Cubicpower Agentic Mixture of Experts (AMoE) Framework for Fine-Tuning NLP Tasks Without GPUs

Chao-Yih Hsia(夏肇毅)
Smart Academy
CubicPower Smart Center
Taipei,Taiwan
chaoyihhsia@gmail.com

Abstract

The rise of Green AI emphasizes minimizing the environmental footprint of AI systems. This paper explores a no-GPU agentic architecture for fine-tuning NLP tasks. It presents our initial experiments applying these no-GPU algorithms in pretraining and fine-tuning tasks on our CubicPower agentic mixture of experts (AMoE) framework, with the aim of contributing to more sustainable AI development. In contrast to the training procedures of neural networks, which consume significant power, the AMoE framework's primary contribution toward power savings is that it requires no training process. We explore non-neural-network methods for solving NLP tasks and employ similarity measures to match predefined patterns for use in a RAG database.

Keywords: Green AI, MoE, RAG, CubicPower, AMoE.

1 Introduction

In recent years, many countries have set a 2050 net-zero emissions goal. Energy conservation has become a top priority across all industries. However, AI neural network algorithms, such as the Bitcoin Proof-of-Work (PoW) algorithm, rely heavily on GPUs or other custom-designed accelerators. These machine learning training processes, using the gradient descent method, can take weeks or months to run on large numbers of high-power-consuming GPUs. Therefore, many solutions have been developed to save energy (Verdecchia et al., 2023). However, we believe that a no-GPU Green AI algorithm could be a new and effective direction (Hsia, 2022), since it eliminates the primary source of power consumption.

Traditional text mining algorithms use parameters to measure word properties, such as TF-IDF and similarity. TF-IDF measures the importance of a word, while similarity measures the distance between words. These algorithms are not neural networks and, of course, do not involve any gradient descent training process. We have developed algorithms based on text similarity to select the most similar text from the pattern pool.

This paper presents our initial experiments applying such no-GPU algorithms in pretraining and fine-tuning tasks on our CubicPower agentic mixture of experts (AMoE) framework, aiming to contribute toward more sustainable AI development.

While MoE and RAG approaches have improved efficiency, most still rely on GPU computation. We propose a GPU-free AMoE framework using similarity-based retrieval to fine-tune NLP tasks.

The main contributions of this paper are as follows:
1. Exploration of non-neural-network methods for solving NLP tasks.
2. Elimination of the training process in the AMoE framework to save power.
3. Use of similarity measures to match predefined patterns for retrieval in a RAG database.

2 Related Work

Early dialogue systems evolved from rule-based methods, such as ELIZA (Weizenbaum, 1966), which applied pattern-matching rules to simulate human-like responses. This approach laid the foundation for later systems, such as GUS (Bobrow, 1977), which introduced a frame-based architecture. In GUS, dialogues were organized into structured templates containing slots, enabling simple task-oriented conversation handling.

Modern systems have shifted toward neural architectures. The sequence-to-sequence (seq2seq) model, originally designed for machine translation (Sutskever et al., 2014; Bahdanau et al., 2015), was later adapted for chatbot design. It uses an encoder-decoder structure and autoregressive generation. These models are typically powered by GPU-intensive training and inference pipelines.

To reduce computation costs, retrieval-based systems have re-emerged, using similarity metrics (e.g., cosine similarity) to find the most relevant response from a pattern database. This is often more power-efficient than generation-based models. Retrieval-Augmented Generation (RAG) combines neural language models with external information retrieval, offering enhanced relevance and scalability (Gao et al., 2023).

Similarity search plays a crucial role in these systems. Johnson et al. (2019) proposed a billion-scale similarity search framework using GPUs, while Han et al. (2023) surveyed vector databases and their indexing strategies. In contrast, Hsia (2022) developed a GPU-free similarity-based system, forming the basis of the CubicPower knowledge base, which enables fast and structured retrieval.

Another important concept for reducing computation is the Mixture of Experts (MoE). MoE architectures achieve scalability by activating only a small subset of the model's parameters for each input, allowing for high model capacity without proportional increases in computation. Shazeer et al. (2017) demonstrated this with the Sparsely-Gated MoE, where only a few experts are selected per example, reducing computational cost while preserving performance.

The rise of Green AI (Verdecchia et al., 2023) emphasizes minimizing the environmental footprint of AI systems. Techniques that reduce power consumption, including rule-based reasoning, task-specific similarity retrieval, and agent-level model decomposition, align with this goal. This paper explores a no-GPU agentic architecture for fine-tuning NLP tasks.

3 Methodology

In this paper, we develop the entire AMoE framework based on the CubicPower Data Processing Engine for similarity computation, following the description in Hsia (2022). The framework was implemented in C# .NET.

3.1 Agentic Architecture

We define AI agents as modular components, each responsible for a specific NLP fine-tuning task, such as question answering (QA), reading comprehension (RC), or chatbot dialogue state tracking. Each agent maintains a local dataset and operates independently, processing only the inputs relevant to its task domain. This follows a Mixture of Experts (MoE) model design but is implemented without neural networks.

3.2 Retrieval-Augmented Module

Figure 1. shows the design of our AMoE framework to perform the retrieval-augmented generation (RAG) function.

Each agent is equipped with a sentence-level retrieval mechanism. It consists of a vector database which stores sequence to sequence (seq2seq) pair records such as question-answers.

Given an input, the agent generates a corresponding sentence vector and compares it against stored records by dot-product to compute

their similarities. Then the system finds the record i with the highest similarity. Extracting the second part of the seq2seq pair, we can find the answer to the question. By leveraging these structures, we can operate the retrieval-augmented generation (RAG) process effectively.

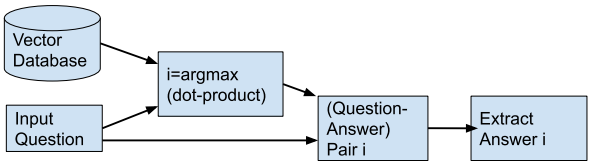

Figure 1. Our approach

3.3 Dataset and Procedures for Fine-Tuning Tasks

All datasets are stored in plain text format to ensure efficient loading and access by intelligent agents. This format facilitates rapid retrieval, parsing, and integration into downstream tasks such as question answering, multiple-choice tasks, and reading comprehension.

3.3.1 Question Answering (QA) Task:

The dataset for the QA task consists of question–answer pairs, as shown in Figure 2. We have collected sets of question–answer pairs. To perform the QA task, we need to analyze the QA training dataset to construct the overall word distribution. First, we sample the QA training dataset to construct the overall QA word distribution:

$$D=\text{Sample}(\text{QA training set}) = M_{W->P} \quad (1)$$

Here D is the distribution of the current QA words. This distribution is used to map each word to a sentence. The output is a word-to-paragraph map $M_{W->P}$. It is used to compute the most similar paragraphs from a group of words in a question.

Following the description in Hsia (2022), we can implement a similarity-based system using the distribution D to find paragraphs from a word. Each paragraph is a question–answer pair.

We then build a paragraph-based RAG module RAG_D to select answers from RAG_D for the questions.

Denote $RAG_D()$ as a RAG module based on the distribution D. Once we feed a question into this module, the output paragraph from this module for a question becomes:

$$\text{paragraph}_{RAGD} = RAG_D(\text{question}) \quad (2)$$

We can therefore obtain the answer to the question as a QA RAG output answer:

$$\text{answer} = \text{answerOf}(\text{paragraph}_{RAGD}) \quad (3)$$

The answerOf() function in (3) returns the answer from a paragraph containing a question–answer pair.

問題	答案
證券商接受客戶委託買賣有價證券時，應遵循哪些相關規範？	證券商接受客戶委託買賣有價證券時，應依據「證券交易法」及「證券商辦理有價證券買賣業務委託契約範本」辦理。證券商應查核委託人身分，確保交易合法，並應揭示風險、收取合法費用。若為電子下單，亦應提供適當資訊安全保護機制。

Figure 2. Dataset for the question answering task

3.3.2 Multiple Choice (MC) Task:

The dataset consists of question–option–answer triples, where each record contains a question, options A–D, and the correct answer, as depicted in Figure 3.

Each multiple-choice question can be reformulated into four independent True or False questions, allowing the system to evaluate each option separately.

Alternatively, the task can be approached as a QA problem by checking the answer to existing questions in the training set.

For unseen questions, we must learn the question–answer relationships from the training set and select the option whose relationship most closely matches the learned patterns.

問題	選項1	選項2	選項3	選項4	答案
企業的主要目的是？	提供就業機會	追求利潤	創造藝術	進行研究	追求利潤

Figure 3: Dataset for the multiple choice task

3.3.3 Reading Comprehension (RC) Task:

Similar to the QA task, we need to analyze the word distribution for the RC task. However, the source of the word distribution is not the training set; it comes from each RC question. Therefore, we must resample the RC question each time to reconstruct the RC word distribution for that question.

In order to answer a question in the RC task, we first resample the RC document i in the test dataset to extract the word distribution D_i of the RC question i.

$$D_i = \text{Resample}(\text{RC document i}) = M_{i\ W->P} \quad (4)$$

Here D_i is the word distribution of the current RC question i. This distribution is used to map each word to a paragraph.

Following the same method as QA, we can build a paragraph-based RAG module RAG_{Di} to select answers for the questions.

Here we denote $RAG_{Di}()$ as a RAG module based on the distribution D_i. The output paragraph for a question becomes:

$$\text{paragraph}_{RAGDi} = RAG_{Di}(\text{question}) \quad (5)$$

We can therefore obtain the answer to the question from the RC RAG output:

$$\text{answer} = \text{answerOf}(\text{paragraphs}_{RAGDi}) \quad (6)$$

The answerOf() function in (6) returns the answer from a paragraph containing a question–answer pair.

閱讀內容	問題1	答案1	問題2	答案2	問題3	答案3
管理是指透過規劃、組織、領導與控制等程序，有效且效率地達成組織目標的過程。有效性代表能完成正確的工作，效率則是指以最少的資源完成任務。現代管理學者強調「以人為本」與「持續改善」的概念，認為管理不僅是資源配置，更涉及激勵員工與創造學習文化。	管理的四大功能有哪些？	規劃、組織、領導與控制	什麼是管理中的「效率」？	以最少資源完成任務	現代管理強調什麼價值觀？	以人為本與持續改善

Figure 4. Dataset for the reading comprehension task

3.3.4 Chatbot (CB) Task:

The chatbot dataset consists of paired utterances, each representing a conversational turn, as illustrated in Figure 5. The task involves predicting the next appropriate response based on the current user input.

聽到	回答
早安，今天的天氣如何？	早安！今天陽光明媚，氣溫約25度，非常適合外出。

Figure 5. Dataset for the chatbot task

3.4 Power-Efficient Design

In contrast to the training procedures of neural networks, which consume significant power, the main contribution of the AMoE framework to power saving is that it requires no training process.

Additionally, the CubicPower AMoE framework consists of many agents. Each agent stores only a small portion of data relevant to its task. This follows the Mixture of Experts (MoE) method (Lepikhin et al., 2020; Fedus et al., 2022). In our system, the experts are agents. Therefore, only a small amount of power is consumed at any given time. Furthermore, we can split the data by language, geographical location, and type, assigning each subset to a different agent. The

system decides which agent should handle the input based on the content of the prompt.

4 Experiments

The experiment in this study relies on a similarity metric. Similarity is measured as the proportion of words in the correct answer that also appear in the predicted answer.

This measure is conceptually similar to BLEU-1 (Bilingual Evaluation Understudy), which assesses word overlap between reference and generated text.

4.1 Experimental Setup

All experiments were conducted on a standard CPU-based machine without GPU acceleration. The framework was implemented using C# .NET and utilized the CubicPower Data Processing Engine's classical text processing libraries for cosine similarity computation.

Each task-specific agent was evaluated independently using a dedicated dataset, split into training and testing subsets. The training set served as the retrieval base for the test queries.

4.2 Datasets

We prepared different datasets for fine-tuning tasks. We used small private datasets collected by CubicPower. Each dataset contains several hundred records.

For the Question Answering task, the dataset consists of a question and an answer field (see Figure 2). When a QA agent receives a QA request with a question, it searches the question field of the database and returns the most similar QA record.

For the Multiple Choice task, our dataset was prepared as shown in Figure 3. For each question, there are four options. The final field contains the answer to the question. Each question is essentially a combination of four true or false questions. By testing each of the four true or false questions, only one of them will be true.

The Reading Comprehension task first provides a document and then asks a series of questions based on that document.

We aim to answer the questions using only the material provided in the document; therefore, we need to build a word space derived from this document. Figure 4 shows a sample of the RC dataset.

Table 1 lists the sizes of the training and test sets for all four fine-tuning tasks used in our experiments.

Task	**Training Set Size**	**Test Set Size**
QA	749	371
MC	440	181
RC	—	619
CB	1121	389

Table 1: tasks train/test Dataset Size

5 Results

5.1 Fine-tuning Tasks Test:

We loaded the training dataset for the QA task into our database and then used it to verify the search results. Figure 6a shows a screenshot of the verification results on the training set. We can see that the top-1 accuracy is 0.847, and the similarity between the question and the returned answer is 0.983.

Then, we used the test dataset to query the training database. Figure 6b shows a screenshot of the test results. The results are near zero since there should be no overlap between the training and the test datasets. The nonzero result indicates

that some data leakage exists between the two datasets.

Figures 7 to 9 show the remaining test result screenshots for the MC, RC, and CB tasks. Table 2 summarizes their test results.

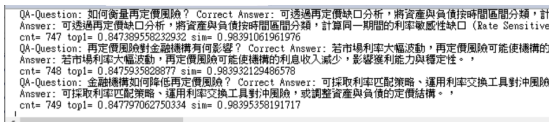

Figure 6a. Figure 6a. QA Train Verification Result

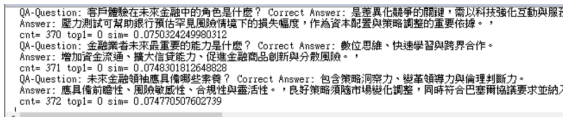

Figure 6b. QA Test Result

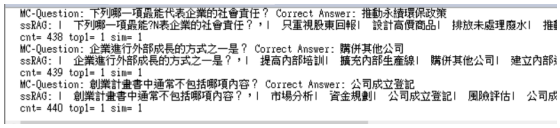

Figure 7a. MC Train Verification Result

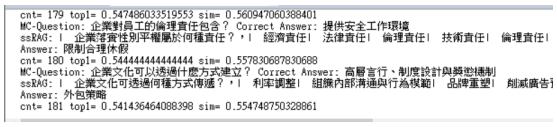

Figure 7b. MC Test Result

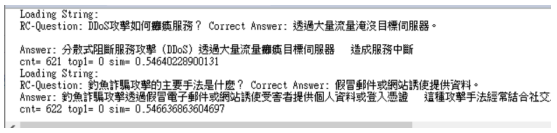

Figure 8. RC RAG Test Result

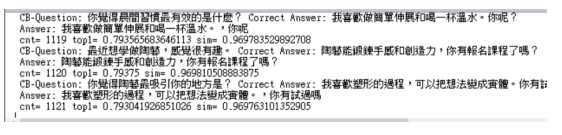

Figure 9a. CB Train Verification Result

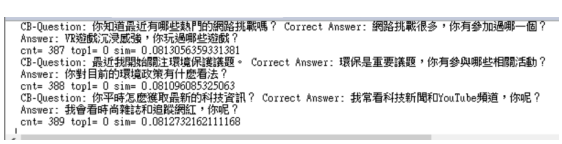

Figure 9b. CB Test Result

Table 2. Task Training/Test Similarity

Task	Similarity (Train)	Similarity (Test)
QA	0.983	0.074
MC	1	0.554
RC		0.546
CB	0.969	0.081

5.2 Exams Test:

We evaluated the performance of our AMoE system using three datasets. The first dataset includes the Taiwan government employee entrance tests and the Financial Institution Certification. The second dataset contains the Taiwan Government Professional Certifications. The third dataset is the Taiwan Massive Multitask Language Understanding Plus (TMMLU+) dataset (Tam et al., 2024).

Figure 10 shows screenshots of the test results, and Table 3 summarizes these results. The first test includes 33,608 training records and achieves an accuracy of 0.354. The second test contains 20,807 training records, achieving an accuracy of 0.283. The third test has 21,120 records and achieves a test accuracy of 0.289.

Table 3. Exam Test Results

	Train set	Data Set	MC Task Accuracy
Financial Institution Certifications / government employee entry test.	33,608	26,985	0.354
Government Professional Certifications	20,807	2,069	0.283
TMMLU+	21,120	2,225	0.289

5.3 Benchmarking Test:

To compare the performance with other Traditional Chinese LLM models, we tested the TMML+ benchmark dataset using zero-shot and 5-shot settings.

Table 4 presents the TMML+ benchmark results for different LLM models reported by Tam et al. (2024). The results show that the zero-shot average accuracy of Breeze-7B-Instruct-v1.0 is 36.1%, which is higher than our 25.1%. However, the other two models, Taiwan-LLaMa-13B and Taiwan-LLaMa-7B, achieved accuracies of 21.3% and 15.6%, respectively. The performance of our AMoE framework in the Traditional Chinese TMMLU+ test ranks second among the compared models.

Table 4. Comparative Results on TMMLU+: (*from Tam et al., 2024)

LLM Models	Zero-shot accuracy (%)	5-shot accuracy (%)
*Breeze-7B-Instruct-v1.0	36.1	28.6
CubicPower AMoE	**25.1**	**25.7**
*Taiwan-LLaMa-13B	21.3	22.3
*Taiwan-LLaMa-7B	15.6	5.1

Figure 10a. Financial Institution Certifications / Government Employee Entry Tests

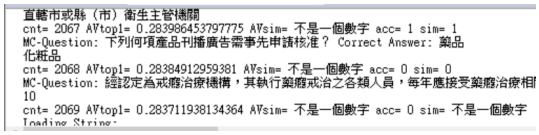

Figure 10b. Government Professional Certifications

Figure 10c. TMMLU+ Test Results

5.4 Discussion

The results indicate that the AMoE framework performs poorly on unseen data in the QA and CB tasks. One possible improvement is to expand the scope of the training dataset.

Additionally, the MC task accuracies in the Government Professional Certifications and TMMLU+ datasets are around 0.28, which is only slightly above random guessing. Although we rank second in the TMMLU+ Traditional Chinese test, there is still considerable room for improvement.

These challenging tests require extensive reasoning before an answer can be generated. As a result, it is difficult to apply a simple QA-style predefined answer list to solve them.

To address this, our next step will be to develop a reasoning agent that applies the chain-of-thought (CoT) method to complex problems.

6. Conclusion

The rise of Green AI emphasizes minimizing the environmental footprint of AI systems. Techniques that reduce power consumption, including rule-based reasoning, task-specific similarity retrieval, and agent-level model decomposition, align with this goal. Traditional text mining algorithms use parameters to measure word properties, such as similarity. We

propose a GPU-free AMoE framework using similarity-based retrieval to fine-tune NLP tasks.

This paper explores a no-GPU agentic architecture for fine-tuning NLP tasks. It presents our initial experiments applying these no-GPU algorithms in pretraining and fine-tuning tasks on our CubicPower agentic mixture of experts (AMoE) framework, with the aim of contributing to more sustainable AI development. In contrast to the training procedures of neural networks, which consume significant power, the AMoE framework's primary contribution to power savings is that it requires no training process. We have developed basic functionalities, but there is still room for improvement. To address this, the next step of our research will be to develop a reasoning agent using the chain-of-thought (CoT) method for complex problems.

References

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, & Illia Polosukhin. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30. https://arxiv.org/abs/1706.03762

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3, 1137–1155. https://doi.org/10.1162/153244303322533223

Bobrow, D. G. (1977). GUS, a frame-driven dialog system. *Artificial Intelligence*, 8(2), 155–173. https://doi.org/10.1016/0004-3702(77)90018-2

Cheng, H., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., ... & Shah, H. (2016). Wide & deep learning for recommender systems. *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems* (DLRS 2016). https://doi.org/10.48550/arxiv.1606.07792

Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/2101.03961

Gao, Y., Zhao, Y., Zhang, Y., Liu, Z., & Ding, G. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint*. https://simg.baai.ac.cn

Han, Y., Liu, C., & Wang, P. (2023). A comprehensive survey on vector database: Storage and retrieval technique, challenge. *arXiv preprint*. https://arxiv.org/abs/2310.11703

Hsia, C.-Y. (2022). Design of CubicPower real-time topic writing knowledge base system based on similarity (以相似度為基礎之CubicPower即時主題寫作知識庫系統設計) [Conference presentation]. *TANET 2022 Taiwan Internet Seminar*, Taiwan. https://drive.google.com/open?id=13PQnzzDIHSEFTfeX4NYWAMydlP0MwNh8

Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 5(2), 1–11. https://doi.org/10.1109/TBDATA.2019.2902270

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., ... & Chen, Z. (2020). GShard: Scaling giant models with conditional computation. *arXiv preprint*. https://arxiv.org/abs/2006.16668

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint*. http://arxiv.org/abs/1301.3781

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint*. https://arxiv.org/abs/1701.06538

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27, 3104–3112. http://cs224d.stanford.edu/papers/seq2seq.pdf

Tam, Z.-R., Pai, Y.-T., Lee, Y.-W., Chen, J.-D., Chu, W.-M., Cheng, S., & Shuai, H.-H. (2024). TMMLU+: An improved Traditional Chinese evaluation suite for foundation models. arXiv.

https://doi.org/10.48550/arXiv.2403.01858

Verdecchia, R., Sallou, J., & Cruz, L. (2023). A systematic review of Green AI. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. https://doi.org/10.1002/widm.1507

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1), 36–45. https://doi.org/10.1145/365153.365168