3DNN-Xplorer: A Machine Learning Framework for Design Space Exploration of Heterogeneous 3D DNN Accelerators

Gauthaman Murali[1], Aditya Iyer[1], Navneeth Ravichandran[2], and Sung Kyu Lim[1]
[1]*School of ECE, and* [2] *Scheller College of Business, Georgia Institute of Technology, Atlanta, GA, USA*
Email: gauthaman@gatech.edu, limsk@ece.gatech.edu

2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD) | 979-8-3503-2225-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCAD57390.2023.10323675

***Abstract*—This paper presents 3DNN-Xplorer, the first machine learning (ML)-based framework for predicting the performance of heterogeneous 3D DNN accelerators. Our ML framework facilitates the design space exploration of heterogeneous 3D accelerators with a 2-tier compute-on-memory configuration, considering 3D physical design factors. Our design space encompasses four distinct heterogeneous 3D integration styles, combining 28nm and 16nm technology nodes for both compute and memory tiers. Using extrapolation techniques with ML models trained on 16, 32, and 64 PE accelerator configurations, we estimate the performance of systems featuring 128, 256, 512, 1024, and 2048 PEs, achieving a maximum absolute error of 12.7%. To ensure balanced tier areas in the design, our framework assumes the same number of PEs or on-chip memory capacity across the four integration styles, accounting for area imbalance resulting from different technology nodes. Our analysis reveals that the heterogeneous 3D style with 28nm compute and 16nm memory is energy-efficient and offers notable energy savings of up to 50% and an 8.8% reduction in runtime compared to other 3D integration styles with the same number of PEs. Similarly, the heterogeneous 3D style with 16nm compute and 28nm memory is area-efficient and shows up to 8.3% runtime reduction compared to other 3D styles with the same on-chip memory capacity.**

I. Introduction

Deep Neural Network (DNN) accelerators find diverse applications across multiple domains, including computer vision, natural language processing, and autonomous vehicles. In recent years, DNN workloads have grown significantly, encompassing numerous layers and billions of parameters.

However, 2D integration faces challenges in accommodating larger on-chip memory, resulting in worse performance and energy consumption due to the reliance on off-chip memories. 3D integration overcomes this issue by enabling the integration of multi-tier on-chip memories with large capacities. Academic research [1]–[3] and industry demonstrations, such as System on Integrated Chips (SoIC) [4] and Foveros [5], have successfully demonstrated multi-tier memory integration.

While the computational capabilities of accelerator systems progress rapidly, memory technology advances at a slower pace. 3D IC design addresses this challenge by allowing the integration of compute logic and memory at different technology nodes on a single chip. Heterogeneous 3D integration offers diverse design possibilities for a single accelerator architecture. However, manually exploring and identifying the most suitable 3D integration style for different accelerator configurations and workloads can be challenging.

While accelerator simulators provide a faster estimation of performance than performing the actual physical design, cycle-accurate simulators run for several hours to days. Further, these simulators are primarily designed for 2D systems where input/output features are shuttled between on-chip and off-chip memories due to limited on-chip memory capacity. On the other hand, in 3D accelerators with larger on-chip memory capacities, output features of an entire DNN layer can fit within the on-chip memory, enabling efficient computation without the need for data transfers on-chip & off-chip memories.

Both industrial and academic research efforts have resulted in the development of parameterizable accelerators, such as TPU [6], SIGMA [7], Eyeriss [8], and MAERI [9], which can be customized for specific applications. Although these accelerators have been optimized for 2D designs, there is a lack of design space exploration (DSE) and optimization techniques tailored for 3D technology. Our research aims to address this gap using an ML-based performance prediction framework for 3D compute-on-memory heterogeneous accelerators.

The contributions of this paper are as follows:

- We present 3DNN-Xplorer, the first ML-based framework for design space exploration of heterogeneous compute-on-memory 3D accelerators, providing a reliable and close approximation to the actual physical design process.
- We train frequency, power, runtime, and energy models using 16, 32, & 64 PE accelerator configurations and perform extrapolation to predict the performance of 128, 256, 512, 1024, and 2048 PE accelerator systems with a 12.7% max. prediction error.
- Our DSE shows that among 4 integration styles combining 28nm and 16nm tech nodes, the style with 28nm compute and 16nm memory is energy-efficient, offering up to 50% energy savings and 8.8% runtime reduction over other 3D designs with the same number of PEs.
- The heterogeneous 3D style with 16nm compute and 28nm memory is area-efficient and offers up to 8.3% runtime reduction over other 3D designs with the same on-chip memory capacity.

II. Background & Related Works

Several studies have proposed different architectures and integration techniques for 3D machine learning (ML) accelerators [3], [10], [11]. While 3D accelerators offer inherent

performance benefits compared to 2D counterparts, optimizing the architecture, integration approach, and workload dataflow can yield highly energy-efficient accelerators.

DSE techniques are commonly used to explore the extensive range of design parameters for 2D accelerators. For instance, Esmaeilzadeh et al. proposed a technique [12] that utilizes automatic machine learning (AutoML) [13] to predict the performance of hardware-accelerated ML algorithms. Their predictive models estimate various performance metrics, including design frequency, chip power, workload runtime, and energy usage for a given 2D accelerator configuration. While they perform DSE of the 2D accelerators, this study is limited by interpolation techniques and a maximum prediction error rate of 53.61%, suggesting the possibility of better configurations beyond the training set.

On the other hand, the 3D accelerator DSE studies are limited. Mathur et al. explored thermal-aware design space for 3D systolic ML accelerators [14]. Their focus was on investigating options for multi-tier 3D integration, but the study was restricted to a narrow range of accelerator architectural configurations and lacked performance prediction frameworks for larger configurations. Li et al. conducted on-chip memory technology DSE for mobile DNN accelerators using 3D vertical RRAM [15]. However, their study primarily concentrated on memory technology with an iso-throughput accelerator configuration, rather than exploring the overall accelerator design space.

While extensive research exists on optimizing DSE for 2D accelerators [12], [16], detailed work on 3D accelerators is limited. In this paper, we present a comprehensive approach to DSE of heterogeneous 3D DNN accelerators, considering various aspects such as accelerator configurations, 3D integration methodology, compute and memory technology nodes, workload dimensions, clock frequency, power, and energy requirements. Importantly, our technique offers highly accurate performance predictions for larger accelerator designs with minimal training runtime using insights from smaller designs.

III. Design and Simulation Tools Used

The objective of our ML-based training framework is to develop trained models using small accelerators that can accurately predict significant physical design and workload metrics of large ones. This will aid in an efficient design space exploration of compute-on-memory 3D accelerator designs. We use simple architecture, 3D integration, and physical design features as input parameters to our training model (as detailed in Section IV-B). The chosen parameters simplify the process of DSE, as it is not necessary to carry out initial synthesis or physical design to use the trained model.

A. Benchmark Architecture

We use MAERI accelerator [9] to devise an extensive approach to DSE for homogeneous and heterogeneous 3D ML accelerators. MAERI is a deep neural network (DNN) accelerator that boasts flexible/programmable interconnects between processing elements (PEs), as shown in Figure 1,

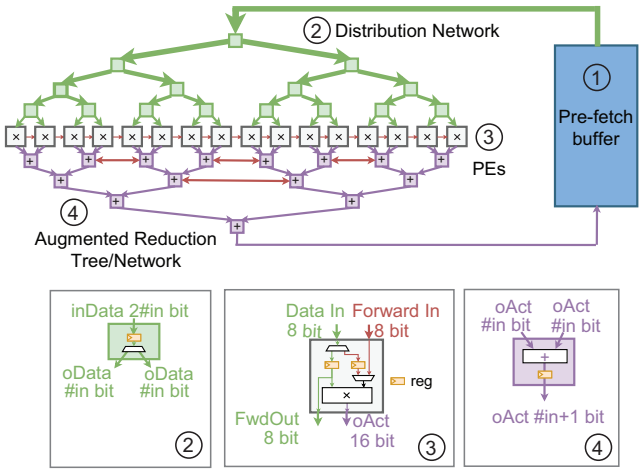

Fig. 1: MAERI architecture - the benchmark accelerator used in this work.

allowing it to achieve high utilization on both sparse and dense workloads. This flexibility permits dataflow optimizations on a per-layer basis, making MAERI highly energy-efficient. Additionally, MAERI is an open-source DNN accelerator that is highly configurable and includes a sophisticated workload simulator called STONNE [17] that allows for custom dataflow capabilities.

While other accelerator types can also be implemented using 3D technology, the ability of MAERI's design and simulation environment to adapt to 3D technology and corresponding dataflow optimizations makes it a more convenient choice for our DSE study. Nonetheless, the methods we propose in this work are generic and can be applied to any type of machine learning accelerator.

B. Architectural Simulator

Simulating 3D accelerator dataflow using a 2D simulator presents a significant challenge. 3D ICs offer several advantages, including low access latency and large on-chip memory compared to their 2D counterparts. These benefits result in reduced energy consumption and runtime, thereby improving the two crucial workload-performance metrics of an accelerator. STONNE [17], used for simulating 2D MAERI, does not model the advantages of having a large on-chip memory. The simulator handles each layer's simulation as an independent run, resulting in the loss of previously stored outputs in the large on-chip memory. Consequently, it retrieves inputs once again from the DRAM, discarding the results from any previous computations. We overcome this limitation by modeling dataflow between layers, as shown in Algorithm 1.

The upgraded simulation framework with modified dataflow is used to compute the runtime and energy of a given workload as follows.

- The STONNE simulator [17] is used to calculate the execution cycles (C) for a given DNN workload layer, assuming all inputs are stored in the on-chip SRAM buffer.

- Using the dataflow shown in Algorithm 1, we compute DRAM accesses and use it to modify the total execution cycles for a given workload.
- We obtain the total execution cycles (TC) using Equation 1.

$$\text{TC} = \text{C} + \text{Accesses}_{DRAM} \times \text{Latency}_{DRAM} \quad (1)$$

- To calculate the compute logic and SRAM access energy (E) required to execute a given workload, we provide STONNE the compute logic and on-chip memory power obtained from the physical design of the accelerator being simulated.
- We calculate the total energy (TE) using Equation 2:

$$\text{TE} = \text{E} + \text{Accesses}_{DRAM} \times \text{DE} \quad (2)$$

where DE is the DRAM access energy per byte. We assume a DRAM access energy of 120pJ/byte as suggested in [14].

Algorithm 1: Dataflow used in 3D MAERI simulation framework.

```
if previous layer outputs fit in on-chip memory then
    read filters from DRAM;
    read previous layer outputs from on-chip memory;
else
    read previous layer outputs & filters from DRAM;
end
if current layer outputs fit in on-chip memory then
    write current layer outputs to on-chip memory;
else
    write current layer outputs to DRAM;
end
```

C. 3D IC Physical Design and Simulation Tools

To generate the necessary Verilog files for the accelerator, we utilize the flexible MAERI RTL generator [18]. Subsequently, we synthesize the netlist using Synopsys Design Compiler (DC). The physical design is then performed using Cadence Innovus. We provide the tool with the floorplan of the memory macros and use the Macro-3D [19] flow to perform compute-on-memory 3D design. Our 3D designs feature a 3D back-end of line (BEOL) with six metal layers on each tier. The spacing, width, and RC parasitics of the metal layers are adjusted based on the technology node of the respective tier, as illustrated in Figure 3.

We extract the timing and power information of the standard cells and memories in the design using Cadence Tempus and use it to calculate the effective design frequency and chip power. We then incorporate these physical design metrics into the STONNE [17] simulator to obtain accurate runtime and energy metrics for each accelerator configuration.

IV. ML Prediction Model Definition

The 3DNN-Xplorer framework has four ML models to predict the performance of a given accelerator configuration. This section defines the models and the set of features used to train these models. Figure 2 shows the models and features used in the 3DNN-Xplorer framework.

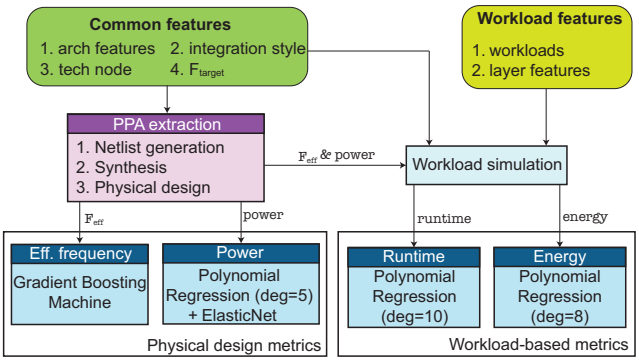

Fig. 2: 3DNN-Xplorer: Training Framework

A. 4 Separate Models Built in This Work

We create prediction models necessary for an easy and reliable DSE of 3D accelerators. We use four significant performance metrics to evaluate an accelerator configuration.

- Effective design frequency (based on Gradient Boosting): The final operational frequency in GHz of a given accelerator configuration.
- Chip power (based on Polynomial Regression): The average chip power in mW, assuming an activity of 10%.
- Workload runtime (based on Polynomial Regression): The overall time in ms required to run an entire ML workload.
- Workload-specific energy (based on Polynomial Regression): The energy consumption of the design in mJ to execute an entire workload.

B. Common Input Features

The following parameters are shared as the inputs for our four ML-based prediction models. Table I summarizes the list of these common input features.

a) Architectural Parameters: The accelerator configuration is determined by the architectural parameters, specifically the number of processing elements (#PE) and the total memory bandwidth of the on-chip SRAM buffer (#BW). The memory bandwidth chosen for training depends on the number of #PEs in the design. Our approach involves selecting the most favorable bandwidth values that can provide a reasonable throughput while minimizing or completely eliminating memory access stalling.

b) Technology Nodes: We use two different technology nodes in this work: 16nm and 28nm.

c) 3D Integration Style: 3D integration offers the advantage of integrating different technology nodes on different tiers. The computation speed and power depend greatly on the technology node of both compute and memory tiers. Furthermore, the capacity of on-chip memory is significantly influenced by the technology nodes and available silicon area of both tiers. In this work, we perform homogeneous and heterogeneous 3D designs and compare them in terms of PPA, workload runtime & energy. As we are utilizing two different technology nodes in this study, we construct two

variants each of homogeneous and heterogeneous 3D designs. Table II shows the four different 3D integration styles used in this work. While typically the memory is at an older node in heterogeneous 3D designs [3], [20], in this work, we explore a heterogeneous 3D style with memory at a more recent node than the compute.

d) Target Frequency (F_{target}): The design frequency is a crucial factor in exploring the design space of any accelerator since it impacts the power consumption, workload runtime, and energy efficiency of the accelerator. We conduct a frequency sweep ranging from 0.1 GHz to 4 GHz, in increments of 0.1 GHz, for every combination of architectural parameters and 3D integration style. For both the 16 and 28nm technology nodes, the maximum design frequency of any MAERI accelerator configuration falls comfortably within the chosen range.

C. Workload Features

The following parameters are exclusive to the workload-specific runtime and energy models.

a) Workload Parameters: Besides the architectural features, the overall workload execution runtime and energy are also influenced by various parameters related to the workload itself. These parameters include the size of the input features, the size of the filters, the number of channels, the number of kernels, and the stride size. Instead of training the workload-related models on the entire workload, we train them specifically using these features. This approach allows for more flexibility in predicting and extrapolating workload performance to unseen designs and workloads. Table III lists the workload-specific features used in training the ML-based prediction models.

D. Training Design Space

In total, we perform 960 synthesis and physical design runs combining the features listed in Table I. These runs take around 219 hours and were performed on six 2.10 GHz Intel® Xeon® Gold 6130 servers, using 64 cores in each server. It should be noted that this runtime is close to the design time of a single large MAERI accelerator with 2048 PEs.

Our 3D designs involve 2 tiers, as shown in Figure 3: one for compute and one for memory. We integrate the two tiers in a face-to-face fashion using hybrid bonds of 1 μm pitch in homogeneous 3D (16nm or 28nm) and heterogeneous 3D (16nm & 28nm) fashions, as shown in Figure 3. Although there are various approaches to implementing the two tiers in a 3D IC, we prioritize area balancing as a crucial factor for determining their respective areas. Table II shows the relationship between #PEs and the total on-chip memory capacity that keeps the area imbalance between the memory and compute tiers within 5%.

V. ML Model Training Methodology

This section describes the training techniques employed for each model (shown in Figure 2) to perform an accurate prediction of unseen design configurations through extrapolation. Unseen configurations pertain to any values of the input parameters listed in Table I that were not employed during the training process.

TABLE I: Physical design features used in ML training. Total datapoints = 960.

Feature	Values	Total	Description
PEs	16, 32, 64	3	Total #PEs in the design
Memory BW	$\frac{\#PE}{4}$, $\frac{\#PE}{2}$	2	Global buffer bandwidth
Tech node	16nm, 28nm	2	Tech node of each tier
Integration style	homogeneous, heterogeneous	2	Compute-on-Memory 3D integration style
Frequency	0.1 - 4 GHz	40	Increments of 0.1 GHz

TABLE II: Different 3D design styles explored in this work. The relation between on-chip memory and #PEs varies based on the tech node to balance the compute & memory tier areas.

3D integration style	Compute Node	Memory Node	On-chip memory for a given #PE (KB)
Homogeneous 1	28nm	28nm	#PEs
Heterogeneous 1	28nm	16nm	2×#PEs
Heterogeneous 2	16nm	28nm	#PEs/4
Homogeneous 2	16nm	16nm	#PEs/2

The ML models for effective frequency and chip power are trained using the corresponding metrics obtained from the actual physical design of the 960 accelerator configurations listed in Table I. Sections V-A & V-B explain the training methodology used for effective frequency and power prediction models, respectively.

While the physical design metrics provide a general understanding of an accelerator's performance, workload performance prediction models offer an application-specific assessment. The overall runtime required to execute a workload is influenced by the effective clock frequency of the accelerator, and the energy consumption depends on the chip power and the DRAM access energy. The ML models for predicting workload execution runtime and energy consumption models are explained in Sections V-C and V-D, respectively

A. Training Frequency Model

We first train the frequency prediction model using the entire training dataset. Unlike other performance metrics, there is a maximum limit on the effective design frequency. It is impossible to increase the design frequency beyond a certain limit depending on the technology nodes involved.

TABLE III: DNN workload-specific parameters used in ML training

Parameter	Comments
R	filter row dimension
S	filter column dimension
C	Number of input channels
K	Number of filter kernels
X	input row dimension
Y	input column dimension
Strides	Stride of the layer
TC	Input channels mapped per cycle
TK	Filter kernels mapped per cycle

Workloads
ResNet-50
ResNet-34
GoogLeNet
AlexNet
MobileNetv1

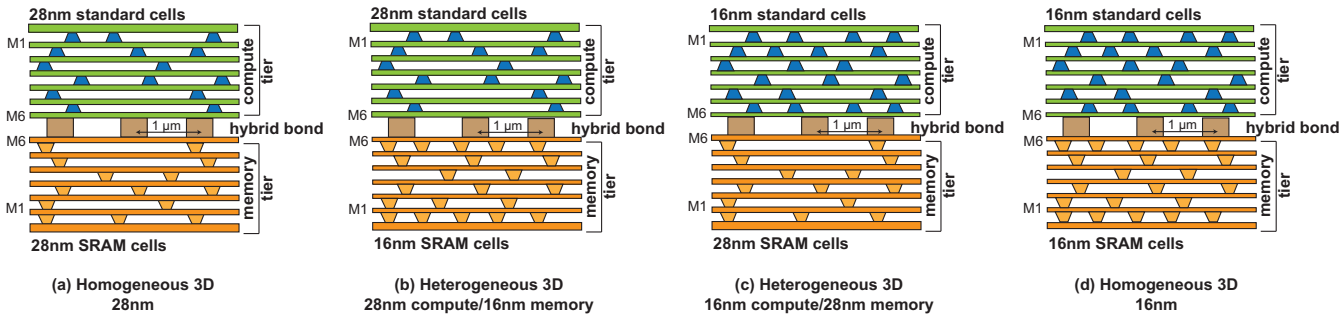

Fig. 3: Four different Face-to-Face Compute-on-Memory (CoM) 3D integration styles used in this work. We use hybrid bonds of $1\mu m$ pitch and a 6 metal layer back-end of line in each tier.

A significant proportion of the input configurations present in our training dataset exhibit maximum effective frequency saturation at approximately 3 GHz. Nevertheless, we extend the range of the target frequency up to 4 GHz and incorporate the resulting effective frequencies into the training dataset. This approach guarantees that the ML model is trained to accurately predict the saturation limits of MAERI, considering the number of PEs and memory bandwidth. Gradient boosting algorithm offers excellent extrapolation when trained using an extensive dataset and when the prediction values lie within the training range [21]. The effective frequencies of larger designs are always less than those of the smaller designs. Therefore, we use the gradient boosting algorithm in the scikit-learn framework [22] with a depth = 5 and 1000 boosting stages to train the frequency model. We use the Huber function shown in Equation 3 as the loss function (L_{freq}) in the gradient boosting algorithm, with the alpha-quantile (α) set to 0.9.

$$L_{freq} = \begin{cases} \dfrac{(F_{act} - F_{pred})^2}{2}, & \text{if}(F_{act} - F_{pred}) \leq \alpha \\ \alpha|(F_{act} - F_{pred})| - \dfrac{\alpha}{2}, & \text{otherwise} \end{cases} \tag{3}$$

where F_{act} is the actual observed effective frequency and F_{pred} is the effective frequency predicted by the model.

The loss function utilized in our frequency model incorporates both absolute and squared prediction errors. The alpha-quantile is set to 0.9 to enhance the robustness of the model against outliers present in the frequency curve, particularly when dealing with values beyond the maximum attainable effective frequency of the design. The accuracy of the trained model is demonstrated by its impressive R2 score of 0.999997.

B. Training Power Model

Unlike frequency, power displays a direct yet non-linear correlation with all the input parameters used in the training process. Often, ensemble algorithms, such as Gradient Boosting Machine and Random Forest, offer a poor prediction for such metrics on unseen configurations [23] (More details in Section V-C and Table IV). Therefore, we employ Polynomial Regression for training the power prediction model. This method typically involves creating polynomial features from the input data in the pre-processing stage, followed by applying linear regression to these features. However, linear regression does not employ any regularization techniques and treats all the polynomial features equally, which leads to overfitting the data. Therefore, we perform ElasticNet regression on the polynomial features extracted in our approach.

ElasticNet uses both L1 (Lasso) and L2 (Ridge) regularizations offering a fine balance between feature selection and coefficient regularization. The polynomial features extracted involve higher-order, and co-dependent combinations of the input training parameters, such as $\#PE$, F^n_{target}, $\#PE \times \#BW$, $\#BW \times F_{target}$, etc. ElasticNet regression helps in prioritizing the most relevant features leading to a highly accurate model generation. The loss function L_{power} of ElasticNet regression is given by Equation 4.

$$L_{power} = \frac{\sum_{i=1}^{n} |P_{i,act} - P_{i,pred}|^2}{2n} + \lambda_1 \|\beta\|_1 + 0.5\lambda_2 \|\beta\|_2^2 \tag{4}$$

where, $P_{i,act}$ and $P_{i,pred}$ are the actual observed and predicted power for a given input configuration i, $\|\beta\|_1$ and $\|\beta\|_2^2$ represent the L1 and L2 norms of the polynomial coefficients, respectively, and λ_1, λ_2 are regularization parameters.

As a trade-off between feature selection and coefficient regularization, we set $\lambda_1 + \lambda_2 = 0.5$ and $\frac{\lambda_1}{\lambda_1+\lambda_2} = 0.5$, as suggested by scikit-learn framework [22]. With the degree of the polynomial feature extraction stage set to 5, the trained power model has an R2 score of 0.9994.

C. Training Runtime Model

The runtime prediction model used in our study incorporates both architectural parameters from Table I and workload-specific parameters described in Table III. This comprehensive approach enhances the accuracy and reusability of our prediction models for runtime estimation of various workloads.

To generate the dataset for the runtime model, we simulated ResNet-50, ResNet-34, AlexNet, GoogLeNet and MobileNetV1 using the STONNE [17] simulator. MAERI's high configurability allows multiple mapping possibilities per convolution layer, leading to sub-optimal mappings that underutilize the available PEs in the design. Since our goal was to identify the best design, we specifically simulated mappings that maximized the utilization of all PEs.

TABLE IV: Ensemble vs Regression techniques for runtime prediction. ML framework used: scikit-learn [22]

Ensemble methods				Regression	
Algorithms	Parameter	Range	Max R2 Score	Degree	R2 Score
Ensemble of GBM, RF, DT	n_estimators	[100-1000]	-0.15	6	0.772
	learning_rate	[0.1-1.0]		8	0.978
	max_depth	[3,100]		10	0.998

We explored two ML approaches to create a highly accurate model, as detailed in Table IV: 1) stacked ensemble of Gradient Boosting Machine (GBM), Random Forest (RF), and Decision Trees (DT), and 2) Polynomial Regression. While ensemble algorithms exhibit strong performance when interpolating within the training dataset, their effectiveness diminishes when extrapolating beyond the dataset's convex hull [23]. After assessing the model scores, we use the Polynomial Regression model of degree 10, as it exhibits the highest prediction score on extrapolation to new data. To address features having an inversely proportional relationship (#PEs, #BW) with the response variable, we transform them by taking their reciprocals during the training process.

D. Training Energy Model

We train the energy model to predict the compute energies of the linear network of PEs. While we can analytically calculate the energies associated with SRAM and DRAM accesses, the compute energies cannot be derived analytically due to the lack of per-cycle activity information. To overcome this limitation, we employ a model that predicts the compute energies. Following the methodology outlined in Section III-B, and conducting experiments similar to those presented in Table IV, we select a Polynomial Regression model.

VI. ML Inference Methodology

The first step in the prediction flow is to find the effective frequency. We calculate the effective frequency of the new design being explored using Equation 5.

$$\begin{aligned} F_{eff} &= \text{pred\_eff\_frequency}(pe, bw, is, F_{target}); \\ pe = 2^x, x \in [4,11], bw \in \{\frac{pe}{4}, \frac{pe}{2}\}, is \in \{0,1,2,3\} \end{aligned} \quad (5)$$

TABLE V: Prediction model accuracy based on randomly chosen 128 & 256 PE configurations.

Model	ML Algorithm used	Mean error, Max. Error
Frequency	Gradient Boosting Machine	0.8%, 4.9%
Power	Polynomial Regression	3.1%, 8.3%
Runtime	Polynomial Regression	ResNet-50: 2.8%, 7.7%
		ResNet-34: 3.4%, 8.0%
		GoogLeNet: 1.6%, 8.9%
		AlexNet: 0.1%, 0.12%
		MobileNetv1: 0.5%, 2.0%
Energy	Polynomial Regression	ResNet-50: 7.3%, 12.4%
		ResNet-34: 8.4%, 10.4%
		GoogLeNet: 8.2%, 10.0%
		AlexNet: 6.9%, 10.9%
		MobileNetv1: 8.7%, 12.7%

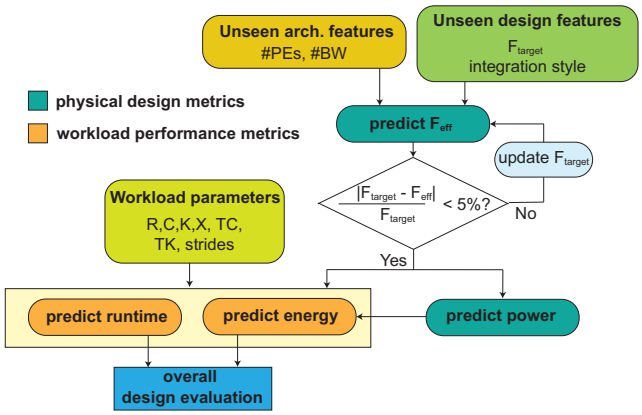

Fig. 4: Inference methodology used in 3DNN-Xplorer to evaluate unseen accelerator configurations during DSE.

where, F_{eff} is the predicted effective frequency, pe, and bw are the total number of processing elements and the on-chip memory bandwidth in the design being explored, is is the integer-coded 3D integration style, and F_{target} is the target frequency. We have tested the prediction model for up to a total #PE count of 2^{11} or 2048, and therefore we restrict the input variable pe to 2048. Based on the prediction methodology shown in [12], we define a frequency range of interest (F_{ROI}) for the prediction models to ensure maximum accuracy. The F_{ROI} is given by Equation 6.

$$F_{ROI} \in \forall F_{eff} : \frac{|F_{target} - F_{eff}|}{F_{target}} \leq 5\% \quad (6)$$

If the predicted effective frequency is beyond a 5% margin from the desired target frequency, it becomes futile to investigate that particular design configuration further. We keep updating the target frequency until the predicted effective frequency is within the 5% limit. We then predict the chip power. Subsequently, using the effective frequency, we predict the runtime for one of the five workloads used in training. Using the chip power, and frequency values, we use the energy model to predict the overall energy for a chosen workload. Figure 4 shows the prediction methodology employed in 3DNN-Xplorer. The predicted energy and runtime are the final metrics used to evaluate the selected design configuration.

Our primary objective is to develop resilient prediction models that facilitate precise extrapolation of performance metrics for unseen accelerator configurations. Large accelerator designs with $\geq$ 128 processing elements demand substantial design time, rendering the design space exploration for such configurations extremely time-consuming. Using appropriate training strategies tailored to each performance metric, we are able to construct prediction models capable of extrapolating these metrics to designs larger than those exposed to the ML models in the training phase. Table V shows the accuracy of our models compared to the actual physical design and simulation data for various metrics. All our models have a maximum error $\leq$ 12.71%.

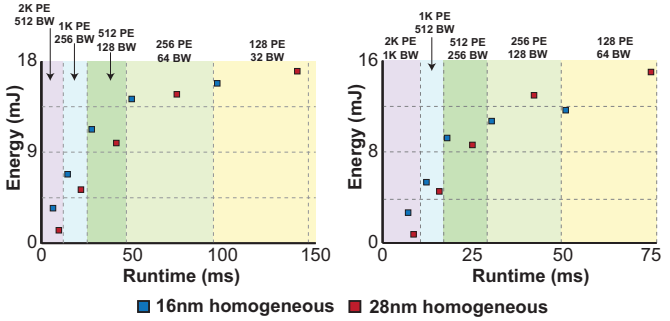

Fig. 5: DSE of the two homogeneous 3D MAERI configurations with the same #PEs and $\frac{\#PE}{4}$ (left) or $\frac{\#PE}{2}$ (right) words/cycle bandwidth. The on-chip memory capacity varies based on the tech node. Workload used: ResNet-50.

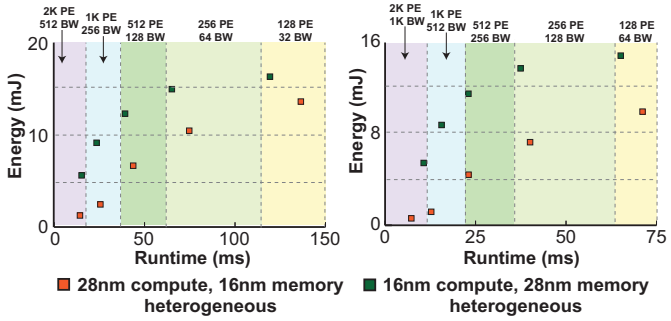

Fig. 6: DSE of the two heterogeneous 3D MAERI configurations with the same #PEs and $\frac{\#PE}{4}$ (left) or $\frac{\#PE}{2}$ (right) words/cycle bandwidth. The on-chip memory capacity varies based on the tech node. Workload used: ResNet-50.

VII. Design Space Exploration Results

A. Experimental Setup

We conduct an extensive design space exploration encompassing the design configurations shown below.

Design space explored				
#PEs	**#BW**	**Frequency**	**Nodes**	**Styles**
128, 256, 512, 1024, 2048	$\frac{\#PE}{4}$, $\frac{\#PE}{2}$	0.5-4 GHz	16nm, 28nm	Homogeneous 3D, Heterogeneous 3D

By sweeping the target frequency range from 0.5 to 4 GHz, we determine the maximum effective design frequency for each integration style and configuration. Subsequently, we estimate the chip power based on the predicted maximum design frequency. Utilizing the predicted physical design metrics, we estimate the overall energy consumption and runtime to execute ResNet-50 on each configuration using the parameterized workload prediction models.

We perform two different design space explorations using the predicted ResNet-50 runtime and energy values.

1) We compare designs with the same number of PEs across different integration styles. In this case, the on-chip memory varies based on the integration style according to the relation given in Table II.
2) We compare designs with the same on-chip memory across various integration styles. In this case, the #PEs differ based on the relationship shown in Table II.

We assume that the compute and memory tiers have a balanced total silicon area in both comparisons. This ensures a fair and consistent evaluation of the different configurations, allowing us to isolate and analyze the impact of other design parameters on performance and efficiency. Our approach combines the power of advanced prediction models with the flexibility of DSE, allowing us to make informed decisions regarding integration styles and design parameters.

B. Homogeneous 3D Integration

This section presents a comparison of runtime and energy consumption for the ResNet-50 workload across different configurations of the MAERI accelerator, designed in 16 and 28nm homogeneous 3D styles. Figure 5 shows the DSE of homogeneous 3D MAERI designs with the same number of PEs across different integration styles. Our design space exploration reveals that the optimal design in the space explored is 28nm homogeneous 3D MAERI with 2048 PEs, a bandwidth of 1024 words/cycle, and 2 MB on-chip memory. This configuration achieves a lower energy consumption of 1.22 mJ and a runtime of 9.35 ms on the ResNet-50 workload. While the runtime of the 16nm homogeneous 3D design with 2048 PE is better than that of its 28nm counterpart, it uses almost 5$\times$ more energy to execute ResNet-50. This is due to the fact that the 16nm 2048 PE design has half the on-chip memory capacity of its 28nm counterpart.

In contrast, the exploration focusing on maintaining the same on-chip memory capacity (see Table VI) identifies 16nm homogeneous 3D MAERI as a better design over corresponding 28nm designs as it offers better energy and runtime.

Overall, the 28nm 3D accelerators are more energy-efficient than the 16nm ones for #PEs $\geq$ 512. Conversely, the smaller area and shorter runtime of the 16nm 3D accelerators make them more area-efficient compared to the 28nm accelerators.

C. Heterogeneous 3D Integration

Figure 6 shows the DSE of heterogeneous 3D accelerators with the same number of PEs across various integration styles. Among all heterogeneous 3D designs, the design with 28nm compute and 16nm memory, featuring 2048 PEs, a bandwidth of 1024 words/cycle, and an on-chip memory capacity of 4 MB is the most optimal design. This design executes ResNet-50 in 7.25 ms & with an energy consumption of 0.61 mJ.

Conversely, when maintaining the same memory capacity across the two heterogeneous styles (See Table VI) and striving for area balance between tiers, the heterogeneous style with 16nm compute and 28nm memory emerges as the most favorable option. This configuration entails 2048 PEs, a bandwidth of 1024 words/cycle, and an on-chip memory capacity of 512 KB. It offers the shortest runtime of 11.17 ms and a low energy consumption of 5.4 mJ to execute ResNet-50 among other designs with the same on-chip memory. In summary, the heterogeneous style with 28nm compute exhibits higher

TABLE VI: DSE of 3D MAERI configurations with same on-chip memory capacity. The #PEs varies based on tech node. Workload used: ResNet-50.

		Homogeneous 3D						Heterogeneous 3D					
		#PE		Runtime (ms)		Energy (mJ)		#PE		Runtime (ms)		Energy (mJ)	
On-chip memory	#BW	16nm	28nm	16nm	28nm	16nm	28nm	28nm C 16nm M	16nm C 28nm M	28nm C 16nm M	16nm C 28nm M	28nm C 16nm M	16nm C 28nm M
256 KB	$\frac{\#PE}{4}$	512	256	31.24	74.69	10.97	14.24	128	1024	129.86	23.60	14.06	9.44
	$\frac{\#PE}{2}$			18.87	43.44	8.52	12.53			71.14	15.59	9.95	8.78
512 KB	$\frac{\#PE}{4}$	1024	512	18.91	43.81	6.78	9.69	256	2048	70.42	15.34	10.79	5.79
	$\frac{\#PE}{2}$			**12.17**	26.07	**5.4**	8.50			40.07	**11.17**	7.27	**5.4**

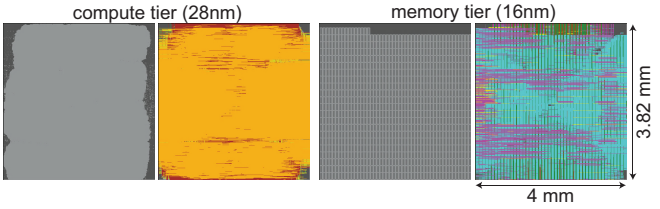

Fig. 7: Final layouts of energy-efficient 2048 PE 3D MAERI.

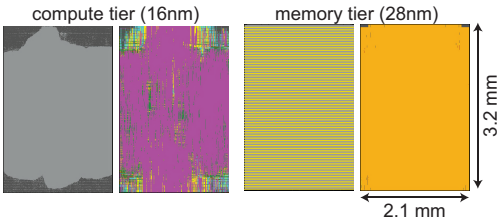

Fig. 8: Final layouts of area-efficient 2048 PE 3D MAERI.

energy efficiency, while the other style with 16nm compute demonstrates superior area efficiency.

D. Homogeneous vs Heterogeneous 3D Integration

In this section, we compare the energy-efficient and area-efficient designs of homogeneous integration styles against their heterogeneous versions. The energy-efficient heterogeneous design with 28nm compute and 16nm memory, featuring 2048 PEs & 1024 words/cycle bandwidth, achieves 50% energy savings and an 8.8% reduction in runtime over the 28nm homogeneous 3D design with the same number of PEs. The area-efficient heterogeneous design with 16nm compute and 28nm memory, featuring 512 KB on-chip memory & 1024 words/cycle bandwidth, demonstrates an 8.3% runtime improvement over its area-efficient 16nm homogeneous counterpart. These findings highlight the superior performance of the heterogeneous 3D designs, showcasing their potential for achieving both energy savings and runtime improvements.

We perform the physical design of the two best configurations. Figures 7 and 8 show the final layouts of the most energy-efficient and area-efficient 3D MAERI designs, respectively. The performance analyses of these designs are summarized in Table VII. The observed performance metrics are very close to our predictions and differ only by 7.1%.

TABLE VII: PPA analysis of the best MAERI configurations.

Metric	Energy-efficient 3D MAERI	Area-efficient 3D MAERI
Compute node	28nm	16nm
Memory node	16nm	28nm
#PEs	2048	2048
#BW (words/cycle)	1024	1024
On-chip memory (MB)	4	0.5
Chip area (mm^2)	15.2	6.7
Max. frequency (GHz)	1.67	1.94
Chip power (W)	28.3	18.5
ResNet-50 runtime (ms)	7.0	10.4
ResNet-50 energy (mJ)	0.62	5.35
Energy efficiency ($\frac{TOPS}{W}$)	**19.4**	2.24
Area efficiency ($\frac{TOPS}{mm^2}$)	0.11	**0.18**

VIII. Conclusion

In this study, we conducted an extensive DSE of compute-on-memory 3D MAERI accelerators, encompassing homogeneous and heterogeneous integration styles. To navigate this vast design space, we introduced an ML-based performance prediction framework called 3DNN-Xplorer. This framework facilitated the evaluation of various accelerator configurations and integration styles, delivering reliable and robust performance estimates. By training prediction models on different smaller accelerators, we achieved highly accurate performance extrapolation for larger systems. Our findings identified the most optimal design configurations for both homogeneous and heterogeneous approaches, maintaining the same number of PEs or on-chip memory capacities. Notably, the heterogeneous integration styles emerged as exemplary configurations, offering significant energy savings and runtime reductions compared to their homogeneous counterparts. Furthermore, our results highlighted the trade-off between energy efficiency and runtime performance when considering different technology nodes. 28nm 3D accelerators demonstrated higher energy efficiency, while 16nm accelerators exhibited improved runtime performance. Therefore, heterogeneous integration offers an excellent trade-off between runtime and energy-efficiency.

Acknowledgment

This work was supported in part by CHIMES, one of the seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

REFERENCES

[1] Prachi Shukla, Vasilis F. Pavlidis, Emre Salman, and Ayse K. Coskun. Temperature-aware monolithic 3d dnn accelerators for biomedical applications. In *Design, Automation and Test in Europe Conference (DATE)*, 2022.

[2] Wei Lu, Po-Tsang Huang, Hung-Ming Chen, and Wei Hwang. An energy-efficient 3d cross-ring accelerator with 3d-sram cubes for hybrid deep neural networks. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 11(4):776–788, 2021.

[3] Gauthaman Murali, Xiaoyu Sun, Shimeng Yu, and Sung Kyu Lim. Heterogeneous mixed-signal monolithic 3-d in-memory computing using resistive ram. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 29(2):386–396, 2021.

[4] Ming-Fa Chen, Fang-Cheng Chen, Wen-Chih Chiou, and Doug C.H. Yu. System on integrated chips (soic(tm) for 3d heterogeneous integration. In *2019 IEEE 69th Electronic Components and Technology Conference (ECTC)*, pages 594–599, 2019.

[5] D. B. Ingerly et al. Foveros: 3d integration and the use of face-to-face chip stacking for logic devices. In *2019 IEEE International Electron Devices Meeting (IEDM)*, pages 19.6.1–19.6.4, 2019.

[6] Norman P. Jouppi et al. A domain-specific supercomputer for training deep neural networks. *Commun. ACM*, 2020.

[7] Eric Qin et al. Sigma: A sparse and irregular gemm accelerator with flexible interconnects for dnn training. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2020.

[8] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks. *IEEE Journal of Solid-State Circuits*, 2017.

[9] Hyoukjun Kwon, Ananda Samajdar, and Tushar Krishna. A communication-centric approach for designing flexible dnn accelerators. *IEEE Micro*, 2018.

[10] Kota Shiba, Tatsuo Omori, Mototsugu Hamada, and Tadahiro Kuroda. A 3d-stacked sram using inductive coupling technology for ai inference accelerator in 40-nm cmos. In *2021 26th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 97–98, 2021.

[11] Chin-Yang Lo, Po-Tsang Huang, and Wei Hwang. Energy-efficient accelerator design with 3d-sram and hierarchical interconnection architecture for compact sparse cnns. In *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 320–323, 2020.

[12] Hadi Esmaeilzadeh et al. Physically accurate learning-based performance prediction of hardware-accelerated ml algorithms. In *2022 ACM/IEEE 4th Workshop on Machine Learning for CAD (MLCAD)*, pages 119–126, 2022.

[13] Erin LeDell and Sebastien Poirier. H2O AutoML: Scalable automatic machine learning. *7th ICML Workshop on Automated Machine Learning (AutoML)*, July 2020.

[14] Rahul Mathur, Ajay Krishna Ananda Kumar, Lizy John, and Jaydeep P. Kulkarni. Thermal-aware design space exploration of 3-d systolic ml accelerators. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 7(1):70–78, 2021.

[15] Haitong Li, Mudit Bhargava, Paul N. Whatmough, and H.-S. Philip Wong. On-chip memory technology design space explorations for mobile deep neural network accelerators. In *Proceedings of the 56th Annual Design Automation Conference 2019*, DAC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[16] Ye Yu, Yingmin Li, Shuai Che, Niraj K. Jha, and Weifeng Zhang. Software-defined design space exploration for an efficient dnn accelerator architecture. *IEEE Transactions on Computers*, 70(1):45–56, 2021.

[17] Francisco Muñoz-Matrínez, José L. Abellán, Manuel E. Acacio, and Tushar Krishna. Stonne: Enabling cycle-level microarchitectural simulation for dnn inference accelerators. In *2021 IEEE International Symposium on Workload Characterization (IISWC)*, 2021.

[18] Hyoukjun Kwon, Ananda Samajdar, and Tushar Krishna. Maeri project. https://github.com/maeri-project/MAERI_bsv, 2019. Accessed: 20 May 2023.

[19] Lennart Bamberg, Alberto García-Ortiz, Lingjun Zhu, Sai Pentapati, Da Eun Shim, and Sung Kyu Lim. Macro-3d: A physical design methodology for face-to-face-stacked heterogeneous 3d ics. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 37–42, 2020.

[20] Sai Surya Kiran Pentapati and Sung Kyu Lim. Heterogeneous monolithic 3d ics: Eda solutions, and power, performance, cost tradeoffs. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 925–930, 2021.

[21] Alexey Malistov and Arseniy Trushin. Gradient boosted trees with extrapolation. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 783–789, 2019.

[22] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[23] Roozbeh Yousefzadeh and Xuenan Cao. To what extent should we trust AI models when they extrapolate? *CoRR*, abs/2201.11260, 2022.