MM-GRADE: A Multi-Modal EDA Tool Documentation QA Framework Leveraging Retrieval Augmented Generation

Yuan Pu[1,2*], Zhuolun He[1,2] Shutong Lin[1] Jiajun Qin[1] Xinyun Zhang[1] Hairuo Han[1] Haisheng Zheng[2]
Yuqi Jiang[3] Cheng Zhuo[3] Qi Sun[3] David Pan[4] Bei Yu[1]
[1]The Chinese University of Hong Kong [2]ChatEDA Tech [3]Zhejiang University [4] University of Texas at Austin

***Abstract*—The complexity of EDA tools necessitates the development of advanced documentation query answering systems to enhance user efficiency and reduce the associated learning curve. Recent innovations in the use of Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG) for EDA tool documentation have demonstrated significant progress; however, these approaches typically lack the multi-modal capabilities required to effectively handle visual data, such as circuit layout and GUI screenshots provided through user input. To address the concern, we introduce a multi-modal RAG system that incorporates two domain-customized modules: a multi-modal retriever model finetuned by the customized bilevel hard negative mining (BHNM) strategy, and a vision large language model (VLLM) finetuned using a tailored extract-score-answer pipeline. Moreover, we have manually curated ORD-MMBench, a multi-modal QA benchmark comprising 120 high-quality question-document-answer triplets based on OpenROAD documentation. Experimental results demonstrate that our customized RAG framework outperforms state-of-the-art multi-modal RAG flows and models on ORD-MMBench.**

I. Introduction

Electronic Design Automation (EDA) encompasses software tools essential for designing, analyzing, and verifying electronic systems. Both commercial and academic tools, such as OpenROAD [1] and iEDA [2], offer complex functionalities. These tools are accompanied by detailed documentation to assist users, and vendors often provide document-oriented question-answering (QA) systems, which may be either automated or human-involved. In the realm of EDA tool documentation QA, text-only queries frequently fall short in conveying the complexities of user issues. Consequently, end users typically supplement their queries with visual elements, comprising both textual descriptions and relevant images such as software GUI screenshots, design layouts or error tracebacks.

Visual information plays a crucial role in EDA tool documentation QA from two perspectives: First, certain information of user queries, such as the positional details of design layouts, congestion heatmaps, design rule violation check (DRC) specifics, or power distribution network (PDN) patterns, are difficult to convey clearly and efficiently through text alone, whereas images provide an intuitive representation of such information. Second, while some visual elements, like error traceback logs or critical path details in timing analysis, can be described with text, using images simplifies the process for users by reducing the effort required to organize textual information, thereby enhancing design efficiency and user experience. Fig. 1 shows an example of multi-modal question and its answering process under the scenario of OpenROAD: The user query is composed of two parts, namely, the text description and the image. The text description asks how to avoid to place pins on a specific region of the layout, while the image, which is the screenshot of the layout, provides extra and essential information that the region is the bottom boundary of the layout (highlighted by the green rectangle in the screenshot). Combining the multi-modal information of the user query and the retrieved document about the command `place_pins`, it is sufficient

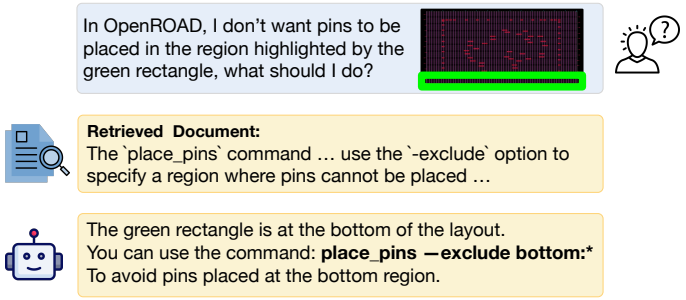


Fig. 1 An example of multi-modal QA for OpenROAD.

to infer that the solution is to use the "-exclude bottom:*" option of the `place_pins` command.

Recently, with the rapid development of large language models (LLM) and many previous work focusing on LLM for EDA in the sub-fields of HDL generation [3]–[12], script generation [13], [14], verification [15]–[21] and documentation QA [22]–[25], retrieval augmented generation (RAG) is raised up as the trend for EDA tool documentation QA. For example, RAG-EDA [22], OpenROAD-Assistant [23] and ORAssistant [24] build the RAG-based automatic QA system for OpenROAD, enhancing chat LLMs by incorporating retrieved relevant documents into the answer generation process for user queries, while EDA Corpus [25] provides a OpenROAD-QA evaluation benchmark consisting of 196 QA pairs. Despite these advancements, one major limitation of these previous work is that they can only deal with text-only user queries, neglecting the essential information conveyed through the query-related images, thereby leading to limited application scenarios and inferior QA quality. While extensive research has been conducted on multi-modal retrieval augmented generation (MM-RAG) [26]–[32], its application to multi-modal EDA tool documentation QA faces two challenges: (1) The retrieval stage of previous MM-RAG approaches primarily cater to images of natural scenes or graphic elements like charts and tables. However, under the scenarios of multi-modal EDA tool documentation QA, the images usually deliver critical information about the design layout, GUI components (such as buttons and options) and error traceback, which are too complex for existing multi-modal retrievers to handle. (2) For the answer generation stage, the VLM models of existing MM-RAG work lack EDA domain knowledge and fail to extract essential and EDA-specific information from the images, leading to spurious understanding of the user queries and thus poor performance in answer generation.

To overcome the above challenges and to facilitate the development of robust domain-specific multi-modal QA systems, we propose **MM-GRADE**, a retrieval augment generation framework customized for the task of multi-modal EDA tool documentation question-answering. MM-GRADE consists of two customized modules, namely, the retriever and the generator. On the one hand, the multi-modal retriever model projects the query and the documents into an embedding

* This work was performed during the first author's visit to UT Austin.

space for relevant document retrieval. One particular scenario for multi-modal EDA tool documentation QA is that based on the same screenshot image, queries targeting at different perspectives (and thus should be answered referring to different documents) can be raised. Unfortunately, finetuned under the so-called document-level hard negative mining strategy [33], existing multi-modal retriever models usually encode these essentially different queries at close proximity in the embedding space, leading to poor document retrieval performance. To effectively project different queries from identical images distinctly separated in the embedding space for retrieval accuracy improvement, we customize a bilevel hard negative mining (BHNM) strategy for the contrastive learning scheme of the retriever model. On the other hand, we adopt the ViT-MLP-LLM architecture for the generator and finetune the model by the customized extract-score-answer pipeline. Following the pipeline, the generator extracts EDA-related information from the image and scores the relevance degree between the query and each document before answering the query.

Moreover, to evaluate the effectiveness of MM-GRADE, we refer to the documentation of OpenROAD and design ORD-MMBench, one multi-modal EDA-tool documentation QA benchmark which consists of 120 high-quality query-document-answer triplets and is open-source at https://github.com/lesliepy99/MM-GRADE-Benchmarks-ICCAD.

The major contributions of this paper are listed as follows:

- We design and release ORD-MMBench, a multi-modal EDA tool documentation QA evaluation benchmark consisting of 120 high-quality and real-scenario question-document-answer triplets.
- We analyze the limitations of applying existing MM-RAG approaches on multi-modal EDA tool documentation QA, and propose a customized RAG flow, MM-GRADE, to solve these limitations. Experimental results demonstrate that MM-GRADE outperforms SOTA MM-RAG flows on ORD-MMBench.
- For the contrastive learning scheme of the multi-modal retriever finetuning, we customize a bilevel hard negative mining (BHNM) strategy to accommodate with the particular scenario in EDA tool documentation QA.
- For the finetuning and inference of the vision-language model (VLM) generator, we design an extract-score-answer pipeline for answering the multi-modal EDA-tool questions.

The remainder of this paper is structured as follows: In Section II, we provide an introduction to the concept of Retrieval-Augmented Generation (RAG) and the relevant performance evaluation metrics. Section III details the overall workflow and specific techniques of the proposed multi-modal EDA tool QA framework. Subsequently, Section IV presents the evaluation benchmarks and experimental results. Finally, Section V concludes the paper.

II. Preliminaries

A. Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) [34] is a framework designed to enhance content generation by AI models through the incorporation of factual information retrieved from external data sources. This approach is particularly effective in knowledge-intensive applications, such as question-answering (QA), where maintaining factual accuracy and reliability in the generated output is crucial. A typical RAG system is composed of two main components: the retriever and the generator. When presented with a user query, the retriever module searches external databases or knowledge repositories to identify and extract relevant documents. These retrieved documents are then combined with the original query and passed as input to the generator, which is often a large language model (e.g., a conversational AI model). The generator processes this augmented input to produce an informed and contextually accurate response.

B. Performance Measurement

To evaluate the effectiveness of the retriever model in terms of its ability to retrieve relevant documents, we use recall@k as the metric, which is defined as the proportion of relevant documents that are retrieved among the top k results returned by the retriever. Let R be the set of all relevant documents in the corpus, and let R_k be the set of relevant documents that appear in the top k results returned by the retriever. Then, recall@k is formulated as:

$$\text{recall@}k = \frac{|R_k|}{|R|}. \tag{1}$$

To rigorously assess the efficacy of the generator (a chat VLLM) in answering the complex EDA-tool multi-modal queries, we implement an LLM-scoring evaluation strategy (referred to as LLM-Score). For each query in ORD-MMBench, we manually derive a scoring criteria based on its ground-truth answer, and the scoring criteria is open-source. During the assessment of a query, we prompt GPT-4o to rigorously evaluate the generated answer referring to the corresponding scoring criteria and give the LLM-Score, and the score point ranges from 0 to 100, with higher scores indicating superior answers. Additionally, to enhance objective and automated evaluation, we incorporate the metrics of BLEU [35] and ROUGE-L [36] to further evaluate the performance of answer generation.

Bilingual Evaluation Understudy (BLEU), originally developed for machine translation evaluation, is now widely used for assessing text generation quality. BLEU calculates the precision of n-grams in the candidate text by measuring the proportion that overlaps with the reference text, while applying a brevity penalty (BP) to penalize overly short outputs. The metric is defined as:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right), \tag{2}$$

where BP adjusts for short translations, w_n are weights (commonly uniform) for each n-gram precision, and N is the maximum n-gram order, often set to 4.

Meanwhile, ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation - Longest Common Subsequence) evaluates the similarity between a generated answer (X) and a reference answer (Y) by analyzing their longest common subsequence (LCS). It computes recall and precision based on the LCS and combines them into an F-measure, defined as:

$$\begin{aligned} \text{recall}_{\text{LCS}} &= \frac{|\text{LCS}(X,Y)|}{|Y|}, \\ \text{precision}_{\text{LCS}} &= \frac{|\text{LCS}(X,Y)|}{|X|}, \\ \text{ROUGE-L} &= \frac{(1+\beta^2)\cdot(\text{precision}_{\text{LCS}}\cdot\text{recall}_{\text{LCS}})}{(\beta^2\cdot\text{precision}_{\text{LCS}})+\text{recall}_{\text{LCS}}}, \end{aligned} \tag{3}$$

where β is typically set to 1.

III. Algorithm

A. Overall Flow

Fig. 2 illustrates the overall flow of MM-GRADE, our proposed RAG flow customized for the task of multi-modal EDA tool documentation QA. Initially in the data preparation stage, the entire EDA tool documentation is segmented into reasonably-sized chunks and stored in a vector database. In response to a multi-modal user query, MM-GRADE operates in two stages: In the first document retrieval stage, the user query (contains the text description and the image) and document chunks are encoded by the customized multi-modal retriever

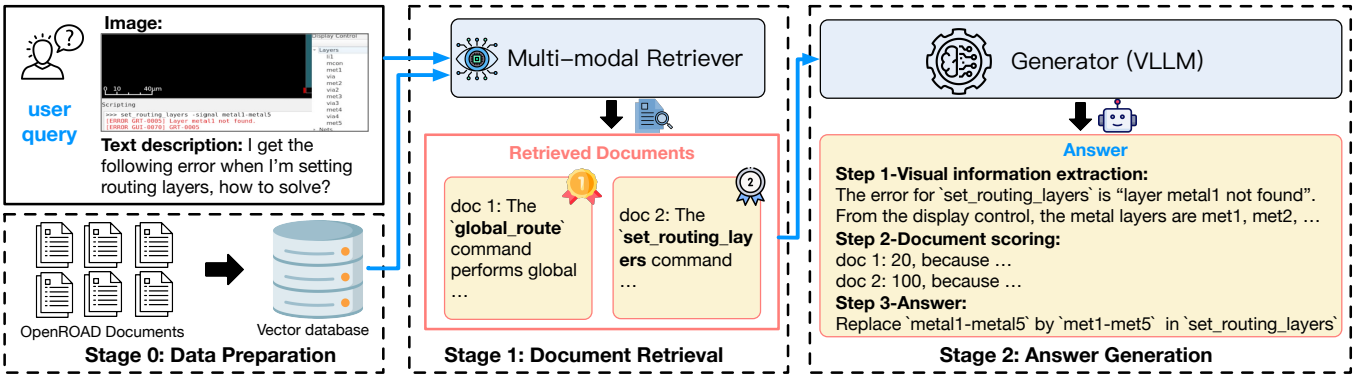


Fig. 2 The overall flow of MM-GRADE.

into a high-dimensional embedding space. Top-k document chunks which have the most similar embeddings to the query are retrieved (in Fig. 2, k is set to 2). In the second answer generation stage, the user query and the retrieved documents are concatenated and fed to our fine-tuned vision large language model (VLLM) generator. The generator answers the user query following the 3-step extract-score-answer pipeline. The visual information extraction step first extracts extra information from the image which is useful to understand and answer the user query. In this example, we obtain two items of key information from the image: the first information is the error is about the "layer metal1 not found" error for the 'set_routing_layers' command, which is helpful in understanding the exact error of this query; the second information is the metal layers in this design is named as "met1", "met2", etc, which contains the answer to this query. Then, the document scoring step analyzes the relevance relationship between the query and each retrieved document. Each document will be given a relevance score ranging from 0 to 100. The scoring criteria is listed in TABLE I. Finally, combining the generated content in the previous two steps, the generator will output the ultimate answer to the query. In the following subsections, we will introduce the motivation, model architecture and finetuning strategies of the two modules in MM-GRADE, namely, the multi-modal retriever and the VLLM generator.

B. MM-GRADE Retriever: Model Architecture

During the document retrieval phase of the multi-modal EDA tool documentation QA, a multi-modal retriever model encodes user queries (composed of textual description and image) and text-only document chunks into high-dimensional vectors. These vectors, which represent domain-specific semantic and visual features, are organized within an embedding space where vectors of relevant queries and documents are closely positioned. For relevant document retrieval, the retriever model selects the top-k closest document chunks to the query based on their proximity in this space. To guarantee high retrieval accuracy under the scenario of multi-modal EDA tool documentation QA, two important characteristics should be possessed by the multi-modal retriever model: (1) Possession of EDA-tool-specific knowledge to effectively discern and extract critical information from both queries and documents. (2) Capability to handle multi-modal information retrieval, specifically image&text-to-text retrieval as required under our scenario. Unfortunately, previous research work [28], [33], [37]–[40] on multi-modal information retrieval does not possess these two characteristics simultaneously. On the one hand, they focus on general scenarios of multi-modal information retrieval and mainly deal with natural objects, demonstrating poor performance in manipulating EDA-tool-related images, which contain sophisticated information of

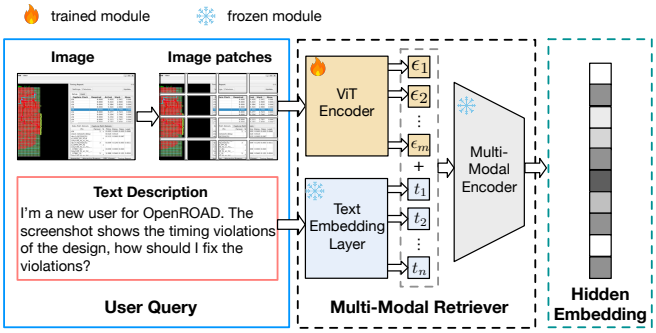


Fig. 3 The model architecture of our multi-modal retriever.

the circuit layout, EDA tool GUI and error traceback. On the other hand, many of the previous works are mainly designed to deal with cross-modal information retrieval (such as text-to-image retrieval or image-to-text retrieval), and is ineffective for the task of multi-modal retrieval.

To overcome the above two limitations, we adopt the model architecture of VISTA [33] as our multi-modal retriever model. As shown in Fig. 3, the model consists of three modules, namely, the ViT encoder, the text embedding layer and the multi-modal encoder. Given one user query containing the image q^{img} and the text description q^{txt}, for the processing of q^{img}, the image is first segmented into m image patches $P = \{p_1, p_2, \cdots, p_m\}$. Then, the ViT encoder projects P into the image embeddings as follows:

$$\{\epsilon_1, \epsilon_2, \cdots, \epsilon_m\} = \texttt{ViT}(\{p_1, p_2, \cdots, p_m\}). \tag{4}$$

Meanwhile for the text description $q^{\text{txt}} = \{q_1^{\text{txt}}, q_2^{\text{txt}}, \cdots, q_n^{\text{txt}}\}$, the text embedding layer TE, substantially the embedding layer of a BERT model, maps q^{txt} into the text embeddings as follows:

$$\{t_1, t_2, \ldots, t_n\} = \texttt{TE}(\{q_1^{\text{txt}}, q_2^{\text{txt}}, \cdots, q_n^{\text{txt}}\}). \tag{5}$$

Finally, the image embeddings and the text embeddings, sharing the same dimensionality, are concatenated and encoded by the multi-modal encoder MME to obtain the embedding representation e^q of the user query or document:

$$e^q = \texttt{MME}(\{\epsilon_1, \epsilon_2, \ldots, \epsilon_m\} \| \{t_1, t_2, \ldots, t_n\}), \tag{6}$$

where $\|$ denotes concatenation.

In the context of multi-modal EDA tool documentation question answering (QA), where user queries consist of both images and text but the document chunks are exclusively text-based, the retriever model is designed to process only textual content for generating document embeddings. This imposes a critical requirement for the retriever to effectively extract and represent semantic information

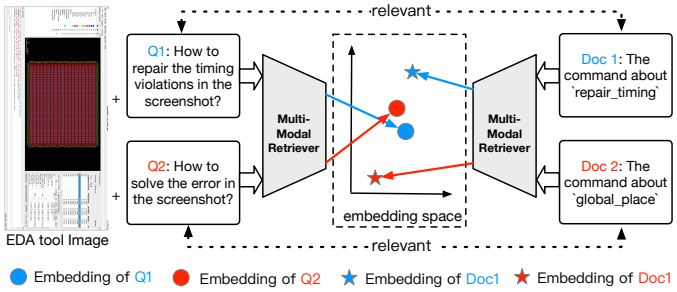


Fig. 4 A failure case by the existing multi-modal retriever models in encoding EDA-tool-related queries and documents.

from text-only inputs. To address this requirement, our retriever model architecture leverages components from a pre-trained text embedding model. Specifically, the embedding layer of the text embedding model is adopted as the text embedding layer, while its encoder layer is utilized as the multi-modal encoder. During training, we focus solely on fine-tuning the Vision Transformer (ViT) encoder to align the visual representations with the pre-existing textual modules. The text embedding components remain unchanged to preserve their inherent ability to capture semantic nuances. This design ensures that the retriever retains the robust semantic extraction capabilities of the text embedding model while seamlessly integrating visual alignment. As a result, the multi-modal retriever achieves enhanced retrieval accuracy, optimizing performance in the multi-modal EDA tool documentation QA scenario.

C. MM-GRADE Retriever: Training Strategy

Ideally, in the high-dimensional embedding space constructed by the multi-modal retriever model, the embeddings of the user query and the relevant documents are at close proximity; meanwhile, the embeddings of the user query and the irrelevant documents will be separated from each other. To achieve the above goal, existing multi-modal retriever models such as VISTA [33] usually adopt the document-level hard negative mining (HNM) strategy for contrastive learning. Given one user query consisting of the image q_i^{img} and the text description q_i^{txt}, its relevant document d_i^+ is regarded as the positive sample, while its n irrelevant documents $d_i^- = \{d_{i,1}^-, d_{i,2}^-, ..., d_{i,n}^-\}$ are mined as hard negative samples, and the contrastive learning loss function by the document-level HNM strategy can be written as:

$$L_d^i = -\log \frac{e^{f(q_i^{\text{img}} \| q_i^{\text{txt}})^\top f(d_i^+)/\tau}}{e^{f(q_i^{\text{img}} \| q_i^{\text{txt}})^\top f(d_i^+)/\tau} + \sum_{j=1}^{n} e^{f(q_i^{\text{img}} \| q_i^{\text{txt}})^\top f(d_{i,j}^-)/\tau}}, \quad (7)$$

where $f(\cdot)$ denotes the generated embedding by the multi-modal retriever model, and τ denotes the temperature.

In practice, a single screenshot from an EDA tool often contains complex and diverse information, capable of triggering multiple distinct queries. For example, as shown in Fig. 4, one part of the screenshot highlights setup timing violations, prompting a query ($Q1$) about resolving these violations using the `repair_timing` command, which corresponds to a relevant document ($Doc1$). Simultaneously, the same screenshot contains an error message stating "*Instance xxx is not placed*," which generates a different query ($Q2$) on how to address this issue, solvable via the `global_place` command, referred to another document ($Doc2$). Unfortunately, directly applying the existing multi-modal retriever models such as VISTA [33] to the above scenario may lead to low accuracy for document retrieval. The reason is that for existing multi-modal retriever models, the visual information plays a pivotal role for the embedding generation, causing queries about different aspects of the same image, such as $Q1$ and $Q2$ in Fig. 4, to be situated closely within the embedding space, despite their different textual descriptions. Conversely, the documents relevant to these queries, which pertain to separate documentation aspects, are positioned distantly in this space. Consequentially, $Q1$ and $Q2$ can not be simultaneously projected to the close proximity of their corresponding relevant documents $Doc1$ and $Doc2$, leading to inevitably lower document retrieval accuracy, as illustrated by Fig. 4.

To overcome the above limitation, we propose the query-level hard negative mining (HNM) strategy to augment the contrastive learning for multi-modal retriever finetuning. Based on one EDA-tool-related screenshot image q_i^{img}, m different user queries with corresponding text descriptions $\{q_{i,1}^{\text{txt}}, q_{i,2}^{\text{txt}}, ..., q_{i,m}^{\text{txt}}\}$, can be asked. These m queries correspond to different relevant documents, denoted as $\{d_{i,1}^+, d_{i,2}^+, ..., d_{i,m}^+\}$. The fine-tuning objective is to spatially separate the embeddings of different queries associated with the same image q_i^{img}, while ensuring that the embeddings of each query $\{q_i^{\text{img}}, q_{i,j}^{\text{txt}}\}$ ($j \in [1, m]$) and its corresponding relevant document $d_{i,j}^+$ are closely positioned in the embedding space. Under the proposed query-level hard negative mining (HNM) strategy, for any $j \in [1, m]$ and the query $\{q_i^{\text{img}}, q_{i,j}^{\text{txt}}\}$, we treat its relevant document $d_{i,j}^+$ as the positive sample, and the remaining $m-1$ queries with different text descriptions (i.e., $\{(q_i^{\text{img}}, q_{i,k}^{\text{txt}}) | k \in [1, m] \wedge k \neq j\}$) are mined as the hard negative samples. The contrastive learning objective function augmented by the query-level HNM strategy, denoted as L_q^i, can be written as:

$$L_q^i = -\sum_{j=1}^{m} \log \frac{e^{f(q_i^{\text{img}} \| q_{i,j}^{\text{txt}})^T f(d_{i,j}^+)/\tau}}{e^{f(q_i^{\text{img}} \| q_{i,j}^{\text{txt}})^T f(d_{i,j}^+)/\tau} + \sum_{k \neq j} e^{f(q_i^{\text{img}} \| q_{i,j}^{\text{txt}})^T f(q_i^{\text{img}} \| q_{i,k}^{\text{txt}})/\tau}}, \quad (8)$$

where the denotations of $f(.)$ and τ are the same as Equation (7).

Finally, we integrate the document-level HNM strategy in Equation (7) and the query-level HNM strategy in Equation (8) to form the bilevel hard negative mining (BHNM) strategy. Given one dataset $\{q_i^{\text{img}}, \{q_{i,1}^{\text{txt}}, q_{i,2}^{\text{txt}}, ..., q_{i,m}^{\text{txt}}\}, \{d_{i,1}^+, d_{i,2}^+, ..., d_{i,m}^+\}\}^N$ of size N, the BHNM-augmented contrastive learning loss function L can be written as:

$$L = \sum_{i=1}^{N} L_d^i + L_q^i. \quad (9)$$

On the one hand, the contrastive learning objective function L_d^i under the document-level HNM strategy trains the multi-modal retriever model to project queries and relevant documents in close proximity in the embedding space, improving the relevant document retrieval accuracy. On the other hand, the query-level-HNM-augmented contrastive learning objective function L_q^i separates the embeddings of different queries with the same images apart from each other and closer to their corresponding relevant documents, facilitating the capacity of the retriever model in handling the customized document retrieval scenario of multi-modal EDA tool documentation QA.

D. MM-GRADE Generator

During the multi-modal EDA tool documentation QA process, after retrieving relevant documents, both the documents and the user query (including the screenshot and text) are input into a generator, typically a Vision Large Language Model (VLLM), for answer generation. Due to the abundant information contained in the EDA-tool screenshot images and the sophisticated nature of EDA-tool-related queries, the VLLM generator should be able to deal with the domain-specific inference logic involved in multi-modal EDA tool documentation QA, including determining the relevance between the query and each

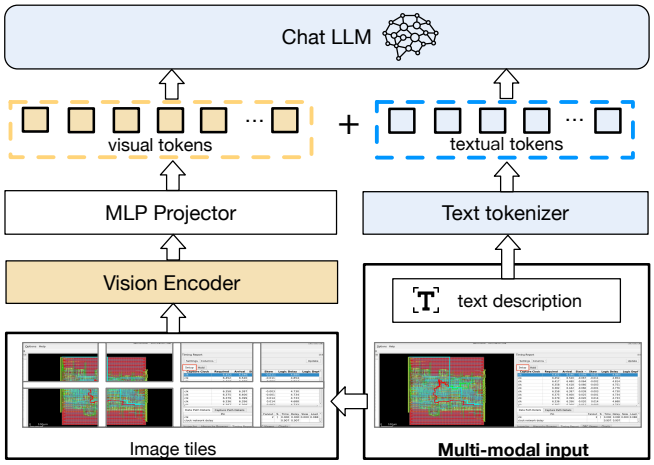


Fig. 5 Model architecture of InternVL2, the VLLM we adopt as the generator for multi-modal EDA tool documentation QA.

retrieved document, extracting useful information from the image for answer generation, etc. To satisfy this requirement, we adopt InternVL2 [41], [42] as our generator model architecture, and propose a domain-customized extract-score-answer pipeline for model fine-tuning and inference.

Model Architecture. We utilize the InternVL2 [41], [42] model series as the backbone for our generator architecture. In InternVL2, the vision encoder has an extensive number of parameters, comparable to those of the language model, enabling the Vision-Language Large Model (VLLM) to effectively process and interpret the complex visual information present in EDA tool screenshots. As illustrated in Fig. 5, the architecture of our VLLM begins by dividing the input image into uniform tiles. These tiles are passed through the vision encoder, which extracts visual embeddings. These embeddings are then projected into visual tokens through a multi-layer perceptron (MLP), aligning them with the language modality. Meanwhile, textual inputs are tokenized into textual tokens. The visual tokens and textual tokens are subsequently concatenated and fed into a chat large language model (LLM) to generate the output. By leveraging a highly capable vision encoder, our VLLM generator is well-equipped to efficiently handle and extract the rich, detailed, and intricate information embedded in EDA-tool screenshots.

Extract-Score-Answer (ESA) Pipeline.

Due to the sophisticated nature of EDA tool documentation and the abundant information contained in the EDA tool screenshots, answering the corresponding multi-modal queries requires complicated and domain-specific logical inference, which is challenging for existing chat VLLMs. Motivated by the previous work of CoT (chain of thought) [43], the complex user queries can be decomposed into a series of sub-tasks with progressive logical relationship, and a multi-round dialogue can be applied by the chat VLLM to solve the sub-tasks step by step and ultimately answer the user queries. Customized for the scenario of multi-modal EDA tool documentation QA, we propose the extract-score-answer pipeline for the chat VLLM to answer the queries. Following the pipeline, the answering process is decomposed into 3 steps. In the first step of visual information extraction, the extra information about the layout, GUI and error traceback of the EDA tool is extracted from the user screenshot image to help understand the user query. The second step is document relevance scoring. Given the user query, the essential information extracted from the screenshot image and the retrieved documents,

TABLE I Scoring criterion for the relevance degrees between one query q and one document d. This scoring criterion is used in the document relevance scoring step.

Score	Description
100	d is perfectly relevant to q and can be used to directly answer q.
80	d is relevant to q and can partially answer q.
60	d is relevant to q but can not be directly used to answer q.
40	It is possible that d is relevant with q, but more information is required to verify the relevance.
20	d is basically not relevant with q.
0	d is totally irrelevant with q.

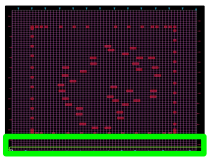
this step analyzes the logical relevance between the user query and each document chunk. Each retrieved document chunk will be given a relevance score ranging from 0 to 100, and the scoring criteria is shown in TABLE I. The final step (answer) refers to the extracted key information in the first step and the retrieved documents with high relevance scores in the second step, and answers the user query in the domain-specific perspective. The answer generation stage in Fig. 2 shows one example of applying the 3-stage extract-score-answer pipeline to solve the EDA tool query about one error encountered during setting routing layers. In the first visual information extraction step, the VLLM extracts the essential information that the error indicates "*layer metal1 not found*", while the display control window shows the names of metal layers in the design are "met1", "met2", etc. In the second step of document relevance scoring, the command "`set_routing_layers`" is evaluated as relevant with the user query, while the command "`global_route`" is regarded as irrelevant. Finally, combining the information and relevant documents obtained from the first and second step, the VLLM generator infers that replacing "metal1-metal5" by "met1-met5" in the "`set_routing_layers`" command can resolve the error.

Fig. 6 shows one example of applying the 3-step extraction-scoring-answer strategy to answer the visual EDA tool query: The user query is to avoid the pins to be placed in the green rectangle in the screenshot, and the retrieved documents are about the commands "place_pin" (used to place a specific pin) and "place_pins" (used to place all pins in the design). In the first step, namely, visual information extraction, the VLLM extracts the essential information that the green rectangle in the screenshot highlights the bottom region of the layout, which is essential for specifying the correct command option in the answer phase. In the second step of document relevance scoring, the command "place_pins" is evaluated as relevant with the user query, while the command "place_pin", which can only place one specific pin at a time, is regarded as irrelevant. Finally, combining the information and relevant documents obtained from the first and second steps, the VLLM generator infers that using the "-exclude bottom:*" option of the command "place_pins" can answer the visual user query.

The extract-score-answer pipeline is used for both the inference and finetuning phase of the VLLM generator. During inference, we adopt the 3-round dialogue manner and each dialogue round corresponds to one step in the pipeline. Meanwhile for the fietuning stage, we use the next-token prediction objective and minimize the cross entropy loss shown in Equation (10):

$$L = -\sum_{t=1}^{n} \log P(x_t | x_{<t}; \theta), \tag{10}$$

where $x = \{x_1, x_2, ..., x_n\}$ denotes the input sequence, and θ represents the parameter of the VLLM generator.



User visual query-image:

User visual query text description:
I don't want the pins to be placed on the region of the green rectangle in the screenshot, what should I do?

Retrieved Documents:
Doc 1: `place_pin`: the command to place a specific pin …

Doc 2: use the option `exclude` of `place_pins` …

Extraction-Scoring-Answer Strategy

Stage 1: Visual information extraction
Prompt: Extract the information from the image which can be used to understand and answer the question.
Answer: The pins highlighted in the green rectangle is the bottom region of the layout.

Stage 2: Document Relevance Scoring
Prompt: Now given the retrieved documents, analyse and score the relevance between each document and the visual query. The score ranges from 0 to 100, higher scores represent higher relevance.
Answer:
Doc 1: 20, because the query is about the command to place all pins, but `place_pin` command only places one pin.
Doc 2: 100, because the document introduces the usage of `place_pins` and the `exclude` flag.

Stage 3: Answer
Prompt: answer this user query according the extracted visual information in stage 1 and the document relevance scoring results in stage 2.
Answer:
you can use the command: `place_pins -hor_layers <hor layers> -ver_layers <ver layers> -exclude bottom:*` to disable pins placed on the bottom region.

Fig. 6 The domain-customized extraction-scoring-answer pipeline for model fine-tuning and inference.

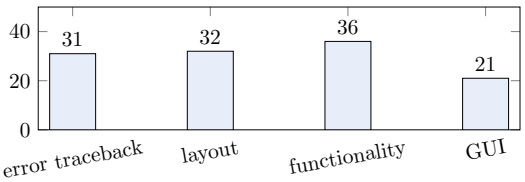


Fig. 7 The number distribution of question categories in ORD-MMBench.

IV. Experimental Results

A. Evaluation Benchmark

To enable a systematic measurement of MM-GRADE, we design and release one benchmark, namely, ORD-MMBench, for multi-modal EDA tool documentation QA. ORD-MMBench is open-source at https://github.com/lesliepy99/MM-GRADE-Benchmarks-ICCAD, which consists of 120 high-quality manually-designed question-document-answer triplets based on the documentation of OpenROAD [1]. To construct the benchmark, we initially segmented the OpenROAD documentation into 332 document chunks. Then, we run the test scripts for different modules of OpenROAD to obtain a candidate pool of tool-related screenshot images. Finally, we sampled 120 high-quality and informative images from the screenshot image pool, and for each screenshot image, we proposed one user query under the real user scenario, selected one to three the most relevant document chunks as the reference, and manually developed the corresponding answer. Among these questions, 109 of them are from one single document chunk, while 8 and 3 rely on two and three document chunks, respectively. The questions in ORD-MMBench cover almost the whole flow and all modules of OpenROAD, and due to the content of the questions, they can be classified into 4 categories, namely, *error traceback*, *layout*, *functionality* and *GUI*, and the number distribution of each question category in ORD-MMBench is visualized in Fig. 7. For the *error traceback* question, the screenshot is usually one image of the error traceback log of the tool, and the question inquires the reason of or solution to the error. The *layout* category covers inquiries about the design layout after specific VLSI flows such as floorplan, placement or routing. The *functionality* questions inquiry the usage of specific modules/commands/options in OpenROAD, based on the design information provided in the image. Finally, the *GUI* questions are mainly about the manipulation of the graphic user interface (GUI) of OpenROAD, one example of the query is "*How should I visualize the clock tree in the window?*".

B. Training Dataset Construction

We initially run the test scripts under github repositories OpenROAD[1] and OpenROAD-flow-scripts[2] to screenshot 1584 images of the design-related information, laytout, GUI components, TCL scripts, error traceback log, etc. Note that the test scripts of OpenROAD and OpenROAD-flow-scripts target at different technology nodes (ASAP7, Nangate45, Sky130, etc) and different circuit designs, guaranteeing the diversity and universality of the screenshot images obtained. For the training dataset construction for multi-modal retriever finetuning, for one screenshot image q_i^{img} belonging to the OpenROAD module m, each time we randomly sample one document chunk d_i^+ of module m, and then prompt gpt-4o to propose one corresponding user query q_i^{txt} if d_i^+ is relevant to q_i^{img}. Finally, for each $\{q_i^{\text{img}} \| q_i^{\text{txt}}, d_i^+\}$, we apply VISTA [33] to retrieve document chunks for the user query, and use gpt-4o to filter out 3 irrelevant documents $d_i^- = \{d_{i,1}^-, d_{i,2}^-, d_{i,3}^-\}$ as the negative document samples. Ultimately, we obtain 2383 $\{q_i^{\text{img}} \| q_i^{\text{txt}}, d_i^+, d_i^-\}$ corpus items for the document-level-HNM-augmented contrastive learning objective L_d in Equation (7). Among the above corpus items, we collect 1075 $\{q_i^{\text{img}}, \{q_{i,1}^{\text{txt}}, q_{i,2}^{\text{txt}}, ..., q_{i,m}^{\text{txt}}\}, \{d_{i,1}^+, d_{i,2}^+, ..., d_{i,m}^+\}\}$ corpus items with the same screenshot image q_i^{img} but different user queries, which are used for the query-level-HNM-augmented contrastive learning objective L_q in Equation (8). To collect the training corpus for VLLM generator finetuning, for each $\{q_i^{\text{img}} \| q_i^{\text{txt}}, d_i^+, d_i^-\}$, we first prompt gpt-4o to give the answer of $q_i^{\text{img}} \| q_i^{\text{txt}}$ referring to the ground truth reference document d_i^+. Then, based on the generated answer, gpt-4o is conducted to extract the essential information from the image q_i^{img} and evaluate the relevance scores between the query and each document. By the above procedure, we finally obtain 2307 query-document-answer triplets following the extract-score-answer prompt template.

C. Experimental Setting

For the fine-tuning of the multi-modal retriever model, we use bge-visualized-m3[3] as the base model. During the fine-tuning process, the textual modules are frozen, following the same configuration as VISTA [33]. The hyper-parameters are set as follows: the batch size is 8, the learning rate is 2×10^{-5}, the maximum sequence length is 512, and the temperature is set to 0.02. For negative sampling, the group sizes in Equation (7) and Equation (8) are set to 3 and 2, respectively. The model is fine-tuned for 3000 steps to achieve the desired performance. We adopt InternVL2-26B[4] as the base model for the VLLM generator. This model is a multi-modal large language model consisting of a 20B language model and a 6B vision model. The fine-tuning process for the VLLM is conducted over 2 epochs, with a batch size of 1, a learning rate of 2×10^{-5}, and a maximum

[1] https://github.com/The-OpenROAD-Project/OpenROAD
[2] https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts
[3] https://huggingface.co/BAAI/bge-visualized/blob/main/Visualized_m3.pth
[4] https://huggingface.co/OpenGVLab/InternVL2-26B

TABLE II Performance of the multi-modal RAG flows on ORD-MMBench.

RAG Flow	ORD-MMBench▶error traceback			ORD-MMBench▶layout			ORD-MMBench▶functionality			ORD-MMBench▶GUI			ORD-MMBench▶all		
	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L
VisRAG [31]	41.129	0.080	0.223	12.500	0.098	0.242	18.750	0.111	0.247	4.762	0.024	0.162	20.417	0.084	0.225
EchoSight [32]	51.613	0.087	0.241	54.688	0.165	0.300	62.500	0.191	0.322	64.286	0.091	0.214	57.917	0.139	0.276
RAG-EDA [22]	22.419	0.064	0.187	61.719	0.156	0.276	59.722	0.190	0.332	51.190	0.107	0.267	49.125	0.134	0.268
VISTA-RAG [33]	65.323	0.126	0.266	68.750	0.216	0.348	71.528	0.232	0.368	73.810	0.116	0.261	69.583	0.180	0.317
MM-GRADE w/o ESA	71.129	0.159	0.303	78.125	0.273	0.418	75.833	0.291	0.442	**90.476**	0.160	0.312	77.792	0.229	0.377
MM-GRADE (ours)	**82.258**	**0.244**	**0.391**	**80.469**	**0.307**	**0.468**	**80.000**	**0.294**	**0.450**	83.333	**0.178**	**0.323**	**81.292**	**0.264**	**0.417**

TABLE III Performance of document retrieval on ORD-MMBench.

Model type	recall@1	recall@2	recall@3	recall@4	recall@5
BGE-M3 [44]	0.590	0.694	0.731	0.761	0.776
RAG-EDA-retriever [22]	0.530	0.612	0.679	0.724	0.739
DEDR [40]	0.112	0.187	0.231	0.284	0.299
BLIP [38]	0.127	0.209	0.276	0.299	0.313
VisRAG-retriever [31]	0.025	0.042	0.050	0.083	0.108
EchoSight-retriever [32]	0.383	0.492	0.583	0.625	0.667
VISTA: bge-visualized-m3 [33]	0.582	0.716	0.739	0.761	0.799
MM-GRADE-retriever w. d-level HNM	0.664	0.784	0.806	0.828	0.851
MM-GRADE-retriever w. q-level HNM	0.582	0.694	0.746	0.761	0.828
MM-GRADE retriever (ours)	**0.679**	**0.828**	**0.843**	**0.866**	**0.896**

sequence length of 8192 to handle long-context multi-modal inputs. The multi-modal retriever is fine-tuned using 4 NVIDIA A100 GPUs, each with 40GB of memory, ensuring sufficient resources for efficient training. Meanwhile, the VLLM generator is fine-tuned on 8 NVIDIA A100 GPUs with 80GB of memory each.

D. Evaluation: Multi-Modal RAG Flow

In this subsection, we evaluate our customized MM-GRADE flow and other SOTA multi-modal RAG flows on the task of answering the queries in ORD-MMBench. During the evaluation of one MM-RAG flow, given one user query q, the retriever model is first leveraged to retrieve 4 documents D_q. Then q and D_q are concatenated and fed to the generator model, the later follows the extract-score-answer pipeline to generate the answer. We select 3 SOTA MM-RAG flows, namely, VisRAG [31], EchoSight [32] and VISTA-RAG [33], along with RAG-EDA [22], the SOTA text-only RAG flow customized for EDA tool documentation QA, as our baselines. To further evaluate the effectiveness of the extract-score-answer (ESA) pipeline, we replace the ESA pipeline in MM-GRADE with a direct-answer pipeline, where the MM-GRADE-generator is prompt to directly answer the user query referring to the retrieved documents. The MM-GRADE flow with the direct-answer pipeline is denoted as "MM-GRADE w/o ESA" in TABLE II. During the document retrieval stage, these baselines leverage their retrieval models to encode the queries and all documents into the hidden embedding space, and conduct similarity search for relevant document retrieval. For the answer generation phase, GPT-4o is employed as the generator model for the three multi-modal RAG baselines, and the same ESA pipeline as MM-GRADE is used during inference. For RAG-EDA, as its modules are specifically customized for EDA tool QA, we retain its original generator and associated prompt within the flow. For performance measurement, we adopt LLM-Score, BLEU and ROUGE-L, which are detailed in Section II-B.

The experimental results presented in TABLE II indicate that RAG-EDA exhibits relatively poor performance across all metrics due to its inability to process visual information, highlighting the critical role of visual information in EDA tool documentation QA. Furthermore, the retrieval accuracy of the VisRAG retriever on ORD-MMBench is notably low, resulting in inferior QA performance for VisRAG. By leveraging a domain-customized multi-modal retriever model, the extract-score-answer (ESA) pipeline, and a fine-tuned VLLM generator, MM-GRADE achieves superior performance compared to all SOTA baselines across all query categories in ORD-MMBench. Finally, replacing the ESA pipeline in MM-GRADE with a direct-answer pipeline results in the variant "MM-GRADE w/o ESA" achieving overall performance ranked second only to MM-GRADE, further demonstrating the effectiveness of the ESA pipeline in addressing multi-modal EDA tool QA tasks.

E. Evaluation: Multi-Modal Retriever

In this subsection, we evaluate the relevant document retrieval accuracy of our finetuned multi-modal retriever model, namely, MM-GRADE-retriever, on the 120 queries in ORD-MMBench. To begin with, we adopt the embedding model under evaluation to encode all document chunks into the hidden embedding space. Then, each user query is encoded by the same embedding model to obtain the embedding e^q, and the top-k document chunks whose embeddings are closest to e^q are retrieved. We use the metric recall@k introduced in Section II-B to measure the accuracy of the embedding model for relevant document retrieval, and set k from 1 to 5. For baseline models, we first select two text-only embedding models, namely, BGE-M3 (which is the text embedding module used in MM-GRADE-retriever and VISTA) and RAG-EDA-retriever [22]. For the multi-modal retriever models, we choose DEDR [40], BLIP [38], VisRAG-retriever [31], EchoSight-retriever [32] and bge-visualized-m3 (the model proposed by VISTA [33]) as the baselines. Among them, VisRAG-retriever and EchoSight-retriever are the retriever models of the SOTA MM-RAG flows VisRAG [31] and EchoSight [32], respectively. To demonstrate the effectiveness of our proposed bilevel hard negative mining (BHNM) strategy, we conduct the ablation studies and train our retriever model solely using either the document-level-HNM-augmented contrastive learning objective L_d in Equation (7) or the query-level-HNM-augmented contrastive learning objective L_q in Equation (8), and the trained models are denoted as "MM-GRADE-retriever w. d-level HNM" and "MM-GRADE-retriever w. q-level HNM", respectively. Experimental results in TABLE III demonstrate that our finetuned MM-GRADE-retriever outperforms all the baselines for the document retrieval on ORD-MMBench. The ablation studies demonstrate that our customized bilevel hard negative mining (HNM) strategy overcomes the domain-specific retrieval challenges mentioned in Section III-C.

To provide statistical evidence supporting the BHNM strategy, we conducted an experiment to analyze how bge-visualized-m3 (base model) and MM-GRADE-retriever (finetuned model) embed queries associated with the same image but different document chunks. In this experiment, we sampled 500 query pairs from the training dataset, each pair represented as $\{q_i^1, q_i^2, q_i^{img}\}$, where q_i^1 and q_i^2 are two distinct OpenROAD-related queries based on the same image (q_i^{img}) but proposed from different document chunks. For each pair, both models encoded the concatenated inputs $\{q_i^{img} \| q_i^1\}$ and $\{q_i^{img} \| q_i^2\}$, generating hidden embeddings e_i^1 and e_i^2. We then computed the cosine similarity between e_i^1 and e_i^2 for all 500 pairs and calculated the average similarity. Higher average cosine similarity suggests that the model relies heavily on visual information during embedding generation, resulting in embeddings of different queries with the same image being in close proximity in the hidden embedding space,

TABLE IV Performance of the VLLMs as generators on ORD-MMBench.

Model	ORD-MMBench▶error traceback			ORD-MMBench▶layout			ORD-MMBench▶functionality			ORD-MMBench▶GUI			ORD-MMBench▶all		
	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L	LLM-Score	BLEU	ROUGE-L
llava-v-1.6-34b	60.323	0.051	0.154	68.750	0.093	0.193	84.722	0.083	0.200	75.000	0.050	0.144	72.458	0.071	0.177
InternVL2-26B	69.839	0.087	0.202	76.563	0.157	0.295	84.722	0.175	0.303	72.619	0.074	0.203	76.583	0.130	0.257
InternVL2-40B	58.065	0.083	0.198	64.844	0.132	0.248	77.778	0.131	0.259	85.714	0.088	0.206	70.625	0.111	0.231
GPT-4o	74.032	0.147	0.284	81.250	0.270	0.384	**94.444**	0.290	0.415	83.333	0.141	0.287	83.708	0.222	0.351
MM-GRADE-generator	**87.097**	**0.239**	**0.381**	**88.281**	**0.330**	**0.491**	91.667	**0.302**	**0.459**	**88.095**	**0.166**	**0.323**	**88.958**	**0.270**	**0.424**

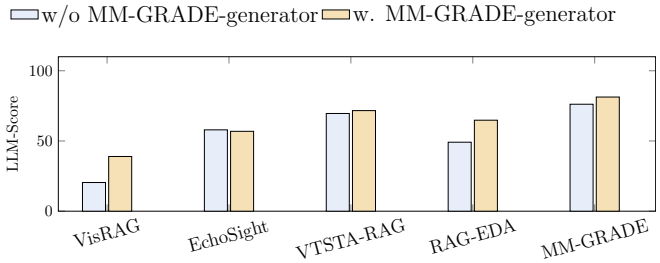


Fig. 8 Ablation study by replacing the generator models in the RAG flows under evaluation. For the "w/o MM-GRADE-generator" column, the RAG flows of VisRAG, EchoSight, VISTA-RAG and MM-GRADE are equipped with GPT-4o, while the RAG-EDA-generator is adopted in RAG-EDA.

which can degrade retrieval accuracy. The average cosine similarity is 0.789 for bge-visualized-m3 and 0.618 for MM-GRADE-retriever, indicating that the finetuned MM-GRADE-retriever better separates embeddings of different queries associated with the same image. This result supports our observations and validates the effectiveness of the BHNM strategy.

F. Evaluation: VLLM Generator

To evaluate the effectiveness of our finetuned generator for answering the EDA tool related queries, for each query q in ORD-MMBench, we combine it with its ground-truth document chunks, feed the combination to the generator under evaluation, and prompt the generator to answer the user query following the extract-score-answer pipeline. For VLLM generators, we select OpenAI gpt-4o [45] and several SOTA chat VLLMs including llava-v1.6-34b [46], InternVL2-26B and InternVL2-40B [41], [42], as baselines. For performance measurement metrics, we adopt LLM-Score, BLEU and ROUGE-L, the same metrics used for MM-RAG flow evaluation. For each generator model under evaluation, we measure the above metrics independently on the four categories of questions (*error traceback*, *layout*, *functionality* and *GUI*) in ORD-MMBench, and integrate the average results on the whole benchmark (the average result is denoted as ORD-MMBench▶all). Experimental results in TABLE IV demonstrates that compared with all selected SOTA VLLM models, our finetuned generator (MM-GRADE-generator) shows superior performance on the ORD-MMBench evaluation benchmark for almost all categories of queries.

G. Ablation Study & Discussion

To further highlight the advantages of our fine-tuned multi-modal generator (MM-GRADE-generator) in the context of EDA tool QA, we conduct an ablation study. In this experiment, we replace the generator model (either GPT-4o or the RAG-EDA generator) in all RAG baselines (VisRAG, EchoSight, VISTA-RAG, and RAG-EDA) with our fine-tuned MM-GRADE generator. Simultaneously, we replace the generator in MM-GRADE with GPT-4o. The reconfigured RAG flows are then evaluated on ORD-MMBench. As shown in Fig. 8, the results indicate that substituting the generator in EchoSight and VISTA-RAG has negligible effect on their LLM-Score. However, when integrated into VisRAG, RAG-EDA, and MM-GRADE, the MM-GRADE-generator yields significant performance improvements, outperforming GPT-4o by 18.5%, 15.7%, and 5.1% in LLM-Score, respectively.

We now examine the universality and transferability of the proposed MM-GRADE flow and its underlying methodologies, which are designed to meet the diverse and evolving needs of circuit design automation. As described in Section IV-B, the training dataset is constructed by generating source images through the execution of a carefully designed set of module-test scripts within the OpenROAD framework. These scripts cover a wide range of EDA processes, incorporating circuit designs of varying complexities, functionalities, and configurations across multiple technology nodes. This approach ensures the training corpus is both diverse and representative of modern circuit design workflows, capturing the variability required for robust generalization. By training on this heterogeneous dataset, MM-GRADE learns to address queries related to circuit designs and technology nodes of different types and scales, demonstrating strong generalization capabilities and adaptability to unseen scenarios.

Transferability is a critical and necessary characteristic of an EDA tool documentation QA flow. Intuitively, verifying the transferability of MM-GRADE would involve evaluating its performance on other graphical EDA tools. However, to the best of our knowledge, OpenROAD is the only open-source EDA tool that offers comprehensive documentation and functionalities, as well as a flow and GUI that closely resemble those of commercial EDA tools. Furthermore, designing and testing QA datasets for commercial EDA tools poses significant challenges. Most commercial EDA tools explicitly prohibit dataset creation derived from their systems under their user agreements and licensing terms. Conducting such tests could therefore lead to potential violations of these agreements. As a result, we limit our evaluation of MM-GRADE to OpenROAD's documentation QA. Despite this limitation, OpenROAD is a highly professional EDA tool that shares many similarities with commercial solutions. Consequently, the QA evaluation conducted on OpenROAD documentation serves as a strong proxy and reliable demonstration for assessing the transferability of the proposed techniques in MM-GRADE.

V. Conclusion

In this paper, we propose MM-GRADE, a retrieval augmented generator framework customized for the scenario of multi-modal EDA tool documentation question answering (QA). For multi-modal retriever finetuning, we propose a customized bilevel hard negative mining (BHNM) strategy to cater to the particular document retrieval scenario of EDA-tool related queries. Meanwhile for the generator, a domain-specific extract-score-answer pipeline is proposed for generator finetuning and inference during the QA process. Furthermore, we manually design and open-source ORD-MMBench, an evaluation benchmark consisting of 120 high-quality multi-modal query-document-answer triplets. Experimental results on ORD-MMBench demonstrate that MM-GRADE outperforms other SOTA MM-RAG flows on the task of multi-modal EDA-tool documentation QA.

REFERENCES

[1] T. Ajayi, V. A. Chhabria, M. Fogaça, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem *et al.*, "Toward an open-source digital flow: First learnings from the openroad project," in *Proc. DAC*, 2019, pp. 1–4.

[2] X. Li, S. Tao, Z. Huang, S. Chen, Z. Zeng, L. Ni, Z. Huang, C. Zhuang, H. Wu, W. Li *et al.*, "ieda: An open-source intelligent physical implementation toolkit and library," *arXiv preprint*, 2023.

[3] M. Liu, T.-D. Ene, R. Kirby, C. Cheng, N. Pinckney, R. Liang, J. Alben, H. Anand, S. Banerjee, I. Bayraktaroglu *et al.*, "Chipnemo: Domain-adapted llms for chip design," *arXiv preprint*, 2023.

[4] K. Chang, Y. Wang, H. Ren, M. Wang, S. Liang, Y. Han, H. Li, and X. Li, "Chipgpt: How far are we from natural language hardware design," *arXiv preprint*, 2023.

[5] M. Liu, N. Pinckney, B. Khailany, and H. Ren, "Verilogeval: Evaluating large language models for verilog code generation," in *Proc. ICCAD*. IEEE, 2023, pp. 1–8.

[6] Y. Fu, Y. Zhang, Z. Yu, S. Li, Z. Ye, C. Li, C. Wan, and Y. C. Lin, "Gpt4aigchip: Towards next-generation ai accelerator design automation via large language models," in *Proc. ICCAD*. IEEE, 2023, pp. 1–9.

[7] J. Blocklove, S. Garg, R. Karri, and H. Pearce, "Chip-chat: Challenges and opportunities in conversational hardware design," in *Proc. MLCAD*. IEEE, 2023, pp. 1–6.

[8] S. Thakur, J. Blocklove, H. Pearce, B. Tan, S. Garg, and R. Karri, "Autochip: Automating hdl generation using llm feedback," *arXiv preprint*, 2023.

[9] Y. Lu, S. Liu, Q. Zhang, and Z. Xie, "RTLLM: An open-source benchmark for design rtl generation with large language model," in *Proc. ASPDAC*. IEEE, 2024, pp. 722–727.

[10] S. Thakur, B. Ahmad, Z. Fan, H. Pearce, B. Tan, R. Karri, B. Dolan-Gavitt, and S. Garg, "Benchmarking large language models for automated verilog rtl code generation," in *Proc. DATE*. IEEE, 2023, pp. 1–6.

[11] S. Thakur, B. Ahmad, H. Pearce, B. Tan, B. Dolan-Gavitt, R. Karri, and S. Garg, "Verigen: A large language model for verilog code generation," *ACM TODAES*, 2023.

[12] Z. Pei, H.-L. Zhen, M. Yuan, Y. Huang, and B. Yu, "BetterV: Controlled Verilog Generation with Discriminative Guidance," *arXiv preprint*, 2024.

[13] Z. He, H. Wu, X. Zhang, X. Yao, S. Zheng, H. Zheng, and B. Yu, "ChatEDA: A Large Language Model Powered Autonomous Agent for EDA," in *Proc. MLCAD*, 2023, pp. 1–6.

[14] H. Wu, Z. He, X. Zhang, X. Yao, S. Zheng, H. Zheng, and B. Yu, "Chateda: A large language model powered autonomous agent for eda," *IEEE TCAD*, 2024.

[15] Y. Tsai, M. Liu, and H. Ren, "Rtlfixer: Automatically fixing rtl syntax errors with large language models," *arXiv preprint*, 2023.

[16] M. Orenes-Vera, A. Manocha, D. Wentzlaff, and M. Martonosi, "Autosva: Democratizing formal verification of rtl module interactions," in *Proc. DAC*. IEEE, 2021, pp. 535–540.

[17] R. Kande, H. Pearce, B. Tan, B. Dolan-Gavitt, S. Thakur, R. Karri, and J. Rajendran, "Llm-assisted generation of hardware assertions," *arXiv preprint*, 2023.

[18] X. Meng, A. Srivastava, A. Arunachalam, A. Ray, P. H. Silva, R. Psiakis, Y. Makris, and K. Basu, "Unlocking hardware security assurance: The potential of llms," *arXiv preprint*, 2023.

[19] S. Paria, A. Dasgupta, and S. Bhunia, "Divas: An llm-based end-to-end framework for soc security analysis and policy-based protection," *arXiv preprint*, 2023.

[20] B. Ahmad, S. Thakur, B. Tan, R. Karri, and H. Pearce, "Fixing hardware security bugs with large language models," *arXiv preprint*, 2023.

[21] X. Yao, H. Li, T. H. Chan, W. Xiao, M. Yuan, Y. Huang, L. Chen, and B. Yu, "HDLdebugger: Streamlining HDL debugging with Large Language Models," *arXiv preprint*, 2024.

[22] Y. Pu, Z. He, T. Qiu, H. Wu, and B. Yu, "Customized Retrieval Augmented Generation and Benchmarking for EDA Tool Documentation QA," *arXiv preprint*, 2024.

[23] U. Sharma, B.-Y. Wu, S. R. D. Kankipati, V. A. Chhabria, and A. Rovinski, "OpenROAD-Assistant: An Open-Source Large Language Model for Physical Design Tasks," in *Proc. MLCAD*, 2024, pp. 1–7.

[24] A. Kaintura, S. S. Luar, I. I. Almeida *et al.*, "ORAssistant: A Custom RAG-based Conversational Assistant for OpenROAD," *arXiv preprint*, 2024.

[25] B.-Y. Wu, U. Sharma, S. R. D. Kankipati, A. Yadav, B. K. George, S. R. Guntupalli, A. Rovinski, and V. A. Chhabria, "EDA Corpus: A Large Language Model Dataset for Enhanced Interaction with OpenROAD," *arXiv preprint*, 2024.

[26] W. Chen, H. Hu, X. Chen, P. Verga, and W. Cohen, "MuRAG: Multimodal Retrieval-Augmented Generator for Open Question Answering over Images and Text," in *Proc. NIPS*, 2022, pp. 5558–5570.

[27] W. Lin and B. Byrne, "Retrieval Augmented Visual Question Answering with Outside Knowledge," in *Proc. NIPS*, 2022, pp. 11 238–11 254.

[28] W. Lin, J. Chen, J. Mei, A. Coca, and B. Byrne, "Fine-grained late-interaction multi-modal retrieval for retrieval augmented visual question answering," *Proc. NIPS*, vol. 36, pp. 22 820–22 840, 2023.

[29] H. Liu, K. Son, J. Yang, C. Liu, J. Gao, Y. J. Lee, and C. Li, "Learning customized visual models with retrieval-augmented knowledge," in *Proc. CVPR*, 2023, pp. 15 148–15 158.

[30] D. Caffagni, F. Cocchi, N. Moratelli, S. Sarto, M. Cornia, L. Baraldi, and R. Cucchiara, "Wiki-LLaVA: Hierarchical Retrieval-Augmented Generation for Multimodal LLMs," in *Proc. CVPR*, 2024, pp. 1818–1826.

[31] S. Yu, C. Tang, B. Xu, J. Cui, J. Ran, Y. Yan, Z. Liu, S. Wang, X. Han, Z. Liu *et al.*, "VisRAG: Vision-based Retrieval-augmented Generation on Multi-modality Documents," *arXiv preprint*, 2024.

[32] Y. Yan and W. Xie, "EchoSight: Advancing Visual-Language Models with Wiki Knowledge," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 1538–1551.

[33] J. Zhou, Z. Liu, S. Xiao, B. Zhao, and Y. Xiong, "VISTA: Visualized Text Embedding For Universal Multi-Modal Retrieval," *arXiv preprint*, 2024.

[34] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Proc. NIPS*, vol. 33, pp. 9459–9474, 2020.

[35] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proc. ACL*.

[36] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Workshop on Text Summarization Branches Out*, 2004, pp. 74–81.

[37] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *Proc. ICML*. PMLR, 2021, pp. 8748–8763.

[38] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *Proc. ICML*. PMLR, 2022, pp. 12 888–12 900.

[39] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. Hoi, "InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning," 2023. [Online]. Available: https://arxiv.org/abs/2305.06500

[40] A. Salemi, J. Altmayer Pizzorno, and H. Zamani, "A symmetric dual encoding dense retrieval framework for knowledge-intensive visual question answering," in *Proc. SIGIR*, 2023, pp. 110–120.

[41] Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu *et al.*, "Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks," in *Proc. CVPR*, 2024, pp. 24 185–24 198.

[42] Z. Chen, W. Wang, H. Tian, S. Ye, Z. Gao, E. Cui, W. Tong, K. Hu, J. Luo, Z. Ma *et al.*, "How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites," *arXiv preprint*, 2024.

[43] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Proc. NIPS*, vol. 35, pp. 24 824–24 837, 2022.

[44] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation," 2024.

[45] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford *et al.*, "GPT-4o System Card," *arXiv preprint*, 2024.

[46] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," in *Proc. CVPR*, 2024, pp. 26 296–26 306.