Enhancing the RAG Retrieval Engine through Multi-Encoder Fusion

Jie Shen[1,2] , Huaiyuan He[1,3], Wenzhuo Shen[4], Tiyan Shen[1,3,*]
[1]PKU-WUHAN Institute for Artificial Intelligence, Wuhan, 430223, China
[2]Wuhan Research Institute of Posts and Telecommunications, Wuhan, 430074, China
[3]School of Government Peking University, Beijing, 100871, China
[4] School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, 100876, China
[*] Corresponding author: tyshen@pku.edu.cn

2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI) | 979-8-3503-8694-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICECAI62591.2024.10674962

***Abstract*—With the rapid advancement of large language model technology, Retriever-Augmented Generation (RAG), which is based on vector databases, has demonstrated extensive potential for application. This paper focuses on the text retrieval phase of RAG, enhancing text recall by introducing multiple text encoders. Consequently, this paper designs a neural network model and loss function for multi-encoder fusion and determines the optimal weight allocation for multi-encoder fusion through neural network training. Experimental results demonstrate that the predicted fusion weights from the neural network trained in this paper can be closely aligned with the optimal fusion weights. By using the predicted BCE and BGE encoding model weights, an improvement of 6% in Mean Reciprocal Rank (MRR) is achieved.**

Keywords-component; Retriever-Augmented Generation; Vector database; Large Language Model

I. INTRODUCTION

As a data storage and retrieval system, vector databases rely on vector-based data representation and retrieval techniques. The crux lies in swiftly pinpointing data highly analogous to a given query within the vector database by computing the similarity between vectors. This capability addresses diverse query requirements for complex data across various domains such as image recognition, natural language processing, and recommendation systems. In the domain of image recognition, for instance, images are transformed into vector data facilitating the rapid retrieval of other images resembling the target image[1]. In natural language processing, texts are transformed into vectors enabling the search for similar texts, sentences, or phrases[2]. Similarly, in recommendation systems, user profiles and product vectors are vectorized, facilitating personalized recommendations by searching for products that closely align with user preferences through similarity computation[3].

Following the emergence of large language models (LLMs), their generated text exhibits not only the drawback of "hallucination" but also a deficiency in domain-specific knowledge. To address this limitation, industries commonly employ auxiliary knowledge repositories to supplement additional domain expertise. Retriever-Augmented Generation technology represents a crucial application approach for LLMs integrated with vector databases[4]. The specific procedure entails converting text from the corpus into vector form using an encoder and storing it in the vector database. Similarly, user queries are encoded into vectors, enabling similarity-based retrieval from the database to retrieve relevant text. When combined with the large language model, this approach furnishes users with precise answers.

The inability of Retrieval-Augmented Generation (RAG) retrievers to retrieve relevant text, or worse, to retrieve irrelevant text, can significantly impact the quality of generated text. Given the direct influence of retriever recall on the quality of generated text answers, enhancing this metric is paramount in the field of text retrieval. Recognizing the diverse strengths of different vector encoders in feature extraction and emphasis, an effective approach to enhancing overall vector retrieval efficiency involves integrating multiple encoders. Through this method, a significant improvement in text recall rate can be achieved, thus enhancing the effectiveness of text retrieval[5].

II. PREPARATORY KNOWLEDGE

In the process of optimizing the performance of Retrieval-Augmented Generation (RAG) retrievers, the selection of text encoding models occupies a central position. An exemplary text encoding model not only effectively distinguishes between different texts but also ensures that these texts maintain significant distance differences within the vector space. Such distance discrepancies play a crucial role in enhancing the distinctiveness between texts.

In the process of text vectorization encoding, this paper selects the following three outstanding open-source text encoding models. BCEmbedding[6], an advanced vector encoding model released by NetEase, serves as the core encoding model for the QAnything search engine. BCEmbedding significantly enhances the performance of the search engine with its efficient and precise text representation capabilities. BGE, short for BAAI General Embedding, is a text encoding model developed by the BAAI (Beijing Academy of Artificial Intelligence) Research Institute team[7]. M3E, Moka Massive Mixed Embedding, is an advanced text embedding model in the field of natural language processing. This model is trained, open-sourced, and evaluated by MokaAI, aiming to address semantic understanding and representation in scenarios involving mixed Chinese and English text[8].

Vector encoding models are typically trained on large sets of similar text data, each employing distinct training methods and datasets. In order to fully leverage the strengths of different encoding models and compensate for their shortcomings, this paper proposes a method of multi-encoder fusion. We utilize the Qwen-14B-Chat[9] large language model to construct an RAG question-answer dataset. During this process, the fusion model

2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI) | 979-8-3503-8694-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICECAI62591.2024.10674962

is trained on the dataset to determine appropriate fusion weights. By designing a loss function, the fusion model adjusts the weights of multi-encoder fusion during the training process until the optimal model weights that yield the best fusion performance are identified.

III. SYSTEM MODEL

As illustrated in Figure 1, the model flowchart of the proposed multi-encoder fusion is depicted. The "Query" represents the user's query, which undergoes encoding by two encoding models. Subsequently, text retrieval is conducted in two vector databases. These databases store identical texts but with different encoded vectors. Following the vector retrieval strategy, disparate similarity scores are computed using similarity algorithms for the same text. Different weights are then assigned for fusion based on these scores.

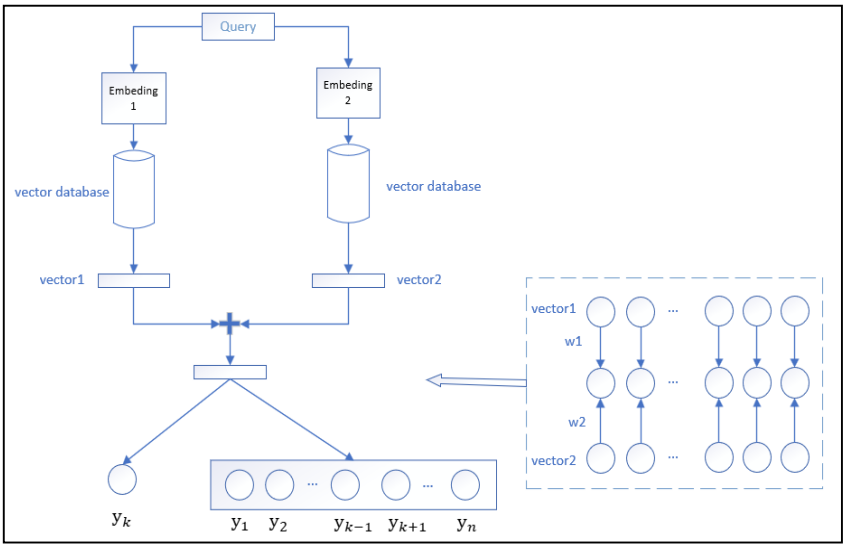

Figure 1. Flowchart of the Multi-Encoder Fusion Model

Let $node1_i$ denote the node recalled by encoder 1, and $node2_i$ denote the corresponding node recalled by encoder 2. Let y_i represent the new node obtained after the fusion of the two.

$$y_i = w_1 \times node1_i + w_2 \times node2_i \quad (1)$$

Equation (1) presents the network architecture of the fusion model, employing a linear layer to integrate nodes recalled by different encoders. w_1 and w_2 are learnable parameters, which are adjusted during the process of model training.

Assuming y_k represents the target node expected to be recalled by the query, partitioning y_k from the candidate nodes is performed. This partitioning is employed to compute the difference between y_k and other nodes, thereby seeking optimal values for w_1 and w_2. These optimal weights ensure that the fusion of the two encoders yields the optimal node.

$$v = \log\left(\sum_{i \neq k}^{n} e^{y_i - y_k}\right) \quad (2)$$

$$loss = Loss(v, -1000) \quad (3)$$

A fusion model loss function is designed, which consists of two parts as shown in Equations (2) and (3), detailing the computation method of the loss function during the training process of the multi-encoder model. Herein, y_k is denoted as the Label node of the candidate set, y_i represents the Others nodes in the candidate set, and n denotes the total number of samples. The comparison of v with the target value of -1000 is conducted, followed by the calculation of error loss. This method ensures that during backpropagation[10], the model adjusts the parameters of the fusion model's Linear layer based on the computed loss value, thereby optimizing the performance of the model.

Through the subtraction operation between Others and Label, the relative magnitudes of the two are determined. Specifically, when a value in Others exceeds that of Label, the resulting difference is positive; conversely, if it is less than Label, the difference is negative. Subsequently, the exponential function is employed to process the differences: if the result is a small negative number, it tends towards zero, while if it is positive, it significantly increases. This processing essentially weights the importance of recalling different texts. Following this, by summing all the differences, smaller values are submerged during the summation process, focusing the model's attention on data points in y_i that are greater than y_k. The objective is to highlight the impact of nodes in Others that exceed Label on the entire loss function, while the model disregards nodes smaller than Label as they already meet the requirements. After the summation of exponential functions, the resulting values may become excessively large, leading to the phenomenon of gradient explosion. To address this, a logarithmic function (Log) is applied to scale the summation result. This step not only ensures numerical stability but also aids in optimizing the convergence speed and performance of the algorithm.

In the design of the loss function, the consideration of setting the reference value to -1000 is as follows: ideally, when the value of y_k is relatively large, $y_i - y_k$ is negative, and $e^{y_i - y_k}$ approaches zero. Subsequently, the logarithmic transformation yields a significantly negative value, thus setting it to -1000. For the dataset under consideration in this paper, this already represents a considerably large negative value, which facilitates the model's convergence to the global optimum more rapidly during the training process.

IV. NUMERICAL EXPERIMENT

This paper experimentally evaluates the effectiveness of the proposed multi-encoder fusion RAG model in terms of recall performance, utilizing the BGE, M3E, and BCE encoding models.

$$\text{Rank} = \frac{1}{n}\sum_{i=1}^{n} rank_i \tag{4}$$

$$\text{MRR} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{rank_i} \tag{5}$$

This paper introduces $rank_i$ to denote the position of the target node among the recalled candidate nodes. Equation (4) defines the calculation method of the average ranking Rank, which measures the average position of all target nodes. A lower numerical value of this metric indicates that the recalled target nodes are ranked higher overall among the candidate nodes.

To comprehensively assess the recall effectiveness, the commonly used *MRR* defined by Equation (5) is introduced. This metric evaluates the accuracy of the system's queries by averaging the reciprocal ranks. The *MRR* value ranges from 0 to 1, with higher values indicating better recall performance. Specifically, for each query, the *MRR* is obtained by calculating the reciprocal rank, and then averaging the *MRR* values of all queries to derive the final *MRR* metric.

Table 1 Evaluation of a Single Encoder Retrieval Model

	Rank	MRR
BGE	22.04	0.59
M3E	30.54	0.50
BCE	20.17	0.57

As shown in table 1, this evaluation focuses on the performance of a single encoder in retrieval tasks. From the data presented in the table, it is evident that both the BGE and BCE models exhibit superior recall rates, whereas the M3E model performs slightly less favorably. The ranking of the BCE model surpasses that of the BGE in terms of Rank, while the BGE outperforms the BCE in *MRR*. However, the performance of the M3E model is inferior to both the BCE and BGE models in both metrics, indicating its relatively weaker performance.

Table 2 displays the performance evaluation results of three different encoder models in the fusion experiments. These models, namely M3E, BGE, and BCE, underwent pairwise fusion experiments, with corresponding evaluation metrics recorded.

Table 2 Evaluation of Multi-Encoder Retrieval Models

Model1	Model2	W1/W2	Rank	MRR
M3E	BGE	0.55	18.66	0.60
BCE	BGE	0.53	15.43	0.65
M3E	BCE	0.64	18.39	0.58

As shown in TABLE II, the performance evaluation of multi-encoder fusion techniques was conducted for the RAG text recall task. Here, *Model1* and *Model2* represent two different encoding models utilized in the fusion process. During fusion, the parameters *W1/W2* reflect the relative weight ratio between Model 1 and Model2. Specifically, when W1/W2 is less than 1, it indicates that Model2 obtains a higher weight in the fusion process; conversely, when *W1/W2* is greater than 1, Model 1 receives a higher weight. By comparing the candidate nodes recalled after fusing different encoding models and reordering them, the ranking results shown in the Rank column were obtained.

In TABLE 1, it is observed that the performance of the M3E model is noticeably lower than that of the BCE and BGE models. In TABLE II, when the M3E model is fused with either BGE or BCE, the performance of the fused model surpasses that of the individual BCE and BGE models, demonstrating the positive effect of fusion strategies.

As shown in TABLE 2, in the pairwise fusion experiments conducted among the three encoding models under investigation, the originally excellent-performing models, BCE and BGE, demonstrated even more outstanding performance after fusion. When these two models were fused, a significant improvement was achieved, characterized by an average ranking increase from 20.17 to 15.43, a rise of 4.74; and an increase in *MRR* from 0.59 to 0.65, representing a 6% improvement.

As depicted in Figure 2, it illustrates a schematic diagram of multi-encoder fusion, where the horizontal axis represents the w1/w2 ratio, and the vertical axis represents the *MRR* value of the fused model. The fusion effects of the three encoding models are depicted in the aforementioned figure. Specifically, the left panel of Figure 2 shows the fusion of M3E and BGE, the middle panel illustrates the fusion of M3E and BCE, and the right panel depicts the fusion of BCE and BGE.

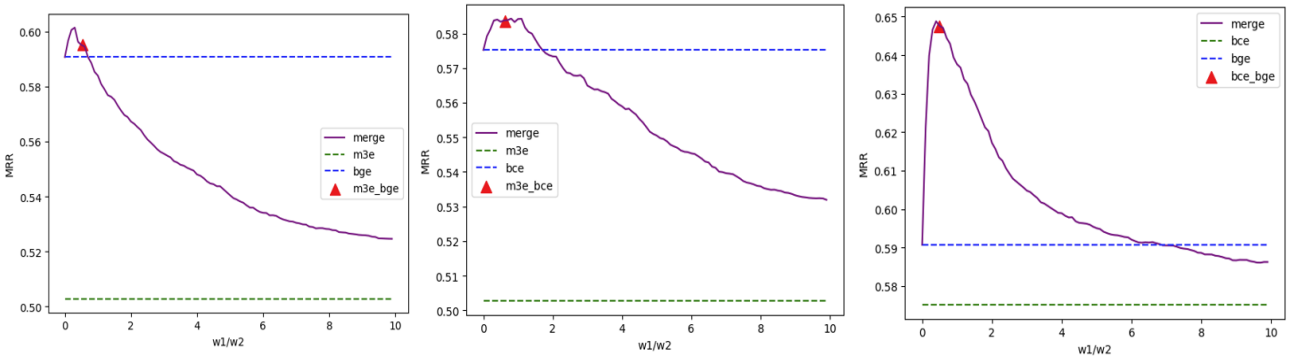

Figure 2. Visualization of Multi-Encoder Fusion Results

The green horizontal dashed line represents the *MRR* value of encoding model 1, while the blue horizontal dashed line corresponds to the *MRR* value of encoding model 2. When w1/w2 is 0, it indicates that the weight of encoding model 1 is 0, and only encoding model 2 is utilized. Therefore, the purple curve intersects with the blue dashed line at the point of 0. The precision of w1/w2 for the purple merge curve is set at 0.1, and the corresponding *MRR* values are calculated through exhaustive search to validate the effectiveness of the proposed multi-encoder fusion approach. As *W1/W2* increases, the *MRR* first increases to a peak and then begins to decline. Hence, the objective of the proposed multi-encoder fusion is to identify the w1/w2 value that corresponds to the peak *MRR*.

Taking the example of the fusion between BCE and BGE, which exhibits the best performance as shown in the right panel of Figure 2, the fusion process is elaborated in detail. The green horizontal dashed line represents the *MRR* value of 0.57 when only the BCE model is used, while the blue horizontal dashed line represents the *MRR* value of 0.59 when only the BGE model is used. Here, $w1/w2$ denotes the ratio of fusion weights between the BCE and BGE models during the fusion process. The BCE_BGE point represents the optimal fusion point predicted by the fusion model, which is located near the peak of the *MRR*, indicating superior predictive performance of the fusion model. The objective of the multi-encoder fusion model designed in this paper is to find the $w1/w2$ value when the MRR reaches its peak. To achieve this goal, we employ the loss functions defined in Equations (2) and (3), train neural networks on the dataset, and then predict the optimal $w1/w2$ value through the model. As observed in Figure 2, the fusion model points represented by triangles are all near the peak, validating the effectiveness of the multi-encoder fusion strategy and loss function design proposed in this paper.

V. CONCLUSIONS

The effectiveness of RAG generation primarily depends on the text retrieved by the retriever. In order to improve the accuracy of text retrieval by the retriever, this paper proposes to fuse multiple encoders to enhance the accuracy of retriever recall. A multi-encoder fusion model and its corresponding training loss calculation method are designed in this paper. Based on the RAG dataset constructed using Qwen-14B-Chat, the effectiveness of the proposed multi-encoder fusion is validated. The fusion model is first trained on the dataset and then evaluated. Experimental results demonstrate that the fused model can improve the accuracy of the retriever, confirming the effectiveness of the proposed multi-encoder fusion theory.

ACKNOWLEDGEMENT

Supported by Wuhan East Lake High-Tech Development Zone (also known as the Optics Valley of China, or OVC) National Comprehensive Experimental Base for Governance of Intelligent Society.

REFERENCES

[1] S. Yang, L. Li, S. Wang, W. Zhang, Q. Huang and Q. Tian, "SkeletonNet: A Hybrid Network With a Skeleton-Embedding Process for Multi-View Image Representation Learning," in IEEE Transactions on Multimedia, vol. 21, no. 11, pp. 2916-2929, Nov. 2019, doi: 10.1109/TMM.2019.2912735.

[2] J. Lilleberg, Y. Zhu and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), Beijing, China, 2015, pp. 136-140, doi: 10.1109/ICCI-CC.2015.7259377.

[3] Y. Jun and L. Peilin, "A New Method of Group Information Recommendation Based on the User Dynamic Profile Information Optimization," 2021 7th International Conference on Information Management (ICIM), London, United Kingdom, 2021, pp. 57-61, doi: 10.1109/ICIM52229.2021.9417144.

[4] Zhao Penghao, Zhang Hailin, Yu Qinhan, et al. Retrieval-Augmented Generation for AI-Generated Content: A Survey. arXiv.2024.

[5] Hanzhuo Tan, Qi Luo, Ling Jiang, et al, (2024). "Prompt-based Code Completion via Multi-Retrieval Augmented Generation," arXiv preprint, arXiv:2405.07530.

[6] NetEase Youdao, Inc. (2023) BCEmbedding: Bilingual and Crosslingual Embedding for RAG. https://github.com/netease-youdao/BCEmbedding.

[7] Jianlv Chen, Shitao Xiao, Peitian Zhang, et al. BGE.M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv. 2024.

[8] Wang Yuxin, Sun Qingxuan, He sicheng, (2023). M3E: Moka Massive Mixed Embedding Model. https://huggingface.co/moka-ai/m3e-base.

[9] Jinze Bai, Shuai Bai, Yunfei Chu, et al. Qwen Technical Report. arXiv.2023.

[10] Rumelhart, D.E., Hinton, G.E., Williams, R.J, (1986). Learning representations by back-propagating errors. Nature, 323: 533-536. http://doi.org/10.1038/323533a0.