8	**SDCL: Students Discrepancy-Informed Correction Learning for Semi-supervised Medical Image Segmentation** VNet and ResNet labeled images of 8, unlabeled images of 72, batch size of 8, learning rate of 0.001, gamma of 0.5, mu of 0.05	GitHub	semi-supervised learning with discrepancy correction learning	92.35	85.83	4.22	1.44
9	**PMT: Progressive Mean Teacher via Exploring Temporal Consistency for Semi-Supervised Medical Image Segmentation** V-Net labeled percentage of 10%, EMA decay rate of 0.99, batch size of 4, iterations of 6000	GitHub	Progressive Mean Teacher framework with pseudo-label filtering and discrepancy-driven alignment	90.81	83.23	5.61	1.5
10	**Adaptive Mix for Semi-Supervised Medical Image Segmentation** V-Net labeled data percentage of 20%, mix-up patch size of 32, maximum number of mix-up patches of 16	GitHub	AdaMix-MT framework (Mean-Teacher paradigm)	91.87	85.36	5.53	1.65
11	**Self-Paced Sample Selection for Barely-Supervised Medical Image Segmentation** SPSS framework with 16 labeled slices learning rate of 0.01, iterations of 6000, decay of 0.1 every 2500 iterations	GitHub	self-paced sample selection framework with SU and SC components	86.19	75.89	-	3.49
12	**Leveraging Task-Specific Knowledge from LLM for Semi-Supervised 3D Medical Image Segmentation** V-Net backbone labeled data percentage of 10%, unlabeled data percentage of 90%	-	co-training framework with unified segmentation loss	91.45	84.31	4.66	1.62
13	**Rethinking Barely-Supervised Volumetric Medical Image Segmentation from an Unsupervised Domain Adaptation Perspective** V-Net labeled data percentage of 5%	GitHub	Barely-supervised learning via unsupervised domain adaptation (BvA)	87.4	-	-	2.37
14	**Leveraging Fixed and Dynamic Pseudo-labels for Semi-supervised Medical Image Segmentation** V-Net labeled data ratio of 5%, unlabeled data ratio of 95%	-	co-training framework with fixed and dynamic pseudo-labels	89.55	81.18	5.48	1.99
15	**CrossMatch: Enhance Semi-Supervised Medical Image Segmentation with Perturbation Strategies and Knowledge Distillation** V-Net labeled data percentage of 10%, confidence threshold (tau) of 0.85, distillation balance (eta) of 0.3	GitHub	Self-training with knowledge distillation and perturbation strategies	91.33	84.11	5.29	1.53
16	**Mixed Prototype Consistency Learning for Semi-supervised Medical Image Segmentation** V-Net backbone labeled scans of 16 (20%), unlabeled scans of 64 (80%), batch size of 4, learning rate of 0.01	-	Mixed Prototype Consistency Learning framework with Mean Teacher and auxiliary network	91.98	85.02	4.77	1.58
17	**An Evidential-enhanced Tri-Branch Consistency Learning Method for Semi-supervised Medical Image Segmentation** ETC-Net with V-Net backbone labeled scans of 8, unlabeled scans of 72, batch size of 4, learning rate of 0.1	GitHub	semi-supervised learning with evidential tri-branch consistency	91.15	83.8	5.45	1.65
18	**EPL: Evidential Prototype Learning for Semi-supervised Medical Image Segmentation** V-Net architecture learning rate of 0.001, batch size of 3, iterations of 10000	-	semi-supervised learning with 20% labeled data	92.3	85.72	4.73	1.38
19	**Uncertainty-aware Evidential Fusion-based Learning for Semi-supervised Medical Image Segmentation** V-Net labeled_ratio of 100%, unlabeled_ratio of 0%	-	semi-supervised learning with evidential fusion-based framework	92.62	85.24	4.47	1.33
20	**Guidelines for Cerebrovascular Segmentation: Managing Imperfect Annotations in the context of Semi-Supervised Learning** UA-MT (Uncertainty-Aware Mean-Teacher) learning rate of 0.01, final weight for consistency loss of 0.01	GitHub	semi-supervised learning with uncertainty-aware consistency regularization	89.51	81.01	-	-

Figure 5: A leaderboard (20 lines) of semi-supervised medical image segmentation on the LA dataset, using GPT4-o for table extraction and Qwen2.5-14B for leaderboard construction & refinement.

Image Quality Assessment Leaderboard: LIVE dataset

Papers due: 2024 November

Latest 20 papers

	Model		Experimental Setting	Metrics	
No.	Model Name	Code	Training Strategy	SROCC	PLCC
1	**Dual-Representation Interaction Driven Image Quality Assessment with Restoration Assistance** DRI-IQA model	GitHub	Dual-Representation Interaction method with restoration assistance	0.982	0.984
2	**Study of Subjective and Objective Quality in Super-Resolution Enhanced Broadcast Images on a Novel SR-IQA Dataset** ARNIQA model	-	5-fold cross-validation	0.86	0.911
3	**Exploring Rich Subjective Quality Information for Image Quality Assessment in the Wild** RichIQA model with three-stage quality prediction network	-	multi-label training strategy using MOS, DOS, and SOS	0.8943	0.9121
4	**Q-Ground: Image Quality Grounding with Large Multi-modality Models** Mask2Former	GitHub	semantic segmentation finetuning	-	-
5	**Dual-Branch Network for Portrait Image Quality Assessment** Dual-Branch Network with Swin Transformer-B backbones	GitHub	Pre-trained on LSVQ and GFIQA datasets, followed by learning-to-rank optimization	0.85	0.86
6	**Cross-IQA: Unsupervised Learning for Image Quality Assessment** ViT (Vision Transformer) with Cross-IQA pretraining	-	unsupervised pretraining followed by fine-tuning	0.965	0.976
7	**Deep Bi-directional Attention Network for Image Super-Resolution Quality Assessment** BiAtten-Net	GitHub	Bi-directional attention mechanism for full-reference IQA	0.981	0.982
8	**High Resolution Image Quality Database** HR-BIQA model with modified ResNet50 and ViT	GitHub	patch-based BIQA model designed for high-resolution images	0.92	0.925
9	**Deep Shape-Texture Statistics for Completely Blind Image Quality Evaluation** EfficientNet-b7	-	Shape-Texture Adaptive Fusion (STAF) module with shape and texture CNN branches	0.935	0.931
10	**JOINT DEEP IMAGE RESTORATION AND UNSUPERVISED QUALITY ASSESSMENT** QAIRN (Quality-Aware Image Restoration Network)	-	Joint restoration and unsupervised quality assessment	0.879	0.87
11	**Perceptual Assessment and Optimization of HDR Image Rendering** HDR-NeRF with multilayer perceptron (MLP)	GitHub	Perceptual optimization using HDR quality metrics	0.869	0.873
12	**Blind Image Quality Assessment via Transformer Predicted Error Map and Perceptual Quality Token** ViT-B/16 (Vision Transformer backbone)	GitHub	Pre-training on KADID-10K dataset followed by fine-tuning on LIVE dataset	0.976	0.977
13	**Gap-closing Matters: Perceptual Quality Evaluation and Optimization of Low-Light Image Enhancement** IACA (Illumination Aware and Content Adaptive model)	GitHub	Deep learning-based IQA model trained on SQUARE-LOL database	0.875	0.878
14	**Explainable Image Quality Assessments in Teledermatological Photography** EfficientNet-B0, 15 MB	-	supervised learning with class-weighted training	-	-
15	**Image Quality Assessment with Gradient Siamese Network** Gradient Siamese Network (GSN)	-	Trained on the entire KADID-10k dataset and tested on LIVE dataset	0.932	0.922
16	**DeepWSD: Projecting Degradations in Perceptual Space to Wasserstein Distance in Deep Feature Space** DeepWSD with VGG16 backbone	GitHub	No training with quality labels, pre-trained network	0.9624	0.9609
17	**Perceptual Quality Assessment for Fine-Grained Compressed Images** Proposed method with gradient-based and texture-based features	-	Full-reference image quality assessment (FR-IQA) method	0.973	0.9612
18	**SPQE: Structure-and-Perception-Based Quality Evaluation for Image Super-Resolution** SPQE metric with HR as reference	-	end-to-end training with adaptive tradeoff mechanism	0.9317	0.9641
19	**Multi-Scale Features and Parallel Transformers Based Image Quality Assessment** MSFPT-avg (Multi-Scale Features and Parallel Transformers)	GitHub	Full-Reference IQA with multi-scale feature extraction and parallel transformers	0.977	0.972
20	**Content-Variant Reference Image Quality Assessment via Knowledge Distillation** CVRKD-IQA with FR-teacher	GitHub	Knowledge distillation from FR-teacher to NAR-student	0.973	0.969

Figure 6: A leaderboard (20 lines) of image quality assessment on the LIVE dataset, using GPT4-o for both table extraction and leaderboard construction & refinement.

Image Quality Assessment Leaderboard: LIVE dataset

Papers due: 2024 November

Latest 5 papers

	Model		Experimental Setting	Metrics				
No.	Model Name	Code	Training Strategy	SROCC	PLCC	RMSE	mIoU	mAcc
1	**Dual-Representation Interaction Driven Image Quality Assessment with Restoration Assistance** DRI-IQA model learning rate of 2e-4, batch size of 64	GitHub	Dual-Representation Interaction method with restoration assistance	0.982	0.984	-	-	-
2	**Study of Subjective and Objective Quality in Super-Resolution Enhanced Broadcast Images on a Novel SR-IQA Dataset** ARNIQA model scaling factor x2, iterations 1000	-	5-fold cross-validation	0.86	0.911	0.699	-	-
3	**Exploring Rich Subjective Quality Information for Image Quality Assessment in the Wild** RichIQA model with three-stage quality prediction network Adam optimizer with an initial learning rate of 0.00001, batch size of 8	-	multi-label training strategy using MOS, DOS, and SOS	0.8943	0.9121	8.2312	-	-
4	**Q-Ground: Image Quality Grounding with Large Multi-modality Models** Mask2Former learning rate of 0.0003, batch size of 2	GitHub	semantic segmentation finetuning	-	-	-	0.403	0.646
5	**Dual-Branch Network for Portrait Image Quality Assessment** Dual-Branch Network with Swin Transformer-B backbones Initial learning rate of 1e-5, batch size of 12	GitHub	Pre-trained on LSVQ and GFIQA datasets, followed by learning-to-rank optimization	0.85	0.86	-	-	-

Figure 7: A leaderboard (5 lines) of image quality assessment on the LIVE dataset, using GPT4-o for both table extraction and leaderboard construction & refinement.

Table 4: Leaderboard Quality Criteria.

Criteria	Scores
Coverage	The ratio of the number of papers used for leaderboard generation to the total number of papers searched. $(P_{used}/P_{total}) * 5$
Latest	The ratio of the number of papers published after the certain date to the total number of papers searched. $(P_{new}/P_{total}) * 5$
Structure	*Score 1*: The structure of the leaderboard lacks logic, making it difficult to understand and navigate. The table header and each row are not clearly organized and connected. *Score 2*: The structure of the leaderboard have some contents arranged in a disordered or unreasonable manner. However, the overall structure is reasonable and coherent. *Score 3*: The survey is generally comprehensive in coverage but still misses a few key points that are not fully discussed. *Score 4*: The structure of the leaderboard is generally reasonably logical, with most header items arranged orderly, though some header items may be repeated or redundant. *Score 5*: The structure of the leaderboard has good logical consistency, with each line strictly related to the header items and the previous line. But it can be optimized in terms of easy understanding.
Multi-Aspect	The evaluation metric for multi-leaderboard. Specifically, a research topic T may have N different datasets, and thus we can get N leaderboards, the score of the Multi-Aspect is computed based on the average of all the N scores. $(N_{Coverage} + N_{Latest} + N_{Structure})/(3 * N)$.

```
{
    "title": "The title of the paper (String)",
    "number of tables": "The number of tables in the paper, denoted as n (Int)",
    "classification of tables": "The classification of tables in the paper,
      including 4 types: main result/comparison tables(0), ablation tables(1),
      hyperparameter tables(2), and others(3). You should output a dictionary with
      the number of tables and their corresponding types, formed as {"0":0, "1":2,
      ..., "n-1":3} (Dict)",
    "selected table's index": "The index of the main result table focused on the
      specified dataset [SPECIFIED DATASET], denoted as i (Int)",
    "metrics": "The evaluation metrics chosen to assess performance of the method
      proposed in this paper. This information is extracted from the textual portion
       of the 'Experimental' related section (String)",
    "selected table's metrics": "Metrics used in the selected main result table, it
      should be almost the same as the metrics extracted from the textual. Remove
      the latex format syntax (String)",
    "selected table's core results": "A dictionary only containing this paper's
      model best performance on the selected dataset, with the metrics as keys and
      the corresponding values (Dict)",
    "selected table's settings (model & size)": "In computer vision, the model
      usually means the backbone architecture of the network, such as ResNet, ViT,
      and so on. The size can be omitted if not specified. In NLP, the model and
      size are usually organized as a string, such as 'LLAMA-7B', 'GPT-3', and so on
       (String)",
    "selected table's settings (training strategy)": "Training strategy usually
      refers to the concepts like: fine-tuning, transfer learning, linear-probing,
      reinforce learning, one-shot, few-shot, prompt-learning, semi/self supervised
      and so on (String)",
    "selected table's settings (hyperparameter selection)": "The hyperparameters
      used in the model, such as learning rate, batch size, and so on. You should
      output a dictionary with the hyperparameters and their values (Dict)",
    "github": "The link to the gitHub repository containing the code for this paper,
       if available (String)"
}
```

Table 5: The example JSON file of the table extraction agent with table classification COT.

contrast, Leaderboard 2 discards the hyperparameter information compared to Leaderboard 3. This is because we require LAG to extract hyperparameter information in a way that not only maintains completeness but also focuses on the intrinsic connections between different items. If the deviation is too large (i.e., if it cannot provide users with a concise and effective summary), the information should be discarded. Therefore, when the number of input papers for LAG increases from 5 to 20, the hyperparameter settings in the topic of image quality assessment do not have a clear and unified theme, and thus are ultimately ignored.

C Cost Analysis

We calculate the average number of input & output tokens required to generate a 20-entry leaderboard, along with the cost analysis using different LLMs, as shown in Table 6. The computational cost of all models remains within 14$, indicating that LAG is also economically efficient. Overall, the LAG framework consumes more input tokens, while the output tokens represent only a small proportion. OpenAI prices output tokens significantly higher

<instruction>
You are an expert in summarizing and extracting key content from LaTeX-formatted academic papers on computers and artificial intelligence. Please output your reply in the following JSON format:
<format>[EXAMPLE JSON]</format>
==========
There are some key points to note:

- In the "selected table's core results", other models' results are of no concern and should be omitted.
- The table's header metrics should be the same as the evaluation metrics chosen.
- The number of items in the "classification of tables" dict should be equal to the "number of tables" int value. These two items help you to identify the main result tables better.
- All three items about the settings in the JSON output should be corresponding to the proposed method's best performance in the selected table.
- Sometimes in the selected table, the proposed method's performance may not be unique (e.g., different hyperparameters or training strategies), you need to choose the best one and it usually appears in the last row of the table.
- If there are multiple tables that meet the requirements (both being the main result table and based on the specified dataset), choose the one with richer information.

==========
Here, I provide you with an example of the complete process to help you understand your task.
First, I provide you an article:
<article>[EXAMPLE ARTICLE]</article>
Afterwards, I specify the dataset as **[EXAMPLE DATASET]**, you should output:
<format>[EXAMPLE RESPONSE]</format>
</instruction>

Box 1: The prompt of the table extraction agent, with the table classification COT procedure.

than input tokens, often reaching 4-5 times the cost of input tokens. However, considering the disparity in token numbers, the overall cost remains accaptable.

<instruction>
You are an expert in summarizing and extracting key content from LaTeX-formatted academic papers on computers and artificial intelligence. Please output your reply in the following JSON format:
<format>[EXAMPLE JSON]</format>
==========
There are some key points to note:

- In the "main result table's core results", other models' results are of no concern and should be omitted.
- The main result table's header metrics should be the same as the evaluation metrics chosen.
- All three items about the settings in the JSON output should correspond to the proposed method's best performance in the selected table.
- Sometimes in the main result table, the proposed method's performance may not be unique (e.g., different hyperparameters or training strategies), you need to choose the best one and it usually appears in the last row of the table.
- If there are multiple tables that meet the requirements (both being the main result table and based on the specified dataset), choose the one with richer information.

==========
Here, I provide you with an example of the complete process to help you understand your task.
First, I provide you an article:
<article>[EXAMPLE ARTICLE]</article>
Afterwards, I specify the dataset as **[EXAMPLE DATASET]**, you should output:
<format>[EXAMPLE RESPONSE]</format>
</instruction>

Box 2: The prompt of the table extraction agent, w/o the table classification COT procedure.

Input tokens	Output tokens	Qwen2.5-7/14B	kimiAI-128k	GPT4-o	O1-preview
834.7K	8.9K	0	50.616 ¥	2.176 $	13.055 $

Table 6: Cost of LAG

<instruction>
You are an expert in constructing the Artificial Intelligence leaderboard. Please refer to the content I provide you to answer the user's questions. The contents I provide you are a number of structured summaries extracted from computer/artificial intelligence papers.
You need to build a markdown format leaderboard (showcase the performance of the models on the same dataset, each line representing a specific model) based on the titles, experimental settings, and evaluation metrics of these articles. Please output your reply in the Markdown format.
Here, I list a complete example of the question and the answer to help you understand your task. For example, I provide you a list of JSON files containing the extracted content of the articles: **[JSON LIST]**
The expected leaderboard that you generate should be: **[EXAMPLE LEADERBOARD]**
==========
Pay attention: The leaderboard should be in the Markdown format and reflect all the articles provided! The leaderboard in the dictionary format is forbidden!
In the above case, selecting Pre and Rec as the metrics in the final leaderboard is not appropriate because in most articles the corresponding performance values are absent.
==========
Here, the target list of extracted content of the articles is as follows: **[TARGET JSON LIST]**
Please give me the well-organized leaderboard of the provided articles. The leaderboard should have consistent performance metrics, github links, settings and the title of the article, etc. The leaderboard should be in the Markdown format and reflect all the articles provided.
Warning:

- I need a well-organized markdown-format leaderboard containing all the articles' information. The leaderboard's max serial number in the "No." column should equal to the number of articles provided.
- When selecting metrics, you need to consider their text descriptions. The same metric may have multiple different abbreviations. In the final table, there must not be any duplicate metrics (it is unacceptable to have duplicates where different abbreviations represent the same meaning).
- Large-scale omissions are not allowed! For each model, only a small portion of the results are missing under the selected metrics. The vast majority of the metrics have corresponding values. The abbreviations for the same metric may be different, but you need to avoid being misled by the abbreviations.
- Use approximate intersections to select metrics from the given articles, while avoiding a large amount of data waste. Allow some models to have a certain degree of data missing under the selected metrics.
- The content in the "Experimental Setting" column should be concise and non-descriptive, just a few words.
- When different articles use different units for the same metric, please note that you need to convert them when integrating them so that the units in the final leaderboard are consistent. For example, 50% is equal to 0.5. "50" and "0.5" should not be presented in the same column of a leaderboard.
- Check each column corresponding to the selected metrics in the final leaderboard. If more than 60% of the values in that column are missing or represented by placeholders, the metric should be discarded.

</instruction>

Box 3: The prompt of the leaderboard construction agent.