Priors in probabilistic numerics

Zi Wang

ProbNum @ Dagstuhl

Google Research

Prior, engineering prior and data prior

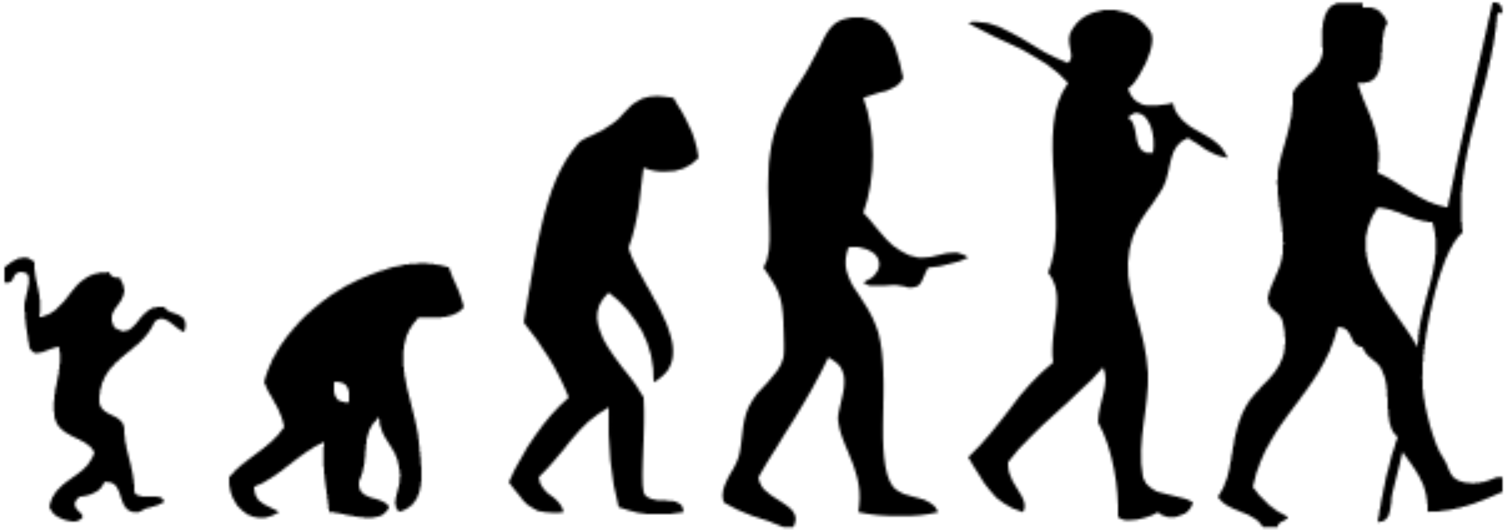
Where does the prior come from?

Priors in nature

video: https://youtu.be/Jyee1fJMLAk

image: commons.wikimedia.org

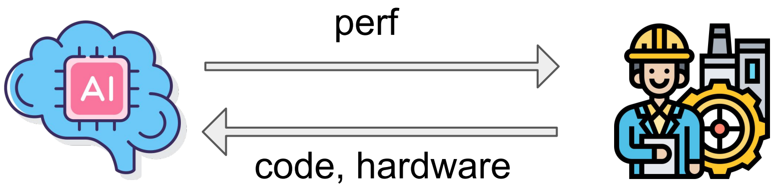
Priors in nature through evolution



Google Research

image: Flaticon.com

Priors in software: engineering prior

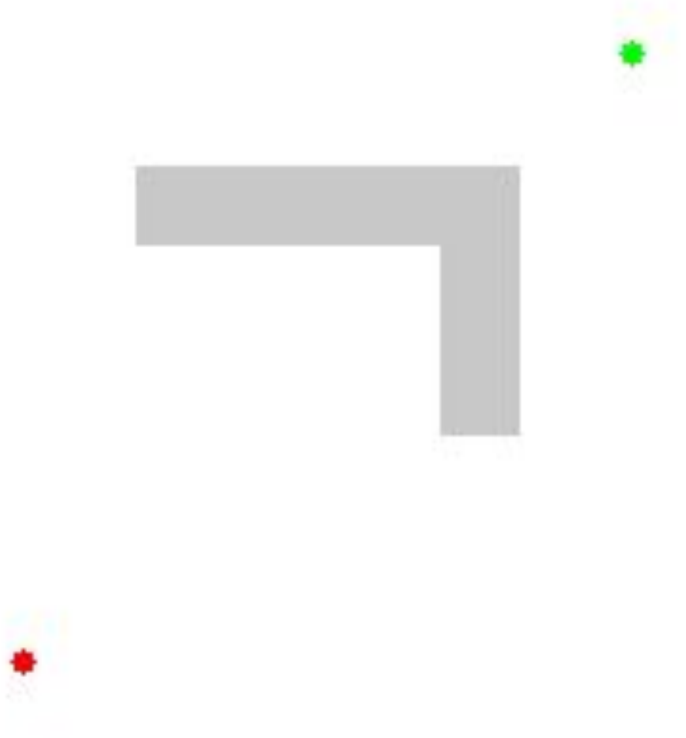


- engineering prior ≈ (a sample of) the posterior of engineers
- posterior of engineers builds on human knowledge as a whole (education, books, journals, blogs + engineer experience with models)

Google Research

https://en.wikipedia.org/wiki/A*_search_algorithm

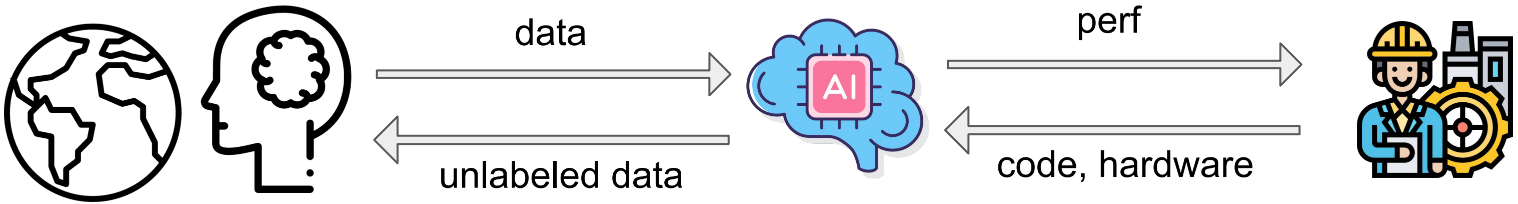
Example of engineering prior: A*



- Very generalizable;
- Wide applications in robot planning and path planning in games;
- Algorithm entirely built-in by expert knowledge and abstraction of how humans solve path planning problem.

image: Flaticon.com

Software: engineering prior + data

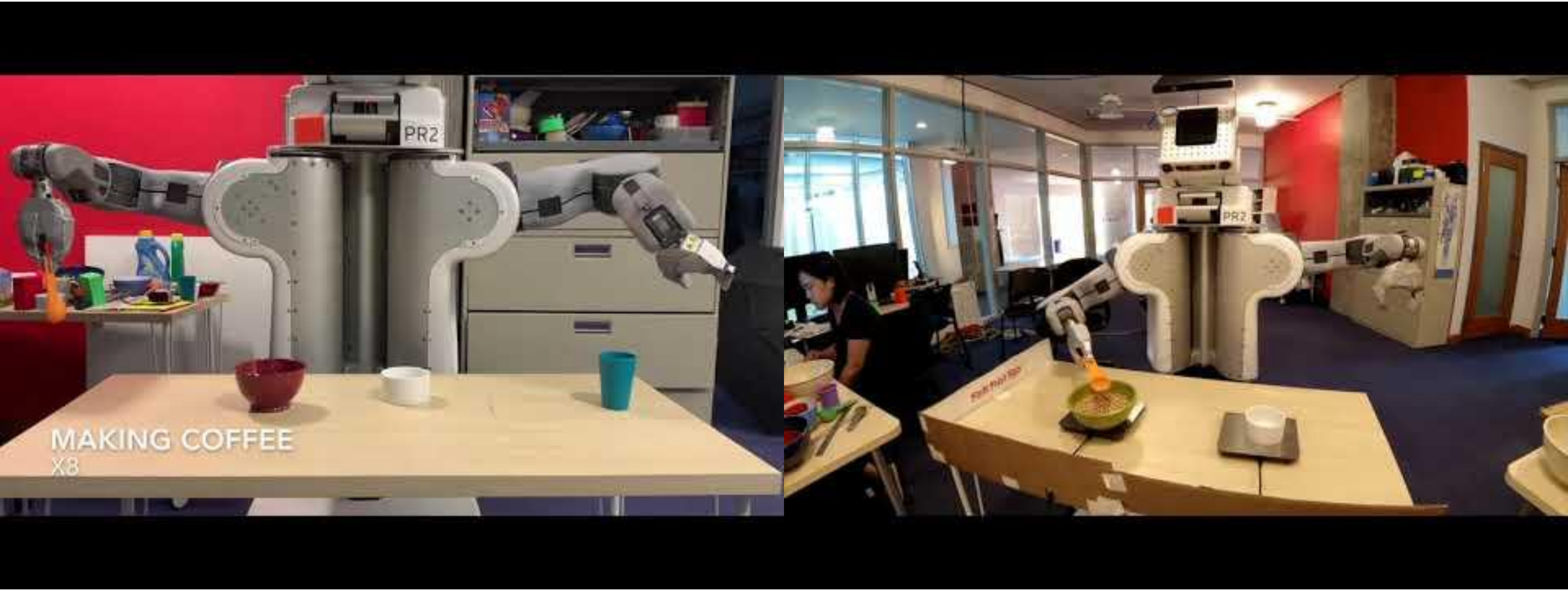


- engineering prior ≈ (a sample of) the posterior of engineers
- data = a set of data points returned by the world or human annotators

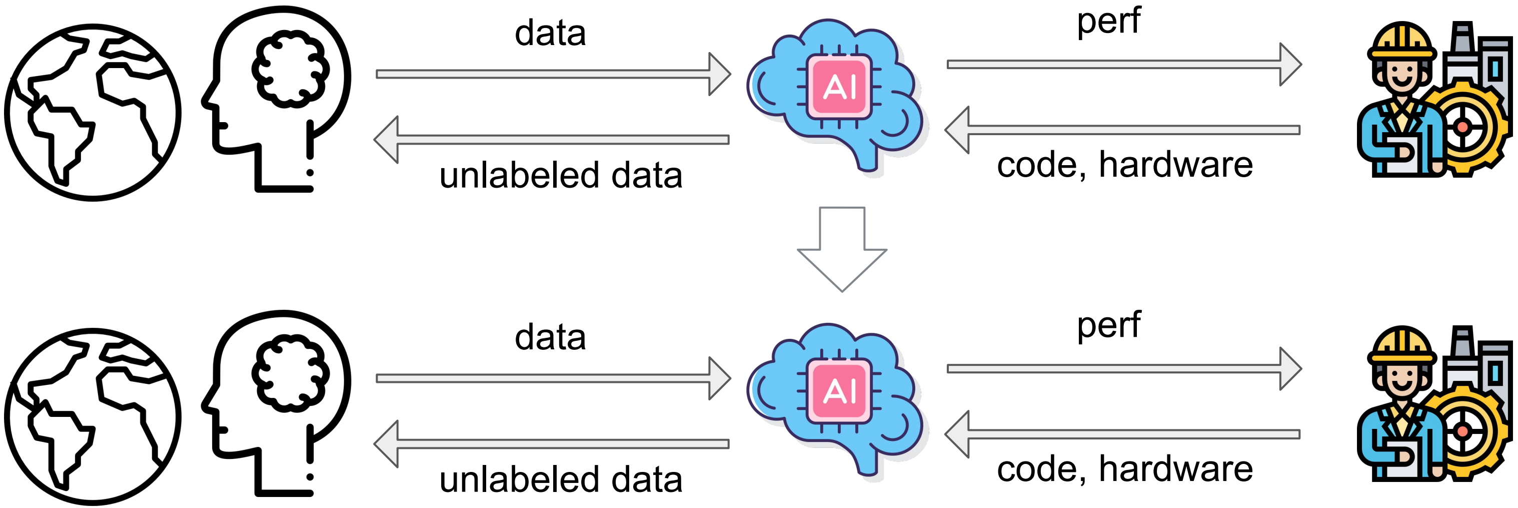
Google Research

[Wang, PhD Thesis 2020]

Example: robot learning with strong engineering priors



Data prior: priors in the evolution of software



- data prior ≈ pretraining

Obtaining a data prior through meta learning

Task 1	(x_11, y_11)	(x_12, y_12)	……	(x_1M, y_1M)
Task 2	(x_21, y_21)	(x_22, y_22)	……	(x_2M, y_2M)
……	……	……	……	……
Task N	(x_N1, y_N1)	(x_N2, y_N2)	……	(x_NM, y_NM)
New Task	?	?	……	?

- The software to solve a new task corresponds to a new generation of models.
- A data prior can be obtained through all previous generations.

[Wang, PhD Thesis 2020]
[Kim&Wang&Kaelbling&Lozano-Pérez, IJRR 2019]
[Wang*&Kim*&Kaelbling, NeurIPS 2018]

Data prior example on BayesOpt

"HyperBO does not assume the knowledge of any GP parameters; instead, we learn the GP mean function, kernel function, and possible observation noise from data in the form of point sets, i.e. *i.i.d.* sets of correlated points."
— Automatic prior selection for meta Bayesian optimization with a case study on tuning deep neural network optimizers.

Joint work with G. E. Dahl, K. Swersky, C. Lee, Z. Mariet, Z. Nado, J. Gilmer, J. Snoek, Z. Ghahramani. https://arxiv.org/abs/2109.08215

Google Research

Assumption: evaluation functions on hyperparameters for all tasks are *i.i.d.* function samples from a GP

Task f_1	(x_11, y_11)	(x_12, y_12)	……	(x_1M, y_1M)
……	……	……	……	……
Task f_i	(x_i1, y_i1)	(x_i2, y_i2)	……	(x_iM, y_iM)
……	……	……	……	……
Task f_N	(x_N1, y_N1)	(x_N2, y_N2)	……	(x_NM, y_NM)
New Task	?	?	……	?

D_{f_i}

$D_N = \{D_{f_i}\}_{i=1}^N$

$$f_i \sim \mathcal{GP}(\mu, k)$$

$$f \sim \mathcal{GP}(\mu, k)$$

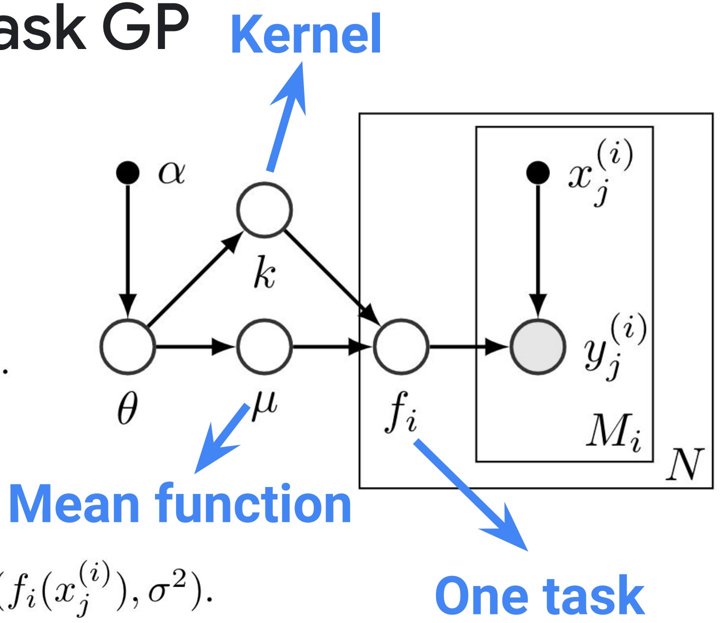
$$\text{solve } \max_{x \in \mathfrak{X}} f(x)$$

Google Research

Hierarchical Gaussian process: a different viewpoint of multi-task GP

All tasks are IID samples from a GP

- Draw parameter θ from $p(\theta; \alpha)$.
- Draw mean function μ and kernel function k from $p(\mu, k \mid \theta)$.
- For each outer iteration i from 1 to N,
  - Draw a function f_i from $\mathcal{GP}(\mu, k)$.
  - For each inner loop iteration from 1 to M_i,
    - Given input $x_j^{(i)}$, we draw the observation $y_j^{(i)} \sim \mathcal{N}(f_i(x_j^{(i)}), \sigma^2)$.



Instead of learning correlations among tasks, we learn the GP that generated all tasks

HyperBO: a practical meta Bayesian optimization method

Algorithm 1 HyperBO with acquisition function $\alpha(\cdot)$.

function HYPERBO (f, D_N)

  $\mathcal{GP}(\hat{\mu}, \hat{k}) \leftarrow \text{TRAINGP}(D_N)$

  $D_f \leftarrow \emptyset$

  for $t = 1, \cdots, T$ **do**

    $x_t \leftarrow \underset{x \in \mathfrak{X}}{\arg\max}\, \alpha\left(x; \mathcal{GP}(\hat{\mu}, \hat{k} \mid D_f)\right)$

    $y_t \leftarrow \text{OBSERVE}(f(x_t))$

    $D_f \leftarrow D_f \cup \{(x_t, y_t)\}$

  end for

  return D_f

end function

Optimize data likelihood

Optimize “distance” between sample mean/covariance estimates and predictions

Optimize data likelihood

$$\log p(D_N \mid \mu, k, \sigma^2) = \sum_{i=1}^{N} \log p(D_{f_i} \mid \mu, k, \sigma^2)$$
$$= \sum_{i=1}^{N} \left(-\frac{1}{2} \bar{\boldsymbol{y}}_{(i)}^{\top} K^{-1} \bar{\boldsymbol{y}}_{(i)} - \frac{1}{2} \log |K| - \frac{M_i}{2} \log 2\pi \right)$$

$$\bar{\boldsymbol{y}}_{(i)} = \boldsymbol{y}^{(i)} - \mu\left(\boldsymbol{x}^{(i)}\right), K = k\left(\boldsymbol{x}^{(i)}\right) + \sigma^2 \boldsymbol{I}, \boldsymbol{x}^{(i)} = [x_j^{(i)}]_{j=1}^{M_i} \text{ and } \boldsymbol{y}^{(i)} = [y_j^{(i)}]_{j=1}^{M_i}$$

Also possible to search over different mean/kernel architectures

Google Research

Optimize distance between sample mean/covariance estimates and predictions (for same inputs across tasks)

Task 1	(x_1, y_11)	(x_2, y_12)	……	(x_M, y_1M)
Task 2	(x_1, y_21)	(x_2, y_22)	……	(x_M, y_2M)
……	……	……	……	……
Task N	(x_1, y_N1)	(x_2, y_N2)	……	(x_M, y_NM)
New Task	?	?	……	?

$$\boldsymbol{y} = [\boldsymbol{y}_j]_{j=1}^{M} \in \mathbb{R}^{M \times N}$$

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\boldsymbol{y}1_N \in \mathbb{R}^M$$

$$\hat{K} = \frac{1}{N}(\boldsymbol{y} - \hat{\boldsymbol{\mu}}1_N^\top)(\boldsymbol{y} - \hat{\boldsymbol{\mu}}1_N^\top)^\top \in \mathbb{R}^{M \times M}$$

$$\mathcal{D}_{\mathrm{KL}}(\hat{\boldsymbol{\mu}}, \hat{K}, \mu(\boldsymbol{x}), k(\boldsymbol{x}) + \boldsymbol{I}\sigma^2) = \frac{1}{2}\left(\mathrm{tr}(K^{-1}\hat{K}) + (\mu(\boldsymbol{x}) - \hat{\boldsymbol{\mu}})^\top K^{-1}(\mu(\boldsymbol{x}) - \hat{\boldsymbol{\mu}}) + \ln\frac{|K|}{|\hat{K}|} - M\right)$$

Google Research

Preliminary Results

Google Research

Reduced complexity in #Tasks

$O(M^3 N)$

Theoretical results: near-zero regret with unknown GP priors

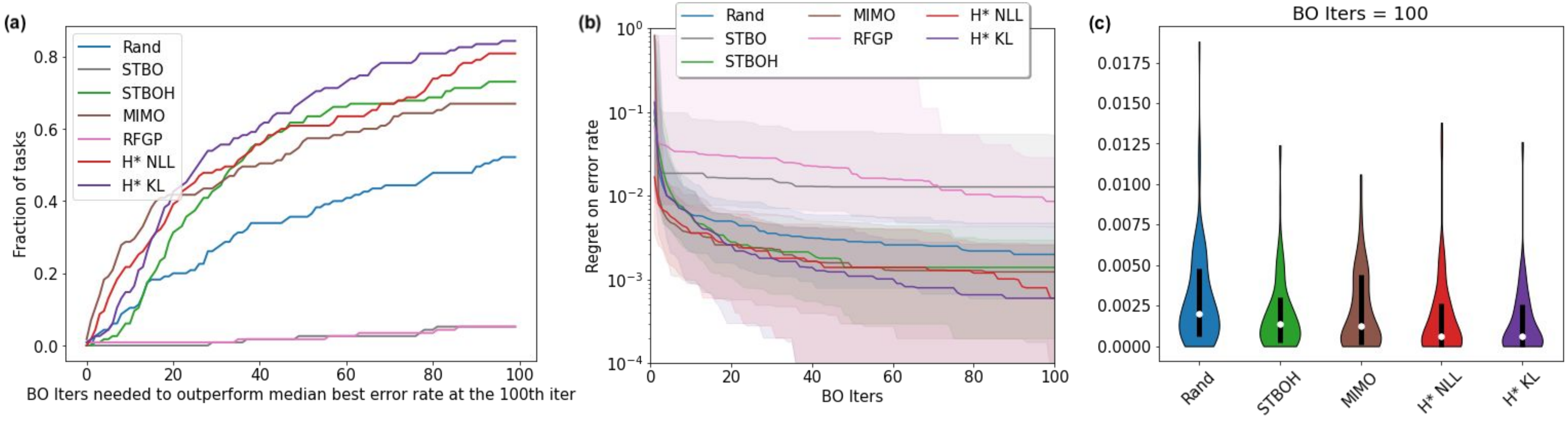
Proposition 1. *For any* $M, d, N \in \mathbb{Z}^+, \boldsymbol{x} \in \mathbb{R}^{M\times d}, \boldsymbol{\mu} \in \mathbb{R}^M, V \in \mathbb{R}^{N\times M}$ *and* $K = V^\top V$, *there exists a Gaussian process* $\mathcal{GP}(\hat{\mu}, \hat{k})$ *such that* $\mathcal{D}_{EUC}(\hat{\mu}(\boldsymbol{x}), \hat{k}(\boldsymbol{x}), \boldsymbol{\mu}, K) \equiv 0$.

Theorem 2. *Let* $N \geq 4\log\frac{6}{\delta} + T + 2$. *With probability at least* $1-\delta$, *simple regret in* T *iterations of HyperBO with special cases of either GP-UCB or PI satisfies*

$$R_T < O\left(\sqrt{\frac{1}{N-T}} + (\log\frac{1}{\delta})^{\frac{1}{2}}\right) O(\rho_T/T + \sigma), \quad (1)$$

where $\rho_T = \max\limits_{A\subset\mathfrak{X}, |A|=T} \frac{1}{2}\log|\boldsymbol{I} + \sigma^{-2}k(A)|$.

Hyper BO achieves better empirical results on offline optimizer hyperparameter tuning tasks



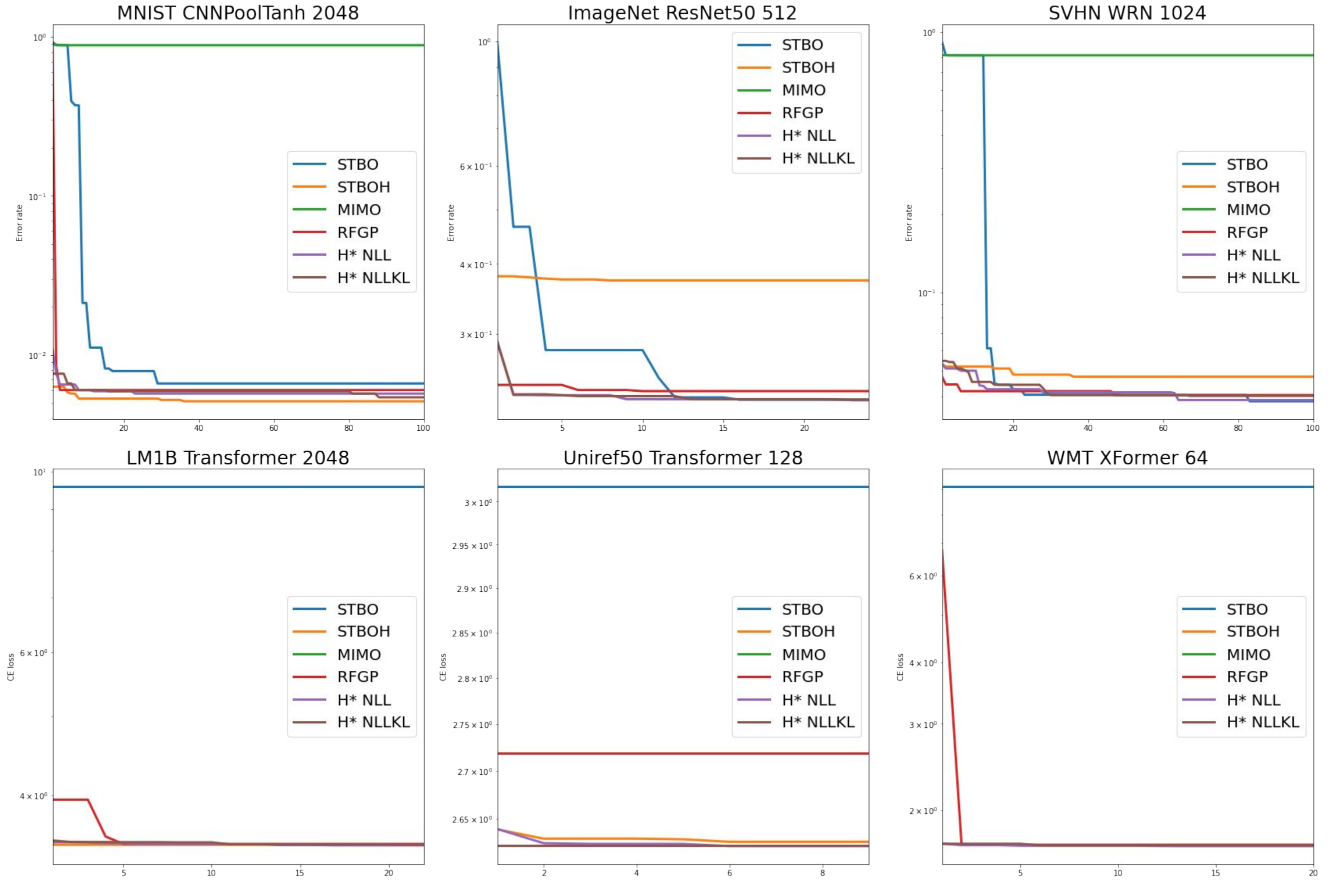
hyperparameter tuning dataset generated by github.com/google/init2winit

Best validation error rates for each task in the offline leave-one-out experiments (after 100 BO iterations)

HyperBO w/ different objectives

	Rand	STBOH	MIMO	MAF	H* NLL	H* KL
WMT XFormer 64	34.27 ± 0.16	34.15 ± 0.15	34.40 ± 0.13	34.09 ± 0.09	**33.91 ± 0.01**	33.97 ± 0.02
Uniref50 Transformer 128	79.06 ± 0.04	78.92 ± 0.12	79.17 ± 0.13	79.34 ± 0.27	78.71 ± 0.06	**78.64 ± 0.00**
LM1B Transformer 2048	61.96 ± 0.03	61.95 ± 0.04	61.96 ± 0.05	62.02 ± 0.10	**61.81 ± 0.01**	**61.81 ± 0.01**
SVHN WRN 1024	3.99 ± 0.04	4.05 ± 0.10	**3.83 ± 0.04**	4.10 ± 0.09	4.10 ± 0.02	4.08 ± 0.01
SVHN WRN 256	3.71 ± 0.01	3.72 ± 0.02	**3.65 ± 0.01**	3.69 ± 0.03	3.78 ± 0.01	3.72 ± 0.03
ImageNet ResNet50 256	23.03 ± 0.07	22.66 ± 0.07	22.73 ± 0.07	26.44 ± 1.98	**22.53 ± 0.00**	22.58 ± 0.04
ImageNet ResNet50 512	23.02 ± 0.11	22.74 ± 0.05	23.01 ± 0.05	25.46 ± 1.41	**22.65 ± 0.02**	22.79 ± 0.03
MNIST CNNPoolTanh 2048	0.55 ± 0.01	**0.53 ± 0.01**	**0.53 ± 0.01**	**0.52 ± 0.01**	0.59 ± 0.02	0.54 ± 0.00
MNIST CNNPoolTanh 256	0.51 ± 0.01	0.48 ± 0.01	**0.47 ± 0.00**	**0.47 ± 0.01**	**0.46 ± 0.01**	**0.47 ± 0.01**
MNIST CNNPoolReLU 2048	0.69 ± 0.01	0.73 ± 0.02	0.67 ± 0.02	0.68 ± 0.01	**0.64 ± 0.00**	0.70 ± 0.03
MNIST CNNPoolReLU 256	0.51 ± 0.01	0.55 ± 0.03	0.50 ± 0.01	0.51 ± 0.01	**0.49 ± 0.00**	**0.49 ± 0.00**
MNIST CNNReLU 2048	1.14 ± 0.03	1.20 ± 0.09	1.10 ± 0.01	1.17 ± 0.02	**1.06 ± 0.00**	1.11 ± 0.02
MNIST CNNReLU 256	1.09 ± 0.02	1.06 ± 0.01	1.08 ± 0.02	1.07 ± 0.02	**1.03 ± 0.00**	1.04 ± 0.01
Fashion CNNPoolTanh 2048	7.14 ± 0.06	7.10 ± 0.05	**7.01 ± 0.04**	7.12 ± 0.04	**7.00 ± 0.04**	**7.02 ± 0.07**
Fashion CNNPoolTanh 256	6.51 ± 0.03	6.67 ± 0.18	6.40 ± 0.05	6.47 ± 0.03	6.40 ± 0.04	**6.34 ± 0.04**
Fashion CNNPoolReLU 2048	**7.47 ± 0.02**	**7.48 ± 0.04**	7.54 ± 0.06	7.63 ± 0.04	**7.47 ± 0.03**	**7.47 ± 0.02**
Fashion CNNPoolReLU 256	6.78 ± 0.04	**6.74 ± 0.01**	7.03 ± 0.07	6.84 ± 0.05	**6.74 ± 0.03**	6.81 ± 0.05
Fashion CNNReLU 2048	7.70 ± 0.03	**7.47 ± 0.09**	7.60 ± 0.04	40.40 ± 17.80	**7.54 ± 0.01**	7.57 ± 0.02
Fashion CNNReLU 256	7.70 ± 0.04	7.46 ± 0.11	7.84 ± 0.06	24.13 ± 14.54	**7.29 ± 0.05**	**7.25 ± 0.05**
CIFAR100 WRN 2048	21.28 ± 0.27	**20.78 ± 0.19**	21.75 ± 0.15	50.70 ± 15.44	21.22 ± 0.23	**20.82 ± 0.19**
CIFAR100 WRN 256	19.17 ± 0.19	19.02 ± 0.03	19.12 ± 0.04	19.84 ± 0.13	**19.00 ± 0.00**	19.04 ± 0.05
CIFAR10 WRN 2048	3.73 ± 0.05	**3.43 ± 0.07**	**3.46 ± 0.05**	**3.40 ± 0.06**	3.55 ± 0.10	**3.43 ± 0.05**
CIFAR10 WRN 256	2.84 ± 0.04	2.88 ± 0.06	2.89 ± 0.06	3.04 ± 0.05	2.82 ± 0.03	**2.74 ± 0.01**

Online experiment results: HyperBO is most stable



Thank you!

zi-wang.com

Google Research