AutoDiffusion: Training-Free Optimization of Time Steps and Architectures for Automated Diffusion Model Acceleration

Lijiang Li[1], Huixia Li[2], Xiawu Zheng[1,3,4,5], Jie Wu[2], Xuefeng Xiao[2],
Rui Wang[2], Min Zheng[2], Xin Pan[2], Fei Chao[1]*, Rongrong Ji[1,3,4,5],
[1]Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, Department of Artificial Intelligence, School of Informatics,
Xiamen University. [2]ByteDance Inc. [3]Peng Cheng Laboratory.
[4]Institute of Artificial Intelligence, Xiamen University. [5]Fujian Engineering
Research Center of Trusted Artificial Intelligence Analysis and Application, Xiamen University.
lilijiang@stu.xmu.edu.cn, zhengxw01@pcl.ac.cn, wujie10558@gmail.com, {feichao, rrji}@xmu.edu.cn
{lihuixia, xiaoxuefeng.ailab, ruiwang.rw, zhengmin.666, panxin.321}@bytedance.com

Abstract

*Diffusion models are emerging expressive generative models, in which a large number of time steps (inference steps) are required for a single image generation. To accelerate such tedious process, reducing steps uniformly is considered as an undisputed principle of diffusion models. We consider that such a uniform assumption is not the optimal solution in practice; i.e., we can find different optimal time steps for different models. Therefore, we propose to search the optimal time steps sequence and compressed model architecture in a unified framework to achieve effective image generation for diffusion models without any further training. Specifically, we first design a unified search space that consists of all possible time steps and various architectures. Then, a two stage evolutionary algorithm is introduced to find the optimal solution in the designed search space. To further accelerate the search process, we employ FID score between generated and real samples to estimate the performance of the sampled examples. As a result, the proposed method is (i).**training-free**, obtaining the optimal time steps and model architecture without any training process; (ii). **orthogonal** to most advanced diffusion samplers and can be integrated to gain better sample quality. (iii). **generalized**, where the searched time steps and architectures can be directly applied on different diffusion models with the same guidance scale. Experimental results show that our method achieves excellent performance by using only a few time steps, e.g. 17.86 FID score on ImageNet* 64×64 *with only four steps, compared to 138.66 with DDIM.*

*Corresponding author: fchao@xmu.edu.cn

1. Introduction

Diffusion models are a class of generative models that exhibit remarkable performance across a broad range of tasks, including but not limited to image generation [14, 24, 8, 2, 29, 4, 15, 38], super-resolution [33, 39, 6], inpainting [22, 31], and text-to-image generation [25, 32, 27, 10]. These models utilize the diffusion process to gradually introduce noise into the input data until it conforms to a Gaussian distribution. They then learn the reversal of this process to restore the data from sampled noise. Consequently, they achieve exact likelihood computation and excellent sample quality. However, one major drawback of diffusion models is their slow generation process. For instance, on a V100 GPU, generating a 256×256 image with StyleGAN [16] only takes 0.015s, whereas the ADM model requires multiple time steps for denoising during generation, leading to a significantly longer generation time of 14.75s.

Extensive studies have focused on reducing the number of time steps to improve the generation process of diffusion models. Some of these studies represent the generation process as either stochastic differential equations (SDEs) or ordinary differential equations (ODEs), and then utilize numerical methods to solve these equations [36, 20, 6, 21]. The samplers obtained by these numerical methods can typically be applied to pre-trained diffusion models in a plug-and-play manner without re-training. The other studies proposed to utilize knowledge distillation to reduce the number of time steps [34, 23]. These methods decrease the time steps required for the generation process and then allow the noise prediction network to learn from the network of the original generation process. Although these methods are effective in improving the sampling speed of diffusion mod-

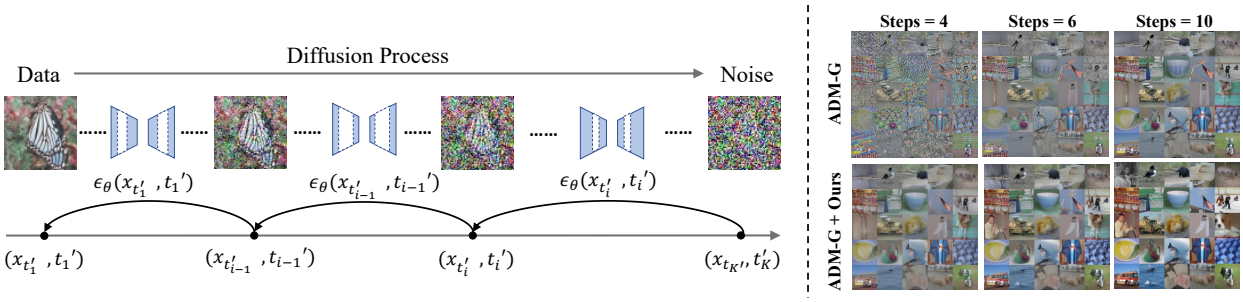


Figure 1. Left: We propose to search the optimal time steps sequence and corresponding compressed network architecture in a unified framework. Right: Samples by ADM-G [8] pre-trained on ImageNet 64×64 with and without our methods (AutoDiffusion), varying the number of time steps.

els, we observe that they have paid little attention to the selection of time step sequences. When reducing the number of time steps, most of these methods sample the new time steps uniformly or according to a specific procedure [36]. We argue that there exists an optimal time steps sequence with any given length for the given diffusion model. And the optimal time steps sequence varies depending on the specific task and the super-parameters of diffusion models. We believe that the generation quality of diffusion models can be improved by replacing the original time steps with the optimal time steps.

Therefore, we introduce AutoDiffusion, a novel framework that simultaneously searches optimal time step sequences and the architectures for pre-trained diffusion models without additional training. Fig. 1 (Left) shows the schematic of AutoDiffusion. Our approach is inspired by Neural Architecture Search (NAS) techniques that are widely used for compressing large-scale neural networks [28, 42, 26, 18, 1]. In our method, we begin with a pre-trained diffusion model and a desired number of time steps. Next, we construct a unified search space comprising all possible time step sequences and diverse noise prediction network architectures. To explore the search space effectively, we use the distance between generated and real samples as the evaluation metric to estimate performance for candidate time steps and architectures. Our method provides three main advantages. First, we demonstrate through experiments that the optimal time steps sequence obtained through our approach leads to significantly better image quality than uniform time steps, especially in a few-step regime, as illustrated in Fig. 1 (Right). Second, we show that the searched result of the diffusion model can be applied to another model using the same guidance scale without repeating the search process. Furthermore, our approach can be combined with existing advanced samplers to further improve sample quality.

Our main contributions are summarized as follows:

- Our study reveals that uniform sampling or using a fixed function to sample time steps is suboptimal for diffusion models. Instead, we propose that there exist an optimal time steps sequence and corresponding noise prediction network architecture for each diffusion model. To facilitate this, we propose a search space that encompasses both time steps and network architectures. Employing the optimal candidate of this search space can effectively improve sampling speed for diffusion models and complement the most advanced samplers to enhance sample quality.

- We propose a unified training-free framework, AutoDiffusion, to search both time steps and architectures in the search space for any given diffusion model. We utilize a two-stage evolutionary algorithm as a search strategy and the FID score as the performance estimation for candidates in the search space, enabling an efficient and effective search process.

- Extensive experiments show that our method is training-free, orthogonal to most advanced diffusion samplers, and generalized, where the searched time steps and architectures can be directly applied to different diffusion models with the same guidance scale. Our method achieves excellent performance by using only a few time steps, *e.g.*, 17.86 FID score on ImageNet 64×64 with only four steps, compared to 138.66 with DDIM. Furthermore, by implementing our method, the samplers exhibit a noteworthy enhancement in generation speed, achieving a $2\times$ speedup compared to the samplers lacking our method.

2. Related Work

2.1. Diffusion Models

Given a variable $x_0 \in \mathbb{R}^D$ that sampled from an unknown distribution $p_{data}(x_0)$, diffusion models define a diffusion process $\{x_t\}_{t\in[0:T]}$ to convert the data x_0 into sample x_T by T diffusion steps. The distribution of the sample x_T denoted as $p(x_T)$ is usually simple and tractable, such as standard normal distribution. In the diffusion process, the

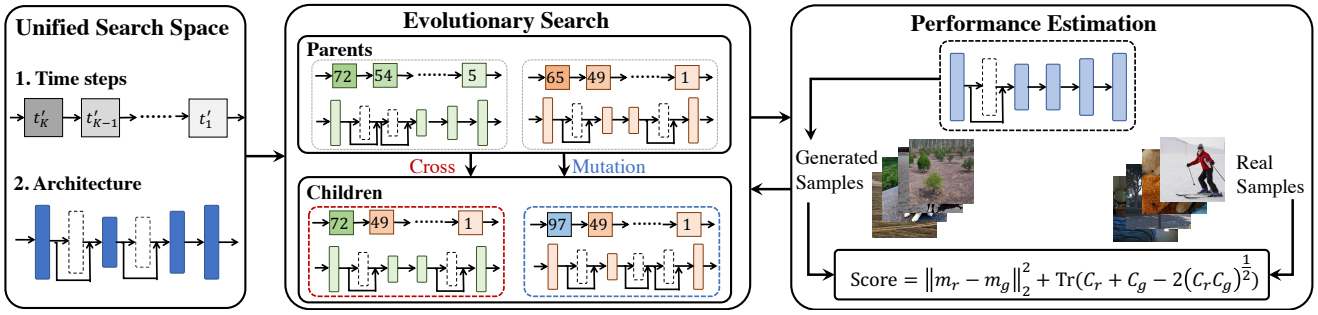


Figure 2. Overview on AutoDiffusion. Given a pre-trained diffusion model, we first design a unified search space that consists of both time steps and architectures. After that, we utilize the FID score as the performance estimation strategy. Finally, we apply the evolutionary algorithm to search for the optimal time steps sequence and architecture in the unified search space.

distribution of variable x_t at time step t satisfies:

$$q(x_t|x_0) = \mathcal{N}(x_t|\alpha_t x_0, \beta_t^2 I) \tag{1}$$

where $\{\alpha_1, \alpha_2, \cdots, \alpha_T\}$ and $\{\beta_1, \beta_2, \cdots, \beta_T\}$ are super-parameters of diffusion models that control the speed of converting x_0 into x_T.

After that, diffusion models define a reverse process $p_\theta(x_{t-1}|x_t)$ parameterized by neural network θ and optimize it by maximizing the log evidence lower bound (ELBO) [24]:

$$\begin{aligned} L_{elbo} = \mathbb{E}[&\log p_\theta(x_0|x_1) \\ &- \Sigma_{t=1}^{T} D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \\ &- D_{KL}(q(x_T|x_0)||p(x_T))] \end{aligned} \tag{2}$$

where D_{KL} denote the KL-divergence.

In practice, diffusion models use a noise prediction network $\epsilon_\theta(x_t, t)$ to estimate the noise component of the noisy sample x_t at time step t. Therefore, the loss function in Eq. 2 can be simplified as follow [14]:

$$L_{simple} = \|\epsilon_\theta(x_t, t) - \epsilon\|^2 \tag{3}$$

where ϵ represent the noise component of x_t and we have $x_t = \alpha_t x_0 + \beta\epsilon$ according to Eq. 1. In most diffusion models, the noise ϵ is sampled from standard normal distribution $\mathcal{N}(0, I)$ when generating noisy sample x_t.

When the noise prediction network $\epsilon_\theta(x_t, t)$ is trained, diffusion models define a generation process to obtain samples. This process begins with noisy data sampled from $p(x_T)$, yielding progressively cleaner samples $x_{T-1}, x_{T-2}, \cdots, x_0$ via the learned distribution $p_\theta(x_{t-1}|x_t)$. This process needs T forward of the noise prediction network ϵ_θ to obtain final sample x_0. To hasten this, many studies tried to reduce the number of time steps to $K < T$. They proposed many advanced samplers to compensate for the loss of sample quality caused by reducing time steps. But most of them overlooked optimal time step selection and usually sampled new time steps based on simple functions. For example, DDIM [36] select time steps in the linear or quadratic procedure. The linear procedure generate new time steps sequence with length K such that $[0, \frac{T}{K}, \cdots, \frac{KT}{K}]$. Our key contribution is searching the K-length optimal time steps sequence for diffusion models.

2.2. Neural Architecture Search

The aim of NAS algorithms is to automatically search for an appropriate neural network architecture within an extensive search space. NAS is composed of three essential components: the search space, the search strategy, and the performance estimation strategy [9]. The search space specifies the set of architectures to be explored and determines the representation of candidate neural networks. The search strategy outlines the approach employed to explore the search space. Typically, the strategy involves selecting a new candidate from the search space based on the performance estimation of the currently selected candidate. The performance estimation strategy defines the approach for evaluating the performance of a candidate neural network in the search space. An effective performance estimation strategy ensures accurate and swift evaluations, underpinning both the efficacy and speed of the NAS [44].

NAS algorithms have been applied to design suitable network architecture in various fields. Therefore, in this work, we aim to optimize the time steps and architecture of diffusion models using this technique.

2.3. Fast Sampling For Diffusion Models

Numerous studies aim to improve the generation speed of diffusion models. Some approaches model the generation process with SDEs or ODEs, leading to training-free samplers [36, 20, 21]. However, when the number of steps drops below 10, these methods often degrade image quality [3]. Other methods accelerate diffusion models via knowledge distillation [34, 23, 3] or by learning a fast sampler [40]. For example, progressive distillation (PD) uses knowledge distillation to halve the number of time steps [34]. This distillation is iteratively conducted until the number of steps is less than 10, often demanding substantial computational resources. DDSS treats sampler design as a differentiable

optimization problem, utilizing the reparametrization trick and gradient rematerialization to learn a fast sampler [40]. Although DDSS offers notable speedups, it lacks flexibility, as samplers tailored for one model may not fit another, requiring distinct learning stages. Compared with these methods, AutoDiffusion is much more efficient and flexible, as substantiated by our experiments. Its searched result can be transferred to another diffusion model using the same guidance scale without re-searching. Furthermore, AutoDiffusion utilizes a unified search space for time steps and model layers, while existing methods only focus on step reduction.

3. Method

In this section, we introduce our AutoDiffusion, which aims to search for the optimal time steps sequence and architecture for given diffusion models. The overview of our method is shown in Fig. 2. In the following contents, we first discuss the motivation of our method in Sec. 3.1. Then, we introduce the search space in Sec. 3.2. After that, we elaborate the performance evaluation in Sec. 3.3. Finally, we introduce the evolutionary search in Sec. 3.4.

3.1. Motivation

Many well-recognized theories pointed out that the generation process of diffusion models is divided into several stages, in which the behavior of diffusion models is different at each stage [5, 7]. For example, Ref [5] illustrated that the behavior of diffusion models at each time step can be classified into creating coarse features, generating perceptually rich contents, and removing remaining noise. Intuitively, the difficulty of these tasks is different. In other words, the denoise difficulty of diffusion models varies with the time steps. Inspired by these studies, we hypothesize that the importance of each time step in the generation process is different. In this case, we argue that there exists an optimal time steps sequence for diffusion models among all possible time steps sequences.

To investigate our hypothesis, we conduct an experiment in which we obtain samples, denoted as x_t, and calculate the Mean Squared Error (MSE) $\|x_t - x_{t+100}\|^2$ for each time step t. The results are presented in Fig. 3, which shows that the samples obtained for $t \in [600, 1000]$ are dominated by noise and thus illegible. Conversely, when $t \in [300, 600]$, the diffusion model generated the main contents of the image, and the objects in the generated image become recognizable. It is observed that the diffusion model primarily removes noise at $t \in [0, 300]$, resulting in similar samples for $t \in [0, 300]$. Furthermore, Fig. 3 indicates that the MSE is low at $t \in [0, 100]$ and $t \in [700, 900]$, while it becomes high at $t \in [200, 600]$. Based on the findings in Fig. 3, it is apparent that different time steps play varying roles in the generation process of diffusion models. Specifically, when t is small or large, the content of the generated

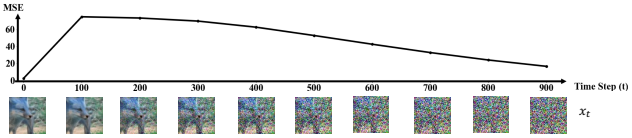


Figure 3. Sample x_t and MSE $\|x_t - x_{t+100}\|^2$ over time steps t.

samples changes slowly. In contrast, when t is in the middle, the content changes rapidly. Therefore, we contend that uniform time steps are suboptimal and that an optimal time step sequence exists for the generation process of diffusion models. Further, since the denoise difficulty varies depending on time steps, we believe that the model size of the noise prediction network is not necessarily the same at each time step. Thus, we search the time steps and architectures in a unified framework.

3.2. Search Space

In this section, we discuss how the search space is designed in AutoDiffusion. Given a diffusion model with timesteps $[t_1, t_2, \cdots, t_T]$ ($t_i < t_{i+1}$), it needs T calls of the noise prediction network ϵ_θ to yield a batch of images. To accelerate the generation process, two approaches are usually employed: reducing the number of time steps or the number of layers in the ϵ_θ. To this end, we propose a search space comprising two orthogonal components: 1) the temporal search space that takes time steps as the searched object; 2) the spatial search space that takes the architectures of the noise prediction network ϵ_θ as the searched object. In our search space, the candidate $cand$ is defined as follows:

$$\begin{aligned} cand = \{&\boldsymbol{\mathcal{T}} = [t'_1, t'_2, \cdots, t'_K]; \\ &\boldsymbol{\mathcal{L}} = [\boldsymbol{L_1}, \boldsymbol{L_2}, \cdots, \boldsymbol{L_K}]\}, \\ &0 < t'_{i+1} - t'_i < t_T - t_1, \\ &t'_i \in [t_1, t_2, \cdots, t_T] \ (i = 1, 2, \cdots, K) \end{aligned} \tag{4}$$

where $\boldsymbol{\mathcal{T}}$ denotes the sampled time steps sequence, and $[t'_1, t'_2, \cdots, t'_K]$ is a sub-sequence of the original time steps sequence $[t_1, t_2, \cdots, t_T]$. $\boldsymbol{\mathcal{L}}$ denotes the sampled architectures, where $\boldsymbol{L_i} = [l_i^1, l_i^2, \cdots, l_i^{n_i}]$ is the architecture of the noise prediction model at time step t'_i. n_i is the number of architecture layers at time step t'_i , which must be no more than the layers number of ϵ_θ. Each $l_i^j \in \boldsymbol{L_i}$ represents one layer of the noise prediction network ϵ_θ at time step t'_i, thus $\boldsymbol{L_i}$ can be viewed as a sub-network of ϵ_θ. In practice, we constrain the sum of model layers at each time step to be no more than N_{max}, *i.e.* $\sum_{i=1}^{K} n_i \leq N_{\text{max}}$, where N_{max} is determined according to the expected generation speed of diffusion models.

In the temporal aspect, we search for the optimal time steps sequence among all possible time steps. In the spatial aspect, we search for the model layers of the noise prediction network at each time step. Therefore, we can search for the best time steps sequence and the compressed noise

prediction model in a unified framework. Notably, the sub-network $\boldsymbol{L_i}$ may not be the same across all time steps during the search, as the difficulty of denoising varies at different time steps. We believe that the number of layers n_i at each time step t'_i reflects the level of denoising difficulty at t'_i.

Since the noise prediction networks ϵ_θ are usually U-Net, we don't add up-sample or down-sample layers into the search space. In practice, if a model layer is not selected in a candidate, the model layer will be replaced by a skip connection. Besides, the searched sub-networks of ϵ_θ are not retrained or fine-tuned in the search process.

3.3. Performance Estimation

After the search space is determined, we need to select evaluation metrics to provide fast and proper performance estimation for the search process. There are two classes of evaluation metrics that may meet the requirements, one is the distance between learned distribution $p_\theta(x_{t_{i-1}}|x_{t_i})$ and posteriors $q(x_{t_{i-1}}|x_{t_i}, x_0)$, the other is the distance between the statistics of generated samples and real samples.

The distance between distribution $p_\theta(x_{t_{i-1}}|x_{t_i})$ and posteriors $q(x_{t_{i-1}}|x_{t_i}, x_0)$ is usually estimated using KL-divergence. Therefore, the performance estimation of a sorted candidate time steps $[t'_1, t'_2, \cdots, t'_K]$ can be obtained by using KL-divergence [24] as follows:

$$
\begin{aligned}
L &= L_{t'_1} + L_{t'_2} + \cdots + L_{t'_K} \\
L_{t'_i} &= \begin{cases} D_{KL}(q(x_{t'_i}|x_0)||p(x_{t'_i})), & t'_i = t_T \\ -\log p_\theta(x_{t'_i}|x_{t'_{i+1}}), & t'_i = 0 \\ D_{KL}(q(x_{t'_i}|x_{t'_{i+1}}, x_0)||p_\theta(x_{t'_i}|x_{t'_{i+1}}), & others \end{cases}
\end{aligned} \tag{5}
$$

Given a trained diffusion model, the image x_0 sampled from the training dataset, and the candidate time steps $[t'_1, t'_2, \cdots, t'_K]$, we use Eq. 5 to calculate the KL divergence, which allows a fast performance estimation. However, prior work has pointed out that optimizing the KL-divergence can not improve sample quality [41, 37]. To verify this conclusion, we use the time steps sequence $[t_1, t_2, \cdots, t_T]$ of a diffusion model trained on ImageNet 64×64 as the search space. Then, we sample subsequences $[t'_1, t'_2, \cdots, t'_K]$ from this search space randomly and calculate the FID score, sFID score, IS score, precision, recall, and the KL-divergence of these subsequences. After that, we analyze the relevancy between FID, sFID, IS, precision, recall, and KL-divergence of these subsequences by calculating the Kendall-tau [17] between them. Tab. 1 shows that the Kendall-tau values between all these metrics and KL-divergence are low, which means that the KL-divergence can not represent the sampled quality.

The distance between the statistics of generated samples and real samples can be estimated using the KID score or FID score. Daniel *et al.* proposed to optimize the sampler of diffusion models by minimizing KID loss [40]. Inspired by this work, we use FID score as the performance estimation metric. The FID score is formulated as follows [13]:

FID	sFID	IS	Precision	Recall
0.126	0.200	-0.126	-0.190	-0.165

Table 1. Kendall-tau [17] between matrices and KL-divergence.

$$
\text{Score} = \|m_r - m_g\|_2^2 + \text{Tr}\left(C_r + C_g - 2(C_r C_g)^{\frac{1}{2}}\right) \tag{6}
$$

where m_r and m_g are the mean of the feature of real samples and generated samples; while C_r and C_g are covariances of the feature of real samples and generated samples. Usually, the feature of generated samples and real samples can be obtained by pretrained VGG [35] models.

However, we must generate at least 10k samples when calculating precise FID scores, which will slow down the search speed. To address this, we reduce the number of samples for calculating FID scores. We apply Kendall-tau [17] to determine the reduced number of samples. Specifically, we still use the full time steps sequence $[t_1, t_2, \cdots, t_T]$ as search space and sample N_{seq} subsequences $[t'_1, t'_2, \cdots, t'_K]$ randomly from it. Then, we generate 50k samples using each of these subsequences and obtain corresponding FID scores $\{F_1, F_2, \cdots, F_{N_{seq}}\}$. After that, we obtain a subset of N_{sam} samples from 50k samples and calculate their FID score $\{F'_1, F'_2, \cdots, F'_{N_{seq}}\}$. We calculate the Kendall-tau between $\{F_1, F_2, \cdots, F_{N_{seq}}\}$ and $\{F'_1, F'_2, \cdots, F'_{N_{seq}}\}$. The optimal number of samples is the minimum N_{sam} that makes Kendall-tau greater than 0.5.

3.4. Evolutionary Search

We utilize the evolution algorithm to search for the best candidate from the search space since evolutionary search is widely adopted in previous NAS works[28, 11, 12, 19]. In the evolutionary search process, given a trained diffusion model, we sample candidates from the search space randomly using Eq. 4 to form an initial population. For each candidate, we generate samples by utilizing the candidate's time steps and corresponding architecture. After that, we calculate the FID score based on the generated samples. At each iteration, we select the Top k candidates with the lowest FID score as parents and apply cross and mutation to generate a new population. To perform cross, we randomly exchange the time steps and model layers between two parent candidates. To perform mutation, we choose a parent candidate and modify its time steps and model layers with probability p.

When searching for time steps and architectures, we utilize a two-stage evolutionary search. Specifically, we use the full noise prediction network and search time steps only in the first several iterations of the evolutionary search. Then, we search the time steps and model architectures together in the remaining search process.

4. Experimentation

4.1. Experiment Setting

In order to demonstrate that our method is compatible with any pre-trained diffusion models, we apply our method to prior proposed diffusion models. Specifically, we experiment with the ADM and ADM-G models proposed by Prafulla *et al.*[8] that trained on ImageNet 64×64 [30] and LSUN dataset [43]. In addition, we applied our method on Stable Diffusion [29] to verify the effectiveness of our method on the text-to-image generation task. Besides, we also combine our method with DDIM [36], PLMS [20], and DPM-solver [21] and apply them to the Stable Diffusion to demonstrate that our proposed method can be combined with most of the existing advanced samplers and improve their performance. In all experiments, we use the pre-trained checkpoint of these prior works since our method does not need to retrain or fine-tune the diffusion models.

Our method optimizes the generation process of diffusion models from the perspective of both time steps and architecture. Sec. 4.2 illustrates that we can accelerate the generation process by only searching for the optimal time steps. And on this basis, Sec. 4.4 demonstrates that we can improve the sample quality and generation speed further by searching time steps and architecture together. In all experiments, the hyperparameters of evolution algorithm search are set as follows: we set the population size $P = 50$; top number $k = 10$, mutation probability $p = 0.25$, max iterations $MaxIter = 10$ when searching for time steps only, and $MaxIter = 15$ when searching for time steps and architectures. For the experiments without our methods, the diffusion models generate samples with uniform time steps and the full noise prediction network. Besides, all experiments with ADM or ADM-G use DDIM [36] sampler. We evaluate the quality of generated images with FID and IS scores as most previous work.

4.2. Quantitative and Qualitative Results

We apply our method with the pre-trained ADM-G and ADM on various datasets, and the results are shown in Tabs. 2 to 3. Note that we only search time steps without searching model layers of the noise prediction network in these experiments. Our method can improve the sample quality significantly of diffusion models in the few-step regime. In particular, our method exhibits impressive performance when the number of time steps is extremely low. For example, the FID score of ADM-G on ImageNet 64×64 is 138.66, and our method can reduce it to 17.86, which shows that our method can generate good samples in the extremely low-step regime.

We combine our method with DPM-Solver [21], DDIM [36], and PLMS [20] to demonstrate that our method can be integrated with advanced samplers. Fig. 4 shows that our method can improve the sample quality based on these samplers, especially in the low-step case where steps = 4. These results illustrate that our method can be combined with most advanced samplers to further improve their performance. In addition, Fig. 4 illustrates that the samplers with our method can achieve admirable performance within 10 steps, which is $2\times$ faster than the samplers without our method.

Fig. 5 shows the generated samples for Stable diffusion using DPM-Solver with and without our method in a few-

Ours	Steps	FID ↓	IS ↑
×	4	138.66	7.09
✓	4	17.86 (-120.8)	34.88 (+27.79)
×	6	23.71	31.53
✓	6	11.17 (-12.54)	43.47 (+11.94)
×	10	8.86	46.50
✓	10	6.24 (-2.62)	57.85 (+11.35)
×	15	5.38	54.82
✓	15	4.92 (-0.46)	64.03 (+9.21)
×	20	4.35	58.41
✓	20	3.93 (-0.42)	68.05 (+9.64)

Table 2. FID (↓) and IS (↑) scores for ADM-G[8] with and without our method on ImageNet 64×64, varying the number of time steps. The (+number) denotes the improve compare to the resulte without our method.

Ours	Steps	LSUN Bedroom	LSUN Cat
×	5	33.42	48.41
✓	5	23.19 (-10.23)	34.61 (-13.8)
×	10	10.01	20.05
✓	10	8.66 (-1.35)	17.29 (-2.76)
×	15	6.36	14.86
✓	15	5.83 (-0.53)	13.17 (-1.69)

Table 3. FID score (↓) for ADM[8] with and without our method on LSUN dataset, varying the number of time steps.

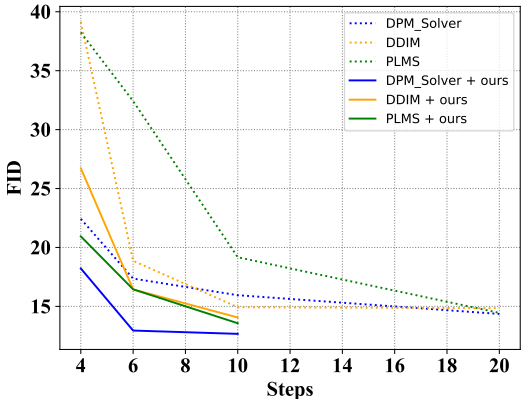


Figure 4. FID score for Stable Diffusion [29] using different samplers with and without our methods. Our method can improve the FID score of DDIM, PLMS, and DPM-solver.

Figure 5. Samples by Stable diffusion [29] with and without our methods using the same random seed, varying the number of time steps. Input prompts are "An astronaut riding a horse" and "An oil painting of a corgi wearing a party hat".

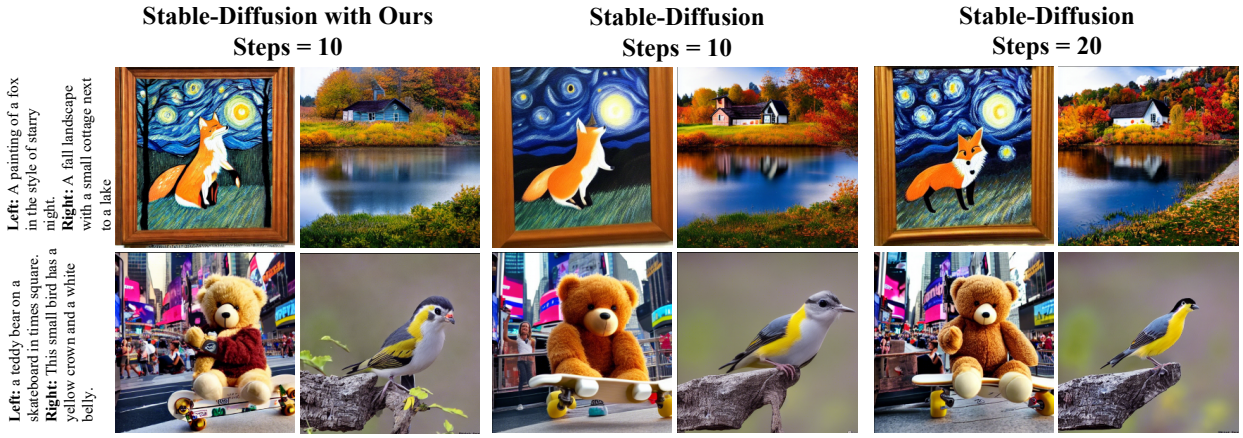
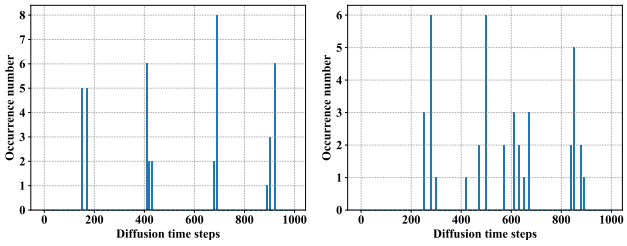


Figure 6. The proposed method is also compatible with the widely-used sampler DPM-Solver. The samples generated by our method with 10 steps are comparable to those generated by 20 steps, and better than those generated by 10 steps using DPM-Solver.

step regime. We find that the samples generated with our method have more clear details than other samples. Fig. 6 demonstrates that the images generated by our method with DPM-Solver at step = 10 are comparable to those generated solely by DPM-Solver at step = 20, and superior to those generated solely by DPM-Solver at step = 10.

4.3. Migrate Search Results

We observe that the guidance scale in the generation process influences the search results significantly, and an optimal time steps sequence derived from one diffusion model can be transferred to another using the same guidance scale. Specifically, we search the optimal time steps sequence of length 4 for ADM-G on the ImageNet 64×64 at guidance scales 1.0 and 7.5. The distribution of searched time steps for ADM-G with these guidance scales differ significantly, as shown in Fig. 7(a) and Fig. 7(b). Further, using a 7.5 guidance scale, we apply the optimal time steps of ADM-G on ImageNet 64×64 to Stable Diffusion on COCO dataset, achieving an FID score of 24.11. In comparison, uniform



(a) (b)

Figure 7. The occurrence number of time steps of top-10 candidates in Evolutionary search. (a). Time steps occurrence number of ADM-G on ImageNet64×64 with guidance scale 1.0. (b). Time steps occurrence number of ADM-G on ImageNet64×64 with guidance scale 7.5. We observe that the distribution of occurrence number is changed depending on the guidance scale in generation process.

time steps and the optimal time steps specifically searched for Stable Diffusion lead to FID scores of 38.25 and 20.93. This result suggests that we can obtain a desirable time steps

searched time steps	searched model layers	Steps	FID ↓	IS ↑	sampling time (s)	N_{max}
✓	✓	4	14.53	38.24	4455	232
✓	×	4	18.07	35.26	4476	232
✓	✓	6	10.26	45.35	6535	350
✓	×	6	10.91	44.93	6712	350
✓	✓	10	6.08	54.62	10655	580
✓	×	10	7.51	55.32	10719	580

Table 4. FID score and IS scores for ADM-G[8] with our proposed method on ImageNet 64×64 dataset. "Sampling time (s)" means the time to generate 50k samples.

sequence without repeating the search process when given a new diffusion model with the same guidance scale. However, we also find that applying the searched results from Stable Diffusion with guidance scale of 7.5 to ADM-G with guidance scale of 1.0 results in poor sample quality. This implies that the searched results from diffusion models with different guidance scales might not be transferable.

4.4. Search for Time Steps and Architecture

We find that our method can achieve satisfactory performance when searching time steps only, but the performance can be further improved by searching model layers together with time steps. In this case, we constrain the sum of model layers at each time step to be less than N_{max}. We repeat the experiment under $N_{\text{max}} = 232$, $N_{\text{max}} = 350$, and $N_{\text{max}} = 580$, while the number of layers in noise prediction model is fixed to 58. After searching, we evaluate the FID score and IS score of diffusion models using the searched time steps and model layers. Besides, we also evaluate the performance of the diffusion model that only uses the searched time steps without using the searched model layers (*e.g.* these diffusion models use a full noise prediction network to generate samples). In all these experiments, we don't retrain or fine-tune the searched subnet of the noise prediction network.

Tab. 4 illustrates that the diffusion model with the searched model layers outperforms the model that employs a full noise prediction network in terms of both FID scores and generation speed. This result suggests that certain layers in the noise prediction network are superfluous.

We conduct an analysis on the searched architecture of Tab. 4. We prune entire residual block and attention block from the noise prediction network in these experiments and observe that the importance of residual and attention blocks varies with the time-step length. Both residual and attention blocks are equally essential for the small time-step length, but attention blocks became increasingly important with more steps.

4.5. Comparison to the Prior Work

We experiment with the DDPM provided by Alexander *et al.* [24] on ImageNet 64×64 against DDSS [40] which proposed to optimize the noise and time step schedule with differentiable diffusion sampler search. Tab. 5 demonstrates that our method can achieve a better FID score and IS score than DDSS.

Method \Steps	5	10	15
DDSS	55.14 / 12.9	37.32 /14.8	24.69 /17.2
Ours	46.83 / 11.4	26.12 / 15.1	23.29 / 14.8

Table 5. FID score / IS score for our method against DDSS for the DDPM trained on ImageNet 64×64 with L_{simple} [24]

Approach	Steps	Method Type	Total Cost (GPU days)
AutoDiffusion	5	Training-free Search	1.125
DDSS	5	Reparameterization	3.55
Progressive Distil.(PD)	4	Distillation	359
Progressive Distil.(PD)	8	Distillation	314

Table 6. Efficiency comparison. We assessed the computational resource demand of AutoDiffusion, PD, and DDSS using our reconstructed Improved-Diffusion codebase and ImageNet 64×64 on a single V100 GPU. For DDSS, we approximated the computational resource consumption by running 50k training steps of U-Net and multiplying the training time by the time steps, as it executes the entire generation process in each training step.

4.6. The efficiency of AutoDiffusion

AutoDiffusion is highly efficient and surpasses existing methods that demand additional computational resources such as PD [34] and DDSS [40] in computational resource requirements. AutoDiffusion uses a training-free search to determine time steps and diffusion models architecture, with search time depending on image resolution, time step length, and model size. Tab. 6 demonstrates the superior efficiency of AutoDiffusion compared to DDSS and PD. The computational resource required by DDSS and PD is approximately $3.15\times$ and $279\times$ that of AutoDiffusion.

5. Conclusion

In this paper, we propose AutoDiffusion to search the optimal time steps and architectures for any pre-trained diffusion models. We design a unified search space for both time steps and architectures, and then utilize the FID score as the evaluation metric for candidate models. We implement the evolutionary algorithm as the search strategy for the AutoDiffusion framework. Extensive experiments demonstrate that AutoDiffusion can search for the optimal time steps sequence and architecture with any given number of time steps efficiently. Designing more sophisticated methods that can evaluate the performance of diffusion models faster than FID score can improve the search speed and performance of AutoDiffusion, which we leave as future work.

Acknwoledgement. This work was supported by National Key R&D Program of China (No. 2022ZD0118202), the National Science Fund for Distinguished Young Scholars (No.62025603), the National Natural Science Foundation of China (No. U21B2037, No. U22B2051, No. 62176222, No. 62176223, No. 62176226, No. 62072386, No. 62072387, No. 62072389, No. 62002305 and No. 62272401), and the Natural Science Foundation of Fujian Province of China (No. 2021J01002, No. 2022J06001).

References

[1] Metin Ersin Arican, Ozgur Kara, Gustav Bredell, and Ender Konukoglu. Isnas-dip: Image-specific neural architecture search for deep image prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1960–1968, 2022.

[2] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022.

[3] David Berthelot, Arnaud Autef, Jierui Lin, Dian Ang Yap, Shuangfei Zhai, Siyuan Hu, Daniel Zheng, Walter Talbot, and Eric Gu. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

[4] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14347–14356, 2021.

[5] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11472–11481, 2022.

[6] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12413–12422, 2022.

[7] Kamil Deja, Anna Kuzina, Tomasz Trzcinski, and Jakub M. Tomczak. On analyzing generative and denoising capabilities of diffusion-based deep generative models. *CoRR*, abs/2206.00070, 2022.

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[9] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

[10] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, pages 10686–10696, 2022.

[11] Yu-Chao Gu, Shang-Hua Gao, Xu-Sheng Cao, Peng Du, Shao-Ping Lu, and Ming-Ming Cheng. Inas: integral nas for device-aware salient object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4934–4944, 2021.

[12] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 544–560. Springer, 2020.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[15] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23:47:1–47:33, 2022.

[16] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8107–8116. Computer Vision Foundation / IEEE, 2020.

[17] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[18] Changlin Li, Tao Tang, Guangrun Wang, Jiefeng Peng, Bing Wang, Xiaodan Liang, and Xiaojun Chang. Bossnas: Exploring hybrid cnn-transformers with block-wisely self-supervised neural architecture search. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12261–12271, 2021.

[19] Haojia Lin, Lijiang Li, Xiawu Zheng, Fei Chao, and Rongrong Ji. Searching lightweight neural network for image signal processing. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2825–2833, 2022.

[20] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[21] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[22] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[23] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[24] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[25] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning, ICML*, volume 162, pages 16784–16804, 2022.

[26] Jiefeng Peng, Jiqi Zhang, Changlin Li, Guangrun Wang, Xiaodan Liang, and Liang Lin. Pi-nas: Improving neural architecture search by reducing supernet training consistency shift. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12334–12344, 2021.

[27] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[28] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[31] Chitwan Saharia, William Chan, Huiwen Chang, Chris A. Lee, Jonathan Ho, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference*, pages 15:1–15:10. ACM, 2022.

[32] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, 2022.

[33] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[34] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[36] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[37] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428, 2021.

[38] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[39] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*, 2022.

[40] Daniel Watson, William Chan, Jonathan Ho, and Mohammad Norouzi. Learning fast samplers for diffusion models by differentiating through sample quality. In *International Conference on Learning Representations*, 2022.

[41] Daniel Watson, William Chan, Jonathan Ho, and Mohammad Norouzi. Learning fast samplers for diffusion models by differentiating through sample quality. In *International Conference on Learning Representations*, 2022.

[42] Hang Xu, Lewei Yao, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Auto-fpn: Automatic network architecture adaptation for object detection beyond classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6649–6658, 2019.

[43] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[44] Xiawu Zheng, Rongrong Ji, Qiang Wang, Qixiang Ye, Zhenguo Li, Yonghong Tian, and Qi Tian. Rethinking performance estimation in neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11356–11365, 2020.