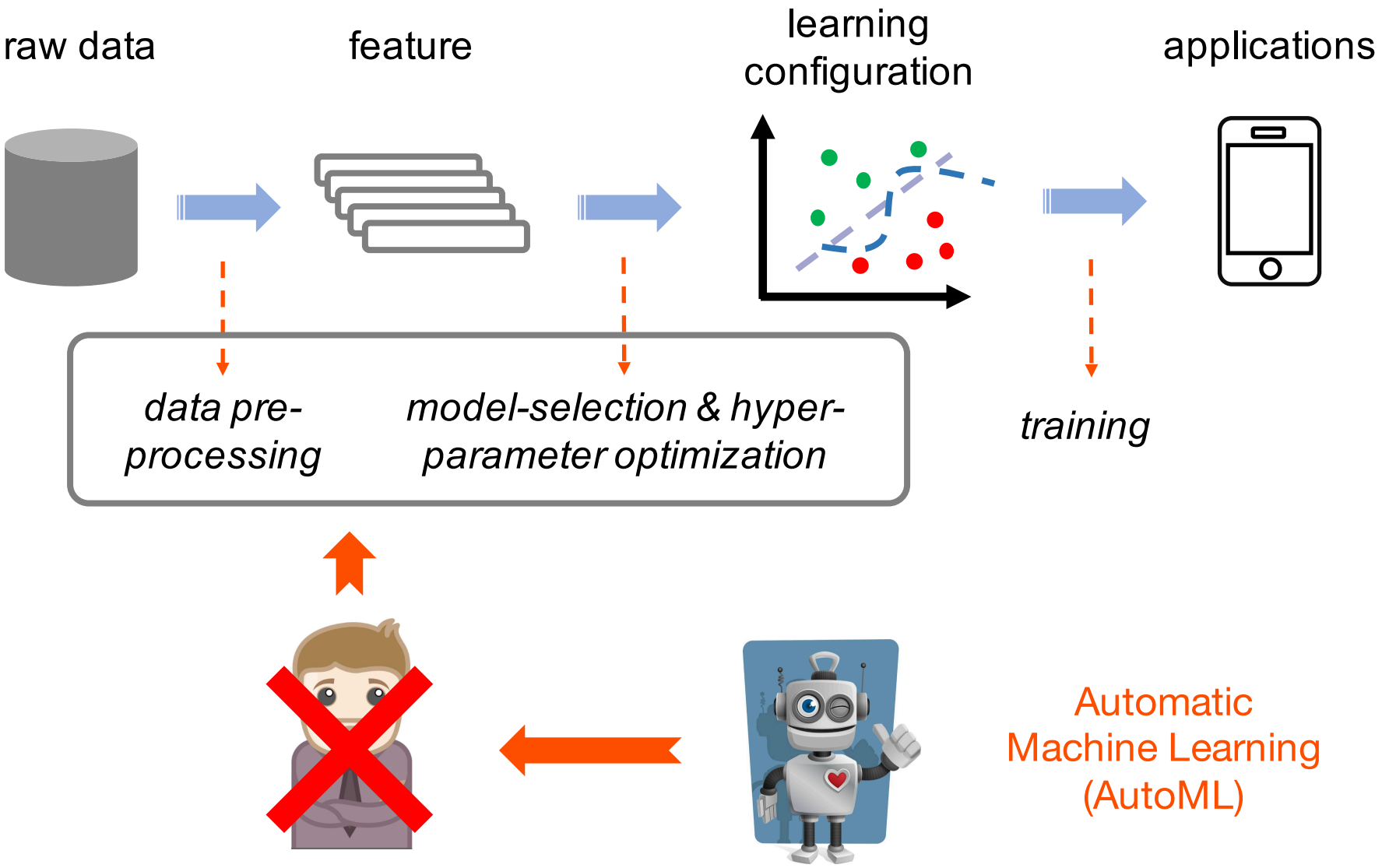
Some Progress from Derivative-Free Optimization to Experienced Derivative-Free Optimization

Yi-Qi Hu

LAMDA Group, Nanjing University, China
4Paradigm Inc., China

2018.08.28

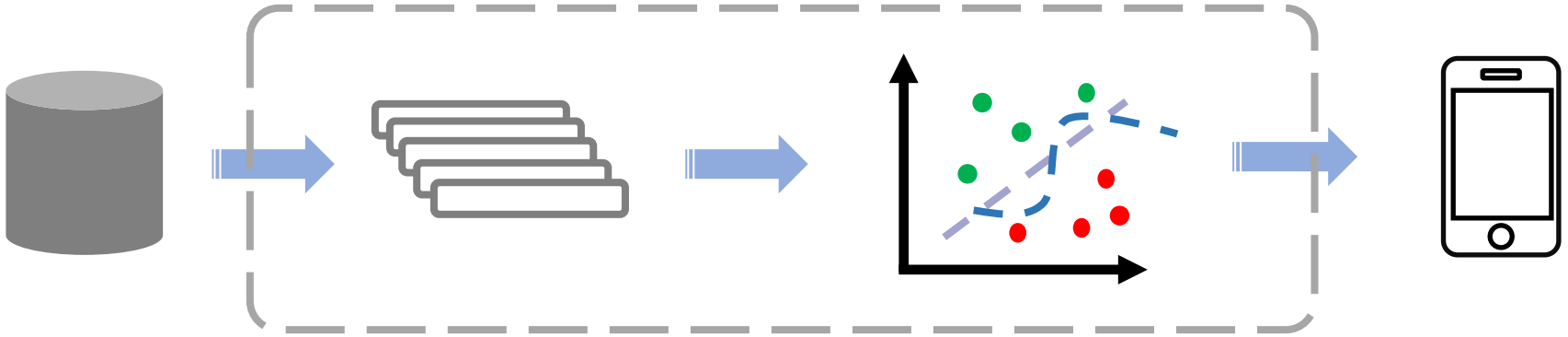
Automatic Machine Learning



Automatic Machine Learning

Purpose of AutoML:

AutoML aims to make machine learning configuration involve less human expert knowledge.



$$A^\star, \boldsymbol{\lambda}_\star \in \underset{A^{(j)} \in \mathcal{A}, \boldsymbol{\lambda} \in \Lambda^{(j)}}{\operatorname{argmin}} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(A_{\boldsymbol{\lambda}}^{(j)}, D_{train}^{(i)}, D_{valid}^{(i)}).$$

learning model → $A^\star, \boldsymbol{\lambda}_\star \in$ ← *dataset*

hyper-parameter → $A^{(j)} \in \mathcal{A}, \boldsymbol{\lambda} \in \Lambda^{(j)}$

$\frac{1}{K} \sum_{i=1}^{K}$ ← *k-fold cross validation*

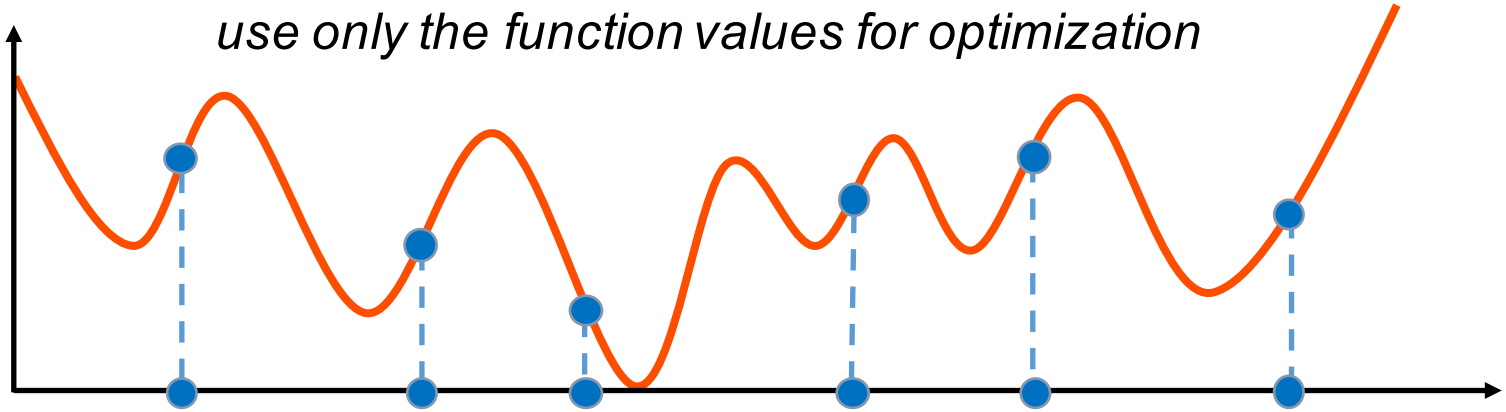
$\mathcal{L}(A_{\boldsymbol{\lambda}}^{(j)}, D_{train}^{(i)}, D_{valid}^{(i)})$ ← *loss function*

Derivative-Free Optimization (DFO)

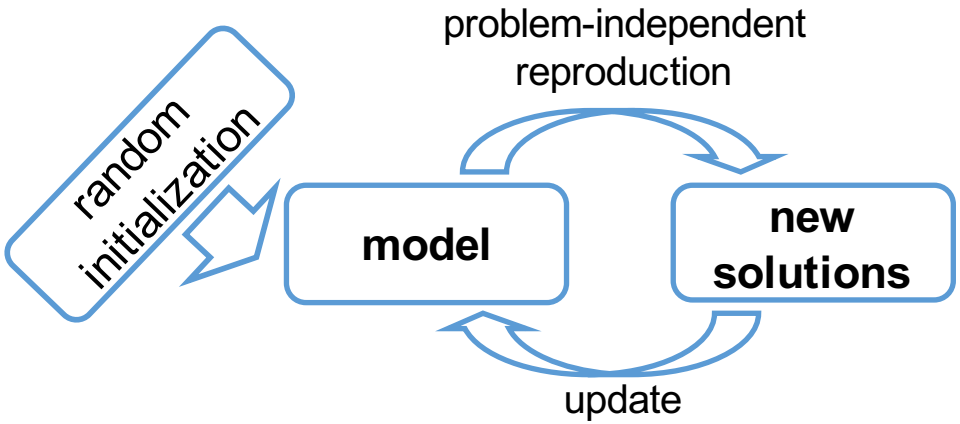
DFO Background

Optimization problems:

$$\arg\min_{x \in X} f(x)$$



Algorithm structure:



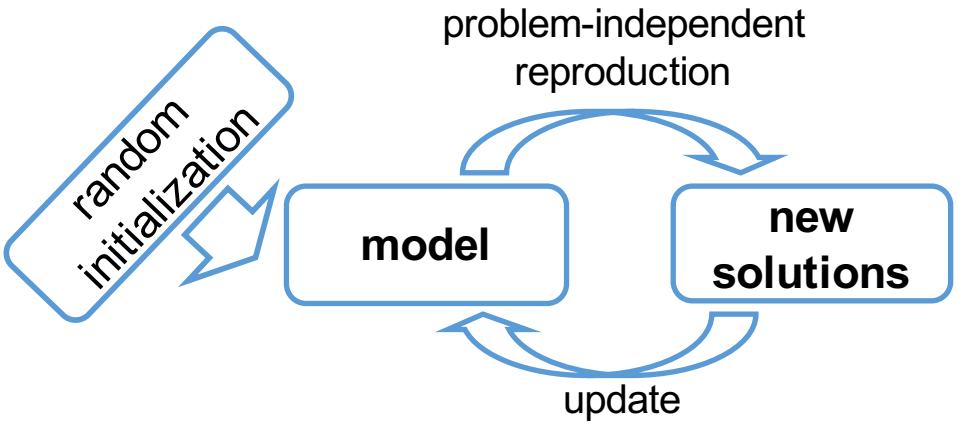
weak theoretical support
- CMA-ES [Hansen, 2013]
- Cross-entropy [Rubinstein, 1997]

low scalability
- Bayesian optimization [Kawaguchi, 2015]

Can we propose a method which has both theoretical support and high efficiency?

A: classification-based optimization [Yu, et al. 2016]

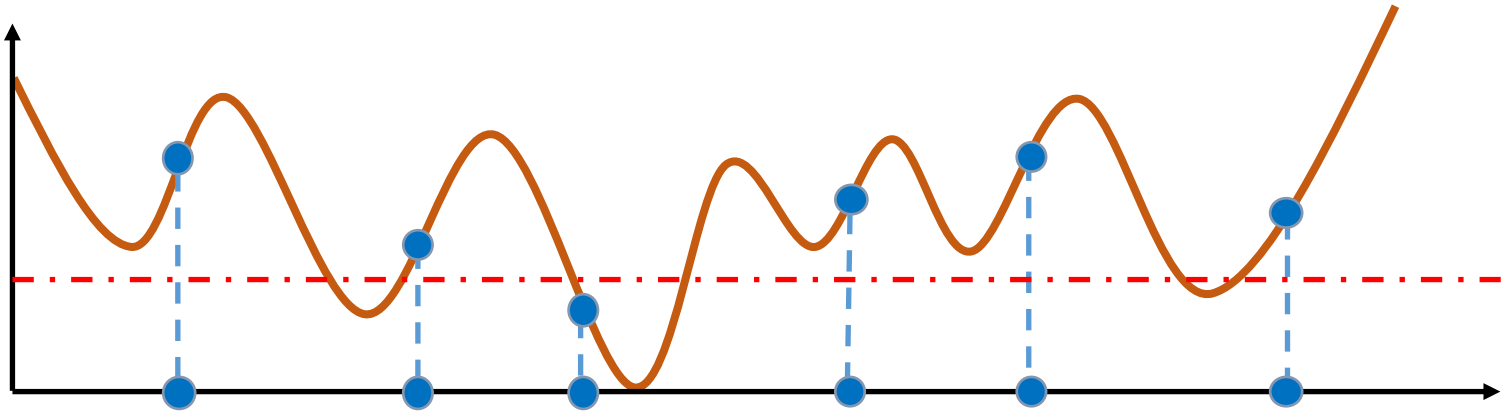
Classification-Based Optimization (I)



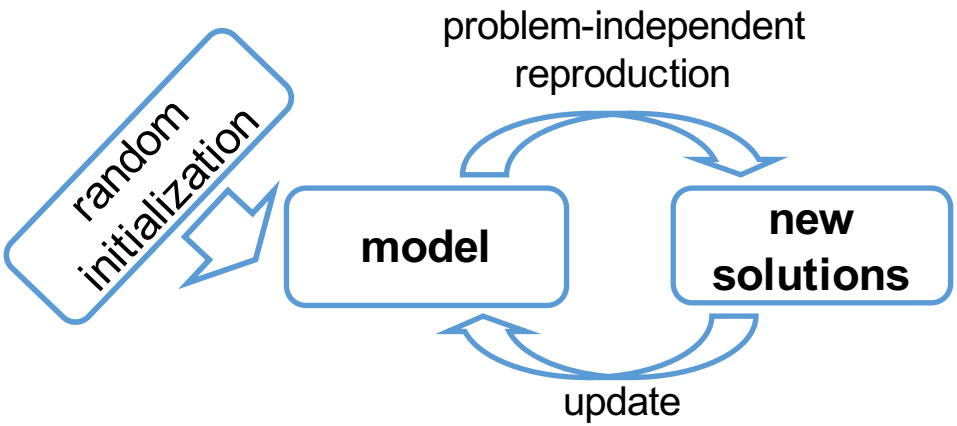
Model: classifier: bounded error

Sampling: uniformly from positive area

Update: learn a new classifier



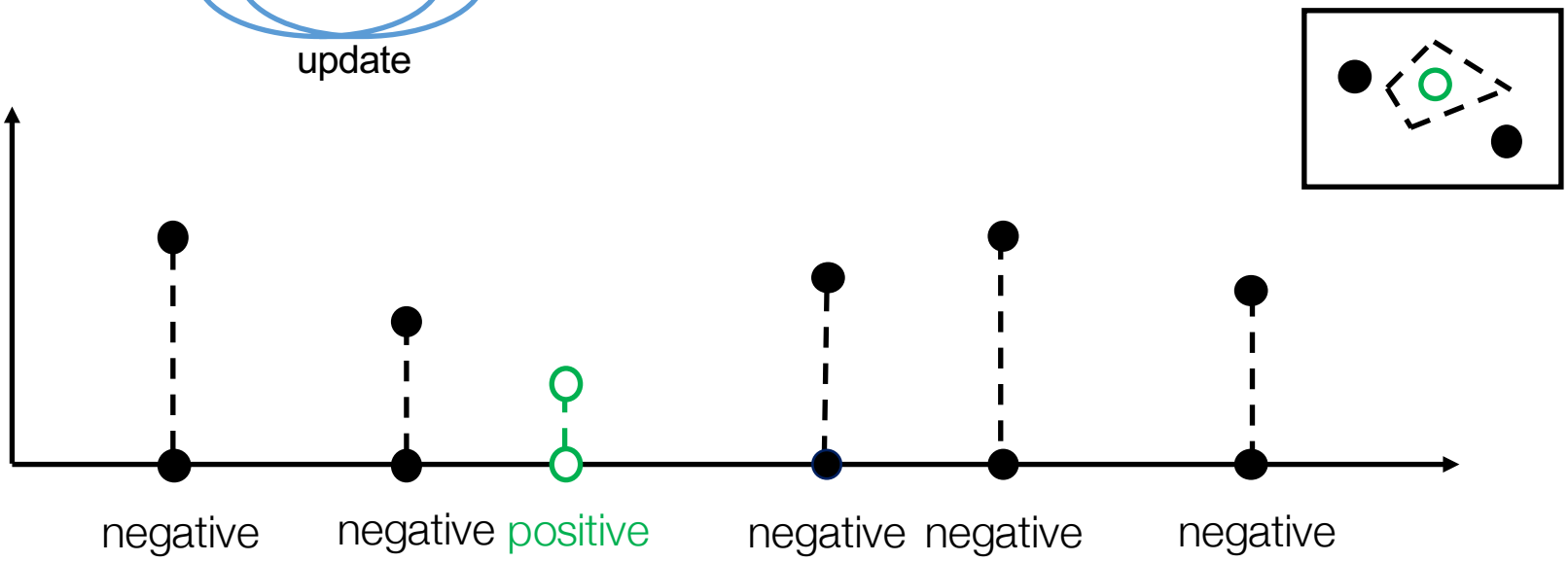
Classification-Based Optimization (I)



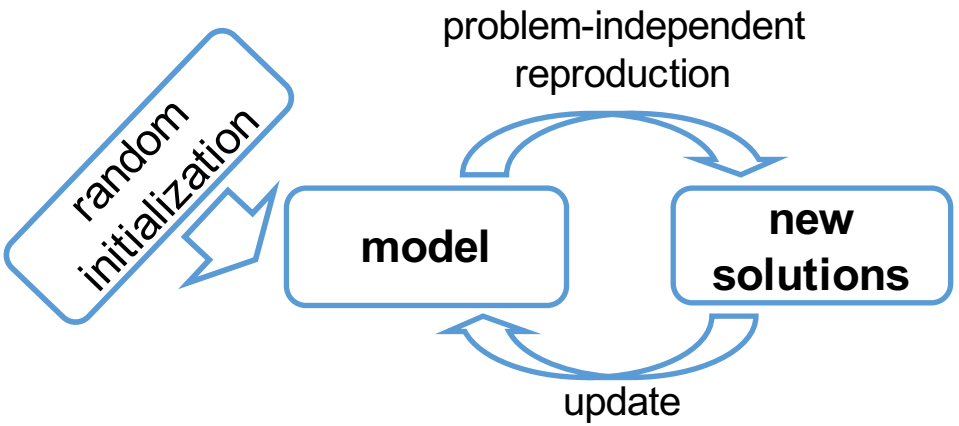
Model: classifier: bounded error

Sampling: uniformly from positive area

Update: learn a new classifier



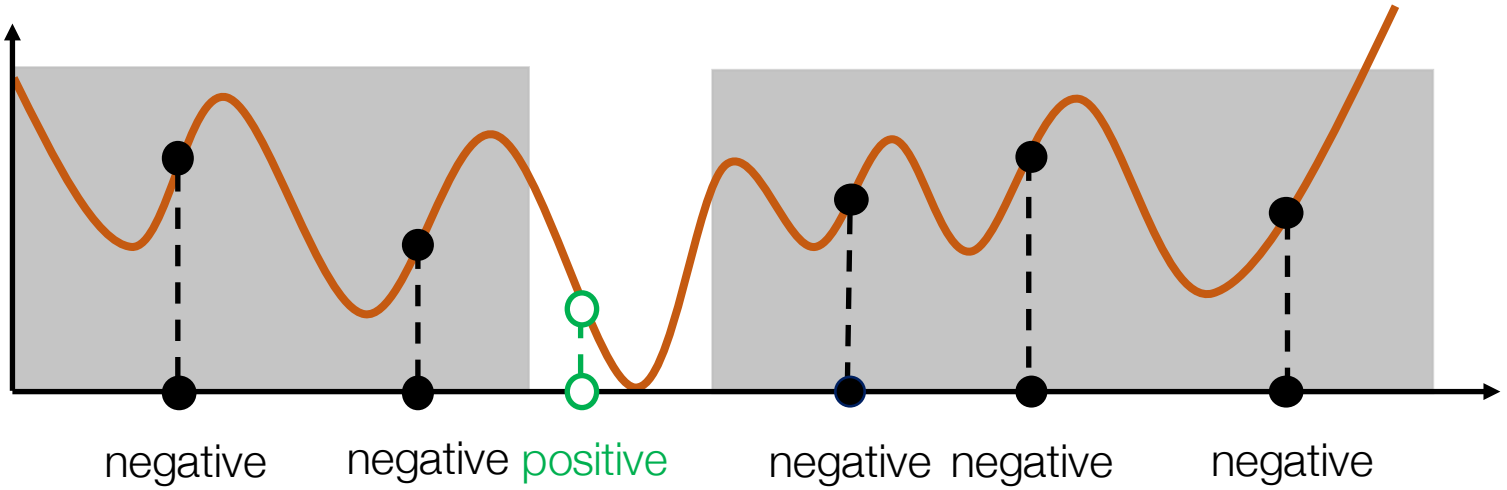
Classification-Based Optimization (I)



Model: classifier: bounded error

Sampling: uniformly from positive area

Update: learn a new classifier



Classification-Based Optimization (II)

Querying complexity:

$$O\left(\frac{1}{|D_\epsilon|}\left((1-\lambda)+\frac{\lambda}{\gamma T}\sum_{t=1}^{T}\frac{1-Q\cdot R_{\mathcal{D}_t}-\theta}{|D_{\alpha_t}|}\right)^{-1}\ln\frac{1}{\delta}\right),$$

$$\text{where } Q=1/(1-\lambda).$$

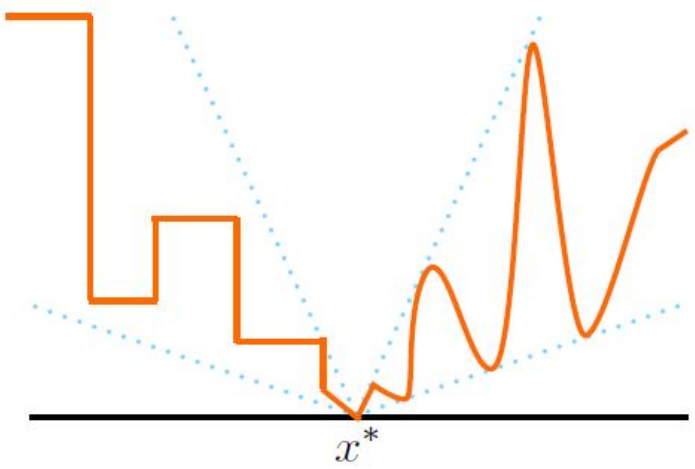
On Local Lipschitz continuous functions:

binary space

$$L_2\|x-x^*\|_H^{\beta_2}\le f(x)-f(x^*)\le L_1\|x-x^*\|_H^{\beta_1}$$

continuous space

$$L_2\|x-x^*\|_2^{\beta_2}\le f(x)-f(x^*)\le L_1\|x-x^*\|_2^{\beta_1}$$



$$poly(\tfrac{1}{\epsilon}, n, \tfrac{1}{\beta_1}, \beta_2, \ln L_1, \ln\tfrac{1}{L_2})\cdot\ln\tfrac{1}{\delta}$$

classification-based optimization is efficient

Classification-Based Optimization (II)

Querying complexity:

$$O\left(\frac{1}{|D_\epsilon|}\left((1-\lambda)+\frac{\lambda}{\gamma T}\sum_{t=1}^{T}\frac{1-Q\cdot R_{\mathcal{D}_t}-\theta}{|D_{\alpha_t}|}\right)^{-1}\ln\frac{1}{\delta}\right),$$

where $Q = 1/(1-\lambda)$.

Smaller Θ the better: the classifier should be highly randomized
Smaller Υ the better: the learnt positive area should be small

Classification-Based Optimization (III)

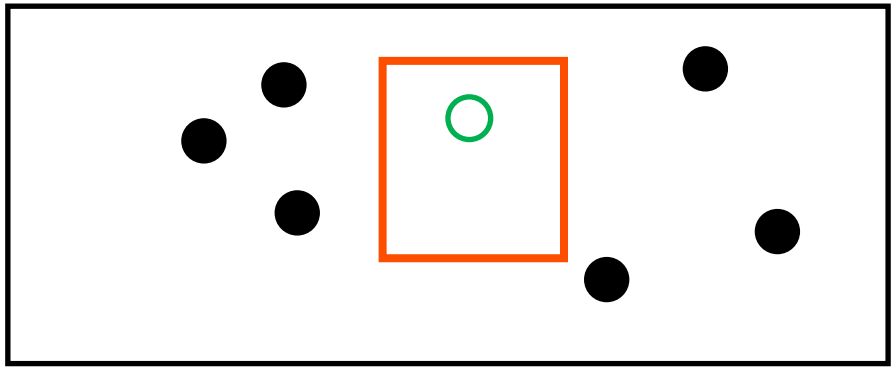
Classification model design:

Considerations:

1. a classifier with a sampled positive area

Implementation:

learn an axis-parallel region



Classification-Based Optimization (III)

Classification model design:

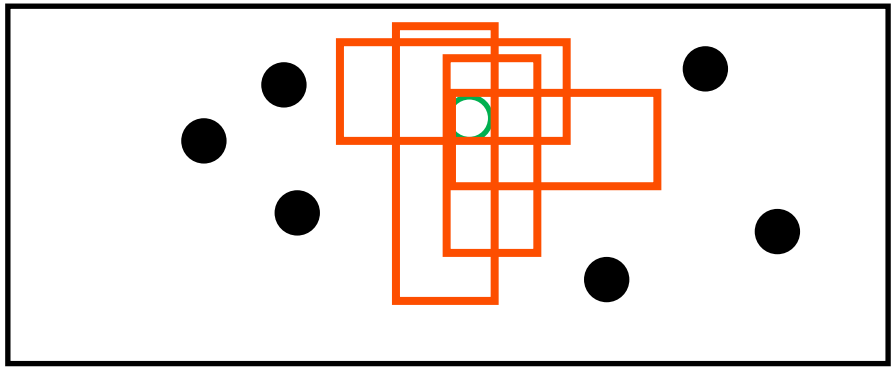
Considerations:

1. a classifier with a sampled positive area
2. smaller Θ -> less dependent

Implementation:

learn an axis-parallel region

with randomness



Classification-Based Optimization (III)

Classification model design:

Considerations:

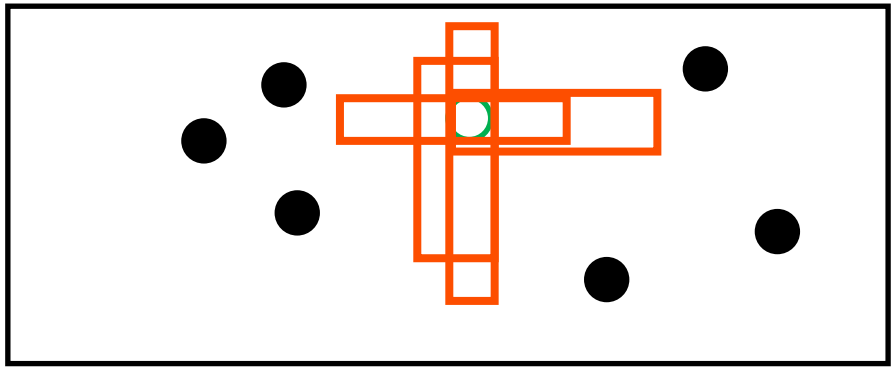
1. a classifier with a sampled positive area
2. smaller Θ -> less dependent
3. smaller γ -> small positive area

Implementation:

learn an axis-parallel region

with randomness

as small as possible



RAndomized COordinate Shrinking (RACOS)

Sequential-Model Optimization (I)

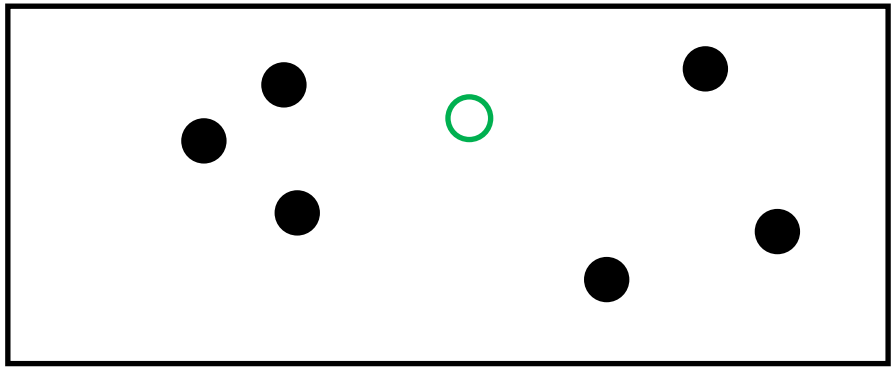
Motivation

RACOS samples and evaluates many solutions in each iteration

in some problem, evaluation is expensive and sequential

can we update the model after each evaluation?

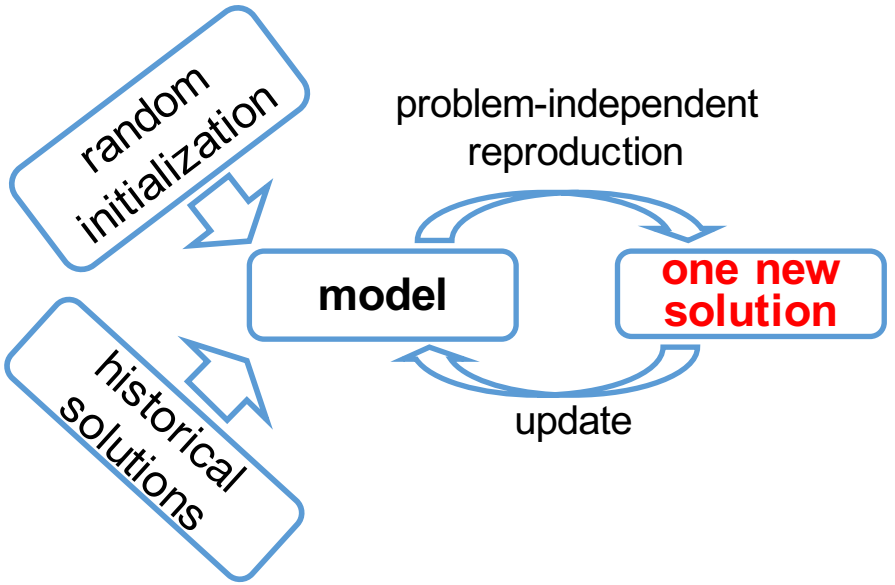
A: sequential-mode RACOS [Hu, et al. 2017]

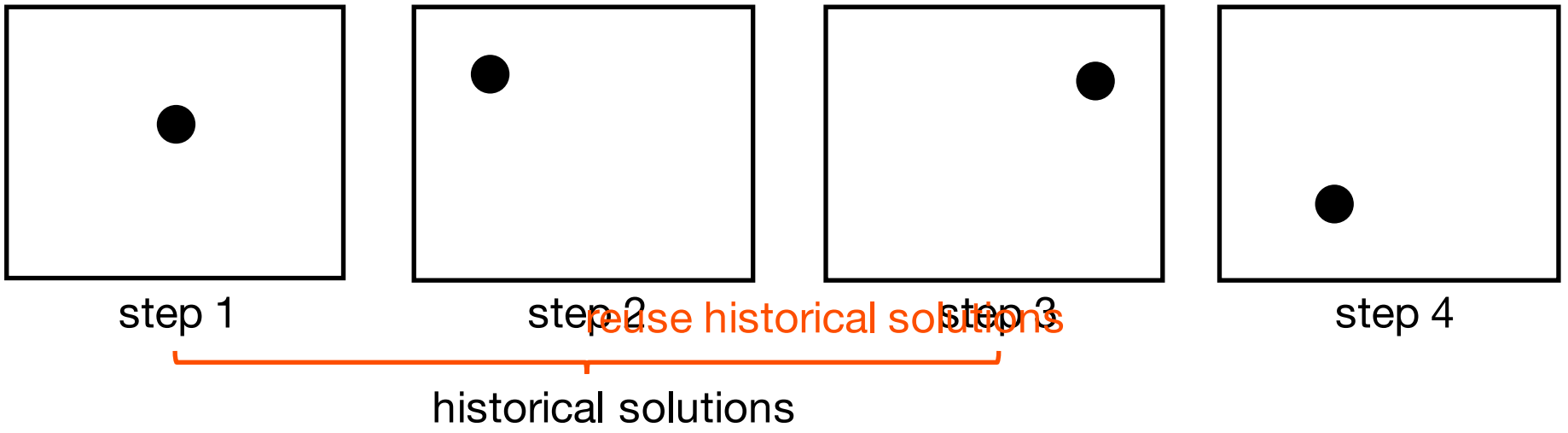


RAndomized COordinate Shrinking (RACOS)

Sequential-Model Optimization (II)

Sequential RACOS (SRACOS)

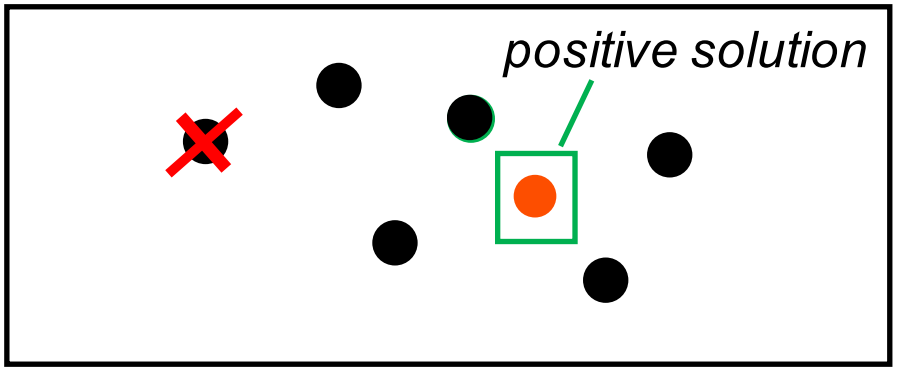




Sequential-Model Optimization (III)

Sequential RACOS (SRACOS)

with fixed size of training data set



Strategies of replacing solution:

- replace the solution with the worst evaluation value (WR)
- replace a randomly selected solution (RR)
- replace the solution having the largest distance to the best-so-fat solution (LM)

Sequential-Model Optimization (III)

Querying complexity:

$$O\left(\frac{1}{|D_\epsilon|}\left((1-\lambda)+\frac{\lambda}{\gamma T}\sum_{t=1}^{T}\frac{1-Q\cdot R_{\mathcal{D}_t}-\theta}{|D_{\alpha_t}|}\right)^{-1}\ln\frac{1}{\delta}\right),$$

where $Q = 1/(1-\lambda)$.

RACOS

$$O\left(\max\left\{\frac{1}{|D_\epsilon|}\left((1-\lambda)+\frac{\lambda}{\gamma(N-r)}\sum_{t=r+1}^{N}\Phi_t\right)^{-1}\ln\frac{1}{\delta}, N\right\}\right),$$

where $\Phi_t = \left(1-R_{\mathcal{D}_t}-\#X\sqrt{\frac{1}{2}D_{KL}(\mathcal{D}_t\|\mathcal{U}_X)}-\theta\right)\cdot|D_{\alpha_t}|^{-1}$ and $\#X$ is the volume of X.

SRACOS

THEOREM 2

Ignoring the constant factor and fixing θ and γ, the sequential classification-based optimization algorithm can have a better (or worse) query complexity upper bound than the batch-mode if for any iteration t

$$R_{\mathcal{D}_t^S} < (\text{or } >)\frac{1}{1-\lambda}R_{\mathcal{D}_t^B} - \#X\sqrt{\frac{1}{2}D_{KL}(\mathcal{D}_t^S\|\mathcal{U}_X)}.$$

sequential can be better

Experiments (I)

Compared methods:

CMA-ES [Hansen, 2013]: evolutionary strategies in state-of-the-art;

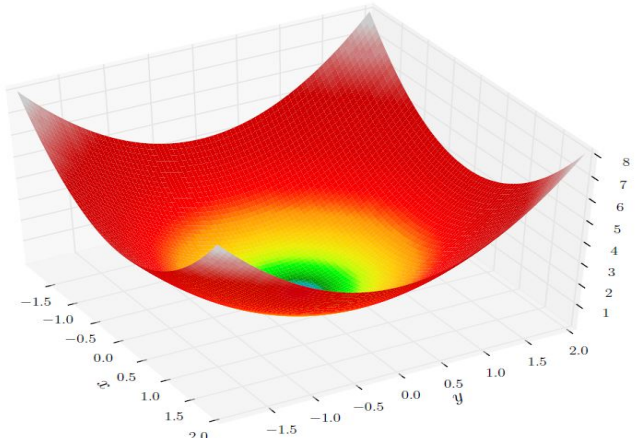
DE [Storn, 1997]: differential evolution;

CE [Rubinstein, 1997]: cross entropy;

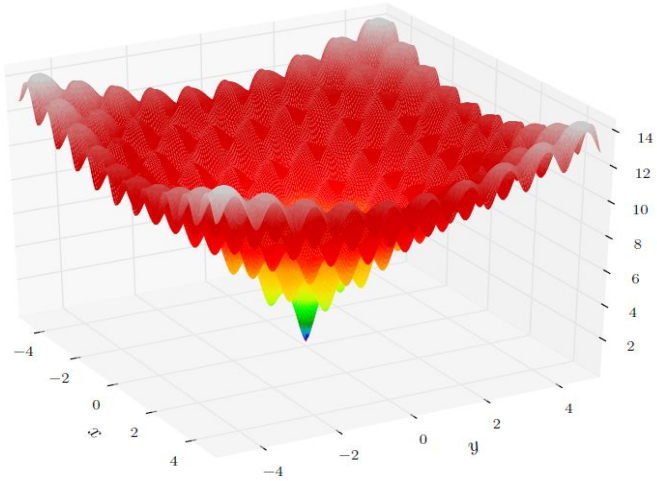
IMGPO [Kawaguchi, 2015]: a Bayesian optimization method;

RACOS [Yu, 2016]: batch-based RACOS.

Selected functions:



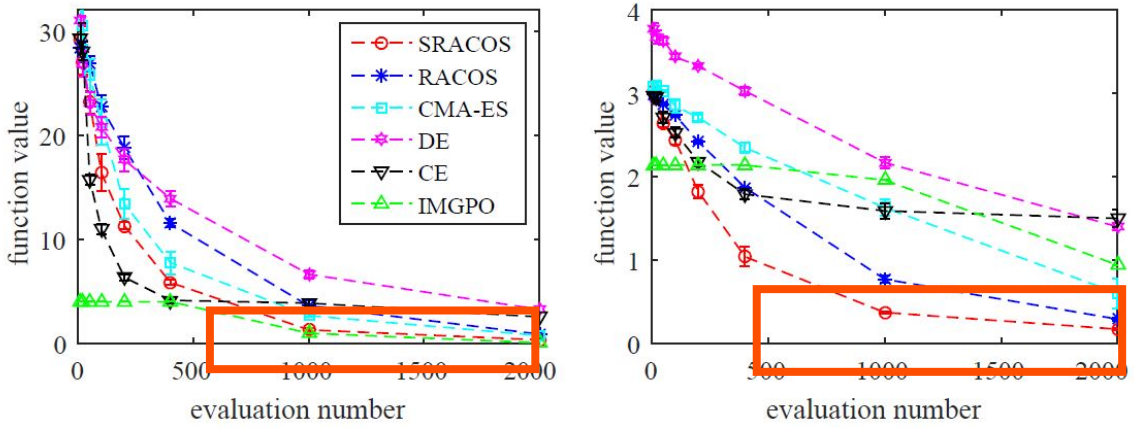
Sphere



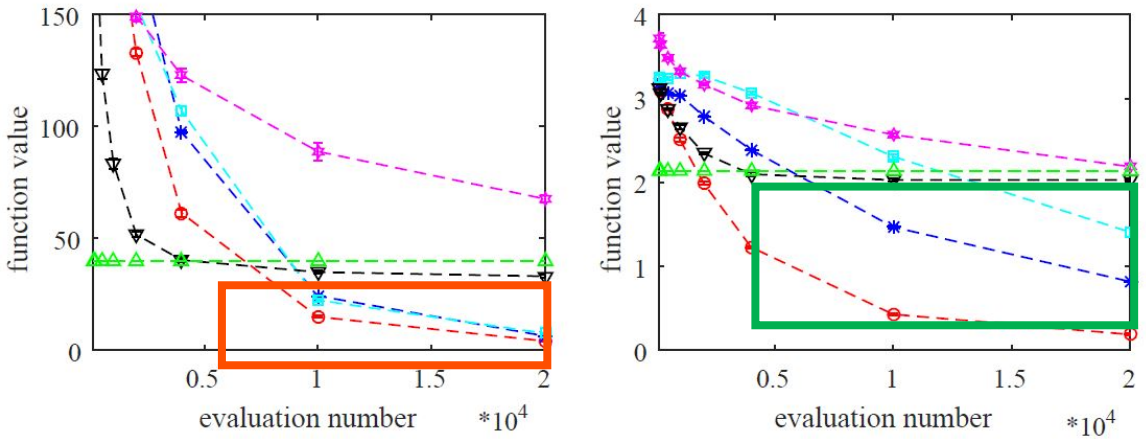
Ackley

Experiments (II)

Convergence:



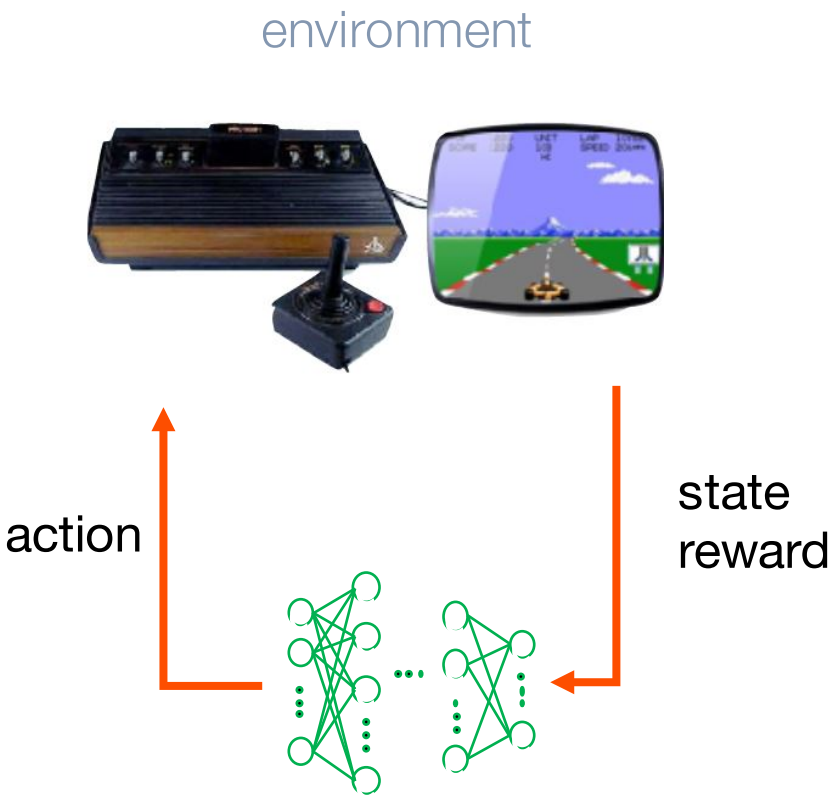
(a) Sphere, $n = 100$ (b) Ackley, $n = 100$

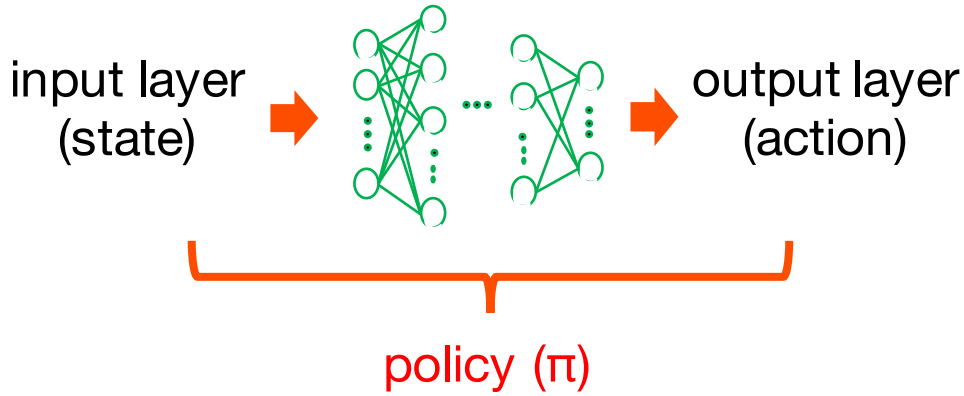


(c) Sphere, $n = 1000$ (d) Ackley, $n = 1000$

Experiments (III)

Direct policy search on reinforcement learning:



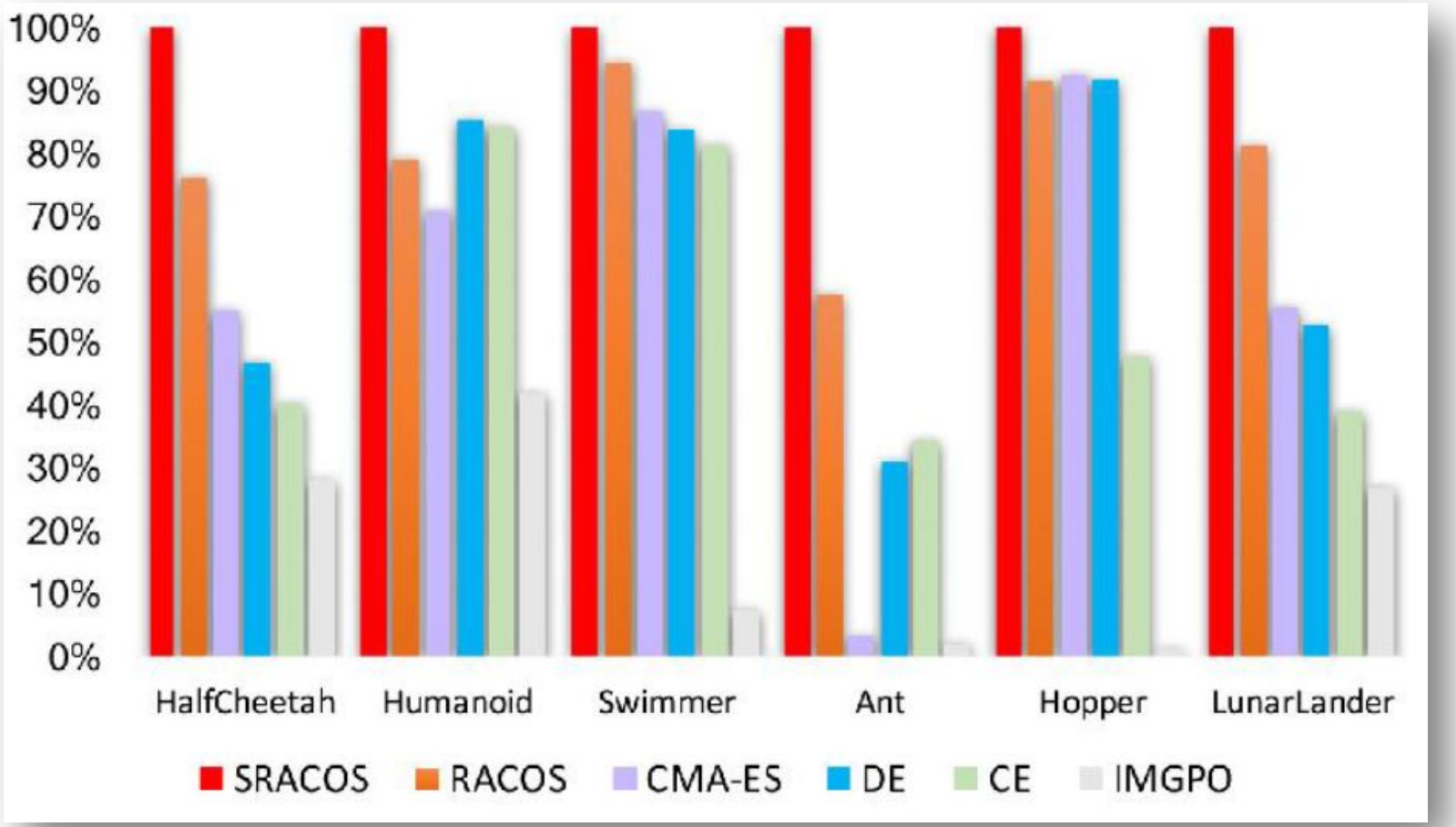


Objective function of policy:

$$f(\pi) = \sum_{i=1}^{T} R_i^{\pi}$$

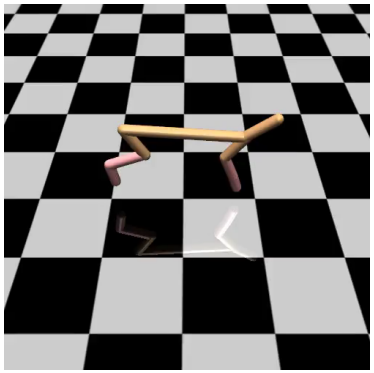
Experiments (IV)

Gym tasks:

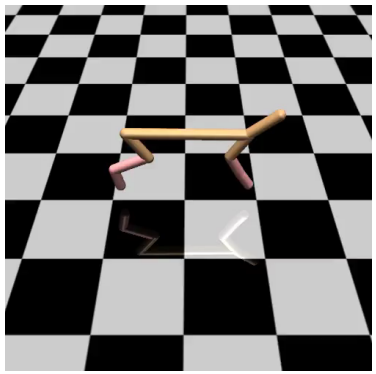


Experiments (IV)

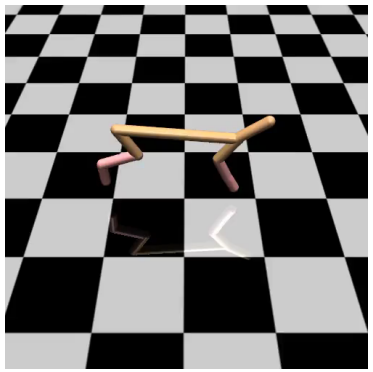
Half-Cheetah:



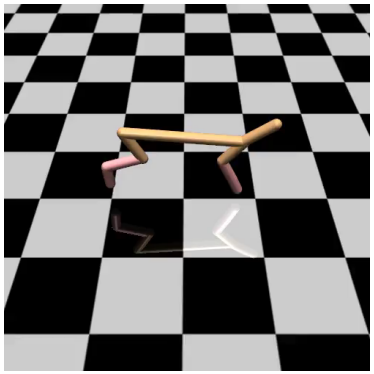
SRACOS



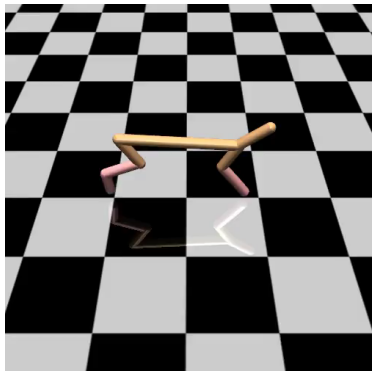
RACOS



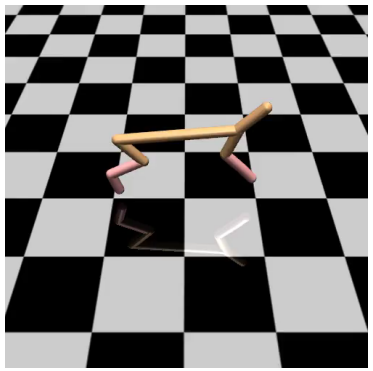
CMA-ES



CE



DE



IMGPO

Is SRACOS efficient enough?

AutoML problem formulation [Feurer et al., 2015]:

high evaluation cost

$$A^\star, \boldsymbol{\lambda}_\star \in \underset{A^{(j)} \in \mathcal{A}, \boldsymbol{\lambda} \in \Lambda^{(j)}}{\operatorname{argmin}} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(A_{\boldsymbol{\lambda}}^{(j)}, D_{train}^{(i)}, D_{valid}^{(i)}).$$

- SMAC [Hutter et al., 2011]
- TPE [Bergstra et al., 2011]
- SRACOS [Hu et al., 2016]

Because can't get gradient, DFO get better sample by random search.

Many evaluations will be spent

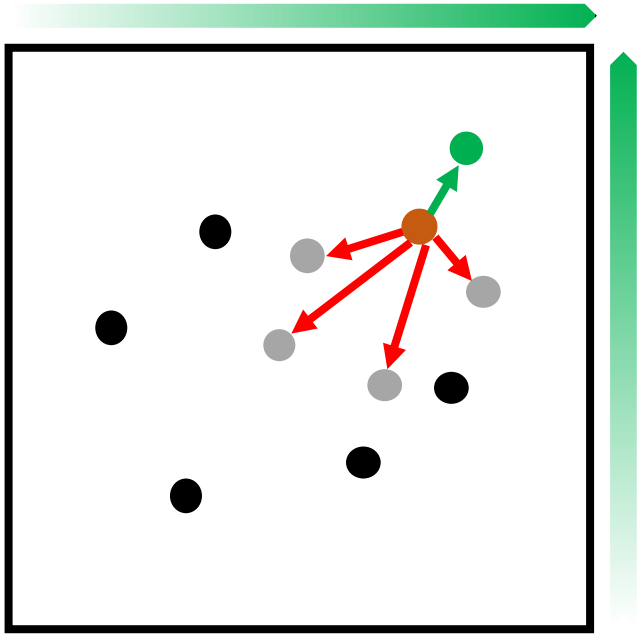
Only a few evaluations are allowed for DFO

SRACOS is not efficient enough!

Why DFO Needs Many Evaluations

Exploration in derivative-free optimization

- the points in i-th iteration
- the best point in i-th iteration
- the points in (i+1)-th iteration
- the points which is better point in i-th iteration



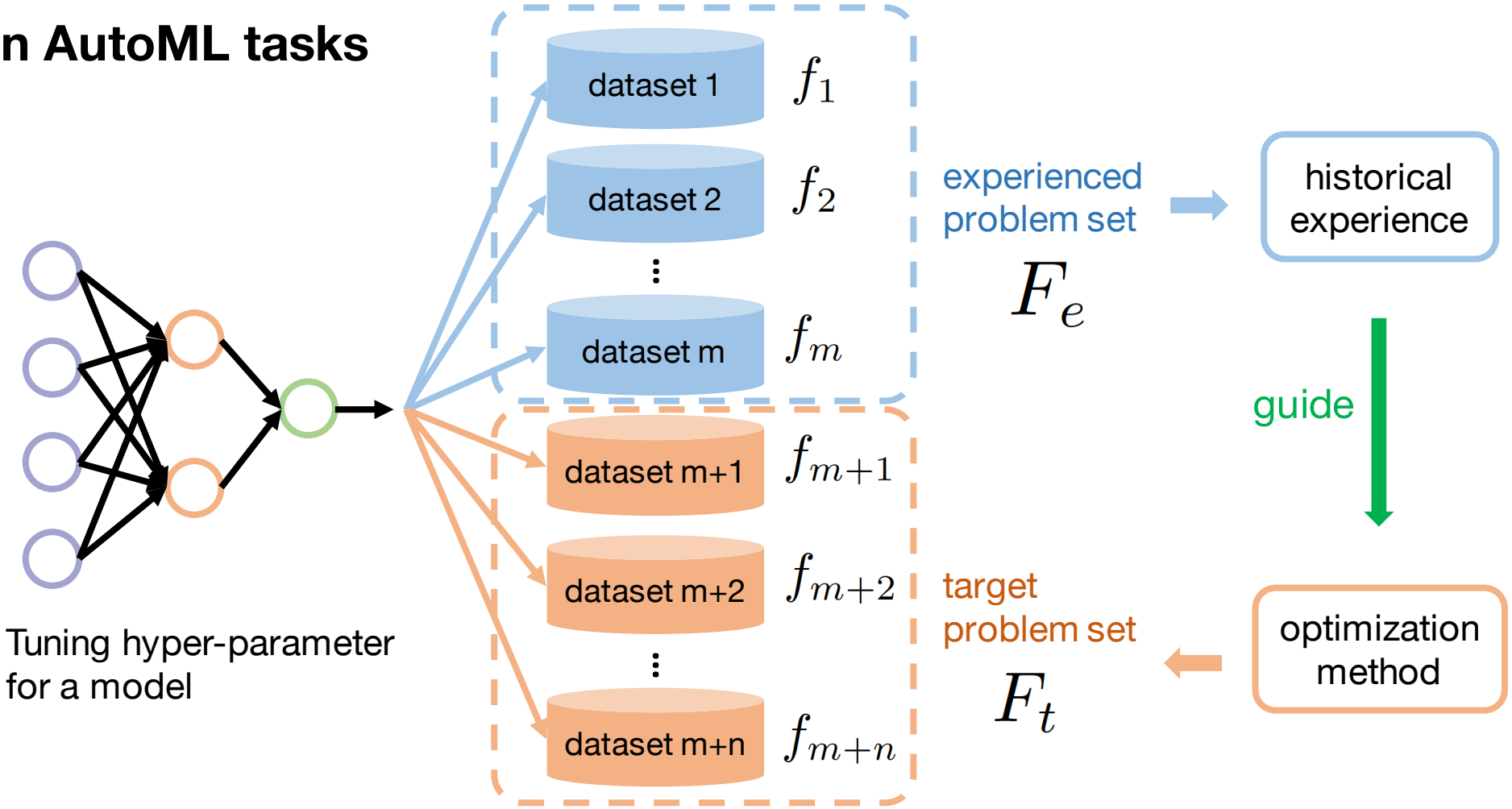
Without gradient, DFO should spend many evaluations on exploration.

If we know the search direction, the exploration on DFO can be avoided.

How can we know the direction?

Consider AutoML Problems' Property

In AutoML tasks



We consider a problem distribution, but not a single problem.

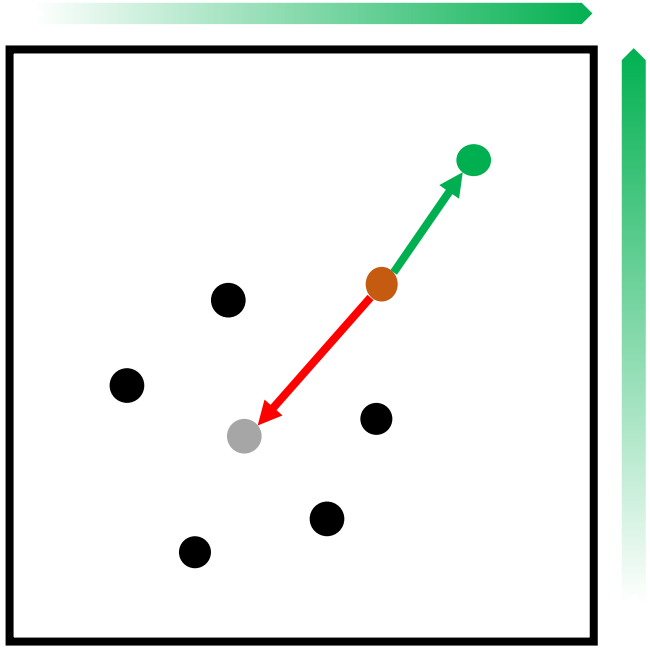
We can get directions from experience! [Hu, et al. 2018]

Get Experience (I)

In i-th iteration of DFO

We can log the stored samples and label the direction to get experience.

- the points in i-th iteration
- the best point in i-th iteration
- the points in (i+1)-th iteration
- the points which is better point in i-th iteration
- the bad sample direction
- the good sample direction



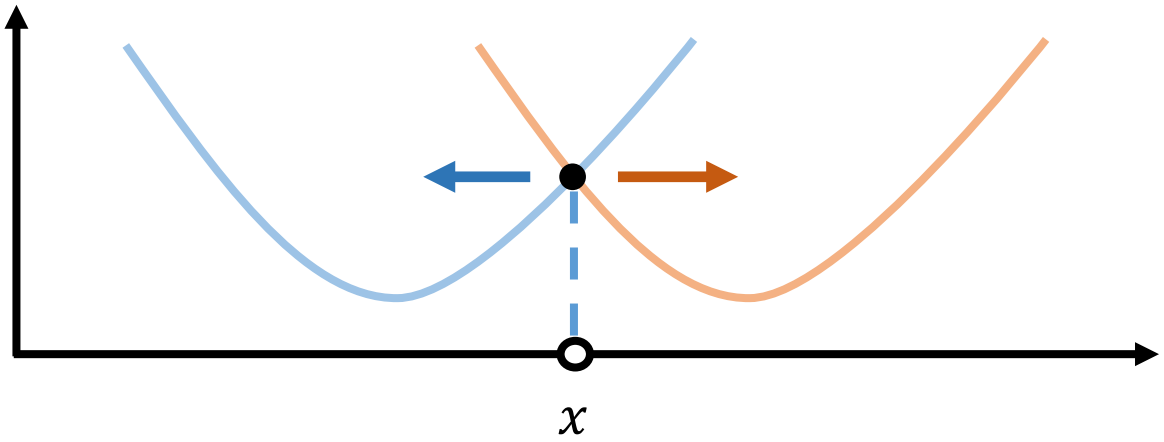
1. sample a bad point, and label the direction is negative
2. sample a good point, and label the direction is positive

Can this experience be used on new problems?

Not directly!

Get Experience (II)

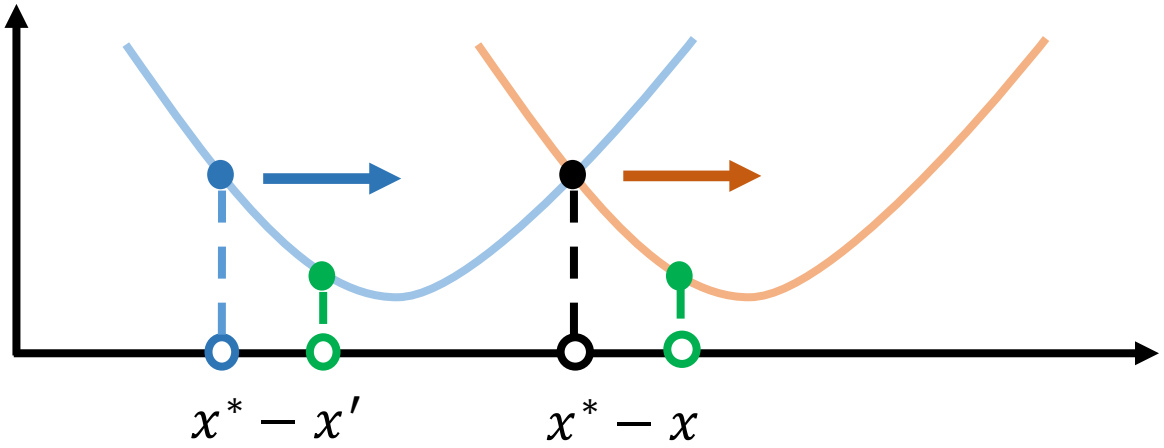
For a simple example



positive direction on experience problem

positive direction on target problem

Centralization



using centralization to align the direction

Get Experience (III)

In 2-D problems

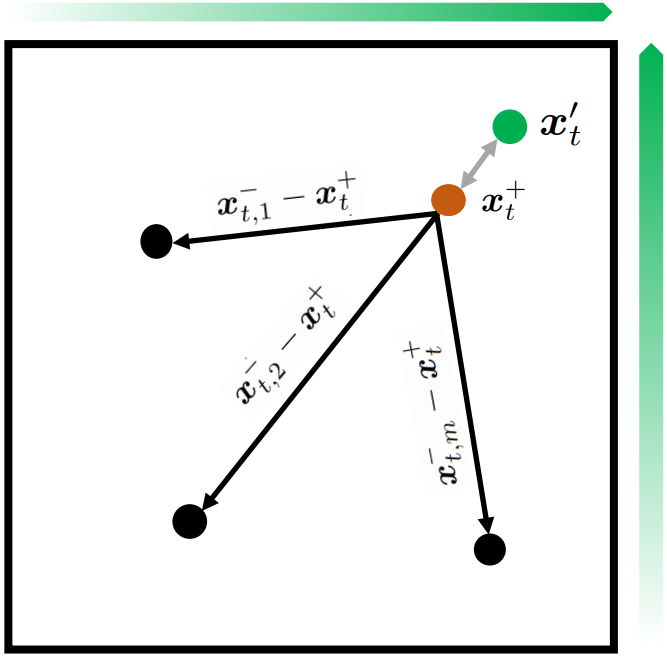
feature:

$$\boldsymbol{\kappa}_t = \begin{bmatrix} \boldsymbol{x}_{t,1}^- - \boldsymbol{x}_t^+ \\ \boldsymbol{x}_{t,2}^- - \boldsymbol{x}_t^+ \\ \vdots \\ \boldsymbol{x}_{t,m}^- - \boldsymbol{x}_t^+ \end{bmatrix}$$

centralization

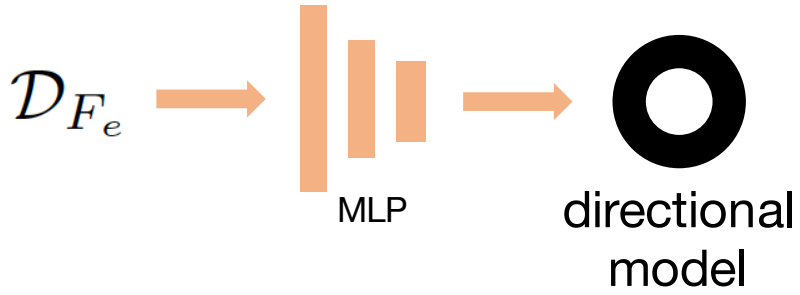
label:

$$\ell_t\left([\boldsymbol{\kappa}_t; \boldsymbol{x}'_t]\right) = \begin{cases} 1, & f(\boldsymbol{x}'_t) < f(\tilde{\boldsymbol{x}}_t); \\ 0, & f(\boldsymbol{x}'_t) \geq f(\tilde{\boldsymbol{x}}_t). \end{cases}$$



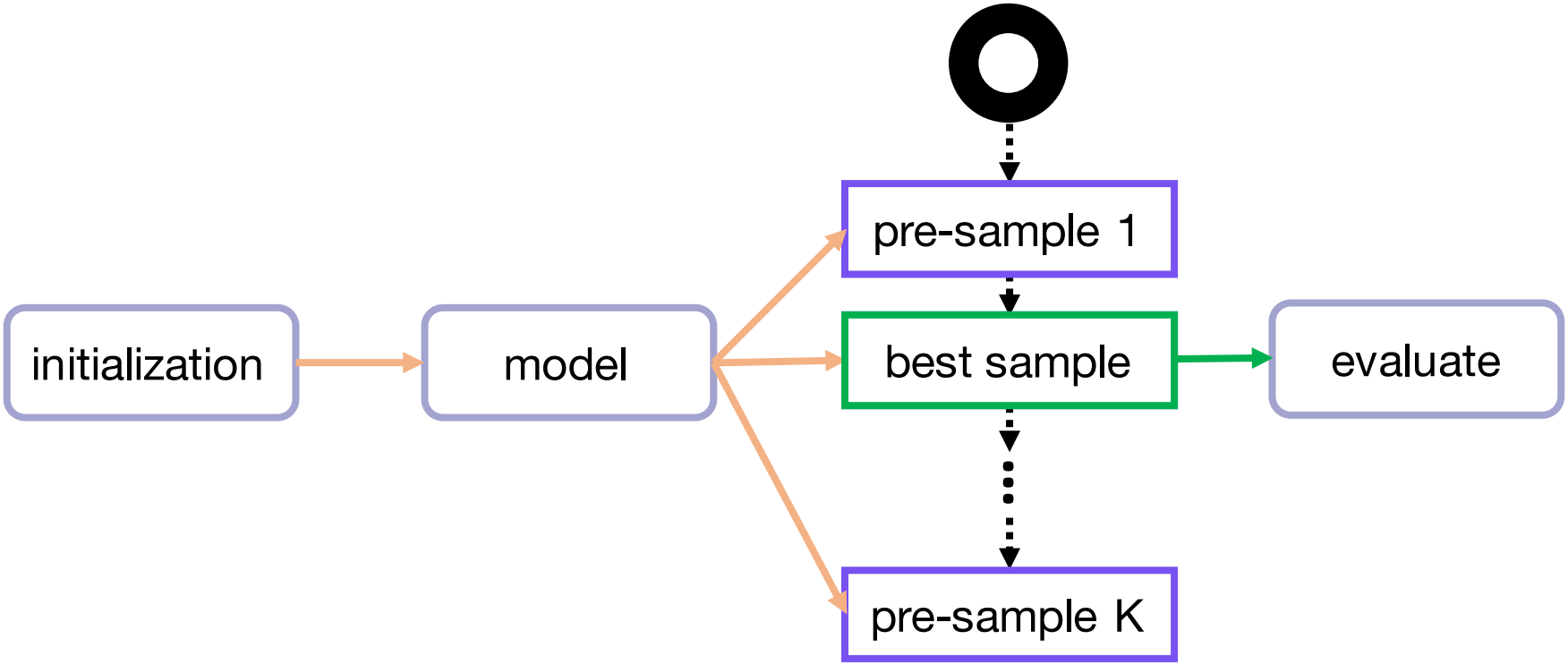
$$\mathcal{D}_{F_e} = \{([\boldsymbol{\kappa}_1; \boldsymbol{x}'_1], \ell_1), ([\boldsymbol{\kappa}_2; \boldsymbol{x}'_2], \ell_2), \dots\}$$

Learning the directional model on the experience dataset.



Utilize Experience

Utilizing the directional model to predict the direction of the next sample.



Discussion: The directional model can predict which sample is most valuable to be evaluated. In this way, ExpSRacos improves efficiency by avoiding wasting many evaluations for exploration.

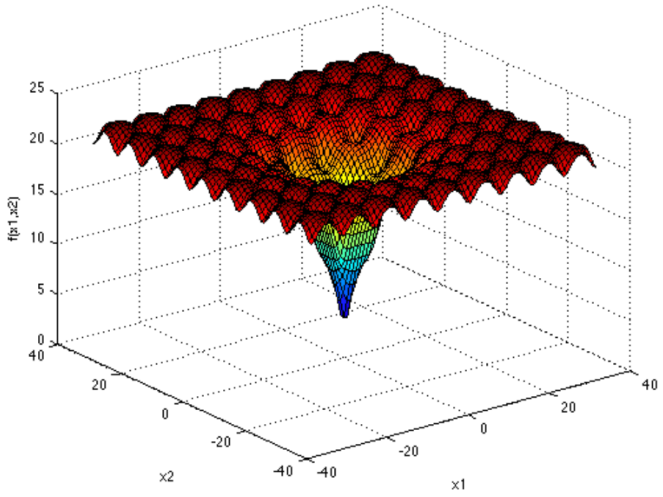
Experiments (I)

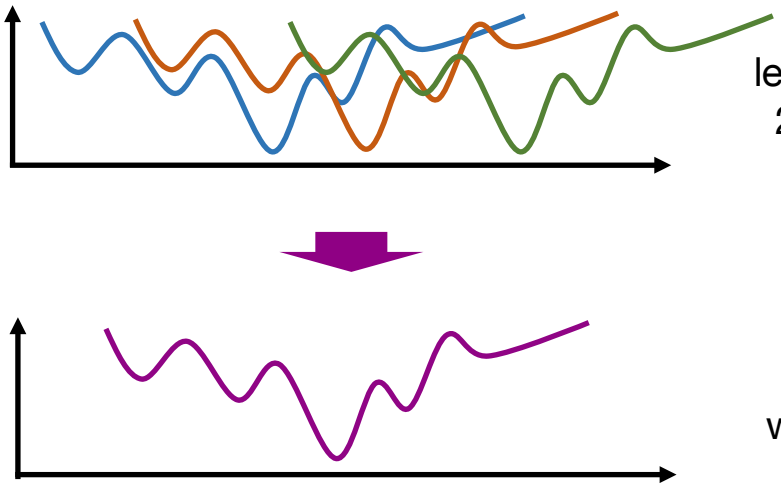
Compared method

We compared ExpSRACOS with CMAES [Hansen et al., 2003], SMAC [Hutter et al., 2011], TPE [Bergstra et al., 2011] and SRACOS [Hu et al., 2017].

Exp 1: On synthetic function

Ackley function: $f(\boldsymbol{x}) = -20e^{(-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - x_i^*)^2})} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi(x_i - x_i^*)} + e + 20.$





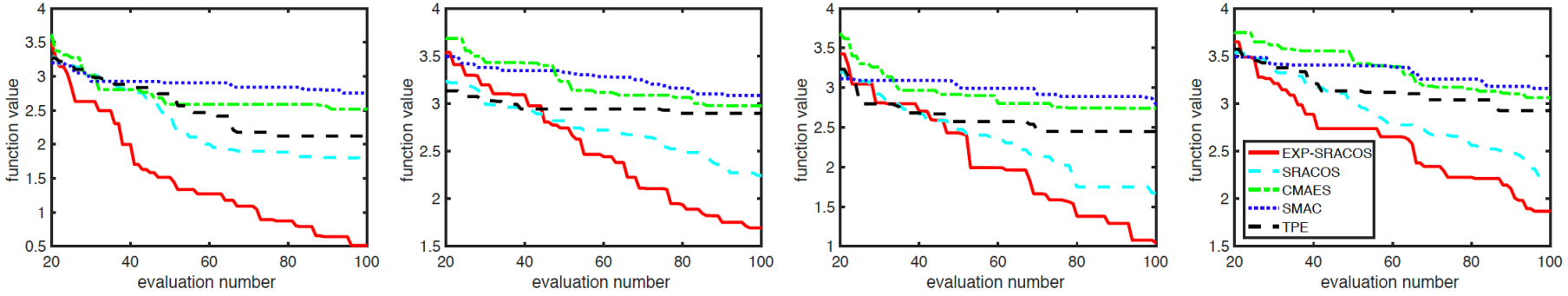
learn directional model from
200 experience problems
with 400 evaluations

test on
100 target problems
with only 100 evaluations

Experiments (II)

Convergence performance:

only 100 evaluations



(a) $x_i^* = 0.04, n = 10$ (b) $x_i^* = 0.04, \quad n = 20$ (c) $x_i^* = 0.2, \quad n = 10$ (d) $x_i^* = 0.2, \quad n = 20$

Easy — Experience directional mode guide optimization on target problems — Hard

Average performance:

We test all compared methods with just only 50 evaluation budget.

Settings		ExpSRacos	SRacos	CMAES	SMAC	TPE
X^{shift}	n					
$[-0.1, 0.1]^n$	10	**1.67**±0.42	2.37±0.36	2.85±0.36	3.01±0.21	2.69±0.26
$[-0.1, 0.1]^n$	20	**2.76**±0.26	2.87±0.20	3.41±0.25	3.31±0.13	3.09±0.17
$[-0.5, 0.5]^n$	10	**2.26**±0.41	2.42±0.36	2.93±0.39	2.95±0.25	2.71±0.29
$[-0.5, 0.5]^n$	20	**2.89**±0.27'	2.98±0.22	3.56±0.28	3.35±0.18	3.18±0.18

Experiments (III)

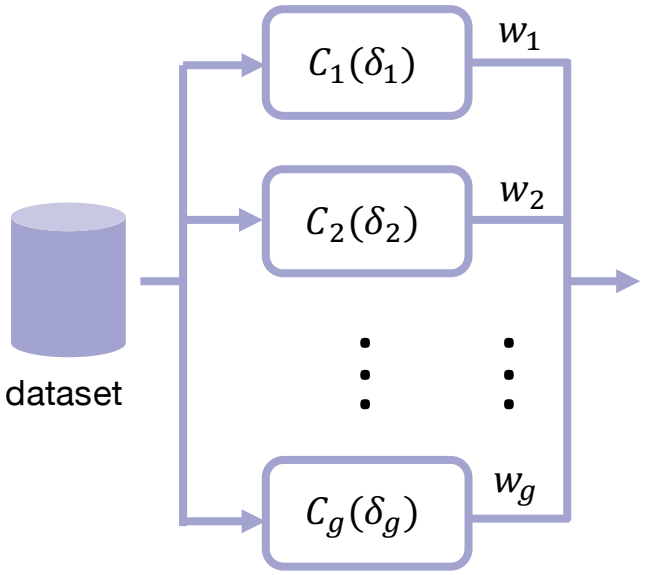
Exp 2: On AutoML tasks

Problem distribution: We optimize hyper-parameters for a specific learning model on different datasets.

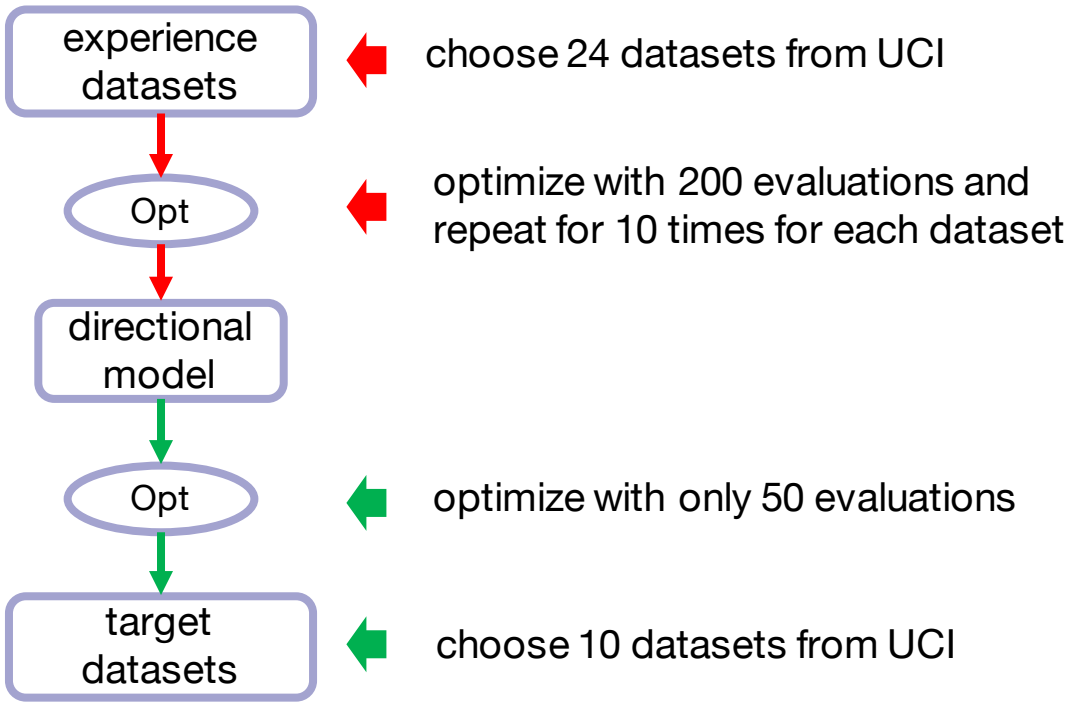
Criterion: k-fold cross validation error rate on training data.

Support: This experiment is based on Scikit-learn [Pedregosa et al., 2011].

Exp 2.1: On ensemble classifier



weighted voting ensemble classifier



Experiments (IV)

Results:

	Dataset	Optimization performance on training samples					Generalization performance on testing samples					Default C_V
		ExpSRacos	SRacos	SMAC	TPE	Best C	ExpSRacos	SRacos	SMAC	TPE	Best C	
	Annealing	.0401●	.0460○	.0877	.0522	.0590	.0000●	.0100○	.0200	.0100○	.0100○	.0400
	Arcene	.1386○	.1656	.2395	.1478	.1257●	.1866○	.2266	.2600	.3000	.1600●	.3700
	Balance S.	.0822●	.0848○	.1142	.1042	.1063	.0834●	.2354	.1666	.1904	.0968○	.2300
	Banknote	.0000●	.0000●	.0000●	.0018	.0000●	.0000●	.0000●	.0000●	.0000●	.0000●	.0000
	Breast C. W.	.0357●	.0405○	.0429	.0538	.0466	.0638○	.0685	.0780	.0567●	.0936	.0638
	Car	.0260○	.0231●	.0744	.0758	.1439	.3410●	.3786	.3526	.3757	.3421○	.3872
	Chess	.0089●	.0096○	.0462	.0497	.0755	.0671●	.1130○	.1312	.1453	.1375	.1109
	CMC	.4315●	.4355○	.4451	.4462	.4520	.4155●	.4346○	.4459	.4391	.4391	.4290
	CNAE9	.0439●	.0444	.0477	.0441○	.0454	.0416●	.0493	.0416●	.0509	.0527	.0555
	Credit	.0889●	.0938○	.1243	.1244	.0981	.2571●	.2757	.2661	.2589○	.3741	.2517
tasets	Cylinder	.1370●	.1506○	.3827	.3189	.4100	.4036○	.3957●	.4220	.4220	.4128	.4128

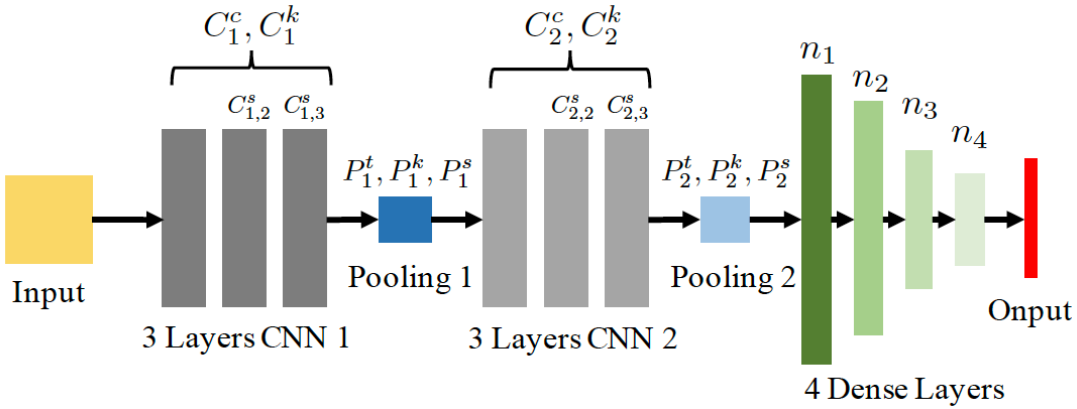
Dataset		Optimization performance on training samples					Generalization performance on testing samples				
		ExpSRacos	SRacos	SMAC	TPE	Best C	ExpSRacos	SRacos	SMAC	TPE	Best C
Source dataset	*1st/2nd/3rd*	**20/4/0**	4/13/5	1/1/6	0/2/9	3/2/4	**15/6/3**	5/5/5	3/4/7	2/5/7	4/9/4
	Avg. rank	**1.1667**	2.2083	3.8750	3.6250	3.6667	**1.5000**	2.9583	3.2500	3.1667	2.7917
Target dataset	*1st/2nd/3rd*	**9/1/0**	2/6/2	0/0/4	0/0/3	0/2/1	**6/3/1**	1/4/2	0/1/3	1/1/2	2/1/3
	Avg. rank	**1.1000**	2.0000	3.6000	3.9000	4.2000	**1.5000**	2.9000	3.9000	3.5000	3.1000

	Seismic	.0624○	.0617●	.0763	.0677	.0657	.0676○	.2121	.0793	.2030	.0657●	.0831
	WDBC	.0351●	.0359○	.0396	.0440	.0440	.0573●	.0695	.0608○	.0869	.0695	.0782
	WPBC	.1715●	.1909	.1833○	.1837	.1971	.1376●	.2032	.1951	.1951	.1463○	.2439
	1st/2nd/3rd	20/4/0	4/13/5	1/1/6	0/2/9	3/2/4	15/6/3	5/5/5	3/4/7	2/5/7	4/9/4	-
	Avg. rank	1.1667	2.2083	3.8750	3.6250	3.6667	1.5000	2.9583	3.2500	3.1667	2.7917	-
	Mushroom	.0000●	.0000●	.0001	.0009	.0407	.1511○	.1619	.1642	.1623	.1023●	.1642
	Occupancy	.0030●	.0032○	.0147	.0085	.0391	.0399○	.0468	.0816	.0392●	.0768	.0509
	Spambase	.0543●	.0557○	.0750	.0782	.0801	.0588●	.0684○	.0716	.0694	.0738	.0966
	Statlog S.	.0205●	.0216○	.0233	.0238	.0254	.0238●	.0339	.0281○	.0432	.0311	.0389
	Wilt	.0165●	.0172○	.0204	.0204	.0202	.0314○	.0146●	.1120	.0420	.1060	.1000
	Wine Q. R.	.3241○	.3177●	.4217	.4209	.4258	.4409●	.4813	.4720	.4751	.4565○	.4472
	Yeast	.4025●	.4103○	.4556	.4561	.4640	.4186●	.4313○	.4417	.4481	.4320	.4983
	Gisette	.0190●	.0203○	.0216	.0210	.0270	.0180●	.0259	.0214	.0210○	.0214	.0270
	Jsbach	.2334●	.2535	.2537	.2578	.2474○	.3135●	.3154○	.3317	.3408	.3723	.4047
	Nursery	.0064●	.0310	.0340	.0713	.0295○	.3141	.3097○	.3350	.3151	.3028●	.3149
	1st/2nd/3rd	9/1/0	2/6/2	0/0/4	0/0/3	0/2/1	6/3/1	1/4/2	0/1/3	1/1/2	2/1/3	-
	Avg. rank	1.1000	2.0000	3.6000	3.9000	4.2000	1.5000	2.9000	3.9000	3.5000	3.1000	-

Target datasets

Experiments (V)

Exp 2.1: On neural network architecture optimization



Hyper-parameter number: 19

Datasets: MNIST and SVHN. MINIST is the source dataset and SVHN is the target dataset.

Experience dataset: We apply SRacos to optimize hyper-parameters on the source dataset with 200 evaluation budget and repeat for 5 times.

optimizing CNN architecture for image classification tasks.

Results:

[Springenberg et al., 2015]

Dataset	optimization error				generalization error				HC-Net
	ExpSRacos	SRacos	SMAC	TPE	ExpSRacos	SRacos	SMAC	TPE	
Source dataset (MNIST)	**.0069**	.0079	.0103	.0093	**.0078**	.0081	.0095	.0091	.0083
Target dataset (**SVHN**)	**.0557**	.0617	.0782	.0772	**.0567**	.0664	.0759	.0796	.0634

Conclusion

AutoML tasks

$$A^\star, \boldsymbol{\lambda}_\star \in \underset{A^{(j)} \in \mathcal{A}, \boldsymbol{\lambda} \in \Lambda^{(j)}}{\operatorname{argmin}} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(A_{\boldsymbol{\lambda}}^{(j)}, D_{train}^{(i)}, D_{valid}^{(i)}).$$

can't get gradient ↓ *high evaluation cost*

High efficient derivative-free optimization

↓ *Theoretical analysis*

Classification-based optimization (RACOS)

↓ *Consider optimization structure*

Sequential classification-based optimization (SRACOS)

↓ *Consider AutoML task property*

Experienced derivative-free optimization

The End

Joint work with:

Prof. Yang Yu

LAMDA Group,
Nanjing University

Ph.D. Hong Qian

LAMDA Group,
Nanjing University

Support by:

4Paradigm

AI for everyone.

4Paradigm Inc., China

Thank you for your attention!