Supplementary Material for Neural Architecture Search with Representation Mutual Information

1. The proof of Theorem 1

Theorem 1. *Assuming that $P(\boldsymbol{\alpha})$ obeys the uniform distribution on the domain of definition for an arbitrary black box function $f(\boldsymbol{\alpha})$. For a specific threshold τ, it holds that*

$$\arg\max_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) = \arg\max_{\boldsymbol{\alpha}} P(\boldsymbol{\alpha} | f(\boldsymbol{\alpha}) + \sigma\epsilon > \tau), \quad (1)$$

where $\sigma > 0,\ \epsilon \sim \mathcal{N}(0,1)$.

Proof. Let y denote the observation with additive Gaussian noise, *i.e.*,, $y = f(\boldsymbol{\alpha}) + \sigma\epsilon$, where $\sigma > 0, \epsilon \sim \mathcal{N}(0,1)$. Then

$$f(\boldsymbol{\alpha}) = E[y|\boldsymbol{\alpha}] = \int_{-\infty}^{\infty} y \cdot p(y|\boldsymbol{\alpha}) \mathrm{d}y.$$

We first define $I(\boldsymbol{\alpha}) := \max(y - \tau, 0)$, so we can derive the Expected Improvement (EI) [14, 20] in the form of goal maximization as follows:

$$\begin{aligned}
EI(\boldsymbol{\alpha}) &= E_{y\sim\mathcal{N}(f(\mathbf{x}),\sigma^2)}[I(\mathbf{x})] = E_{\epsilon\sim\mathcal{N}(0,1)}[I(\mathbf{x})] \\
EI(\boldsymbol{\alpha}) &= \int_{-\infty}^{\infty} I(x)\phi(\epsilon)d\epsilon \\
EI(\boldsymbol{\alpha}) &= \int_{(\tau-f)/\sigma}^{\infty} (f - \tau + \sigma\epsilon)\phi(\epsilon)d\epsilon \\
EI(\boldsymbol{\alpha}) &= (f-\tau)\Phi\left(\frac{f-\tau}{\sigma}\right) + \sigma\int_{(\tau-f)/\sigma}^{\infty} \epsilon\phi(\epsilon)d\epsilon \\
EI(\boldsymbol{\alpha}) &= (f-\tau)\Phi\left(\frac{f-\tau}{\sigma}\right) - \frac{\sigma}{\sqrt{2\pi}}\int_{(\tau-f)/\sigma}^{\infty} (-\epsilon)e^{-\epsilon^2/2}d\epsilon \\
EI(\boldsymbol{\alpha}) &= (f-\tau)\Phi\left(\frac{f-\tau}{\sigma}\right) - \frac{\sigma}{\sqrt{2\pi}}e^{-\epsilon^2/2}\Big|_{(\tau-f)/\sigma}^{\infty} \\
EI(\boldsymbol{\alpha}) &= (f-\tau)\Phi\left(\frac{f-\tau}{\sigma}\right) - \sigma\left(0 - \phi\left(\frac{\tau-f}{\sigma}\right)\right) \\
EI(\boldsymbol{\alpha}) &= (f-\tau)\Phi\left(\frac{f-\tau}{\sigma}\right) + \sigma\phi\left(\frac{f-\tau}{\sigma}\right),
\end{aligned}$$

where Φ and ϕ are the CDF and PDF of the standard normal distribution, respectively. Since σ is a constant, we can obtain the monotonic [9] increase of EI with respect to f, by calculating the derivative:

$$\begin{aligned}
\frac{\mathrm{d}EI}{\mathrm{d}f} &= \Phi\left(\frac{f-\tau}{\sigma}\right) + \sigma^{-1}(f-\tau)\phi\left(\frac{f-\tau}{\sigma}\right) + \phi'\left(\frac{f-\tau}{\sigma}\right) \\
&= \Phi\left(\frac{f-\tau}{\sigma}\right) > 0.
\end{aligned}$$

With this monotonicity, we have

$$\arg\max_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) = \arg\max_{\boldsymbol{\alpha}} EI(\boldsymbol{\alpha}). \quad (2)$$

And then, we use some conclusions from Louis et al. [22]. For completeness, we reproduce the derivations.

$$\begin{aligned}
EI(\boldsymbol{\alpha}) &= \int_{-\infty}^{\infty} max(y-\tau, 0) P(y \mid \boldsymbol{\alpha})\, \mathrm{d}y \\
&= \int_{\tau}^{\infty} (y-\tau) P(y \mid \boldsymbol{\alpha})\, \mathrm{d}y + \int_{-\infty}^{\tau} 0 \cdot P(y \mid \boldsymbol{\alpha})\, \mathrm{d}y \\
&= \frac{1}{P(\boldsymbol{\alpha})}\int_{\tau}^{\infty} (y-\tau) P(\boldsymbol{\alpha} \mid y)\, P(y)\, \mathrm{d}y.
\end{aligned} \quad (3)$$

We use Bayes' rule for the denominator part:

$$\begin{aligned}
P(\boldsymbol{\alpha}) &= \int_{-\infty}^{\infty} P(\boldsymbol{\alpha} \mid y)\, P(y)\, \mathrm{d}y \\
&= \ell(\boldsymbol{\alpha}) \int_{-\infty}^{\tau} P(y)\, \mathrm{d}y + g(\boldsymbol{\alpha}) \int_{\tau}^{\infty} P(y)\, \mathrm{d}y \\
&= \gamma g(\boldsymbol{\alpha}) + (1-\gamma)\ell(\boldsymbol{\alpha}),
\end{aligned} \quad (4)$$

where $\gamma = \Phi(\tau) := p(y > \tau), \ell(\mathbf{x}) := P(\mathbf{x}|y \le \tau)$, and $g(\mathbf{x}) := P(\mathbf{x}|y > \tau)$. While for the molecular part:

$$\begin{aligned}
&\int_{\tau}^{\infty} (y-\tau) P(\boldsymbol{\alpha} \mid y)\, p(y)\, \mathrm{d}y \\
&\quad = g(\boldsymbol{\alpha}) \int_{\tau}^{\infty} (y-\tau) P(y)\, \mathrm{d}y \\
&\quad = g(\boldsymbol{\alpha}) \int_{\tau}^{\infty} y p(y)\, \mathrm{d}y - g(\boldsymbol{\alpha})\tau \int_{\tau}^{\infty} P(y)\, \mathrm{d}y \\
&\quad = g(\boldsymbol{\alpha}) \int_{\tau}^{\infty} y P(y)\, \mathrm{d}y - \gamma\tau g(\boldsymbol{\alpha}) \\
&\quad = K \cdot g(\boldsymbol{\alpha}),
\end{aligned} \quad (5)$$

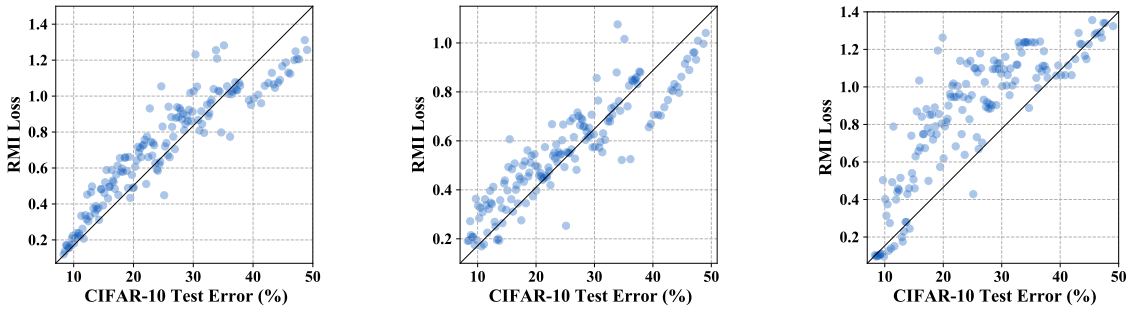

Figure 1. CIFAR-10 [15] test errors of different architectures with their corresponding RMI losses to the teacher network. (a) Selecting a well-performing architecture in NAS-Bench-201 [10] as the teacher network. (b) Selecting ResNet-20 [12] as the teacher network. (c) Selecting ResNet-20 as the teacher network while training for only 20 epochs.

Method	Search Cost (GPU-days)	ImageNet Test Err. (%) top1	ImageNet Test Err. (%) top5	Search Method
MobileNet [13]	-	29.4	10.5	Manual
ShuffleNet 2x (v1) [25]	-	26.4	10.2	Manual
ShuffleNet 2x (v2) [19]	-	25.1	-	Manual
AmoebaNet-C [21]	3150	24.3	7.6	Evolution
NASNet-A [27]	1800	26	8.4	RL
PNAS [17]	225	25.8	8.1	SMBO
BayesNAS [26]	0.2	26.5	8.9	Gradient
ProxylessNAS (ImageNet) [1]	8.3	24.9	7.5	Gradient
PC-DARTS [24]	0.1	25.1	7.8	Gradient
PC-DARTS (ImageNet) [24]	3.8	24.2	7.3	Gradient
GDAS [11]	0.21	26.0	8.5	Gradient
DARTS (2nd) [18]	4.0	26.7	8.7	Gradient
SNAS (mild) [23]	1.5	27.3	9.2	Gradient
P-DARTS [5]	0.3	24.4	7.4	Gradient
P-DARTS (CIFAR100) [5]	0.3	24.7	7.5	Gradient
PARSEC [2]	1	26.0	8.4	Gradient
DARTS- (ImageNet) [6]	4.5	23.8	7.0	Gradient
SDARTS-ADV [4]	1.3	25.2	7.8	Gradient
SGAS [16]	0.25	24.2	7.2	Gradient
TE-NAS [3]	0.05	26.2	8.3	Gradient
TE-NAS (ImageNet) [3]	0.17	24.5	7.5	Gradient
FairDARTS-B [7]	0.4	24.9	7.5	Gradient
Ours	**0.08**	**24.7**	**7.6**	**Random Forest**

Table 1. Classification accuracy and average search cost for RMI-NAS and other NAS algorithms on DARTS [18] search space and ImageNet [8] dataset.

where $K = \int_{\tau}^{\infty} yp\left(y\right) \mathrm{d}y - \gamma\tau$. Combining Eq. (3), (4) and (5):

$$EI(\boldsymbol{\alpha}) \propto \frac{g(\boldsymbol{\alpha})}{\gamma g(\boldsymbol{\alpha}) + (1-\gamma)\ell(\boldsymbol{\alpha})}, \tag{6}$$

then by the definition of ℓ, g and γ, we simplify the above formula as follows:

$$\begin{aligned} &\frac{g(\boldsymbol{\alpha})}{\gamma g(\boldsymbol{\alpha}) + (1-\gamma)\ell(\boldsymbol{\alpha})} \\ &= \frac{P(\boldsymbol{\alpha} \mid y > \tau)}{\gamma \cdot P(\boldsymbol{\alpha} \mid y > \tau) + (1-\gamma) \cdot P(\boldsymbol{\alpha} \mid y \leq \tau)} \\ &= \frac{\frac{P(y>\tau|\boldsymbol{\alpha})P(\boldsymbol{\alpha})}{P(y>\tau)}}{\gamma \cdot \frac{P(y>\tau|\boldsymbol{\alpha})P(\boldsymbol{\alpha})}{P(y>\tau)} + (1-\gamma) \cdot \frac{P(y\leq\tau|\boldsymbol{\alpha})p(\boldsymbol{\alpha})}{P(y\leq\tau)}} \\ &= \frac{P(y > \tau \mid \boldsymbol{\alpha})}{\gamma}. \end{aligned} \tag{7}$$

Note that $P(\boldsymbol{\alpha})$ follows the uniform distribution, and using Bayes' theorem, we have

$$P(y > \tau \mid \boldsymbol{\alpha}) = \frac{P(y > \tau) \cdot P(\boldsymbol{\alpha} | y > \tau)}{P(\boldsymbol{\alpha})} \propto P(\boldsymbol{\alpha} | y > \tau). \tag{8}$$

Finally combining Eq. (2), (6), (7) and (8), we obtain

$$\begin{aligned} \arg\max_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) &= \arg\max_{\boldsymbol{\alpha}} EI(\boldsymbol{\alpha}) \\ &= \arg\max_{\boldsymbol{\alpha}} \frac{g(\boldsymbol{\alpha})}{\gamma g(\boldsymbol{\alpha}) + (1-\gamma)\ell(\boldsymbol{\alpha})} \\ &= \arg\max_{\boldsymbol{\alpha}} \frac{P(y > \tau \mid \boldsymbol{\alpha})}{\gamma} \\ &= \arg\max_{\boldsymbol{\alpha}} P(\boldsymbol{\alpha} | y > \tau). \end{aligned} \tag{9}$$

□

2. Results on ImageNet

ImageNet [8] is a widely-used dataset in Computer vision tasks. In this work, we follow the prior art methods' settings to perform our results, comparing performance under the mobile setting of ImageNet. For purpose both on verifying our searched architecture's capability and as a general approach, we transfer our searched architecture in CIFAR-10 [15] to Imagenet. From the results in Tab. 1, RMI-NAS shows comparable test accuracy but marginal efficiency improvement among all methods.

Notably, the architecture we search for has the same normal cells and reduction cells, but still has performance comparable to other methods. This can be further improved as this setting restricts the architecture performance, and we will search for different cells in the next version to gain better accuracy.

3. Robustness of RMI

In this section, we compare and analyze some experiments that are used to verify the usability and robustness of the proposed RMI approach.

We first present an overview of all available architectures in the NAS-Bench-201 [10] search space using RMI in Fig. 2. Generally, it demonstrates positive correlation between RMI and accuracy over the full space. Since we aim to find an optimal architecture, we need to focus more on architectures with better performance. Our method shows higher correlation in best-performing architectures, distributing more densely at the top 5%.

Furthermore, considering that there may be significant differences between good and bad architectures, the sampling method should reflect as much variability as possible. Due to the low percentage of well-performing architectures in the whole space, uniform sampling can better distinguish between networks with different performance.

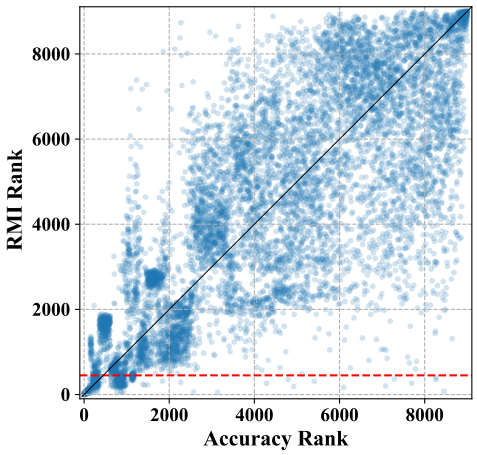

Figure 2. Correlation between RMI loss ranking and accuracy ranking of architectures in NAS-Bench-201 [10]. The red line refers to the top 5% RMI score, where our method shows high correlation in best-performing architectures and distributes more densely.

In practice, we uniformly sample architectures by accuracy from the NAS-Bench-201 search space, and choose different well-performing ones as the teacher network to calculate RMI. As shown in Fig. 1(a), our method owns high correlation when selecting a well-performing architecture in the same search space. The same results are obtained when using ResNet-20 [12] as the teacher model, referring to Fig. 1(b). This highlights the robustness of proposed RMI method when applied to different teacher networks.

We further test our method when sampled architectures are trained for only 20 epochs, and the observation in Fig. 1(c) demonstrates that even when not fully trained, our method still shows great correlation of architectures' accuracy and their corresponding RMI loss. This result further highlights that when with an accurate indicator, it is possible to distinguish between architectures with different performance by training for a small number of epochs.

References

[1] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *ICLR*, 2019. 2

[2] Francesco Paolo Casale, Jonathan Gordon, and Nicolo Fusi. Probabilistic neural architecture search. *arXiv*, 2019. 2

[3] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. In *ICLR*, 2020. 2

[4] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *ICML*, pages 1554–1565. PMLR, 2020. 2

[5] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. *arXiv*, 2019. 2

[6] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. Darts-: robustly stepping out of performance collapse without indicators. *arXiv*, 2020. 2

[7] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair darts: Eliminating unfair advantages in differentiable architecture search. In *ECCV*, pages 465–480. Springer, 2020. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 2, 3

[9] R Jones Donald. Efficient global optimization of expensive black-box function. *J. Global Optim.*, 13:455–492, 1998. 1

[10] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In *International Conference on Learning Representations*, 2019. 2, 3

[11] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *CVPR*, 2019. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 3

[13] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv*, 2017. 2

[14] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998. 1

[15] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009. 2, 3

[16] Guohao Li, Guocheng Qian, Itzel C Delgadillo, Matthias Muller, Ali Thabet, and Bernard Ghanem. Sgas: Sequential greedy architecture search. In *CVPR*, pages 1620–1630, 2020. 2

[17] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *ECCV*, 2018. 2

[18] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *ICLR*, 2019. 2

[19] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018. 2

[20] Chao Qin, Diego Klabjan, and Daniel Russo. Improving the expected improvement algorithm. *arXiv*, 2017. 1

[21] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. *arXiv*, 2018. 2

[22] Louis C Tiao, Aaron Klein, Matthias Seeger, Edwin V Bonilla, Cedric Archambeau, and Fabio Ramos. Bore: Bayesian optimization by density-ratio estimation. *arXiv*, 2021. 1

[23] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. *arXiv*, 2018. 2

[24] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient differentiable architecture search. *arXiv*, 2019. 2

[25] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018. 2

[26] Hongpeng Zhou, Minghao Yang, Jun Wang, and Wei Pan. Bayesnas: A bayesian approach for neural architecture search. In *ICML*, pages 7603–7613. PMLR, 2019. 2

[27] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018. 2