Learning A Continuous and Reconstructible Latent Space for Hardware Accelerator Design

Qijing Huang **Charles Hong**
John Wawrzynek Mahesh Subedar Yakun Sophia Shao

jennyhuang@nvidia.com, charleshong@berkeley.edu
https://github.com/hqjenny/vaesa.git

Berkeley UNIVERSITY OF CALIFORNIA intel

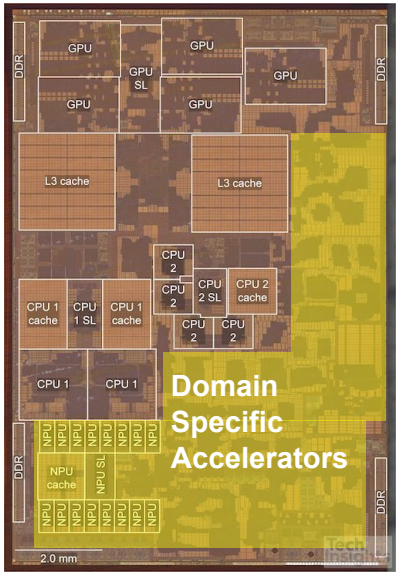
Motivation: Hardware acceleration is everywhere

Hardware acceleration is the driving force for many innovations.

Robots

Drones

Mobile phones



Apple A15

Augmented Reality

Autonomous Vehicles

Genomics

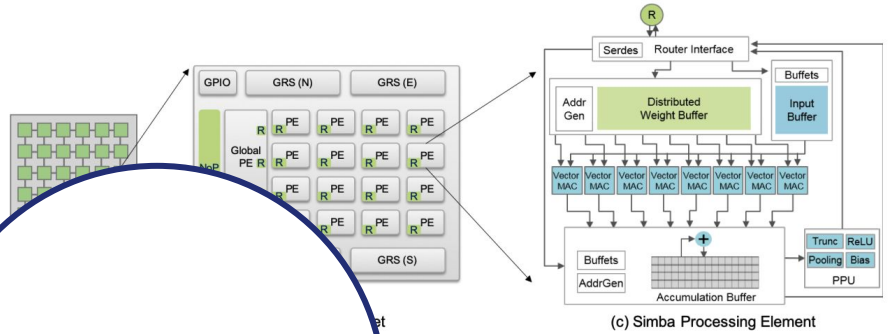
*TechInsights.com Apple iPhone 13 teardown

Motivation: Designing accelerators is challenging

Hardware design space exploration (DSE) challenges:

1. High-dimensional and discrete
2. Multi-objective and nonlinear
3. Costly

Challenge #1: High-dimensional and discrete



(c) Simba Processing Element

from package to processing element (PE).

Hardware Designs

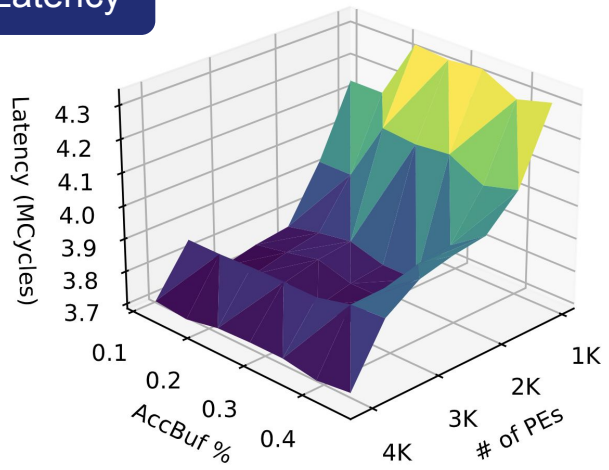
$\sim 10^{17}$

Parameter	Max	# of Possible Values
# of PEs	64	5
# of MAC units	4096	64
Accum. buffer size	96 KB	128
Weight buffer size	8 MB	32768
Input buffer size	256 KB	2048
Global buffer size	256 KB	131072

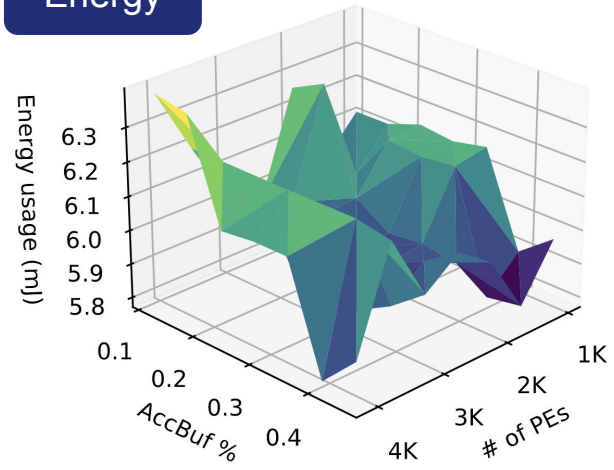
From Shao et al. “Simba: Scaling Deep-Learning Inference with Multi-Chip-Module-Based Architecture.” MICRO 2019.

Challenge #2: Multi-objective and nonlinear

Latency

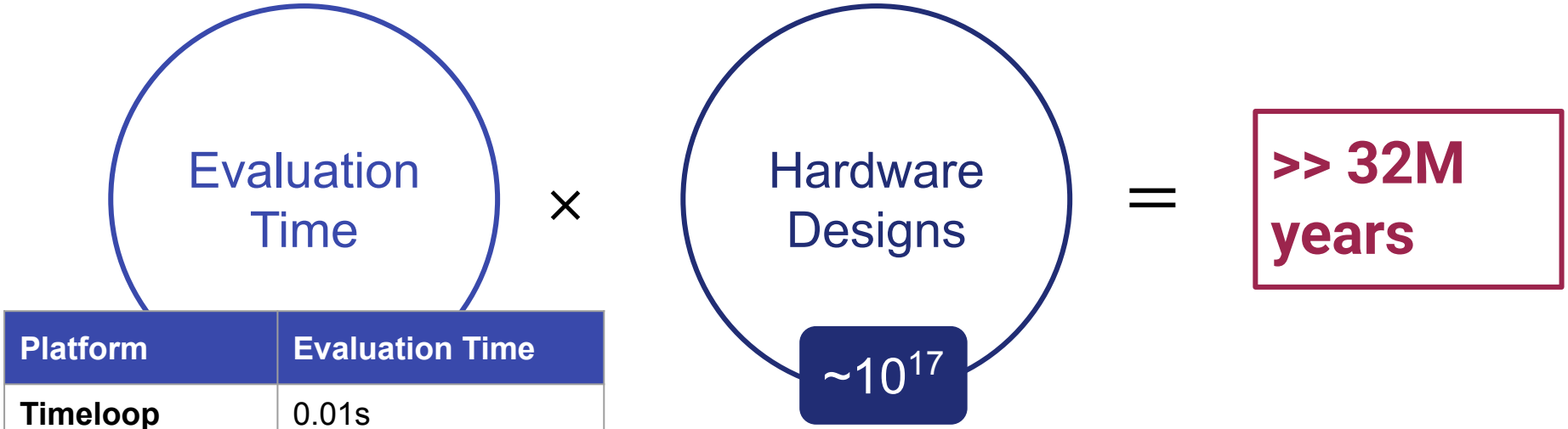


Energy



Performance of ResNet-50 as # of PEs and accumulation buffer size change

Challenge #3: Costly



Platform	Evaluation Time
Timeloop	0.01s
VCS	10 mins
FPGA	2 mins

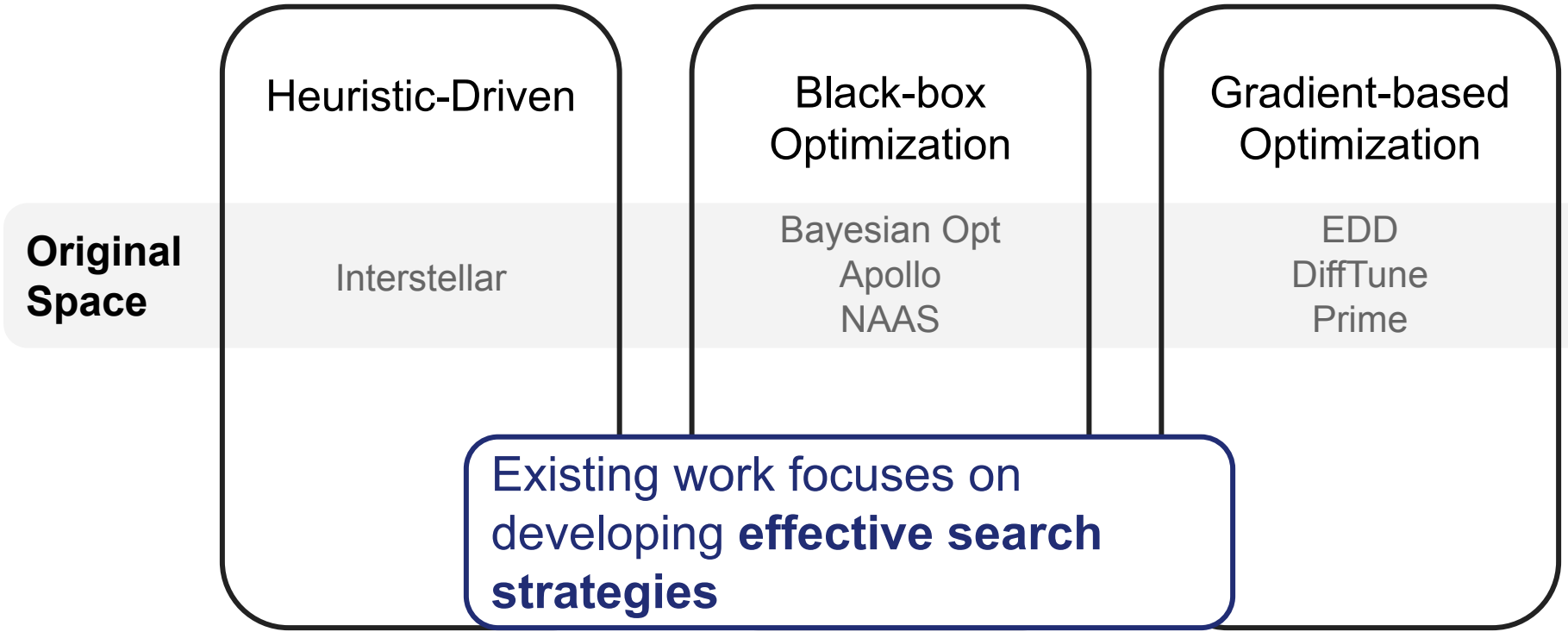
Problem Statement

How can we efficiently navigate the accelerator design space for deep learning algorithms?

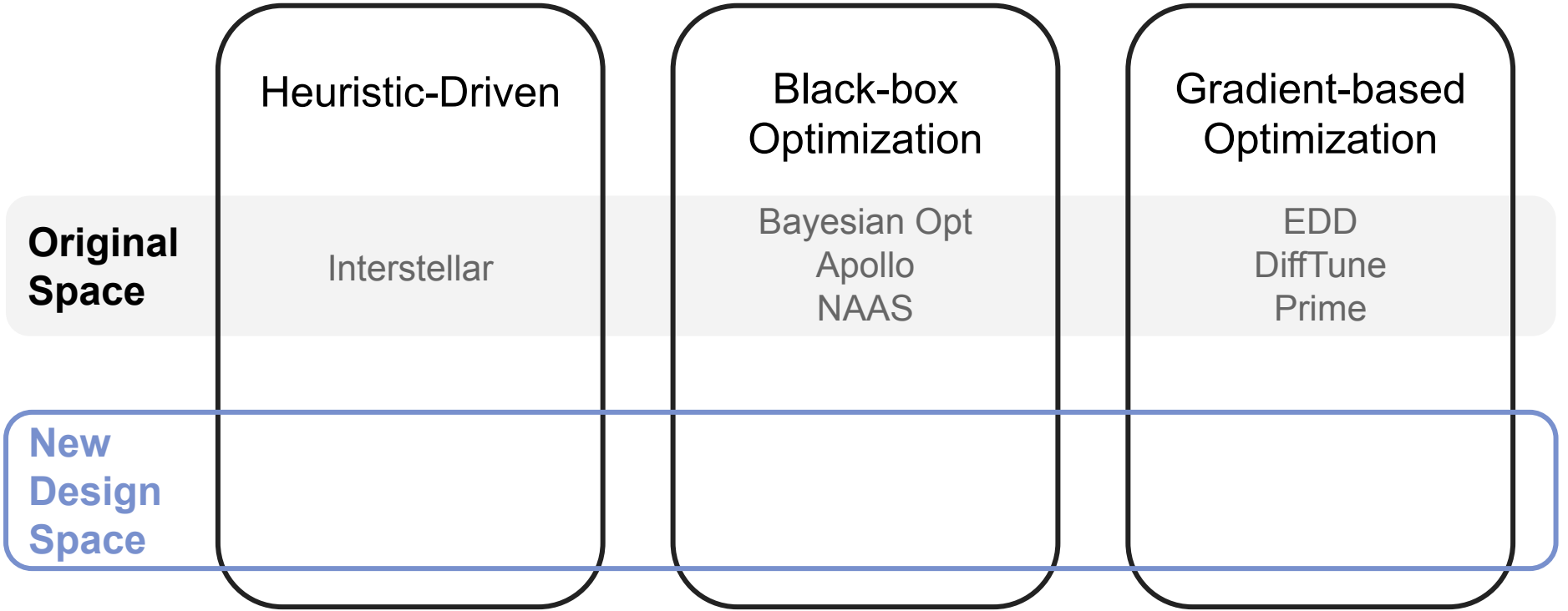
Prior work: Search strategy oriented

Heuristic-Driven	Black-box Optimization	Gradient-based Optimization
Interstellar	Bayesian Opt Apollo NAAS	EDD DiffTune Prime

Prior work: Search strategy oriented

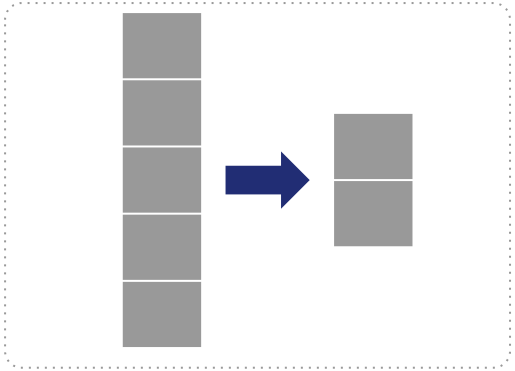


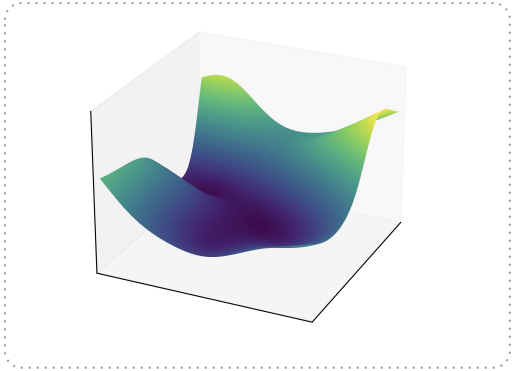
Prior work: Search strategy oriented

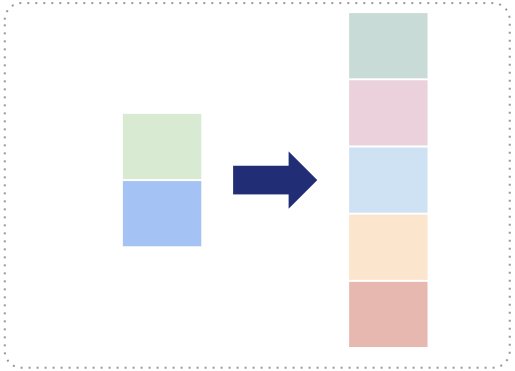


Desirable hardware design space properties

1. Reduced dimensionality
2. Smooth surface
3. Reconstructible



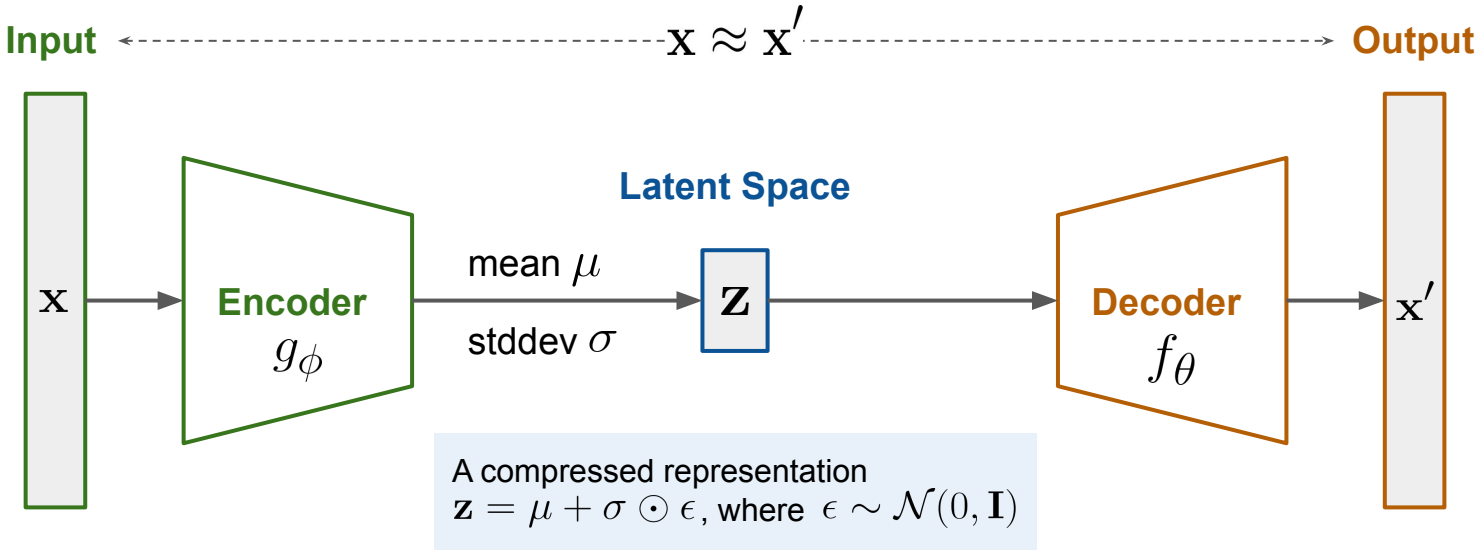




Variational Autoencoder (VAE)

Background: Variational Autoencoder (VAE)

A **model** that learns a compressed representation $\mathbf{z}$ of input data $\mathbf{x}$

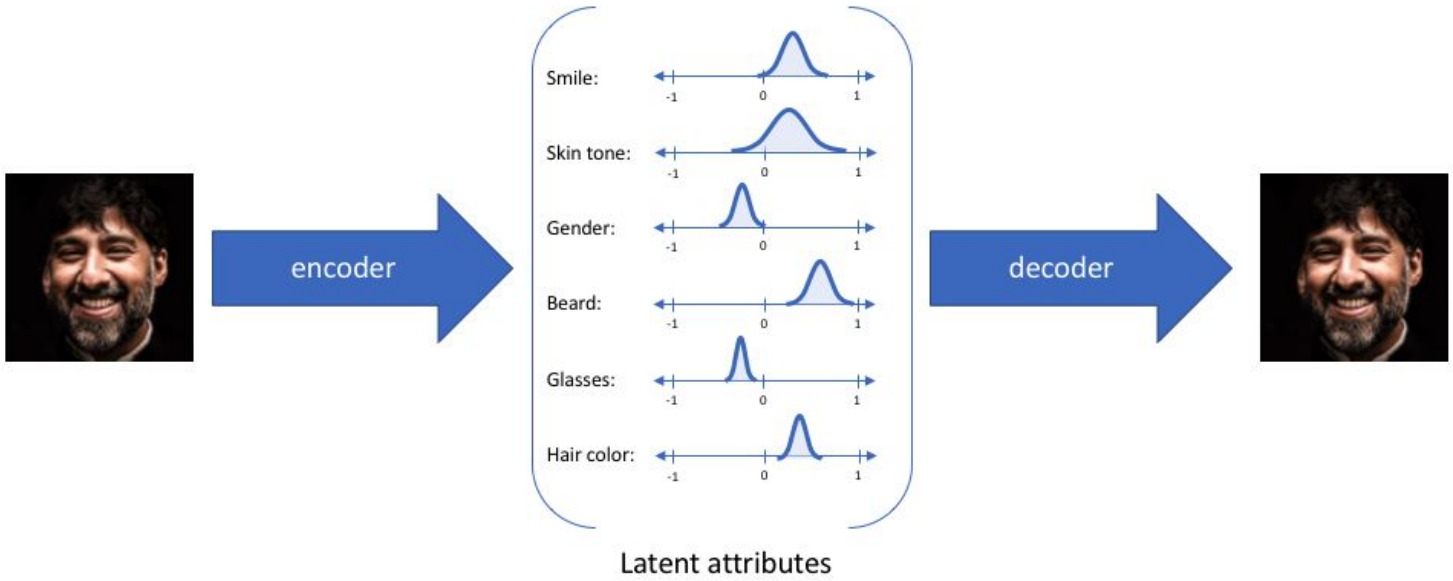


- Training minimizes reconstruction error, regularizes μ and σ towards the standard normal

Diederik P Kingma, Max Welling. “Auto-Encoding Variational Bayes”. ICLR 2014.

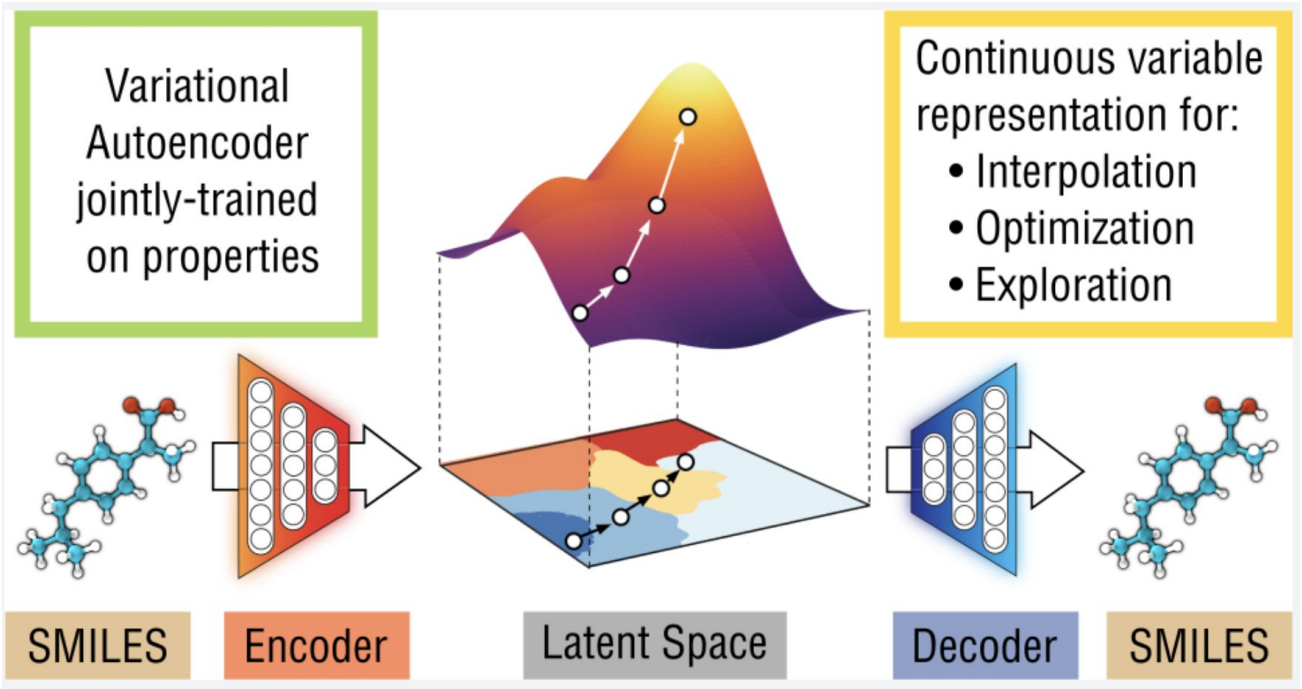
Background: Variational Autoencoder (VAE)

- Learns underlying (latent) features by identifying structure in data



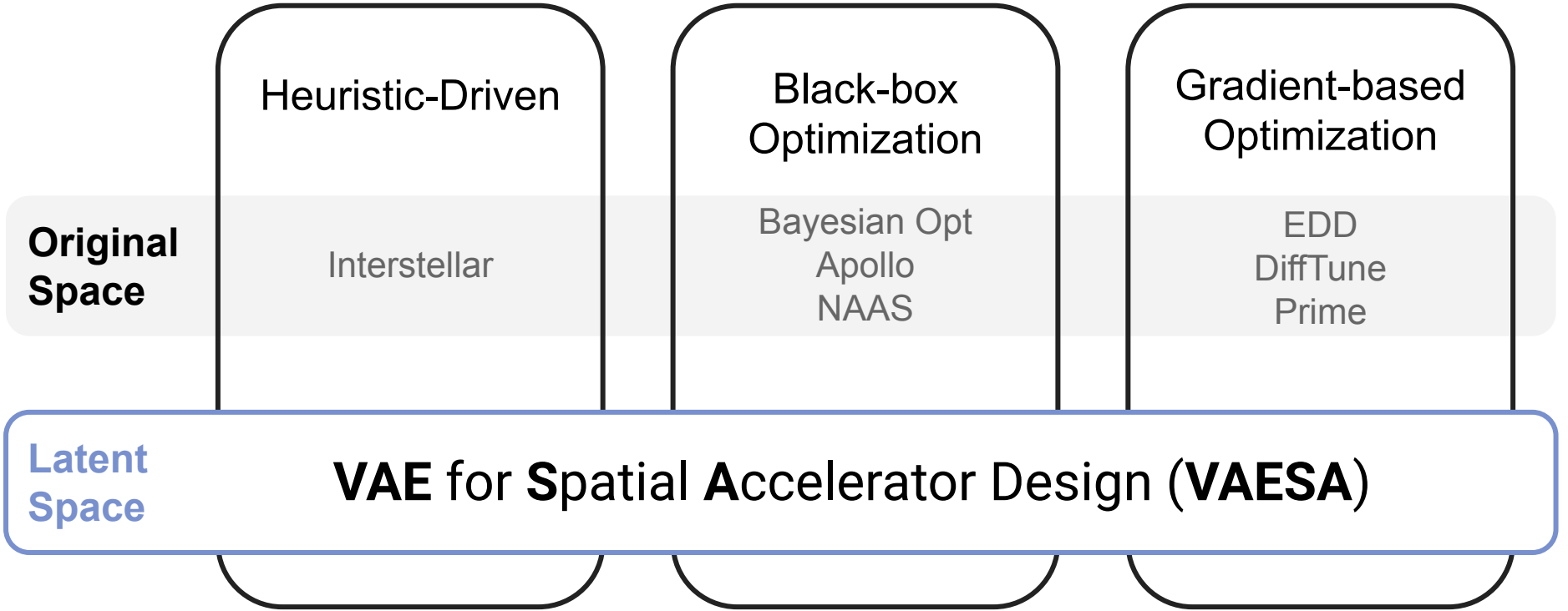
Jeremy Jordan. “Variational Autoencoders.” 2018.

VAE Application: Chemical Design

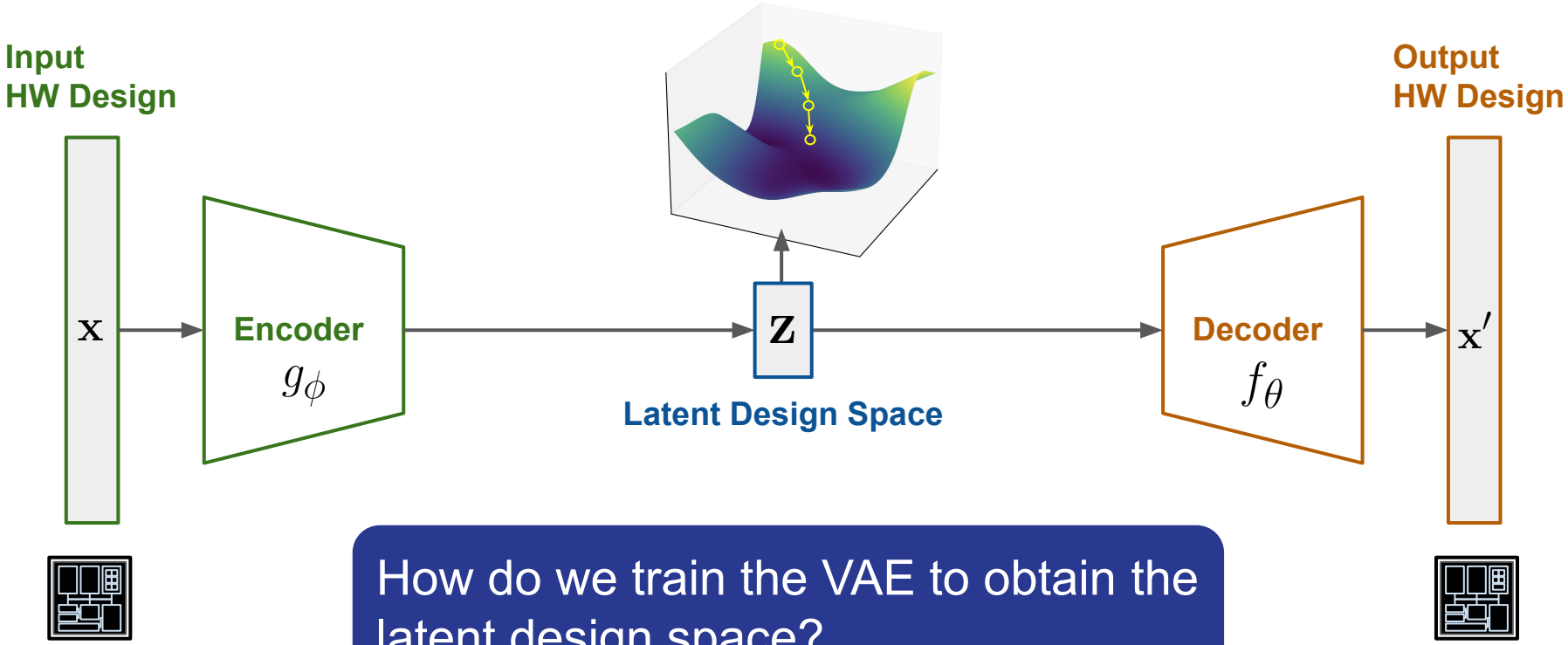


Gómez-Bombarelli et al. "Automatic chemical design using a data-driven continuous representation of molecules." *ACS Central Science*, 2018.

Our work: Search space oriented

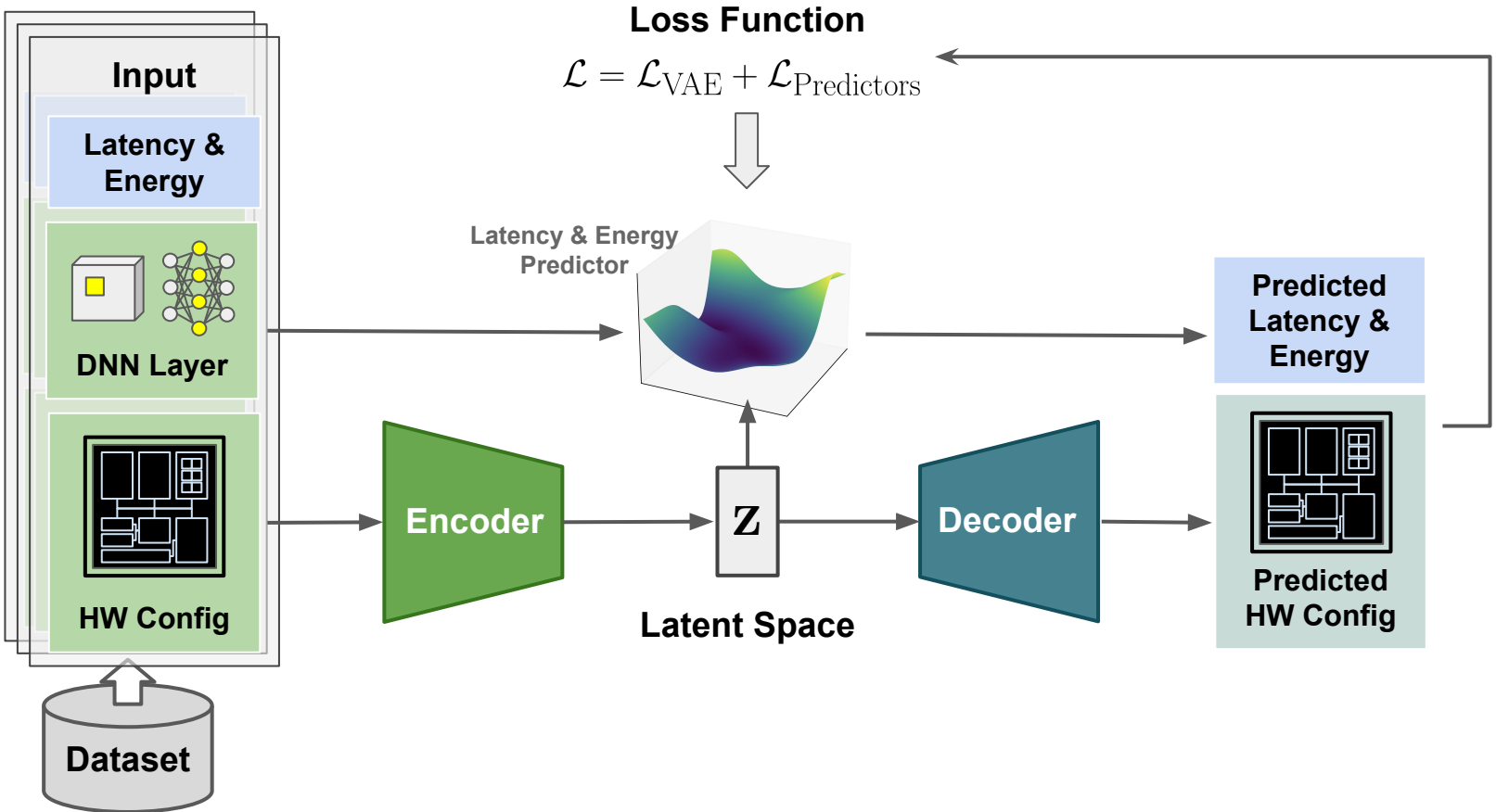


Our Framework - VAESA



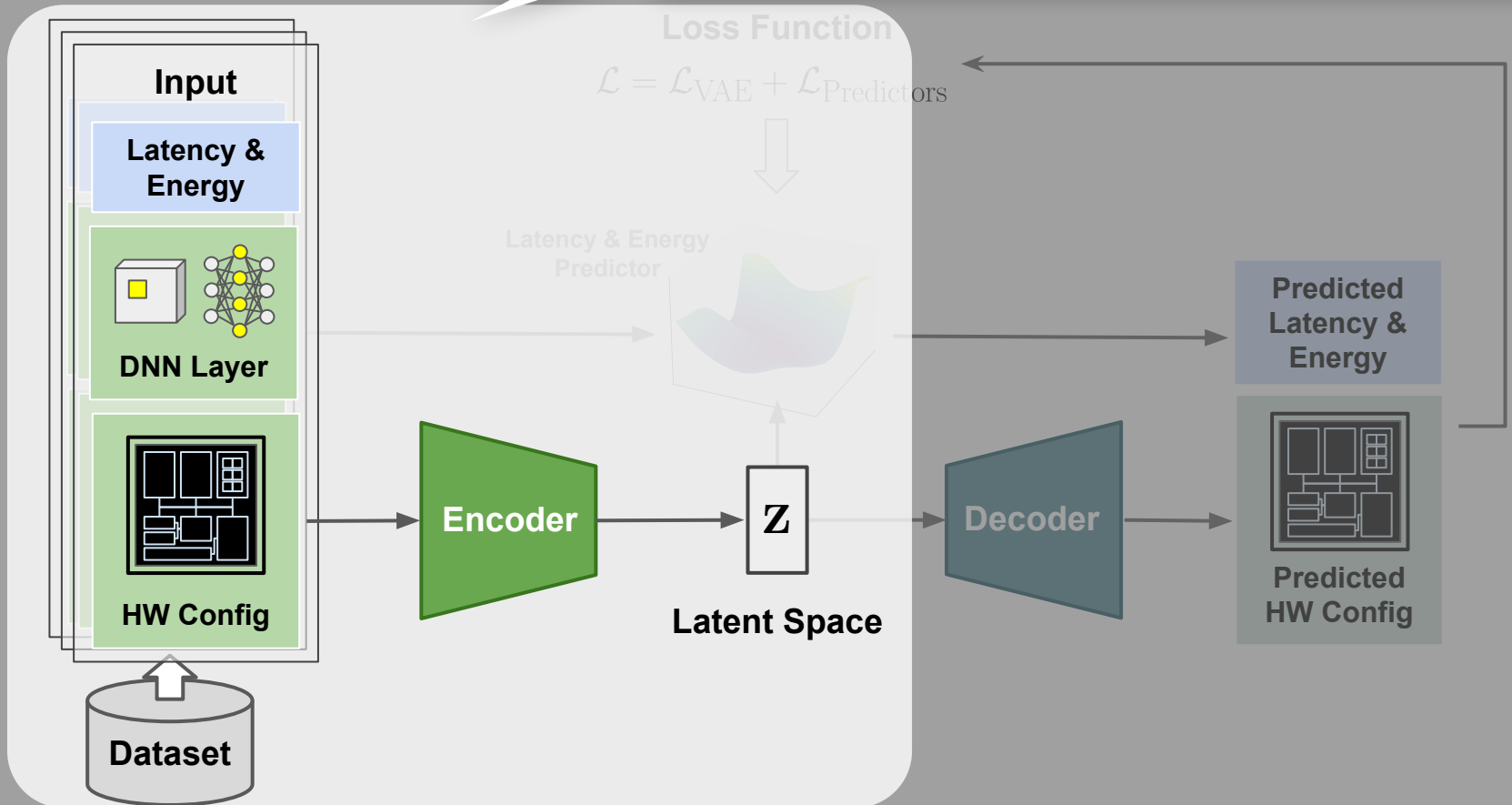
How do we train the VAE to obtain the latent design space?

VAESA Training

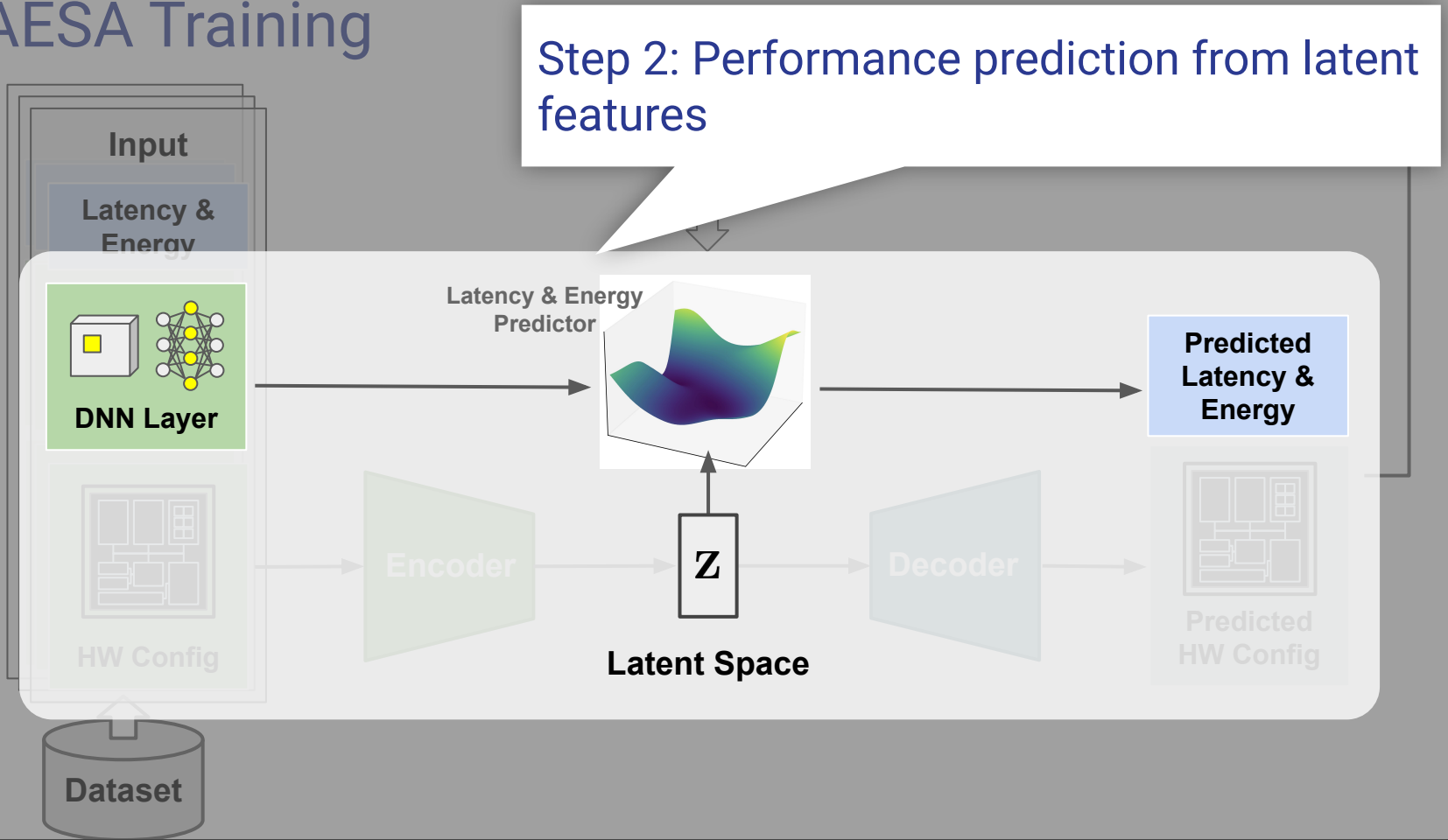


VAESA Training

Step 1: Encode to a compact, continuous search space

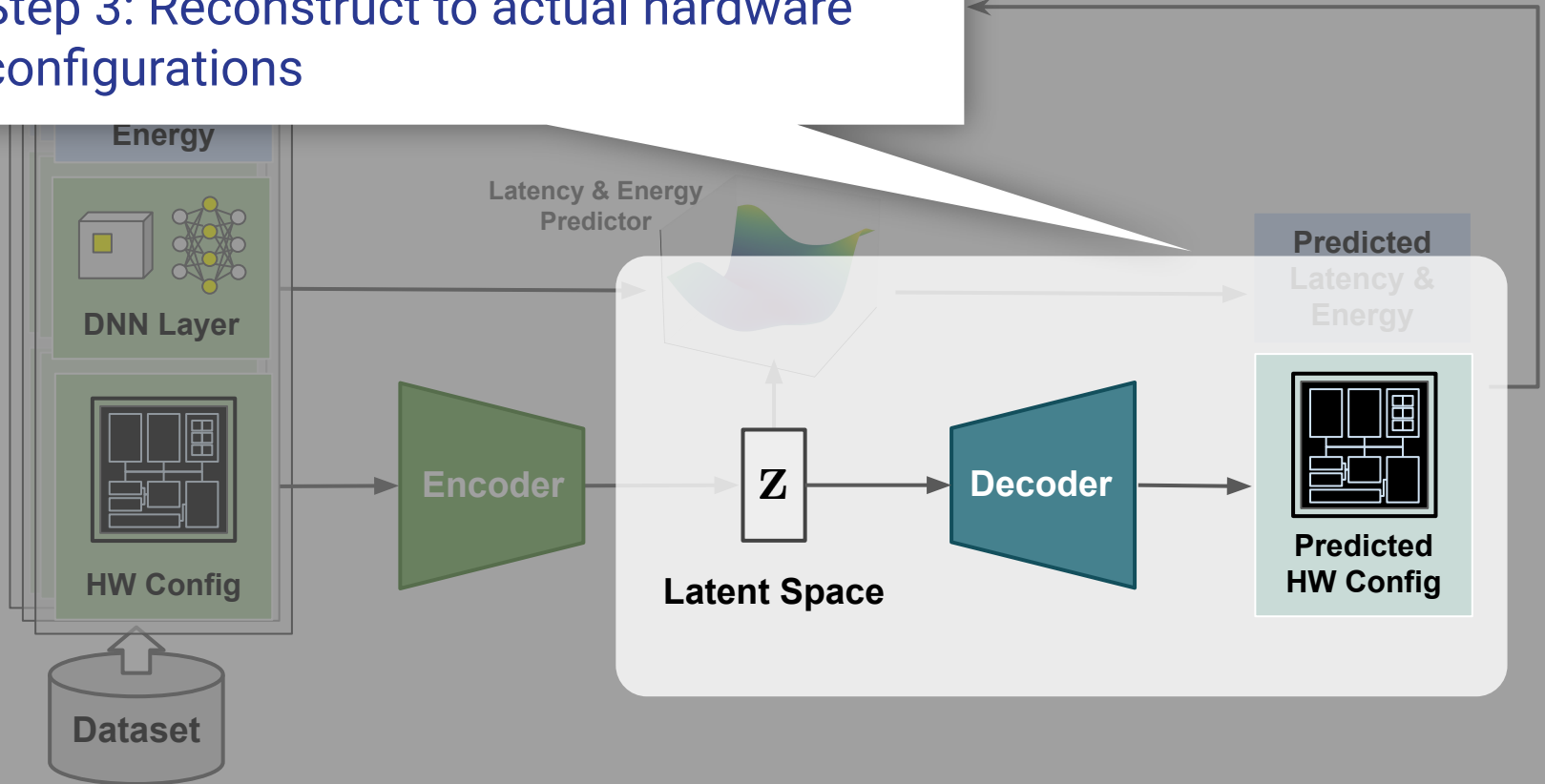


VAESA Training

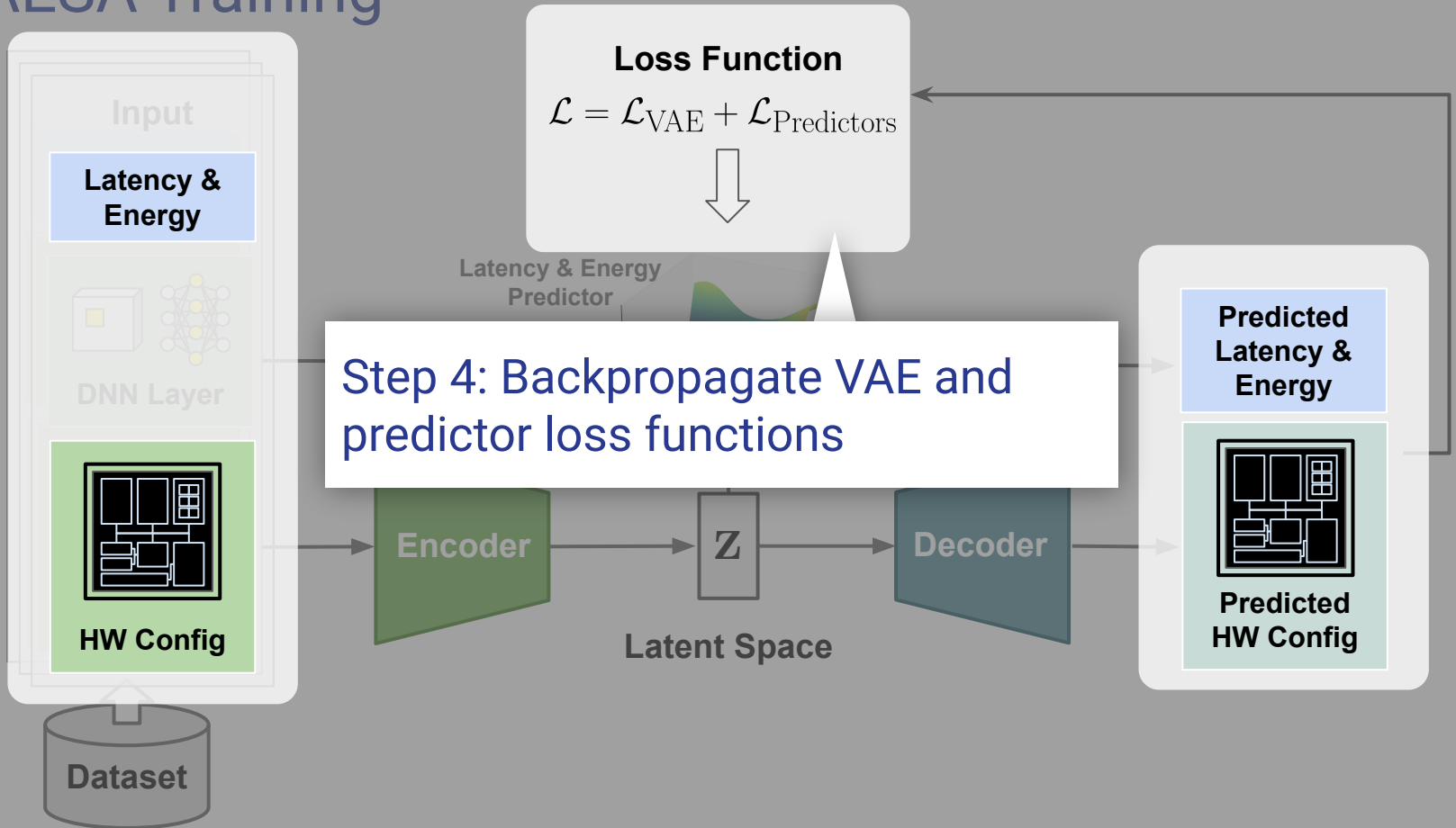


VAESA Training

Step 3: Reconstruct to actual hardware configurations



VAESA Training

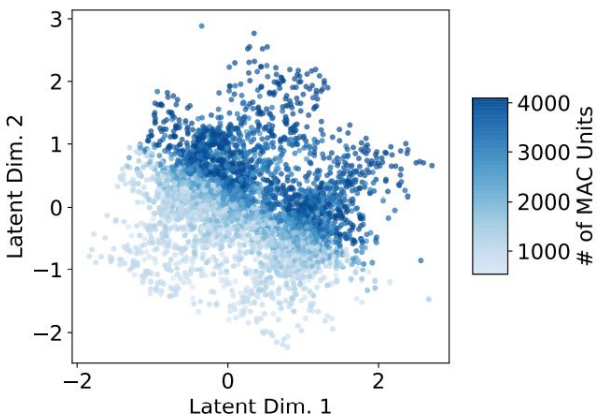


Step 4: Backpropagate VAE and predictor loss functions

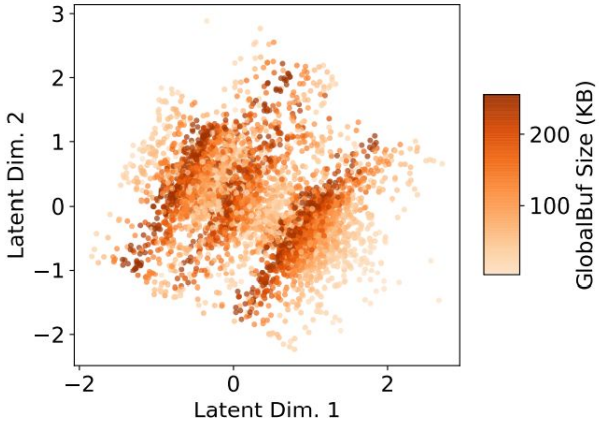
VAESA Visualization (2D)

Learned latent space

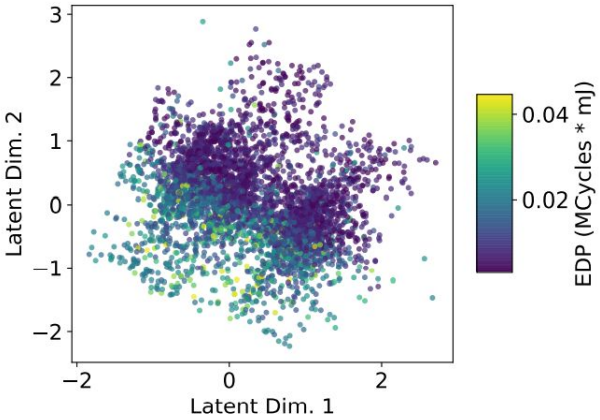
a) Number of MAC units



b) Global buffer size



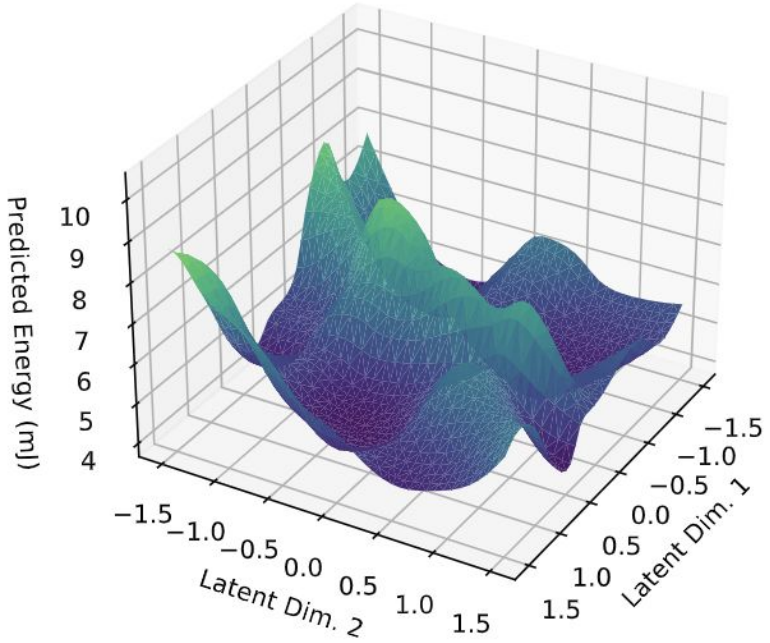
c) Energy-delay product*



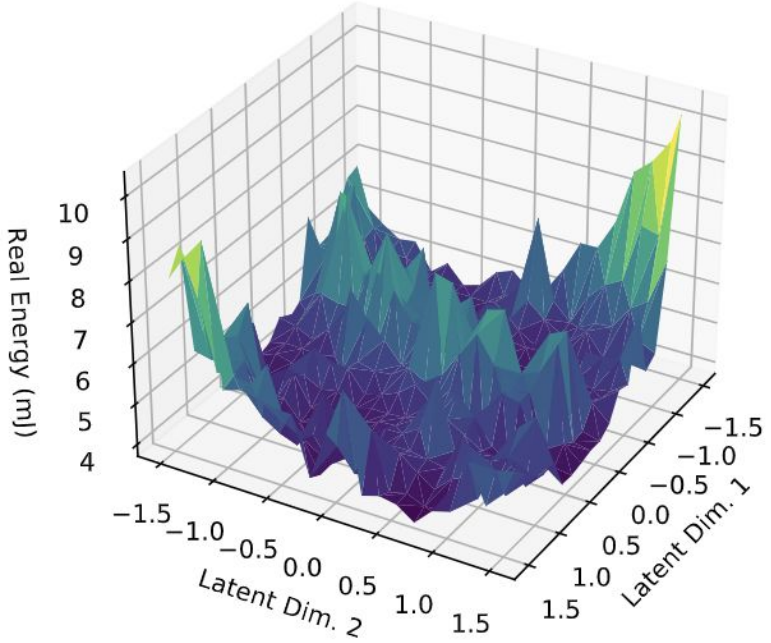
* on ResNet-50

VAESA Visualization (2D)

Predicted performance: Energy



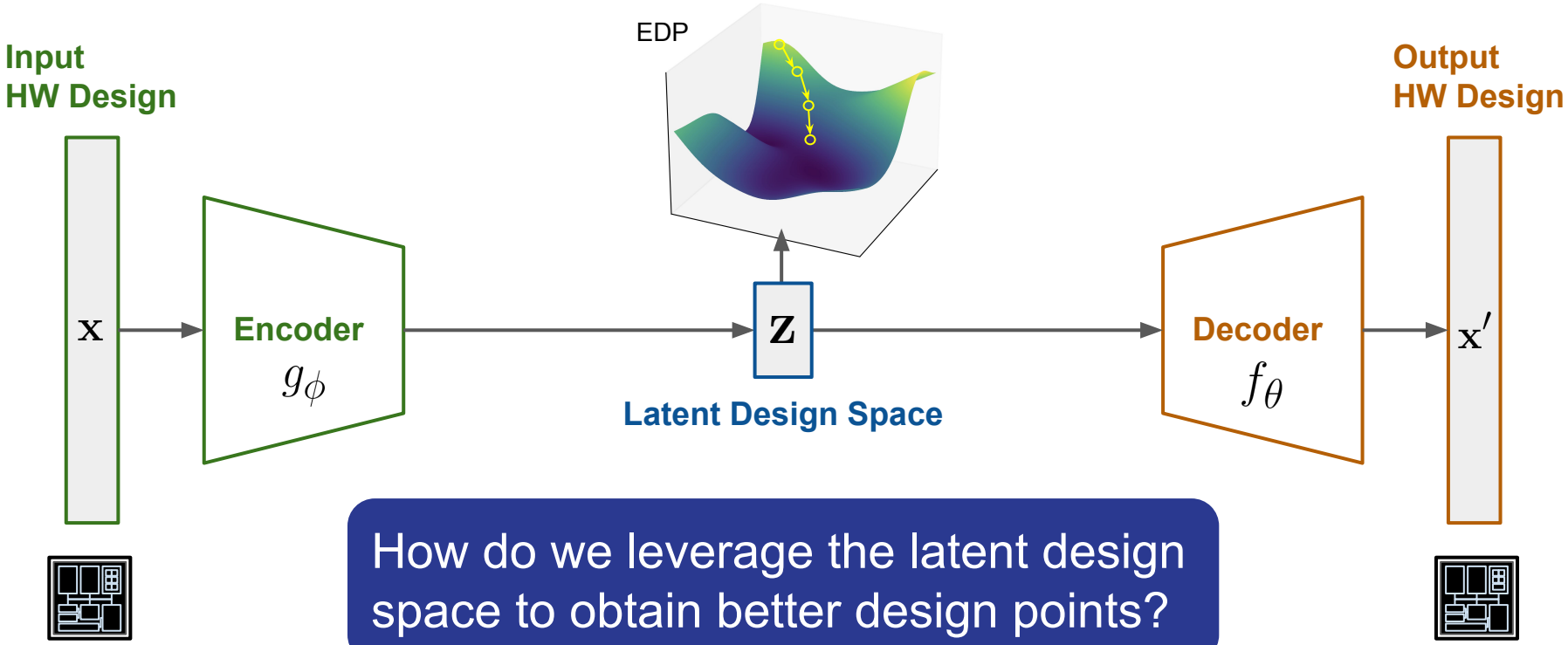
(c) Predicted energy usage



(d) Real energy usage of decoded accelerator

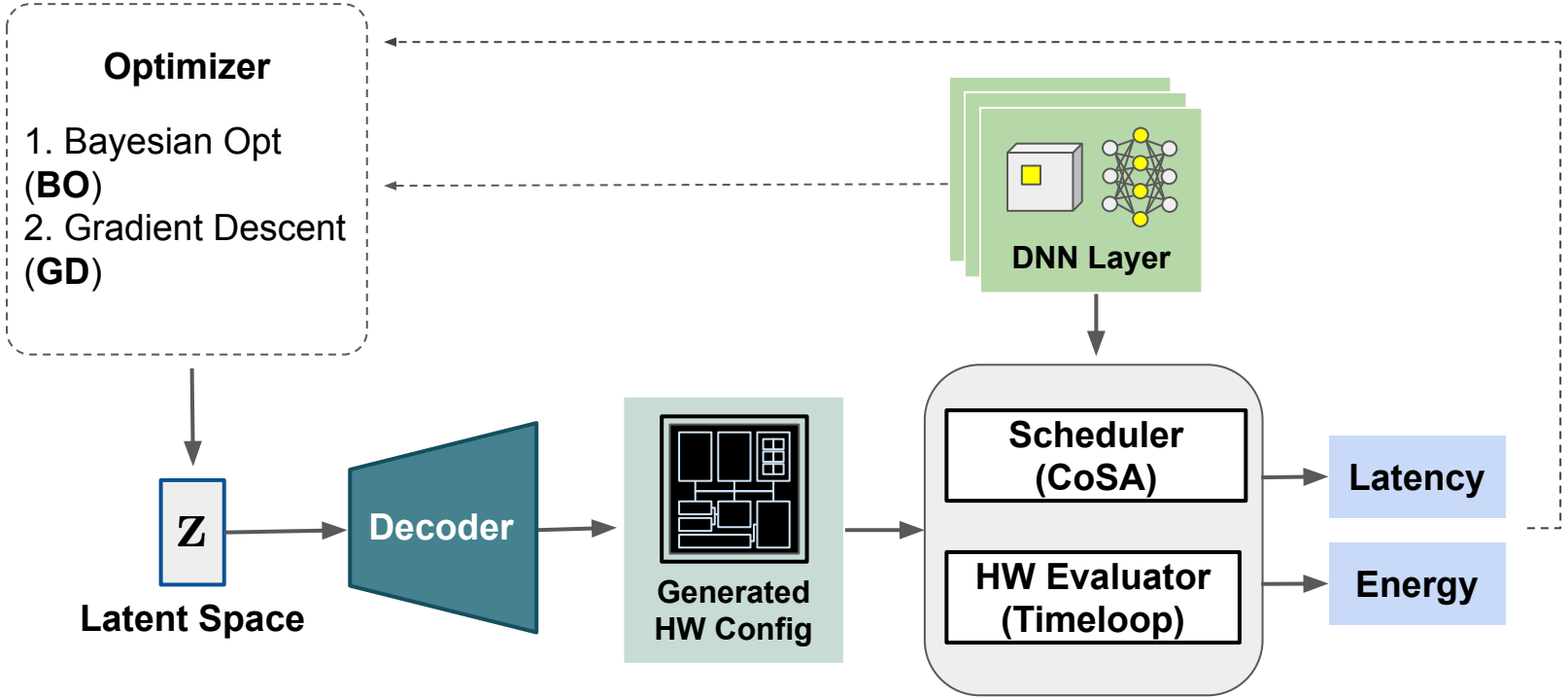
Performance values for ResNet-50

Our Framework - VAESA



How do we leverage the latent design space to obtain better design points?

VAESA Inference

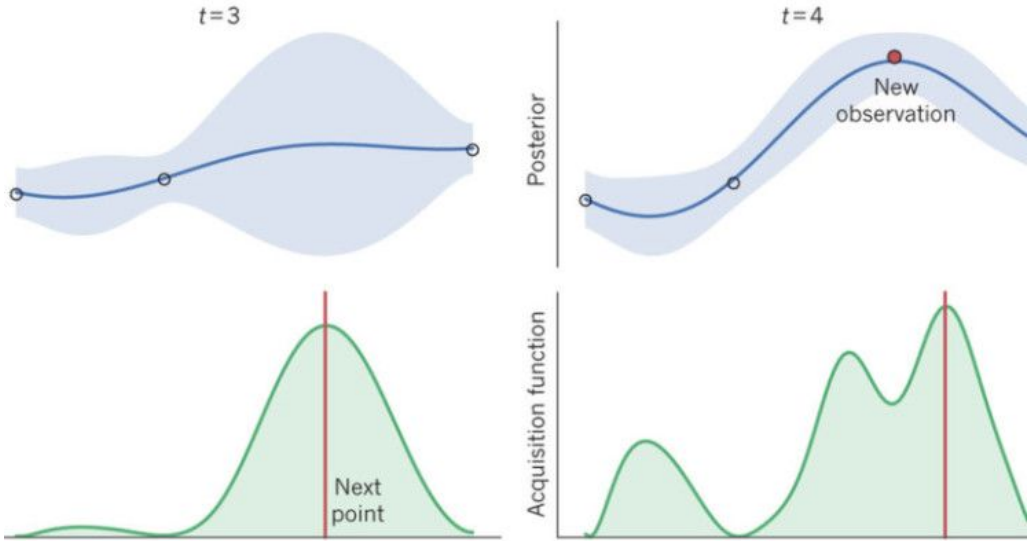


Huang et al. “CoSA: Scheduling by Constrained Optimization for Spatial Accelerators.” ISCA 2021.
Parashar et al. “Timeloop: A Systematic Approach to DNN Accelerator Evaluation.” ISPASS 2019.

VAESA Inference

Bayesian Optimization (BO)

- BO iteratively updates **a statistical model** to approximate the unknown objective function and uses **an acquisition function** to decide which input to sample next.

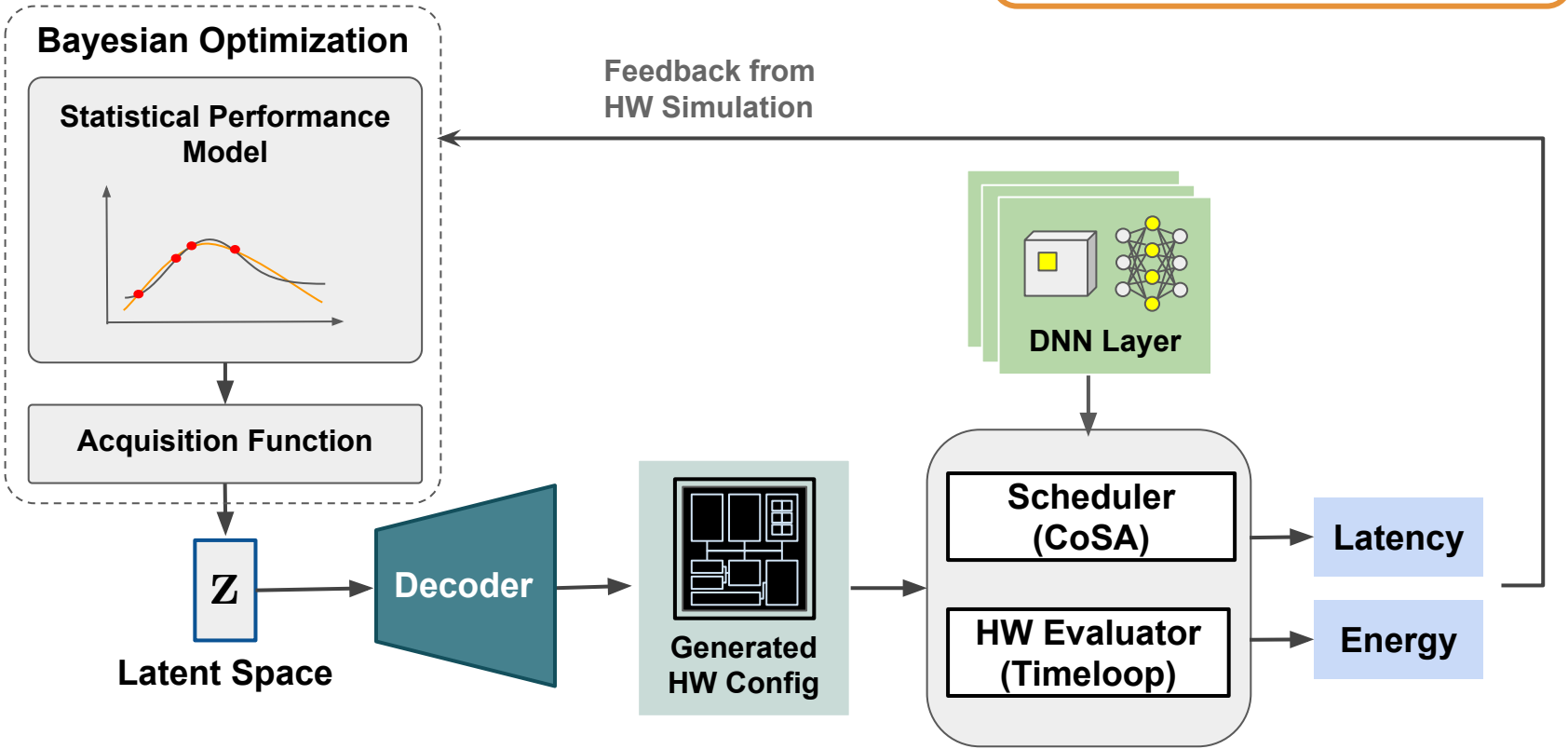


Ghahramani, Zoubin. "Probabilistic machine learning and artificial intelligence." Nature 521.7553 (2015): 452-459.

VAESA Inference

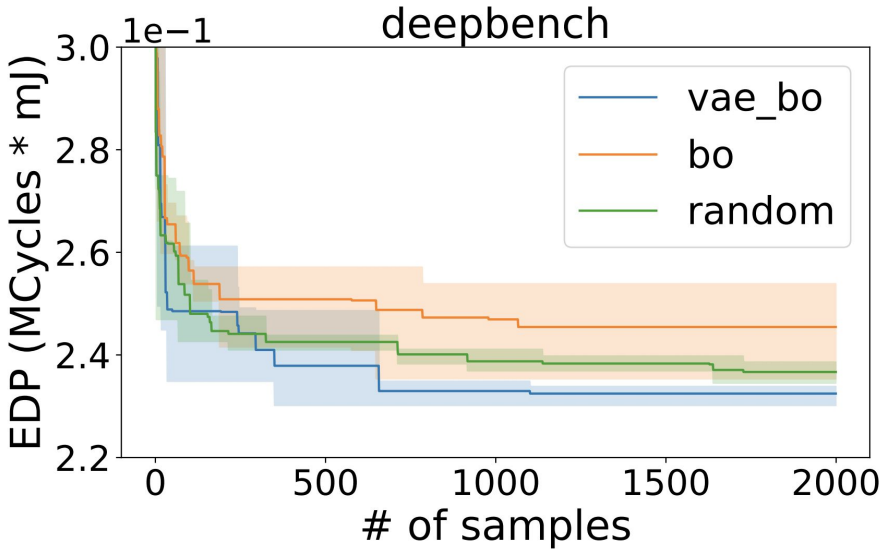
VAESA+BO

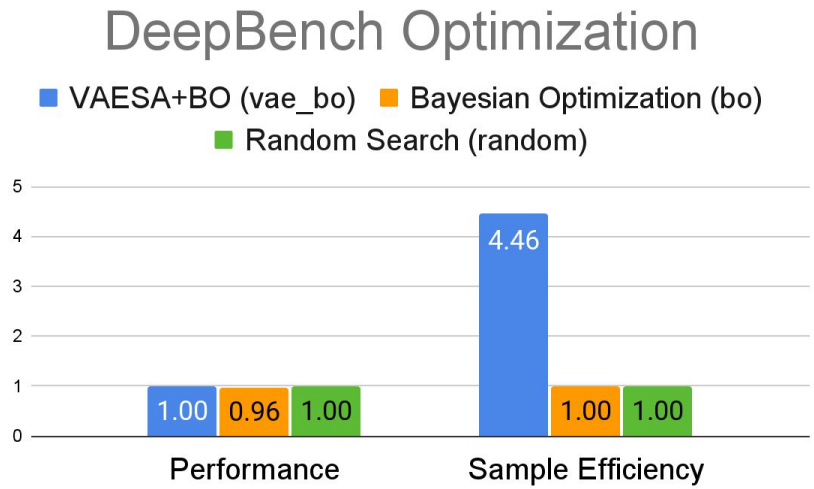
Black-box optimization on the latent space



Results

VAESA+BO Comparison



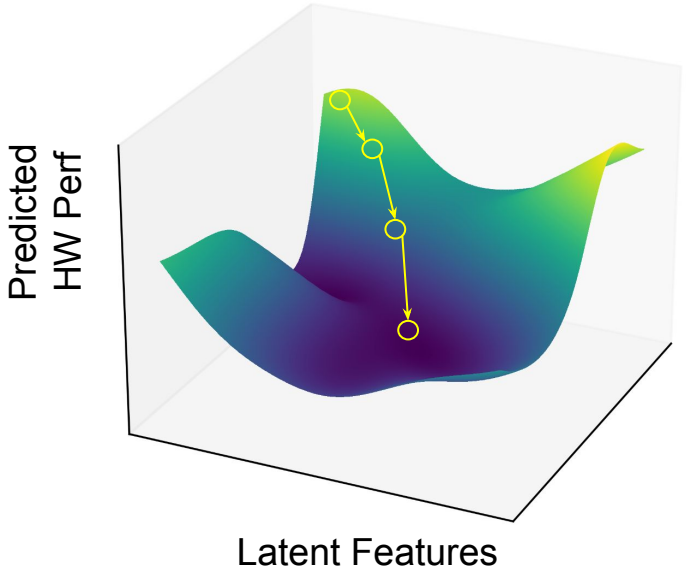


VAESA+BO improves the sample efficiency of BO and finds optimal accelerator designs.

VAESA Inference

Gradient Descent (GD)

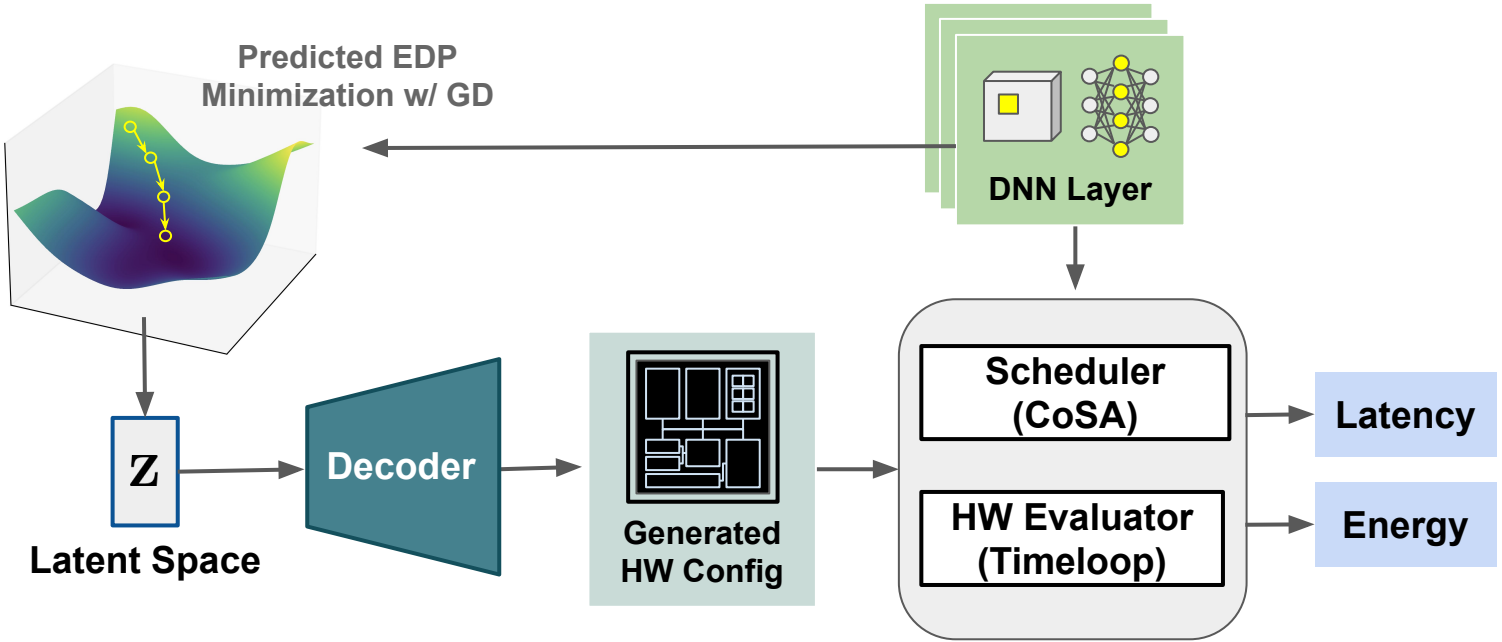
- GD is an iterative method for optimizing an objective function with suitable smoothness properties by take repeated steps **in the opposite direction of the gradient** of the function at the current point.



VAESA Inference

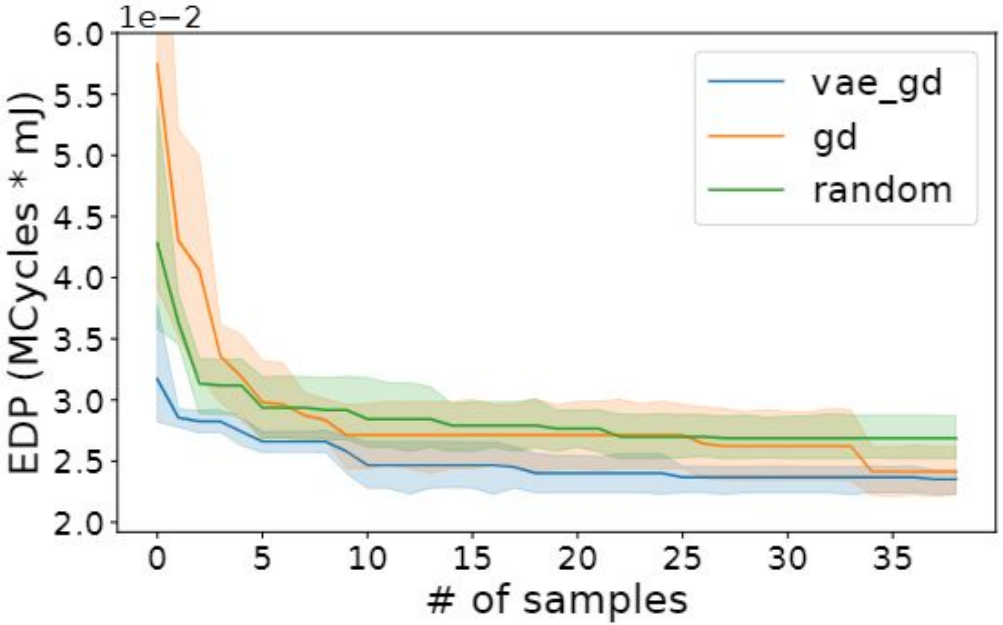
VAESA+GD

Predictor-based search on the latent space



Results

VAESA+GD Comparison



Average EDP improvement of GD compared to random search over 12 test layers. Experiments repeated for 5 random seeds.

GD on the latent space achieves better design points faster than GD on the original space.

Conclusion

In VAESA,

- We introduce an DSE framework where the search is performed on a **continuous** and **reconstructible** latent space
- We train a rigorous VAE model and use the trained models to enhance two state-of-the-art algorithms: *the black-box BO* and *the predictor-based GD algorithm*

Email: jennyhuang@nvidia.com, charleshong@berkeley.edu

Github: https://github.com/hqjenny/vaesa.git