Developing a Local Backend Assistant using MCP Tools for Briefing Deep Learning Experiments

Yu Ando[1], Junghwan Cho[2]

[1]Department of Biomedical Science, Kyungpook National University, Republic of Korea
[2]Clinical Omics Institute, Kyungpook National University Daegu, Republic of Korea
[1]yuando@knu.ac.kr

1. INTRODUCTION

In the rapidly evolving field of artificial intelligence, the ability to efficiently and accurately evaluate the performance of large language models is paramount. This work presents an elementary analysis of four open weight models: gemma 3n, Qwen3, Deepseek R1, and Mistral. By leveraging a variety of tools and methodologies, we aim to assess the capabilities of these models in handling diverse tasks, ranging from querying datasets and models to conducting detailed experiment comparisons. The insights gained from this study will contribute to the development of more robust and user-friendly AI systems, ultimately enhancing the overall user experience.

2. MATERIALS AND METHODS

We gave requests to each model for resources available on the server including models, datasets, logged experiments and metrics. We tested four open weight models obtained from huggingface: gemma 3n (gemma_3_n, 2025), Qwen3 (Yang et al, 2025), Distilled DeepSeek R1 (Zhao et al, 2025), and Mistral (MistralAI, 2025). The following tools are available to the large language model: querying datasets and models available on the server, recent 10 experiments conducted on the server, full experiment logs queried by metric name (accuracy, loss, etc.), and pairwise comparison between experiments by their metric logs.

The prompts were designed with various levels of difficulty: naïve tool calling (query dataset/model recent experiments), context length availability (experiment logs having 50, 100, 200 epochs of data), and planned invocation (comparing multiple experiments and sequential calling of tools). We evaluated the response as successful if it is not trivially incorrect; a response containing false information, incomplete information, or failure to respond is counted as unsuccessful.

Test (Difficulty)	Prompt	Expected Results
Tool Call Test (Low)	Can you tell me what datasets and models are available in this system for deep learning experiments?	Returns data from server.
	What recent experiments have been scheduled?	Returns data from server.
Large Context Test (Medium)	Observe accuracy trends for the experiment with id X. What findings do you see?	Describes the trend with multiple data points.
Planned Invocation (High)	Compare experiments with ids X, Y, and Z. What findings do you see?	Compare experiments with pair-wise tool calls.
	I ran some experiments called `test-name`. Could you brief me on the results of the experiments by comparing them?	Planned tool calling by first retrieving experiment ids and later uses the comparison tools pairwise.

Table 1. Testing different prompts with varied levels of difficulty and the expected responses from the language model.

All experiments were conducted on two Intel ARC A770 graphical processing units. llama.cpp was employed to serve large language models and a basic FastMCP app was developed to access a log database holding typical deep learning experiment logs. Context length was set to 4096 when running the large language models. Open WebUI was employed as a client user interface to prompt large language models. Table 1 outlines the prompts for various levels of question difficulty.

3. RESULTS AND DISCUSSION

The experiments focused on evaluating large language models using the llama.cpp backend with a context length of 4096 tokens. Interactions were managed via the Open WebUI client, and logs were systematically captured using the FastMCP application. The study included a range of experimental prompts targeting different question difficulty levels. Several experiment runs were executed, and their outcomes are summarized in Table 2.

The results indicate that successful responses do not depend on the model architecture itself but in the implementation of the communication between the tool server and the large language model. For instance, querying data gives consistent successful response regardless of model.

Prompts for direct retrieval of experiment logs containing logged metric values per epoch were not successful possibly due to context length limitations. Only one model, gemma-3n, was able to respond with coherent information but failed for logs containing more than 100. In contrast, a pairwise comparison tool allowed comparing experiments via a summarizing function. Although the tool is implemented for pair comparisons, all models except Mistral were able to infer that three pairwise comparisons allow ranking of the compared experiments. Mistral, on the other hand, compared hyperparameter information obtained via querying recent experiments. Technically, this is not incorrect and thus shows that prompt engineering may be a vital step in implementing such systems.

The final prompt was designed to be a multi-step problem: the model must find corresponding ids for the logs and use them to compare using pairwise calls to the comparison tool. Unfortunately, all models failed this prompt. We suspect that there is a limitation in multi-step tool calling when the tool servers do not explicitly guide the model for this specific case or limited capability in exploring. An in-depth investigation is required for these multi-step problems in local large language models where information is not sufficient to chain tool calls step by step.

	Datasets and Models	Recent Experiments	Experiment Logs (N=50, 100, 200)	Comparing 3 Experiments	Finding and compare
gemma-3n-E4B-it-Q4_K_M	✓✓✓	✓✓✓	✓✗✗	✓✓✓	✗✗✗
Qwen3-14B-UD-Q8_K_XL	✓✓✓	✓✓✓	✗✗✗	✓✓✓	✗✗✗
Deepseek-R1-Distill-Qwen-14B-Q8_0	✓✓✓	✓✓✓	✗✗✗	✓✓✓	✗✗✗
Mistral-Small-3.2-24B-Instruct-2506-UD-Q4_K_XL	✓✓✓	✓✓✓	✗✗✗	✓✓✓	✗✗✗

Table 2. Query success results for each language model.

4. CONCLUSION

Aided with tools, chatbots using large language models have the capability to improve user experience in rudimentary tasks including experiment comparison, experiment log briefing, and basic tasks like informing server capability. However, we observed that local models have context length

limitation and difficulty in exploration to obtain more information if not explicitly specified by the tool server. Developing local agents, therefore, requires careful design in data transfers and guides between servers and agents to allow for consistent and correct responses which will help user experience.

5. DATA AVAILABILITY

We plan to release chat records in due time to a public repository but can be shared at reasonable request.

6. ACKNOWLEDGMENT

The author would like to express appreciation to Clinical Omics Institute, Kyungpook National University and the BK21FOUR KNU Convergence Biomedical Science Future Creative Talent Cultivation Education & Research Group. The author thanks Copilot for corrections and initial layout of this writing.

This work was supported by the Brain Pool Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science and ICT (Grant No. RS-2023-00283791) and the Ministry of Education, Korea (Grant No. 2021R1I1A3056903 & RS-2024-00459836).

Y. Ando did initial draft, experiment conceptualization, its execution, and prepared required software development and hardware. J. Cho was in part of organizing the challenge and symposium.

7. REFERENCES

1. Google Team, 2025, Gemma 3n, Google DeepMind, https://ai.google.dev/gemma/docs/gemma-3n
2. Yang, An, et al., 2025, Qwen3 Technical Report, arXiv preprint arXiv:2505.09388
3. Zhao, Han, et al. 2025, 1.4 million open-source distilled reasoning dataset to empower large language model training. arXiv preprint arXiv:2503.19633
4. MistralAI, 2025, Mistral-Small-3.2-24B-Instruct-2026, HuggingFace, https://huggingface.co/mistralai/Mistral-Small-3.2-24B-Instruct-2506