A Proof

Theorem 1: Suppose that DARTS obtains the optimized architecture parameter α with supernet weights θ^* after supernet training, α changes to $\hat{\alpha}$ when conducting architecture discretization, and the train-from-scratch validation loss for $\hat{\alpha}$ is $\mathcal{L}(\hat{\theta}^*, \hat{\alpha})$. If the third-derivative of the loss function $\mathcal{L}$ at optimum is zero or sufficiently small [4], and with $\frac{\partial \mathcal{L}(\hat{\theta}^*, \hat{\alpha})}{\partial \theta} = 0$, we have

$$\begin{aligned} \Delta\mathcal{L} &= \mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) \approx \mathcal{L}(\theta^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) \\ &\quad - 1/2 * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}. \end{aligned} \tag{9}$$

Proof of Theorem 1: Based on Eq.(4), we now only need to calculate $\Delta\theta$ to get $\Delta\mathcal{L}$ after we change the α to $\hat{\alpha}$ as other parts could be directly obtained, while we try to avoid the time-consuming retraining process through leveraging implicit function to estimating $\hat{\theta}^*$. First, based on the local optimal, we have

$$\frac{\partial \mathcal{L}(\hat{\theta}^*, \hat{\alpha})}{\partial \theta} = 0. \tag{10}$$

When we consider the second-order Taylor expansion on θ, we have

$$\frac{\partial(\mathcal{L}(\theta^*, \hat{\alpha}) + (\frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta})^T * \Delta\theta + \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta} * \Delta\theta)}{\partial \theta} \approx 0. \tag{11}$$

$$\frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} + \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta} * \Delta\theta \approx 0. \tag{12}$$

Eq. (12) is obtained since we assume the third-derivative of the loss function $\mathcal{L}$ at optimum is zero or sufficiently small [4]. And we have

$$\Delta\theta \approx -\frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}. \tag{13}$$

So,

$$\Delta\theta^T \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} \approx -\frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} \tag{14}$$

$$\Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta} * \Delta\theta \approx \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} \tag{15}$$

In this way, based on Eq.(4), (14) and (15), we have

$$\begin{aligned} \Delta\mathcal{L} &= \mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) \\ &\approx \mathcal{L}(\theta^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) - 1/2 * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} \end{aligned} \tag{16}$$

In this way, we can approximate every change in the α.

□

Corollary 1: Based on the Assumption 1-3, we could bound the error between the approximated validation loss $\mathcal{L}(\hat{\theta}^*, \hat{\alpha})$ and the ground-truth $\tilde{\mathcal{L}}_2(\hat{\theta}^*, \hat{\alpha})$ in DARTS with $E = \left\| \mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \tilde{\mathcal{L}}(\hat{\theta}^*, \hat{\alpha}) \right\| \leqslant \frac{K^3}{6} \max \left| \frac{\partial \mathcal{L}^3}{\partial \theta^3} \right|$, where $K = \frac{C_L}{\lambda} * \|\Delta\alpha\| + \frac{C_H * C_a^2}{2 * \sigma_{min}^2 * \lambda} * \|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^4)$.

Proof of Corollary 1: Before our analysis on the error bound of approximation in Theorem 1, we first restate the following common assumptions in the bi-level optimization [9, 15–17, 51].

Assumption 1: For any θ and α, $\mathcal{L}(\cdot, \alpha)$ and $\mathcal{L}(\theta, \cdot)$ are Lipschitz continuous with constant $C_f > 0$ and constant $C_L > 0$, respectively.

Assumption 2: $\mathcal{L}(\theta, \alpha)$ is twice differentiable with constant C_H and is λ-strongly convex with θ around $\theta^*(\alpha)$.

Assumption 3: $\left\|\nabla^2_{\theta\alpha}\mathcal{L}\right\|$ is bounded with constant $C_a > 0$.

Now we can estimate the error bound produced by Theorem 1. First, the error is due to the second-order Taylor expansion on θ in Eq. (4). We have the error

$$E = \left\|\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \tilde{\mathcal{L}}(\hat{\theta}^*, \hat{\alpha})\right\| \leq \frac{|\Delta\theta|^3}{6}\max\left|\frac{\partial\mathcal{L}^3}{\partial\theta^3}\right|, \tag{17}$$

where we should notice that the approximated validation loss $\mathcal{L}(\hat{\theta}^*, \hat{\alpha})$ is Taylor expansion of $\hat{\mathcal{L}}$ in the point $\hat{\theta}^* - \Delta\theta$. Since we set that $\Delta\theta = -\frac{\partial^2\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}^{-1} * \frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}$ in Eq. 13, this error E could be easily bound. However, we should notice that our $\Delta\theta = -\frac{\partial^2\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}^{-1} * \frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}$ is an approximation, that $\theta^* + \Delta\theta \neq \hat{\theta}^*$, and we define that $\theta^* + \Delta\theta = \hat{\theta}^*_e$ and $\Delta\theta_e = \hat{\theta}^* - \hat{\theta}^*_e$.

First, we have

$$\|\Delta\theta\| = \left\|\theta^* - \hat{\theta}^* + \hat{\theta}^* - \hat{\theta}^*_e\right\| \leq \left\|\theta^* - \hat{\theta}^*\right\| + \left\|\hat{\theta}^* - \hat{\theta}^*_e\right\| \tag{18}$$

For the first part in Eq. (18), based on Assumption 2 that $\mathcal{L}(\theta, \alpha)$ is λ-strongly convex with θ around θ^*, we have:

$$\left\|\nabla\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \nabla\mathcal{L}(\theta^*, \hat{\alpha})\right\| \geq \lambda * \left\|\hat{\theta}^* - \theta^*\right\|, \tag{19}$$

and

$$\left\|\theta^* - \hat{\theta}^*\right\| \leq \frac{1}{\lambda} * \|\nabla\mathcal{L}(\theta^*, \hat{\alpha}^*)\|. \tag{20}$$

When we consider that $\mathcal{L}(\theta^*, \alpha)$ is C_L Lipschitz continuous with α, we have

$$\|\nabla\mathcal{L}(\theta^*, \hat{\alpha}^*) - \nabla\mathcal{L}(\theta^*, \alpha^*)\| \leq C_L * \|\Delta\alpha\|. \tag{21}$$

Baesd on Eq. (20) and (21), we have

$$\left\|\theta^* - \hat{\theta}^*\right\| \leq \frac{C_L}{\lambda} * \|\Delta\alpha\| \tag{22}$$

As to the second part, when we assume that $\mathcal{L}(\theta, \alpha)$ is λ-convex near the local optimal, we also have

$$\left\|\nabla\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \nabla\mathcal{L}(\hat{\theta}^*_e, \hat{\alpha})\right\| \geq \lambda * \left\|\hat{\theta}^* - \hat{\theta}^*_e\right\| \tag{23}$$

where $\hat{\theta}^*$ is the true optimal, and $\hat{\theta}^*_e$ is the approximate local optimal when we utilize the implicit function theorem to approximate $\Delta\theta$ in Eq. 13.

so that, we have

$$\|\Delta\theta_e\| = \left\|\hat{\theta}^* - \hat{\theta}^*_e\right\| \leq \frac{1}{\lambda} * \left\|\nabla\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \nabla\mathcal{L}(\hat{\theta}^*_e, \hat{\alpha})\right\| \tag{24}$$

Consider the local optimal that $\nabla\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) = 0$, we have

$$\|\Delta\theta_e\| \leq \frac{1}{\lambda} * \left\|\nabla\mathcal{L}(\hat{\theta}^*_e, \hat{\alpha})\right\| \tag{25}$$

Now, we need to bound $\left\|\nabla\mathcal{L}(\hat{\theta}^*_e, \hat{\alpha})\right\|$.

$$\left\|\nabla\mathcal{L}(\hat{\theta}^*_e, \hat{\alpha})\right\| = \|\nabla\mathcal{L}(\theta^* + \Delta\theta, \hat{\alpha})\| \tag{26}$$

Since we utilize Eq. (13) that $\Delta\theta = -\frac{\partial^2\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}^{-1} * \frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}$ to conduct the second-order Taylor expansion, we have

$$-\frac{\partial^2\mathcal{L}(\theta^*, \hat{\alpha})}{\partial\theta\partial\theta} * \Delta\theta_{=}\frac{\partial\mathcal{L}(\theta^*, \hat{\alpha})}{\partial\theta} \tag{27}$$

So

$$\begin{aligned}
&\left\|\nabla\mathcal{L}(\hat{\theta}_e^*,\hat{\alpha})\right\| = \left\|\nabla_\theta\mathcal{L}(\theta^*+\Delta\theta,\hat{\alpha}) - \nabla_\theta\mathcal{L}(\theta^*,\hat{\alpha}) - \nabla_\theta^2\mathcal{L}(\theta^*,\hat{\alpha})*\Delta\theta\right\| \\
&= \left\|\int_0^1 \left(\nabla_\theta^2\mathcal{L}(\theta^*+t*\Delta\theta,\hat{\alpha}) - \nabla_\theta^2\mathcal{L}(\theta^*,\hat{\alpha})\right)*\Delta\theta\ dt\right\| \\
&= \left\|\Delta\theta * \int_0^1 \left(\nabla_\theta^2\mathcal{L}(\theta^*+t*\Delta\theta,\hat{\alpha}) - \nabla_\theta^2\mathcal{L}(\theta^*,\hat{\alpha})\right)\ dt\right\| \\
&= \|\Delta\theta\| * \left\|\int_0^1 \left(\nabla_\theta^2\mathcal{L}(\theta^*+t*\Delta\theta,\hat{\alpha}) - \nabla_\theta^2\mathcal{L}(\theta^*,\hat{\alpha})\right)\ dt\right\| \\
&\leq \|\Delta\theta\| * \left\|\int_0^1 C_H * t * \Delta\theta\ dt\right\| = \frac{C_H}{2}\|\Delta\theta\|^2 \\
&= \frac{C_H}{2}\left\|\frac{\partial^2\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}^{-1} * \frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}\right\|^2 \\
&\leq \frac{C_H}{2*\sigma_{min}^2}\left\|\frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}\right\|^2 \\
&\leq \frac{C_H * C_l^2}{2*\sigma_{min}^2}
\end{aligned} \tag{28}$$

where the second row is calculated since $\int \nabla_\theta\mathcal{L}^2(\theta^*+t*\Delta\theta,\hat{\alpha})*\Delta\theta\ dt = \nabla_\theta\mathcal{L}(\theta^*+t*\Delta\theta,\hat{\alpha})$. The fifth row is calculated as we assume that $\mathcal{L}$ is convex and twice-differentiable with C_H. And we assume that the $\frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}$ is bounded by C_l. σ_{min} is the smallest eigenvalue of Hessian matrix $\frac{\partial^2\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}$.

In addition, when we consider a more tight bound, and consider a Taylor expansion on α, we have

$$\begin{aligned}
\left\|\frac{\partial\mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}\right\|^2 &= \left\|\frac{\partial(\mathcal{L}(\theta^*,\alpha) + \Delta\alpha * \frac{\partial\mathcal{L}(\theta^*,\alpha)}{\partial\alpha} + o(\Delta\alpha))}{\partial\theta}\right\|^2 \\
&\leq \left\|\Delta\alpha\frac{\partial^2\mathcal{L}(\theta^*,\alpha)}{\partial\alpha\partial\theta}\right\|^2 + o(\|\Delta\alpha\|^2) \\
&\leq C_a^2\|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^2),
\end{aligned} \tag{29}$$

where the second row is obtained based on the local optimal $\frac{\partial\mathcal{L}(\theta^*,\alpha)}{\partial\theta} = 0$. So, we have

$$\|\nabla\mathcal{L}(\theta_e^*,\hat{\alpha})\| \leq \frac{C_H * C_a^2}{2*\sigma_{min}^2} * \|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^2), \tag{30}$$

and

$$\|\Delta\theta_e\| \leq \frac{C_H * C_a^2}{2*\sigma_{min}^2*\lambda} * \|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^2). \tag{31}$$

Based on Eq. (18), (22), and (31), we have

$$\|\Delta\theta\| \leq \frac{C_L}{\lambda} * \|\Delta\alpha\| + \frac{C_H * C_a^2}{2*\sigma_{min}^2*\lambda} * \|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^2) \tag{32}$$

The final error bound can be calculated as:

$$E \leq \frac{K^3}{6}\max\left|\frac{\partial\mathcal{L}^3}{\partial\theta^3}\right| \tag{33}$$

where $K = \frac{C_L}{\lambda} * \|\Delta\alpha\| + \frac{C_H * C_a^2}{2*\sigma_{min}^2*\lambda} * \|\Delta\alpha\|^2 + o(\|\Delta\alpha\|^2)$.

□

Derivation of Eq. (6)**:** In Section 4, when we consider a first-order Taylor expansion on α as we only apply an infinitesimal change of α, and we alsoconsider a second-order Taylor expansion on θ, we have

$$
\begin{aligned}
\Delta\mathcal{L} =& \mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) \approx \mathcal{L}(\hat{\theta}^*, \alpha) + \Delta\alpha^T \frac{\partial \mathcal{L}(\hat{\theta}^*, \alpha)}{\partial \alpha} - \mathcal{L}(\theta^*, \alpha) \\
&\approx \Delta\theta^T * \frac{\partial \mathcal{L}(\theta^*, \alpha)}{\partial \theta} + 1/2 \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} \Delta\theta + \Delta\alpha^T * \frac{\partial \mathcal{L}(\hat{\theta}^*, \alpha)}{\partial \alpha} \\
&= 1/2 \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} \Delta\theta + \Delta\alpha^T * \frac{\partial \mathcal{L}(\hat{\theta}^*, \alpha)}{\partial \alpha},
\end{aligned} \tag{34}
$$

where the last row is obtained is obtained due to the local optimal $\frac{\partial \mathcal{L}(\theta^*, \alpha)}{\partial \theta} = 0$. $\square$

Theorem 2: Supposed that DARTS obtains the optimized architecture parameter α with supernet weights θ^* after supernet training, and we pose an infinitesimal change on α. Based on implicit function theorem and under the assumption that the third-derivative of the loss function at optimum is zero or sufficiently small [4], the change of validation performance can be estimated as:

$$
\Delta\mathcal{L} = \mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) \approx -1/2 \Delta\alpha^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \alpha \partial \theta} * H^{-1} * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \alpha} * \Delta\alpha, \tag{35}
$$

where $H = \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta}$ is the Hessian matrix.

Proof of Theorem 2: Although Section 3 provides an iterative solution to measure the operation importance based Eq. (5) through individually removing each operation, we need to calculate n times (the number of all candidate operations in the supernet work) to estimate the importance of all candidate operations. More important, since removing one operation poses a considerable change on α ($\alpha_i \to 0$), the approximation error produced by Eq.(5) is non-negligible which may affects the accuracy as stated by Corollary 1. So, a more practical solution to illustrate the importance of each operation is to estimate how the validation performance will change after posing an infinitesimal change on α, a.k.a. operation sensitivity, and we can directly get the operation importance with one calculation.

When we consider that DARTS obtains the optimized architecture parameter α with supernet weights θ^* after supernet training, and we pose an infinitesimal change on α, we can conduct a first-order Taylor expansion on α a second-order Taylor expansion on θ. So that we have get the Eq. (34) as stated in the above. When we further conduct a second Taylor expansion on $\hat{\theta}^*$ for $\frac{\partial \mathcal{L}(\hat{\theta}^*, \alpha)}{\partial \alpha}$, we have

$$
\begin{aligned}
\Delta\mathcal{L} &\approx 1/2 \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} \Delta\theta + \Delta\alpha^T * \frac{\partial \mathcal{L}(\hat{\theta}^*, \alpha)}{\partial \alpha} \\
&\approx 1/2 \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} \Delta\theta + \Delta\alpha^T * (\frac{\partial (\mathcal{L}(\theta^*, \alpha) + \Delta\theta^T * \frac{\partial \mathcal{L}(\theta^*, \alpha)}{\partial \theta} + \Delta\theta * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} * \Delta\theta^T)}{\partial \alpha}) \\
&\approx 1/2 \Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \theta \partial \theta} \Delta\theta + \Delta\alpha^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \alpha \partial \theta} * \Delta\theta,
\end{aligned} \tag{36}
$$

where the last row is obtained when we neglect the third and higher derivatives of the loss function with considering the chain rule that $\frac{\partial \mathcal{L}}{\partial \alpha} = \frac{\partial \mathcal{L}}{\partial \theta} * \frac{\partial (\theta^*(\alpha))}{\partial \alpha}$, and assume α is a local optimal for the continuous magnitude optimization after architecture search. From the above, we can find that, the challenge is still that it is intractable to calculate $\Delta\theta$. Similar as before, when we consider implicit function and second-order Taylor expansion on θ, we have

$$
\frac{\partial \mathcal{L}(\hat{\theta}^*, \hat{\alpha})}{\partial \theta} = 0 \tag{37}
$$

$$
\frac{\partial (\mathcal{L}(\theta^*, \alpha) + \Delta\alpha^T * \frac{\partial \mathcal{L}(\theta^*, \alpha)}{\partial \alpha} + \Delta\theta^T * \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta} + \Delta\theta * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta} * \Delta\theta^T)}{\partial \theta} \approx 0 \tag{38}
$$

When we neglect the third and higher derivatives and the local optimal $\frac{\partial \mathcal{L}(\theta^*, \alpha)}{\partial \theta} = 0$, we have

$$
\Delta\theta^T * \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta} \approx -\Delta\alpha^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial \alpha \partial \theta} \tag{39}
$$

Then, similar as before, when we apply a first-order Taylor expansion on $\hat{\alpha}$, with the chain-rule and neglect third and higher derivatives, we have

$$\Delta\theta^T = -\Delta\alpha^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha^*)}{\partial\alpha\theta} * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha^*)}{\partial\theta\partial\theta}^{-1} \tag{40}$$

After applying Eq. (40) on Eq. (36), we have

$$\Delta\mathcal{L} = -1/2\Delta\alpha^T * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha^*)}{\partial\alpha\theta} * H^{-1} * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha^*)}{\partial\theta\partial\alpha} * \Delta\alpha \tag{41}$$

□

B Practical Implementation

To practically calculate the proposed influential magnitude as Definition 1, we need to estimate the Inverse-Hessian-Vector products (IHVPs) $H^{-1}v$, and $v * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial\alpha\partial\theta}$. In the following, we first described two methods to estimate $H^{-1}v$, and the following DARTS to described the practical calculation on $v * \frac{\partial^2 \mathcal{L}(\theta^*, \alpha)}{\partial\alpha\partial\theta}$.

B.1 Neumann series approximation for Lemma 1

The most common method for the IHVPs is the Neumann series that, for $\|I - A\| < 1$, we have:

$$A^{-1} = \sum_{k=0}^{\infty} (I - A)^k. \tag{42}$$

However, it is impossible to guarantee that $\|I - H\| < 1$. Different from [22], we consider $H^{-1} = \gamma(\gamma H)^{-1}$. When assuming H is bounded with σ_{max}, $\left\| \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right\| < \sigma_{max}$. With $\gamma < \frac{1}{\sigma_{max}}$, we have $\left\| I - \gamma \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right\| < 1$ [29, 36]. When we conduct the Neumann series approximation for $\left[\frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right]^{-1}$ in the optimal point, we have:

$$\left[\frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right]^{-1} = \gamma(I - I + \gamma \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta})^{-1} = \gamma \sum_{j=0}^{\infty} \left[I - \gamma \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right]^j. \tag{43}$$

$$H^{-1} = \gamma \sum_{k=0}^{\infty} \left[I - \gamma \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\theta} \right]^k \approx \gamma \sum_{k=0}^{K} [I - \gamma H]^k, \tag{44}$$

when we only consider the first K terms and H is positive in θ^* (Please note that, generally, we could only guarantee H is positive when in optimal point). Accordingly, the Lemma 1 is derived.

As suggested by [51], a medium K is large enough to approximate IHVPs, where $K = 2$ obtains similar results as $K \geqslant 3$ while with less computational cost. In this paper, we also observed a similar phenomenon and also set $K = 2$ by default in our Neumann series approximation based IHVPs.

B.2 Sherman-Morrison formula for Lemma 2

Apart from the Neumann series approximation, we could also follow the Sherman-Morrison formula to exactly calculate the inverse of Hessian when we consider the empirical Fisher to replace the Fisher matrix. First, we know that the Hessian equals to fisher information matrix in the optimal point.

$$H = \nabla_\theta \nabla_\theta L = \hat{F} - \frac{1}{n}(y_i - f(x_i)) \nabla_\theta \nabla_\theta f \in \mathbb{R}^{p \times p} \tag{45}$$

We name the $r = \frac{1}{n}(y_i - f(x_i)) \nabla_\theta \nabla_\theta f$, which is the residual contribution to Hessian H. At the end of training, if we can achieve zero training error, namely, $y_i = f(x_i)$, the residual will converge to zero. We have

$$H = \hat{F} = \nabla_\theta f (\nabla_\theta f)^T. \tag{46}$$

Please note that the dimension of $\nabla_\theta f$ is $M * m$ where m is the number of classes (output dimension) and M is the number of parameters. Practically, we usually consider an empirical Fisher to approximate the Fisher matrix that

$$F = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \mathcal{L}(y_n, f(x_n)) (\nabla_\theta \mathcal{L}(y_n, f(x_n))^T \tag{47}$$

So, in the remaining text, the F indicates the empirical Fisher for simplicity.

The Sherman-Morrison formula is defined as:

$$(A + \mathrm{uv}^T)^{-1} = A^{-1} - \frac{A^{-1}\mathrm{uv}^T A^{-1}}{1 + \mathrm{v}^T A^{-1}\mathrm{u}} \tag{48}$$

When we define that $F_n = \frac{1}{N}\sum_{n=1}^{n} \nabla_\theta \mathcal{L}(y_n, f(x_n))(\nabla_\theta \mathcal{L}(y_n, f(x_n))^T$, we can notice that the empirical Fisher can be represented as F_N, and we have:

$$F_n = F_{n-1} + \frac{1}{N}\nabla_\theta \mathcal{L}_n \, \nabla_\theta {\mathcal{L}_n}^T \tag{49}$$

where $\mathcal{L} = \ell + \eta\mathcal{R}(\theta)$ that ℓ is a cross-entropy loss and $\mathcal{R}$ is the regularization term, and η is the weight-decay, $F_0 = \eta I$. $\mathcal{L}_n$ is the loss for the *n-th* batch dataset. And based on the Sherman-Morrison formula, we have

$$F_{n+1}^{-1} = F_n^{-1} - \frac{F_n^{-1}\nabla_\theta \mathcal{L}_{n+1}\nabla_\theta {\mathcal{L}_{n+1}}^T F_n^{-1}}{N + \nabla_\theta {\mathcal{L}_{n+1}}^T F_n^{-1}\nabla_\theta \mathcal{L}_{n+1}} \tag{50}$$

Please notice that, whether using the Neumann series or the Sherman-Morrison formula, we always need to calculate the Inverse-Hessian Vector Products (IHVPs) rather than only the Inverse-Hessian. It is easy to implement the IHVPs based on the Neumann series through the Hessian-vector products based on Lemma 1. In the following, we discuss how to implement the IHVPs based on Sherman-Morrison formula. When we assume that we have $n+1$ batches datasets, we have

$$H^{-1}v = F_N^{-1}v = F_n^{-1}v - \frac{F_{N-1}^{-1}\nabla_\theta \mathcal{L}_N \nabla_\theta {\mathcal{L}_N}^T F_{N-1}^{-1}}{N + \nabla_\theta {\mathcal{L}_N}^T F_n^{-1}\nabla_\theta \mathcal{L}_N} v \tag{51}$$

We can find that, to solve the previous equation, we need to first recurrently calculate $\mathrm{r}_N = F_{N-1}^{-1}\nabla_\theta \mathcal{L}_N$.

$$F_n^{-1}v = F_{n-1}^{-1}v - \mathrm{r}_n \frac{{\mathrm{r}_n}^T v}{N + \nabla_\theta {l_n}^T \mathrm{r}_n} = \lambda^{-1}v - \sum_{j=1}^{n} \mathrm{r}_j \frac{{\mathrm{r}_j}^T v}{N + \nabla_\theta {\mathcal{L}_j}^T \mathrm{r}_j} \tag{52}$$

We first need to calculate the $\mathrm{r}_n = F_{n-1}^{-1}\nabla_\theta \mathcal{L}_n$.

$$\mathrm{r}_n = F_{n-1}^{-1}\nabla_\theta \mathcal{L}_n = \lambda^{-1}\nabla_\theta \mathcal{L}_n - \sum_{j=1}^{n-1} \mathrm{r}_j \frac{{\mathrm{r}_j}^T \nabla_\theta \mathcal{L}_n}{N + \nabla_\theta {\mathcal{L}_j}^T \mathrm{r}_j}. \tag{53}$$

So, we can first iterative calculate r_n, and then calculate $F_n^{-1}v$.

B.3 Practical Calculation on Matrix-Vector product

Apart from the IHVPs, we also need to calculate $v_a * \frac{\partial^2 \mathcal{L}}{\partial\alpha\partial\theta}$ and $v_m * \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\alpha}$, which are called as matrix-vector product and computational-expensive. Following DARTS [27], we also utilize the finite difference approximation for the practical calculation.

Let ξ be a small scalar, and consider the function $\frac{\partial \mathcal{L}(\theta, \alpha + \xi * v_a)}{\partial\theta}$ with Taylor expansion on α, we have

$$\begin{aligned} \frac{\partial \mathcal{L}(\theta, \alpha + \xi * v_a)}{\partial\theta} &= \frac{\partial \mathcal{L}(\theta, \alpha)}{\partial\theta} + \xi * v_a \frac{\partial^2 \mathcal{L}(\theta, \alpha)}{\partial\alpha\partial\theta} + ..., \\ \frac{\partial \mathcal{L}(\theta, \alpha - \xi * v_a)}{\partial\theta} &= \frac{\partial \mathcal{L}(\theta, \alpha)}{\partial\theta} - \xi * v_a \frac{\partial^2 \mathcal{L}(\theta, \alpha)}{\partial\alpha\partial\theta} + ..., \end{aligned} \tag{54}$$

when we only consider the first two terms, we have

$$v_a * \frac{\partial^2 \mathcal{L}}{\partial\alpha\partial\theta} = \frac{\frac{\partial \mathcal{L}(\theta, \alpha + \xi * v_a)}{\partial\theta} - \frac{\partial \mathcal{L}(\theta, \alpha - \xi * v_a)}{\partial\theta}}{2\xi}. \tag{55}$$

Similarly, the $v_m * \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\alpha}$ can be approximated as:

$$v_m * \frac{\partial^2 \mathcal{L}}{\partial\theta\partial\alpha} = \frac{\frac{\partial \mathcal{L}(\theta + \xi * v_m, \alpha)}{\partial\alpha} - \frac{\partial \mathcal{L}(\theta - \xi * v_m, \alpha)}{\partial\alpha}}{2\xi}. \tag{56}$$

Table 4: Comparison results with NAS baselines on NAS-Bench-201.

Method	CIFAR-10		CIFAR-100		ImageNet-16-120	
	Valid(%)	Test(%)	Valid(%)	Test(%)	Valid(%)	Test(%)
Random baseline	83.20±13.28	86.61±13.46	60.70±12.55	60.83±12.58	33.34±9.39	33.13±9.66
RandomNAS [26]	80.42±3.58	84.07±3.61	52.12±5.55	52.31±5.77	27.22±3.24	26.28±3.09
ENAS [33]	37.51±3.19	53.89±0.58	13.37±2.35	13.96±2.33	15.06±1.95	14.84±2.10
GDAS [10]	89.88±0.33	93.40±0.49	70.95±0.78	70.33±0.87	41.28±0.46	41.47±0.21
SETN [11]	84.04±0.28	87.64±0.00	58.86±0.06	59.05±0.24	33.06±0.02	32.52±0.21
SNAS [42]	90.10±1.04	92.77±0.84	69.69±2.39	69.35±1.98	42.84±1.79	43.16±2.64
PC-DARTS [43]	89.96±0.15	93.41±0.30	67.12±0.39	67.48±0.89	40.83±0.08	41.31±0.22
DARTS (1st) [27]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
DARTS (2nd) [27]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
DARTS-PT [40]	87.34±0.43	89.63±0.19	62.48±2.89	62.35±2.14	36.35±2.76	36.51±2.13
DARTS-IF	90.13±0.54	91.84±0.84	65.47±1.33	67.94±1.23	42.78±3.57	42.50±3.30
DARTS-IM	**90.92±0.34**	**93.61±0.23**	**71.21±0.55**	**71.31±0.40**	**44.70±0.74**	**44.98±0.36**
optimal	91.61	94.37	74.49	73.51	46.77	47.31

DARTS-IM best single run achieves **94.29**%, **72.67**%, and **45.93**% test accuracy on CIFAR-10, CIFAR-100, and ImageNet, respectively.

C Why perturbation-based method fails

The perturbation-based method, e.g. the leave-one-out retraining [22], is an intuitive way to measure the importance of every candidate operations. More detailed, we can individually remove every candidate operation in each edge, and then fine-tune the supernet to the validation performance drop $\mathcal{L}(\hat{\theta}^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha)$. However, DARTS-PT abandons the retraining part that directly measure the validation performance drop. An concern will be raised here as the pruned supernet is not trained to optimal, and it is inaccurate to measure the performance drop by $\mathcal{L}(\theta^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha)$. More importantly, since the *skip-connection*'s magnitude is generally much higher than other candidate operations, and directly removing *skip-connections* usually greatly deteriorates the supernet performance if we do not fine-tune the supernet. This is also the reason why we empirically find that DARTS-PT still prefers *skip-connection* than other operations. Figure 8 (a) plots the searched architecture in NAS-Bench-201 for DARTS-PT, which contains 3 *skip-connections*. In contrast, the architecture selected by our influential magnitude only contains 1 *skip-connection*. This phenomenon also exists in the experiments on the search space S1 and S2 of [45], where although DARTS-PT can avoid selecting all *skip-connections* in all edges, its searched architectures still contains intensive *skip-connections*.

D Comparison with SOTAs on NAS-Bench-201

We also compared our DARTS-IM with different NAS baselines on the NAS-Bench-201 search space. Table 4 reports the comprehensive validation and test results on all three datasets, including CIFAR-10, CIFAR-100, and ImageNet. As shown, DARTS is not a valid method for differentiable architecture search, which even leads to poorer results than random baseline. Although considering other supernet training methods can also improve the performance, we can also found that, with only replacing the operation selection metric with our influential magnitude, our DARTS-IM can achieve more competitive results. Moreover, our DARTS-IM achieves near-optimal results, a **94.29**%, **72.67**%, and **45.93**% test accuracy on CIFAR-10, CIFAR-100, and ImageNet, respectively, in a single run. Our DARTS-IF, which follows the perturbation-based selection as DARTS-PT while with an additional approximation term, also outperforms DARTS-PT that again verifies the effectiveness of the influence function explanation. In addition, due to the requirement of fine-tuning in the perturbation-based paradigm, DARTS-PT cost much more than our DARTS-IM. Compared with the perturbation-based selection paradigm, we found that the magnitude paradigm is more efficient and reliable [2].

E Comparison with SOTAs on DARTS space

DARTS search space is a much more complicated search space than NAS-Bench-201 and NAS-Bench-1shot1, and it is intractable to get the ground truth for all architectures in this space. Generally, most existing works only report the best searched architecture, making the reproducibility of architecture search weak in this space. For fair comparison, we also follow the common setting [27, 43] to conduct the architecture search on the

[2]The example codes and training log files could be found in the supplementary material. The trained supernet used in this experiment can be found in https://github.com/anonymous-submission1991/DARTS-IM.

Table 5: Comparison results with state-of-the-art differentiable NAS approaches.

Method	Test Error (%)			Param	$+\times$	Search
	CIFAR-10	CIFAR-100	ImageNet	(M)	(M)	Cost
SETN [11]	2.69	17.25	25.7 / 8.0	4.6	610	1.8
SNAS [42]	2.85±0.02	20.09	27.3 / 9.2	2.8	474	1.5
PARSEC [6]	2.86±0.06	-	26.3	3.6	620	0.6
MdeNAS [54]	2.55	17.61	25.5 / 7.9	3.6	506	0.16
GDAS [10]	2.93	18.38	26.0 / 8.5	3.4	545	0.21
PC-DARTS [43]	2.57±0.07	17.11	25.1 / 7.8	3.6	586	0.3
DARTS (1st) [27]	2.94	17.76	-	2.9	513	0.15†
DARTS [27]	2.76±0.09	17.54	26.9 / 8.7	3.4	574	0.4†
DARTS-PT [27]	2.61±0.08	17.49	26.1 / 8.2	3.3	536	0.8†
DARTS-IM	**2.50±0.10**	**17.02**	**25.0 / 7.6**	3.8	599	0.4†

"Param" is the model size when applied on CIFAR-10, while "$+\times$" (the number of multiply-add operations) is calculated based on the ImageNet dataset. In our experiments, we adjust the number of initial filters to restrict "$+\times$" to be less than 600M on the ImageNet and CIFAR10. † means the computational time is calculated with the same environment as [40]. The unit in "Search Cost" is GPU day.

Table 6: Analyze different approximation method on NAS-Bench-201.

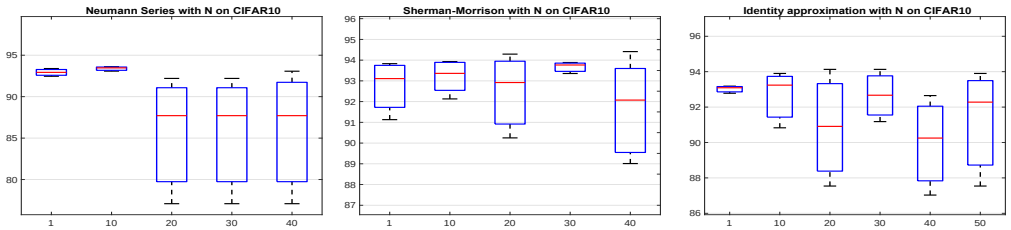
Method	CIFAR-10		CIFAR-100		ImageNet-16-120	
	Valid(%)	Test(%)	Valid(%)	Test(%)	Valid(%)	Test(%)
Random baseline	83.20±13.28	86.61±13.46	60.70±12.55	60.83±12.58	33.34±9.39	33.13±9.66
DARTS [27]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
DARTS-PT [40]	87.34±0.43	89.63±0.19	62.48±2.89	62.35±2.14	36.35±2.76	36.51±2.13
DARTS-IM-I	89.89±0.24	93.11±0.17	69.50±0.60	70.17±0.60	44.45±0.73	44.44±0.32
DARTS-IM-D	89.62±1.70	92.82±1.20	69.17±2.61	69.45±2.72	42.16±3.97	42.07±3.50
DARTS-IM-NS	90.05±0.51	93.35±0.38	69.96±1.13	70.26±1.13	44.43±0.95	44.03±0.75
DARTS-IM-SM	**90.92±0.34**	**93.61±0.23**	**71.21±0.55**	**71.31±0.40**	**44.70±0.74**	**44.98±0.36**

CIFAR-10 dataset in this search space, where the best-found cell is repeatedly stacked to form the full structure for evaluation on CIFAR-10, CIFAR-100, and ImageNet datasets. Please note that, rather than using a default 36 for the number of initial filters, we adjust it to restrict the number of multiply-add operations ("$+\times$") to be less than 600M on the ImageNet and to comparable model size with SOTAs on CIFAR10.

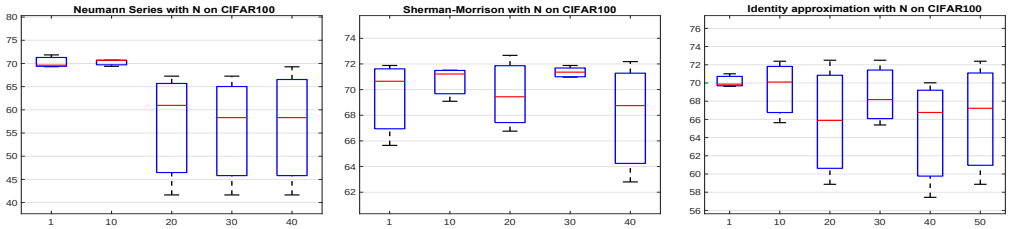
Table 5 presents the comparison results with the SOTA differentiable NAS methods. As shown, our DARTS-IM improves the DARTS baselines by a large margin, in terms of test error on CIFAR-10, CIFAR-100, and ImageNet, respectively, again verifying the effectiveness of the proposed influential magnitude. More interesting, instead of changing the complicated supernet training part, we found that simply replacing the magnitude with the proposed influential magnitude could consistently improve the performance of DARTS with only additional hundred seconds computational time. Different from the DARTS-PT which contains the supernet retraining that greatly increases the search time, our DARTS-IM only poses additional hundred seconds in DARTS space. We could also find that, although the best CIFAR-10 test error by DARTS-PT is comparable with our DARTS-IM, our best searched architecture obtains lower test error in CIFAR-100 and ImageNet, showing better transferability.

F Analyse the inverse Hessian approximation

In this section, we detailed discuss the effects of different inverse Hessian approximation methods on the proposed influence magnitude. Apart from the two devised method, Neumann series and Sherman-Morrison formula, we also consider two commonly-used approaches in approximating the inverse Hessian, identity approximation and diagonal approximation, which consider the Hessian matrix as an identity matrix and diagonal matrix, respectively. We conduct a series of experiments on NAS-Bench-201 to analyze different approximation methods, and all comparison results are reported in Table 6. As shown, even with the identity approximation, our influential magnitude $\mathcal{I}_{\mathcal{M}}$ can achieve satisfying results compared with the random baseline and two heuristic methods. This result indicates that the second-order information $\frac{\partial^2 \mathcal{L}(\theta^*,\alpha)}{\partial \alpha \partial \theta}$ and $\frac{\partial^2 \mathcal{L}(\theta^*,\alpha)}{\partial \theta \partial \alpha}$ can provide useful information for operation strength estimation. More interesting, when considering the diagonal Hessian information, we found that our DARTS-IM-D could also achieve excellent performance. The results of



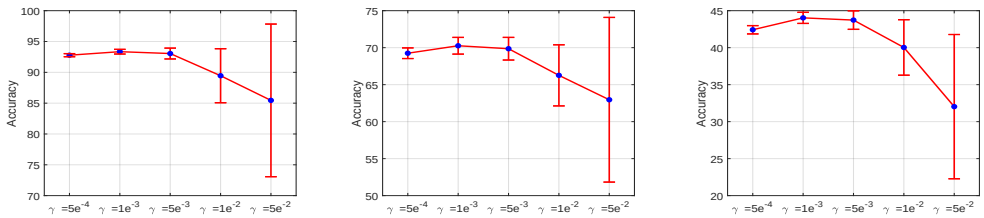
(a) Neumann series. (b) Sherman-Morrison formula. (c) Identity



(d) Neumann series. (e) Sherman-Morrison formula. (f) Identity

(g) Neumann series. (h) Sherman-Morrison formula. (i) Identity

Figure 5: Ablation study on N under two approximation methods, where x-axis is N and y-axis represents test accuracy on CIFAR-10, CIFAR-100, and ImageNet, respectively.



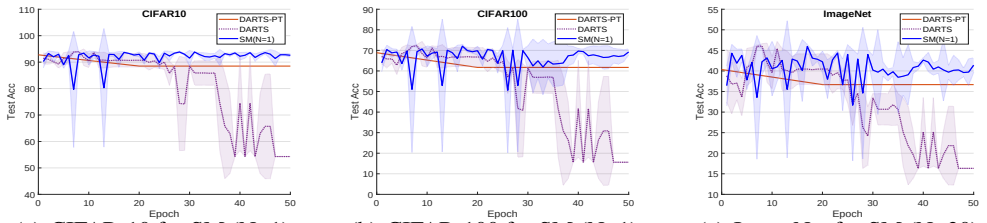
(a) CIFAR-10 (b) CIFAR-100 (c) ImageNet-16-120

Figure 6: Hyperparameter γ analysis of DART-IM-NS on the NAS-Bench-201 benchmark dataset.

identity approximation and diagonal approximation demonstrate the robustness of our influential magnitude, which can still achieve excellent results with different inverse Hessian approximation methods.

However, we also found that an incorrect approximation may lead to poor performance. Similar to Figure 2, we plots the full NAS-Bench-201 results of our influential magnitude under different inverse Hessian matrix approximation methods along with the number of batches N, on the CIFAR10, CIFAR100, and ImageNet, respectively. We compared our DARTS-IM with the DARTS baseline and also the DARTS-PT whose results are directly obtained from [40]. Please note that, DARTS-PT smooth the line extremely with only report the tendency of the architecture selection during the search. We can found that the Neumann series approximation could not obtain stable results, especially when the number of batches increases that the performance of Neumann series even performs more poorly than identity approximation. As discussed in Section 6.1, these results also verify that the stochastic Neumann series estimation for IHVP is somehow erroneous. Generally, we found that the Sherman-Morrison formula is the most practical in estimating the inverse Hessian for our influential magnitude.

Different from Sherman-Morrison formula that only has a hyperparameter N to be tuned and η is fixed in the optimizer, the Neumann series approximation has a key hyperparameter γ [29, 36] to be tuned, which is assumed



(a) CIFAR-10 for SM (N=1). (b) CIFAR-100 for SM (N=1). (c) ImageNet for SM (N=30)

(d) CIFAR-10 for SM (N=30). (e) CIFAR-100 for SM (N=30). (f) ImageNet for SM (N=1)

Figure 7: Track performance of the derived architectures during the search on NAS-Bench-201 with Sherman-Morrison formula under different N for CIFAR-10, CIFAR-100, and ImageNet, respectively.

to be small enough that $\gamma < \frac{1}{\sigma_{max}}$ in Appendix B.1. In this way, we conduct the hyperparameter analysis on γ for our DARTS-IM when employing the Neumann series approximation. Figure 6 compares the performance of DARTS-IM with different γ on the NAS-Bench-201. As shown, the Neumann series approximation is sensitive to the γ, where a large γ significantly deteriorates the performance of DARTS-IM. For example, a $\gamma = 0.05$ even makes our DART-IM similar to the performance of the random baseline. Compared with the Neumann series, we found that the Sherman-Morrison approximation is more robust.

G Visualization of Searched Architectures on Different Spaces

In Figure 8 and 9, we visualize all searched architectures by DARTS baselines, DARTS-PT and also our DARTS-IM on different search space, including NAS-Bench-201 [12], tool search spaces proposed by [45], and DARTS search space [27].

First, we analyze these architectures searched on NAS-Bench-201 [12] and tool search spaces proposed by [40, 45], since which helps us to clearly demonstrate the robustness of different architecture selection methods. As observed by [40, 45], the most notable drawback of DARTS baseline is the robustness and generalization, where DARTS fails to select intensive *skip-connections* with the search progressing. Figure 8 (a-c) plots the searched architectures on NAS-Bench-201, respectively. As shown, the DARTS baseline selects *skip-connection* for all edges, while DARTS-PT can find a valid architecture with only 3 *skip-connections* when considering a perturbation-based selection paradigm. More inspiring, the proposed influential magnitude ($\mathcal{I}_{\mathcal{M}}$) can select more competitive architecture which also only contains one *skip-connection*. Please note that, the trained supernet keeps identical for DARTS, DARTS-PT, and our DARTS-IM, which only adopt different architecture selection methods, e.g., *argmax*, perturbation-based selection, and the proposed influential magnitude.

Similarly, we can also observe this superiority of our $\mathcal{I}_{\mathcal{M}}$ in the tool search spaces proposed by [40, 45], especially in the search space S1 and S2. We can find that, our DARTS-IM clearly outperforms DARTS and DARTS-PT. Although DARTS-PT can partially relieves this instability, it still selects intensive *skip-connections*. In the contrary, our DARTS-IM generally prefers convolutional operations. However, although our DARTS-IM outperforms DARTS baseline in the search space S3, we found that DARTS-PT achieves excellent results in this space. One potential is that this space contains intensive *none* operations which is naturally removed by perturbation-based selection, since removing *none* will not affect the supernet performance. We also conduct the architecture search on the search space S4, while we found that our DARTS-IM also performs poorly as DARTS that select some *noise* operations. One underlying reason is that it is impossible to train the supernet to optimal in this space since the *noise* operation will return a random noise regardless of the input. In addition, DARTS-PT only locally compares the operations' strength in each edge, and a fine-tuning is needed after every step of edge discretization. Differently, our DARTS-IM in this paper consider a global comparison among all operations in a supernet, which may bring some negative operation coupling. To further improve our proposed DARTS-IM, a simple and straight approach is to gradually discretize edges in a supenet with fine-tuning, which we leave

Table 7: Comparison in test error (%) with DARTS baselines on S1-S4 space.

Dataset	Space	DARTS	DARTS-ES	DARTS-PT	DARTS-IM (Avg.)	DARTS-IM (Best)
C10	S1	4.66±0.71	3.05±0.07	3.50	**2.85±0.17**	2.68
	S2	4.42 ± 0.40	3.41 ± 0.14	2.79	**2.51±0.05**	**2.46**
	S3	4.12 ± 0.85	3.71 ± 1.14	**2.49**	3.91±0.07	2.83
	S4	6.95 ± 0.18	4.17 ± 0.21	**2.64**	3.95±0.17	3.72
C100	S1	29.93 ± 0.41	28.90 ± 0.81	24.48	**22.80 ± 0.32**	**22.45**
	S2	28.75 ± 0.92	24.68 ± 1.43	23.16	**21.97 ± 0.61**	**21.27**
	S3	29.01 ± 0.24	26.99 ± 1.79	**22.03**	24.02 ± 0.81	23.12
	S4	24.77 ± 1.51	23.90 ± 2.01	**20.80**	23.58 ± 0.73	22.70
SVHN	S1	9.88 ± 5.50	2.80 ± 0.09	2.62	**2.47 ± 0.12**	2.35
	S2	3.69 ± 0.12	2.68 ± 0.18	2.53	**2.50 ± 0.09**	**2.42**
	S3	4.00 ± 1.01	2.78 ± 0.29	2.42	**2.38 ± 0.11**	**2.28**
	S4	2.90 ± 0.02	2.55 ± 0.15	**2.42**	2.84 ± 0.10	2.73

for our future work. We also show the comparison results in the Table 7 with different DARTS baselines. We directly report the results of DARTS and DARTS-ES from [45], which contain the mean and std. The results of DARTS-PT is also inherited from the original paper [40], which only report the best single run. In Table 7, we report the average and best single run of our DARTS-IM together for better comparison.

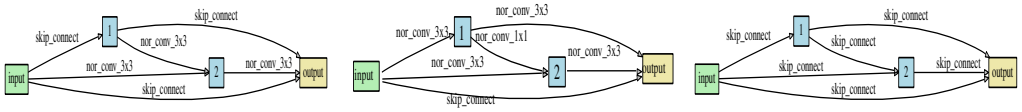
Then, we also visualized our searched architectures in the DARTS search space. Interesting, compared with the DARTS baseline, architecture searched by our DARTS-IM contains several *sep_conv* 5×5 operations in the normal cell, where this phenomenon has never been observed by existing popular DARTS based algorithms. Similarly, we also found that our reduction cell contains several *sep_conv* 5×5 operations, which is also in line with another influence based method DARTS-PT and PC-DARTS. More important, different from most existing works that the normal cell generally contains more parameters than the reduction cell, we found that our searched architecture prefers more parameters in the reduction cell. This observation is more consistent with the common experience in the network pruning [24, 25, 39], where we found that most pruning methods will retain more parameters in the reduction cell and prune more redundant parameters in the normal cell.

H Overall Algorithm Framework Description

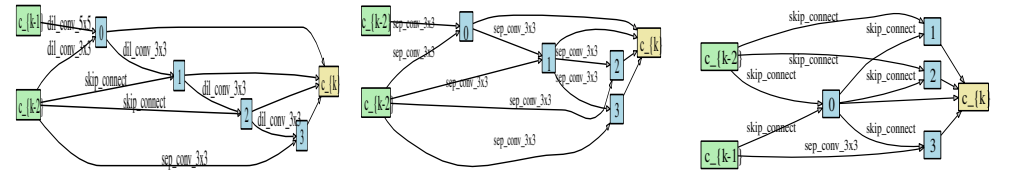
In this paper, we proposed two different approaches for the operation selection in DARTS, called DARTS-IF (described in Sec.3) and DARTS-IM (described in Sec.4). This two approaches are based on the influence functions to measure the operation importance, where the main different is that DARTS-IF removes a candidate operation in each step to measure the loss change in a perturbation manner, while DARTS-IM calculate the loss sensitivity when applying an infinitesimal change on α. The framework of our DARTS-IF and DART-IM are sketched in Algorithm 1 and 2. Our DARTS-IF shares a similar perturbation paradigm as DARTS-PT, while without any fine-tuning. Rather than using the validation performance drop to indicate the operation importance, DARTS-IF leverages influence functions to predict the loss change based on Eq. (5). In our practical implementation, we consider the Neumann series as described in Lemma 1, or the Sherman-Morrison formula as described in Lemma 2, to approximate the IHPV $\frac{\partial^2 \mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta\partial\theta}^{-1} \frac{\partial \mathcal{L}(\theta^*,\hat{\alpha})}{\partial\theta}$. More important, different from DARTS-PT that gradually discretize edges and finetune the supernet, our DARTS-IF only approximates the effect of independently removing one candidate operation for each edge. In this way, we can get the strength for all candidate operations under every edge in a supernet.

Different from DARTS-IF and DARTS-PT, our DARTS-IM is with a similar paradigm as DARTS, while whose operation importance is calculated based on $\mathcal{I}_{\mathcal{M}}$, rather that the optimized α. In addition, the operation importance in our DARTS-IM can be calculated by one-shot, rather than iteratively going through all candidates operations in a supernet as perturbation paradigm.

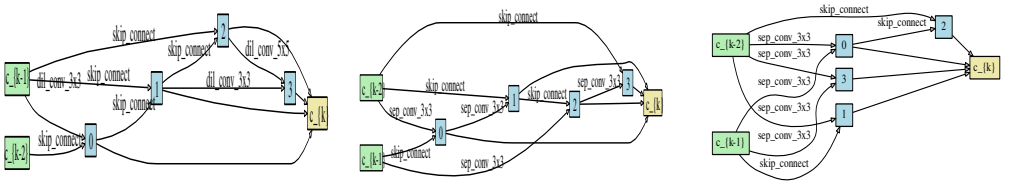
In calculating the influence magnitude in our DARTS-IM or the loss change in DARTS-IF, we have three more hyperparameters, γ for Neumann series approximation, η for Sherman-Morrison approximation, and batch size N for both. In our experiments, we set $\gamma = 0.001$ that is same as the learning rate in optimizing θ, and η is the weight decay for regularization, which both can be obtained form the optimizer by default. In the hyperparameter analysis in Figure 2, we found that $N = 10$ is enough for both Neumann series and Sherman-Morrison approximation.



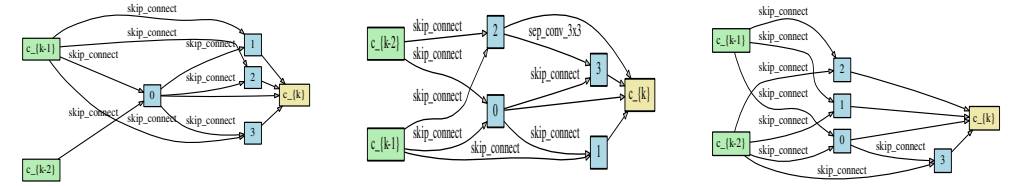
(a) DARTS-PT: NAS-BENCH-201 (b) DARTS-IM: NAS-BENCH-201 (c) DARTS: NAS-BENCH-201



(d) DARTS-IM: S1 (e) DARTS-IM: S2 (f) DARTS-IM: S3



(g) DARTS-PT: S1 (h) DARTS-PT: S2 (i) DARTS-PT: S3

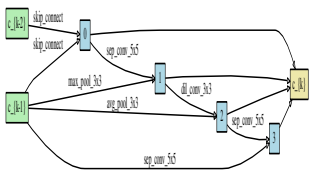


(j) DARTS: S1 (k) DARTS: S2 (l) DARTS: S3

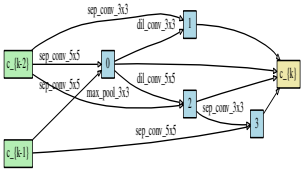
Figure 8: Searched architectures on NAS-Bench-201 and several tool search spaces in [40, 45].

Algorithm 1 N Differentiable Architecture Search with Influence Functions (DARTS-IF)

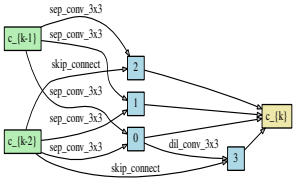
Input: A pretrained supernet after bi-level training process (θ^*, α), candidate operations for each edge $\mathcal{O}$, and set of edges $\mathcal{E}$ from the supernet.
output: A discrete architecture α^*.
for $e \in \mathcal{E}$ **do**
  for $o \in \mathcal{O}$ **do**
    Remove candidate operation o from edge e;
    Calculate the predictive loss chance $\Delta\mathcal{L}_{o,e}$ based on Eq. (5), that $\Delta\mathcal{L}_{o,e} \approx \mathcal{L}(\theta^*, \hat{\alpha}) - \mathcal{L}(\theta^*, \alpha) - 1/2 \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}^T \frac{\partial^2 \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta \partial \theta}^{-1} \frac{\partial \mathcal{L}(\theta^*, \hat{\alpha})}{\partial \theta}$, as the operation strength;
    Restore o to $\mathcal{O}$;
  end for
end for
Apply *argmax* on the operation strength $\Delta\mathcal{L}$ and derive the discrete architecture α^* accordingly.



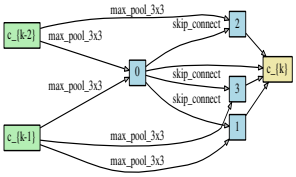
(a) Influential Magnitude ($\mathcal{I}_{\mathcal{M}}$): Normal Cell



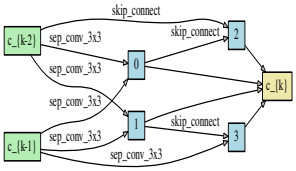
(b) Influential Magnitude ($\mathcal{I}_{\mathcal{M}}$): Reduction Cell



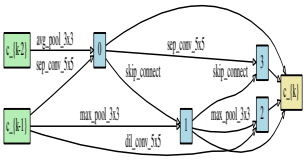
(c) DARTS: Normal Cell



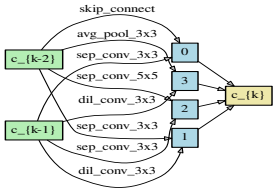
(d) DARTS: Reduction Cell



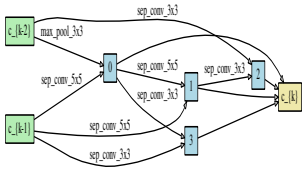
(e) DARTS-PT: Normal Cell



(f) DARTS-PT: Reduction Cell



(g) PC-DARTS: Normal Cell



(h) PC-DARTS: Reduction Cell

Figure 9: Searched architectures on DARTS search space.

Algorithm 2 Differentiable Architecture Search with Influence Magnitude (DARTS-IM)

1: **Input**: A pretrained supernet after bi-level training process (θ^*, α), candidate operations for each edge $\mathcal{O}$, and set of edges $\mathcal{E}$ from the supernet.
2: **output:** A discrete architecture α^*.
3: Calculate the influence magnitude $\mathcal{I}_{\mathcal{M}} = -\mathbf{1}^T \frac{\partial^2 \mathcal{L}(\theta^*,\alpha)}{\partial \alpha \partial \theta} H^{-1} \frac{\partial^2 \mathcal{L}(\theta^*,\alpha)}{\partial \theta \partial \alpha}$ based on Definition1;
4: Apply *argmax* on the influence magnitude $\mathcal{I}_{\mathcal{M}}$ and derive the discrete architecture α^* accordingly.