IEEE/ACM
2023 INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN
ICCAD
42nd Edition

# IT-DSE: Invariance Risk Minimized Transfer Microarchitecture Design Space Exploration

Ziyang Yu[1], Chen Bai[1], Shoubo Hu[2], Ran Chen[2], Taohai He[3], Mingxuan Yuan[2], **Bei Yu**[1], Martin Wong[1]

[1] The Chinese University of Hong Kong
[2] Huawei Noah's Ark Lab
[3] HiSilicon

Oct. 31, 2023

# Outline

# Background

## Microarchitecture Design Space exploration:

- Microarchitecture determines the detailed structures of a microprocessor.
- Microarchitecture design space exploration involves finding different configurations with desired performance, power and area (PPA).

Two major challenges:

- Complexity of design spaces growing rapidly: number of potential configurations, inter-feature interactions, mixed-type parameters.
- Getting desired evaluation metric reports for one microarchitecture configuration through VLSI integration flow is timing consuming.

# Previous Solutions and Limitations

## Previous solution

In industry:

- Manually configure design parameters by computer architects.
- **Limitation:** Requires both sufficient domain knowledge and a large amount of human labor for computer architects.

In academia:

- Machine learning-based process simulation models or surrogate models.
- **Limitations:** Effectiveness depdends on quality of training data and model granularity.

# Historical Configurations are Useful

Two facts:

- Different generations of microprocessors are developed based on a common baseline microarchitecture. Minor improvements are applied across consecutive microarchitecture generations.
- Some general relations exist for design features due to the similar microarchitecture design paradigm.

ANN-TL[1]: Leveraging historical configurations for new tasks with cross-domain mixup and Artificial Neural Network (ANN) feature.
**Limitation:** ignores intricate feature interactions, lacks interpretability, presumes identical design spaces for source and target tasks.

[1] Jianwang Zhai, Yici Cai, and Bei Yu (2023). "Microarchitecture Power Modeling via Artificial Neural Network and Transfer Learning". In: *2023 28th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 1–6.

# Motivation

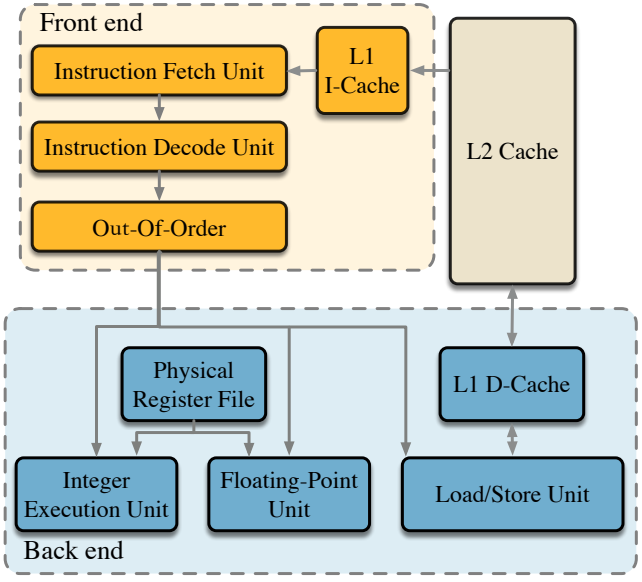
The characteristics of certain design features have similar or even identical effects on PPA values in different generations of a microarchitecture design.

Extract the relationship that is approximately invariant among different historical source design tasks, even equipped with different design spaces.

Transfer domain knowledge from different source tasks to the target design space exploration task.

# Preliminaries

# Typical Microprocessor Components




Visualization of typical microarchitecture pipeline.

# Invariant Risk Minimization (IRM)[2]

Prediction function $f' : \mathcal{X} \to \mathcal{Y} \Rightarrow \phi_{\boldsymbol{u}}(\cdot)$: feature extractor; $h_{\boldsymbol{w}}(\cdot)$: regressor.
$\boldsymbol{u}$, $\boldsymbol{w}$: parameter.
**IRM:**

$$\begin{aligned} \min_{\boldsymbol{u},\boldsymbol{w}} \quad & \sum_{s=1}^{S} \mathcal{R}^s(\boldsymbol{u},\boldsymbol{w}), \\ \text{s.t.} \quad & \boldsymbol{w} \in \operatorname*{argmin}_{\boldsymbol{w}^s} \mathcal{R}^s(\boldsymbol{u},\boldsymbol{w}^s), \end{aligned} \tag{1}$$

$\mathcal{R}^s(\boldsymbol{u},\boldsymbol{w}) = \mathbb{E}_{X^s,Y^s}[\mathcal{L}(f'(X^s),Y^s)]$: risk under $s$-th source task.
$\mathcal{L}(\cdot)$: the loss function.
**Simplified IRMv1:**

$$\min_{\boldsymbol{u},\boldsymbol{w}} \sum_{s=1}^{S} \mathcal{R}^s(\boldsymbol{u},\boldsymbol{w}) + \lambda ||\nabla_w \mathcal{R}^s(\boldsymbol{u},\boldsymbol{w})||^2, \tag{2}$$

[2]Martin Arjovsky et al. (2019). "Invariant risk minimization". In: *arXiv preprint arXiv:1907.02893*.

# Problem Definition (1)

## Source Task

Previously explored microarchitecture designs tasks.
Composed of the explored sample dataset $\mathcal{D}^s$ containing $n_s$ parameter vectors $X^s = [\boldsymbol{x}_1^s, \ldots, \boldsymbol{x}_{n_s}^s]^\top \in \mathbb{R}^{n_s \times d_s}$ and the evaluated PPA vectors $Y^s = [\boldsymbol{y}_1^s, \ldots, \boldsymbol{y}_{n_s}^s] \in \mathbb{R}^{n_s \times 3}$.

## Target Task

The new microarchitecture design space exploration task with only the set of legal parameter configuration vectors $X^t = [\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_{n_t}^t]^\top \in \mathbb{R}^{n_t \times d_t}$ available.

## Pareto Optimality

For a multi-objective minimization problem with $M$ objectives, $\boldsymbol{x}_1$ is deemed to dominate solution $\boldsymbol{x}_2$ ($\boldsymbol{x}_1 \succeq \boldsymbol{x}_2$) if

$$\begin{aligned} f_m(\boldsymbol{x}_1) &\leqslant f_m(\boldsymbol{x}_2), \forall m \in \{1, ..., M\}, \\ f_k(\boldsymbol{x}_1) &< f_k(\boldsymbol{x}_2), \exists k \in \{1, ..., M\}. \end{aligned} \tag{3}$$

**Pareto-optimal set:** The collection of solutions that remain non-dominated by others.

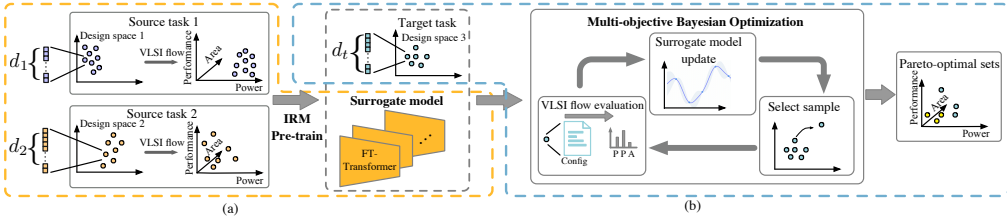
# Problem Definition (2)

## Microarchitecture Transferring Design Space Exploration

Given $S$ source tasks with explored datasets $\{\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^S\}$ and the target sample $\mathcal{D}^t$, the objective is to utilize the information from historical source tasks and improve the efficiency of finding a series of microprocessor design configurations $X$ in the target task that forms the Pareto optimality among the associated subset of $Y \subset \mathcal{Y}$, so that $X = \{\boldsymbol{x} | \boldsymbol{x} \succeq \boldsymbol{x}', \forall \boldsymbol{x}' \in X^t\}, Y = \{f(\boldsymbol{x}) | X \in X^t\}$.

# IT-DSE Algorithm

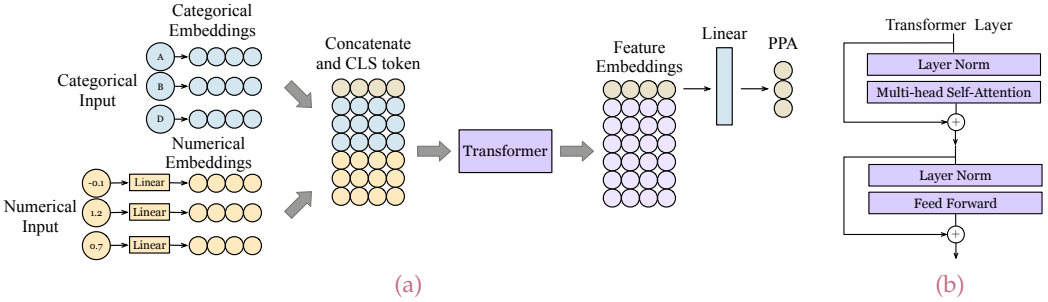
# Workflow




Workflow of proposed IT-DSE.

- Accommodate varying design spaces: FT-Transformer ensemble
- Knowledge fusion and warm start new task: pre-training with invariant risk minimization paradigm

# Customized Surrogate Model with Transformer




(a) FT-Transformer architecture. Firstly, Feature Tokenizer transforms input parameters into embeddings. The embedding is then processed by the Transformer module. (b) One Transformer layer.

$$\boldsymbol{t}_i^{num} = x_i^{num} \cdot \boldsymbol{w}_i^{num} + b_i^{num}, \tag{4}$$

$$\boldsymbol{t}_j^{cat} = \boldsymbol{e}_j^{\top} \boldsymbol{w}_j^{cat} + b_j^{cat}, \tag{5}$$

$$\boldsymbol{T} = \text{stack}(\texttt{[CLS]}, \boldsymbol{t}_1^{num}, \ldots \boldsymbol{t}_{d_n}^{num}, \boldsymbol{t}_1^{cat}, \ldots, \boldsymbol{t}_{d_c}^{cat}). \tag{6}$$

# Pre-training on Multiple Source Tasks

Comprehensive union of extracted features:

Source task 1: $\boldsymbol{x}^1 = [x_1, x_2]$, source task 2: $\boldsymbol{x}^2 = [x_2, x_3, x_4]$,

Embedding weight: $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3, \boldsymbol{w}_4]^\top \in \mathbb{R}^{4\times d}$,

Embedding bias: $\boldsymbol{B} = [b_1, b_2, b_3, b_4]^\top \in \mathbb{R}^{4\times d}$,

Transformed embeddings: $\boldsymbol{T}^1 = \text{stack}(\texttt{[CLS]}, \boldsymbol{t}_1, \boldsymbol{t}_2)$, $\boldsymbol{T}^2 = \text{stack}(\texttt{[CLS]}, \boldsymbol{t}_2, \boldsymbol{t}_3, \boldsymbol{t}_4)$.

**Focusing more on task-invariant features.**

For $s$-th source task, original data: $\mathcal{D}^s = \{X^s, Y^s\} = \{\boldsymbol{x}_i^s, \boldsymbol{y}_i^s\}_{i=1}^{n_s}$,

transferred data: $\mathcal{T}^s = \{\phi_{\boldsymbol{u}}(\boldsymbol{x}_i^s), \boldsymbol{y}_i^s\}_{i=1}^{n_s}$,

collection of transformed datasets of all $S$ source tasks: $\mathcal{T}^c = \cup_{s=1}^{S} \mathcal{T}^s$.

**Objective for Bayesian IRM**:

$$\max_{\boldsymbol{u}} \sum_{s} \mathbb{E}_{g_{u(w)}}[\ln p(\mathcal{T}^s|\boldsymbol{w}, \boldsymbol{u})] + \lambda \left( \mathbb{E}_{g_{u(w)}}[\ln p(\mathcal{T}^s|\boldsymbol{w}, \boldsymbol{u})] - \mathbb{E}_{g_{\boldsymbol{u}}^s(\boldsymbol{w}^s)}[\ln p(\mathcal{T}^s|\boldsymbol{w}^s, \boldsymbol{u})] \right). \tag{7}$$

$g_{\boldsymbol{u}}(\boldsymbol{w}) \approx p(\boldsymbol{w}|\mathcal{T}^c)$, $g_{\boldsymbol{u}}^s(\boldsymbol{w}^s) \approx p(\boldsymbol{w}^s|\mathcal{T}^s)$.

**First term**: Encouraging $\boldsymbol{u}$ to retain information for data distribution fitting.

**Second term**: Requires transformed data distribution to be stable among different tasks with feature extractor $\phi_{\boldsymbol{u}}(\cdot)$.

# Adaptive Solution of Bayesian IRM

**The actual posterior $p(\boldsymbol{w}|\mathcal{T}^c)$ and $p(\boldsymbol{w}^s|\mathcal{T}^s)$ are hard to estimate in large models.** Variational inference: approximate the posterior distributions by maximizing the evidence lower bound (ELBO).

$$g_{\boldsymbol{u}}^s(\boldsymbol{w}^s) = \underset{g' \in \mathcal{G}}{\operatorname{argmax}}\, \mathbb{E}_{g'}[\ln p(\mathcal{T}^s|\boldsymbol{w}, \boldsymbol{u}) - \mathrm{D}_{\mathrm{KL}}(g'||p_0(\boldsymbol{w}))], \tag{8}$$

$$g_{\boldsymbol{u}}(\boldsymbol{w}) = \underset{g'}{\operatorname{argmax}} \sum_{s=1}^{S} \mathbb{E}_{g' \in \mathcal{G}}[\ln p(\mathcal{T}^s|\boldsymbol{w}, \boldsymbol{u}) - \mathrm{D}_{\mathrm{KL}}(g'||p_0(\boldsymbol{w}))], \tag{9}$$

Assume $g_{\boldsymbol{u}}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $g_{\boldsymbol{u}}^s(\boldsymbol{w}^s)$ approaches $g_{\boldsymbol{u}}(\boldsymbol{w})$ when more task-invariant features are extracted:

$$g_{\boldsymbol{u}}^s(\boldsymbol{w}^s) = \mathcal{N}(\boldsymbol{\mu} - \nabla_{\boldsymbol{\mu}} \mathbb{E}_{g_{\boldsymbol{u}}(\boldsymbol{w})} \ln p(\mathcal{T}^s|\boldsymbol{w}, \boldsymbol{u}), \boldsymbol{\Sigma}), \tag{10}$$

Use Monte Carlo samples to estimate penalty term in Equation (7):

$$\boldsymbol{w} = \boldsymbol{\mu} + \epsilon\boldsymbol{\Sigma}, \quad \boldsymbol{w}^s = \boldsymbol{\mu}^s + \epsilon\boldsymbol{\Sigma}^s. \tag{11}$$

# Algorithm of Bayesian IRM

**Algorithm** Feature extraction with Bayesian IRM

**Require:** Feature extractor $\phi_{\boldsymbol{u}}$, regressor $h_{\boldsymbol{w}}$, collection of data from $S$ source task $\{\mathcal{D}^s\}_{s=1}^{S}$
**Ensure:** the learned task-invariant feature extractor $\phi_{\boldsymbol{u}}$, regressor $h_{\boldsymbol{w}}$.

Initialization: prior $\phi_0 \leftarrow \mathcal{N}(\mathbf{0}, \sigma \boldsymbol{I})$.
**repeat**
  **for** $s$ in $1, \ldots, S$ **do**
    Sample a batch of data $(X^s_{batch}, Y^s_{batch})$ from $\mathcal{D}^s$;
    Transform data $\mathcal{T}^s_{batch} \leftarrow (\phi_{\boldsymbol{u}}(X^s_{batch}), Y^s_{batch})$;
  **end for**
  Update $g_{\boldsymbol{u}}(\boldsymbol{w})$; ▷ Equation (9)
  **for** $s$ in $1, \ldots, S$ **do**
    Update $g^s_{\boldsymbol{u}}(\boldsymbol{w}^s)$; ▷ Equation (10)
  **end for**
  Sample $g_{\boldsymbol{u}}(\boldsymbol{w})$ and $g^s_{\boldsymbol{u}}(\boldsymbol{w}^s)$; ▷ Equation (11)
  Update $\boldsymbol{u}$ to maximize objective; ▷ Equation (7)
**until** Training finished.
Return $\phi_{\boldsymbol{u}}$, $h_{\boldsymbol{w}}$;

# Target Task DSE in Feature Space: Deep Ensemble Surrogate Model

Given $n$ evaluated samples $\mathcal{D}^t = \{X^t, f'(X^t)\}$ from target task,
$f'(X^t) = \{(f'(\boldsymbol{x}_1^t))^\top, f'((\boldsymbol{x}_2^t))^\top, \ldots, f((\boldsymbol{x}_n^t))^\top\}^\top \in \mathbb{R}^{n\times 3}$,
For new point $\boldsymbol{x}_*^t$, posterior distribution $p(\boldsymbol{y}_*^t|\boldsymbol{x}_*^t)$ is approximated as a Gaussian distributions:

$$p(\boldsymbol{y}_*^t|\boldsymbol{x}_*^t) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{y}_*^t|f'(X^t)}, \boldsymbol{\sigma}^2_{\boldsymbol{y}_*^t|f'(X^t)}\right), \tag{12}$$

The posterior mean and variance:

$$\boldsymbol{\mu}_{\boldsymbol{y}_*^t|f'(X^t)} = \frac{1}{M}\sum_{m=1}^{M}\boldsymbol{\mu}_{\theta_m}(\boldsymbol{x}_*^t), \tag{13}$$

$$\begin{aligned}\boldsymbol{\sigma}^2_{\boldsymbol{y}_*^t|f'(X^t)} = &\frac{1}{M}\sum_{m=1}^{M}(\boldsymbol{\mu}_{\boldsymbol{y}_*^t|f'(X^t)} - \boldsymbol{\mu}_{\theta_m}(\boldsymbol{x}_*^t))^2 \\ &+ \frac{1}{M}\sum_{m=1}^{M}\boldsymbol{\sigma}^2_{\theta_m}(\boldsymbol{x}_*^t).\end{aligned} \tag{14}$$

# Target Task DSE in Feature Space: Pareto EHVI Acquisition Function[3]

Hypervolume given an approximate Pareto frontier $\mathcal{P}(\boldsymbol{y})$:

$$HV_{\boldsymbol{y}_{ref}}(\mathcal{P}(\boldsymbol{y})) = \lambda_M(\cup_{\boldsymbol{y}\in\mathcal{P}(\boldsymbol{y})}[\boldsymbol{y}, \boldsymbol{y}_{ref}]), \tag{15}$$

$[\boldsymbol{y}, \boldsymbol{y}_{ref}]$: hyperrectangle bounded by $\boldsymbol{y}$ and reference point $\boldsymbol{y}_{ref}$.
Hypervolume improvement (HVI) for a new given point $\boldsymbol{y}_*$ given $\mathcal{P}(\boldsymbol{y})$:

$$HVI(\boldsymbol{y}_*|\mathcal{P}(\boldsymbol{y}), \boldsymbol{y}_{ref}) = V_{\boldsymbol{y}_{ref}}(\mathcal{P}(\boldsymbol{y}\cup\boldsymbol{y}_*)) - V_{\boldsymbol{y}_{ref}}(\mathcal{P}(\boldsymbol{y})). \tag{16}$$

EHVI Monte Carlo integration approximation with $K$ sampling times:

$$\begin{aligned} EHVI(\hat{\boldsymbol{x}}) &= \mathbb{E}_{P(f'(\hat{\boldsymbol{x}}_*)|\mathcal{D})}\left(HVI(f'(\hat{\boldsymbol{x}})|\mathcal{P}(\boldsymbol{y}), \boldsymbol{y}_{ref})\right) \\ &\approx \frac{1}{n_k}\sum_{k=1}^{K} HVI(\tilde{f}'_k(\hat{\boldsymbol{x}})|\mathcal{P}(\boldsymbol{y}), \boldsymbol{y}_{ref}). \end{aligned} \tag{17}$$

$\tilde{f}'_k$: $k$-th function evaluation from distribution $p(f'(\boldsymbol{x}|\mathcal{D}))$.

[3] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy (2020). "Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization". In: *Advances in Neural Information Processing Systems* 33, pp. 9851–9864.

# Experiments

# Experimental Settings

- Experiments are performed on an in-house 64-bit high-performance commercial microprocessor.
- VLSI flow is used to evaluate the PPA values of each microarchitecture configuration.
- Register-transfer-level (RTL) design: generated from Chisel[4].
- Performance and power: obtained from the benchmark simulation after the physical implementation.
- Area value: reported from the physical implementation tool.
- Due to the commercial confidentiality of microarchitecture, we normalize the original PPA values.

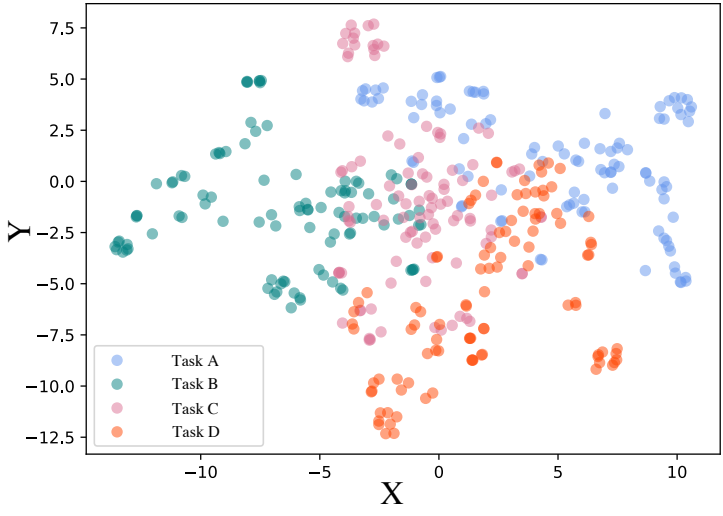
[4]Jonathan Bachrach et al. (2012). "Chisel: Constructing Hardware in a Scala Embedded Language". In: *ACM/IEEE Design Automation Conference (DAC)*, pp. 1216–1225.

# Transfer Design Space Exploration Experiment Setup

Table: Statistics of our microarchitecture design space

| Module | # Linear | #Pow | # Categorical | # Combinations |
|---|---|---|---|---|
| IFU | 8 | 4 | 0 | $\sim 4 \times 10^8$ |
| OoO | 11 | 0 | 0 | $\sim 5 \times 10^9$ |
| IEX | 12 | 0 | 3 | $\sim 1 \times 10^9$ |
| FSU | 5 | 0 | 0 | $\sim 2 \times 10^3$ |
| LSU | 6 | 3 | 0 | $\sim 1 \times 10^7$ |
| L2C | 1 | 2 | 0 | $\sim 8 \times 10^2$ |
| Overall | 43 | 9 | 3 | $\sim 3 \times 10^{40}$ |

- 55 parameters with more than $3 \times 10^{40}$ combinations.
- Task A, task B, and Task C share the same design space, for task D, we can only tune 53 parameters out of 55 parameters listed.
- Task A, task B, task C, and Task D contain 1237, 377, 1835 and 3453 evaluated samples, respectively.




Data distribution for 4 tasks.

# Evaluation Metrics:

Hypervolume:

$$HV_{\boldsymbol{y}_{ref}}(\mathcal{P}(\boldsymbol{y})) = \lambda_M(\cup_{\boldsymbol{y}\in\mathcal{P}(\boldsymbol{y})}[\boldsymbol{y}, \boldsymbol{y}_{ref}]), \tag{18}$$

Average distance to reference set (ADRS):

$$ADRS(\mathcal{P}^*, \mathcal{P}) = \frac{1}{|\mathcal{P}^*|}\sum_{\boldsymbol{\alpha}\in\mathcal{P}^*}\min_{\boldsymbol{\beta}\in\mathcal{P}} l_2(\boldsymbol{\alpha}, \boldsymbol{\beta}), \tag{19}$$

# Transferring Performance within Same Design Space

Table: Comparison of transfer performance in same design space

| Methodologies | A, B → C | | A, C → B | | B, C → A | |
|---|---|---|---|---|---|---|
| | ADRS | HV | ADRS | HV | ADRS | HV |
| Ground Truth | 0.0 | 0.0984 | 0.0 | 0.0684 | 0.0 | 0.0809 |
| ANN-TL[5] | 0.069 | 0.0891 | 0.045 | 0.0643 | 0.031 | 0.0749 |
| Deep-Ens[6] | 0.072 | 0.0840 | 0.066 | 0.0599 | 0.055 | 0.0727 |
| ERM[7] | 0.080 | 0.0877 | 0.064 | 0.0629 | 0.043 | 0.0742 |
| IRMv1[8] | 0.025 | 0.0938 | 0.023 | 0.0667 | 0.027 | 0.0766 |
| Ours | **0.021** | **0.0944** | **0.017** | **0.0679** | **0.020** | **0.0796** |

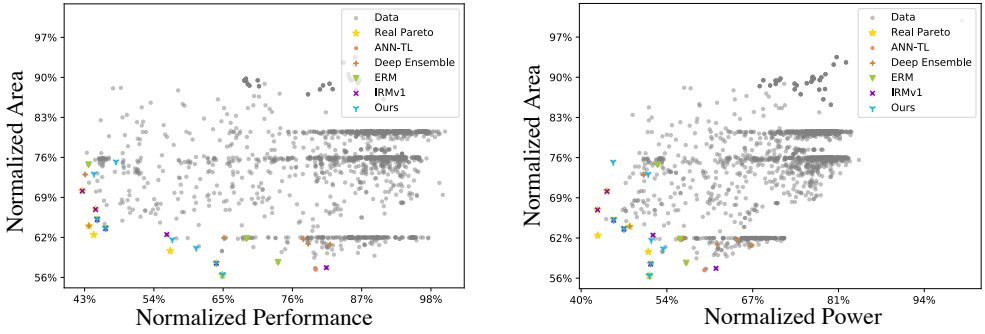
[5]Jianwang Zhai, Yici Cai, and Bei Yu (2023). "Microarchitecture Power Modeling via Artificial Neural Network and Transfer Learning". In: *2023 28th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 1–6.

[6]Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell (2017). "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Advances in neural information processing systems* 30.

[7]VN Vapnik (1992). *Principles of risk minimization for learning theory, Advances in Neural Information Processing NIPS 4 (pp. 831±838)*.

[8]Martin Arjovsky et al. (2019). "Invariant risk minimization". In: *arXiv preprint arXiv:1907.02893*.

# Pareto-optimal Sets Visulation




Pareto-optimal sets using source task data from same design space. Left: performance versus area; right: power versus area.

# Transferring Performance within Different Design Space

To the best of our best knowledge, all previous methods cannot spread knowledge across different design spaces.
This serves as an ablation study.

Table: Comparison of transfer performance in different design spaces

| | A, D $\rightarrow$ C | | A, C $\rightarrow$ D | |
|---|---|---|---|---|
| Methodologies | ADRS | HV | ADRS | HV |
| Ground Truth | 0.0 | 0.0984 | 0.0 | 0.7792 |
| w/o. Pre-train | 0.0533 | 0.0857 | 0.0853 | 0.7465 |
| w/o. Ensemble | 0.0275 | 0.0892 | 0.0722 | 0.7531 |
| w/o. IRM | 0.0293 | 0.0910 | 0.0701 | 0.7569 |
| w/. IRMv1 | 0.0232 | 0.0914 | 0.0687 | 0.7602 |
| Ours | **0.0217** | **0.0924** | **0.0641** | **0.7624** |

# THANK YOU!