

RATuner: Retrieval-Augmented VLSI Flow Design Parameter Tuning Framework

Peng Xu^{†1}, Ziyang Yu^{†1}, Yuan Pu¹, Xinyun Zhang¹, Donger Luo², Hao Geng², Siyuan Xu³, Tsung-yi Ho¹, Bei Yu¹
¹CUHK ²ShanghaiTech University ³Huawei

Abstract—Optimizing configurable parameters in the Very-large-scale Integration (VLSI) design space is a key process for achieving high Quality-of-Result (QoR) metrics, including performance, power, and area. However, this task is severely challenged by the enormous design space, the lack of analytical mapping functions with QoR, and the high computational cost of evaluating each design choice. While Bayesian Optimization (BO) offers a balanced trade-off between exploitation and exploration, standard BO methods typically do not incorporate domain knowledge specific to VLSI design flow parameters. To address these limitations, we propose RATuner, a retrieval-augmented framework for high-dimensional VLSI Flow Design Space Exploration (DSE). RATuner integrates domain knowledge through a document-retrieval-based embedding method to guide Bayesian optimization, using design parameter embeddings constructed from EDA documentation. It further employs a stage-wise causal attention mechanism to model both intra-stage parameter interactions and the critical inter-stage causal dependencies present in the sequential VLSI design flow. Finally, an iterative Bayesian optimization strategy is utilized to achieve an efficient trade-off between exploitation and exploration. Experimental results on RISC-V and Blackparrot benchmarks show that RATuner achieves up to 33% improvement in Pareto-driven QoR metrics compared to representative state-of-the-art VLSI Flow DSE methods. The proposed framework bridges the gap between black-box optimization and VLSI domain expertise by incorporating domain knowledge, thereby improving the efficiency and quality of automatic VLSI design closure.

I. INTRODUCTION

VLSI flow design space exploration is a critical factor in design closure, as it seeks Pareto-optimal solutions that balance performance, power, area, and manufacturability. Empirical studies indicate that sub-optimal parameters typically require 3–4 additional iterations, extending the tape-out schedule [1]. Because commercial EDA tools expose only tunable parameters while keeping their core algorithms proprietary, systematic design space exploration becomes highly valuable. Nevertheless, exhaustive parameter tuning is computationally infeasible due to the size of the search space. The combined configuration space of synthesis, floorplan, placements, and routing parameters can exceed 10^{70} combinations under typical industrial configurations [2]. Moreover, as illustrated in Fig. 1, the VLSI flow consists of a sequence of stages, including synthesis to routing. This sequential decomposition complicates the quantification of interdependencies between step-wise algorithmic improvements. The complicated correlation necessitates optimization approaches that consider interactions across the whole design space. Absent effective design exploration, even functionally correct architectures can miss competitive PPA targets, potentially compromising commercial viability.

Although existing VLSI Flow DSE methods employ black-box optimization to improve Pareto-optimal QoR, design parameter descriptions provided in EDA-tool manuals have rarely been exploited [3]. Recent work has combined heuristic and statistical approaches [4]–[8], with Bayesian optimization (BO) becoming

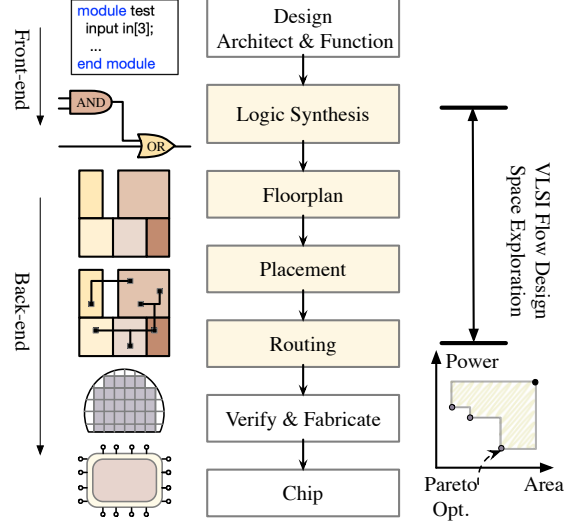


Fig. 1 The typical VLSI design flow, including the Front-end and Back-end design. In the VLSI design, design flow optimization across stages is particularly critical, which decides the final QoR.

prevalent owing to its sample-efficient balance between exploration and exploitation [2], [9], [10]. Bayesian optimization, however, scales poorly in high-dimensional design spaces. By projecting the search into a lower-dimensional latent space, Latent Bayesian Optimization (LBO) has shown improved scalability in domains such as drug discovery [11], [12]. These applications rely on structured priors (e.g., SMILES strings or molecular graphs) that are absent in VLSI design. The tabular format commonly used in VLSI Flow DSE lacks a structural encoding with design expertise. Consequently, LBO adoption in this field remains limited [13], [14]. Exploiting the documented information could provide a promising direction toward navigating the high-dimensional VLSI Flow DSE.

Language models (LMs), especially Large language models (LLMs), offer a promising pathway to address the limitations of insufficient semantic information [15]. Models like BERT [16] and GPT-4 [17] demonstrate transformative capabilities in understanding complex linguistic patterns and generating contextually coherent responses through semantic embeddings and in-context learning [18]. However, there is currently no established paradigm that effectively incorporates such semantically rich embeddings into the VLSI Flow DSE. To address this critical gap, we propose the following research question: *Can retrieval embeddings from EDA documentations enhance latent space construction for VLSI design parameters?*

We present **RATuner** (Retrieval-Augmented VLSI Flow Design Parameter Tuning Framework), an intelligent retrieval-augmented framework for VLSI Flow DSE. RATuner uses document-based retrieval on EDA-tool manuals to construct embedding vectors for

[†]Equal contributors.

This work is partially supported by The Research Grants Council of Hong Kong SAR (No. CUHK14211824 and No. CUHK14210723).

parameter context. We then propose a hybrid causal attention mechanism to capture both intra-stage interactions and inter-stage causal dependencies. These parameter embeddings are projected into a low-dimensional latent space and fed into a deep-kernel Gaussian process surrogate model. Our contributions are summarized as follows:

- We present a retrieval-augmented framework that explores the high-dimensional VLSI design space by leveraging textual descriptions in EDA documentation;
- We propose RCNet, a causality-aware embedding model that employs structured attention masks to encode causal dependencies, mapping parameters and their context to a latent space.
- We design an iterative retrieval-augmented Bayesian optimization loop that uses a deep Gaussian process surrogate and Thompson sampling. It refines the latent space dynamically and reduces the number of expensive evaluations compared with standard BO.
- Experimental results across three industrial-level benchmarks show our framework outperforms the recent VLSI Flow DSE methods, achieving up to 33% improvement in Pareto-driven QoR.

II. RELATED WORK AND PRELIMINARIES

A. VLSI Flow Design Space Exploration

Traditional VLSI Flow Design Space Exploration relied on heuristics (evolutionary strategies [4], [5], ant colony optimization [6]), deep matrix factorization [7], and tree-based models (e.g., FIST [8]). Bayesian optimization (BO) has become prominent for its sample efficiency and exploration-exploitation balance. Recent BO advances encompass multi-objective optimization [2], multi-task GPs in multi-objective contexts [9], scalable latent embedding methods like REMOTune [13], hybrid techniques with mixed-type kernels [10], and preference-based approaches such as RankTuner [14]. Nevertheless, existing black-box methods still lack integration of VLSI flow parameters’ physical significance and domain-specific knowledge.

B. Latent Bayesian Optimization

Latent Bayesian Optimization (LBO) [11], [19] overcomes traditional BO’s limitations in high-dimensional/discrete spaces by embedding inputs into a continuous latent space, typically using Variational Autoencoders (VAEs) [20], [21]. Key enhancements include improving reconstruction quality, incorporating modeling uncertainty [22], and developing specialized transformer-based architectures like LaMBO [23] and its diffusion-based successor [24]. Recent innovations integrate trust regions, enhance solution diversity (e.g., ROBOT [25]), optimize latent-function alignment (e.g., CoBO [26]), and employ bijective mappings via normalizing flows (e.g., NF-BO [27]).

C. Problem Statement

The VLSI Flow DSE aims to identify high-performing parameter configurations $\mathbf{x} \in \mathcal{X}$ from a high-dimensional space $\mathcal{X} \subseteq \mathbb{R}^d$ to maximize QoR metrics $\mathbf{y} = (y_1, \dots, y_m)$ (e.g., timing, power, area). The objective function $f(\mathbf{x}) : \mathcal{X} \rightarrow \mathbb{R}^m$ lacks an analytical form and is computationally intensive to evaluate, typically requiring hours per design parameter configuration. The key challenges of the VLSI Flow DSE task include:

- **High dimensionality:** $\mathcal{X}$ contains numerous design parameters spanning sequential EDA stages.

Design Stage:	Floorplan
Design Parameter:	-coreMarginsBy
Related context:	Specifies whether the core margins are calculated using the core-to-I/O boundary or the core-to-die boundary. Default: io
Design Choices:	{io die}
Related context:	Specifies the floorplan dimensions by size; or by die, I/O, or core coordinates. The floorPlan command initializes the floorplan and calls the add_tracks command to create new routing tracks.

Fig. 2 The context information for VLSI design parameter “-coreMarginsBy” from the floorplan stage.

- **Underutilized domain knowledge:** While EDA design documentation $\mathcal{D}$ describes design parameter semantics and constraints, it is typically not incorporated into existing black-box optimizers.
- **Sample efficiency requirement:** The substantial computational cost per evaluation necessitates highly sample-efficient optimization techniques.

The proposed RATuner methodology aims to utilize $\mathcal{D}$ to construct a low-dimensional latent embedding space $\mathcal{E}$ to guide Bayesian optimization within $\mathcal{X}$.

III. METHOD

A. The Overall Framework

As shown in Fig. 3, our RATuner employs retrieval context information to enhance the efficiency of high-dimensional VLSI Flow DSE in three key stages:

- 1) *Contextualized Retrieval:* Processing EDA manuals via a conventional retrieval process and a retriever model fine-tuned on EDA data to generate a contextual embedding for each design parameter;
- 2) *Retrieval-augmented Latent Embedding:* Mapping design parameter value and context vector pairs with sequential dependencies across EDA stages to latent space using a customized transformer model;
- 3) *Retrieval-augmented Bayesian Optimization:* Bayesian optimization using a Deep Gaussian Process (DGP) is performed in the retrieval-augmented latent space, where the next evaluation point is selected via Thompson sampling, achieving highly sample-efficient optimization.

B. Retrieval-augmented Latent Embedding

Optimizing VLSI design parameters is challenging due to the high-dimensional search space. Recent works use latent spaces for dimensionality reduction in BO [13], [14]. Existing latent space construction methods for design parameters often fail to adequately capture correlations among parameters and their values, as observed in REMETuner [13] and RankTuner [14]. A key limitation is the inadequate incorporation of parameter semantics, where parameters are treated as generic tabular data columns. Our solution, RATuner,

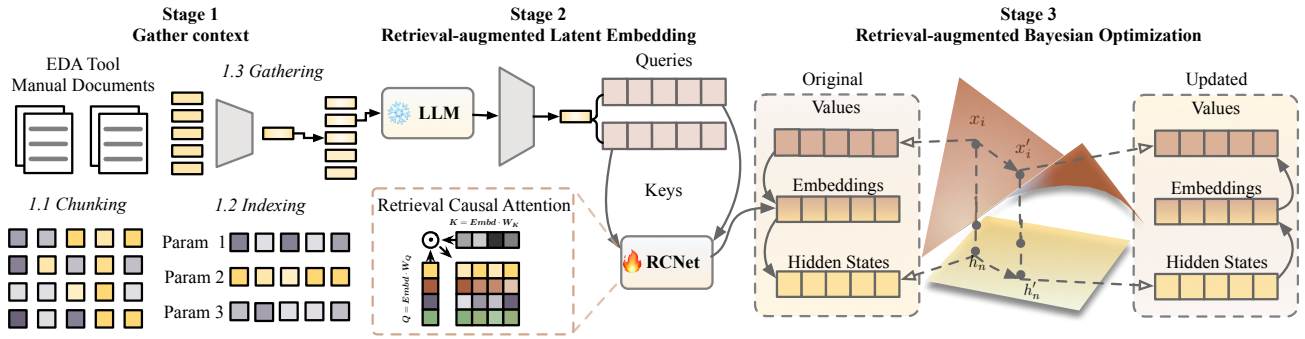


Fig. 3 The overall framework of RATuner consists of three stages: (1) Contextualized retrieval using an EDA domain-adapted retriever for each parameter; (2) Retrieval-augmented latent embedding via a customized Transformer network, providing causality-aware embeddings; and (3) Iterative Bayesian optimization in the latent space for efficient exploration of the parameter search space.

addresses this challenge by integrating domain-specific semantic information from documentation.

Gathering Design-related Context.. We began by processing VLSI design parameter documentation to obtain specific explanatory text segments. Our text processing pipeline follows the general retrieval works [28], [29]: (1) Chunking: Converting diverse EDA manuals into uniformly segmented text chunks; (2) Indexing: Identifying parameter names via rule-based parsing to build a searchable index; (3) Gathering: Aggregating all chunks associated with each parameter’s keywords as Parameter Context. We visualize an example of parameter context information in the floorplan stage in Fig. 2.

Retrieval Embedding Process.. Text embedding models convert document fragments into dense vectors representing semantic content. As noted in [30], standard retrieval models like [31], [32] exhibit limited effectiveness in capturing EDA-specific terminology, resulting in suboptimal vector quality and reduced retrieval precision. To address this, we employ **C-RAG** [30], an EDA-adapted retriever. Specifically, this retriever is trained via supervised contrastive learning to cluster sentences sharing identical EDA terms while separating structurally similar sentences with different terms.

C. Retrieval-Augmented Embedding Model

Typically, the EDA workflow involves sequential causal relationships: parameters selected in earlier stages (e.g., synthesis) affect downstream stages (e.g., placement and routing), ultimately influencing final QoR. For instance, synthesis choices impact floorplanning constraints, which constrain subsequent placement and routing options. Neglecting these dependencies may reduce optimization efficiency and interpretability.

Based on the domain knowledge, we propose **RCNet**, a **Retrieval-based, Causality-constrained** embedding network as detailed in Fig. 4, designed for latent Bayesian optimization on VLSI Flow DSE. Given the retrieval embeddings of relevant documentations for parameters $D \in \mathbb{R}^{n \times d_r}$ where d_r is the retrieval embedding dimension from documentation retrieved via parameter-keyword matching and input features $X \in \mathbb{R}^{B \times n}$, where B is the batch size:

$$Q = DW_Q \in \mathbb{R}^{n \times d}, \quad (1)$$

$$K = DW_K \in \mathbb{R}^{n \times d}, \quad (2)$$

$$V = XW_V \in \mathbb{R}^{B \times n \times d}. \quad (3)$$

Based on the query, key, and value embeddings of the parameter context, we introduce a stage-wise causal attention:

$$\text{Attention}(Q, K, V) = M_{sc} \cdot \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V. \quad (4)$$

Whereas standard causal masks implement unidirectional attention [33], VLSI design parameters exhibit: (1) Known Global causal stages (e.g., synthesis $\rightarrow$ routing), (2) Indeterminate Intra-stage parameter ordering. To model this, M_{sc} enforces inter-stage causality while permitting flexible intra-stage attention:

$$M_{sc} = \begin{bmatrix} \mathbf{1}_{n_1 \times n_1} & \mathbf{0}_{n_1 \times n_2} & \cdots & \mathbf{0}_{n_1 \times n_K} \\ \mathbf{1}_{n_2 \times n_1} & \mathbf{1}_{n_2 \times n_2} & \cdots & \mathbf{0}_{n_2 \times n_K} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}_{n_K \times n_1} & \mathbf{1}_{n_K \times n_2} & \cdots & \mathbf{1}_{n_K \times n_K} \end{bmatrix}, \quad (5)$$

where $\mathbf{1}_{a \times b}$, $\mathbf{0}_{a \times b}$ denote all-ones/zeros matrices ($K = 4$ stages). This mask restricts stage k parameters to attend to all prior stages (1 to k). By encoding known dependencies while accommodating local uncertainty, our approach is designed to enhance interpretability and improve optimization.

D. Retrieval-Augmented Bayesian Optimization

Our retrieval-augmented Bayesian optimization iteratively operates through three stages, as shown in Fig. 5: (1) **Preparation**: Define parameter space and initialize latent embedding; (2) **Optimization**: Refine Deep Gaussian Process (DGP) via marginal likelihood maximization using candidate parameters, balancing exploration-exploitation via acquisition functions; (3) **Model Updating**: Adjust latent embedding and DGP with evaluated parameters to update the search space and optimum estimates.

Learning the Retrieval-augmented Latent Space.. We apply a variational autoencoder to learn a variational distribution $q_\phi(z|x, u)$ approximating the true posterior $p_\theta(z|x, u)$, as shown in Fig. 6. Specifically, u is the context vector, ϕ and θ parameterize the RCNet encoder and MLP decoder. Given $x \sim \mathcal{X}$, we optimize ϕ by minimizing the negative Evidence Lower Bound (ELBO):

$$\mathcal{L}(\phi; x) = \mathbb{E}_{q_\phi}[\log p_\theta(x|z, u)] + \mathcal{D}_{KL}(q_\phi||p(z|u)) \quad (6)$$

where $\mathcal{D}_{KL}(q(\cdot)||p(\cdot))$ is the Kullback-Leibler divergence between distributions $q(\cdot)$ and $p(\cdot)$. We use a standard Gaussian prior $p(z|u) = \mathcal{N}(0, I)$ during the initialization phase. After fixed iterations, we augment the latent space with a QoR prediction head and reconstruction loss following [23].

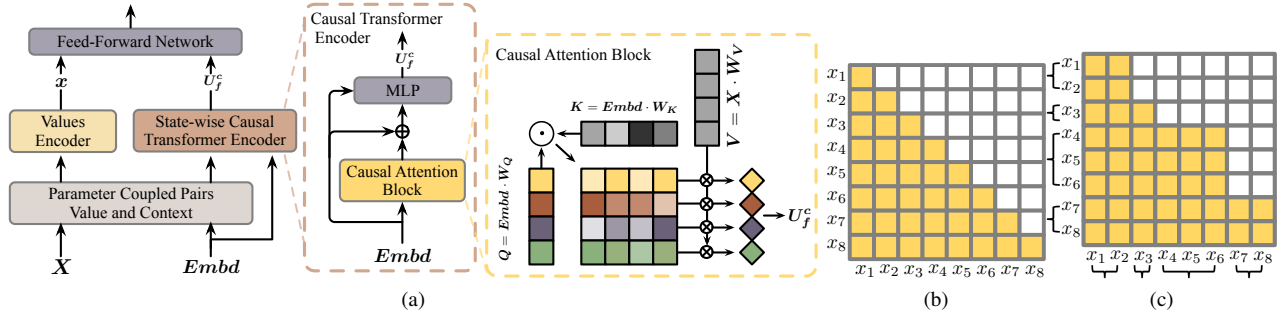


Fig. 4 RCNet introduces a causality-aware transformer encoder that uses a stage-wise causal attention mask to enforce causal dependencies between EDA design stages while allowing flexible intra-stage interactions: (a) The architecture of RCNet; (b) the standard causal attention mask; (c) the stage-wise causal attention mask.

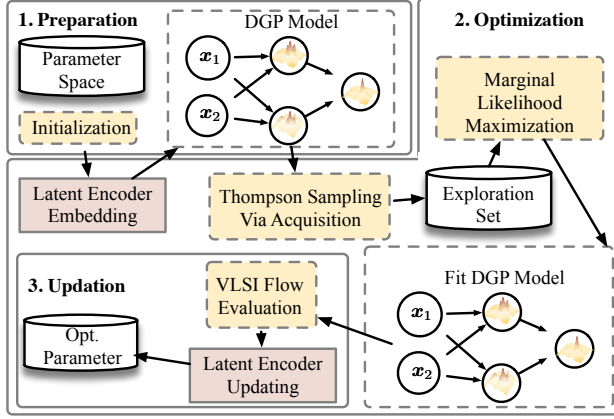


Fig. 5 Our retrieval-augmented Bayesian optimization framework consists of three main stages, i.e., stage 1: initialize the latent embedding model and DGP surrogate model based on the search space, stage 2: optimize the DGP model with Thompson sampling exploration, stage 3: update the latent embedding model with VLSI flow evaluation results.

Deep Gaussian Process as Surrogate. To model complex parameter dependencies and provide calibrated uncertainty in high dimensions, we use a Deep Gaussian Process (DGP) [34], [35] as our surrogate model.

Given observations $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^n$, the DGP posterior at $\mathbf{x}$ is Gaussian with mean $\mu(\mathbf{x})$ and variance $\sigma^2(\mathbf{x})$:

$$\begin{cases} \mu(\mathbf{x}_*) = \tilde{\mathbf{k}}_\phi(\mathbf{x}_*, \mathbf{X})^\top [\mathbf{K} + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{Y}, \\ \sigma^2(\mathbf{x}_*) = \tilde{\mathbf{k}}_\phi(\mathbf{x}_*, \mathbf{x}_*) - \tilde{\mathbf{k}}_\phi(\mathbf{x}_*, \mathbf{X})^\top [\mathbf{K} + \sigma_n^2 \mathbf{I}]^{-1} \tilde{\mathbf{k}}_\phi(\mathbf{x}_*, \mathbf{X}), \end{cases}$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ are the observed design parameters and $\mathbf{Y}$ corresponds to the vector of observed outputs from the search space $\mathcal{X}$. $\mathbf{K}_{ij} = \tilde{\mathbf{k}}_\phi(\mathbf{x}_i, \mathbf{x}_j)$ is the Gram matrix, and σ_n^2 is the noise variance. The feature extractor $h_\phi(\cdot)$ provides the GP with a compact, informative space for efficient modeling and improved uncertainty estimates to guide Bayesian acquisition. To fit the DGP model, the hyperparameters of the kernel function are optimized by maximizing the marginal likelihood of the observed data:

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2} \mathbf{y}^\top (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K} + \sigma_n^2 \mathbf{I}|.$$

1) *Acquisition Function in Latent Space.*: Candidate samples for the acquisition function are generated by randomly selecting points within a region centered on the current optimum. We employ

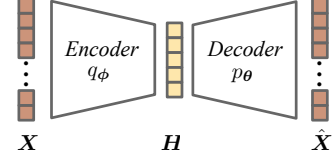


Fig. 6 The variational auto-encoder encoder for design parameter latent embedding.

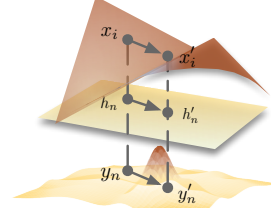


Fig. 7 Thompson sampling generates candidate samples around the current optimal value, then scores them using the surrogate model's uncertainty. The parameter $\mathbf{x}'$ is extracted by passing the latent vector through the VAE decoder

Thompson sampling as the acquisition function within the Bayesian optimization framework. The surrogate model provides a prior belief about the objective function. As shown in Fig. 7, Thompson sampling utilizes this model's predictive uncertainty to draw sample paths, which are used to select the next point for evaluation. Specifically, we first select candidate samples and evaluate each candidate using the posterior of the surrogate model to identify the candidate maximizing the acquisition function value. Additionally, we map the candidate parameter values $\mathbf{x}$ to the latent space using the encoder q_ϕ , obtaining the corresponding latent vectors $\hat{\mathbf{z}}$. Separately, the parameter values $\hat{\mathbf{x}}^*$ corresponding to the current optimal latent vector $\mathbf{z}^*$ are obtained by decoding it with p_θ , i.e., $\hat{\mathbf{x}}^* \sim p_\theta(\mathbf{z}^*)$.

IV. EXPERIMENTS

A. Experimental Setting

Benchmarks. We evaluate our proposed framework on three benchmark processors: *RISCV-32I* [36], *RISCV-Rocket* [37], and *BlackParrot (BP)* [38]. All benchmarks are implemented on TSMC's 65nm technology node. *RISCV-32I* and *RISCV-Rocket* are RISC-V cores. *RISCV32I* (7.6k cells) is a manually designed Verilog implementation, while *Rocket* (14.2k cells) is generated from Chisel source

TABLE I Comparison of VLSI Flow DSE methods on $RISCV_{32I}$ and $RISCV_{Rocket}$ benchmark. Best results are in **bold**.

Method	HV ($\times 10^5$)	HV _{0,1} ($\times 10^3$)	HV _{0,2} ($\times 10^3$)	HV _{1,2} ($\times 10^3$)	MPI1 (%)	MPI2 (%)	MAI (%)	MPPI (%)	MPAI (%)
<i>RISCV_{32I}</i> Benchmark									
FIST	1.57	2.85	2.94	2.97	3.16	3.90	5.47	6.94	8.46
Recommender	1.55	2.72	2.99	2.97	2.54	2.12	7.18	4.51	9.53
AutoTuner	1.68	2.95	3.07	3.14	3.81	5.23	7.10	8.83	10.63
MOBO	1.63	3.00	3.00	3.00	5.00	5.12	4.64	9.88	9.41
PTPT	1.48	2.70	2.95	2.79	3.56	0.85	5.15	4.37	8.52
REMOTune	1.75	3.05	3.12	3.23	4.38	6.27	7.45	10.38	11.53
HyBO	1.44	2.63	2.84	2.77	2.08	0.68	4.74	1.30	5.43
RankTuner	1.84	3.44	3.43	3.00	13.64	5.04	5.12	13.73	12.26
RATuner (Ours)	2.05	3.67	3.65	3.34	15.54	8.48	7.44	10.15	12.36
<i>RISCV_{Rocket}</i> Benchmark									
FIST	1.47	3.03	3.02	2.42	12.38	-0.51	-1.01	11.93	11.50
Recommender	1.19	2.79	2.75	1.85	14.50	-6.70	-7.25	8.77	8.30
AutoTuner	1.50	3.16	3.16	2.23	16.72	-2.42	-2.55	14.70	14.60
MOBO	1.35	2.93	2.94	2.19	13.44	-2.99	-3.32	10.85	10.67
PTPT	1.31	2.85	2.84	2.20	11.97	-2.83	-3.39	9.48	8.99
REMOTune	1.61	3.35	3.18	2.51	16.11	1.57	-1.31	17.43	15.01
HyBO	1.36	2.97	2.63	2.40	7.34	2.16	-3.79	5.46	1.02
RankTuner	1.67	3.45	3.06	2.69	13.00	5.09	-0.96	13.11	6.68
RATuner (Ours)	1.73	3.59	3.10	2.76	13.85	6.69	-0.96	14.54	7.60

TABLE II Ablation study on the different processors.

Method	<i>RISCV_{32I}</i> HV ($\times 10^5$)	<i>RISCV_{Rocket}</i> HV ($\times 10^5$)	<i>BlackParrot</i> HV ($\times 10^5$)	Avg.
FIST	1.57	1.47	0.98	1.34
Recommender	1.55	1.19	1.10	1.28
AutoTuner	1.68	1.50	1.01	1.40
MOBO	1.63	1.35	0.99	1.32
PTPT	1.48	1.31	1.00	1.26
REMOTune	1.75	1.61	1.11	1.49
RankTuner	1.84	1.67	1.18	1.56
RATuner (Ours)	2.05	1.73	1.26	1.68

code. In contrast, *BlackParrot* is a multicore processor capable of running on Linux, and with 43.2k cells, represents a processor of significantly larger scale and higher design complexity. The parameter space comprises 134 parameters and is based on established parameter spaces used in prior VLSI Flow DSE research [9], [13]. Prior work employing similar parameter spaces has demonstrated effectiveness in both academic and industrial settings [9], [13].

Compared methods. We compare our approach against the following state-of-the-art (SOTA) baselines as in [14]: (1) **FIST** [8], which uses XGBoost ensemble trees and importance sampling for EDA parameter adjustment; (2) **Recommender** [7], employing tensor decomposition and regression for collaborative prediction to reduce exploration effort; (3) **MOBO** [2], applying multi-objective Bayesian optimization for EDA parameter space exploration; (4) **AutoTuner** [5], an open-source platform integrating evolutionary algorithms and tree-structured Parzen estimators; (5) **PTPT** [9], using multi-objective Bayesian optimization for Pareto-optimal parameters and multi-task Gaussian processes for objective correlation; (6) **REMOTune** [13], employing random embedding and multi-objective trust-region Bayesian optimization for guided exploration; (7) **Explorer** [10], an attention-based explorer using hybrid Gaussian processes to model continuous and discrete parameter interactions; (8) **RankTuner** [14], a ranking-based exploration framework achieving state-of-the-art results in VLSI Flow DSE.

Evaluation metrics. The proposed methods target the optimization of key QoR metrics in VLSI Flow DSE. The QoR metrics evaluated in this work, selected based on their common usage in recent research [13], [14], include hypervolume (HV), Maximum Performance Improvement 1 (MPI1), Maximum Power Improvement (MPI2), Maximum Area Improvement (MAI), Maximum Performance-Power Improvement (MPPI), and Maximum Performance-Area Improvement (MPAI). HV is calculated using a reference point of [150.0, 150.0, 150.0], corresponding to the worst acceptable values. We also evaluate HV in specific two-dimensional subspaces: HV_{0,1} for performance and power, HV_{0,2} for performance and area, and HV_{1,2} for power and area. MPI1, MPI2, and MAI assess improvements in clock period, power, and area, respectively. MPPI and MPAI evaluate improvements in the performance-power and performance-area products, respectively.

Implementation Details. We utilize the CRAG retriever model in RATuner, using the identical weights [30]. RCNet adopts a 2-layer causal transformer encoder with 32-dimensional embeddings, and the decoder is a 2-layer MLP with 128 hidden units. For initialization, we sample 10,000 valid parameters without performing VLSI evaluation, employing the ELBO objective trained for 100 epochs. All methods are evaluated on the same hardware/software platform using identical tool versions, including Cadence Genus and Innovus v17.1. All methods are implemented using BoTorch, Optuna, and PyTorch, with hyperparameter settings following [13], [18]. A maximum iteration budget of 210 is set for all VLSI Flow DSE methods.

B. Comparison Results

The comprehensive evaluation presented in TABLE I and Fig. 8 demonstrates that RATuner achieves higher QoR metric values compared to other evaluated VLSI Flow DSE methods across the primary RISC-V benchmarks, showing improvement over the exploration process. For the *RISCV_{32I}* benchmark, RATuner achieves a hypervolume (HV) of 2.05×10^5 , which is 11.4% higher than the result obtained by RankTuner. Similarly, on the *RISCV_{Rocket}*

TABLE III Ablation study for different modules of RATuner on *RISCV_{32l}*, including Bayesian Optimization (Bayesian Opt.), Parameter Latent Embedding (Emb.), and Retrieval Augmentation (Retrieval Aug.). In the brackets, we show the improvements achieved by RATuner over the corresponding variant. We bold the best performance and the largest improvement.

Method	Bayesian Opt.	Latent Emb.	Retrieval Aug.	HV ($\times 10^5$)	MPI1 (%)	MPI2 (%)	MAI (%)
RATuner (Imps.)	✓			1.70 (+20.59%)	3.79 (+310.03%)	5.79 (+46.46%)	7.07 (+5.23%)
	✓	✓		1.75 (+17.14%)	4.39 (+253.99%)	6.27 (+35.25%)	7.45 (-0.13%)
	✓	✓	✓	2.05	15.54	8.48	7.44

TABLE IV Comparison between the different LBO variants.

Method	<i>RISCV_{32l}</i> HV ($\times 10^5$)	<i>RISCV_{Rocket}</i> HV ($\times 10^5$)	<i>BlackParrot</i> HV ($\times 10^5$)	Avg.
REMOTune	1.75	1.61	1.11	1.49
Sparse RA.	1.87	1.70	1.16	1.58
Ours	2.05	1.73	1.26	1.68

TABLE V Comparison between RankTuner^{RA} and RankTuner (w/o. retrieval augmentation).

Method	<i>RISCV_{32l}</i> HV ($\times 10^5$)	<i>RISCV_{Rocket}</i> HV ($\times 10^5$)	<i>BlackParrot</i> HV ($\times 10^5$)	Avg.
RankTuner	1.84	1.67	1.18	1.56
RankTuner ^{RA}	1.92	1.72	1.25	1.63

benchmark, RATuner attains an HV of 1.73×10^5 , representing a 3.6% improvement over RankTuner. These results indicate that RATuner can effectively identify Pareto-optimal tool parameters for VLSI design.

RATuner also performs well across multiple QoR metrics and specific objective combinations. On *RISCV_{32l}*, RATuner achieves the highest HV_{0,1} with 6.7% higher than REMOTune and HV_{0,2} with 4.8% higher than REMOTune, while its HV_{1,2} is the second highest among the methods. RATuner also achieves the highest scores for MPI1 with 13.8% higher than AutoTuner and MPI2 with 35.4% higher than REMOTune. On *RISCV_{Rocket}*, RATuner leads in HV_{0,1} and HV_{1,2}. Additionally, RATuner achieves the best MPI2 with 31.4% higher than RankTuner, while its MAI result is comparable to the best baseline. These results demonstrate RATuner’s ability to achieve improvements across multiple QoR metrics.

C. Ablation Studies

Performance on Heterogeneous Processor. Experimental results indicate that RATuner achieves the highest Hypervolume (HV) scores across the evaluated platforms, including *RISCV_{32l}*, *RISCV_{Rocket}*, and *BlackParrot*, with improvements over the next-best method (RankTuner) ranging from 11% to 21%. These results suggest that RATuner is robust and adaptable to diverse processor architectures, as evidenced by its average HV of 1.68×10^5 , which outperforms all competing methods.

Comparison between Different LBO Variants. TABLE IV compares three latent Bayesian optimization variants: LBO with Random Embedding (LBO), LBO with Sparse Retrieval Augmentation (Sparse RA), and our LBO with Dense Retrieval Augmentation (Dense RA). Dense RA achieves the highest observed hypervolume (HV) scores across all test cases and exhibits optimal average performance, with an average improvement of 6.3% over Sparse RA and 12.8% over baseline LBO. Detailed performance metrics, including standard deviations, are provided in TABLE IV. These

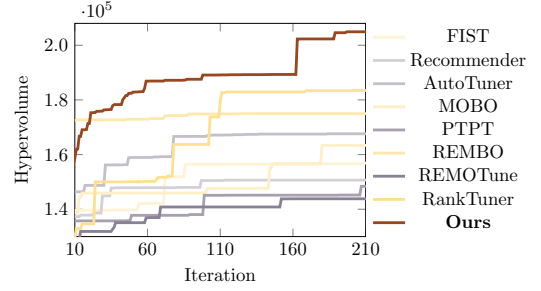


Fig. 8 Convergence for RATuner and other SOTA methods in terms of Hypervolume.

results provide evidence that dense retrieval offers a critical advantage for VLSI Flow DSE, particularly in terms of hypervolume performance. We provide additional comparison between retrievers in the Appendix.

Retrieval-augmentation Scheme at Work. To validate our retrieval-augmentation scheme, we test it on other methods. As RankTuner outperforms all baselines, we augment it with retrieval (RankTuner^{RA}). TABLE V shows RankTuner^{RA} surpasses the unaugmented version, indicating that retrieval-based latent space enhancement improves VLSI Flow DSE performance.

Effectiveness Investigation of Each Module. TABLE III evaluates the effectiveness of RATuner modules through iterative addition: (1) Bayesian Optimization (vanilla BO) struggles in high-dimensional search spaces; (2) Latent Embedding (tabular embedding) fails to account for physical meanings; (3) Retrieval Augmentation (our method) contributes the most to QoR gains. Removing any single module results in significant performance degradation, supporting the effectiveness of all components in the RATuner pipeline. The substantial contribution of Retrieval Augmentation highlights the importance of well-constructed latent embeddings in the VLSI Flow DSE workflows.

V. CONCLUSION

In this paper, we present RATuner, a retrieval-augmented VLSI Flow DSE framework that integrates design knowledge extracted from EDA documentation to improve sample efficiency. Our method constructs semantic-aware parameter embeddings via an EDA-adapted retriever and models inter-stage causal dependencies with a stage-wise attention mechanism. These embeddings are used in a deep-kernel Gaussian process surrogate model with Thompson sampling, which iteratively refines the latent space and reduces the number of expensive evaluations compared to baseline methods. Experimental results on RiscV and BlackParrot show that RATuner achieves up to 33% improvement in HyperVolume in the best case, with an average improvement of 23% over representative SOTA VLSI Flow DSE methods.

REFERENCES

- [1] E. Yao and A. Basu, "Vlsi extreme learning machine: A design space exploration," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 25, no. 1, pp. 60–74, 2016.
- [2] Y. Ma, Z. Yu, and B. Yu, "CAD tool design space exploration via bayesian optimization," in *ACM/IEEE Workshop on Machine Learning CAD (MLCAD)*, 2019.
- [3] D. Luo, Q. Sun, P. Xu, S. Zheng, Q. Xu, T. Chen, B. Yu, and H. Geng, "Attention-based eda tool parameter explorer: From hybrid parameters to multi-qor metrics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pp. 1–1, 2025.
- [4] M. M. Ziegler, H.-Y. Liu, G. Gristede, B. Owens, R. Nigaglioni, and L. P. Carloni, "A synthesis-parameter tuning system for autonomous design-space exploration," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2016, pp. 1148–1151.
- [5] J. Jung, A. B. Kahng, S. Kim, and R. Varadarajan, "METRICS2.1 and flow tuning in the IEEE CEDA robust design flow and OpenROAD," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.
- [6] R. Liang, J. Jung, H. Xiang, L. Reddy, A. Lvov, J. Hu, and G.-J. Nam, "FlowTuner: A multi-stage EDA flow tuner exploiting parameter knowledge transfer," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.
- [7] J. Kwon, M. M. Ziegler, and L. P. Carloni, "A learning-based recommender system for autotuning design flows of industrial high-performance processors," in *ACM/IEEE Design Automation Conference (DAC)*, 2019.
- [8] Z. Xie, G.-Q. Fang, Y.-H. Huang, H. Ren, Y. Zhang, B. Khailany, S.-Y. Fang, J. Hu, Y. Chen, and E. C. Barboza, "FIST: A feature-importance sampling and tree-based method for automatic design flow parameter tuning," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2020.
- [9] H. Geng, T. Chen, Y. Ma, B. Zhu, and B. Yu, "Ptpt: physical design tool parameter tuning via multi-objective bayesian optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 1, pp. 178–189, 2022.
- [10] L. Donger, S. Qi, X. Qi, C. Tinghuan, and G. Hao, "Attention-based eda tool parameter explorer: From hybrid parameters to multi-qor metrics," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2024.
- [11] R. Moriconi, M. P. Deisenroth, and K. Sesh Kumar, "High-dimensional bayesian optimization using low-dimensional feature spaces," *Machine Learning*, vol. 109, pp. 1925–1943, 2020.
- [12] S. Daulton, X. Wan, D. Eriksson, M. Balandat, M. A. Osborne, and E. Bakshy, "Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization," in *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 35, 2022, pp. 12760–12774.
- [13] S. Zheng, H. Geng, C. Bai, B. Yu, and M. D. Wong, "Boosting vlsi design flow parameter tuning with random embedding and multi-objective trust-region bayesian optimization," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 28, no. 5, pp. 1–23, 2023.
- [14] P. Xu, S. Zheng, Y. Ye, C. Bai, S. Xu, H. Geng, T.-Y. Ho, and B. Yu, "Ranktuner: When design tool parameter tuning meets preference bayesian optimization," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2024, pp. 1–7.
- [15] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, vol. 1, no. 2, 2023.
- [16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019, pp. 4171–4186.
- [17] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.
- [18] G. Xun, X. Jia, V. Gopalakrishnan, and A. Zhang, "A survey on context learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 38–56, 2016.
- [19] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, "Automatic chemical design using a data-driven continuous representation of molecules," *ACS central science*, vol. 4, no. 2, pp. 268–276, 2018.
- [20] D. P. Kingma and M. Welling, "Auto-encoding variational {Bayes}," in *International Conference on Learning Representations (ICLR)*, 2024.
- [21] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations (ICLR)*, 2017.
- [22] P. Notin, J. M. Hernández-Lobato, and Y. Gal, "Improving black-box optimization in vae latent space using decoder uncertainty," in *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 34, 2021, pp. 802–814.
- [23] S. Stanton, W. Maddox, N. Gruver, P. Maffettone, E. Delaney, P. Green-side, and A. G. Wilson, "Accelerating bayesian optimization for biological sequence design with denoising autoencoders," in *International Conference on Machine Learning (ICML)*, 2022, pp. 20459–20478.
- [24] N. Gruver, S. Stanton, N. Frey, T. G. Rudner, I. Hotzel, J. Lafrance-Vanasse, A. Rajpal, K. Cho, and A. G. Wilson, "Protein design with guided discrete diffusion," in *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 36, 2023, pp. 12489–12517.
- [25] N. Maus, K. Wu, D. Eriksson, and J. R. Gardner, "Discovering many diverse solutions with bayesian optimization," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023.
- [26] S. Lee, J. Chu, S. Kim, J. Ko, and H. J. Kim, "Advancing bayesian optimization via learning correlated latent space," in *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 36, 2023, pp. 48906–48917.
- [27] S. Lee, J. Park, J. Chu, M. Yoon, and H. J. Kim, "Latent bayesian optimization via autoregressive normalizing flows," in *International Conference on Learning Representations (ICLR)*, 2015.
- [28] Q. Tang, J. Chen, B. Yu, Y. Lu, C. Fu, H. Yu, H. Lin, F. Huang, B. He, X. Han *et al.*, "Self-retrieval: Building an information retrieval system with one large language model," *arXiv e-prints*, pp. arXiv–2403, 2024.
- [29] X. Li, J. Jin, Y. Zhou, Y. Zhang, P. Zhang, Y. Zhu, and Z. Dou, "From matching to generation: A survey on generative information retrieval," *ACM Transactions on Information Systems (TOIS)*, vol. 43, no. 3, pp. 1–62, 2025.
- [30] Y. Pu, Z. He, T. Qiu, H. Wu, and B. Yu, "Customized retrieval augmented generation and benchmarking for eda tool documentation qa," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2024, pp. 1–9.
- [31] V. Karpukhin, B. Oguz, S. Min, P. S. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *The Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6769–6781.
- [32] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-pack: Packed resources for general chinese embeddings," in *ACM SIGIR Conference*, 2024, pp. 641–649.
- [33] X. Pei, T. Huang, Y. Ma, and C. Xu, "Rethinking causal mask attention for vision-language inference," *arXiv preprint arXiv:2505.18605*, 2025.
- [34] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Bayesian optimization approach for analog circuit synthesis using neural network," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 1463–1468.
- [35] M. Wistuba and J. Grabocka, "Few-shot bayesian optimization with deep kernel surrogates," *arXiv preprint arXiv:2101.07667*, 2021.
- [36] J. E. Stine, R. Ridley, and T.-D. Ene, "Osú datapath/control rv32 single-cycle and pipelined architecture in sv," 2021. [Online]. Available: <https://github.com/stineje/osu-riscv>
- [37] K. Asanovic, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelvitz *et al.*, "The rocket chip generator," *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17*, vol. 4, 2016.
- [38] D. Petrisko, F. Gilani, M. Wyse, D. C. Jung, S. Davidson, P. Gao, C. Zhao, Z. Azad, S. Canakci, B. Veluri, T. Guarino, A. Joshi, M. Os-kin, and M. B. Taylor, "BlackParrot: An agile open-source RISC-V multicore for accelerator SoCs," *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, vol. 40, no. 4, pp. 93–102, 2020.