LITMUS++ : An Agentic System for Predictive Analysis of Low-Resource Languages Across Tasks and Models

Avni Mittal[1] **Shanu Kumar**[2] **Sandipan Dandapat**[3] **Monojit Choudhury**[2]
[1]Microsoft Corporation, India
[2]Mohamed bin Zayed University of Artificial Intelligence (MBZUAI)
[3]Indian Institute of Technology Hyderabad
avnimittal@microsoft.com {shanu.kumar, monojit.choudhury}@mbzuai.ac.ae sdandapat@cse.iith.ac.in

Abstract

We present LITMUS++, an agentic system for *predicting* language-model performance for queries of the form "How will a *Model* perform on a *Task* in a *Language*?", a persistent challenge in multilingual and low-resource settings, settings where benchmarks are incomplete or unavailable. Unlike static evaluation suites or opaque LLM-as-judge pipelines, LITMUS++ implements an agentic, auditable workflow: a Directed Acyclic Graph of specialized *Thought Agents* that generate hypotheses, retrieve multilingual evidence, select predictive features, and train lightweight regressors with calibrated uncertainty. The system supports interactive querying through a chat-style interface, enabling users to inspect reasoning traces and cited evidence. Experiments across six tasks and five multilingual scenarios show that LITMUS++ delivers accurate and interpretable performance predictions, including in low-resource and unseen conditions. Code is available at https://github.com/AvniMittal13/litmus_plus_plus.

1 Introduction

Large Language Models (LLMs) now support diverse tasks such as reasoning, summarization, code synthesis, and multilingual communication across more than a hundred languages (OpenAI, 2023; Huang et al., 2024). Yet, evaluating their performance remains a critical bottleneck. Benchmark-driven resources such as XTREME-R and XGLUE (Ruder et al., 2021; Liang et al., 2020), along with broader stress tests like BIG-Bench and HELM (Srivastava et al., 2023; Liang et al., 2023), provide systematic measurement but cannot scale to the vast *Task–Model–Language* space, especially in low-resource settings. LLM-as-judge approaches (Zhou et al., 2024; Tan et al., 2024) offer scalability but raise concerns about bias, opacity, and reproducibility.

Predictive multilingual analysis has gained traction as an alternative to direct evaluation. Early methods like LangRank leveraged typological and corpus features for transfer prediction (Lin et al., 2019), while lightweight proxies such as LEEP and LogME offered fast transferability estimates (Nguyen et al., 2020; You et al., 2021). The foundational **LITMUS** predictor (Srinivasan et al., 2022) combined diverse linguistic and task features but required expert feature design and manual setup. More recent approaches, including Bayesian factorization and information-parity models (Schram et al., 2023; Tsvetkov and Kipnis, 2024), and multi-task zero-shot prediction frameworks (Ahuja et al., 2022b), improved scalability but still rely on predefined features and static configurations. Overall, existing predictors struggle to generalize under data scarcity and lack automation.

We introduce **LITMUS++**, an agentic system that transforms predictive evaluation into a fully autonomous workflow. A Directed Acyclic Graph (DAG) of specialized Thought Agents (Zhang et al., 2024) hypothesizes, gather multilingual evidence, select predictive features, and train lightweight regressors with calibrated uncertainty. This design enables interpretable and auditable predictions for unseen *Task–Model–Language* combinations, including challenging low- and zero-resource cases. The system is accessible through a browser-based interface (Figure 1), which combines three complementary views: a chat entry point for user queries, a live reasoning trace of DAG orchestration, and an evidence panel showing citations and exportable reports. Users can pose questions such as "How will a *Model* perform on a *Task* in a *Language*?", observe autonomous reasoning unfold in real time, and inspect the provenance of predictions.

We evaluate LITMUS++ across six representative tasks in five multilingual settings, measuring predictive accuracy, Q&A correctness, and reasoning quality dimensions such as plausibility, co-

Proceedings of The 14th International Joint Conference on Natural Language Processing and The 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: System Demonstrations, pages 47–54

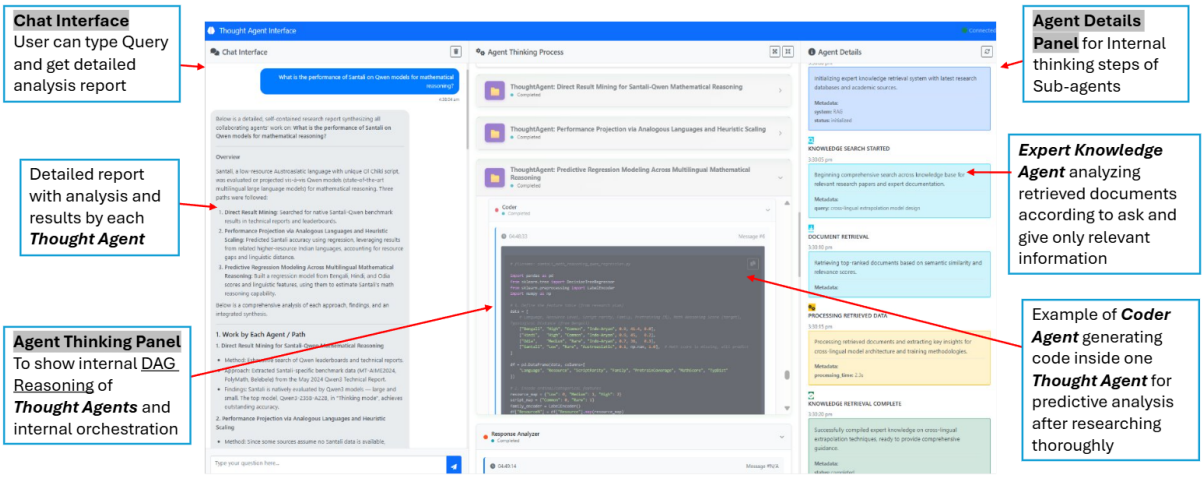

Figure 1: The interactive interface of LITMUS++. The system is organized into three coordinated panels: (left) the chat interface where users submit queries and receive structured analysis reports, (center) the DAG reasoning view showing the orchestration of *Thought Agents* and their outputs, and (right) the agent details panel exposing internal reasoning steps and expert knowledge retrieval. This design emphasizes transparency, allowing users to follow how predictions are generated and to inspect the provenance of evidence used in multilingual evaluation.

herence, and hallucination control. Our results show that DAG-based orchestration consistently reduces errors and enhances reasoning quality compared to single-agent and generalist multi-agent baselines. We have hosted a live demo at https://litmusplusplus.azurewebsites.net/.

2 System Overview

LITMUS++ is a multi-agent orchestration framework for automated, interpretable, and extensible evaluation of language models in multilingual and low-resource settings. The system transforms what is traditionally a manual research process into a fully autonomous workflow. At its core, a DAG architecture of collaborative *ThoughtAgents* enables structured decomposition of queries, parallel investigation of hypotheses, and traceable reasoning paths. By allowing multiple branches to expand or be pruned dynamically, the DAG ensures both efficiency and transparency. This design provides scalability across languages and tasks while preserving auditability.

2.1 End-to-End Workflow

Query Ingestion and Initialization: A natural language query (e.g., *"How will model X perform on task Y in language Z?"*) is first received by the *MainAgent*, which distinguishes between new and follow-up queries. For new queries, the *ThoughtCreatorAgent* generates hypotheses and spawns corresponding *ThoughtAgents* as nodes in the DAG. For the follow-up queries, the *ThoughtAnalyzerAgent* routes new information, spawns additional nodes, or prunes irrelevant ones to refine the DAG.

DAG-Based Reasoning: Each node in the DAG corresponds to a *ThoughtAgent*, which validates a single hypothesis. Dependencies across nodes allow for both parallel and conditional reasoning. The DAG evolves dynamically, expanding when new evidence emerges and pruning branches that are unproductive. Figure 3 illustrates the internal pipeline of a single *ThoughtAgent*, which itself orchestrates multiple specialized sub-agents.

Agent Internal Structure: A ThoughtAgent is in itself a groupchat of multiple collaborative sub-agents. The *Research Planner* coordinates investigations and decides what should be done next based on the current evidence provided by other agents such as *Web Search and Crawl*, *Expert Knowledge* and *Coder* agents. Creating these as separate agents, rather than just providing tools for web search, coding, and querying the curated multilingual knowledge base, leads to improved context management and allows individual iterative reasoning of sub agents with only relevant context for the groupchat. The *Send User Message* observes the conversation and produces a detailed report of the conversation at the end when either the hypothesis testing is complete successfully or some clarification is needed from the User. Together, they maintain a shared history of tool calls, reasoning steps, and outputs, ensuring reproducibil-

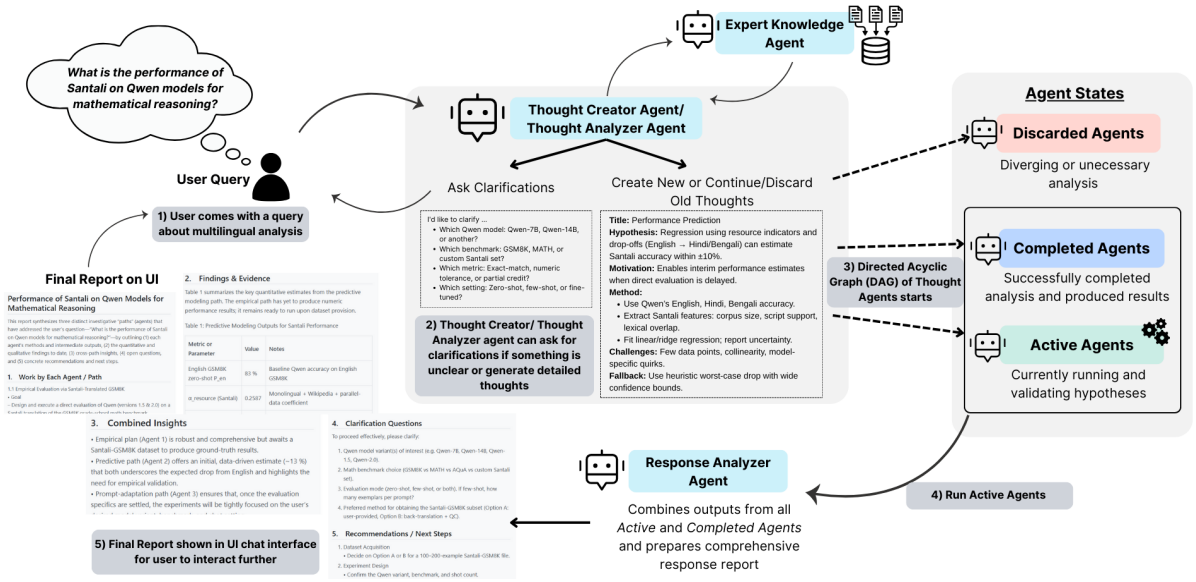

Figure 2: LITMUS++ pipeline: a user query triggers orchestration that initializes and refines a DAG of *ThoughtAgents*. Each agent retrieves evidence from the curated knowledge base, web search, and benchmarks, and may perform predictive modelling with lightweight regressors. The *Response Analyzer* aggregates results with uncertainty estimates and delivers both predictions and full reasoning traces to the user interface.

ity and transparency. More details on the design and curation of the knowledge base are provided in Appendix C.

Iterative Management: The *ThoughtAnalyzerAgent* monitors active hypotheses and manages progression. It routes clarifications, creates new agents, marks completed ones, and discards irrelevant branches. This iterative refinement keeps the system aligned with evolving queries while maintaining focus on the evaluation goal. The full lifecycle of *ThoughtAgents* is detailed in Appendix A.

Execution and Aggregation: Active *ThoughtAgents* run in parallel, producing validated hypotheses, predictive outputs, and evidence traces. Results are aggregated by the *ResponseAnalyzerAgent*, which synthesizes a final response that includes predictions, supporting evidence, confidence measures, and tradeoffs.

2.2 User Interface

The browser-based interface is organized into three coordinated panels as shown in Figure 1. The *Chat Window* (left) displays user queries and final system responses. The *Agent Reasoning View* (middle) logs the main orchestration flow, with expandable views of ThoughtAgents. The *Sub-agent Panel* (right) exposes detailed conversations of *Web Search and Crawl* and *Expert Knowledge* Agents. This design promotes transparency, allowing researchers to inspect intermediate reasoning and intervene when needed. The interface outputs comprehensive reports that combine retrieved evidence, predictions, and uncertainty estimates. These reports can be exported for reproducibility and integration into research workflows.

Implementation: LITMUS++ runs locally or in-browser with minimal setup, requiring only an API key for external search. The agentic backend uses *Autogen*[1] for orchestration, *ChromaDB*[2] as the curated knowledge base, and *Firecrawl*[3] for web search and scraping. Further details are provided in Appendix B.

3 Evaluation Framework

We design an evaluation framework to probe both the predictive accuracy and the reasoning quality of LITMUS++ in realistic multilingual conditions. The framework combines representative tasks, controlled scenarios, constrained knowledge access, and multi-dimensional evaluation metrics, balancing correctness with interpretability.

3.1 Evaluation Tasks and Scenarios

The benchmark spans six tasks: code generation, mathematical reasoning, question answering, text classification, text summarization, and machine

[1] https://microsoft.github.io/autogen/stable//index.html
[2] https://github.com/chroma-core/chroma
[3] https://www.firecrawl.dev/

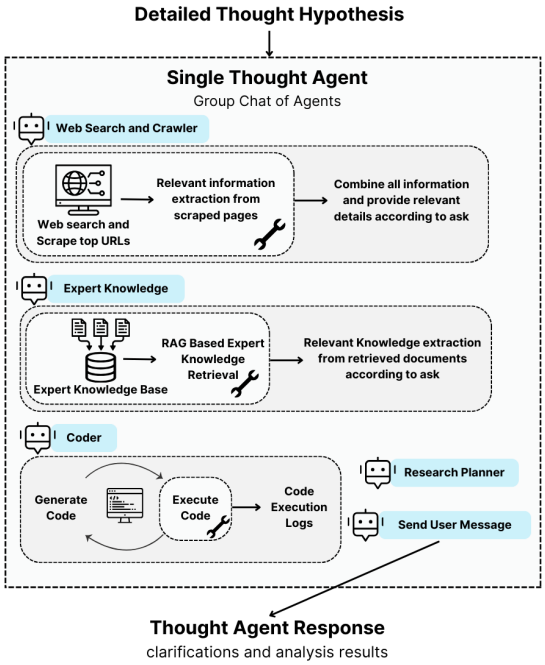

Figure 3: Single *ThoughtAgent* pipeline. Each agent operates as a group chat of specialized sub-agents (e.g., Web Search, Expert Knowledge, Coder) that validate a hypothesis and return structured results. These agents form the nodes of the DAG.

translation. To reflect practical multilingual challenges, each task is evaluated under five controlled scenarios: *Scenario 1 (Same Lang + Same Model)*: the same language and model are available. *Scenario 2 (Same Lang + Diff Model)*: the language is available but only with a different model. *Scenario 3 (Similar Lang + Same Model)*: transfer from a typologically related language using the same model. *Scenario 4 (Distant Lang + Same Model)*: transfer from a distant language with the same model. *Scenario 5 (No Lang + No Model)*: neither the language nor the model is represented.

Each scenario contains 60 questions (10 per task), covering high- to zero-resource conditions and testing how well the system adapts as prior evidence decreases. Scenario labels are used consistently in the Results.

3.2 Evaluation Sets and Knowledge Access Constraints

We evaluate the system using two complementary query sets, each consisting of 150 questions. The *PredSet* contains predictive analysis questions of the form "How will a model perform on a task in a language?", probing the system's ability to generate quantitative performance estimates. The *QnASet* contains comparative and factoid-style questions about models, languages, and benchmarks. Figure 4 shows representative examples.

While the system supports unrestricted web access, we constrain evidence sources during evaluation. The web-search tool is redirected to a fixed corpus of research papers, retrieving the top candidate via a retrieval-augmented setup that ranks paper abstracts by embedding similarity to the generated query. This ensures consistent, controlled conditions across scenarios while providing a realistic retrieval signal for multilingual evaluation.

Q1: *How does cross-lingual summarization work for low-resource Ukrainian?*
Answer: 13.5

Q2: *What is the performance of GPT-4o on MT for Amharic?*
Answer: 14

Q3: *Which models have been benchmarked on code generation in Sanskrit?*
Answer: Gemini 1.5, Gemini 2.0, LLaMA 7B

Q4: *Which model performs best for Math Reasoning in Italian?*
Answer: LLaMA 3.1 70B

Q5: *Compare Aquila-VL2 and Aria-MoE for QA/VQA in German.*
Answer: Aria-MoE

Figure 4: Illustrative queries from *PredSet* (predictive analysis) and *QnASet* (Q&A).

3.3 Evaluation Metrics

Outputs are judged using the LLM-as-Judge paradigm (Şahinuç et al., 2025; Li et al., 2024), complemented with task-specific correctness. For the *PredSet*, we measure *mean absolute error (MAE)* between predicted and ground truth performance values. For the *QnASet*, we report *accuracy* based on exact or task-appropriate matching.

Ground-truth values come from the fixed corpus of research papers. Scenario conditions are simulated by removing papers containing the target language–task–model results, the values in these removed papers serve as ground truth. In *Scenario 1*, the relevant paper remains in the corpus, making the task a retrieval case. In all other scenarios, ground-truth papers are excluded, creating controlled evidence scarcity for prediction. We plan to release additional dataset details in future work.

Beyond correctness, we evaluate multiple aspects of reasoning quality, including predictive plausibility under low-resource settings, citation verification (whether cited works exist and are relevant), citation emphasis (how strongly reasoning is

grounded in citations), the depth of feature selection and modeling choices, and overall coherence in logical flow and linguistic fluency. We further conduct human validation of the LLM judge's outputs, with details provided in Appendix D.

4 Results

We present quantitative results of LITMUS++, evaluating its predictive accuracy and Q&A performance under the five controlled scenarios introduced in Section 3. All experiments use GPT-4.1[4] as the underlying LLM. Detailed task-level numbers are provided in the supplementary material due to space constraints.

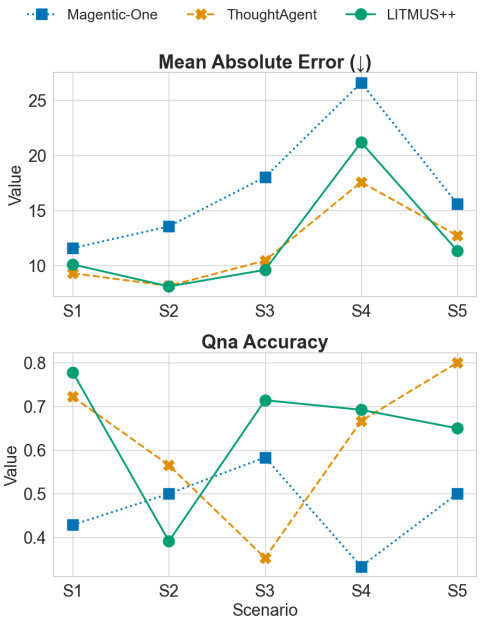

Figure 5: Quantitative results for *PredSet* (Mean Absolute Error, top) and *QnASet* (Accuracy, bottom) across the five scenarios (S1–S5). Lower is better for *PredSet*, higher is better for *QnASet*.

We compare against two baselines. The first, *ThoughtAgent*, is a simplified variant that processes the full query with a single agent, isolating the benefits of DAG-based coordination. The second, *Magentic-One*, is a generalist multi-agent framework from Microsoft,[5] included as a strong open-ended, general-purpose baseline.

4.1 Predictive and Q&A Performance

Figure 5 shows results on the two evaluation sets. On the *PredSet* (left), we report mean absolute error (↓). We observe that error generally increases from Scenario 1 to Scenario 4 as the prediction task becomes harder: in S1 and S2, the target language and model are available, so predictions are relatively straightforward; in S3 and especially S4, the system must rely on increasingly distant evidence and construct more complex transfer paths, which increases error. Interestingly, S5 shows a drop compared to S4: since neither language nor model is available, most systems fall back to predicting consistently low performance, which reduces variance and makes the case less challenging than S4 where nuanced feature-based modeling is required.

Across systems, *Magentic-One* shows the highest errors, especially in mid- and low-resource conditions, reflecting its lack of task-specific orchestration. Both LITMUS++ and *ThoughtAgent* maintain mean absolute error below or close to 12 in all scenarios except S4, highlighting the effectiveness of specialized reasoning even in harder conditions. Between the two, LITMUS++ achieves lower errors on average, showing the benefit of orchestrated DAG reasoning over a single-agent baseline.

On the *QnASet* (bottom in Figure 5), there is no uniform trend across scenarios: accuracy fluctuates depending on the combination of task and resource availability. Overall, LITMUS++ achieves the best performance in most scenarios, while *Magentic-One* consistently underperforms. *ThoughtAgent* remains competitive, often close to LITMUS++, but falls behind in scenarios requiring more complex reasoning (e.g., S3 and S4). These results confirm that orchestration in LITMUS++ provides a measurable advantage, though the single *ThoughtAgent* performs strongly, likely because the queries are relatively simple, reducing the need for multi-step reasoning and causing LITMUS++ to occasionally overdo the reasoning.

4.2 Reasoning Quality

We evaluate reasoning quality across five dimensions: predictive plausibility, feature selection, coherence, citation emphasis, and hallucination rate (Figure 6). Starting with *Predictive Plausibility*, LITMUS++ achieves the strongest and most stable scores (∼4.0–4.5), consistently producing reasonable and interpretable predictions even in challenging scenarios. *ThoughtAgent* remains competitive but slightly weaker (∼3.4–3.6), while *Magentic-One* trails at ∼3.0 across all scenarios, highlighting the value of structured orchestration.

On *Feature Selection*, the advantages of or-

[4] https://openai.com/index/gpt-4-1/

[5] https://microsoft.github.io/autogen/stable//user-guide/agentchat-user-guide/magentic-one.html

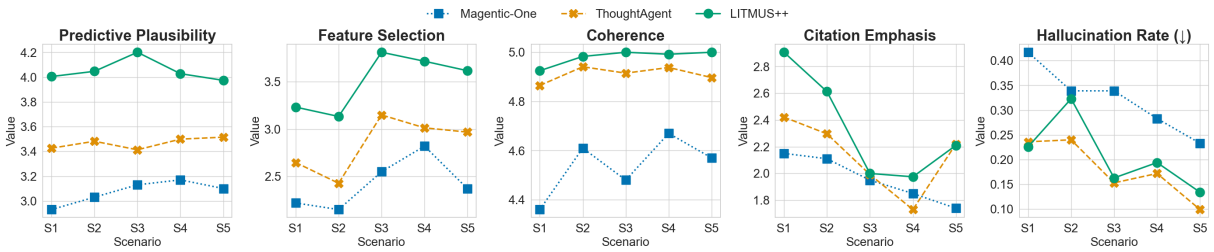

Figure 6: Qualitative results across non-accuracy metrics. Each subplot reports performance averaged across all tasks and scenarios, enabling comparison of model behavior under multiple evaluation dimensions.

chestration become even clearer. LITMUS++ reaches as high as ∼3.8 in S3–S4, precisely the settings where feature-driven reasoning is essential. *ThoughtAgent* improves under the same conditions but stays ∼0.5 points behind, while *Magentic-One* struggles at ∼2.2–2.8. For *Coherence*, both LITMUS++ and *ThoughtAgent* maintain excellent fluency and logical consistency (∼4.9–5.0), with LITMUS++ slightly ahead. *Magentic-One*, however, lags behind with lower scores of ∼4.3–4.6, underscoring its weaker reasoning discipline.

For *Citation Emphasis*, LITMUS++ grounds its outputs more consistently in cited evidence, scoring ∼2.5–3.0 in S1–S2. Although this decreases in S3–S4, it remains above both baselines. *ThoughtAgent* follows the same trend at lower levels, while *Magentic-One* is lowest throughout. *Hallucination Rate* shows an unexpected pattern: instead of rising as evidence grows scarcer, hallucinations actually decrease from S1 to S5. In high-resource cases like S1, models sometimes hallucinate nonexistent papers due to strong priors, whereas in low-resource cases they adhere strictly to constrained citations. *Magentic-One* hallucinates the most (42%→24%), while LITMUS++ and *ThoughtAgent* remain substantially lower (23%→10%). Overall, while *ThoughtAgent* stays close to LITMUS++ in surface-level coherence, it lags behind on citation grounding, feature-driven reasoning, and hallucination control, whereas *Magentic-One* underperforms across all dimensions.

4.3 Ablation Study

To examine the impact of the underlying LLM in LITMUS++, we ran an ablation study in the Web Search configuration, testing o3-mini,[6] a model reported to have stronger reasoning than GPT-4.1 on 30 questions. Table 1 reports results across plausibility, feature selection, citation emphasis, and coherence. Despite o3-mini's reasoning-oriented design, results were highly competitive: GPT-4.1 achieved stronger citation grounding, while o3-mini offered slight gains in feature selection. Overall, LITMUS++ remains robust across backbones, indicating that orchestration matters more than the choice of a single LLM. Extending this to open-source LLMs is left for future work.

Model	Predictive Plausibility	Feature Selection	Citation Emphasis	Coherence
o3-mini	3.97	3.68	1.19	5.00
GPT-4.1	3.90	3.10	2.10	4.97

Table 1: Ablation study of LITMUS++ with different underlying LLMs in the Web Search configuration, evaluated on reasoning quality metrics.

5 Conclusion

We introduced LITMUS++, a demo system for multilingual performance prediction that combines DAG-based orchestration of thought agents with transparent reasoning and uncertainty-aware outputs. The system enables users to query tasks, inspect evidence traces, and obtain plausible predictions even under distant and zero-resource conditions. Compared to strong baselines such as *ThoughtAgent* and *Magentic-One*, LITMUS++ achieves lower prediction error, higher Q&A accuracy, and stronger reasoning quality, making it both effective and trustworthy. The demo illustrates how complex evaluation workflows can be transformed into interactive, auditable experiences, lowering the barrier for researchers and practitioners to explore multilingual model behavior. We have hosted a live demo for review while a broader public release is under active development. Future work will focus on optimizing latency, expanding task coverage, and extending the curated knowledge base to further strengthen the system's utility.

[6] https://openai.com/index/openai-o3-mini/

References

Kabir Ahuja, Sandipan Dandapat, Sunayana Sitaram, and Monojit Choudhury. 2022a. Beyond static models and test sets: Benchmarking the potential of pre-trained models across tasks and languages. *arXiv preprint arXiv:2205.06356*.

Kabir Ahuja, Shanu Kumar, Sandipan Dandapat, and Monojit Choudhury. 2022b. Multi task learning for zero shot performance prediction of multilingual models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5454–5467, Dublin, Ireland. Association for Computational Linguistics.

Błażej Dolicki and Gerasimos Spanakis. 2021. Analysing the impact of linguistic features on cross-lingual transfer. *arXiv preprint arXiv:2105.05975*.

Kaiyu Huang, Fengran Mo, Xinyu Zhang, Hongliang Li, You Li, Yuanchi Zhang, Weijian Yi, Yulong Mao, Jinchen Liu, Yuzhuang Xu, and 1 others. 2024. A survey on large language models with multilingualism: Recent advances and new frontiers. *arXiv preprint arXiv:2405.10936*.

Shanu Kumar, Soujanya Abbaraju, Sandipan Dandapat, Sunayana Sitaram, and Monojit Choudhury. 2023. Ditto: A feature representation imitation approach for improving cross-lingual transfer. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 385–406.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024. Llms-as-judges: a comprehensive survey on llm-based evaluation methods. *arXiv preprint arXiv:2412.05579*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Featured Certification, Expert Certification, Outstanding Certification.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, and 5 others. 2020. XGLUE: A new benchmark dataset for cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. 2020. Leep: A new measure to evaluate transferability of learned representations. In *International Conference on Machine Learning*, pages 7294–7305. PMLR.

OpenAI. 2023. Gpt-4 technical report. `https://cdn.openai.com/papers/gpt-4.pdf`.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. 2021. XTREME-R: Towards more challenging and nuanced multilingual evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10215–10245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Furkan Şahinuç, Subhabrata Dutta, and Iryna Gurevych. 2025. Expert preference-based evaluation of automated related work generation. *arXiv preprint arXiv:2508.07955*.

Viktoria Schram, Daniel Beck, and Trevor Cohn. 2023. Performance prediction via bayesian matrix factorisation for multilingual natural language processing tasks. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1790–1801.

Anirudh Srinivasan, Gauri Kholkar, Rahul Kejriwal, Tanuja Ganu, Sandipan Dandapat, Sunayana Sitaram, Balakrishnan Santhanam, Somak Aditya, Kalika Bali, and Monojit Choudhury. 2022. Litmus predictor: An ai assistant for building reliable, high-performing and fair multilingual nlp systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 13227–13229.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, and 431 others. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*. Featured Certification.

S. Tan and 1 others. 2024. Judgebench: A benchmark for evaluating llm-based judges. https://openreview.net/forum?id=G0dksFayVq.

Alexander Tsvetkov and Alon Kipnis. 2024. Information parity: Measuring and predicting the multilingual capabilities of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7971–7989.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. 2021. Logme: Practical assessment of pre-trained models for transfer learning. In *International Conference on Machine Learning*, pages 12133–12143. PMLR.

Yifan Zhang, Yang Yuan, and Andrew Chi-Chih Yao. 2024. On the diagram of thought. *arXiv preprint arXiv:2409.10038*.

R. Zhou and 1 others. 2024. Is llm a reliable reviewer? a comprehensive evaluation of llm on automatic paper reviewing tasks. In *LREC-COLING 2024 (Proceedings)*.

Ethical Considerations

LITMUS++ operates in sensitive settings such as multilingual fairness. Although evaluated under controlled evidence access, it provides predictive estimates and is not a replacement for ground-truth benchmarks. We mitigate risks through curated knowledge bases and transparent reasoning traces and will extend coverage responsibly in future.

A Agent Lifecycle Details

Each *ThoughtAgent* transitions between three core states: **Active**: investigating a hypothesis with assigned tools; **Completed**: finished investigation and returned validated evidence; **Discarded**: pruned when deemed irrelevant, redundant, or divergent. State transitions are managed by the *ThoughtAnalyzerAgent*, which monitors progress and determines whether to continue, complete, or discard a *ThoughtAgent*. This lifecycle ensures only relevant outputs contribute to the final analysis, while providing auditable reasoning paths.

B Implementation Details

Tooling. Agents access a modular suite of reusable tools, including web search and scraping utilities, knowledge-base retrieval functions, and code executor. Tools are independently registered and can be added, replaced, or modified without altering agent logic, enabling easy integration of new APIs or analysis modules. **Deployment.** The system supports both local and hosted execution, running via a command-line interface or local server, with a hosted variant for reproducible experiments. External search and LLM calls require user-provided API keys; all other components run offline. **Performance.** Predictions include evidence traces and calibrated uncertainty estimates, offering transparent and confidence-aware reasoning. **Extensibility.** The DAG orchestration is model- and task-agnostic. New agents or tools are added by registering them within the *MainAgent* or *ThoughtAgent* logic, without modifying control flow, supporting ongoing extensibility as evaluation needs evolve.

C Knowledge Base Curation

A curated multilingual knowledge base grounds LITMUS++ in linguistic and computational evidence. It integrates (i) **literature-derived resources** from peer-reviewed papers, benchmarks, and typological databases (Lauscher et al., 2020; Dolicki and Spanakis, 2021; Srinivasan et al., 2022; Ahuja et al., 2022a; Kumar et al., 2023), and (ii) **expert annotations** for under-documented or low-resource languages. It is organized as detailed reports over language–task–model combinations, combining few-shot examples, known failure modes, and best-practice guidelines. We employ a retrieval-augmented generation setup, where top-K chunks from this knowledge base are passed as tool outputs to the Expert Knowledge Agent, which synthesizes answers for the current query. The knowledge base can be expanded as new research, experimental findings, and expert inputs become available, supporting hypothesis generation, feature selection, and provenance tracking.

D Human Validation of LLM-as-Judge Evaluation

To assess the reliability of LLM-as-Judge, we ran a human validation study on a subset of reports. Annotators received the report, LLM reasoning, rating criteria and scores (1–5) on four metrics: predictive plausibility, coherence, feature selection, and emphasis on citations. Their task was to mark agreement or disagreement (binary 1/0) with each score. The results showed that an annotator agreed with the LLM evaluations in 81.25% of the cases, while the second annotator agreed in 78.1% of the cases. The high agreement indicates that the LLM-as-Judge framework provides evaluations that are generally consistent with human judgment, though some divergences remain.