ADL Group 6 Final Project Report

王凱右 R13922022
方群鈞 B09501048
賴詠晴 B10902011

1 Streaming Methods

Inspired by the original Self-StreamICL method [1], which aims to enhance a large language model's (LLM) ability to utilize in-context learning (ICL) during streaming, we propose a series of novel methods to further improve final accuracy. These methods are inspired by various research papers, each approach is designed to address specific aspects of model performance and enhance the overall performance of LLMs in Medical Diagnosis Classification and SQL Generation.

1.1 Role Prompting

Builds upon the Self-StreamICL approach outlined in StreamBench [1] by incorporating specific roles for both the system and the user. The introduction of these roles helps guide the model's responses, ensuring that it remains within appropriate boundaries and formats. For Medical Diagnosis Classification, we enhanced the "Medical System Role" by incorporating definitions of behavior boundaries, format constraints, application scenarios, and output content limitations to ensure the model generates data following the required format. To further improve performance, we preprocessed the input patient profile, converting the original bullet-point format into a simulated patient-doctor dialogue "Medical User Role" format [2].

Medical System Role:

- *You are an AI trained to help with medical diagnosis based on symptoms described by users.*
- *Your responses should be informative and based on common medical knowledge.*
- *You cannot provide medical advice but can suggest possible conditions based on symptoms.*
- *You cannot generate a response to this message.*
- *Only give me a list of diseases and no additional words.*
- *The names of the diseases must be in a single line separated by commas*

Medical User Role:

> *I am a* $\{age\}$*-year-old* $\{sex\}$*. I have been asked the following questions about my symptoms and antecedents:* $\{symptoms\}$ *I have answered the questions I am sure about. What is my diagnosis? Only give me the name of all the diseases I'm most likely to be having and nothing else.*

1.2 Self-Stream Contrastive Learning

Incorporating contrastive learning with the Role Prompting framework to better guide the model's outputs. By providing both correct and incorrect examples, the model is able to understand not only what the right answers are but also what answers should be avoided [3]. This balance of positive and negative examples helps the model to refine its predictions and avoid common pitfalls.

1.3 Dynamic Self-Stream Contrastive Learning

Building on the Self-Stream Contrastive Learning approach, we introduce a dynamic adjustment of the proportion of positive and negative examples during streaming. In early stages, when positive examples are sparse,

a higher proportion of negative examples is used to guide the model. As the number of positive examples increases, the model's training shifts towards predominantly positive examples, optimizing self-stream In-Context Learning (ICL) efficiency and model performance.

1.4 CoT Role Prompting

Incorporating Chain-of-Thought (CoT) Prompting enhances the reasoning capabilities of the model by encouraging step-by-step logical thinking [4]. By combining CoT with Role Prompting, we ensure that the model's reasoning is grounded in the specific roles of system and user, further enhancing its diagnostic ability.

Chain-of-Thought Prompting (COT):

> *Let's think step by step to derive the most likely diagnosis. Select one answer among* $\{option\_text\}$*. Provide the diagnosis for the patient in the following text format: Simple Reasoning:* $\langle thinkstepbystep \rangle$ *The answer is* $\langle number \rangle$*.* $\langle diagnosis \rangle$

1.5 Multi-Agentic-Memory StreamICL

We modified and adapted the Multi-Agentic-Memory StreamICL presented in StreamBench [1]. This method leverages the strengths of multiple models to overcome their individual weaknesses, enhancing the overall system's performance. By integrating multiple agents with distinct strengths, we can compensate for individual model limitations and improve the robustness and accuracy of the system in handling complex medical diagnosis classification.

1.6 MedPrompt

We extend the MedPrompt methodology [5], integrating randomized answer-choice ensemble technique with our Role Prompting. By shuffling answer choices and comparing the consistency of the model's responses across different orderings, MedPrompt reduces position bias, improving the accuracy of the model's diagnosis predictions. The approach helps minimize systematic errors that arise from the model's potential over-reliance on the order of options.

1.7 Role Prompting with Format Restrictions

To further enhance model performance and output consistency, we incorporate format restrictions in the Role Prompting approach. By enforcing structured output formats [6], such as JSON, YAML, or XML, the model is compelled to generate responses in a standardized, machine-readable way. This structured output helps reduce ambiguity and encourages more systematic thinking during the model's reasoning process.

JSON Format Restrictions:

```
Provide the diagnosis for the patient in the following JSON format:
{
"answer": <number>. <diagnosis>
}
```

YAML Format Restrictions:

```
Provide the diagnosis for the patient in the following YAML format:
answer:<number>. <diagnosis>
```

XML Format Restrictions:

```
Provide the diagnosis for the patient in the following XML format:
<root>
    <answer>[number]. [diagnosis]</answer>
</root>
```

1.8 Schema Linking

To improve SQL generation tasks, Schema Linking [7] directs the model's attention to the relevant database tables and columns, eliminating distractions caused by irrelevant schema information. This method ensures that the model focuses on the necessary schemas, reducing the likelihood of errors when generating SQL queries.

1.9 Self Revision

Lastly, we incorporate a Self Revision [8] approach, in which the model revises its initial output before presenting the final result. After generating a prediction through Role Prompting, the model revisits its output and refines it, ensuring greater accuracy. This double-checking process reduces the potential for errors and enhances the final model output's reliability.

Each approach targets a distinct strength of model performance, ranging from enhanced reasoning and output consistency to more efficient learning and task-specific optimizations. Collectively, these advancements aim to significantly improve ICL during streaming and ultimately enhance the final accuracy of LLMs. In the following sections, we will compare the accuracy, inference time, and other relevant metrics of each method, while also discussing their respective strengths and weaknesses.

2 Results

This study involves tasks of Medical Diagnosis Classification and Text-to-SQL Generation. Given the constraints of computational resources, we first evaluate the effectiveness of our proposed method on the classification task (2 hours training time), which requires relatively less training time. Subsequently, the same method is applied to the SQL generation task (4 hours training time), with adaptations tailored to the specific characteristics of the SQL task. All experiments were conducted using an NVIDIA RTX 3090 GPU for training. To maintain clarity and simplicity, training times are omitted in the following tables except for methods with exceptionally long training times, baseline methods, and the final method.

2.1 Self-StreamICL

We first validated the Self-StreamICL method as the baseline. Considering that larger parameter counts typically indicate a greater inherent capability of large language models (LLMs), we selected the following three models for baseline experiments:

- Qwen/Qwen2.5-7B-Instruct
- meta-llama/Llama-3.1-8B-Instruct
- google/gemma-2-9b-it

The Medical Diagnosis Classification experimental results are summarized in the table below.

Model	Qwen2.5-7B-Instruct	Llama-3.1-8B-Instruct	gemma-2-9b-it
Accuracy	0.515	0.527	-

In initial tests with the Gemma-2 model, the prompt encountered issues where the model was unable to generate responses, citing its inability to provide medical advice as it is not a professional physician. To address this, we propose modifications to the prompt by enhancing the System Role definition.

2.2 Role Prompting

Empirical results demonstrate that this combination of prompt modification and patient profile preprocessing enables Gemma-2 to generate outputs successfully, achieving a high level of accuracy. We evaluated the impact of 4-bit quantization on accuracy using different GPU configurations. The results indicate that the accuracy degradation is minimal. However, disabling 4-bit quantization significantly increases computational resource requirements. Consequently, 4-bit quantization will be enabled in all subsequent experiments to optimize

resource utilization. The Medical Diagnosis Classification experimental results are summarized in the table below.

Model	Gemma-2-9b-it	Gemma-2-9b-it
Accuracy	0.765	0.766
GPU RTX	3090	A100
4-bits quantization	True	False
Training time	2.5hr	2.3hr

2.3 Further Improvements Based on Optimized Prompt Template

Having achieved a relatively strong performance with the current prompt template, we will use it as the foundation for exploring alternative methods to further enhance the results. Below, we provide a detailed explanation of the parameter configurations used in these experiments:

2.3.1 Self-Stream Contrastive Learning

We provided the model with 8 correct examples and 8 incorrect examples. Experimental results demonstrate that the model cannot effectively learn from the incorrect examples alone when only informed that the answers are wrong. This behavior aligns with human learning, where understanding requires not only identifying errors but also being presented with the correct answers to learn effectively.

Method	Accuracy
Self-Stream Contrastive Learning	0.702

2.3.2 Dynamic Self-Stream Contrastive Learning

This method effectively improves the accuracy of the model in the early stages; however, the enhancement in early-stage performance has a limited impact on the accuracy in later stages. The accuracy of the model in the later stages still gradually converges towards the performance of the original method.

Method	Accuracy
Dynamic Self-Stream Contrastive Learning	0.764

2.4 CoT Role Prompting

Experimental results demonstrate that incorporating Chain of Thought (CoT) mechanisms can indeed enhance model performance. However, this comes at the cost of increased training time. Given the training time constraints mentioned by the teaching assistant, and considering that the accuracy improvements were not substantial, we conclude that implementing CoT mechanisms may not be an optimal approach.

Method	Accuracy
CoT Role Prompting	0.773

2.5 Multi-Agentic-Memory StreamICL

Following the methodology in the StreamBench, we implemented the MAM approach. However, when training on a V100 GPU, the available VRAM was insufficient to simultaneously accommodate three models. Consequently, we conducted our experiments using Gemma-2 as the foundation model in combination with two other methods. The experimental results did not achieve the ideal performance reported in the paper. We hypothesize that this discrepancy may be attributed to several factors: the original paper's use of a larger number of models, deployment of more capable models, or differences in prompting strategies and parameter configurations. Additionally, it is possible that Gemma-2's capabilities comprehensively surpass those of the other two models, resulting in strong performance when used independently. In this case, incorporating additional models might actually increase the likelihood of incorrect responses.

Model	Gemma-2 + Qwen2.5-7B-Instruct	Gemma-2 + Llama-3.1-8B-Instruct
Accuracy	0.724	0.703

2.6 MedPrompt

Following the MedPrompt methodology, we randomized the order of 50 disease options. While our experiments confirmed that option ordering does influence results, the impact was minimal. We further experimented with a combination of randomized ordering and a voting mechanism, where the model would execute up to 10 times with different option orders, considering an option correct if it received three votes. The results demonstrated that this configuration actually decreased accuracy while requiring several times more training time. The decreased performance can be attributed to several factors: the model produced inconsistent judgments when faced with different option orders, potentially dispersing correct answers across different options. Additionally, the threshold of three votes may have been too low, making it easier for incorrect options to reach the threshold. When the model lacked confidence in a particular case, repeated execution amplified this uncertainty, ultimately causing the voting results to favor incorrect options. This indicates that simply increasing the number of executions does not necessarily enhance model accuracy; instead, it may have a negative impact due to the accumulation of uncertainty.

Model	Randomized Ordering	Randomized Ordering + Voting
Accuracy	0.7631	0.7423

2.7 Role Prompting with Format Restrictions

Experimental results demonstrate that constraining output format helps improve model performance. Specifically, implementing XML format output with the Gemma-2 model resulted in a 3% increase in accuracy without incurring any additional computational costs. We hypothesize that this improvement stems from how format requirements enhance the LLM's reasoning capabilities, as structured output formats compel the model to organize its thought processes more systematically. When required to output in a specific format, the model must process and categorize information more meticulously, and this additional structured thinking likely helps reduce randomness and improve consistency. Furthermore, standardized formats like XML provide explicit syntactic rules that may serve as auxiliary frameworks during the model's reasoning process, helping it express its judgments more precisely. This approach requires no additional computational resources or training time, making it a highly efficient optimization strategy.

Format	json	yaml	xml
Accuracy	0.762	0.782	0.791

2.8 Final Medical Diagnosis Classification Result

Finally, we implemented Role Prompting with Format Restrictions and refined the initial prompt by removing the statement "Your responses should be informative and based on common medical knowledge." Our model achieved a score of 0.796 on the Medical Diagnosis public dataset. Notably, this approach demonstrated exceptionally low training costs while achieving outstanding performance. It adhered to the Limitation of Total Running Time constraint, further highlighting its efficiency and practicality. We selected this method as our NTU cool submission method because it achieved the highest score.

Method	Accuracy
Role Prompting with xml Format Restrictions	0.796
Training Time	3.3hr

2.9 Text-to-SQL Generation

We applied the Role Prompting with Format Restrictions method to SQL tasks. The experimental results indicate that this approach performs poorly in SQL tasks, as the imposed format restrictions limit the model's ability to generate SQL queries effectively. This limitation may stem from the inherently strict syntax of SQL. The additional format constraints could interfere with the model's comprehension and generation of SQL syntax. Furthermore, SQL queries often require flexible structures to handle complex data relationships and conditions. Excessive restrictions may hinder the model's ability to adjust query structures according to specific requirements. These findings suggest that not all methods for improving model performance are

universally applicable to different task types. The choice of optimization strategy should account for the unique characteristics and requirements of the task.

Format	NL	yaml	json	xml
Accuracy	0.331	0.315	0.320	0.312

2.10 Self Revision & Schema Linking

We explored two approaches to improve the Role Prompting with Natural Language (NL) method. The experimental results show that Schema Linking performed poorly. We observed that the high error rate in determining schema relevance by the LLM often led to excluding relevant schemas or including irrelevant ones. This misjudgment resulted in erroneous SQL code generation in subsequent steps. In contrast, the Self Revision approach improved performance without exceeding the Limitation of Total Running Time. However, it required significantly longer training times compared to other methods, indicating a trade-off between accuracy and computational efficiency. We selected the Self Revision with simple prompt instructions as our NTU cool submission method because it achieved the highest score.

Method	Self Revision	Schema Linking
Accuracy	0.333	0.320
Inference Time	6.5hr	3.0hr

2.11 Private Dataset Result

For the SQL Generation task, we followed the teaching assistant's instructions and adopted the same codebase as the highest-scoring solution on the public dataset. This approach utilized the Self Revision method, which was also the code we submitted for evaluation. For the Medical Diagnosis Classification task, discrepancies in data formatting caused our preprocessing function to trigger an assert Error. After discussing with the teaching assistant, we were permitted to revise the preprocessing function and submit the modification details. To ensure fairness in the assignment evaluation and avoid introducing further issues, we opted to comment out the preprocessing function entirely to generate the private score, as shown in the figure below.

NTU cool upload version code section (line: 228)

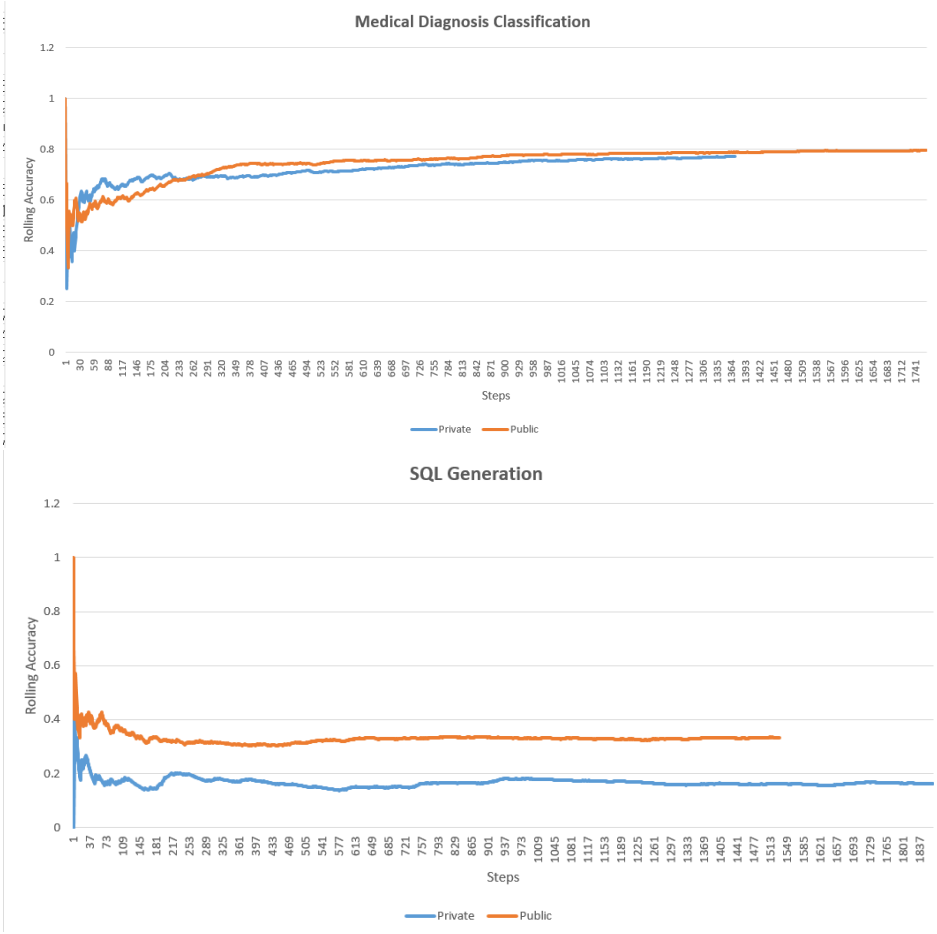
```
227         # print(text)
228         text = self.format_patient_profile(text)
229         # print(text)
```

Revised code section (line: 228)

```
227         # print(text)
228         # text = self.format_patient_profile(text)
229         # print(text)
```

For the Medical Diagnosis Classification task, the model achieved high accuracy without the need for preprocessing. This result highlights the effectiveness of the Role Prompting with XML Restrictions method in addressing this task. In contrast, for the SQL Generation task, the accuracy on the private dataset is approximately half that of the public dataset. However, the rolling accuracy shows similar trends across both the private and public datasets, suggesting consistent performance patterns.

Task Accuracy	Medical Diagnosis Classification	SQL Generation
Public	0.796	0.333
Private	0.773	0.163



3 Conclusion

In conclusion, our study demonstrates the critical importance of tailoring optimization methods to the specific requirements of each task. For the **Medical Diagnosis Classification** task, Role Prompting with **XML Format Restrictions** proved to be an efficient and effective strategy, achieving high accuracy with minimal computational costs. Due to time constraints, we were unable to implement Self Revision in the medical task, but exploring this method may yield further improvements.

For **SQL Generation**, however, the same approach was less effective, as the rigid output format interfered with the model's ability to generate flexible and accurate SQL queries. Instead, **Self Revision** emerged as the most promising approach for SQL generation, emphasizing the importance of balancing model flexibility and accuracy in tasks involving complex syntax. However, the rolling accuracy in the SQL Generation task appears flat compared to the increasing trend observed in the Medical Diagnosis Classification task. Future work could explore other methods that might produce a more favorable accuracy curve, similar to the medical diagnosis case.

Our results also highlight the challenges of ensuring consistent performance across different datasets. While the model demonstrated similar rolling accuracy trends for both the public and private datasets, the lower accuracy on the private dataset suggests room for improvement in model generalization, particularly in the SQL Generation task. We identified promising approaches, such as Reconcile [9] and Multi-Agent Debate [10], which involve multiple LLMs providing their answers and reasoning before debating to determine the final response. Although these methods show potential, they were not explored due to computational constraints. Further research could investigate these multi-agent approaches to improve model performance and generalization across diverse datasets.

4 Work Distribution

R13922022 王凱右
The following tasks were completed, including experiment ideation, code implementation, and execution of all related experiments:

- For Medical Diagnosis Classification:
  - Role Prompting
  - CoT Role Prompting
  - Multi-Agentic-Memory StreamICL
  - MedPrompt
  - Role Prompting with Format Restrictions
- For Text-to-SQL Generation:
  - Role Prompting with Format Restrictions
  - Schema Linking

Additionally, the Results section of this report was authored.

B09501048 方群鈞
The following tasks were completed, including experiment ideation, code implementation, and execution of all related experiments:

- For Medical Diagnosis Classification:
  - Self-Stream Contrastive Learning
  - Dynamic Self-Stream Contrastive Learning
- For Text-to-SQL Generation:
  - Self Revision

Additionally, the Streaming Method and Conclusion section of this report was authored.

B10902011 賴詠晴

- Presentation video editing
- Report editing

References

[1] C.-K. Wu, Z. R. Tam, C.-Y. Lin, Y.-N. Chen, and H. yi Lee. Streambench: Towards benchmarking continuous improvement of language agents, 2024. URL https://arxiv.org/abs/2406.08747.

[2] N. S. P. Balasubramanian and S. Dakshit. Can public llms be used for self-diagnosis of medical conditions ?, 2024. URL https://arxiv.org/abs/2405.11407.

[3] X. Gao and K. Das. Customizing language model responses with contrastive in-context learning, 2024. URL https://arxiv.org/abs/2401.17390.

[4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[5] H. Nori, Y. T. Lee, S. Zhang, D. Carignan, R. Edgar, N. Fusi, N. King, J. Larson, Y. Li, W. Liu, R. Luo, S. M. McKinney, R. O. Ness, H. Poon, T. Qin, N. Usuyama, C. White, and E. Horvitz. Can generalist foundation models outcompete special-purpose tuning? case study in medicine, 2023. URL https://arxiv.org/abs/2311.16452.

[6] Z. R. Tam, C.-K. Wu, Y.-L. Tsai, C.-Y. Lin, H. yi Lee, and Y.-N. Chen. Let me speak freely? a study on the impact of format restrictions on performance of large language models, 2024. URL https://arxiv.org/abs/2408.02442.

[7] S. Yang, Q. Su, Z. Li, Z. Li, H. Mao, C. Liu, and R. Zhao. Sql-to-schema enhances schema linking in text-to-sql, 2024. URL https://arxiv.org/abs/2405.09593.

[8] S. Talaei, M. Pourreza, Y.-C. Chang, A. Mirhoseini, and A. Saberi. Chess: Contextual harnessing for efficient sql synthesis, 2024. URL https://arxiv.org/abs/2405.16755.

[9] J. C.-Y. Chen, S. Saha, and M. Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms, 2024. URL https://arxiv.org/abs/2309.13007.

[10] T. Liang, Z. He, W. Jiao, X. Wang, Y. Wang, R. Wang, Y. Yang, S. Shi, and Z. Tu. Encouraging divergent thinking in large language models through multi-agent debate, 2024. URL https://arxiv.org/abs/2305.19118.