McPAT-Calib: A RISC-V BOOM Microarchitecture Power Modeling Framework

Jianwang Zhai, Chen Bai, Binwu Zhu, Yici Cai, *Senior Member, IEEE*, Qiang Zhou, *Senior Member, IEEE*, and Bei Yu, *Member, IEEE*

***Abstract*—Power efficiency has become a nonneglected issue of modern CPUs. Therefore, accurate and robust power models are highly demanded in academia and industry. However, it is hard for existing power models to balance modeling speed, generality, and accuracy well. This article introduces McPAT-Calib, a microarchitecture power modeling framework, which combines McPAT with machine learning (ML) calibration and active learning (AL) sampling. McPAT-Calib can quickly and accurately estimate the power of different benchmarks executed on different CPU configurations, and provide an effective evaluation tool for the early design stage. First, McPAT-7nm is introduced to support the preliminary analytical power modeling for the 7-nm technology node. Then, a wide range of modeling features are identified, and automatic feature selection and advanced nonlinear regression are used to calibrate the McPAT-7nm modeling results, greatly improving the accuracy. Moreover, a novel AL approach termed power greedy sampling (PowerGS) embedded with domain knowledge is leveraged to reduce the modeling cost effectively. We use up to 15 configurations of the RISC-V Berkeley out-of-order machine (BOOM) along with 80 benchmarks, targeting 7-nm technology, to extensively evaluate McPAT-Calib. Compared with state-of-the-art (SOTA) microarchitecture power models, McPAT-Calib can reduce the mean absolute percentage error (MAPE) under different cross-validation (CV) strategies by 3.64%–6.14% (absolute reduction). Meanwhile, PowerGS is superior to the existing AL approaches, which can significantly reduce the demand for labeled samples to speed up model construction. The effectiveness of the overall modeling and estimation flow with AL sampling has also been verified.**

***Index Terms*—Active learning (AL), machine learning (ML) calibration, McPAT, microarchitecture power modeling, RISC-V Berkeley out-of-order machine (BOOM).**

I. Introduction

POWER modeling of integrated circuits (ICs) is a broad and lasting research topic [1]. With the slowdown of Moore's law and the breakdown of Dennard scaling, power consumption has become the main challenge in high power-efficiency CPU design. In industry, it is necessary to conduct an accurate power-performance tradeoff at the early design stage to ensure excellent CPU designs. However, the components of modern CPUs, e.g., instruction fetch, decoding units, and cache structures, are too complex to be accurately modeled. At the same time, the design space of modern CPUs is very large [2], and designers need to conduct extensive design space exploration (DSE) and optimization. When performing DSE, the entire design space, i.e., different design configurations along with different workloads, should be accurately modeled.

Increasingly stringent CPU power consumption design targets make power modeling a great challenge and put forward higher requirements for modeling speed, generality, and accuracy.

1) *Speed:* The running time of the power model itself and the overall time required for the entire modeling flow.
2) *Generality:* The ability to model the power of different CPU configurations under different workload programs.
3) *Accuracy:* Low error between the ground truths and predicted power values.

Commercial gate-level power analysis tools (e.g., PrimeTime PX) can accomplish the most accurate per-cycle power estimation with the flow shown as the dotted line in Fig. 1. In the later design stages, designers feed gate-level netlist into simulation and power analysis tools, usually taking hours or even days to complete. For the early stage of CPU design, i.e., microarchitecture level, this will be an unbearable high cost.

Academia has proposed a series of methods for different design stages to accelerate power modeling. The comparison is shown in Table I, illustrating the difficulty of simultaneously balancing the modeling speed, generality, and accuracy. At the register-transfer level (RTL) or gate level, power models (e.g., PRIMAL [4] and GRANNITE [3]) are constructed based on simulation traces using feature engineering and machine learning (ML) methods, enabling accurate fine-grained time-based modeling. However, these models require netlist design and simulation, which is highly time consuming, and most models are design specific and hard to estimate unknown

TABLE I
Comparison of Different Power Models

Model	Level	Speed	Generality	Accuracy	Type
PrimeTime PX	Gate	`Low`	High	High	Time-based
GRANNITE [3]	Gate	`Medium`	`Medium`	High	Time-based
PRIMAL [4]	RTL	`Medium`	`Medium`	High	Time-based
TCAD'17 [5]	Runtime	High	`Low`	High	Time-based
McPAT [6]	Arch	High	High	`Low`	Average
McPAT-Calib	Arch	High	High	High	Average

Manuscript received 5 December 2021; revised 5 March 2022 and 8 April 2022; accepted 12 April 2022. Date of publication 22 April 2022; date of current version 22 December 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61834002; in part by the Research Grants Council of Hong Kong under Grant CUHK24209017; and in part by HiSilicon. The preliminary version has been presented at the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) in 2021 [DOI: 10.1109/ICCAD51958.2021.9643508]. This article was recommended by Associate Editor C. Bolchini. *(Corresponding author: Yici Cai.)*

Jianwang Zhai, Yici Cai, and Qiang Zhou are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: zhaijw18@mails.tsinghua.edu.cn; caiyc@tsinghua.edu.cn).

Chen Bai, Binwu Zhu, and Bei Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, SAR.

Digital Object Identifier 10.1109/TCAD.2022.3169464

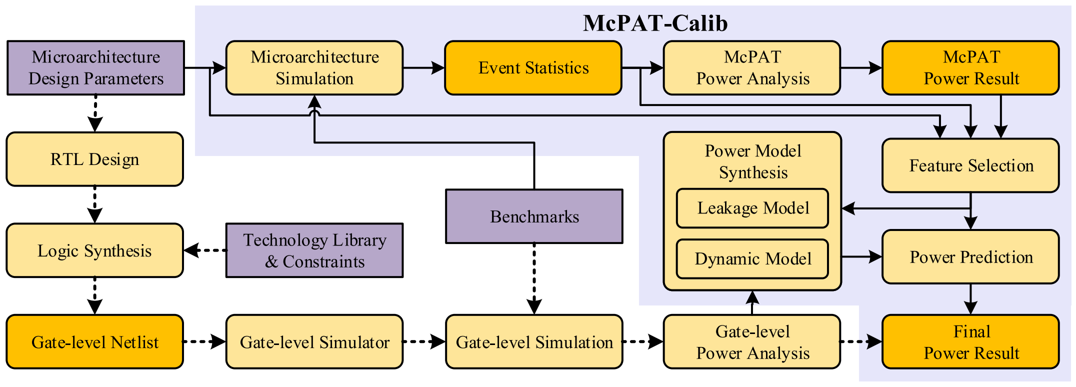

Fig. 1. Power modeling flow.

configurations well. At the microarchitecture level, analytical power models (e.g., McPAT [6]) use design parameters and machine activities for coarse-grained average power modeling, which have the fastest modeling speed and higher generality. Nonetheless, they cannot capture hardware details and lack accuracy guarantees for emerging CPUs [7]. Runtime power models (e.g., TCAD'17 [5]) based on performance monitoring counters (PMCs) can accomplish fine-grained average power modeling with equivalent microarchitecture events [8], but they are design specific and cannot support DSE.

The construction cost of power models is also an issue that we must consider. Analytical models require in-depth hardware analysis and a great effort to build the internal representation of the CPU, which is cumbersome and error prone. ML-based models are data-driven and need many labeled samples to train models with better fitting and generalization abilities. Consequently, the cost of labeling samples is very high.

We focus on power modeling at the microarchitecture level, and McPAT [6] serves as the cornerstone analytical power modeling tool. McPAT models the average power consumption following a hierarchical fashion, i.e., from a transistor level to a system architecture level. Predicted power values can be acquired from McPAT with microarchitecture design parameters and events, ignoring RTL implementations and simulations. Nevertheless, two limitations restrict the usage of McPAT: 1) the modeling accuracy is very low due to its incompleteness and architectural disparities in the model and 2) McPAT lacks support for advanced technology nodes, e.g., 7-nm FinFET technology.

In our preliminary work [9], we propose a framework called McPAT-Calib, which combines McPAT internal improvements and ML calibration to solve these problems. First, internal improvements are made to obtain McPAT-7nm, which supports the preliminary analytical modeling for modern CPUs under 7-nm FinFET technology. Second, advanced ML methods are leveraged to calibrate the McPAT-7nm modeling results, improving accuracy while ensuring generality. Moreover, a preclustering sequential AL sampling approach is leveraged to reduce the labeling cost. However, this preliminary AL sampling approach fails to embed domain knowledge to achieve better results further. In addition, academia lacks discussions on an overall power modeling and estimation flow combined with the sampling approach. In this work, we propose a novel power greedy sampling (PowerGS) approach embedded with domain knowledge and add it to the framework to reduce modeling costs better. McPAT-Calib aims to quickly model different benchmarks running on different CPU configurations at a lower cost, thus performing average power estimation.

Our main contributions are summarized as follows.

1) We obtain McPAT-7nm by introducing 7-nm FinFET technology and microarchitecture modification. It is the first step to implement McPAT-Calib.
2) We use ML methods to calibrate dynamic and leakage separately. A wide range of feature sources is selected for the dynamic calibration model, and the comparison of different feature sources proves the superiority.
3) We overcome multicollinearity through automatic feature selection and use a tree ensemble model to handle nonlinearity, ensuring generality and accuracy.
4) We propose a novel AL sampling approach, PowerGS, which embeds domain knowledge to enhance the diversity and representativeness of samples in multiple dimensions.
5) We propose an overall power modeling and estimation flow combined with AL sampling, which can complete model modeling at a lower cost.
6) We conduct an extensive evaluation of the proposed framework to prove its superiority, using up to 80 benchmarks along with 15 representative configurations of Berkeley out-of-order machine (BOOM) Core under 7-nm FinFET technology.

The remainder of this article is organized as follows. Section II introduces the important related work. Section III is an overview of RISC-V BOOM and problem formulation. Section IV provides detailed explanations about methodologies of the McPAT-Calib framework. Section V gives the overall power modeling and estimation flow with AL sampling. Section VI conducts an extensive evaluation. Finally, Section VII concludes this article.

II. Related Work

Academia has proposed various power modeling methods to avoid the high cost of commercial gate-level power analysis. According to whether RTL design is needed, they can be divided into two types: 1) microarchitecture level and 2) RTL (and gate) level.

A. Microarchitecture Power Models

At the microarchitecture level, power models only use limited microarchitecture design parameters and events for modeling average power at different granularities. Since there is no RTL implementation and simulation, the modeling speed

TABLE II
MICROARCHITECTURE DESIGN PARAMETERS AND POWER STATISTICS OF 15 BOOM CONFIGURATIONS

Parameters	SmallBoomConfig			MediumBoomConfig			LargeBoomConfig			MegaBoomConfig			GigaBoomConfig		
	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro
`FetchWidth`	4	4	4	4	4	8	8	8	8	8	8	8	8	8	8
`DecodeWidth`	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
`FetchBufferEntry`	5	8	16	8	16	24	18	24	30	24	32	40	30	35	40
`RobEntry`	16	32	48	64	64	80	81	96	114	112	128	136	125	130	140
`IntPhysRegister`	36	52	68	64	80	88	88	100	112	108	128	136	108	128	140
`FpPhysRegister`	36	48	56	56	64	72	88	96	112	108	128	136	108	128	140
`LDQ/STQEntry`	4	8	16	12	16	20	16	24	32	24	32	36	24	32	36
`BranchCount`	6	8	10	10	12	14	14	16	16	18	20	20	18	20	20
`MemIssue/FpIssueWidth`	1	1	1	1	1	1	1	1	2	1	2	2	2	2	2
`IntIssueWidth`	1	1	2	1	2	2	2	3	3	4	4	4	5	5	5
`DCache/ICacheWay`	2	4	8	4	4	8	8	8	8	8	8	8	8	8	8
`DCache/ICacheTLBEntry`	8	8	16	8	8	16	16	16	32	32	32	32	32	32	32
`DCacheMSHR`	2	2	4	2	2	4	4	4	4	4	4	8	8	8	8
`ICacheFetchBytes`	2	2	2	2	2	4	4	4	4	4	4	4	4	4	4
Min.Power(mW)	9.54	10.22	12.11	11.89	13.10	19.07	21.36	22.81	28.03	26.39	34.10	34.57	37.15	34.12	36.70
Max.Power(mW)	14.13	16.69	19.94	22.64	27.74	32.79	38.07	42.56	50.52	51.36	62.72	64.22	61.80	59.75	63.82
Avg.Power(mW)	11.76	13.53	15.64	16.42	17.94	24.60	28.02	30.02	35.97	36.55	44.06	45.52	45.62	43.26	46.38
Std.Power(mW)	1.22	1.70	1.73	2.81	3.95	3.76	4.62	5.00	5.56	6.06	7.27	7.84	6.00	6.64	7.10

is very fast. Analytical models, e.g., CACTI [10], Wattch [11], and McPAT [6], are mainly used. They try to establish the internal hardware representation and use event statistics from performance simulators [12], [13] to get the final estimate of power. McPAT integrates power, area, and timing modeling for multiprocessors. Due to the misalignment between the internal microarchitecture descriptions and real designs, the modeling error is as high as 20%–40% [7], [14], and there is a lack of support for advanced technology nodes. McPAT-PVT [15] and McPAT-Monolithic [16] update technology node, but the problem of insufficient accuracy is not solved. PowerTrain [14] uses linear regression (LR) with L1 regularization to reweight the power of each component from McPAT, but it lacks scalability for different designs.

In addition to analytical models, many ML-based models can also be used for microarchitecture power modeling. Earlier works [17], [18] use design parameters to perform regression modeling to achieve DSE, but it lacks event statistics and cannot accurately model different benchmarks. The event-based [19] and PMC-based [5], [20] models are more widely used. Jacobson *et al.* [19] used a relatively small number of program events to perform power modeling. Walker *et al.* [5] built a runtime power model that allows static and dynamic power separation by automatically selecting optimal PMC events. Sagi *et al.* [20] used nonlinear transformation to capture relations between PMC events and power values and use least-angle regression (LARS) to complete the multivariate polynomial power regression. Reddy *et al.* [8] transformed the PMC-based empirical model [5] into a microarchitecture power model. However, these machine activities are closely related to CPU configurations. As a result, these models are design specific and hard to make good predictions for new CPU designs.

B. *RTL and Gate-Level Power Models*

RTL and gate-level power models require netlist design and simulation, and are built based on signal proxies. PrEsto [21] uses linear models to characterize different modules through FPGA-acceleration. Yang *et al.* [22] used a feature selection technique based on singular value decomposition (SVD) to construct a linear power model. PRIMAL [4] uses a convolutional neural network (CNN) to process register switching activities to model the power of reusable circuit building blocks. Simmani [23] uses VCD dumps to construct a toggle-pattern matrix, and selects key signals through clustering to build the power model. GRANNITE [3] represents the gate netlist as a graph, and takes register states and unit inputs from RTL simulation as features to construct the graph neural network (GNN) model to predict gate toggle rates and average power. APOLLO [24] uses the minimax concave penalty (MCP)-based feature selection to select a small number of RTL signals to complete the per-cycle power modeling, which serves as the basis for both a design-time power estimator and a runtime on-chip power meter.

These RTL and gate-level power models can provide more hardware details and simulation traces and often achieve cycle-by-cycle accurate modeling. Nevertheless, these models have two critical shortcomings: 1) netlist design and simulation are needed, which is costly and slow, and is difficult to use at the early design stage and 2) most of them are design specific and challenging to adapt to different CPU configurations.

III. PRELIMINARIES

A. *RISC-V BOOM*

RISC-V is an open-source instruction set architecture (ISA) that has received strong attention and support from academia and industry. Because it can avoid expensive commercial licenses and can be customized according to various application scenarios, RISC-V has received plentiful progress [25]–[28].

BOOM [27], [29] utilizes Chisel [30] to construct a generator for the core, and is a family of out-of-order RISC-V designs rather than a single instance of a core. Due to the page limit, more details about BOOM can be found in [31]. Thanks to the parametric microarchitecture design, designers can obtain different BOOMs by configuring the core with different microarchitecture parameters, i.e., `FetchWidth`, `DecodeWidth`, etc., as shown in Table II. The divergent tradeoffs of BOOM between power and performance are needed to meet various design requirements. However, how to quickly and accurately model the power of different configurations is still a difficult problem. In addition, power is also closely related to the workload (i.e., benchmark) being executed.

B. Problem Formulation

Traditionally, the total power P of IC is modeled at the transistor level, which can be expressed as

$$P = P_{\text{dyn}} + P_{\text{leak}} = \alpha C V_{DD}^2 f + V_{DD} I_{\text{leakage}} \tag{1}$$

where P_{dyn} is the dynamic power and depends on: the activity factor α, the switching capacitance C, the supply voltage V_{DD}, and the frequency f. The leakage power P_{leak} depends on V_{DD} and the leakage current I_{leakage}.

However, modeling a CPU at the transistor level is very time consuming. More importantly, we cannot capture so many hardware details at the early design stage. To accelerate power modeling, we try to model power at a higher level of abstraction, i.e., at the microarchitecture level, which can be expressed as

$$\begin{aligned} P = P_{\text{dyn}} + P_{\text{leak}} &= f(\boldsymbol{e}, \boldsymbol{d}) + g(\boldsymbol{d}) \\ &= \sum_{n=1}^{N} \beta_n f_n(e_n, \boldsymbol{d}) + g(\boldsymbol{d}) \end{aligned} \tag{2}$$

where P_{dyn} is a function of the microarchitecture events $\boldsymbol{e}$ and the microarchitecture design parameters $\boldsymbol{d}$, e_n represents an event, β_n is the weighting factor, and N is the total number of events. P_{leak} is only determined by $\boldsymbol{d}$.

We give the definitions of "configuration" and "benchmark."

Definition 1 (Configuration): The configuration is to be defined as the CPU microarchitecture design characterized by a set of design parameters shown in Table II, such as `FetchWidth`, `DecodeWidth`, etc.

Definition 2 (Benchmark): The benchmark is to be defined as the workload program executed on the target CPU. Different benchmarks will result in different machine activities and power consumption.

The problem can be formulated with these definitions.

Problem 1 (Microarchitecture Power Modeling): Given different benchmarks $\mathcal{B}$ and different CPU configurations $\mathcal{C}$, the objective of microarchitecture power modeling is to estimate the average power P_{ij} of each benchmark $B_j \in \mathcal{B}$ running on each configuration $C_i \in \mathcal{C}$.

IV. McPAT-Calib Methodology

A. Overview of McPAT-Calib

The original McPAT cannot accurately model complex modern CPUs under advanced technology nodes. There are two ways to improve modeling accuracy. The first way is internal improvements. The hierarchical analytical modeling method allows researchers to make internal modifications to provide more accurate power estimates, but it is cumbersome and error-prone. The second way is calibration, but the calibration model obtained by the existing method [14] has poor generality and hardly provides good estimates for unknown CPU configurations. McPAT-Calib combines these two ways and aims to construct the power model with high generality and accuracy at a lower cost.

The McPAT-Calib framework consists of three important parts: 1) McPAT-7nm; 2) ML calibration; and 3) AL sampling. First, McPAT-7nm is used to complete the fast preliminary power modeling. McPAT-7nm is obtained through internal improvements to McPAT, including the introduction of 7-nm FinFET technology and microarchitecture modifications. Second, advanced ML calibration methods are used to calibrate the McPAT-7nm results to obtain more accurate power estimates. We identify a wide range of feature sources and propose an automatic feature selection algorithm to overcome the multicollinearity to ensure the generality, and use an advanced tree ensemble model to capture the nonlinearity in dynamic modeling. Moreover, PowerGS leverages prior knowledge of microarchitecture power modeling, to reduce the labeling cost in data acquisition, and is added to the overall modeling flow. Users can specify the number of labeled training samples according to the budget.

We use gem5 [13] to complete the microarchitecture simulation of RISC-V BOOM to obtain detailed event statistics. Notice that we do not restrict the usage of the simulator, i.e., we can also utilize Sniper [32] to acquire similar event statistics. The commercial flow shown in Fig. 1 is used as a golden flow to obtain the ground truth of power values. When the target microarchitecture configuration is given, RTL design and logic synthesis are required to obtain the gate-level netlist. Then, gate-level simulation and power analysis with the target benchmark are performed.

B. McPAT-7nm

To take advantage of the analytical power model's ease of use and readiness, we first introduce McPAT-7nm, which completes the preliminary power modeling of CPUs targeting 7 nm through simple internal improvements to McPAT. McPAT-7nm is the first part of the proposed framework and can also be used separately. We introduce 7-nm technology parameters and reduce modeling errors through microarchitecture modification and empirical coefficient adjustment. These improvements require small efforts to implement and prove effective in experiments.

Unlike ML-based models, the analytical power model depends on specific technology parameters. Due to the challenge of combining new processes, the original McPAT only supports 180 nm-22 nm CMOS technology modeling. McPAT-PVT [15] and McPAT-Monolithic [16] updated it to 22- and 14-nm FinFET technology through device simulations, respectively, but it still struggles to support more advanced technology nodes. Our target technology library is 7-nm FinFET PDK ASAP7 [33], with key parameters shown in Table III. We obtain some parameters through scaling and then get the physical parameters, e.g., voltage, current intensity, required for McPAT modeling, thus supporting 7-nm FinFET technology. In addition to technology parameters, McPAT uses some empirical undifferentiated core/FU coefficients. We adjust these empirical coefficients to reduce modeling errors in McPAT-7nm.

There is a misalignment between the RISC-V BOOM core and the microarchitecture description in the original McPAT. For example, the default pipeline in McPAT has at least 12 stages, which is different from BOOM, and may cause modeling errors. In addition, the default minimum cache size exceeds the minimum parameter that the BOOM core can support. Therefore, we make microarchitecture modifications to McPAT to support accurate modeling for the BOOM core.

C. Machine Learning Calibration

Due to the high abstraction level of McPAT, it is difficult to capture all hardware details of the CPU, which inevitably

TABLE III
KEY PARAMETERS OF ASAP7

FinFET parameters	Value
Supply voltage, V_{DD} (V)	0.7
Gate length, L_G (nm)	21
Fin height, H_{FIN} (nm)	32
Fin thickness, T_{SI} (nm)	6.5
Fin pitch, F_P (nm)	27
Contacted poly-pitch, CPP (nm)	54

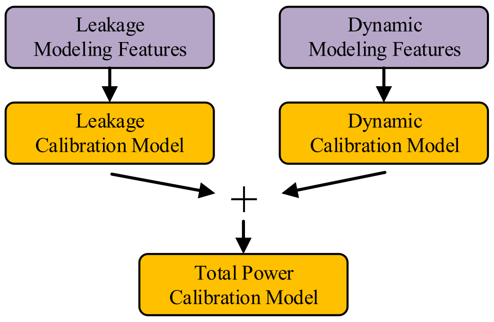
Leakage Modeling Features
Dynamic Modeling Features
Leakage Calibration Model
Dynamic Calibration Model
+
Total Power Calibration Model

Fig. 2. Power calibration model.

brings low modeling accuracy. One promising solution is calibrating McPAT with ML methodology against different CPU configurations. ML calibration helps McPAT compensate for the loss of modeling accuracy caused by missing or unsupported components inside a CPU. Compared with internal improvement, such methodology alleviates many workforce input.

We propose novel advanced ML methods to calibrate McPAT-7nm, which is the most important step of McPAT-Calib. To provide more power details, we choose to calibrate leakage power and dynamic power separately, and select appropriate modeling feature sources, respectively. A feature selection algorithm is proposed to automatically select the optimal features that can reflect different configurations and benchmarks. To handle the nonlinearity, an advanced tree ensemble model, XGBoost [34], is used to build the dynamic calibration model.

1) Calibration Method and Feature Source: As described in Section III-B, the total power is divided into dynamic power and leakage power, which can be calibrated separately, as shown in Fig. 2.

Leakage Calibration: For a certain CPU configuration, its leakage power is a fixed value (under a specific technology and certain operating corners). Therefore, we model the *leakage* of each configuration, i.e., the average leakage of all samples under the same configuration. This will cause the number of samples to be greatly reduced, so we only select two useful features for leakage calibration, namely, *Core.Leakage* and *Core.Area* obtained by McPAT-7nm. Obviously, the key lies in modeling the leakage of unknown configurations.

Dynamic Calibration: Dynamic power is closely related to the configuration and the executed benchmark. It is difficult to achieve high generality and accuracy by only using McPAT modeling results as features. Therefore, we choose to use McPAT-7nm modeling results, microarchitecture design parameters, and event statistics as dynamic modeling feature sources simultaneously. First, for McPAT-7nm results, we use the preliminary estimates of dynamic power for all levels

Algorithm 1 Filter Sequential Feature Selection

```
Require: allFeatures, all modeling features; k, the number
    of features to select; varThreshold, the variance threshold
    used to filter features;
Ensure: selectedList, the selected k optimal features;
 1: for tmpFeature in allFeatures do
 2:     if var(tmpFeature) $\le$ varThreshold then;
 3:         Delete tmpFeature from allFeatures;
 4:     end if
 5: end for
 6: selectedList $= \phi$;
 7: while selectedList.length $< k$ do
 8:     best$R^2 = -inf$;
 9:     for tmpFeature in allFeatures do
10:         Cross-Validation(selectedList + tmpFeature);
11:         if new$R^2 >$ best$R^2$ then;
12:             best$R^2 =$ new$R^2$;
13:             bestFeature = tmpFeature;
14:         end if
15:     end for
16:     Add bestFeature to selectedList;
17:     Delete bestFeature from allFeatures;
18: end while
```

of components. *Core.Area* and *Core.Leakage* are also leveraged, as they can provide overall information from different configurations. Then, all microarchitecture design parameters shown in Table II are utilized as the feature source, providing information on different configurations. Finally, we use microarchitecture events that reflect the working states of components, which are closely related to power consumption. Although gem5 can give thousands of events, we only selected 90 features according to architect experience, reflecting the critical activity of essential components that mainly contribute to the power consumption. All events were divided by *numCycles* for normalization.

Three feature sources are combined to formulate all features, and the correlation between features and dynamic power values is shown in Fig. 3. Section VI-D compares different feature sources. A small number of features will be further selected by Algorithm 1 described in Section IV-C2 to handle multicollinearity. The features highlighted in blue in Fig. 3 are the final selected features that contribute significantly to dynamic power calibration.

2) Automatic Feature Selection: An important consideration when performing regression is multicollinearity, i.e., the high linear correlation between modeling features. Many features with strong multicollinearity will lead to high model complexity, incurring low prediction performance, given unseen configurations or benchmarks. The reason is that the model may lead to overfitting on a small training set. We leverage the variance inflation factor (VIF) to quantify multicollinearity. To find the VIF of a feature, we can use other features to construct an ordinary least squares (OLSs) LR model to predict this feature. Then, we can get

$$\mathrm{VIF} = \frac{1}{1 - R^2} \tag{3}$$

$$R^2 = 1 - \frac{\sum_i^n \left(\hat{y}_i - y_i\right)^2}{\sum_i^n \left(y_i - \bar{y}\right)^2} \tag{4}$$

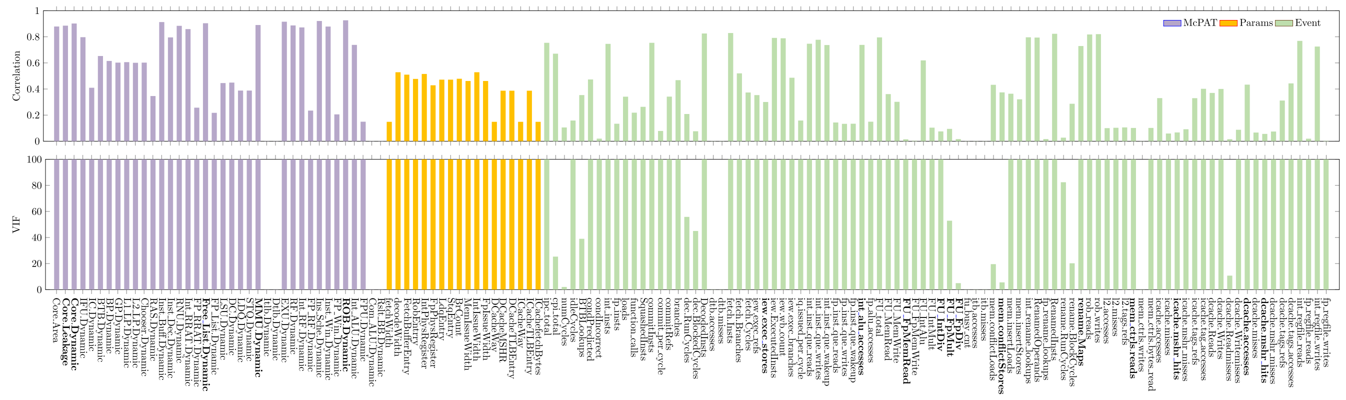

Fig. 3. Correlation and VIF of dynamic modeling features.

where y_i is the ground truth, $\hat{y}_i$ is the prediction, and $\bar{y} = (1/n)\sum_i^n y_i$ is the average of ground truth.

Generally speaking, strong multicollinearity is indicated when VIF > 10. As demonstrated in Fig. 3, we observe that the VIF of most dynamic modeling features exceeds 10, illustrating the existence of severe multicollinearity. We propose an automatic feature selection algorithm to solve this problem accordingly. Algorithm 1 first removes features with low variance (i.e., filters out these features that have little impact on modeling) to prune the search space. Then, Algorithm 1 performs forward sequential selection, sequentially adding the most valuable feature obtained by cross-validation (CV) [35]. To compare the modeling results when using the same number of features from different feature sources, we set the stopping criterion of Algorithm 1 to the number of selected features up to the specified "k" value. It is also possible to stop selection when the desired CV accuracy is reached or the accuracy cannot be improved further. The performance of Algorithm 1 is detailed in Section VI-E.

3) Regression Model: Previous work [20] shows that there is a strong nonlinearity in dynamic modeling. When complex benchmarks are executed on a CPU, the relationship between modeling features $\boldsymbol{x}$ and the resulting dynamic P_{dyn} is nonlinear.

LR models cannot capture this nonlinearity. To obtain the best modeling result, we use an advanced nonlinear model, XGBoost regressor (XGBR) [34], as the final dynamic calibration model. XGBoost is widely used to achieve state-of-the-art (SOTA) results on many ML challenges. It is a tree ensemble model that uses K additive functions $f_k(\boldsymbol{x})$ to predict the output

$$\hat{y}_i = \phi(\boldsymbol{x}_i) = \sum_{k=1}^{K} f_k(\boldsymbol{x}_i). \tag{5}$$

To learn the set of tree functions, the following regularization objective is minimized:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{6}$$

where l is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i. The second term Ω penalizes the complexity of the model, i.e., the regression tree functions, which helps to smooth the learned weights to avoid overfitting.

To evaluate the effectiveness of the proposed methods, we also compare XGBR with 14 other regression models, including: 1) LR; 2) LR with L1 regularization (Lasso); 3) LR with L2 regularization (Ridge); 4) LR with L1 and L2 regularization (ElasticNet); 5) Bayesian ridge regression (BRR); 6) Gaussian process regression (GPR); 7) regression based on K-nearest neighbors (KNNRs); 8) and 9) support vector regression with polynomial (Poly_SVR) and Radial-basis-function (RBF_SVR) kernel; 10) decision tree regressor (DTR); 11) random forest regressor (RFR); 12) adaboost regressor (ABR); 13) gradient boosting regressor (GBR); and 14) bagging regressor (BAGR).

D. AL Sampling With Domain Knowledge Embedded

For data-driven ML models, the more the training samples are, the better the modeling performance is. However, labeling samples (i.e., to get the ground truth of power value) require time-consuming gate-level simulation and power analysis. In our experiments, the golden flow for each sample takes approximately 5–20 h, which is an unacceptable cost. Hence, we use active learning (AL) [36] to alleviate this problem. AL aims to label the most representative training samples to sustain or increase the modeling performance at a lower labeling cost. To the best of our knowledge, our work is the first to apply AL to power modeling.

Contrary to obtaining the ground truth of power value, it is easy to obtain the features of a sample, which only takes a few seconds on average. Therefore, we propose a pool-based AL sampling method to reduce the labeling cost of the training set. Fig. 4 illustrates the pool-based AL cycle. After gem5 simulation and McPAT-7nm modeling, all samples with only features form an unlabeled sample pool. A learner selects a small number of initial samples and feeds them into the golden flow to label, thereby training an ML model, i.e., the power calibration model. Then, the learner selects the next sample to label and updates the ML model with the new knowledge of the newly labeled sample, and the cycle continues. At the end of the cycle, the target labeled training set is obtained.

In this work, we propose a novel AL sampling approach embedded with domain knowledge, termed PowerGS, to further reduce the construction cost of the power model. As

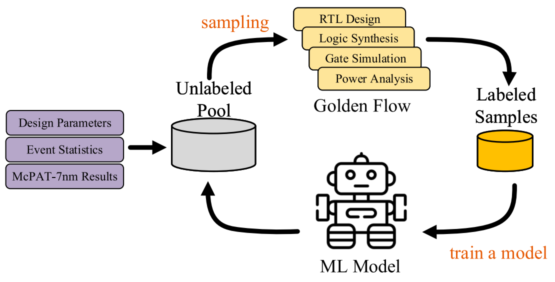

Fig. 4. Pool-based AL cycle.

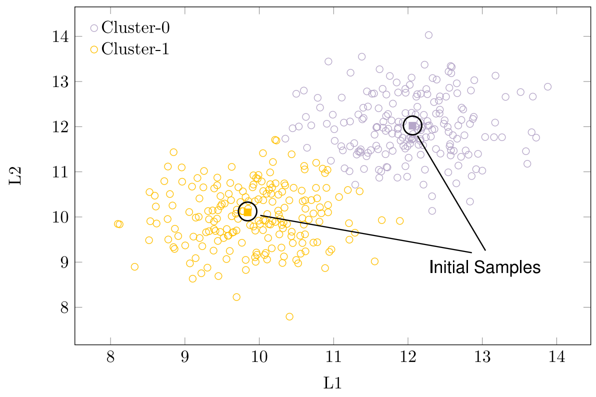

Fig. 5. Illustrative example of initial samples selection. The unlabeled samples are represented as points in a 2-D feature space, and the k-means ($k = 2$) clustering is performed. The sample closest to the center in each cluster is selected, and two initial samples are obtained.

shown in Fig. 6, we can incorporate PowerGS into the data acquisition stage of the overall power modeling flow, to build a high-performance power model under a limited budget.

PowerGS greedily selects diverse and representative samples to label, as shown in Algorithm 2. Suppose we have obtained all N samples as an unlabeled pool $\{\boldsymbol{x}_n\}_{n=1}^N$. First, Algorithm 2 selects representative initial samples $\{\boldsymbol{x}_n\}_{n=1}^d$ to query label $\{y_n\}_{n=1}^d$ through clustering, so that the sample query strategy can build a basic ML model. Then, the greedy sampling is performed iteratively. Each time the most useful sample $\boldsymbol{x}$ is selected to get the label y according to (11), and the newly labeled sample $(\boldsymbol{x}, y)$ is sequentially added to the training set. Finally, labeled samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^M$ are gained.

1) Initial Samples Selection: Some existing AL approaches [37], [38] usually select initial samples randomly, resulting in poor quality. We use a better initialization approach, preclustering, to ensure the diversity and representativeness of initial samples. We select d initial samples, where d is the dimension of modeling features. As shown in Fig. 5, the k-means ($k = d$) clustering on all N unlabeled samples are performed, and then the sample closest to the cluster center is selected from each cluster.

2) Sequential Greedy Sampling: After the initial selection, sequential greedy sampling is entered. In each iteration, assuming that k samples $\{\boldsymbol{x}_m\}_{m=1}^k$ have been labeled with outputs $\{y_m\}_{m=1}^k$, the sample query strategy selects the most useful sample to label from the remaining $N - k$ unlabeled samples $\{\boldsymbol{x}_n\}_{n=k+1}^N$. Our query strategy aims to enhance the diversity and representativeness of the labeled samples.

Our preliminary work [9] uses an advanced query strategy, iGS [39], to increase the diversity in both feature space and label space. First, the feature distance d_{nm}^x between each unlabeled sample $\boldsymbol{x}_n$ and each labeled sample $\boldsymbol{x}_m$ is computed. Then, a model $f(\boldsymbol{x})$ is built using the labeled samples, and the label distance d_{nm}^y between the prediction $f(\boldsymbol{x}_n)$ of each unlabeled sample $\boldsymbol{x}_n$ and the label y_m of each labeled sample $\boldsymbol{x}_m$ is computed. Finally, iGS computes $d_n^{xy} = \min_m d_{nm}^x d_{nm}^y$ of each unlabeled sample and selects the sample with the maximum d_n^{xy} to label. However, this strategy fails to utilize the prior knowledge of microarchitecture power modeling. Therefore, it struggles to achieve better results.

Algorithm 2 PowerGS

```
Require: $\mathcal{U}$, a set of unlabeled samples $\{\boldsymbol{x}_n\}_{n=1}^N$, where $\boldsymbol{x}_n \in \mathbb{R}^d$; $M$, the maximum number of samples to label;
Ensure: $\mathcal{L}$, the training set of labeled samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^M$; $f(\boldsymbol{x})$, the regression model;
1:  $\mathcal{L} = \phi$;
2:  Perform k-means clustering on $\mathcal{U}$ to obtain $d$ clusters, $\mathcal{C}_i, i = 1, ..., d$;
3:  for $i = 1 : d$ do
4:      Select the sample $\boldsymbol{x}$ closet to the center of $\mathcal{C}_i$ to label;
5:      Add $(\boldsymbol{x}, y)$ to $\mathcal{L}$, delete $\boldsymbol{x}$ from $\mathcal{U}$;
6:  end for
7:  Construct the regression model $f(\boldsymbol{x})$ using $\mathcal{L}$ ;
8:  for $i = d + 1 : M$ do
9:      Compute the $d_n^{cbxy}$ of each unlabeled sample $\boldsymbol{x}_n$ in $\mathcal{U}$ according to Equation (11);
10:     Select the sample $\boldsymbol{x}$ with the maximum $d^{cbxy}$ to label;
11:     Add $(\boldsymbol{x}, y)$ to $\mathcal{L}$, delete $\boldsymbol{x}$ from $\mathcal{U}$;
12:     Update the regression model $f(\boldsymbol{x})$ using $\mathcal{L}$ ;
13: end for
```

We now embed domain knowledge from microarchitecture power modeling to improve the strategy. After an in-depth analysis of power samples, we can find that in addition to the two dimensions of feature space and label space, the sample also has two other essential dimensions, i.e., the configuration and benchmark it belongs to. These two dimensions are of great significance for further enhancing the diversity of samples, which is also the point that distinguishes our approach from other general AL methods. Therefore, we propose a novel PowerGS approach, which simultaneously increases diversity in these four dimensions of power samples.

Like GSx and GSy in [39], PowerGS first calculates the feature distance d_n^x and the label distance d_n^y between the unlabeled sample $\boldsymbol{x}_n$ and all labeled samples

$$d_n^x = \min_m ||\boldsymbol{x}_n - \boldsymbol{x}_m||, \qquad m \in [1, k]; n \in [k+1, N] \quad (7)$$

$$d_n^y = \min_m ||f(\boldsymbol{x}_n) - y_m||, \qquad m \in [1, k]; n \in [k+1, N]. \quad (8)$$

On this basis, we consider the two dimensions of configuration and benchmark. To further enhance the diversity, we tend to select the power sample under a configuration (or benchmark) with a smaller number of labels. PowerGS uses the number of labeled samples to calculate the weight w_n^c of the configuration and the weight w_n^b of the benchmark

$$w_n^c = 1 - \frac{1}{1 + e^{-NC_n}}, \qquad n \in [k+1, N] \quad (9)$$

$$w_n^b = 1 - \frac{1}{1 + e^{-NB_n}}, \qquad n \in [k+1, N] \quad (10)$$

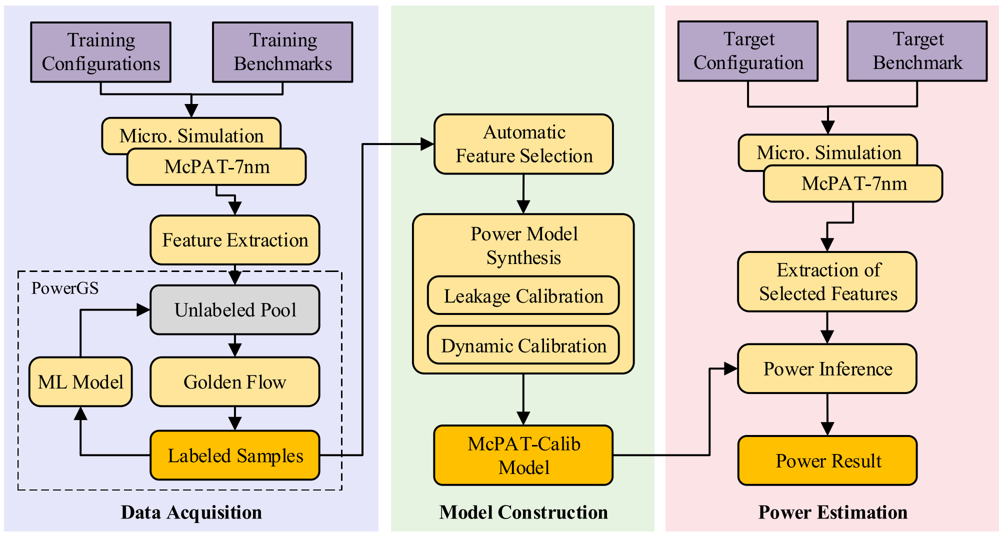

Fig. 6. Microarchitecture power modeling and estimation flow combined with AL sampling approach.

where NC_n is the labeled quantity in the configuration to which the sample $\boldsymbol{x}_n$ belongs; NB_n is the labeled quantity in the benchmark to which the sample $\boldsymbol{x}_n$ belongs.

Finally, PowerGS selects the sample with the maximum d_n^{cbxy} to label

$$d_n^{cbxy} = w_n^c w_n^b d_n^x d_n^y, \qquad n \in [k+1, N]. \tag{11}$$

3) Stop Criteria: To build a high-performance power model within a limited time budget, we can determine the number M of samples to be labeled according to the real budget. When the number of labeled samples reaches M, the sampling stops and all selected labeled samples are used to build the final power model $f(\boldsymbol{x})$. Other criteria can also be used, such as achieving the required CV accuracy.

V. Overall Modeling Flow

The preliminary work did not add AL sampling to the power modeling flow. To complete the microarchitecture power modeling at a lower cost, we propose a novel modeling and estimation flow combined with AL sampling, as illustrated in Fig. 6.

At the data acquisition stage, for all training configurations along with training benchmarks, first, gem5 is used for microarchitecture simulation, and McPAT-7nm is used for preliminary power modeling. Then, the leakage and dynamic modeling features from McPAT-7nm modeling results, event statistics, and microarchitecture design parameters are extracted and preprocessed, respectively, to form an unlabeled sample pool. Finally, the proposed AL sampling approach, PowerGS, sequentially selects samples and sends them to the commercial golden flow to label. The number of labeled samples is specified by the user according to the budget.

Model construction is performed after the labeled training samples are obtained. First, Algorithm 1 is used to complete automatic feature selection based on the training samples. Then, the construction of the leakage calibration model and the dynamic calibration model is achieved, respectively. Finally, the total power calibration model is formulated.

At the power estimation stage, gem5 simulation and preliminary McPAT-7nm modeling w.r.t. the given target configuration and benchmark are performed. Then, the selected modeling features are extracted. Finally, the built power calibration models are used to complete power estimation.

In addition, the use of McPAT-7nm is no different from the original McPAT, and can be directly replaced for integration into existing simulation toolchains like HotSniper [40]. However, the power calibration model is not suitable for application scenarios that require component-level power estimation and cannot be integrated into HotSniper.

VI. Evaluation

The McPAT-Calib framework is implemented in *Python* and executed on Intel(R) Core(TM) i9-9900k CPU@3.60 GHZ with 64-GB main memory. To verify the effectiveness and efficiency of our methods, we conduct a comprehensive evaluation. We first introduce the experimental settings in Section VI-A, and introduce the baseline methods to be compared in Section VI-B. In Sections VI-C–VI-F, the ablation studies of proposed modeling methods are performed. In Sections VI-G–VI-H, the AL sampling approach PowerGS and the McPAT-Calib framework are compared with baselines, respectively. In Section VI-I, the overall power modeling and estimation flow is evaluated.

A. Experiments Settings

We have expanded the official 5 RISC-V BOOM configurations, and obtained a total of 15 representative configurations for evaluation. Their microarchitecture design parameters are evenly distributed in the entire design space, and they are quite different from each other, enough to evaluate the generality of power models. The design parameters and power statistics of different configurations are shown in Table II.

To comprehensively evaluate the power consumption of CPUs under different workloads, we select up to 80 benchmarks as shown in Fig. 9. The sources are as follows: eight benchmarks from *riscv-tests* [41]; 19 representative isa from *riscv-tests* [41] by executing 200 loops to increase the ratio of effective instructions to prevent incorrect estimates; most of

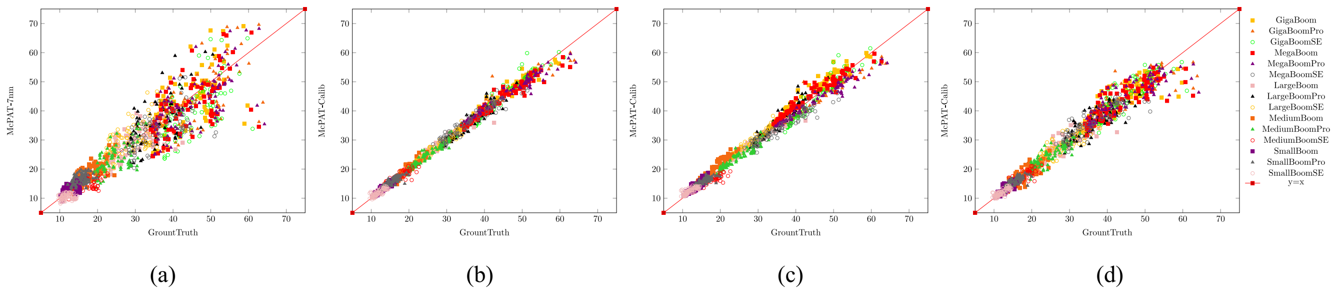

Fig. 7. Total power modeling results versus ground truth. (a) McPAT-7nm: preliminary modeling results. (b) McPAT-Calib: perform 15-fold Shuffle-Split CV. (c) McPAT-Calib: perform 15-fold Config-Split CV. (d) McPAT-Calib: perform 20-fold Bench-Split CV.

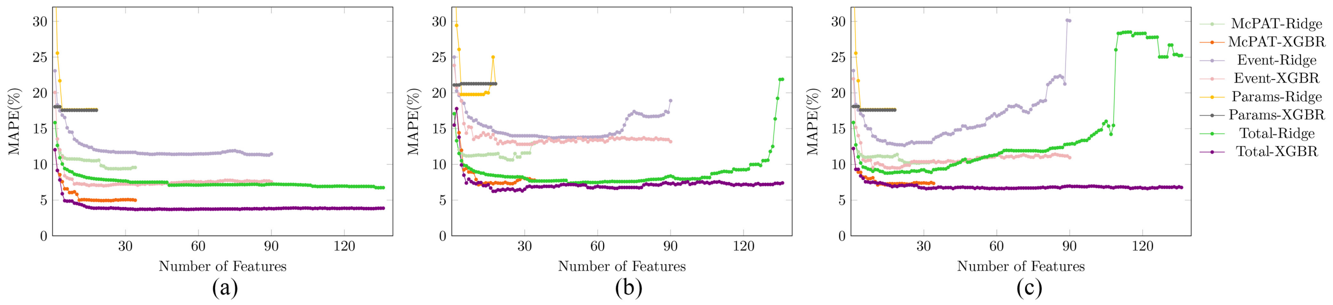

Fig. 8. Dynamic modeling results using different modeling features. (a) Shuffle-Split CV. (b) Config-Split CV. (c) Bench-Split CV.

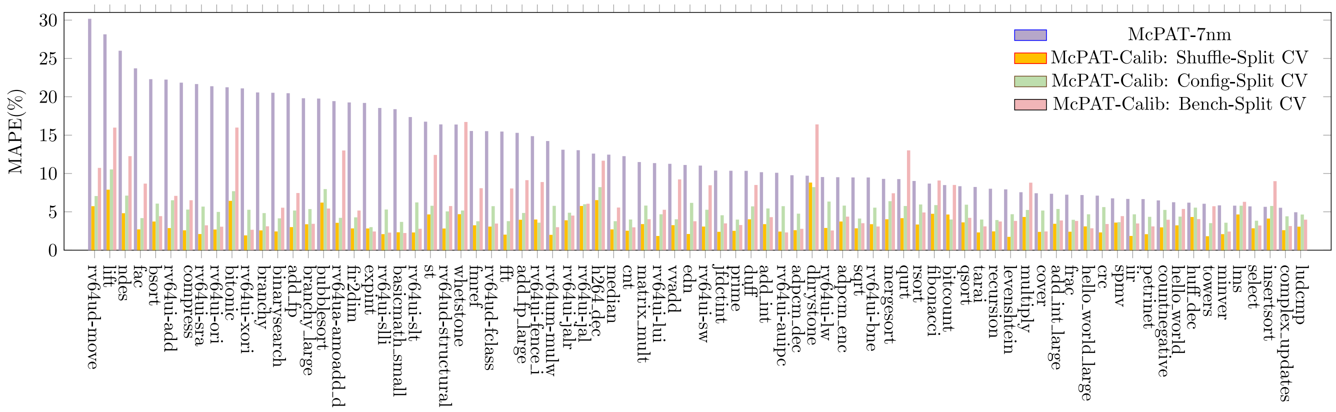

Fig. 9. Power modeling results of different benchmarks.

the rest 53 benchmarks are derived from previous open source projects [42], [43]. We make simple modifications to some benchmarks to enable them to complete RISC-V simulations.

We use commercial gate-level power analysis flow (Cadence Genus 18.12-e012_1 for logic synthesis, Synopsys VCS M-2017.03 for simulation @ 500 MHz, and PrimeTime PX R-2020.09-SP1 for power analysis) based on 7-nm FinFET PDK ASAP7 [33] to obtain the ground truth of power value. All 1200 power samples are obtained for offline evaluation, using 15 BOOM configurations along with 80 benchmarks.

Power modeling is essentially a regression problem. We use mean absolute percentage error (MAPE) and *coefficient of determination* (R^2) as two metrics to evaluate the accuracy of modeling results. MAPE is defined as

$$\text{MAPE} = \frac{1}{n}\sum_{i}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \times 100\% \quad (12)$$

where y_i is the ground truth of power value, and $\hat{y}_i$ is the power prediction. MAPE is a measure of the prediction accuracy of a modeling method. Low MAPE means low modeling error. R^2 has the same definition as (4), which is a measure that represents the proportion of the variance for a dependent variable that is explained by an independent variable or variables in a regression model. High R^2 means a high level of correlation.

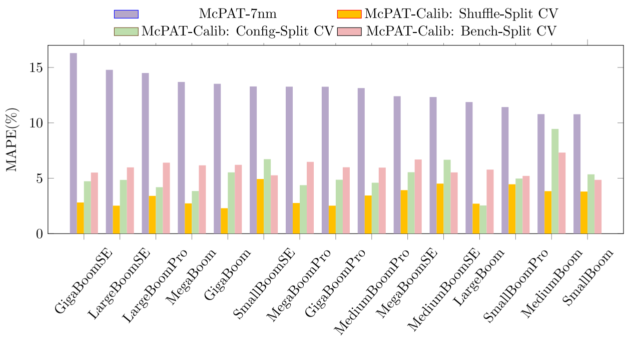

Fig. 10. Power modeling results of different configurations.

B. Baseline Methods

1) Microarchitecture Power Modeling: First, several representative microarchitecture power modeling baselines are compared with McPAT-Calib. The design parameter-based method [18] (HPCA'07) uses microarchitecture parameters to perform regression modeling. The event statistics-based method [5] (TCAD'17) builds a runtime power model that allows the separation of static and dynamic power by automatically selecting optimum PMC events. The event statistics-based method [20] (TCAD'20) uses nonlinear transformation to capture relations between PMC and power, and uses LARS to complete the multivariate polynomial regression. PowerTrain [14] uses LR with L1-norm penalty to reweight the results of each component from McPAT.

TCAD'17 and TCAD'20 are PMC-based runtime power models that average counter values that accumulate specific microarchitecture events. The events obtained by PMC are similar to those obtained by microarchitecture simulation, so PMC-based and simulation-based power models also have a lot in common, and there are many works [8], [14] showing the transformation between the two types of models. Therefore, we use TCAD17 and TCAD20 as microarchitecture design-time models to compare with McPAT-Calib.

2) Active Learning for Regression: Several classic AL approaches are compared with PowerGS. Query-by-committee (QBC) [37] first bootstraps the labeled samples to P copies with duplicates, and then builds P regression models (i.e., committee), and selects the sample with the maximum variance of the committee prediction to label. The Gaussian process (GP) uncertainty-based method [44] uses labeled samples to build a GP model, and selects the sample with the maximum prediction variance to label. GSx [45] considers the diversity in feature space, by computing the minimum distance d_n^x between an unlabeled sample $\boldsymbol{x}$ and all labeled samples, using all features, and selects the sample with the maximum d_n^x to label. GSy [39] considers the diversity in label space, by computing the minimum distance d_n^y between the estimated label and existing labels, and selects the sample with the maximum d_n^y to label. iGS [39] combines GSx and GSy to ensure that feature selection/weighting is considered, by computing the minimum product d_n^{xy} of feature distance and label distance, and selects the sample with the maximum d_n^{xy} to label.

C. McPAT-7nm Modeling Results

As an analytical power model, McPAT-7nm can be used directly without training. Therefore, it can directly estimate the power of all 1200 samples. The scatter diagram of McPAT-7nm modeling results and ground truth is shown in Fig. 7(a), where MAPE $= 13.02\%$ and $R^2 = 0.817$. After our improvement, McPAT-7nm realized the preliminary power modeling of 7-nm BOOM. In fact, McPAT-7nm can also be used separately from the framework.

D. Comparison of Dynamic Modeling Features

Different feature sources are compared to verify the superiority of the wide-ranging dynamic modeling features we have selected. Specifically, McPAT results (McPAT), event statistics (Event), design parameters (Params), and their total features (Total) are used for dynamic calibration, and Algorithm 1 is used to sequentially increase the number of features. The linear model (Ridge) and nonlinear model (XGBoost) are used to evaluate different feature sources.

Unlike analytical power models, data-driven ML methods need to use labeled samples to train the model and then make predictions. Therefore, we use CV [35] to perform the evaluation. More critically, three different CV strategies are proposed to evaluate the generality and accuracy of power models.

1) Shuffle-Split Cross-Validation (Shuffle-Split CV): First, regardless of the configuration or benchmark, the sample belongs to, all samples are treated equally. A 15-fold Shuffle-Split CV is performed to evaluate the power model's modeling ability for known configurations or benchmarks.

2) Modeling Unknown Configuration (Config-Split CV): To determine whether a power model can help the design of CPUs, the most important thing is to evaluate whether it can accurately model unknown CPU configurations. Therefore, a 15-fold Config-Split CV is performed to evaluate the modeling ability for unknown configurations. One previously invisible configuration is used as the testing set, and the samples under the remaining 14 known configurations are used as the training set, repeated 15 times, and averaged.

3) Modeling Unknown Benchmark (Bench-Split CV): Similar to modeling unknown configurations, the power model's ability to model unknown benchmarks is also a manifestation of its generality. Therefore, a 20-fold Bench-Split CV is performed. Four previously invisible benchmarks are used as the testing set, and the remaining 76 benchmarks are used as the training set, repeated 20 times, and averaged.

As shown in Fig. 8, we can see that when the number of features and the regression model are the same, using total modeling feature sources can achieve better results than using any single feature source. This is one of the important contributions of this work, because previous microarchitecture power models only use a single feature source. Fig. 8(b) and (c) also shows the negative effects of multicollinearity. When the number of features is too large, the modeling error for unknown configurations or benchmarks increases instead, especially for the linear model Ridge. This also proves the urgency of feature selection.

E. ML Calibration of Leakage and Dynamic

1) Leakage Calibration: Leakage power is only bound to the CPU configuration. Therefore, we mainly evaluate the modeling ability for unknown configurations. Specifically, the average leakage under each configuration is taken as the modeling target. We use 14 samples representing 14 configurations to train the leakage calibration model and predict another unknown configuration. The modeling results of each regression model are shown in column "Leakage" in Table IV, where the Ploy_SVR can achieve the best results with MAPE $=$ 4.47%.

2) Dynamic Calibration: To evaluate the generality of the dynamic calibration model, we perform a 15-fold Config-Split CV. More importantly, we compared the modeling results using total dynamic features and using the selected features after automatic feature selection. Table IV shows that all models can get better results after feature selection, which can effectively improve the generality of the power model.

Total Features: Column "Dynamic-Total F." lists the dynamic modeling results using total features. The advanced

TABLE IV
LEAKAGE AND DYNAMIC MODELING RESULTS

Regressors	*Leakage*	Dynamic-Total F.		Dynamic-Selected F.		
	MAPE(%)	MAPE(%)	R^2	k^*	MAPE(%)	R^2
LR	7.34	20.85	0.816	48	7.40	0.954
Lasso	8.08	17.97	0.869	48	7.55	0.951
Ridge	7.10	21.88	0.790	48	7.31	0.954
ElasticNet	6.77	16.36	0.889	22	9.20	0.929
BRR	7.74	18.50	0.867	48	7.30	0.954
GPR	7.72	16.29	0.895	15	9.32	0.924
KNNR	8.21	20.64	0.783	13	13.21	0.903
Poly_SVR	**4.47**	35.04	0.462	18	9.34	0.923
RBF_SVR	6.09	31.41	0.504	21	8.99	0.940
DTR	7.76	14.70	0.877	22	11.61	0.914
RFR	7.46	10.56	0.943	6	8.09	0.958
ABR	7.64	14.24	0.907	11	13.26	0.893
GBR	8.88	10.98	0.936	28	9.25	0.943
BAGR	7.59	11.41	0.931	6	9.92	0.933
XGBR	7.81	**7.40**	**0.961**	17	**6.23**	**0.969**

* k: The Number of Selected Features.

nonlinear model XGBR achieves the best modeling accuracy with MAPE = 7.40%, and MAPE of most linear models are up to 15%–20%. It can be seen that when total features are used for dynamic modeling, the results are not ideal. This is caused by the multicollinearity, which leads to the model overfitting on the training samples in known configurations and hard to model the samples in unknown configurations well.

Selected Features: As shown in column "Dynamic-Selected F.," after feature selection, fewer features are used to get better modeling results. The modeling accuracy of all regression models has been improved, especially for linear models. This verifies the effectiveness of the proposed feature selection algorithm, which automatically selects features that can reflect different configurations and benchmarks, and effectively overcomes the adverse effects of multicollinearity. The best modeling result is still achieved by XGBR, where MAPE = 6.23% using 17 selected features (with blue fonts in Fig. 3).

Of the 17 selected features, there are five features from McPAT-7nm results, indicating important CPU components; 12 features from event statistics, representing the most important machine activities for dynamic power calibration; and do not include features from design parameters because they cannot reflect the information of different benchmarks. We have observed that some components have a considerable impact on dynamic calibration in BOOM, e.g., reorder buffer (ROB), memory management unit (MMU), etc. Here, we give some insights using ROB as an example. ROB reserves entries for each in-light instruction executed in the pipeline. It interacts with different modules, e.g., issuing units, execution units, load and store units, etc., when the instruction's running status is changed (hang on or execution). Therefore, the events relative to ROB can represent the current running workloads of the processor, and its dynamic power is implicitly indicative of the overall dynamic power consumption. Accordingly, *ROB.Dynamic* obtained by McPAT-7nm was incorporated into our dynamic calibration model.

F. ML Calibration of Total Power

According to the results in Section VI-E, we use Ploy_SVR to construct the leakage calibration model and use XGBR to construct the dynamic calibration model with 17 selected features. The sum of leakage and dynamic is the estimate of total power. Three CV strategies are performed on all 1200 samples. The modeling results of 15-fold Shuffle-Split CV are shown in Fig. 7(b), where MAPE = 3.38%, R^2 = 0.989. The modeling results of 15-fold Config-Split CV are shown in Fig. 7(c), where MAPE = 5.22%, R^2 = 0.978. The modeling results of 20-fold Bench-Split CV are shown in Fig. 7(d), where MAPE = 5.96%, R^2 = 0.958. Figs. 9 and 10 show the modeling results on different CPU configurations and benchmarks. As shown in these figures, under different evaluation strategies, the modeling accuracy of McPAT-Calib is much higher than McPAT-7nm, which proves the effectiveness of the proposed ML calibration methods.

The test sets for the three CV strategies described in Section VI-D are small due to the small dataset available. To further verify the generality of McPAT-Calib, we conduct supplementary experiments with more test sets. First, we perform a 5-fold Config-Split CV, where three subarchitectures belong to the same main architecture as the test set. Then, We perform 5-fold and 10-fold Bench-Split CV, i.e., 16 or 8 benchmarks as the test set. Finally, we perform 5-fold, 10-fold, and 20-fold Shuffle-Split CV. The results are shown in Table VII with MAPE remaining at comparable levels.

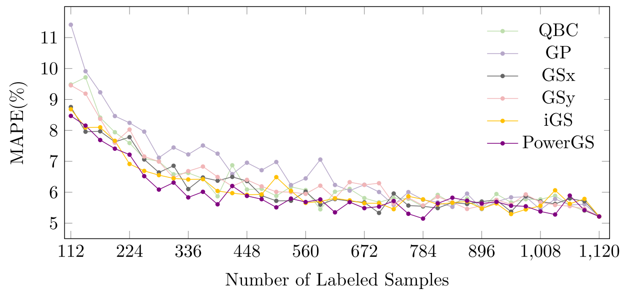

Fig. 11. Test MAPE curve of different AL approaches.

G. Comparison With Previous AL Approaches

The effectiveness of AL sampling is verified, and PowerGS is compared with the five previous AL approaches described in Section VI-B2. A 15-fold Config-Split CV is performed for the total power modeling of all 1200 samples, where the 17 selected features are used for dynamic calibration. In each validation process, the AL approach selects different percentages of useful labeled samples from 1120 training samples, thereby constructing the power model and predicting 80 test samples.

Fig. 11 shows the variation of the test MAPE with the number of labeled samples. The test MAPE presents a fluctuating decrease trend as the number of labeled samples increases. When the sampling percentage reaches 70%, i.e., 784 labeled training samples are selected, the test MAPE curve tends to converge. To better compare the convergence speed of different AL approaches, Table V lists the modeling results under different sampling percentages before 70%. PowerGS achieves the smallest test MAPE under most sampling percentages, with an average MAPE of 6.13% and an average ranking of 1.38, which is better than previous AL approaches.

The results show that AL sampling can effectively reduce the demand for labeled training samples, and high modeling accuracy can be achieved when only a limited number of labeled samples are used. In the meantime, our new sampling

TABLE V
COMPARISON BETWEEN POWERGS AND PREVIOUS AL APPROACHES

Method	Sampling Percentage - MAPE(%)													Average	
	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	MAPE(%)	Rank
QBC [37]	9.48	8.41	7.59	7.00	6.62	5.87	6.09	5.88	6.07	6.02	5.81	5.81	5.54	6.63	3.92
GP [44]	11.41	9.23	8.24	7.12	7.22	7.25	6.95	6.98	6.45	6.24	6.24	5.59	5.76	7.28	5.62
GSx [45]	8.75	7.97	7.78	6.64	6.10	6.37	6.36	5.72	6.02	5.78	5.67	5.96	5.55	6.51	3.23
GSy [39]	9.46	8.37	8.03	6.99	6.68	6.49	6.40	6.01	5.95	5.80	6.24	5.63	5.57	6.74	4.15
iGS [39]	8.68	8.09	**6.91**	6.55	6.41	6.04	5.91	6.49	**5.66**	5.80	5.65	**5.45**	5.77	6.42	2.69
PowerGS	**8.47**	**7.69**	7.22	**6.09**	**5.83**	**5.61**	**5.88**	**5.51**	5.68	**5.35**	**5.49**	5.72	**5.15**	**6.13**	**1.38**

TABLE VI
COMPARISON WITH PREVIOUS POWER MODELS

Method	Shuffle-Split CV			Config-Split CV			Bench-Split CV		
	MAPE(%)	R^2	Norm. Cost	MAPE(%)	R^2	Norm. Cost	MAPE(%)	R^2	Norm. Cost
HPCA'07 [18]	15.31	0.807	1.00	18.37	0.752	1.00	15.34	0.807	1.00
TCAD'17 [5]	11.71	0.899	1.00	14.31	0.875	1.00	13.56	0.842	1.00
TCAD'20 [20]	22.51	0.746	1.00	24.58	0.711	1.00	23.92	0.690	1.00
PowerTrain [14]	9.33	0.926	1.00	11.36	0.906	1.00	9.60	0.921	1.00
McPAT-7nm [9]	13.02	0.817	-	13.02	0.817	-	13.02	0.817	-
McPAT-Calib [9]	**3.38**	**0.989**	1.00	5.22	0.978	1.00	**5.96**	0.958	1.00
McPAT-CalibAL	3.97	0.986	**0.70**	**5.15**	**0.979**	**0.70**	6.05	**0.959**	**0.70**

TABLE VII
RESULTS OF DIFFERENT CV STRATEGIES

MAPE	Shuffle-Split CV	Config-Split CV	Bench-Split CV
5-fold	3.61%	6.95%	6.09%
10-fold	3.50%	-	6.09%
15-fold	3.38%	5.22%	-
20-fold	3.32%	-	5.96%

approach for the field of microarchitecture power modeling, PowerGS, can achieve better modeling results than previous AL approaches, including iGS used in the preliminary work.

H. Comparison With Previous Power Models

The modeling methods of our framework are compared with previous microarchitecture power models described in Section VI-B1. The total power calibration model is implemented in the way described in Section VI-F. The three CV strategies are performed on all 1200 samples, and the results are shown in Table VI. Column "Norm. Cost" denotes the normalized modeling cost, which is measured by the number of labeled samples used to construct the power model. McPAT-7nm is an analytical power model without training. McPAT-Calib with AL sampling (i.e., McPAT-CalibAL) uses PowerGS to select part of labeled samples, and other models use all labeled samples.

Compared with the SOTA results, the preliminary McPAT-Calib [9] can reduce the MAPE of shuffle-split CV by 5.95%. More importantly, it reduces the MAPE of Config-Split CV (estimating unknown CPU configuration) by 6.14%, and the MAPE of Bench-Split CV (estimating unknown benchmark) by 3.64%. The R^2 of the three strategies has also been improved. Meanwhile, McPAT-CalibAL can achieve comparable or even better modeling results when reducing the modeling cost by 30%. This shows that AL sampling can effectively reduce the modeling cost and even improve the modeling accuracy, because fewer training samples help to avoid overfitting. Since TCAD'17 and TCAD'20 are originally low-overhead runtime power models, there may be some accuracy loss when used for microarchitecture power modeling.

I. Evaluation of Overall Modeling Flow

Finally, we evaluate the overall power modeling and estimation flow combined with AL sampling and compare it with the preliminary McPAT-Calib [9].

All 1200 samples with total features are divided into training and test sets using three CV strategies. The data acquisition and model construction are performed on the training set. First, the AL sampling approach is used to select a certain percentage of labeled samples from the training set using total features. Then, automatic feature selection is performed. Finally, the labeled samples are used to construct calibration models using the selected features. The power estimation is performed on the testing set, and the calibration models are used to estimate the power of the testing samples. Since the number of total features exceeds 10% of the training set, the sampling percentages are set from 20% to 90%.

As shown in Table VIII, under different CV strategies, the average MAPE of McPAT-CalibAL using PowerGS is only 94.1%, 95.0%, and 95.6% of the preliminary McPAT-Calib [9], while the running time (column "RT") increases by only 5.0%, 2.0%, and 3.6%. Meanwhile, three R^2 have also been improved. The evaluation is performed offline and "RT" does not include the execution time of the golden flow. We can see that the modeling accuracy (MAPE) under the three CV strategies is comparable. However, the R^2 of Config-Split CV is low, which means that the fitting ability to the variance of different samples under unknown configurations is relatively low. The effectiveness of the new power modeling flow using

TABLE VIII
COMPARISON WITH PRELIMINARY WORK

Percentage	McPAT-Calib [9]									McPAT-CalibAL								
	Shuffle-Split CV			Config-Split CV			Bench-Split CV			Shuffle-Split CV			Config-Split CV			Bench-Split CV		
	MAPE (%)	R^2	RT (s)	MAPE (%)	R^2	RT (s)	MAPE (%)	R^2	RT (s)	MAPE (%)	R^2	RT (s)	MAPE (%)	R^2	RT (s)	MAPE (%)	R^2	RT (s)
20%	7.18	0.950	83	9.10	0.334	84	7.93	0.936	88	6.33	0.962	85	7.63	0.538	86	6.89	0.951	89
30%	5.77	0.966	225	7.10	0.629	229	7.35	0.944	236	5.41	0.972	237	6.87	0.612	235	6.84	0.950	241
40%	5.21	0.973	415	7.19	0.528	421	6.69	0.950	432	4.89	0.976	439	6.64	0.587	434	6.74	0.948	448
50%	4.71	0.977	655	7.67	0.466	665	6.58	0.950	681	4.51	0.980	694	7.45	0.488	684	6.55	0.952	711
60%	4.59	0.979	948	6.73	0.616	960	6.66	0.947	982	4.29	0.981	996	5.90	0.623	978	5.96	0.958	1019
70%	4.25	0.983	1282	6.37	0.653	1298	6.11	0.955	1326	4.13	0.983	1343	5.83	0.749	1320	6.60	0.949	1372
80%	4.00	0.985	1654	6.42	0.665	1670	6.45	0.950	1705	4.01	0.983	1726	7.21	0.569	1700	6.09	0.953	1765
90%	3.87	0.986	2056	6.45	0.617	2075	6.28	0.952	2200	3.83	0.985	2140	6.66	0.522	2110	6.10	0.952	2196
Average	4.95	0.975	915	7.13	0.564	925	6.77	0.948	946	4.66	0.978	958	6.77	0.586	943	6.47	0.952	980
Ratio	1.000	1.000	**1.000**	1.000	1.000	**1.000**	1.000	1.000	**1.000**	**0.941**	**1.003**	1.050	**0.950**	**1.029**	1.020	**0.956**	**1.004**	1.036

PowerGS is proved, and better modeling results are achieved with little additional running time overhead.

Finally, we give the absolute execution times of McPAT-Calib and the golden flow. For 1200 samples, the average execution time for label acquisition using the golden flow was 7.2 h, while the average inference time for McPAT-Calib was 3.46 s.

VII. CONCLUSION AND FUTURE WORK

We propose a microarchitecture power modeling framework named McPAT-Calib that combines McPAT-7nm, ML calibration, and AL sampling. First, we made internal improvements to obtain McPAT-7nm to support preliminary analytical power modeling of 7-nm CPUs. Then, ML calibration methods, such as automatic feature selection and advanced nonlinear regression, are used to improve the accuracy of McPAT-7nm. Finally, a novel AL sampling algorithm PowerGS embedded with domain knowledge is proposed to effectively reduce the construction cost of the ML calibration model. McPAT implements an overall power modeling and estimation flow and aims to construct the power model with high generality and accuracy at a lower cost. Extensive evaluations based on the RISC-V BOOM show the superiority of McPAT-Calib, which will effectively promote the modeling and design of modern CPUs.

McPAT-Calib provides a power modeling flow and methodology, rather than a ready-to-use power calibration model. It trains ML calibration models using several typical configurations of the core to provide power estimates for other configurations to support DSE. Therefore, ML models trained on BOOM cores may not be accurate enough for other cores because the microarchitectures are very different and retraining is necessary.

Our future research is focused on three directions. First, we will develop a more fine-grained time-based power model for design-time or runtime power estimation. Second, we want McPAT-Calib to provide component-level power estimation and integrate it into existing toolchains like HotSniper. Third, we will investigate the transferability of the ML calibration models.

ACKNOWLEDGMENT

The authors thank Wenlong Lyu and Zhitang Chen from Huawei Noah's Ark Lab, Yuzhe Ma from HKUST(GZ), for their excellent critique and feedback, without which this work would not have been possible.

REFERENCES

[1] Y. Nasser, J. Lorandel, J.-C. Prévotet, and M. Hélard, "RTL to transistor level power Modeling and estimation techniques for FPGA and ASIC: A survey," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 3, pp. 479–493, Mar. 2021.

[2] C. Bai, Q. Sun, J. Zhai, Y. Ma, B. Yu, and M. D. F. Wong, "BOOM-explorer: RISC-V BOOM Microarchitecture design space exploration framework," in *Proc. ICCAD*, 2021, pp. 1–9.

[3] Y. Zhang, H. Ren, and B. Khailany, "GRANNITE: Graph neural network inference for transferable power estimation," in *Proc. DAC*, 2020, pp. 1–6.

[4] Y. Zhou, H. Ren, Y. Zhang, B. Keller, B. Khailany, and Z. Zhang, "PRIMAL: Power inference using machine learning," in *Proc. DAC*, 2019, pp. 1–6.

[5] M. J. Walker *et al.*, "Accurate and stable run-time power modeling for mobile and embedded CPUs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 1, pp. 106–119, Jan. 2017.

[6] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Proc. MICRO*, 2009, pp. 469–480.

[7] S. L. Xi, H. Jacobson, P. Bose, G.-Y. Wei, and D. Brooks, "Quantifying sources of error in McPAT and potential impacts on architectural studies," in *Proc. HPCA*, 2015, pp. 577–589.

[8] B. K. Reddy, M. J. Walker, D. Balsamo, S. Diestelhorst, B. M. Al-Hashimi, and G. V. Merrett, "Empirical CPU power modelling and estimation in the gem5 simulator," in *Proc. PATMOS*, 2017, pp. 1–8.

[9] J. Zhai, C. Bai, B. Zhu, Y. Cai, Q. Zhou, and B. Yu, "McPAT-Calib: A Microarchitecture power Modeling framework for modern CPUs," in *Proc. ICCAD*, 2021, pp. 1–9.

[10] S. Li, K. Chen, J. H. Ahn, J. B. Brockman, and N. P. Jouppi, "CACTI-P: Architecture-level modeling for SRAM-based structures with advanced leakage reduction techniques," in *Proc. ICCAD*, 2011, pp. 694–701.

[11] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proc. ISCA*, 2000, pp. 83–94.

[12] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, "The M5 simulator: Modeling networked systems," *IEEE Micro*, vol. 26, no. 1, pp. 52–60, Jul./Aug. 2006.

[13] A. Roelke and M. R. Stan, "RISC5: Implementing the RISC-V ISA in gem5," in *Proc. 1st Workshop Comput. Archit. Res. RISC-V (CARRV)*, 2017, p. 7.

[14] W. Lee, Y. Kim, J. H. Ryoo, D. Sunwoo, A. Gerstlauer, and L. K. John, "PowerTrain: A learning-based calibration of McPAT power models," in *Proc. ISLPED*, 2015, pp. 189–194.

[15] A. Tang, Y. Yang, C.-Y. Lee, and N. K. Jha, "McPAT-PVT: Delay and power modeling framework for FinFET processor architectures under PVT variations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 9, pp. 1616–1627, Sep. 2015.

[16] A. Guler and N. K. Jha, "McPAT-monolithic: An area/power/timing architecture modeling framework for 3-D hybrid monolithic multicore systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 10, pp. 2146–2156, Oct. 2020.

[17] B. C. Lee and D. M. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction," *ACM SIGOPS Oper. Syst. Rev.*, vol. 40, pp. 185–194, Dec. 2006.

[18] B. C. Lee and D. M. Brooks, "Illustrative design space studies with microarchitectural regression models," in *Proc. HPCA*, 2007, pp. 340–351.

[19] H. Jacobson, A. Buyuktosunoglu, P. Bose, E. Acar, and R. Eickemeyer, “Abstraction and microarchitecture scaling in early-stage power modeling,” in *Proc. HPCA*, 2011, pp. 394–405.

[20] M. Sagi, N. A. V. Doan, M. Rapp, T. Wild, J. Henkel, and A. Herkersdorf, “A lightweight nonlinear methodology to accurately model multicore processor power,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 39, no. 11, pp. 3152–3164, Nov. 2020.

[21] D. Sunwoo, G. Y. Wu, N. A. Patil, and D. Chiou, “PrEsto: An FPGA-accelerated power estimation methodology for complex systems,” in *Proc. FPL*, 2010, pp. 310–317.

[22] J. Yang, L. Ma, K. Zhao, Y. Cai, and T.-F. Ngai, “Early stage real-time SoC power estimation using RTL instrumentation,” in *Proc. ASPDAC*, 2015, pp. 779–784.

[23] D. Kim, J. Zhao, J. Bachrach, and K. Asanović, “Simmani: Runtime power modeling for arbitrary RTL with automatic signal selection,” in *Proc. MICRO*, 2019, pp. 1050–1062.

[24] Z. Xie *et al.*, “APOLLO: An automated power modeling framework for runtime power introspection in high-volume commercial microprocessors,” in *Proc. MICRO*, 2021, pp. 1–14.

[25] K. Asanović *et al.*, “The rocket chip generator,” Dept. Electr. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Rep. UCB/EECS-2016-17, 2016.

[26] F. Zaruba and L. Benini, “The cost of application-class processing: Energy and performance analysis of a Linux-ready 1.7-GHz 64-bit RISC-V core in 22-nm FDSOI technology,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 11, pp. 2629–2640, Nov. 2019.

[27] K. Asanovic, D. A. Patterson, and C. Celio, “The Berkeley out-of-order machine (BOOM): An industry-competitive, synthesizable, parameterized RISC-V processor,” Dept. Electr. Eng. Comput. Sci., Univ. California Berkeley, Berkeley, CA, USA, Rep. UCB/EECS-2015-167, 2015.

[28] C. Chen *et al.*, “Xuantie-910: A commercial multi-core 12-stage pipeline out-of-order 64-bit high performance RISC-V processor with vector extension: Industrial product,” in *Proc. ISCA*, 2020, pp. 52–64.

[29] J. Zhao, B. Korpan, A. Gonzalez, and K. Asanovic, “SonicBOOM: The 3rd generation Berkeley out-of-order machine,” in *Proc. 4th Workshop Comput. Archit. Res. RISC-V*, 2020, pp. 1–7.

[30] J. Bachrach *et al.*, “Chisel: Constructing hardware in a scala embedded language,” in *Proc. DAC*, 2012, pp. 1212–1221.

[31] “RISC-V BOOM.” [Online]. Available: https://boom-core.org (Accessed: Mar. 2021).

[32] T. E. Carlson, W. Heirman, S. Eyerman, I. Hur, and L. Eeckhout, “An evaluation of high-level mechanistic core models,” *ACM Trans. Archit. Code Optim.*, vol. 11, no. 3, pp. 1–25, 2014.

[33] L. T. Clark *et al.*, “ASAP7: A 7-nm finFET predictive process design kit,” *Microelectron. J.*, vol. 53, pp. 105–115, Jul. 2016.

[34] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. SIGKDD*, 2016, pp. 785–794.

[35] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proc. IJCAI*, 1995, pp. 1137–1143.

[36] B. Settles, “Active learning literature survey,” Dept. Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, Rep. TR1648, 2009.

[37] T. RayChaudhuri and L. G. C. Hamey, “Minimisation of data collection by active learning,” in *Proc. Int. Conf. Neural Netw. (ICNN)*, 1995, pp. 1338–1341.

[38] W. Cai, Y. Zhang, and J. Zhou, “Maximizing expected model change for active learning in regression,” in *Proc. ICDM*, 2013, pp. 51–60.

[39] D. Wu, C.-T. Lin, and J. Huang, “Active learning for regression using greedy sampling,” *Inf. Sci.*, vol. 474, pp. 90–105, Feb. 2019.

[40] A. Pathania and J. Henkel, “HotSniper: Sniper-based toolchain for many-core thermal simulations in open systems,” *IEEE Embedded Syst. Lett.*, vol. 11, no. 2, pp. 54–57, Jun. 2019.

[41] “riscv-tests.” [Online]. Available: https://github.com/riscv/riscv-tests (Accessed: Mar. 2021).

[42] “About TACLe Benchmarks.” [Online]. Available: https://github.com/mcmenaminadrian/taclebench-riscv-baremetal (Accessed: Mar. 2021).

[43] J. Pallister, S. J. Hollis, and J. Bennett, “BEEBS: Open benchmarks for energy measurements on embedded platforms,” 2013, *arXiv:1308.5174.*

[44] Y. Ma, S. Roy, J. Miao, J. Chen, and B. Yu, “Cross-layer optimization for high speed adders: A Pareto driven machine learning approach,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2298–2311, Dec. 2019.

[45] H. Yu and S. Kim, “Passive sampling for regression,” in *Proc. ICDM*, 2010, pp. 1151–1156.

Jianwang Zhai received the B.E. degree in communication engineering from Beijing Jiaotong University, Beijing, China, in 2018. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Tsinghua University, Beijing.

His research interests include power modeling, design space exploration, and physical design.

Mr. Zhai received the William J. McCalla Best Paper Award from ICCAD 2021.

Chen Bai received the B.E. degree in software engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

His research interests include design space exploration and agile VLSI design methodologies.

Mr. Bai received the William J. McCalla Best Paper Award from ICCAD 2021.

Binwu Zhu received the B.Eng. degree in information engineering from Zhejiang University, Hangzhou, China, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

His current research interest includes machine learning for EDA.

Yici Cai (Senior Member, IEEE) received the B.S. degree in electronic engineering and the M.S. degree in computer science and technology from Tsinghua University, Beijing, China, in 1983 and 1986, respectively, and the Ph.D. degree in computer science and technology from the University of Science and Technology of China, Hefei, China, in 2007.

She is currently a Professor with the Department of Computer Science and Technology, Tsinghua University. Her research interests include VLSI layout theory and algorithms, power/ground network analysis and optimization, high-performance clock synthesis, and low-power physical design.

Qiang Zhou (Senior Member, IEEE) received the B.S. degree in computer science and technology from the University of Science and Technology of China, Hefei, China, in 1983, the M.S. degree in computer science and technology from Tsinghua University, Beijing, China, in 1986, and the Ph.D. degree in control theory and control engineering from the Chinese University of Mining and Technology, Beijing, in 2002.

He has been a Professor with the Department of Computer Science and Technology, Tsinghua University. His current research interests include VLSI layout theory and algorithms.

Bei Yu (Member, IEEE) received the Ph.D. degree from The University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received nine Best Paper Awards from DATE 2022, ICCAD 2021 and 2013, ASPDAC 2021 and 2012, ICTAI 2019, *Integration, the VLSI Journal* in 2018, ISPD 2017, SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD contest awards. He has served as the TPC Chair for ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is an Editor of IEEE TCCPS Newsletter.