Fine-tuning Large Language Models for Text-to-SQL Tasks in Agricultural Census Anomaly Detection

Gede Putra Nugraha
Department of Statistical Computing
Politeknik Statistika STIS
Jakarta, Indonesia
222011683@stis.ac.id

Lya Hulliyyatus Suadaa
Department of Statistical Computing
Politeknik Statistika STIS
Jakarta, Indonesia
lya@stis.ac.id

Setia Pramana
Department of Statistical Computing
Politeknik Statistika STIS
Jakarta, Indonesia
setia.pramana@stis.ac.id

2024 International Conference on Electrical Engineering and Informatics (ICELTICs) | 979-8-3503-6682-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICELTICS62730.2024.10776457

***Abstract*—Detecting anomalies from collected data is essential in producing high-quality data for further analysis, including agricultural census data. This research aimed to improve the efficiency of data anomaly checking by utilizing the CodeL*lama- 7B* Large Language Model (LLM) to translate human questions into SQL queries. Spider and WikiSQL datasets are used in the first fine-tuning phase for translating ability to general queries. The agricultural datasets are constructed using questions and queries in agricultural census database schemas and then used in the second fine-tuning phase. The LoRA method is implemented in the fine-tuning process. This fine-tuning enhanced the model's performance on the SQL-Eval framework, achieving an Average Correct Rate of 69.6%. Combining this approach with Retrieval-Augmented Generation (RAG) and prompt engineering with Chain of Thought (CoT) improvedthe Average Correct Rate to 77.8%.**

Keywords— **LLM, Text-to-SQL, LoRA, SQL-Eval, CodeL-lama**

I. Introduction

Data anomalies are conditions where the data entered fromthe questionnaire into the database shows irregularities or discrepancies that do not match the expected patterns or expectations. The aim is to ensure the collected data are high-quality and free from discrepancies so that accurate analysis results can support decision-making. Statistics Indonesia (*Badan Pusat Statistik*/BPS) is the official government institution responsible for collecting data through censuses and surveys in Indonesia. In 2023, BPS carried out an AgricultureCensus to collect data in the agriculture sector.

Detecting anomalies from collected data is essential in producing high-quality data for further analysis. Since the data are stored in databases, anomaly detection is conducted using SQL query. BPS provide list of query examples for standard data anomaly checking in agricultural census. How- ever, the specific needs of each BPS unit in the district/city are different, requiring unique anomaly detection rules. Challenges arise when sophisticated data anomaly is needed, suchas the InterQuartile Range (IQR) method, which requiresmore advanced SQL programming skills. This effort is time consuming and can hamper employee productivity.

Based on the challenges in performing complex SQL programming for data anomaly checking in agricultural census data, this research aims to develop an advanced solution utilizing Large Language Models (LLMs). The focus is on enhancing the ability to generate accurate and complex SQL queries from natural language instructions, without requiring users to construct queries from scratch. LLMs are chosen for their robustness, flexibility, and accurate results in Natural Language Processing tasks, particularly in understanding context and nuances in user instructions. This study employs the CodeLlama-7B LLM, a fine-tuned version of Llama 2 designed explicitly for coding tasks.

The research adopts a multifaceted approach to improving the LLM's performance in the text-to-SQL task. First,the CodeLlama-7B model is fine-tuned using the LoRA (Low-Rank Adaptation) method, focusing on both general SQL syntax and the specific schema of agricultural census databases. The study incorporates Retrieval-Augmented Generation (RAG) to enhance accuracy and provide relevant con-text from the database schema and sample queries. Prompt engineering techniques, including Chain of Thought (CoT), are applied to guide the model's reasoning process.

The use of LLM for text-to-SQL interactions with databases has been widely adapted by large companies such as Pinterest and Snowflake. Pinterest faced challenges with traditional SQL queries that required users to have a comprehensive understanding of SQL syntax, which could lead to potential errors. Subsequently, Pinterest's developmentteam implemented text-to-SQL, promising an improved user experience, reduced errors, and increased productivity [1]. Meanwhile, Snowflake built a text-to-SQL-based ChatBot assistant with the LLM Mistral-Large model to streamline thetime spent writing SQL queries and maintain the quality of the generated queries [2]. The successful utilization of LLM for text-to-SQL tasks indicates that text-to-SQL generation capabilities have the potential to streamline the anomaly-checking process that requires interaction with the database through SQL queries.

II. Related Work

Text-to-SQL models have made significant advancements in recent years [3]. Initially, a rule-based approach relying on templates to generate SQL was used. This approach showed some effectiveness but was heavily dependent on manually defined rules, making it less scalable and difficult to generalize.

As in [4] explored the capabilities of LLMs in generating code, particularly SQL code for databases, in the context of

2024 International Conference on Electrical Engineering and Informatics (ICELTICs) | 979-8-3503-6682-2/24/$31.00 ©2024 IEEE

Text-to-SQL tasks. The research introduced the LLM-based SQL-PaLM model, leveraging the PaLM-2 model to enhance model performance with two methods: in-context learning and fine-tuning. The execution-based self-consistency boosting approach adopted by SQL-PaLM proved effective for Text-to-SQL tasks, achieving a test accuracy of 77.3% on the Spider dataset. This advantage becomes more significant, with a 4% improvement over previous methods. The research also showed that fine-tuned SQL-PaLM provided a 1% performance improvement.

As in [5] detailed the innovative DIN-SQL approach for Text-to-SQL modeling, utilizing LLMs and few-shot prompting by dividing the Text-to-SQL task into four distinct sub- tasks: schema linking, query classification and decomposition, SQL generation, and self-correction. This study demonstarted significant performance improvements, especially on the Spider and BIRD datasets. Additionally, the researchintroduced an adaptive prompting strategy tailored to query complexity and utilized LLMs for self-correction of minor SQL generation errors. Notably, DIN-SQL achieved state-of- the-art results on the Spider and BIRD datasets, outperform- ing many extensively fine-tuned models, thus demonstrating the potential of LLMs for Text-to-SQL generation.

Furthermore, as in [6] developed a framework for translating natural language questions into SQL queries to answer database questions called RAT-SQL. Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers is an advanced framework designed to address challenges in translating natural language questions into SQL queries for databases with various schemas. This framework operates through an integrated framework centered around a relation-aware-self-attention mechanism, allowing explicit encoding of arbitrary relationships between questions and database schemas. By combining representations using self-attention, RAT-SQL enhances the model's understanding of schema linking and alignment. RAT-SQL also handles schema encoding and schema linking, allowing accessibility for semantic parsers and modeling alignment between database columns and query mentions. With these features, RAT-SQL achievedstate-of-the-art performance on the Spider dataset, increasingexact match accuracy to 57.2% and reaching a new state-of- the-art performance of 65.6% when combined with BERT.

The fine-tuning method used in this research is LoRA. Low-Rank Adaptation (LoRA) was introduced in [7]. LoRA proposes freezing the weights of a pre-trained model and inserting trainable low-rank decomposition matrices at each Transformer layer. This method significantly reduces the number of trainable parameters for specific tasks, outperforming the fine-tuning of GPT-3 175B in terms of efficiency. Experiments showed that LoRA produced model quality equivalent to or better than RoBERTa, DeBERTa, GPT-2, and

GPT-3 with significantly fewer parameters, higher training throughput, and no added inference latency. This research provides insights into the effectiveness of LoRA, and the re- searchers provide integration packages with PyTorch as well as implementations and model checkpoints for RoBERTa, DeBERTa, and GPT-2, including for Text-to-SQL tasks.

III. Methodology

Fig. 1 shows the stages in the development of a model for text-to-SQL tasks to be implemented in the chatbot application.

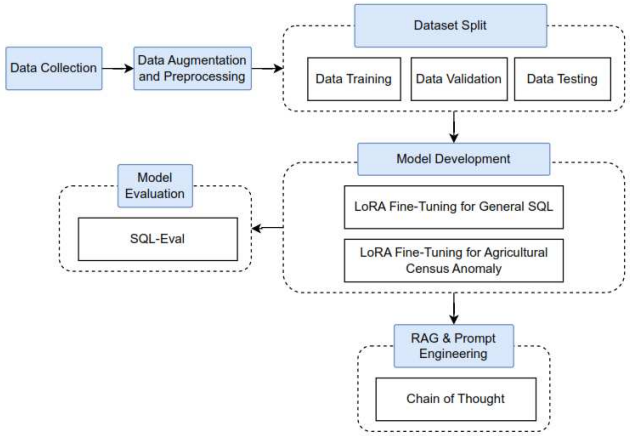

Fig. 1. Text-to-SQL Model Development Flow.

A. Data Collection

The data collected as a dataset for fine-tuning consists of the Spider dataset [8] and WikiSQL [9] obtained from the Hugging Face platform. The variables used are: question, which contains questions in natural language; context, which is the database schema related to the question; and answer, which is the answer to the question in the form of an SQL query. The researchers also collected data related tothe database schema used in the agricultural census query builder system, a system developed by BPS to facilitate anomaly detection through a query builder interface. This query builder system provides a list of anomaly-checking commands, including SQL queries generated from each com-mand. This list was compiled by scraping the query builder website. The results of this scraping will be used to build a particular text-to-SQL dataset for the agricultural census.

B. Units

Data augmentation is carried out to increase the diversity and quantity of available data, as well as provide variety in model testing. This data is generated using data augmentation techniques with the LLM model, Mixtral-8x7B, one of the best open-source models. Fig. 2 shows that the generated dataset will go through a manual validation and filtering stage to ensure that the queries can be executed, as well as to identify duplicates and ensure sufficient data quality and diversity to train the model effectively. After evaluation, the Spider, WikiSQL, and Agricultural Census datasets will be converted into JSON files in the format shown in Fig. 3.

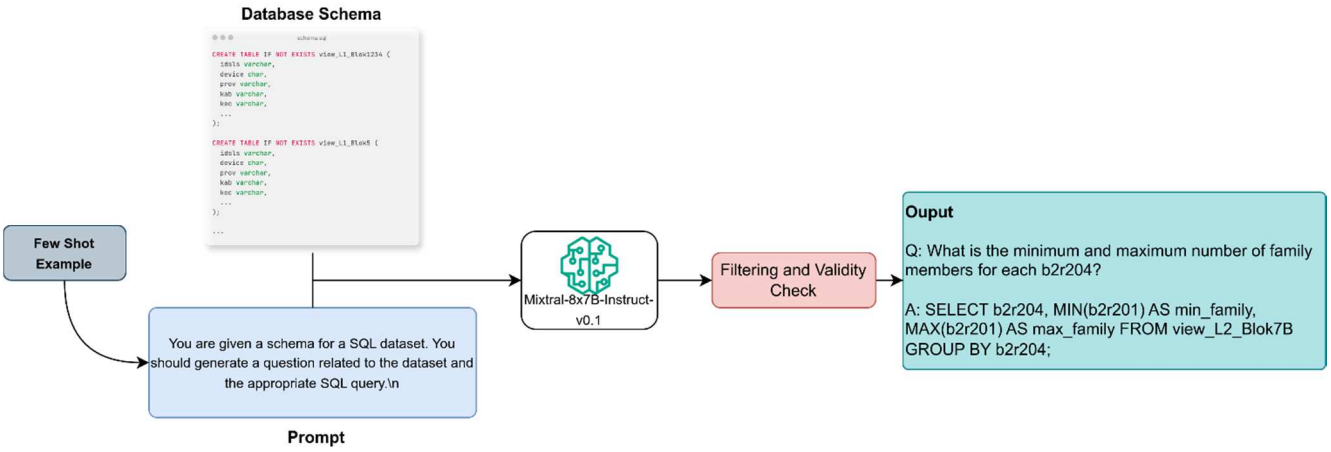

Fig. 2. Augmentation Schema with LLM

fine-tuning.json

```
[
  {
    "question": "What is the 75th percentile of b3r305 for each device?",
    "context": "CREATE TABLE IF NOT EXISTS view_L1_Blok1234 ( idsls varchar, device char, prov varchar, ...);",
    "answer": "SELECT device, percentile_cont(0.75) WITHIN GROUP (ORDER BY b3r305) FROM view_L1_Blok1234 GROUP BY device;"
  },
  ...
]
```

Fig. 3. Dataset Format for Fine-tuning

C. *Data Split*

The datasets created are split into training, validation, and testing sets. The training data is employed for fine-tuning, while the validation data helps ensure model performance and prevents overfitting during the fine-tuning process. Finally, the testing set is used for evaluation.

D. *Two Stage Fine-Tuning*

The CodeL*lama-7B* model will undergo a two-stage fine tuning process using open-source tools Axolotl. In the first stage, the model will be adapted for the general Text-toSQL task by integrating the Spider and WikiSQL datasets. In the second stage, the model will be fine-tuned again to manage Text-to-SQL tasks specific to the agricultural census database schema, utilizing a dataset tailored for that schema. Both stages will employ the Low-Rank Adaptation (LoRA) method. LoRA, introduced by [7], works by freezing the pretrained model weights and adding trainable low-rank decomposition matrices at each Transformer layer. This approach drastically reduces the number of trainable parameters needed for specialized tasks, such as Text-to-SQL. Fig. 4 illustrates how LoRA operates: the input x with dimensions 1 x d is passed through the model, where the pretrained weights W_0 have dimensions $d \times d$ LoRA uses a parameter r, smaller than d, to approximate ΔW via two smaller matrices, A $(d \times r)$ and B $(r \times d)$. During fine-tuning, only A and B are updated. After fine-tuning, the input x is multiplied by both W_0 and the adjusted weights (approximated by $A \times B$). These results are then summed element-wise to generate the final output h.

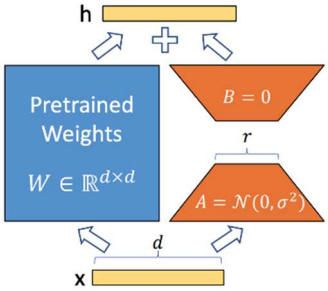

Fig. 4. How LoRA Works

E. *RAG and Prompt Engineering*

The Retrieval-Augmented Generation (RAG) technique is used in this research to enhance the accuracy of generative AI models by leveraging facts from external sources [10]. RAG will be applied to the database schema of the Agricultural Census of Individual Farmer Enterprises (UTP) 2023 to generate accurate SQL queries. Combining RAG with prompting techniques such as Chain of Thought (CoT) will enhance reasoning for questions requiring complex queries. CoT enables the model to generate intermediate reasoning steps, akin to human thought processes in solving multi-step problems [11].

F. Evaluation

The fine-tuned model is evaluated using the SQL-eval framework from Defog.ai to assess the correctness of the generated SQL queries compared to the actual gold queries. This framework validates query results deterministically, accounting for acceptable variations. Firstly, all column combinations from the gold queries are generated and executed on the database, and the results are stored as a panda's data frame.The generated queries are also executed, and their results are compared to the gold queries using an exact match comparison that disregards data types. If an exact match is found,the query is considered correct. To accommodate variations such as column aliases or different row orders, a "subset evaluation" technique is employed. This technique ensures that the gold query results are a subset of the generated query results by checking and normalizing relevant columns, thus marking harmless variations as correct. The metric for this evaluation is the Average Correct Rate, formulated in Equation 1.

$$ACR = \frac{Number\ of\ Correct\ Outputs}{Total\ of\ the\ Outputs} \quad (1)$$

IV. Result and Discussion

A. Dataset Construction

The final datasets are constructed from a combination of the Spider, WikiSQL, and Agricultural Census datasets. The datasets for the first fine-tuning stage consist of 78,600 question, context, and answer pairs from the combined Spider and WikiSQL datasets. For the second fine-tuning stage,the augmented agricultural census dataset consists of 1,447 questions, context, and answer pairs, as shown in Fig. 5. Table I shows the total amount of data used for fine-tuning and evaluating the model. I.

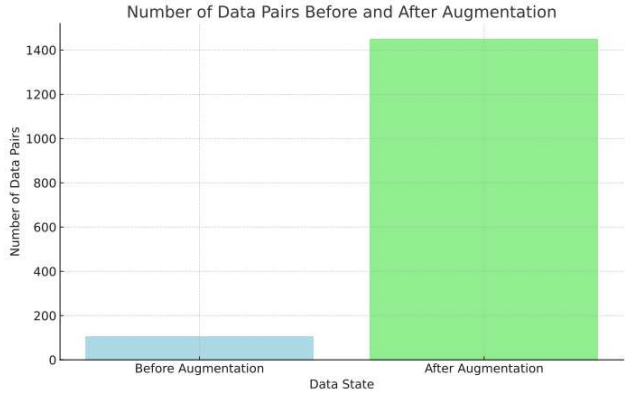

Fig. 5. Dataset Before and After Augmentation

TABLE I. DATASET STATISTICS FOR FINE-TUNING

Dataset	*Training*	*Validation*	*Testing*	*Total*
Spider & WikiSQL	62880	7860	7860	78600
Agricultural Census	1200	174	73	1447

B. Two-Stage Fine-Tuning

For the two-stage fine-tuning, the researchers combined LoRA with DeepSpeed ZeRO-3 to efficiently partition the base model and training state across GPUs, enabling faster and more efficient training [12]. This study also integrated flash attention to support a fast, memory-efficient attention system during training. However, it should be noted that this only works with specific hardware like NVIDIA A100s [13]. Gradient check pointing techniques were also used to reduce VRAM footprint, allowing larger batch sizes and higher training throughput [14].

C. RAG and Prompt Engineering

Retrieval-Augmented Generation (RAG) is used as a tool to store the agricultural census database schema in a contextual format. When a user asks a question, RAG performs a search using Cosine Similarity measurement on the vector database to find schemas with semantic similarity to theuser's question. Fig. 6 shows the implementation of RAG in the ChatBot application. Additionally, RAG is integrated withthe prompting technique, Chain of Thought (CoT). CoT is used to enhance the reasoning ability of the fine-tuned model, particularly in selecting tables and columns, generating SQL queries, and visualizing data.

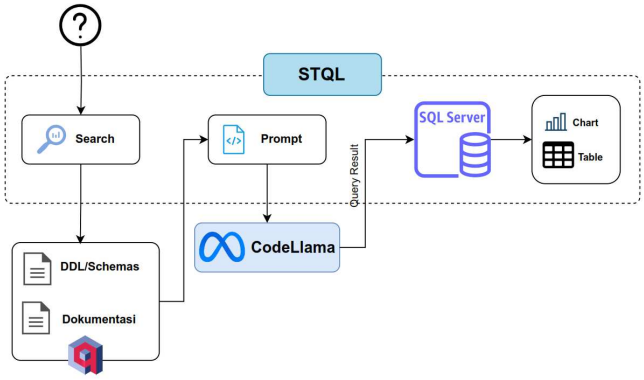

Fig. 6. RAG Schema

D. Evaluation

Table II displays the experimental results of the built model. Based on the results, the CodeLlama-7B model, which has undergone the fine-tuning process, shows improvement, as seen in the Average Correct Rate metric, which increased from 53.9% to 69.6% for the fine-tuned model and 77.8% for the model using RAG and CoT methods.

TABLE II. TEXT-TO-SQL MODEL EXPERIMENT RESULTS USING THE AVERAGE CORRECT RATE SQL-EVAL METRIC

Model	*Average Correct Rate*
CodeLlama-7B Base	53.9%
CodeLlama-7b Fine-tuned	69.6%
CodeLlama-7b Fine-tuned + RAG + CoT	77.8%
CodeLlama-7B Base	53.9%

The performance comparison results of our fine-tuned CodeLlama-7B with GPT models show that the developed model can compete with the well-known GPT model. Table III indicates that the model with significantly smaller parameters, fine-tuned in this study with the integration of RAG and CoT techniques, is able to compete with advanced GPT models.

TABLE III. COMPARISON OF MODEL PERFORMANCE

Model	**Parameters** ***(Billion)***	**Execution Time** ***(s/query)***	***Average Correct Rate***
GPT-3.5-Turbo	20	4	67.0%
GPT-4	1760	10	86.0%
GPT-4-Turbo	1760	5	83.0%
CodeLlama-7b Fine-tuned + RAG + CoT	7	6	77.8%
GPT-3.5-Turbo	20	4	67.0%

Fig. 7 shows the example of output generated when the model was asked a question in natural language.

Query: Are there any records that have a building area larger than the land area?
Original Answer: SELECT b1r112a, b1r112b, b1r112a/b1r112b FROM view_L2_Blok1 WHERE ((b1r112a > b1r112b)
Generated Answer: SELECT b1r112a, b1r112b, b1r112a/b1r112b FROM view_L2_Blok1 WHERE ((b1r112a > b1r112b)

Fig. 7. Generated Output Example

V. CONCLUSION

This study has constructed a dataset for text-to-SQL tasks consisting of question-and-answer pairs using SQL queries on the agricultural census database schema, which can be used in future research. This dataset is used to fine-tune the CodeLlama-7B Large Language Model (LLM) to translate human questions into SQL queries for efficient data anomaly detection. The fine-tuning method using LoRA has proven to enhance the performance of the CodeLlama-7B model, even when training on a small fraction of the model's parameters. The LoRA fine-tuning method increased the performanceof the CodeLlama-7B model by 29.1% on the SQL-Eval evaluation metric, while the implementation of Retrieval-Augmented Generation (RAG) and Chain of Thought (CoT) further boosted the model's performance by 44.34% on the SQL-Eval evaluation metric. This fine-tuning method allows the model to compete with other state-of-the-art models, such as GPT.

REFERENCES

[1] P. Engineering, How we built Text-to-SQL at Pinterest — medium.com. https://medium.com/pinterest-engineering/how-we-built-text-to-sql-at-pinterest-30bad30dabff. [Online]. Available: https://medium.com/pinterest-engineering/how-we-built-text-to-sql-at- pinterest-30bad30dabff.

[2] Y. O. Pieter Verhoeven, A Breakthrough AI-Powered SQL Assistant. https://www.snowflake.com/blog/copilot-ai-powered-sql-assistant/, 2024. [Online]. Available: https://www.snowflake.com/blog/copilot-ai- powered-sql-assistant.

[3] B. Qin, B. Hui, L. Wang, M. Yang, J. Li, B. Li, R. Geng, R. Cao, J. Sun, L. Si, and F. Huang, "A Survey on Text-to-SQL Parsing: Concepts, Methods, and Future Directions," arXiv preprint arXiv:2208.12345, 2022.

[4] R. Sun, S. Ö. Arik, A. Muzio, L. Miculicich, S. Gundabathula, P. Yin, H. Dai, H. Nakhost, R. Sinha, Z. Wang, and T. Pfister, "SQL- PaLM: Improved Large Language Model Adaptation for Text-to-SQL (extended)," arXiv preprint arXiv:2305.13267, 2024.

[5] M. Pourreza and D. Rafiei, DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction. 2023.

[6] B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson, RAT- SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. 2021.

[7] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," arXiv preprint arXiv:2106.09685, 2021.

[8] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, "Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task," arXiv preprint arXiv:1809.08887, 2019.

[9] V. Zhong, C. Xiong, and R. Socher, Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. 2017.

[10] . P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Facebook AI Research, University College London, New York University, 2021.

[11] 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.

[12] R. Y. Aminabadi, S. Rajbhandari, M. Zhang, A. A. Awan, C. Li, D. Li, E. Zheng, J. Rasley, S. Smith, O. Ruwase, and Y. He, "DeepSpeed Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale," 2022.

[13] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. 2022.

[14] A. Griewank and A. Walther, "Algorithm 799: revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation," ACM Trans. Math. Softw., vol. 26, no. 1, pp. 19–45, Mar. 2000.