ProxyBO: Accelerating Neural Architecture Search via Bayesian Optimization with Zero-cost Proxies

- 作者：Yu Shen，Yang Li，Jian Zheng et.al.
- 机构：北大、快手
- 会议：arxiv
- 地址：https://arxiv.org/abs/2110.10423
- 代码：暂无

论文主要内容

摘要

之前的NAS方法能搜到promising结果但是慢，zero-cost proxies 运行很快但是搜索结果不够promising。最近有工作将zero-cost proxies 作为简单的warm-up手段。但受unforeseeable reliability 和 one-shot usage两个局限。文章提出ProxyBO来解决这两个问题，达到SOTA。

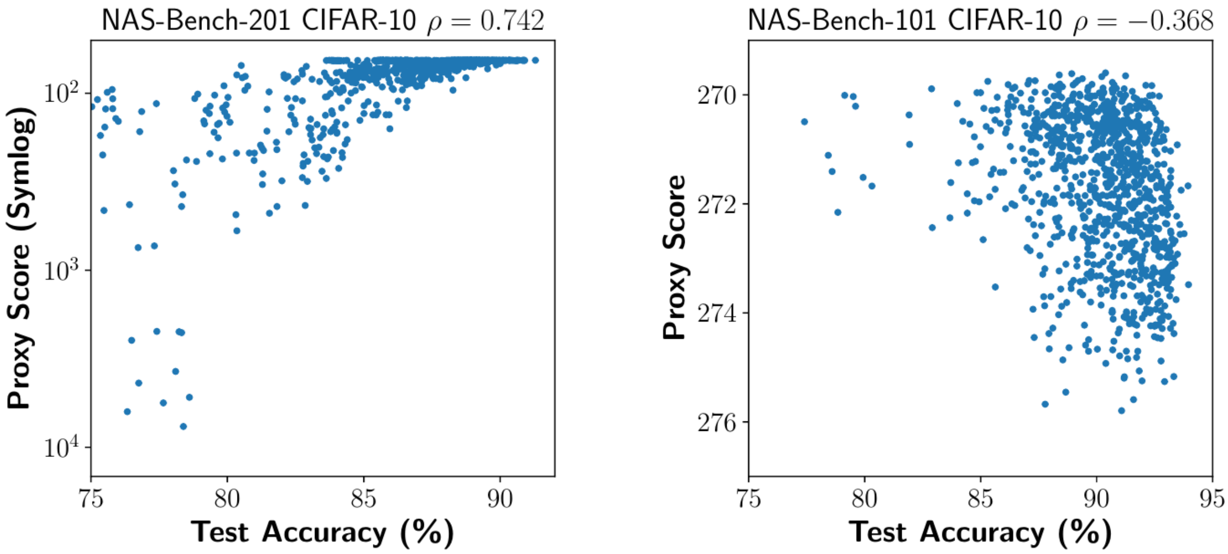
贡献

1. 不需要预先知道proxies对task的适用性（无需计算秩相关系数）**自适应**
2. 第一个effectively组合BO与zero-cost proxies两者优势的算法框架

研究内容

Motivation

- 一方面：目前Baysian Optimization（BO）在NAS上的应用具有的缺点：得到一个好的surrogate需要**相当多的evaluations**，evaluation少的时候surrogate会under-fit
    - 介绍：BO会用已有的评估观测数据去训练一个predictor（surrogate），根据预测结果挑选下一个网络架构做evaluate。已有的工作主要在predictor的选取上（BNN、GNN）
- 另一方面：NAS中使用zero-cost proxies速度很快，但是结果没有BO-based NAS的好
    - 介绍：正常evaluate一个网络架构需要拿去train(耗时)。zero-cost proxies不需要训练来估计网络架构的性能



(a) NAS-Bench-201 CIFAR-10 (b) NAS-Bench-101

Figure 1: Spearman ρ of `jacob_cov` over NAS search spaces using 1000 randomly sampled architectures.

  - 同时zero-cost proxies在不同搜索空间、不同数据集的有效性（秩相关系数）还不一样
  - Unforeseeable reliability：实践中，在真正搜索前秩相关系数是未知的（使用需要先验知识）
  - One-shot usage：先前proxy作为warm-up手段，这样proxy在整个搜索过程中是一次性的作用（在搜索的后期中难以消除bad proxies的影响）。
- 这篇文章就通过结合BO与zero-cost proxies的优点来加速NAS

Intuition

evaluations少的时候用proxies、evaluations多的时候用BO（**dynamic importance**）

方法

介绍

把NAS看作黑盒优化问题：

$$\operatorname{argmin}_{x\in\mathcal{X}} f_{obj}(x)$$

$\mathcal{X}$ 是NAS中的search space， f_{obj} 是架构的性能metric（对于分类任务：在val集上的分类误差）

Zero-cost Proxy

简化讨论：这里所有proxy score都一致设为越小代表越好

- snip
  - $S(\theta)=\left|\frac{\partial L}{\partial \theta} \odot \theta\right|$
  - $P(x)=-\sum_{\theta \in \Theta} S(\theta)$
- Synflow
  - $S(\theta)=\frac{\partial L}{\partial \theta} \odot \theta$
  - $P(x)=-\sum_{\theta \in \Theta} S(\theta)$
- jacob cov
  - $P(x)=-\log \left|K_H\right|$

Table 1: Spearman ρ of proxies for all (and top-10%) architectures in NAS spaces. “NB2” refers to NAS-Bench-201.

	snip	synflow	jacob_cov
NAS-Bench-101	-0.16 (-0.00)	0.37 (0.14)	-0.38 (-0.08)
NB2 CIFAR-10	0.60 (-0.36)	0.72 (0.12)	0.74 (0.15)
NB2 CIFAR-100	0.64 (-0.09)	0.71 (0.42)	0.76 (0.06)
NB2 ImageNet16-120	0.58 (0.13)	0.70 (0.55)	0.75 (0.06)
NAS-Bench-ASR	0.03 (0.13)	0.41 (-0.01)	-0.36 (0.06)

no proxy dominates the others on all the tasks（没有免费的午餐）

核心在于怎么去评判这些indicator的相对重要性

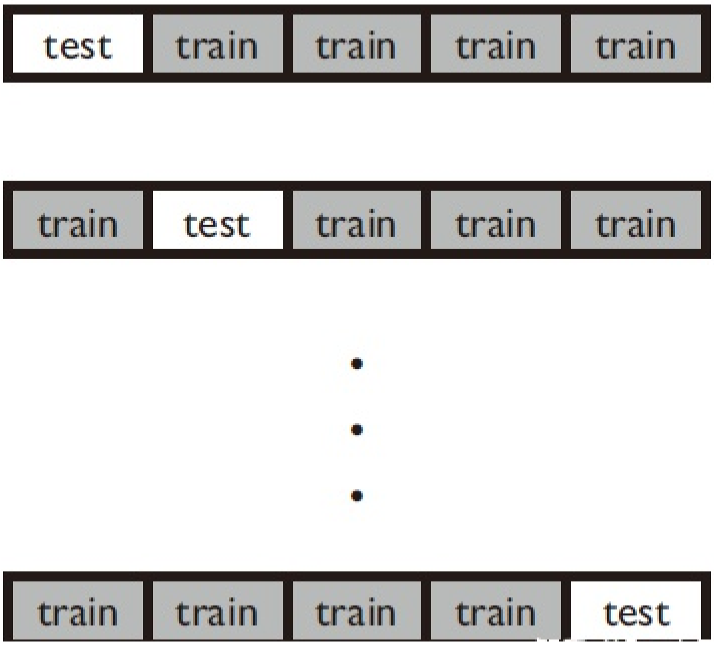
测定indicator泛化能力

$$F\left(P_i ; D\right)=\sum_{j=1}^{|D|} \sum_{k=j+1}^{|D|} 1\left(\left(P_i\left(x_j\right)<P_i\left(x_k\right)\right) \otimes\left(y_j<y_k\right)\right)$$

实际意义：proxy能够正确rank架构的能力（越大越好）

测定BO surrogate泛化能力

k-fold cross-validation



$$F(M;D)=\sum_{j=1}^{|D|}\sum_{k=j+1}^{|D|}1\left(\left(M_{-f(x_j)}(x_j)<M_{-f(x_k)}(x_k)\right)\otimes(y_j<y_k)\right)$$

泛化能力的统一表示

$$G(\cdot;D)=2F(\cdot;D)/(|D|*(|D|-1))$$

有了这个measurement，就能在搜索过程中判定proxies、BO surrogate的对NAS的贡献（权重）

动态加权组合

- 一般的BO迭代中会选择使得acquisition function最大的一个样本作为将要evaluate的架构
- ProxyBO对candidates做一个rank，rank第一的candidate样本作为将要evaluate的架构
- rank具体计算方式：combine rank：

$$CR(x_j)=I(M;D)R_M(x_j)+\sum_{i=1}^{K}I(P_i;D)R_{P_i}(x_j)$$

$$I(\cdot;D)=\frac{\exp(G(\cdot;D)/\tau)}{\sum\exp(G(\cdot;D)/\tau)},\tau=\frac{\tau_0}{1+\log T}$$

- I 理解为权重， τ 温度参数控制softmax， T 迭代次数来控制温度

作用：

1. 在搜索过程中逐渐降低Proxies的权重，增加BO的权重
2. 在搜索过程中不同proxy可以无先验知识的使用（不知道哪个proxy对于当前task是有效的）

算法流程

生成下一个suggest sample

Algorithm 1: Pseudo code for *Sample* in ProxyBO

Input: the observations D, the current number of iteration T, the number of sampled configurations Q, the Bayesian optimization surrogate M, the zero-cost proxies $P_{1:K}$, and the temperature hyper-parameter τ_0.

Output: the next architecture configuration to evaluate.

```
if |D| < 5, then return a random configuration.
compute G(·; D) for each proxy and the surrogate.
compute I(·; D) according to Eq. 8.
draw Q configurations via random and local sampling.
compute the Expected Improvement (EI) based on surrogate M according to Eq. 1 and the proxy values for P_{1:K} according to Eq. 2, 3, 4 for each sampled configuration.
rank the Q configurations and obtain the ranking value of configuration x_j as R_M(x_j) and R_{P_i}(x_j) for the i-th proxy.
calculate the combined ranking CR(x_j) for each configuration x_j according to Eq. 7.
return the configuration with the lowest combined ranking value.
```

整体算法流程

Algorithm 2: Pseudo code for the framework of ProxyBO

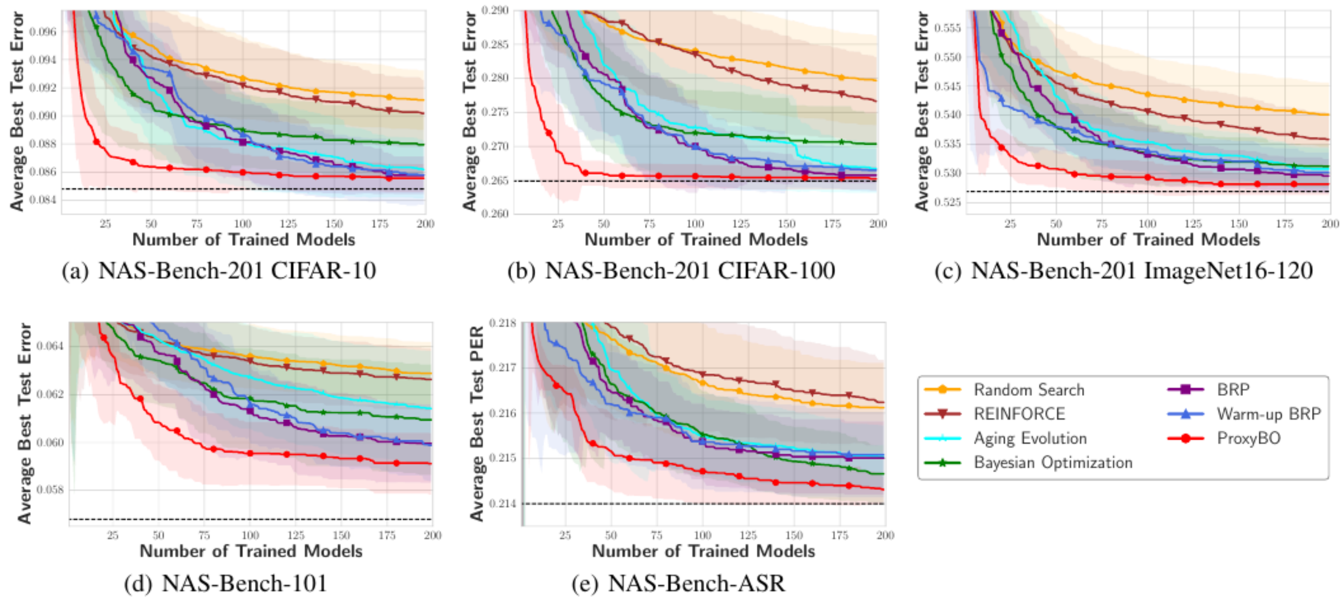
Input: the search budget $\mathcal{B}$, the architecture search space $\mathcal{X}$.
Output: the best observed architecture configuration.

initialize observations $D = \varnothing$.
while budget $\mathcal{B}$ does not exhaust **do**
  build surrogate M based on observations D.
  call *Sample* for the next configuration to evaluate.
  evaluate the selected configuration x_j and obtain its performance y_j.
  augment $D = D \cup (x_j, y_j)$.
end while
return the configuration with the best observed performance.

实验结果

- OpenBox
- 64 ‘AMD EPYC 7702P’ CPU cores and two ‘RTX 2080Ti’ GPUs.

Results



(a) NAS-Bench-201 CIFAR-10
(b) NAS-Bench-201 CIFAR-100
(c) NAS-Bench-201 ImageNet16-120
(d) NAS-Bench-101
(e) NAS-Bench-ASR

Figure 3: Test results during neural architecture search on three benchmarks. The black dash lines refer to the global optima.

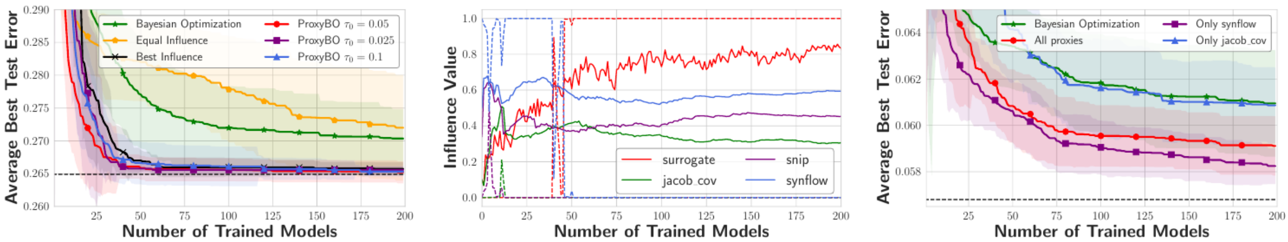
Table 2: Mean ± std. test errors (%) on NAS-Bench-101 and NAS-Bench-201, and test PERs (%) on NAS-Bench-ASR. “NB2 refers to NAS-Bench-201, and “Optimal” refers to the best result in the entire benchmark. The evaluation number of weigh sharing methods and zero-cost proxies is computed by their runtime divided by the average training time of architecture: The zero-cost proxy-based methods apply all three proxies. The results of weight-sharing methods on NAS-Bench-201 an NAS-Bench-101 follow Dong and Yang (2019) and Yu et al. (2019), respectively.

Method	Runtime (#Eval)	NB2-CIFAR-10	NB2-CIFAR-100	NB2-ImageNet16-120	NAS-Bench-101	NAS-Bench-ASR
Regular Methods						
RS	200	9.11 ± 0.21	27.97 ± 0.66	54.01 ± 0.55	6.29 ± 0.12	21.61 ± 0.10
RL	200	9.02 ± 0.24	27.66 ± 0.65	53.58 ± 0.45	6.26 ± 0.14	21.62 ± 0.09
REA	200	8.62 ± 0.21	26.67 ± 0.35	53.08 ± 0.36	6.14 ± 0.24	21.50 ± 0.07
BO	200	8.80 ± 0.22	27.03 ± 0.45	53.12 ± 0.37	6.09 ± 0.23	21.47 ± 0.06
BRP	200	8.58 ± 0.13	26.57 ± 0.12	52.96 ± 0.29	6.00 ± 0.16	21.50 ± 0.08
Weight-sharing Methods						
DARTS	≈9	45.70 ± 0.00	85.38 ± 0.00	83.68 ± 0.00	7.79 ± 0.61	23.59 ± 0.43
ENAS	≈7	46.11 ± 0.58	86.04 ± 2.33	85.19 ± 2.10	8.17 ± 0.42	24.45 ± 0.90
Zero-cost Proxies						
Snip	<1	13.45 ± 1.80	36.41 ± 3.36	71.94 ± 9.09	10.68 ± 2.16	31.61 ± 18.17
Jacob_cov	<1	12.19 ± 1.60	32.99 ± 2.84	60.43 ± 4.46	13.86 ± 1.86	69.95 ± 24.67
Synflow	<1	10.30 ± 0.94	29.55 ± 1.77	56.94 ± 3.57	8.32 ± 1.64	25.70 ± 12.91
Zero-cost Proxy-based Methods						
Warm-up BRP	200	8.58 ± 0.21	26.65 ± 0.31	53.02 ± 0.35	5.99 ± 0.15	21.51 ± 0.10
ProxyBO	200	$\mathbf{8.56 \pm 0.10}$	$\mathbf{26.52 \pm 0.17}$	$\mathbf{52.82 \pm 0.19}$	$\mathbf{5.91 \pm 0.13}$	$\mathbf{21.43 \pm 0.03}$
Optimal	/	8.55	26.49	52.69	5.68	21.40

Table 3: Number of evaluations required to achieve the same average results as REA with 200 evaluations.

	BRP	Warm-up BRP	ProxyBO
NB2 CIFAR-10	170	178	**75**
NB2 CIFAR-100	142	184	**37**
NB2 ImageNet16-120	144	168	**46**
NAS-Bench-101	94	105	**41**
NAS-Bench-ASR	179	213	**51**

消融实验



(a) Sensitivity Analysis (b) Influence Update (c) Effectiveness Analysis

Figure 2: Algorithm analysis of ProxyBO. Figure (a) is conducted on NAS-Bench-201 CIFAR-100 while Figures (b) and (c) are conducted on NAS-Bench-101. The solid lines and dash lines in Figure (b) refer to the generalization ability measurements and influence values, respectively.

- 图a，对温度 τ 进行消融实验：初始温度0.05最好
- 图b，虚线是权重、实线是泛化能力
  - snip 与 jacob cov的秩相关系数是负数，观察到的权重也是变成0
  - synflow在开始的36个evaluation权重最大
  - BO surrogate权重从39个evaluation开始超过synflow，到后面保持1，ProxyBO变成BO

- 图c，选不同proxy的影响（如果**挑选**helpful proxy需要先验知识，比如要知道在该task上的kendall-tau）
    - 当只引入helpless proxy的时候ProxyBO不会比BO差
    - 当引入helpful proxy的时候显著提升