Latency-Aware Neural Architecture Search with Multi-Objective Bayesian Optimization

David Eriksson
Pierce I-Jen Chuang
Samuel Daulton
Peng Xia
Akshat Shrivastava
Arun Babu
Shicong Zhao
Ahmed Aly
Ganesh Venkatesh
Maximilian Balandat

Ax
BoTorch
FACEBOOK

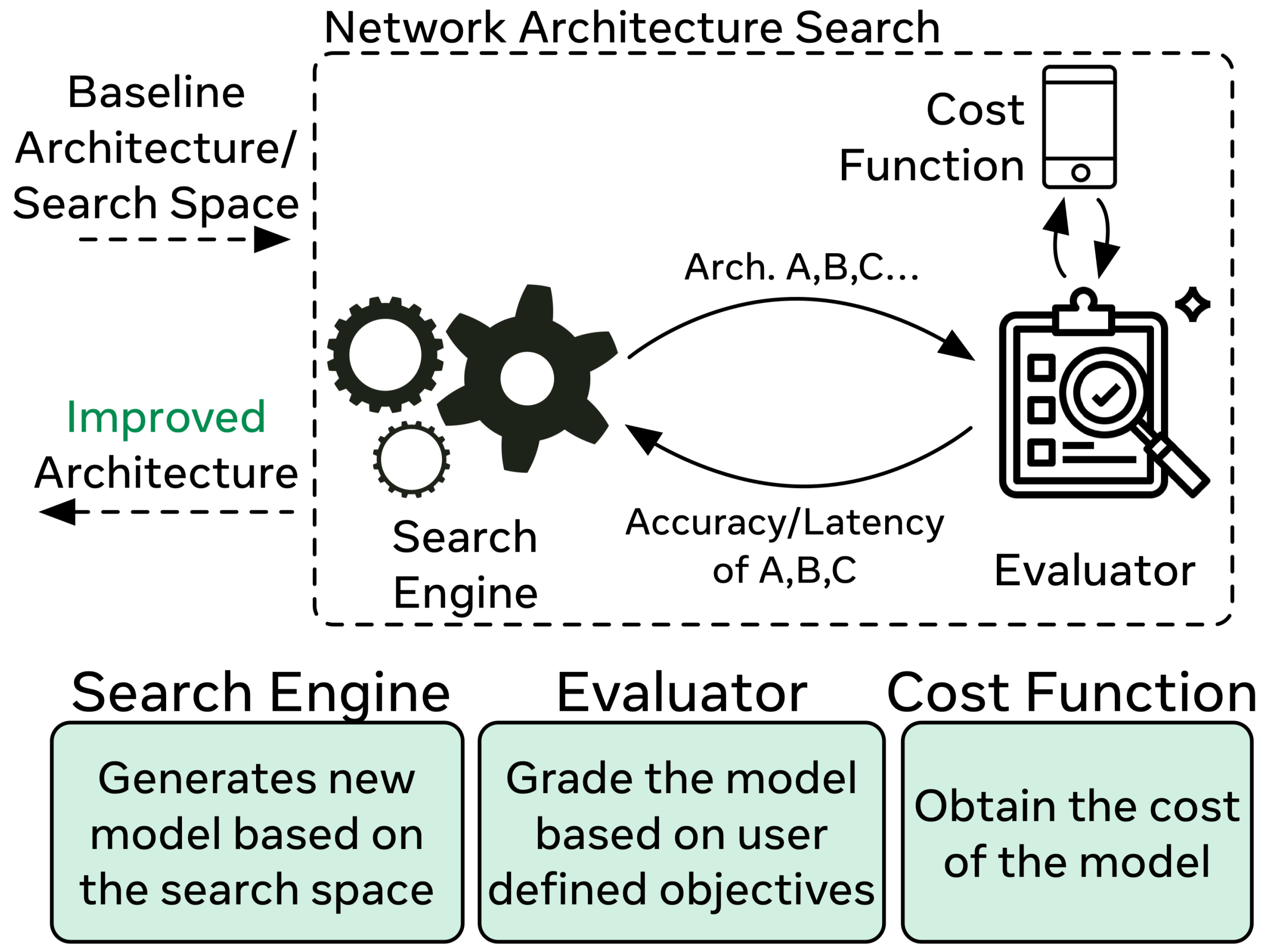
Latency-Aware Neural Architecture Search (NAS)

Problem: Deploying models on end user devices such as mobile phones requires low-latency and accurate predictive models

Goal: Provide an automated framework for identifying neural architectures that are optimal with respect to accuracy and on-device prediction latency

NAS Methodology

- Pre-defined search space (i.e., # of layers, kernel shape)
- Search engine with a user-specified search strategy (i.e., multi-objective Bayesian Optimization)
- Selected architecture is trained/evaluated and deployed to obtain actual inference latency
  - Inference latency is benchmarked on a tier-1 Android device with arm64 kernel through PyTorch lite interpreter
  - Inference latency is defined as the time when the input data becomes available to the time the model generates the final outputs
- Accuracy/latency results are fed back to the search engine
- Depending on the objective, the search engine then selects the next candidate for evaluation
  - Process is repeated until either the maximum number of iteration is reached or the final objective converges


Network Architecture Search
Baseline Architecture/ Search Space
Cost Function
Arch. A,B,C...
Improved Architecture
Accuracy/Latency of A,B,C
Search Engine
Evaluator
Search Engine
Generates new model based on the search space
Evaluator
Grade the model based on user defined objectives
Cost Function
Obtain the cost of the model

Overview of the NAS framework.

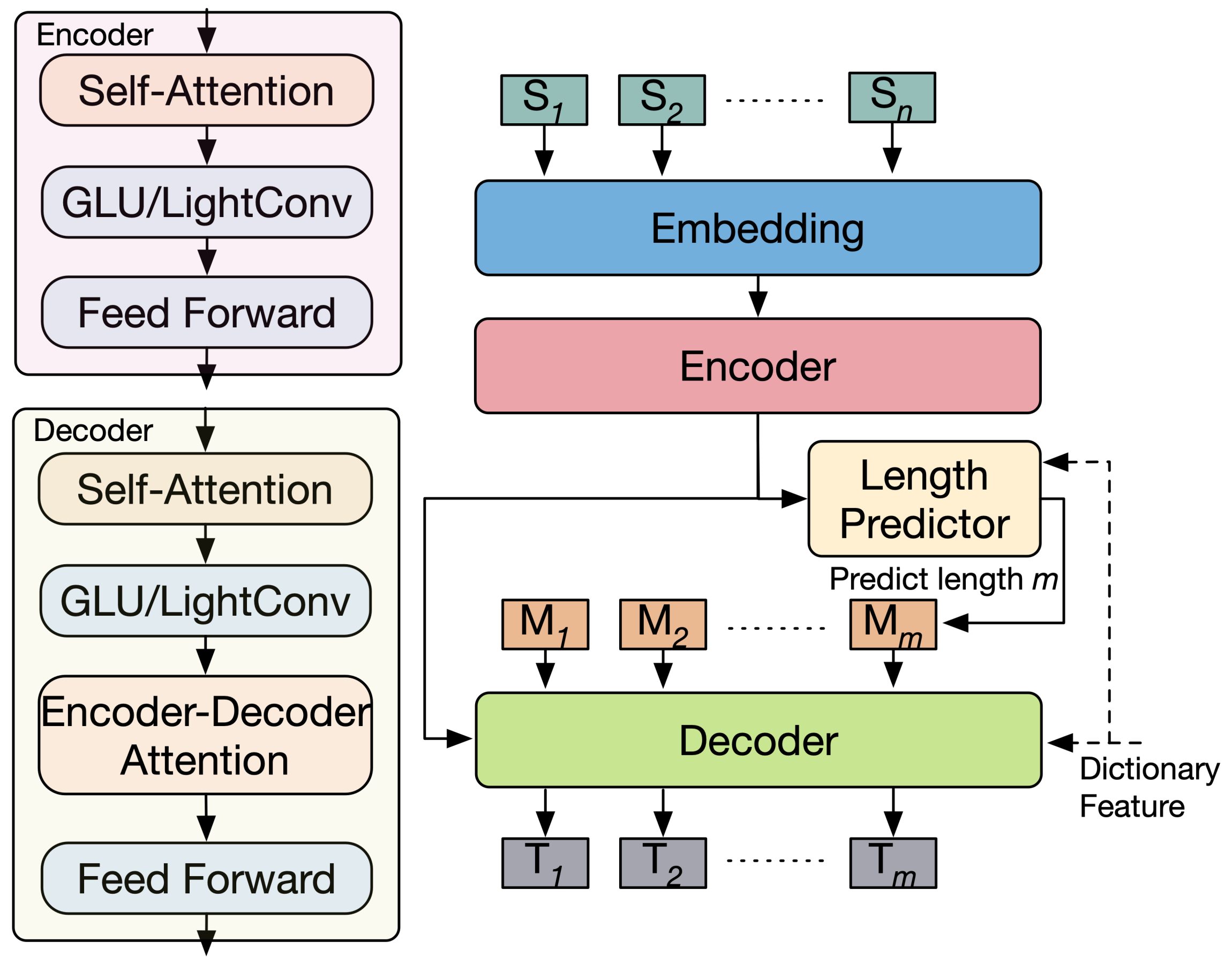
Use Case: On-Device Natural Language Understanding (NLU)

- NLU is commonly used by conversational agents in most mobile devices and smart speakers
- Primarily objective:
  - Understand the user's semantic expression
  - Convert the expression to a structured decoupled span form representation that can be understood by downstream task

Utterance	what is the weather in San Francisco
Index	1 2 3 4 5 6 7
Canonical Form	`[IN: GET_WEATHER [SL: LOCATION San Francisco ] ]`
Span Form	`[IN: GET_WEATHER [SL: LOCATION 6 7 ] ]`

Canonical vs span forms of the decoupled frame representation. Given the utterance "what is the weather in San Francisco", our span form produces endpoints instead of text, reformulating the task from text generation to index prediction

Model Architecture



Non-Autoregressive Model Architecture of the NLU Semantic Parsing

- Non-Autoregressive Transformer-like architecture
- Encoder generates the contextual information
- CNN-based length predictor predicts the length of the final output token
- Decoder to generate the info output tokens in one-shot

Search Space

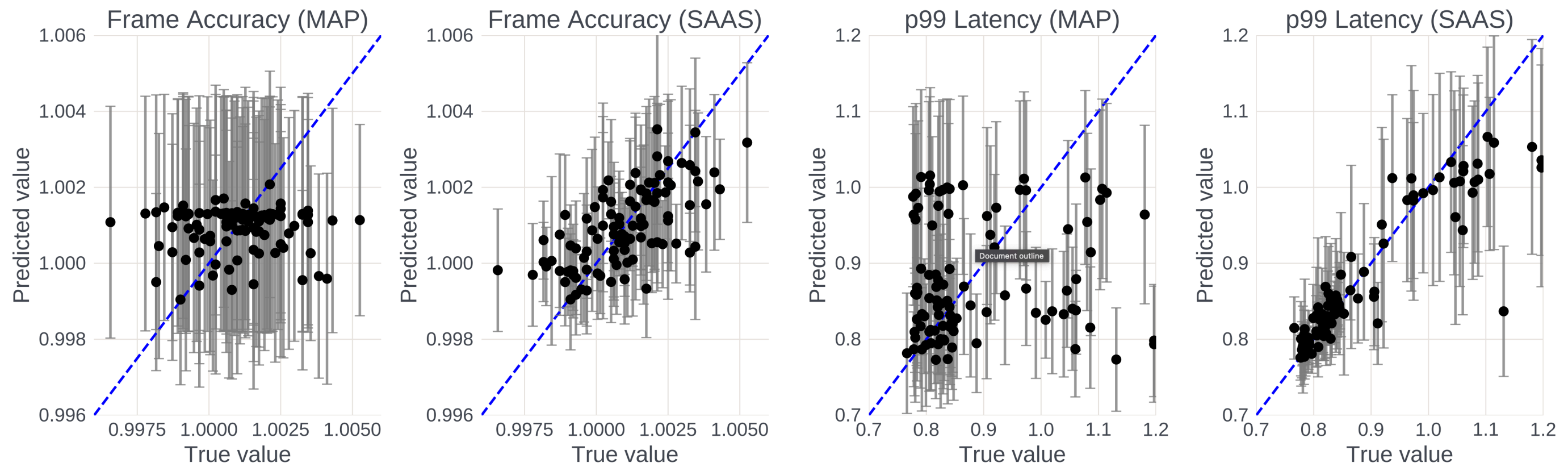
- Search space includes kernel size, number/width of layers, number of attention head, etc
- A total of 24 parameters to be searched

Parameter	Default	Search Space	Description
		Encoder	
kernel_list	[3, 3, 5, 9, 7]	[3, 3, 3, 3], ...	list of length 4-6, drawn from [3, 5, 7, 9]
embed_dim	128	128, 136, ..., 192	input dimension
self_attention	2	1, 2, 4	number of self-attention head
ffn_dim	40	32, 40, ..., 192	feed-forward network (FFN) width
normalized	True	True, False	apply normalization before the FFN
		Decoder	
kernel_list	[13, 9]	[7, 7], [7, 9], ...	list of length 1-2, drawn from [7, 9, 11, 13, 15]
self_attention	1	1, 2, 4	number of self-attention head
attention_heads	2	1, 2, 4	number of cross-attention head
ffn_dim	144	128, 144, ..., 512	FFN width
		Length Predictor	
kernel_list	[3, 7]	[3], [3, 5], ...	list of length 1-2, drawn from [3, 5, 7]
dim	192	32, 40, ..., 192	convolution width
num_head	4	1, 2, 4	number of attention head
		Embedding	
char_embed_dim	8	8, 12, ..., 24	character embedding dimension
proj_dim	12	8, 12, ..., 24	last layer projection dimension

List of the tunable parameters (i.e., search space) of the NLU model

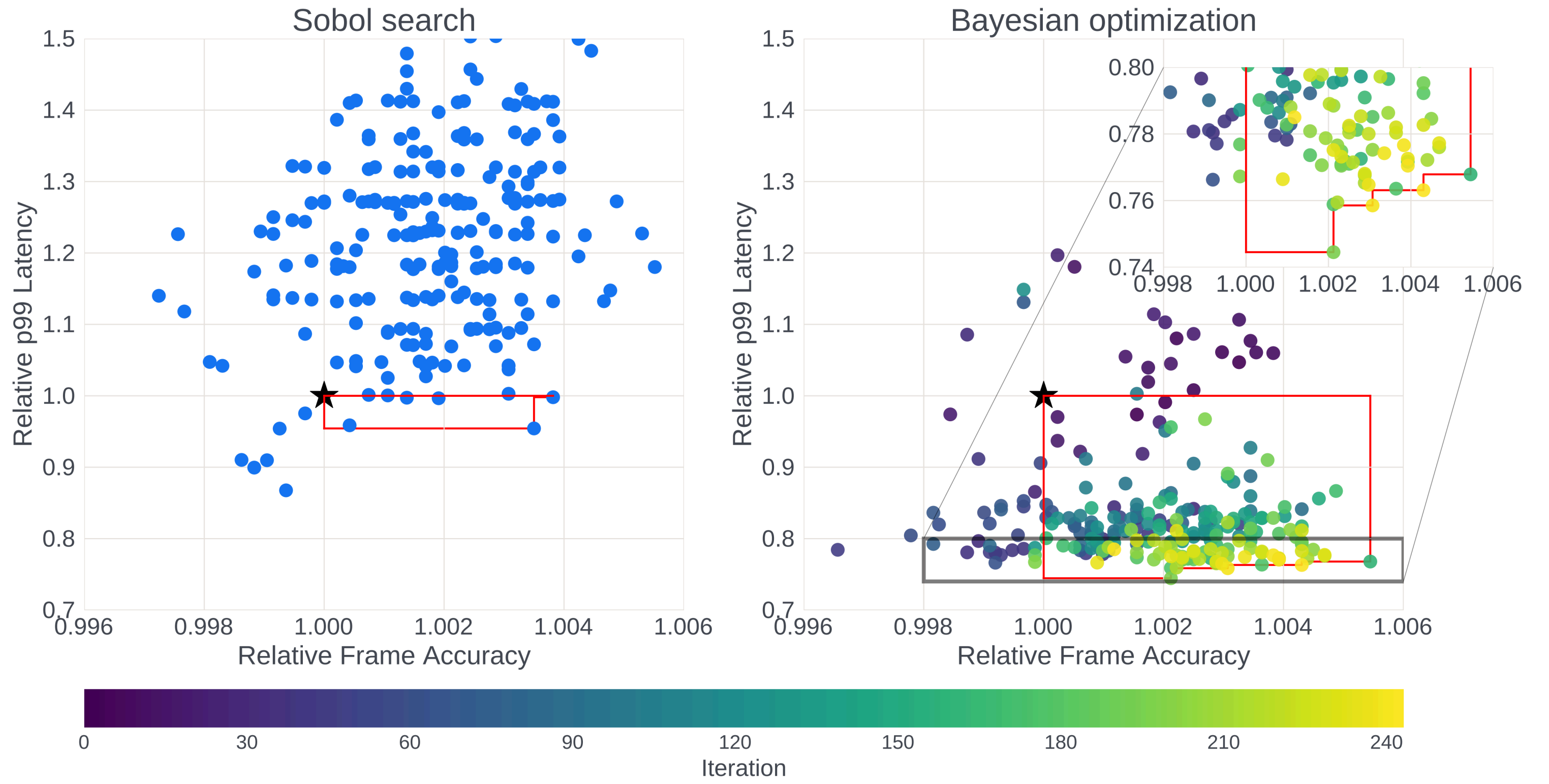
Multi-Objective Bayesian Optimization

- We use Gaussian process surrogate models with sparse axis-aligned subspace priors [2] to model the two objectives over the high-dimensional search space



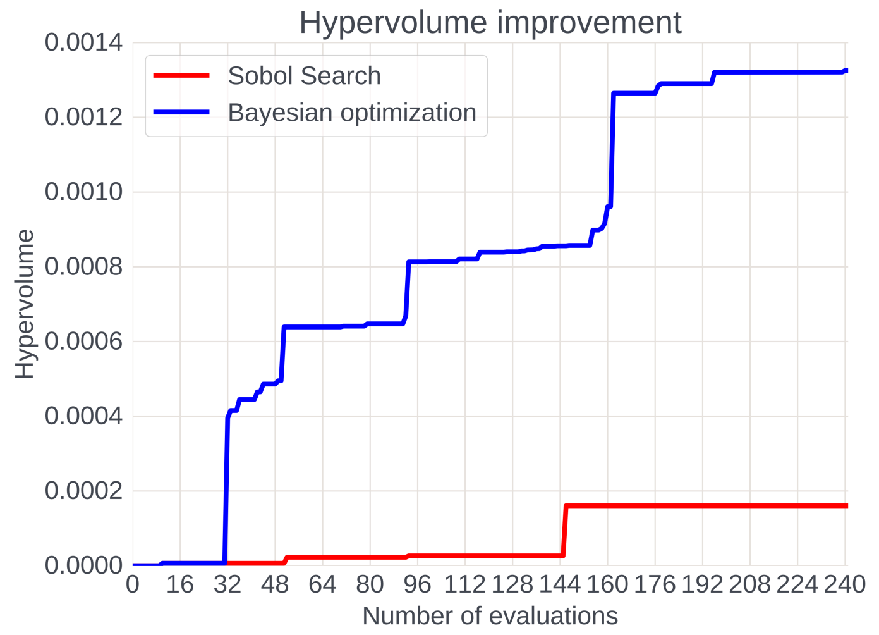
Leave-one-out cross-validation comparison for SAAS [2] and MAP using 100 training configurations. Using the SAAS prior provides good fits for both objectives while MAP estimation is unable to provide accurate model fits

- To select new candidate architectures, we use an integrated variant of the qNEHVI acquisition function [3], where we integrate qNEHVI over the posterior distribution of the GP hyperparameters.
- We find that multi-objective bayesian optimization vastly outperforms (quasi-) random search.



(Left) Sobol search is only able to find two points that improve upon the reference point. (Right) BO is able to successfully explore the trade-ff between latency and accuracy

BO improves the hypervolmme quickly after the initial Sobol batch and makes continuous improvement until the evaluation budget is exhausted



References

[1] A. Shrivasava et al. "Span Pointer Networks for Non-Autoregressive Task-Oriented Semantic Parsing." arXiv preprint arXiv:2104.07275, 2021.

[2] Eriksson, David, and Martin Jankowiak. "High-Dimensional Bayesian Optimization with Sparse Axis-Aligned Subspaces." Conference on Uncertainty in Artificial Intelligence (UAI), 2021.

[3] Daulton, Samuel, Maximilian Balandat, and Eytan Bakshy. "Parallel Bayesian Optimization of Multiple Noisy Objectives with Expected Hypervolume Improvement." arXiv preprint arXiv:2105.08195, 2021.