TF-MAS: Training-free Mamba2 Architecture Search

Yi Fan
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing 210023, China
fanyiplus@smail.nju.edu.cn

Yu-Bin Yang*
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing 210023, China
yangyubin@nju.edu.cn

Abstract

The Mamba-type neural networks have gained significant popularity recently. To effectively and efficiently establish model architectures of Mamba, it is natural to introduce Neural Architecture Search (NAS) methods into Mamba. However, existing NAS methods tailored for Mamba are training-based, leading to substantial time and computational resource expenditure. To address this issue, and considering that Mamba2 is an improved version of the original Mamba, we propose a training-free NAS method specifically designed for Mamba2. Based on rank collapse in stacked State Space Duality (SSD) blocks, we design a proxy that only requires the computation of the transformation matrix and its gradient between two tensors within the network. Additionally, we develop a corresponding search space and introduce a novel approach for determining adjustable hyperparameter ranges. Experimental results show that our method outperforms all existing training-free NAS approaches in terms of both ranking correlation and the performance of search results for Mamba2 architecture. To the best of our knowledge, this is the first training-free NAS method designed for Mamba-type architectures. Our codes are available at https://github.com/fanyi-plus/tf-nas.

1 Introduction

Recently, the introduction of Mamba has provided new impetus for research in State Space Models (SSM) [24], and Mamba-type models possess potential to partially replace Transformer [61, 64, 29]. Subsequently, Mamba2 [13], developed as an enhancement of the original Mamba, further improves network performance. However, similar to other types of networks, the practical use of Mamba-type networks still necessitates the manual specification of hyperparameters to determine the actual model architecture [14]. This process often involves trial-and-error, which can be time-consuming and labor-intensive. Therefore, it is logical to apply Neural Architecture Search (NAS) techniques [52] to the design of Mamba-type architectures. Currently, among the limited work, [50] explores hyperparameters such as network depth and width, while [27, 19] search for scanning orders of image patches for vision-oriented Mamba.

However, these methods rely heavily on extensive training of the networks during the search process. In fact, recent research in the NAS is increasingly leaning towards training-free NAS [35], also known

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

as zero-shot NAS. These approaches avoid the need to train candidate networks and instead compute a proxy relying only on information from the parameters, gradients, latent variables, etc., of the network in its initial state. The proxy is correlated with the actual performance of the network. The earliest training-free NAS methods target convolutional networks [31, 49, 28], followed by methods designed for Transformers [68, 54, 20]. Among these, DSS [68] is a classic method that originates from a finding suggesting that when stacking increasing attention layers in a model, the rank of the output trends towards 1 [18]. Based on this, DSS proposes an indirect measurement of the degree to which the rank approaches 1 as a representation of candidate network performance. Later adaptations of the method [69] introduce layer-wise proxies. Unfortunately, no training-free NAS methods for Mamba-type networks have yet emerged.

To fill this gap in training-free NAS for Mamba-type networks, we propose TF-MAS. Given that the overall performance of typical Mamba-type model, Mamba2 [13], exceeds that of raw Mamba, our search space is designed based on Mamba2. By theoretically demonstrating that when the core structure, State Space Duality (SSD), in Mamba2 is stacked continuously, its output also trends towards 1. Inspired by DSS, we design a training-free NAS method for Mamba2. In this method, we need to compute the transformation matrix between the input tensor of each Mamba2 block and the input tensor of the corresponding SSD. When there are different relationships between the sequence length and feature dimensions of the input tensor of the Mamba2 block, we devise distinct methods to derive the transformation matrices. Once the transformation matrices are obtained, the values of the proxies can be computed. Additionally, we design a search space that includes 4 Adjustable Hyperparameters (AHs). Unlike existing methods, the ranges of these AHs are not artificially set but are determined through precise calculations based on the expected cost for specific tasks. This increases the likelihood of discovering candidate networks with superior performance within the search space.

We construct several NASBenches based on existing pre-trained Mamba2 models, with each dataset containing 500 network architectures sampled from the specific search space along with their corresponding performance metrics on different datasets. We compute the ranking correlation between our proxies and actual performance on these NASBenches. The results indicate that our method outperforms all other training-free NAS approaches. Subsequently, under the overhead constraint, we establish the search space, conduct searches, and train the search results from scratch. The search results exceed the performance of the original Mamba2 while maintaining computational costs not greater than those of the original networks. These results demonstrate the unique effectiveness of our method in searching for Mamba2 architectures. Furthermore, we conduct additional experiments to investigate rank convergence, transformation matrices, and search spaces to validate the rationality and robustness of our method.

We give an brief introduction about the related work and a discussion about societal impacts in Section A and B of the technical appendix, respectively. The contributions of this paper are as follows:

- We introduce, for the first time, a training-free NAS method for the Mamba-type networks.
- We design a search space where the range of AHs is dictated by computational costs.
- We validate the effectiveness of our method through ranking correlation and the performance of search results.

2 Premilinary

A simplified Mamba2 model is illustrated in Figure 1 (excluding the red areas). It consists of several Mamba2 cells arranged in series, with each containing a Mamba2 block. Within the Mamba2 block, the input $\boldsymbol{U}$ with dimension $\mathsf{T} \times \mathsf{W}$ is replicated into three copies, which are processed through fully connected layers, convolutions, and the SiLU activation function, resulting in three variables: $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$, with dimensions $\mathsf{T} \times \mathsf{P}$, $\mathsf{T} \times \mathsf{N}$, and $\mathsf{T} \times \mathsf{N}$, respectively. These three matrices are then fed into the SSD module, whose computation is defined as follows:

$$\boldsymbol{Y} = \left(\boldsymbol{L} \odot \left(\boldsymbol{C}\boldsymbol{B}^{\top}\right)\right)\boldsymbol{X}, \tag{1}$$

where $\boldsymbol{L}$ is computed from a parameter of the network, and $\boldsymbol{Y}$ is the output. Conceptually, the meaning of W, T, P, and N are the width of network (the dimension of each token), sequence length

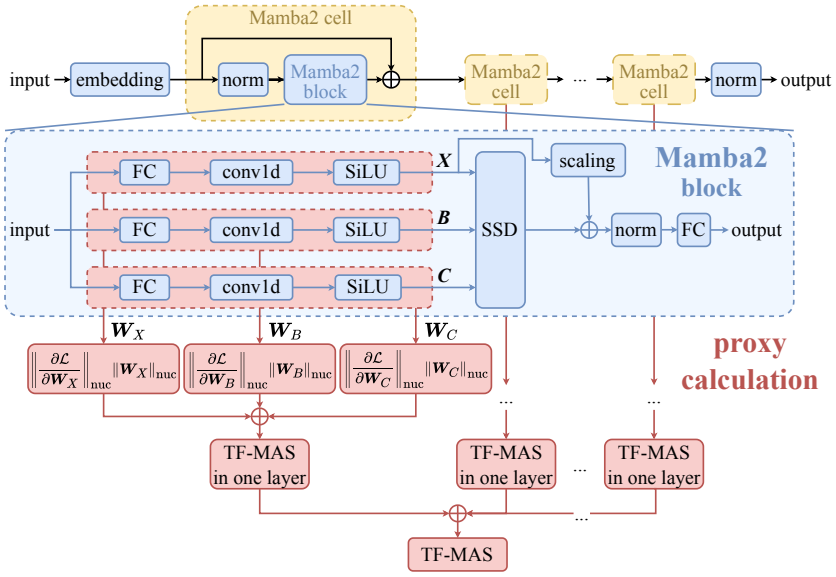


Figure 1: Structure of Mamba2 and the corresponding TF-MAS method. To emphasize the key points, we focus primarily on the components related to the computation of the proxy in the Mamba2 structure, while omitting variables such as $\boldsymbol{Z}$ and $\mathbf{dt}$.

(token number), feature dimension in SSD, and state space size, respectively. Similar to transformer architectures, the SSD module in Mamba2 consists of multiple heads.

[13] indicates that the SSD module shares the same computational pattern as linear attention, where the inputs $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ correspond to the value, key, and query in linear attention, respectively.

3 Method

3.1 Proxy design

[18] has been shown that if multiple attention modules are concatenated without the inclusion of regularization layers, the rank of the output during forward propagation will converge to 1 as the number of attention modules approaches infinity, a phenomenon known as rank collapse. Furthermore, [13] points out that the forward process of the attention module is essentially consistent with that of the SSD module, differing only in the representation of each variable. Therefore, we can reason that stacking multiple SSD modules will also result in rank collapse. A theoretical proof is provided in Subsection C.1 of the technical appendix, and we also verify this by experiments in Subsection 4.4. This phenomenon will affect the expressiveness of the network. This can be regarded as a reason why the Mamba2 network necessitates the use of regularization layers (see Figure 1). However, as described in [18], regularization layers can only partially mitigate rank collapse and cannot entirely eliminate it. Therefore, inspired by the evaluation method for Multi-head Self-Attention (MSA) in ZeroLM [69], we consider designing a new proxy to evaluate the performance of different Mamba2 models by measuring the rank collapse magnitude.

ZeroLM states that rank collapse magnitude of outputs can be measured by calculating the rank of the weights after network initialization. To reduce the computational overhead of the proxy, further theoretical analysis has transformed this into computing the product of the nuclear norm of the weights and the nuclear norm of the gradients of the weights. In attention module, the weights are typically the transformation matrices from input to query/key/value via a fully connected layer, while in Mamba2 block, the operations from $\boldsymbol{U}$ to $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ are not merely a fully connected layer; they also involve convolution and SiLU. Here, we represent the transformation matrices from $\boldsymbol{U}$ to $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ in Mamba2 block as $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$, thus the sizes are $\mathsf{W} \times \mathsf{P}$, $\mathsf{W} \times \mathsf{N}$, and $\mathsf{W} \times \mathsf{N}$, respectively. Due to the existence of convolution and SiLU, $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$ do not equal to the weight of the fully connected layer. Consequently, to compute the score of the network,

we first need to determine $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$. Below, we illustrate the calculation of $\boldsymbol{W}_X$ as an example. We have

$$\boldsymbol{U}\boldsymbol{W}_X = \boldsymbol{X}. \tag{2}$$

The solution of Equation (2) depends on the shape of $\boldsymbol{U}$, i.e, the relationship between T and W, and for different cases, we will adopt different measures to determine $\boldsymbol{W}_X$.

Case 1: $\mathsf{T} = \mathsf{W}$**.** In the absence of prior knowledge about the input data, we can assume that the elements in $\boldsymbol{U}$ are independent and follow a normal distribution; the reasonableness of this assumption is elaborated in Subsection C.2 of the technical appendix. It can be proven that, in this case, the probability of the matrix being full rank is 1 (the proof is provided in Subsection C.3 of the technical appendix). Therefore, we can directly compute $\boldsymbol{W}_X = \boldsymbol{U}^{-1}\boldsymbol{X}$.

Case 2: $\mathsf{T} < \mathsf{W}$**.** It can be shown that the probability of each column of $\boldsymbol{X}$ lying within the column space of $\boldsymbol{U}$ is 1 (the proof is provided in Subsection C.4 of the technical appendix), leading to the conclusion that Equation (2) has infinitely many solutions. To minimize the structural risk of the model, we seek to find $\boldsymbol{W}_X$ that minimizes the 2-norm, i.e., solve

$$\underset{\boldsymbol{W}_X}{\operatorname{argmin}} \left\|\boldsymbol{W}_X\right\|_2, \quad \text{s.t.} \quad \boldsymbol{U}\boldsymbol{W}_X = \boldsymbol{X}. \tag{3}$$

We employ the least squares method for this. First, we perform Singular Value Decomposition (SVD) on $\boldsymbol{U}$, obtaining $\boldsymbol{U} = \boldsymbol{V}_1\boldsymbol{\sigma}\boldsymbol{V}_2^\top$, where $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ are orthogonal matrices, and $\boldsymbol{\sigma}$ is a diagonal matrix. Based on this, the Moore-Penrose pseudoinverse of $\boldsymbol{U}$ can be computed as $\boldsymbol{U}^+ = \boldsymbol{V}_2\boldsymbol{\sigma}^+\boldsymbol{V}_1^\top$, where $\boldsymbol{\sigma}^+$ is the pseudoinverse of $\boldsymbol{\sigma}$. We ultimately obtain

$$\underset{\boldsymbol{W}_X}{\operatorname{argmin}} \left\|\boldsymbol{W}_X\right\|_2 = \boldsymbol{U}^+\boldsymbol{X}. \tag{4}$$

Case 3: $\mathsf{T} > \mathsf{W}$**.** In contrast to Case 2, the probability of each column of $\boldsymbol{X}$ lying within the column space of $\boldsymbol{U}$ is 0 (the proof is provided in Subsection C.5 of the technical appendix), leading to the conclusion that Equation (2) has no solutions. Thus, it is impossible to find a $\boldsymbol{W}_X$ that accurately substitutes for the mapping from $\boldsymbol{U}$ to $\boldsymbol{X}$. Therefore, we seek to find an approximate $\boldsymbol{W}_X$ such that the result of the linear mapping is as close as possible to $\boldsymbol{X}$. That is, we solve

$$\underset{\boldsymbol{W}_X}{\operatorname{argmin}} \left\|\boldsymbol{U}\boldsymbol{W}_X - \boldsymbol{X}\right\|_F. \tag{5}$$

We again utilize the least squares method, yielding the calculation process consistent with Case 2. Considering that $\boldsymbol{W}_X$ is obtained via the pseudoinverse approximation, we quantify how this error propagates and affects proxy reliability in Subsection C.6 of the technical appendix.

Now we finish the calculation of $\boldsymbol{W}_X$. The calculations for $\boldsymbol{W}_B$ and $\boldsymbol{W}_C$ follow a similar process, except that all instances of P in the matrix sizes are replaced with N. It can be observed that the solutions for $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$ essentially resolve a set of matrix equations of the form $\boldsymbol{A}\boldsymbol{W} = \boldsymbol{B}$. To speed up the computation, we can concatenate all equations where $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same dimensions along the batch size dimension before computation, making full use of the parallel acceleration advantages offered by devices.

After obtaining $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$ for each Mamba2 block, we can compute the network score using the method described in [69]. However, unlike the approach in [69], these weights are not parameters of the network, and thus their gradients cannot be directly obtained through backpropagation. Nevertheless, via chain rule and matrix calculus, we can easily calculate their values by $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_X} = \boldsymbol{U}^\top \frac{\partial \mathcal{L}}{\partial \boldsymbol{X}}$, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_B} = \boldsymbol{U}^\top \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}}$, and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_C} = \boldsymbol{U}^\top \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}}$. Besides, there are also weights, $\boldsymbol{W}_{\text{out}}$, in output layer of Mamba2 blocks, which functions similarly to the subsequent linear layer in attention modules; thus, it participates in the score calculation as well. Specifically, the calculation formula for our proxy, TF-MAS, is

$$\text{TF-MAS} = \sum_{i=1}^{l} \left(\sum_{x \in \{X,B,C,\text{out}\}} \left\| \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_x} \right\|_{\text{nuc}} \left\|\boldsymbol{W}_x\right\|_{\text{nuc}} \right). \tag{6}$$

It should be pointed out that although multiplying parameter-related terms and gradient-related terms has already been applied in existing methods [69, 38], none of these methods have provided

theoretical proof. We believe that multiplication can amplify or suppress certain characteristics based on the magnitude of parameters and gradients. When both parameters and gradients are large, their product significantly increases, thereby highlighting this important information in the evaluation metric. This enables architecture search to prioritize network architectures with larger parameters and gradients, as these architectures may have stronger learning ability and adaptability during training.

We presents the pseudocode of our proxy computation process in Section D of the technical appendix.

3.2 Search space

The design of the search space generally follows the design principles outlined in [10]. Specifically, we have set the following 4 AHs:

- Depth (D): the number of Mamba2 blocks.
- Width (W): the dimension of each token.
- State dimension (N): the dimensionality of the state space.
- Number of heads (H): the number of heads in each Mamba2 block.

All 4 AHs are positive integers. We set the dimension of each head to 64, which is consistent with the original Mamba2. This indicates that the number of columns in $\boldsymbol{X}$ is $64\mathsf{H}$. We denote this search space as SSMamba2 (Search Space of Mamba2). Moreover, to enhance the flexibility of the network architecture, and inspired by [10], we extend SSMamba2 to form VWSSMamba2 (Variable Width Search Space of Mamba2). The candidate networks within this search space differ from existing Mamba2; thus, we refer to them as VWMamba2 (Variable Width of Mamba2). This design permits the values of N and H to differ across layers, meaning that for a specific network, N and H are no longer solely positive integers but rather D-dimensional vectors, where the i-th element represents the values of N and H in the i-th layer. It should be noted that our work follows the same protocol as [18] and uniformly employs skip connections in all candidate networks.

The range of values for each AH is determined by the expected computational cost. Suppose the expected computational overhead is provided by an upper limit of parameter number, denoted by c_0. When constructing SSMamba2, we will reference an existing Mamba2 model, referred to as the reference model[2], to construct a virtual benchmark model. This benchmark model must satisfy two conditions: (a) According to the scaling rule, we want this benchmark model to be a proportional scaling of the reference model. (b) The capacity of this benchmark model must equal c_0.

Let the AHs of the reference model be D_r, W_r, N_r, and H_r, and the AHs of the benchmark model be D_b, W_b, N_b, and H_b. According to condition (a), the equations $\mathsf{D}_b = k\mathsf{D}_r$, $\mathsf{W}_b = k\mathsf{W}_r$, $\mathsf{N}_b = k\mathsf{N}_r$, and $\mathsf{H}_b = k\mathsf{H}_r$ (where k is a positive constant) should be satisfied. Besides, the parameter number of an existing Mamba2 model can be computed as $\mathsf{W} + 2\mathsf{VW} + \mathsf{DW} + 10\mathsf{DN} + 387\mathsf{DH} + 2\mathsf{DWN} + 193\mathsf{DWH}$ (calculation details can be found in Section E of the technical appendix), where V is the dictionary length, a predetermined value. According to condition (b), we have $\mathsf{W}_b + 2\mathsf{VW}_b + \mathsf{D}_b\mathsf{W}_b + 10\mathsf{D}_b\mathsf{N}_b + 387\mathsf{D}_b\mathsf{H}_b + 2\mathsf{D}_b\mathsf{W}_b\mathsf{N}_b + 193\mathsf{D}_b\mathsf{W}_b\mathsf{H}_b = c_0$. By substituting $\mathsf{D}_b = k\mathsf{D}_r$, $\mathsf{W}_b = k\mathsf{W}_r$, $\mathsf{N}_b = k\mathsf{N}_r$, and $\mathsf{H}_b = k\mathsf{H}_r$ into the equation and rearranging, we obtain a cubic equation in terms of k,

$$\left(2\mathsf{D}_r\mathsf{W}_r\mathsf{N}_r + 193\mathsf{D}_r\mathsf{W}_r\mathsf{H}_r\right)k^3 + \left(2\mathsf{VW}_r + \mathsf{D}_r\mathsf{W}_r + 387\mathsf{D}_r\mathsf{H}_r + 10\mathsf{D}_r\mathsf{N}_r\right)k^2 + \mathsf{W}_r k = c_0. \quad (7)$$

This equation can be solved using the quadratic formula (it is clear that the equation has a unique solution). Once k is obtained, the AHs of the benchmark model can be determined. It is important to note that the AHs determined using this method are likely not integers; thus, we refer to this benchmark model as *virtual*. The benchmark model does not need to be physically constructed; it merely serves as a reference for determining the range of AHs. Ultimately, we will take $\lceil 0.6 \times \frac{x_b}{I_x} \rceil \cdot I_x$ and $\lfloor 1.6 \times \frac{x_b}{I_x} \rfloor \cdot I_x$ as the minimum and maximum values for the AHs, respectively, where the symbol x can represent D, W, N, or H, and I_x is a positive integer. In colloquial terms, this means that each AH can take on all values between 0.6 times and 1.6 times the corresponding benchmark model AH. The choice of the limits are based on previous design experiences of search spaces [10, 51].

[2]Researchers or engineers proposing a new Mamba2-like model architecture typically define a specific model structure for training and testing, which can serve as a reference model. Thus, we believe a suitable reference model is usually available.

Table 1: Ranking correlation on VMSSMamba2Bench_2.7B. The names of the datasets LD, HS, OBQA, WG are abbreviations for LAMBADA, HellaSwag, OpenbookQA, and WinoGrande, respectively. The names of the proxies #Param, AC, HI, HC are abbreviations for Parameter Number, Attention Confidence, Head Importance, Head Confidence, respectively. For target, G, C, T, M means general, CNN, Transformer, Mamba, respectively. Best results in bold.

Proxy	Target	LD (PPL)	LD (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)	Time (s)
#Param	G	-0.458	0.437	0.214	0.310	0.246	0.323	0.378	0.325	**0.07**
Gradnorm	C	-0.091	0.017	0.043	0.036	0.080	0.027	0.075	0.094	0.96
Synflow [58]	C	-0.040	0.033	-0.018	0.066	0.061	0.112	0.013	0.029	1.47
GraSP [60]	C	-0.064	-0.004	0.044	0.030	0.024	0.042	0.013	0.030	1.54
Fisher [40]	C	-0.081	0.004	0.056	0.052	0.035	0.056	0.079	0.021	1.77
Snip [34]	C	0.024	0.061	-0.007	0.119	0.009	0.029	-0.013	0.035	0.89
Zen-NAS [38]	C	-0.038	0.046	0.056	0.098	0.013	0.017	-0.006	-0.013	0.75
ZiCo [36]	C	-0.093	0.066	0.061	0.020	0.022	0.052	0.055	0.070	0.71
MeCo [31]	C	-0.011	0.057	-0.028	0.038	0.061	0.047	0.061	0.061	0.86
Auto-Prox [62]	C	-0.034	0.047	0.016	0.042	0.050	0.069	0.029	0.064	1.09
AC [54]	T	-0.130	0.112	0.110	0.114	0.081	0.047	0.049	0.108	0.91
HI [54]	T	-0.087	0.082	0.069	0.089	0.045	0.113	0.050	0.092	0.97
HC [54]	T	-0.117	0.146	0.102	0.111	0.034	0.078	0.095	0.123	0.90
DSS++ [69]	T	-0.118	0.070	0.108	0.087	0.043	0.085	0.091	0.089	2.18
AttnNAS [20]	T	-0.246	0.267	0.159	0.183	0.122	0.144	0.232	0.228	1.63
TF-MAS (ours)	M	**-0.685**	**0.661**	**0.381**	**0.468**	**0.339**	**0.452**	**0.483**	**0.519**	1.31

Additionally, within the search space, we must ensure that the values are multiples of I_x. Considering the scales of each AH, we set $I_{\mathsf{D}} = 1$, $I_{\mathsf{W}} = 32$, $I_{\mathsf{N}} = 8$, and $I_{\mathsf{H}} = 1$.

If the expected computational overhead is provided in terms of FLOPs or inference time, it becomes challenging to accurately determine the benchmark model using the aforementioned method. To estimate the rough range of AHs, we approximately assume that both FLOPs and inference time are proportional to the number of parameters.

Upon completing the construction of SSMamba2, we maintain the limits for each AH while extending the values of N and H from integers to D-dimensional vectors, resulting in VWSSMamba2.

Finally, a question worth investigating is how the time complexity of the proxy varies with different AHs. Encouragingly, we derive that when any of W, N, H, P approaches infinity, the time complexity grows linearly. A detailed analysis is provided in Subsection C.7 of the technical appendix. This demonstrates the practicality of our method.

4 Experiments

4.1 Ranking correlation

The effectiveness of a proxy is most directly measured by ranking correlation. This involves sampling several network architectures from the search space, training them, testing their actual performance, and calculating their proxy values. The ranking correlation is just the correlation between the proxy values and the performance of the sampled architectures. A higher ranking correlation indicates that the proxy can more accurately estimate the performance of the network.

To save computation time, some researchers have stored sampled architectures and their corresponding performances to create datasets known as NASBench [43]. To our knowledge, current NASBenches encompass only CNN [65, 17, 57, 16] and Transformer [62], with no one specific to Mamba. Therefore, we will sample architectures from both SSMamba2 and VWSSMamba2 to construct NASBenches pertaining to Mamba. However, training numerous Mamba2 networks comparable to the model sizes reported in [13] from scratch incurs a significant time cost. To mitigate this, we employ weight entanglement proposed in [10]. After sampling the architectures, we directly activate the corresponding part of parameters from Mamba2 pretrained models as the parameters of the sampled architectures. Nevertheless, we cannot guarantee that weight entanglement will function effectively under Mamba2. Therefore, for each sampled network, we perform 10 epoches of

Table 2: Test performance of search results on SSMamba2Bench_130M and VMSS-Mamba2Bench_130M. The abbreviations in datasets are consistent with Table 1. Best results in bold.

Proxy	Model Size (M)	FLOPs (G)	latency (s)	LD (PPL↓)	LD (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
Mamba-2-130M	128.932	8.249	0.160	16.79	45.9	31.0	65.0	47.4	20.9	52.6	18.0
opt Mamba2	127.130	8.133	0.171	15.31	49.5	35.8	66.4	51.7	25.1	53.9	19.1
opt VWMamba2	119.104	7.620	0.151	15.20	50.7	36.9	66.4	51.8	**25.2**	54.7	19.4
opt VWMamba2 w/ bwe	122.572	7.842	0.125	**15.08**	**51.5**	**40.5**	**66.9**	**52.6**	**25.2**	**55.2**	**19.5**

fine-tuning using the Pile dataset. Importantly, weight entanglement primarily injects noise into the observed performance values, and additive noise tends to reduce the Kendall's Tau (as formally proved in Subsection C.8 of the technical appendix). Consequently, the empirical correlations reported here should be regarded as conservative lower bounds; the actual ranking agreement in a noise-free setting is expected to be even higher, indicating that the use of weight entanglement is indeed acceptable.

Currently, there are five publicly available pretrained Mamba2 models with parameter number of 130M, 370M, 780M, 1.3B, and 2.7B. We generate a NASBench for each model on SSMamba2 and VWSSMamba2 with 500 architectures, resulting in a total of 10 NASBenches, which we denote as SSMamba2Bench_X and VMSSMamba2Bench_X (where "X" represents the parameter number). When constructing a NASBench, we set the range of the AHs to be between 0.5 to 1 times the corresponding AHs in the pre-trained model. We evaluate accuracy (where higher is better) on datasets LAMBADA [46], HellaSwag [66], PIQA [7], Arc-E [8], Arc-C [8], WinoGrande [53], and OpenbookQA [45], while for LAMBADA, we also assess perplexity (where lower is better).

Table 1 presents the ranking correlation between various existing proxies and performance metrics on VMSSMamba2Bench_2.7B. Kendall's Tau is used as correlation in this paper. Section F of the technical appendix includes results for other NASBenches. We compare our method with other existing proxies, including the parameter number, which is often treated as a baseline [35]. Given that our method is the first known training-free NAS targeting the Mamba-type models, all other methods are oriented towards CNN or Transformer. It is worth noting that for attention map-based proxies such as Attention Confidence [54] and AttnNAS [20], we consider $\boldsymbol{C}\boldsymbol{B}^\top$ as the attention map, motivated by the duality between attention and SSD. Additionally, some classical proxies require specific operations or structures within the network (e.g., the CNN-oriented NASWOT [44] requires ReLU), which are absent in Mamba2; therefore, we do not include comparisons with these proxies.

It is apparent that the ranking correlation of existing methods is generally lower than the baseline, with only our method surpassing the baseline. This indicates that while existing methods have played a significant role in predicting the performance of CNNs and Transformers, they struggle to adapt to Mamba2, showcasing the value of our method. Furthermore, we observe an interesting phenomenon where Transformer-oriented methods (albeit not as effective as our proposed method) perform slightly better than CNN-oriented methods. We speculate that this may be due to certain structural similarities between Transformers and Mamba2, which provides inspiration for future improvements in proxies targeting Mamba2, suggesting an exploration of insights from existing Transformer-based proxies.

Moreover, recent studies have highlighted that ranking correlation evaluated only on the top-k candidates can be particularly informative in certain scenarios [67, 15]. We therefore computed the corresponding metrics and obtained results consistent with those in Table 1 (see Section F of the technical appendix), demonstrating the robustness of our proxy.

4.2 Performance of search results

While our method demonstrates better performance than other baselines in the previous subsection, the rank correlation coefficient is not sufficiently high in some datasets for the proxy to be considered a fully reliable metric. So validating the feasibility in practical search scenarios is important. In this subsection, we utilize the proposed proxy to conduct evolutionary searches on SSMamba2 and VWSSMamba2. After obtaining the search results, we perform training and testing from scratch. Due to limited computational resources, we use the Mamba2 model with 130M parameters as the reference model, setting the expected computational overhead to "parameter number not exceeding

Table 3: Ranking correlation on SSMamba2Bench_130M and VMSSMamba2Bench_130M when using and not using the weights of the fully connected layer as $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, and $\boldsymbol{W}_C$. The abbreviations in datasets are consistent with Table 1, and "wfc" means weights of the fully connected layer.

Method	LAMBADA (PPL↓)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
SSMamba2 w/o wfc	0.677	0.648	0.406	0.451	0.364	0.435	0.517	0.508
SSVWMamba2 w/o	0.675	0.667	0.357	0.450	0.329	0.467	0.485	0.556
SSMamba2 w/ wfc	0.164	0.186	0.121	0.146	0.131	0.147	0.132	0.156
SSVWMamba2 w/ wfc	0.192	0.181	0.132	0.141	0.111	0.162	0.134	0.171

130M," which aligns with the existing pretrained models for comparison purposes. Clearly, under these conditions, the solution to Equation (7) is $k = 1$, and the benchmark model is just the reference model. The set of values for each AH and the details about the evolutionary method are demonstrated in Section G of the technical appendix.

Our search is conducted on 4 NVIDIA Tesla V100 GPUs, with search times on SSMamba2 and VWSSMamba2 being 0.7 day and 0.6 day, respectively. Table 2 showcases the testing results. It can be observed that our search results, referred to as opt Mamba2 and opt VWMamba2 (where "opt" stands for optimized), demonstrate higher performance compared to the existing Mamba2 model while maintaining slightly lower parameter number, FLOPs, and inference latencies than the 130M model. This suggests that our method is capable of effectively optimizing the model architecture. Notably, the search results for VWSSMamba2 show a slight performance improvement over those for SSMamba2, indicating that making the widths of the layers variable increases the flexibility of the architecture, which is beneficial for constructing better models.

Additionally, inspired by the block-wise evolution described in [69], we repeat the experiments for VWSSMamba2 using an alternative evolutionary method. In this method, instead of calculating the proxy for the entire network at once, we compute the proxy layer by layer (see Section G of the technical appendix for more details). The performance of the VWMamba2 model obtained using this evolutionary method, referred to as opt VWMamba2 w/ bwe (where "w/ bwe" stands for "with block-wise evolution"), is also listed in Table 2. Compared to opt VWMamba2, opt VWMamba2 w/ bwe exhibits a further slight increase in performance. This not only further validates the effectiveness of block-wise evolution as proposed in [69], but also demonstrates that our method continues to be effective at the finer granularity of blocks. In Section H of the technical appendix, we provide the architectures of the search results.

4.3 Ablation study

Effects of transformation matrices. In Subsection 3.1, we indicated that in the calculation of the proxy, $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, $\boldsymbol{W}_C$, should be regarded as the transformation matrix from $\boldsymbol{U}$ to $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$, including convolution and SiLU, rather than the weights of the fully connected layer with $\boldsymbol{U}$ as input. To validate this statement, we replace $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, $\boldsymbol{W}_C$ in the proxy with the weights of the fully connected layer taking $\boldsymbol{U}$ as input and recalculate the correlation on SSMamba2Bench_130M and VMSSMamba2Bench_130M. The results are presented in Table 3. It is evident that this substitution results in a significant drop in correlation for both NASBenches, confirming the appropriateness of our computation method for $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, $\boldsymbol{W}_C$.

Effects of each AH. In Subsection 3.2, we introduce 4 AHs in the search space. One question arises: can our method detect the impact of each AH's variation on network performance? In other words, when utilizing the proxy, are all 4 introduced AHs necessary? To address this, we calculate the correlation while varying only one AH at a time. Specifically, we sample candidate networks using the 130M pretrained model in the search spaces of both SSMamba2 and VWSSMamba2. When investigating the necessity of a specific AH, all sampled candidate networks have only that AH differing, while the values of the other 3 AHs remain consistent with those of the pretrained model. We then compute the ranking correlation for these candidate networks, with results shown in Table 4. Notably, in both search spaces, the ranking correlations for all four AHs are not near zero, indicating that the answer to this question is affirmative.

Table 4: Ranking correlation on SSMamba2 and VWSSMamba2 with only one AH varying. The abbreviations in datasets are consistent with Table 1.

Search space	AH	LAMBADA (PPL↓)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
SSMamba2	D	0.340	0.314	0.191	0.206	0.219	0.226	0.257	0.257
	W	0.410	0.399	0.276	0.324	0.190	0.256	0.365	0.377
	N	0.128	0.128	0.048	0.069	0.076	0.041	0.092	-0.002
	H	0.301	0.217	0.105	0.166	0.156	0.169	0.224	0.193
VWSSMamba2	D	0.302	0.304	0.170	0.241	0.161	0.214	0.281	0.259
	W	0.425	0.427	0.266	0.321	0.201	0.339	0.304	0.331
	N	0.092	0.065	0.016	0.102	0.093	0.046	0.083	0.100
	H	0.227	0.231	0.158	0.267	0.141	0.206	0.214	0.253

Table 5: Ranking correlation on SSMamba2 and VWSSMamba2 when confronting case 2 and 3 in Subsection 3.1. The abbreviations in datasets are consistent with Table 1.

Search space	Case	LAMBADA (PPL↓)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
SSMamba2	1	0.566	0.571	0.321	0.379	0.291	0.404	0.405	0.477
	3	0.476	0.485	0.291	0.323	0.302	0.321	0.394	0.437
VWSSMamba2	1	0.551	0.568	0.354	0.398	0.337	0.430	0.460	0.479
	3	0.478	0.458	0.266	0.356	0.308	0.331	0.346	0.386

4.4 Discussion

Rank collapse in stacked SSD blocks. To valid the rank collapse in stacked SSD model, we construct a stacked SSD blocks with 12 layers. After initialization, we examin the rank collapse metrics (see Section I of the technical appendix for the details about the metric) of the input, along with the output of each layer. Additionally, we include stacked attention block, Transformer, and Mamba2 of the same depth for comparison. The results are illustrated in Figure 2. We observe that as the depth of the network increases, this distance consistently decreases, though at a slower rate in stacked SSD. This indicates that similar to stacked attention blocks, there is a rank collapse phenomenon occurring in stacked SSD blocks, albeit less pronounced than in stacked attention blocks. This validates the theoretical foundation of the proposed proxy.

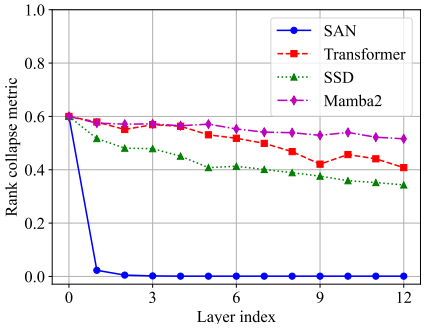
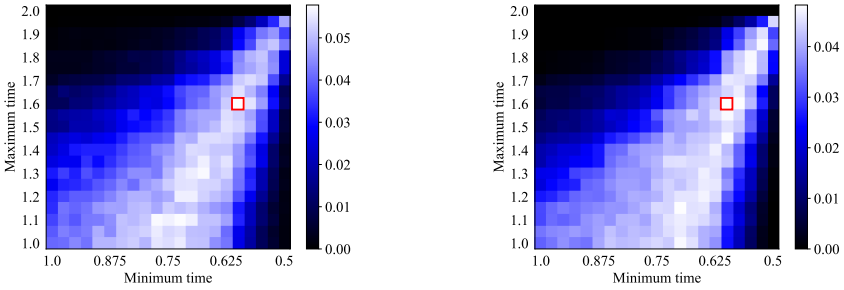


Figure 2: Rank collapse metric in different layers of various architectures.

Validation across different cases. In Subsection 3.1, we discuss three cases for deducing $\boldsymbol{W}_X$, $\boldsymbol{W}_B$ and $\boldsymbol{W}_C$ based on the relationship between sizes T and W. In fact, in all the experiments mentioned above, T is fixed at 64, while the minimum value of W is 384. This indicates that all proxies computed so far utilize case 2 ($\mathsf{T} < \mathsf{W}$). To validate the effectiveness of case 1, we set $\mathsf{T} = \mathsf{W} = 64$, and then perform sampling and calculate ranking correlation similar to that in Subsection 4.1 (though in this instance, W is fixed). To verify the validity of case 3, we set $\mathsf{T} = 128$ and $\mathsf{W} = 64$, subsequently performing the same operations as in case 1. The results are shown in Table 5, where we can see that in both cases, the ranking correlation values still exceed those of the baseline. This indicates that our method is effective across all three cases.

Validation of AH ranges. Based on existing researches on NAS, when capping computational budgets with an overhead limit, the best models often have costs slightly below the upper limit [39, 33]. Therefore, within a given search space, if the overhead of the sampled networks are more



(a) (b)

Figure 3: The metrics corresponding to different AH ranges in SSMamba2(a) and SSVWMamba2 (b). The values on the axes are arranged in this order to ensure that the upper right corner corresponds to larger search ranges. The red boxes indicate the range we use.

Table 6: Test performance of three independent training-testing runs on opt VWMamba2 w/ bwe. The abbreviations in datasets are consistent with Table 1. Run ID 1 is the model reported in Table 2, while Run ID 2 & 3 are models retrained and tested.

Run ID	LAMBADA (PPL↓)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
1	15.08	51.5	40.5	66.9	52.6	25.2	55.2	19.5
2	15.05	51.2	40.5	66.8	52.6	25.2	55.3	19.4
3	15.10	51.4	40.7	67.0	52.9	25.3	55.5	19.5

likely to fall slightly below the upper limit, the chance of discovering superior architectures is greater. The range of the AHs in the search space largely determines this probability. In Subsection 3.2, to establish the ranges of each AH, we introduce a benchmark model and specify that the values of each AH should fall between 0.6 to 1.6 times that of the benchmark model. To validate the reasonableness of these factors, we employ different factors as lower and upper limits to form varying search spaces. For each search space, we sample 10 000 network architectures and calculate a metric related to the architecture number of parameter number falling within the interval $[0.9c_0, c_0]$, where c_0 represents the expected upper limit of parameter number (see Section J of the technical appendix for the metric details). The results are presented in Figure 3. It is evident that when the lower and upper limits for the AHs are set to around 0.6 times and 1.6 times the corresponding values in the benchmark model, the metric tends to be the highest. This indicates that our choice of factors is justified.

Test-performance stability. It is well known that the test performance of the search results are subject to randomness, which may bias the evaluation. To quantify this variance, we conduct two additional training-testing runs for the reported opt-VWMamba2 w/ BWE; the results are given in Table 6. Although minor fluctuations exist, opt-VWMamba2 consistently retains a clear margin over Mamba-2-130M, Opt-Mamba2 and the baseline opt-VWMamba2 listed in Table 2. Owing to time and compute constraints, we do not repeat the same experiments for every model; nevertheless, because the type of all candidates are the same, we expect them to exhibit a similar level of test-variance.

5 Conclusion

We propose the first training-free NAS method designed for the Mamba-type architectures. Based on the rank collapse in stacked SSD layers, we develop a proxy that requires computation of the transformation matrix between the input of the Mamba2 block and the input of the SSD block. Additionally, we design a search space where the ranges of the AHs can flexibly vary according to the expected computational overhead. We validate the effectiveness of our method from two perspectives, ranking correlation and the performance of search results. However, we have only applied our method on Mamba2 so far. In the future, as many new Mamba-type models are proposed, we will validate and optimize our method on a broader range of models.

Acknowledgments and Disclosure of Funding

This work is funded by the National Natural Science Foundation of China (Grant No. 62176119), and Jiangsu Graduate Research Innovation Program (Grant No. KYCX24_0262).

References

[1] Mohamed S Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas D Lane. Zero-cost proxies for lightweight nas. *arXiv preprint arXiv:2101.08134*, 2021.

[2] Sarwat Ali and M Arif Wani. Gradient-based neural architecture search: A comprehensive evaluation. *Machine Learning and Knowledge Extraction*, 5(3):1176–1194, 2023.

[3] Malyaban Bal and Abhronil Sengupta. Rethinking spiking neural networks as state space models. *arXiv e-prints*, pages arXiv–2406, 2024.

[4] Ali Behrouz and Farnoosh Hashemi. Graph mamba: Towards learning on graphs with state space models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 119–130, 2024.

[5] Ali Behrouz, Michele Santacatterina, and Ramin Zabih. Mambamixer: Efficient selective state space models with dual token and channel selection. *arXiv preprint arXiv:2403.19888*, 2024.

[6] Kai Biegun, Rares Dolga, Jake Cunningham, and David Barber. Rotrnn: Modelling long sequences with rotations. *arXiv preprint arXiv:2407.07239*, 2024.

[7] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

[8] Michael Boratko, Harshit Padigela, Divyendra Mikkilineni, Pritish Yuvraj, Rajarshi Das, Andrew McCallum, Maria Chang, Achille Fokoue-Nkoutche, Pavan Kapanipathi, Nicholas Mattei, et al. A systematic classification of knowledge, reasoning, and context within the arc dataset. *ACL 2018*, page 60, 2018.

[9] Guo Chen, Yifei Huang, Jilan Xu, Baoqi Pei, Zhe Chen, Zhiqi Li, Jiahao Wang, Kunchang Li, Tong Lu, and Limin Wang. Video mamba suite: State space model as a versatile alternative for video understanding. *arXiv preprint arXiv:2403.09626*, 2024.

[10] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12270–12280, 2021.

[11] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. In *International Conference on Learning Representations (ICLR)*, 2021.

[12] Xiangxiang Chu, Shun Lu, Xudong Li, and Bo Zhang. Mixpath: A unified approach for one-shot neural architecture search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5972–5981, 2023.

[13] Tri Dao and Albert Gu. Transformers are ssms: generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, pages 10041–10071, 2024.

[14] Yi Deng, Yuanli Zhang, and Jingrui Zhang. The influence of mamba model hyperparameters on training time and accuracy. In *2024 IEEE International Conference on Smart Internet of Things (SmartIoT)*, pages 293–300. IEEE, 2024.

[15] Peijie Dong, Lujun Li, Zhenheng Tang, Xiang Liu, Zimian Wei, Qiang Wang, and Xiaowen Chu. Parzc: Parametric zero-cost proxies for efficient nas. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 16327–16335, 2025.

[16] Xuanyi Dong, Lu Liu, Katarzyna Musial, and Bogdan Gabrys. Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3634–3646, 2021.

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

[18] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International conference on machine learning*, pages 2793–2803. PMLR, 2021.

[19] Chao Fan, Hongyuan Yu, Yan Huang, Liang Wang, Zhenghan Yang, and Xibin Jia. Slicemamba with neural architecture search for medical image segmentation. *IEEE Journal of Biomedical and Health Informatics*, 2025.

[20] Yi Fan and Yu-Bin Yang. Training-free neural architectural search on transformer via evaluating expressivity and trainability. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2024.

[21] Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

[22] Yu Gao, Jiancheng Huang, Xiaopeng Sun, Zequn Jie, Yujie Zhong, and Lin Ma. Matten: Video generation with mamba-attention. *arXiv preprint arXiv:2405.03025*, 2024.

[23] Haifan Gong, Luoyao Kang, Yitao Wang, Yihan Wang, Xiang Wan, Xusheng Wu, and Haofeng Li. nnmamba: 3d biomedical image segmentation, classification and landmark detection with state space model. In *2025 IEEE 22nd International Symposium on Biomedical Imaging (ISBI)*, pages 1–5. IEEE, 2025.

[24] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[25] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[26] Zhenfeng He, Yao Shu, Zhongxiang Dai, and Bryan Kian Hsiang Low. Robustifying and boosting training-free neural architecture search. *arXiv preprint arXiv:2403.07591*, 2024.

[27] Tao Huang, Xiaohuan Pei, Shan You, Fei Wang, Chen Qian, and Chang Xu. Localmamba: Visual state space model with windowed selective scan. *CoRR*, 2024.

[28] Yi-Cheng Huang, Wei-Hua Li, Chih-Han Tsou, Jun-Cheng Chen, and Chu-Song Chen. Up-nas: Unified proxy for neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1675–1684, 2024.

[29] Fady Ibrahim, Guangjun Liu, and Guanghui Wang. A survey on mamba architecture for vision applications. *arXiv preprint arXiv:2502.07161*, 2025.

[30] Yesmina Jaafra, Jean Luc Laurent, Aline Deruyver, and Mohamed Saber Naceur. Reinforcement learning for neural architecture search: A review. *Image and Vision Computing*, 89:57–66, 2019.

[31] Tangyu Jiang, Haodi Wang, and Rongfang Bie. Meco: zero-shot nas with one data and single forward pass via minimum eigenvalue of correlation. *Advances in Neural Information Processing Systems*, 36:61020–61047, 2023.

[32] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[33] Junghyup Lee and Bumsub Ham. Az-nas: Assembling zero-cost proxies for network architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5893–5903, 2024.

[34] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

[35] Guihong Li, Duc Hoang, Kartikeya Bhardwaj, Ming Lin, Zhangyang Wang, and Radu Marculescu. Zero-shot neural architecture search: Challenges, solutions, and opportunities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[36] Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. Zico: Zero-shot nas via inverse coefficient of variation on gradients. *arXiv preprint arXiv:2301.11300*, 2023.

[37] Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

[38] Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot nas for high-performance image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 347–356, 2021.

[39] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[40] Liyang Liu, Shilong Zhang, Zhanghui Kuang, Aojun Zhou, Jing-Hao Xue, Xinjiang Wang, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Group fisher pruning for practical network compression. In *International Conference on Machine Learning*, pages 7021–7032. PMLR, 2021.

[41] Xiao Liu, Chenxu Zhang, and Lei Zhang. Vision mamba: A comprehensive survey and taxonomy. *arXiv preprint arXiv:2405.04404*, 2024.

[42] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 34(2):550–570, 2021.

[43] Yash Mehta, Colin White, Arber Zela, Arjun Krishnakumar, Guri Zabergja, Shakiba Moradian, Mahmoud Safari, Kaicheng Yu, and Frank Hutter. Nas-bench-suite: Nas evaluation is (now) surprisingly easy. *CoRR*, 2022.

[44] Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. Neural architecture search without training. In *International conference on machine learning*, pages 7588–7598. PMLR, 2021.

[45] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, 2018.

[46] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, 2016.

[47] Daniel S Park, Jaehoon Lee, Daiyi Peng, Yuan Cao, and Jascha Sohl-Dickstein. Towards nngp-guided neural architecture search. *arXiv preprint arXiv:2011.06006*, 2020.

[48] Badri N Patro and Vijay Srinivas Agneeswaran. Simba: Simplified mamba-based architecture for vision and multivariate time series. *CoRR*, 2024.

[49] Yameng Peng, Andy Song, Haytham M Fayek, Vic Ciesielski, and Xiaojun Chang. Swap-nas: Sample-wise activation patterns for ultra-fast nas. In *ICLR*, 2024.

[50] Huafeng Qin, Yuming Fu, Jing Chen, Mounim A El-Yacoubi, Xinbo Gao, and Feng Xi. Neural architecture search based global-local vision mamba for palm-vein recognition. *arXiv preprint arXiv:2408.05743*, 2024.

[51] Huan Ren, Jiangtao Guo, Shuli Cheng, and Yongming Li. Pooling-based visual transformer with low complexity attention hashing for image retrieval. *Expert Systems with Applications*, 241:122745, 2024.

[52] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34, 2021.

[53] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[54] Aaron Serianni and Jugal Kalita. Training-free neural architecture search for rnns and transformers. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

[55] Yao Shu, Shaofeng Cai, Zhongxiang Dai, Beng Chin Ooi, and Bryan Kian Hsiang Low. Nasi: Label-and data-agnostic neural architecture search at initialization. *arXiv preprint arXiv:2109.00817*, 2021.

[56] Yao Shu, Zhongxiang Dai, Zhaoxuan Wu, and Bryan Kian Hsiang Low. Unifying and boosting gradient-based training-free neural architecture search. *Advances in neural information processing systems*, 35:33001–33015, 2022.

[57] Julien Siems, Lucas Zimmer, Arber Zela, Jovita Lukasik, Margret Keuper, and Frank Hutter. Nas-bench-301 and the case for surrogate benchmarks for neural architecture search. *arXiv preprint arXiv:2008.09777*, 4:14, 2020.

[58] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33:6377–6389, 2020.

[59] Yujin Tang, Peijie Dong, Zhenheng Tang, Xiaowen Chu, and Junwei Liang. Vmrnn: Integrating vision mamba and lstm for efficient and accurate spatiotemporal forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5663–5673, 2024.

[60] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*.

[61] Junxiong Wang, Daniele Paliotta, Avner May, Alexander Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457, 2024.

[62] Zimian Wei, Peijie Dong, Zheng Hui, Anggeng Li, Lujun Li, Menglong Lu, Hengyue Pan, and Dongsheng Li. Auto-prox: Training-free vision transformer architecture search via automatic proxy discovery. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15814–15822, 2024.

[63] Colin White, Willie Neiswanger, and Yash Savani. Bananas: Bayesian optimization with neural architectures for neural architecture search. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10293–10301, 2021.

[64] Qianxiong Xu, Xuanyi Liu, Lanyun Zhu, Guosheng Lin, Cheng Long, Ziyue Li, and Rui Zhao. Hybrid mamba for few-shot segmentation. *Advances in Neural Information Processing Systems*, 37:73858–73883, 2024.

[65] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *International conference on machine learning*, pages 7105–7114. PMLR, 2019.

[66] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019.

[67] Yuming Zhang, Jun Hsieh, Xin Li, Ming-Ching Chang, Chun-Chieh Lee, and Kuo-Chin Fan. Mote-nas: Multi-objective training-based estimate for efficient neural architecture search. *Advances in Neural Information Processing Systems*, 37:100845–100869, 2024.

[68] Qinqin Zhou, Kekai Sheng, Xiawu Zheng, Ke Li, Xing Sun, Yonghong Tian, Jie Chen, and Rongrong Ji. Training-free transformer architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10894–10903, 2022.

[69] Qinqin Zhou, Kekai Sheng, Xiawu Zheng, Ke Li, Yonghong Tian, Jie Chen, and Rongrong Ji. Training-free transformer architecture search with zero-cost proxy guided evolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[70] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[71] Jingwei Zuo, Maksim Velikanov, Dhia Eddine Rhaiem, Ilyas Chahed, Younes Belkada, Guillaume Kunsch, and Hakim Hacid. Falcon mamba: The first competitive attention-free 7b language model. *arXiv preprint arXiv:2410.05355*, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state in the abstract and introduction that for the first time, we design a training-free neural architecture search method specifically for Mamba-type architectures, and we have conducted validation on Mamba2. Our experiments demonstrate the effectiveness of our method.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Section 5, we state that we have only applied our method on Mamba2 so far. And in the future, as many new Mamba-type models are proposed, we will validate and optimize our method on a broader range of models.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We point out that the rank collapse phenomenon occurs in stacked SSD models in Subsection 3.1, and a theoretical proof is provided in Subsection C.1 of the technical appendix. Besides, for some conclusions used in the proxy design, we also provide proofs in Section C of the technical appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: When constructing NASBench, we provide the publicly available pre-trained models we use in Subsection 4.1. When conducting architecture search, we detail the evolutionary process and the architectures of the search results in Section G and H of the technical appendix, and the language datasets used are also publicly available. For the two metrics used in Subsection 4.4, we provide a detailed calculation process in Section I and J of the technical appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We consider releasing the NASBenches and code after the paper is published.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the details of the training and test settings in Subsection 4.1 and 4.2 of the main body and G of the technical appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: The Mamba2 model, especially the larger models with more parameters, incurs excessive training costs, making it difficult to repeat the search and training process multiple times. Additionally, when constructing NASBench, while fine-tuning each model is not significantly costly, completing the fine-tuning for all 500 models still poses a substantial expense. Therefore, it is challenging to conduct repeated experiments for both ranking correlation and architecture search. As an alternative, we construct 10 NASBenches on models of various scales and test the ranking correlation separately, yielding very favorable results. This can be considered an indirect validation of the stability of the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Subsection 4.2, we provide the computational resources and time costs for the search. For ranking correlation, we present the computational time for each proxy in Table 1. For the search results, we list the model size, FLOPs, and inference latency in Table 2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We conduct in the paper conform with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We give a brief discussion of the potential positive and negative societal impacts of our work in Section B of the technical appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release data or models that have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The pretrained models and datasets we use are all publicly available, and we cite them properly.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: We currently have not released any new assets. NASBench will be released as a new asset after the paper is published.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Related work

Significant progress has been made in the area of Mamba-related research recently. The Mamba model [24], as a novel and efficient sequence modeling framework based on SSM [25], has garnered widespread attention. Later, the introduction of Mamba2 [13] makes the performance improve by 2 to 8 times through enhancements in selective SSM. Mamba2 has demonstrated competitive performance in language modeling against Transformers and has optimized system performance by introducing tensor parallelism. Subsequently, Mamba-type model makes greater progress in vision and multimodal tasks, with excellent models such as Vision Mamba [41], SiMBA [48], VMRNN [59], and MambaMixer [5]. Additionally, foundation models like Falcon Mamba 7B [71], Jamba [37], VideoMamba [9] have been proposed as attempts to replace Transformer models across various tasks. However, some studies indicate that there is not a replacement relationship between Mamba and existing architectures; combining Mamba with current models like Transformer [22], convolutional network [23], graph network [4], recurrent network [6], spiking network [3] may lead to better results. These contributions have not only deepened theoretical understanding of Mamba but also propelled its development through system optimization and application expansion, revealing immense potential across multiple fields. However, many of these models involve numerous hyperparameters that dictate the actual architecture of the models and often require manual tuning.

NAS is initially introduced by [70] and has since led to various optimization strategy-based methods [30, 2, 42, 63, 12]. Considering that these methods still require extensive computational resources during the search process, some researchers propose training-free NAS, also known as zero-shot NAS, inspired by parameter prunning [34, 58, 1]. Subsequently, grounded in neural network learning theory, multiple training-free NAS methods for convolution networks have been proposed by analyzing activation pattern [44, 49], analyzing gradients [56], utilizing neural tangent kernel [55, 11], viewing the performance prediction as a Gaussian process [47], or employ alternative heuristic methods [31]. To achieve better performance, some studies [26, 28, 33] combine multiple proxies and employ optimization strategies to switch among them dynamically. Unfortunately, the design of proxies in these methods necessitates consideration of the characteristics of network structures and training principles, making them unsuitable for the prevalent Transformer architecture. To address this, DSS [68], based on the theoretical insights from [21, 18], pioneers a NAS method for Transformers that measures the synaptic diversity in MSA modules and synaptic significance in Multi-Layer Perceptron (MLP) modules. Building on this, DSS++ [69] proposes a layer-wise proxy approach. Other researchers [54, 20] have evaluated network performance using attention map features. In summary, current training-free NAS designs primarily focus on convolutional networks and Transformers, leaving other architectures, including Mamba, without suitable methods. From another perspective, the existing methods for optimizing Mamba-type models [50, 27, 19] all utilize traditional NAS approaches, which involve significant computational overhead. Therefore, research on training-free NAS for Mamba-type models is highly urgent.

B Societal impacts

In the context of the rapid development of artificial intelligence and deep learning technologies, methods for the automatic optimization of Mamba architecture have multidimensional social implications.

From a positive perspective, these methods are expected to significantly enhance the performance and efficiency of Mamba models, propelling deep learning applications in various fields such as natural language processing and computer vision to new heights. For example, in the field of medical image analysis, optimized Mamba architectures can more accurately and swiftly identify lesions, assisting doctors in diagnosis. This improvement can increase the accuracy and efficiency of disease diagnosis, benefiting a larger number of patients. In autonomous driving scenarios, optimized architectures can more sensitively perceive the surrounding environment and make more precise decisions, thereby accelerating the practical application of autonomous driving technologies. This advancement is of great significance for improving traffic safety and alleviating congestion, enhancing people's travel experiences, and promoting the development of intelligent transportation systems.

Moreover, owning to avoiding training, the method can significantly save computational resources and time costs. Training large deep learning models typically requires substantial computing power and time, while this approach circumvents the iterative training process. This is particularly advantageous in scenarios involving large-scale hyperparameter searches and model architecture selection,

effectively enhancing resource utilization efficiency. Consequently, researchers and enterprises can allocate more resources to the practical application and further optimization of models, facilitating the rapid implementation of technology.

However, this method also has potential negative implications. Although it does not require training, the process of selecting network architectures may still involve data processing and analysis. If data management is inadequately handled, there exists the risk of data privacy breaches, particularly when dealing with datasets that contain personal sensitive information. Such breaches could lead to the exposure of personal information, raising societal concerns regarding data security and privacy protection.

C Theoretical analysis and proofs

C.1 Rank collapse in stacked SSD layers

In this subsection, we proof the rank collapse in stacked SSD layers. Intuitively, based on the rank collapse of stacked attention layers [18], along with the duality of attention and SSD [13], one could directly draw conclusions. However, it is important to emphasize that the attention modules aligned with the forward process of the SSD are not the original attention modules, but rather modifications made to them involving two key improvements: (a) masking operations on the attention map; and (b) the use of linear attention operations [32]. This indicates that the forward computational process of the SSD does not entirely align with the attention modules for the analysis of rank collapse in [18]. Consequently, to confirm that stacking SSD modules will indeed lead to rank collapse, we will use mathematical induction to prove that after k operations, the distance between the matrix $\boldsymbol{X}_k$ and the rank-1 matrix $\boldsymbol{Y}^*$ satisfies

$$\|\boldsymbol{X}_k - \boldsymbol{Y}^*\| \leq \beta\alpha^k. \tag{8}$$

When $k = 0$, assume that the initial matrix $\boldsymbol{X}_0$ has a distance of β from $\boldsymbol{Y}^*$, that is,

$$\|\boldsymbol{X}_0 - \boldsymbol{Y}^*\| \leq \beta, \tag{9}$$

and suppose that after k operations, the distance satisfies

$$\|\boldsymbol{X}_k - \boldsymbol{Y}^*\| \leq \beta\alpha^k. \tag{10}$$

Considering the $(k+1)$-th operation

$$\boldsymbol{X}_{k+1} = (\boldsymbol{L} \odot \boldsymbol{X}_k)\,\boldsymbol{W}, \tag{11}$$

we analyze $\|\boldsymbol{X}_{k+1} - \boldsymbol{Y}^*\|$:

$$\begin{aligned}
\|\boldsymbol{X}_{k+1} - \boldsymbol{Y}^*\| &= \|(\boldsymbol{L} \odot \boldsymbol{X}_k)\,\boldsymbol{W} - \boldsymbol{Y}^*\| \\
&= \left\|(\boldsymbol{L} \odot \boldsymbol{X}_k)\,\boldsymbol{W} - \boldsymbol{a}\boldsymbol{b}^\top\right\| \\
&\leq \left\|(\boldsymbol{L} \odot \boldsymbol{X}_k)\,\boldsymbol{W} - \boldsymbol{L} \odot \left(\boldsymbol{a}\boldsymbol{b}^\top\boldsymbol{W}^+\right)\right\| + \left\|\boldsymbol{L} \odot \left(\boldsymbol{a}\boldsymbol{b}^\top\boldsymbol{W}^+\right) - \boldsymbol{a}\boldsymbol{b}^\top\right\| \\
&\leq \left\|\boldsymbol{L} \odot \left(\boldsymbol{X}_k\boldsymbol{W} - \boldsymbol{a}\boldsymbol{b}^\top\boldsymbol{W}^+\right)\right\| + \left\|\boldsymbol{L} \odot \boldsymbol{a}\boldsymbol{b}^\top\boldsymbol{W}^+ - \boldsymbol{a}\boldsymbol{b}^\top\right\| \\
&\leq \alpha\left\|\boldsymbol{X}_k - \boldsymbol{a}\boldsymbol{b}^\top\right\| + \gamma
\end{aligned} \tag{12}$$

Here, α and γ are constants related to matrices $\boldsymbol{L}$ and $\boldsymbol{W}$, with $0 < \alpha < 1$ and γ being a small constant.

By the inductive hypothesis and the above inequality, we can conclude that

$$\|\boldsymbol{X}_{k+1} - \boldsymbol{Y}^*\| \leq \alpha\,\|\boldsymbol{X}_k - \boldsymbol{Y}^*\| + \gamma \leq \alpha\beta\alpha^k + \gamma = \beta\alpha^{k+1} + \gamma \tag{13}$$

As k becomes large, α^{k+1} approaches zero, so the distance $\|\boldsymbol{X}_{k+1} - \boldsymbol{Y}^*\|$ also approaches zero. The proof is now complete.

This conclusion is derived under the scenario of a single head; however, based on the principles of path decomposition outlined in [18], it is evident that our conclusion also holds true in the case of multiple heads.

C.2 Reasonableness of the Gaussianity and independence assumption for U

Our assumption of a normal distribution for input elements is based on the absence of prior knowledge about the inputs. This approach is also adopted by some training-free NAS methods for CNNs, such as [38], which generate normally distributed tensors to simulate unknown inputs. To validate this assumption, we conducted experiments using a network with width $\mathsf{W} = 96$ and depth $\mathsf{D} = 3$, selecting 1,000 samples and truncating them to meet the condition of token number $\mathsf{T} = 96$. We extract the inputs $\boldsymbol{U}$ of each Mamba2 block and use the Kolmogorov-Smirnov test to confirm that these samples approximately follow a normal distribution. Additionally, we directly inspect the ranks of all 3,000 $\boldsymbol{U}$ matrices (1,000 samples with three Mamba2 block inputs each) and find them all to be full-rank matrices. These results demonstrate that our assumption does not lead to errors in practice.

C.3 Probability of full-rank matrix for Gaussian random matrix

In this subsection, we will prove that, for a matrix sampled from an independent standard normal distribution, the probability of it being full rank is 1.

First, consider the case where the matrix is square. When the matrix is an $n \times n$ square matrix, its determinant is a continuous and non-constant function that maps the elements of the matrix to real numbers. The set of singular matrices (those with a determinant equal to zero) forms a low-dimensional algebraic variety. According to the properties of absolutely continuous probability measures, this set has Lebesgue measure zero. Therefore, the probability that a matrix with independently and identically distributed standard normal elements lies in this zero-measure set is zero. Thus, the probability that the matrix is of full rank is 1.

Next, we consider the case of non-square matrices. For a non-square matrix $\boldsymbol{A}$ (assumed to be of size $m \times n$), if $m > n$, the rank of the matrix is at most n. Conversely, if $m < n$, the rank is at most m. However, for matrices whose elements are drawn from independent standard normal distributions, the row (or column) vectors are linearly independent with probability 1. When $m = n$, the probability that the matrix is of full rank is 1; when $m < n$, the probability that the matrix has full row rank is also 1; similarly, when $m > n$, the probability that the matrix has full column rank is 1.

Additionally, we can use mathematical induction for proof. Assume the probability that the first k row vectors are linearly independent is 1. When we add the $(k+1)$-th row, the probability that this row falls within the k-dimensional subspace spanned by the previous k rows is zero. Given the independence of the standard normal distribution, the probability that the newly added row vector is linearly dependent on the original set of row vectors is zero. Therefore, by mathematical induction, we conclude that the probability of all row vectors being linearly independent is 1, which implies that the probability of the matrix being full rank is also 1.

C.4 Probability of inclusion in the column space of row-full rank matrices

Let $\boldsymbol{A}$ be an $m \times n$ matrix satisfying $m < n$ and having full row rank (i.e., its rank is m). In this section, we will verify that the probability of a random vector $x \in \mathbb{R}^m$, sampled from an independent standard normal distribution, lying within the column space of $\boldsymbol{A}$ is 1.

Since $\boldsymbol{A}$ has full row rank (rank-m), its row space has dimension m, and consequently, its column space also has dimension m. We denote the column space of $\boldsymbol{A}$ as $\mathrm{Col}\,(A)$. Given that $\boldsymbol{A}$ is an $m \times n$ matrix, the column space $\mathrm{Col}\,(A)$ is a subspace of $\mathbb{R}^m$. Moreover, a full row rank matrix will have a column space dimension of m, which matches the dimension of $\mathbb{R}^m$. Therefore, we have $\mathrm{Col}\,(A) = \mathbb{R}^m$. The random vector $\boldsymbol{x}$ is independently sampled from a standard normal distribution, which means its distribution covers the entire $\mathbb{R}^m$ space, except for a set of measure zero. In $\mathbb{R}^m$, the measure of single-point sets (such as the origin) or lower-dimensional subspaces is zero. Since $\mathrm{Col}\,(\boldsymbol{A}) = \mathbb{R}^m$, the probability that $\boldsymbol{x}$ lies in $\mathrm{Col}\,(A)$ is therefore 1. Thus, we have proven this conclusion.

C.5 Probability of inclusion in the column space of column-full rank matrices

Let $\boldsymbol{A}$ be an $m \times n$ matrix satisfying $m > n$ and having full column rank (rank-n). In this subsection, we will verify that the probability of a random vector $\boldsymbol{x} \in \mathbb{R}^m$, sampled from an independent standard normal distribution, lying within the column space of $\boldsymbol{A}$ is 0.

The proof follows a similar structure to that of the previous subsection. Since the matrix $\boldsymbol{A}$ is a full rank $m \times n$ matrix with $m > n$, it has a rank of n. Consequently, the column space of $\boldsymbol{A}$ is an n-dimensional subspace. The vector $\boldsymbol{x}$ is a random vector that is independently sampled from a standard normal distribution, which means its distribution spans the entirety of $\mathbb{R}^m$. In $\mathbb{R}^m$, any subspace of dimension less than m (such as the column space of $\boldsymbol{A}$, which has dimension $n < m$) has a Lebesgue measure of zero. Since the distribution of $\boldsymbol{x}$ is absolutely continuous with respect to Lebesgue measure, the probability that $\boldsymbol{x}$ lies in a set of measure zero is also zero. Thus, we have proven the conclusion.

C.6 Error bounds for pseudoinverse approximations

For Case 3, we analyze the upper bound of $\|\boldsymbol{U}\boldsymbol{W}_X - \boldsymbol{X}\|$ and derived the conclusion $\|\boldsymbol{U}\boldsymbol{W}_X - \boldsymbol{X}\| \leq \|\boldsymbol{X}\|_F$ (the proof is provided below). This implies that as long as the Frobenius norm of $\boldsymbol{X}$ is not excessively large, the resulting error is small. Similar conclusions apply to $\boldsymbol{B}$ and $\boldsymbol{C}$. When calculating the proxy, we can preprocess the input data (e.g., multiplying by a small constant) to keep $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ at small values, thereby controlling the error introduced by this approximation.

The remainder of this subsection is devoted to proving the conclusion stated in the preceding paragraph. The least squares solution $\boldsymbol{W}_X$ corresponds to the orthogonal projection of $\boldsymbol{X}$ onto the column space of $\boldsymbol{U}$. The residual matrix $\boldsymbol{R} = \boldsymbol{U}\boldsymbol{W}_X - \boldsymbol{X}$ lies in the orthogonal complement of $\boldsymbol{U}$. Since orthogonal projection does not increase the norm, the projection of $\boldsymbol{X}$ onto the orthogonal complement (i.e., the Frobenius norm of the residual matrix $\boldsymbol{R}$) must not exceed the Frobenius norm of the original matrix $\boldsymbol{X}$:

$$\|\boldsymbol{R}\|_F = \left\| \left(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^\top \right) \boldsymbol{X} \right\|_F \leq \|\boldsymbol{X}\|_F \tag{14}$$

where $\boldsymbol{I}$ is the identity matrix, and $\boldsymbol{V}$ is the matrix of left singular vectors of $\boldsymbol{U}$.

If $\boldsymbol{X}$ is entirely within the orthogonal complement of $\boldsymbol{U}$, then $\boldsymbol{U}\boldsymbol{W}_X = \boldsymbol{0}$, and $\|\boldsymbol{U}\boldsymbol{W}_X - \boldsymbol{X}\| \leq \|\boldsymbol{X}\|_F$, reaching the upper bound. If $\boldsymbol{X}$ is entirely within the column space of $\boldsymbol{U}$, the residual is zero, and the upper bound holds.

In summary, the original inequality is established, and the proof is complete.

C.7 Time-complexity derivation for our proxy

In this subsection, we begin with deriving the time complexity of the Mamba2 proxy for given W, N, H, and P.

Firstly, the complexity of SVD is $O\left(\mathsf{TW}\min\left(\mathsf{T},\mathsf{W}\right)\right)$, and that of computing the pseudo-inverse is $O\left(\min\left(\mathsf{T},\mathsf{W}\right)\right)$. Calculating $\boldsymbol{U}^+$ has a time complexity of $O\left(\mathsf{WT}^2\right)$, while computing $\boldsymbol{W}_X$, $\boldsymbol{W}_B$, $\boldsymbol{W}_C$ and their gradients has complexities of $O\left(\mathsf{WTP}\right)$, $O\left(\mathsf{WTN}\right)$, and $O\left(\mathsf{WTN}\right)$ respectively. The nuclear norm computation for these matrices has complexities of $O\left(\mathsf{WP}\min\left(\mathsf{WP}\right)\right)$, $O\left(\mathsf{WN}\min\left(\mathsf{WN}\right)\right)$, and $O\left(\mathsf{WN}\min\left(\mathsf{WN}\right)\right)$. Additionally, computing the nuclear norm of $\boldsymbol{W}_{out}$ and its gradient is $O\left(WP\min\left(\mathsf{WP}\right)\right)$. Here, T is input-dependent and can be treated as a constant. Finally, considering multiple heads, the time complexity for one Mamba2 block is $O\left(\mathsf{WHP}\min\left(\mathsf{WP}\right) + \mathsf{WHN}\min\left(\mathsf{WN}\right)\right)$. So it is evident that when any of W, N, H, P approaches infinity, the time complexity grows linearly.

It should be noted that VWSSMamba2 merely expands the search space of SSMamba2 without increasing the scale of candidate networks. The scale of candidate networks is determined by the expected computational overhead. Thus, the time overhead does not increase significantly.

C.8 Proof that additive noise tends to decreases Kendall's Tau

In this subsection, we prove that adding noise to performance data tends to decrease the Kendall's Tau between proxy and performance.

Assume performance data is $(x_1, x_2, \cdots, x_n)$ and proxy data is $(y_1, y_2, \cdots, y_n)$. The Kendall's Tau coefficient is defined as

$$\tau = \frac{C - D}{\frac{1}{2}n\left(n-1\right)}, \tag{15}$$

where C is the number of concordant pairs and D is the number of discordant pairs.

When noise $\epsilon_i \sim \mathcal{N}\left(0, \sigma^2\right)$ is added to each x_i, forming $x_i' = x_i + \epsilon_i$, the comparison between any two samples i and j changes. The difference $\epsilon_i - \epsilon_j$ follows $\mathcal{N}\left(0, 2\sigma^2\right)$. For a pair (i, j) where $x_i > x_j$, the probability that $x_i' > x_j'$ is $1 - \Phi\left(\frac{-d}{\sqrt{2}\sigma}\right)$, where $d = x_i - x_j$. For a pair where $x_i < x_j$, the probability that $x_i' < x_j'$ is $\Phi\left(\frac{d}{\sqrt{2}\sigma}\right)$, where $d = x_j - x_i$.

Noise can turn concordant pairs into discordant pairs and vice versa. The expected number of concordant pairs decreases, while the expected number of discordant pairs increases. Thus, the expected new Kendall's Tau τ' after adding noise is:

$$\mathbb{E}\left[\tau'\right] = \frac{\mathbb{E}\left[C'\right] - \mathbb{E}\left[D'\right]}{\frac{1}{2}n\left(n-1\right)} < \tau. \tag{16}$$

This shows that adding normal noise reduces the expected Kendall's Tau value. The proof is complete.

D Pseudocode of TF-MAS

This section provides the pseudocode of proxy computation process in TF-MAS, listed in Algorithm 1.

Algorithm 1 Proxy of TF-MAS

Input: Network architecture $\mathcal{A}$, input $\boldsymbol{D}$
Output: Proxy value TF-MAS

$\boldsymbol{W}_{\text{out}}^{(1)}, \boldsymbol{W}_{\text{out}}^{(2)}, \ldots, \boldsymbol{W}_{\text{out}}^{(\mathsf{D})} \leftarrow$ Initializa $\mathcal{A}$ and get $\boldsymbol{W}_{\text{out}}$.
$\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \ldots, \boldsymbol{U}^{(\mathsf{D})}, \boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \ldots, \boldsymbol{X}^{(\mathsf{D})}, \boldsymbol{B}^{(1)}, \boldsymbol{B}^{(2)}, \ldots, \boldsymbol{B}^{(\mathsf{D})}, \boldsymbol{C}^{(1)}, \boldsymbol{C}^{(2)}, \ldots, \boldsymbol{C}^{(\mathsf{D})} \leftarrow$ Forward propagation and get $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{C}$.
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(\mathsf{D})}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(\mathsf{D})}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(\mathsf{D})}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{\text{out}}^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{\text{out}}^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{\text{out}}^{(\mathsf{D})}} \leftarrow$ Back propagation and get gradients of $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{W}_{\text{out}}$.
$\{\text{class}_1, \text{class}_2, \ldots, \text{class}_M\} \leftarrow$ Classify by column number($\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \ldots, \boldsymbol{X}^{(\mathsf{D})}, \boldsymbol{B}^{(1)}, \boldsymbol{B}^{(2)}, \ldots, \boldsymbol{B}^{(\mathsf{D})}, \boldsymbol{C}^{(1)}, \boldsymbol{C}^{(2)}, \ldots, \boldsymbol{C}^{(\mathsf{D})}$).
for class_m in $\{\text{class}_1, \text{class}_2, \ldots, \text{class}_M\}$ **do**
    index list $\leftarrow$ Get index(class_m).
    $\boldsymbol{U}$ list $\leftarrow$ Get elements(index list).
    $\boldsymbol{U}_B$, $\mathbf{XBC}_B \leftarrow$ Concatenate($\boldsymbol{U}$ list), Concatenate(class_m) //concatenate to form the batch dimension. $\dim\left(\boldsymbol{U}_B\right)_1$, $\dim\left(\boldsymbol{U}_B\right)_2$, $\dim\left(\boldsymbol{U}_B\right)_3$ are the batch size, row number, and column number of $\boldsymbol{U}_B$, respectively.
    if $\dim\left(\boldsymbol{U}_B\right)_2 = \dim\left(\boldsymbol{U}_B\right)_3$ **then**
        $\boldsymbol{W}_{Bm} \leftarrow \boldsymbol{U}_B^{-1}\mathbf{XBC}_B$.
    else
        $\boldsymbol{\sigma}, \boldsymbol{V}_1, \boldsymbol{V}_2 \leftarrow$ SVD($\boldsymbol{U}_B$).
        $\boldsymbol{\sigma}^+ \leftarrow$ Pseudoinverse($\boldsymbol{\sigma}$).
        $\boldsymbol{W}_{Bm} \leftarrow \boldsymbol{V}_2\boldsymbol{\sigma}^+\boldsymbol{V}_1^\top\mathbf{XBC}_B$.
    end if
end for
$\boldsymbol{W}_X^{(1)}, \boldsymbol{W}_X^{(2)}, \ldots, \boldsymbol{W}_X^{(\mathsf{D})}, \boldsymbol{W}_B^{(1)}, \boldsymbol{W}_B^{(2)}, \ldots, \boldsymbol{W}_B^{(\mathsf{D})}, \boldsymbol{W}_C^{(1)}, \boldsymbol{W}_C^{(2)}, \ldots, \boldsymbol{W}_C^{(\mathsf{D})} \leftarrow$ Rearrange by $\{\text{class}_1, \text{class}_2, \ldots, \text{class}_M\}(\boldsymbol{W}_{B1}, \boldsymbol{W}_{B2}, \ldots, \boldsymbol{W}_{BM})$.
$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_X^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_X^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_X^{(\mathsf{D})}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_B^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_B^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_B^{(\mathsf{D})}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_C^{(1)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_C^{(2)}}, \ldots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_C^{(\mathsf{D})}} \leftarrow \boldsymbol{U}^{(1)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(1)}}, \boldsymbol{U}^{(2)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(2)}}, \ldots, \boldsymbol{U}^{(\mathsf{D})\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}^{(\mathsf{D})}}, \boldsymbol{U}^{(1)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(1)}}, \boldsymbol{U}^{(2)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(2)}}, \ldots, \boldsymbol{U}^{(\mathsf{D})\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}^{(\mathsf{D})}}, \boldsymbol{U}^{(1)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(1)}}, \boldsymbol{U}^{(2)\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(2)}}, \ldots, \boldsymbol{U}^{(\mathsf{D})\top}\frac{\partial \mathcal{L}}{\partial \boldsymbol{C}^{(\mathsf{D})}}$.
TF-MAS $\leftarrow \sum_{i=1}^{l}\left(\sum_{x\in\{X,B,C,\text{out}\}}\left\|\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_x}\right\|_{\text{nuc}}\|\boldsymbol{W}_x\|_{\text{nuc}}\right)$.
return TF-MAS

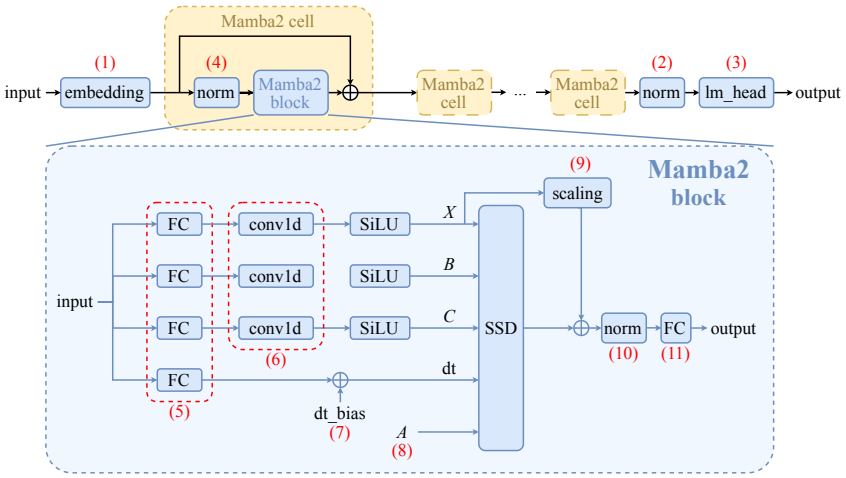


Figure 4: Structure of Mamba2 and the positions of parameters. The parameters are indicated by red numbers enclosed in parentheses. Compared to Figure 1, this figure provides additional details to illustrate all parameters within the model.

E Calculation of parameter number in Mamba2

In Subsection 3.2, we provide the expression for the parameter number of Mamba2 given the hyperparameters V, D, W, N, and H. Here, we present the derivation process for this expression.

Figure 2 illustrates all the parameters contained within the Mamba2 model. The corresponding sizes are as follows:

(1) Embedding: This component includes a weight with a size of $\mathsf{V} \times \mathsf{W}$, resulting in a parameter number of VW.

(2) Normalization after all Mamba2 cells: This component includes a weight with a size (and thus a parameter number) of W.

(3) Language model head: This component includes a weight with a size of $\mathsf{V} \times \mathsf{W}$, resulting in a parameter number of VW.

(4) Normalization in each Mamba2 cell: This component includes a weight with a size (and thus a parameter number) of W.

(5) Fully connected layer in each Mamba2 block: This component includes a weight without bias. The output from this component serves as the data source for $\boldsymbol{X}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\mathbf{dt}$, and $\boldsymbol{Z}$, which have feature dimensions of $64\mathsf{H}$, N, N, H, and $64\mathsf{H}$, respectively, while the input dimension is W. Therefore, the parameter number for this weight is $(64\mathsf{H} + \mathsf{N} + \mathsf{N} + \mathsf{H} + 64\mathsf{H})\,\mathsf{W} = 2\mathsf{WN} + 129\mathsf{WH}$.

(6) Convolution layer in each Mamba2 block: This component includes a one-dimensional convolution with a kernel size of 4, which includes a weight and a bias. As a precursor for generating $\boldsymbol{X}$, $\boldsymbol{B}$, and $\boldsymbol{C}$, the corresponding feature dimension is $64\mathsf{H} + \mathsf{N} + \mathsf{N} = 2\mathsf{N} + 64\mathsf{H}$. Consequently, the parameter number for the weight is $(2\mathsf{N} + 64\mathsf{H}) \times 4 = 8\mathsf{N} + 256\mathsf{H}$, and for the bias, it is $2\mathsf{N} + 64\mathsf{H}$.

(7) dt bias in each Mamba2 block: This is an independent parameter with a size (and thus a parameter number) of H.

(8) $\boldsymbol{A}$ in each Mamba2 block: This is an independent parameter with a size (and thus a parameter number) of H.

(9) Scaling factor in each Mamba2 block: This is an independent parameter with a size (and thus a parameter number) of H.

(10) Normalization in each Mamba2 block: This component includes a weight with a size (and thus a parameter number) of $64\mathsf{H}$.

Table 7: Ranking correlation on SSMamba2Bench_130M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.411	0.424	0.248	0.279	0.233	0.281	0.330	0.381
Gradnorm	C	-0.083	0.041	0.075	0.027	0.005	-0.023	0.040	0.007
Synflow	C	-0.029	0.042	0.074	0.068	0.001	0.080	-0.016	0.092
GraSP	C	-0.020	0.051	-0.001	-0.017	-0.004	0.019	0.043	0.059
Fisher	C	-0.074	0.079	0.038	0.031	0.006	0.044	0.061	0.026
Snip	C	-0.077	0.045	0.009	0.043	0.011	0.037	0.044	-0.037
Zen-NAS	C	-0.113	0.056	0.069	0.054	0.053	0.035	0.042	0.009
ZiCo	C	-0.065	0.044	0.052	0.096	0.089	0.003	0.047	0.071
MeCo	C	-0.065	0.052	0.007	0.031	0.061	0.088	0.038	0.050
Auto-Prox	C	-0.013	0.069	0.034	-0.024	-0.013	0.007	0.108	0.034
AC	T	-0.086	0.132	-0.001	0.096	0.035	0.120	0.149	0.081
HI	T	-0.107	0.134	0.063	0.102	0.140	0.070	0.096	0.120
HC	T	-0.068	0.089	0.106	0.089	0.052	0.124	0.087	0.120
DSS	T	-0.143	0.099	0.074	0.150	0.109	0.094	0.106	0.157
AttnNAS	T	-0.230	0.294	0.121	0.168	0.167	0.181	0.232	0.193
TF-MAS (ours)	M	**-0.677**	**0.648**	**0.406**	**0.451**	**0.364**	**0.435**	**0.517**	**0.508**

(11) Fully connected layer in each Mamba2 block: This component includes a weight without bias. The output dimension of the tensor changes from $64\mathsf{H}$ to W; therefore, the parameter number for the weights is $64\mathsf{WH}$.

The total number of parameters for Mamba2 is the sum of the parameter number from the above components. Notably, components (4)-(11) appear once in each layer, necessitating multiplication by the depth D when calculating them. Ultimately, we derive the total parameter number as $c = \mathsf{VW} + \mathsf{W} + \mathsf{VW} + (\mathsf{W} + 2\mathsf{WN} + 129\mathsf{WH} + 8\mathsf{N} + 256\mathsf{H} + 2\mathsf{N} + 64\mathsf{H} + \mathsf{H} + \mathsf{H} + \mathsf{H} + 64\mathsf{H} + 64\mathsf{WH})\,\mathsf{D} = \mathsf{W} + 2\mathsf{VW} + \mathsf{DW} + 10\mathsf{DN} + 387\mathsf{DH} + 2\mathsf{DWN} + 193\mathsf{DWH}$.

For the VWManba2 model, the AHs N and H are no longer consistent across layers. Therefore, we denote them as N_i and H_i for the i-th layer, leading to the total parameter number $c = \mathsf{VW}+\mathsf{W}+\mathsf{VW}+ \sum_{i=1}^{\mathsf{D}} (\mathsf{W} + 2\mathsf{WN}_i + 129\mathsf{WH}_i + 8\mathsf{N}_i + 256\mathsf{H}_i + 2\mathsf{N}_i + 64\mathsf{H}_i + \mathsf{H}_i + \mathsf{H}_i + \mathsf{H}_i + 64\mathsf{H}_i + 64\mathsf{WH}_i) = \mathsf{W} + 2\mathsf{VW} + \mathsf{DW} + \sum_{i=1}^{\mathsf{D}} (10\mathsf{N}_i + 387\mathsf{H}_i + 2\mathsf{WN}_i + 193\mathsf{WH}_i)$.

F More results about ranking correlation

In this section, we present the ranking correlations for all NASBenches datasets except for VMSSMamba2Bench_2.7B. Tables 7 to 11 correspond to the NASBenches related to Mamba2 (i.e., SSMamba2Bench_130M, SSMamba2Bench_370M, SSMamba2Bench_780M, and SSMamba2Bench_1.3B), while Tables 12 to 15 correspond to the NASBenches related to VWMamba2 (i.e., VWSSMamba2Bench_130M, VWSSMamba2Bench_370M, VWSSMamba2Bench_780M, VWSSMamba2Bench_1.3B, and VWSSMamba2Bench_2.7B). It can be observed that, regardless of the NASBench, the conclusions drawn are consistent with those presented in Table 1. Therefore, our method achieves strong performance across all NASBench datasets.

Besides, in Subsection 4.1, we point out that ranking correlation evaluated only on the top-k candidates sometimes is necessary. Therefore, we also conduct experiments on VMSSMamba2Bench_2.7B to evaluate the ranking correlation for the top 20%, top 10%, and top 5% of candidates. The results are presented in Table 16. These results indicate that the ranking correlation for the top-k candidates is comparable to that of the entire candidate set.

G Details of the search process

In Subsection 4.2, we set the expected computational overhead to "the number of parameters not exceeding 130M". Under this condition, the feasible values for each AH are as follows:

Table 8: Ranking correlation on SSMamba2Bench_370M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.414	0.441	0.233	0.289	0.235	0.320	0.363	0.363
Gradnorm	C	0.003	0.071	0.058	0.057	0.009	0.051	0.054	0.098
Synflow	C	-0.035	0.064	0.066	0.044	0.047	0.027	0.074	0.055
GraSP	C	-0.021	-0.007	0.028	0.047	0.023	0.052	-0.000	0.049
Fisher	C	-0.065	0.089	0.048	0.094	0.084	-0.030	0.063	0.044
Snip	C	-0.084	0.028	-0.014	0.033	0.051	0.025	0.028	-0.003
Zen-NAS	C	-0.130	0.084	0.003	0.035	-0.007	-0.014	0.023	0.035
ZiCo	C	-0.055	0.084	0.014	0.019	0.045	0.076	0.059	-0.003
MeCo	C	-0.058	0.058	0.035	0.029	0.052	-0.004	0.052	0.044
Auto-Prox	C	-0.067	0.024	0.057	0.006	-0.002	0.015	0.041	0.077
AC	T	-0.090	0.124	0.118	0.102	0.068	0.050	0.082	0.044
HI	T	-0.091	0.132	0.064	0.088	0.078	0.054	0.123	0.115
HC	T	-0.149	0.107	0.038	0.061	0.053	0.058	0.084	0.088
DSS	T	-0.150	0.146	0.084	0.063	0.069	0.056	0.112	0.171
AttnNAS	T	-0.248	0.289	0.142	0.214	0.129	0.177	0.233	0.216
TF-MAS (ours)	M	**-0.666**	**0.685**	**0.377**	**0.419**	**0.388**	**0.467**	**0.472**	**0.560**

Table 9: Ranking correlation on SSMamba2Bench_780M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.429	0.439	0.228	0.328	0.260	0.311	0.368	0.359
Gradnorm	C	-0.016	0.060	0.055	0.036	0.009	0.080	0.077	0.043
Synflow	C	-0.089	-0.022	-0.030	0.020	0.052	0.037	0.052	0.050
GraSP	C	-0.067	0.040	0.034	0.057	0.023	0.045	0.039	0.045
Fisher	C	-0.043	0.069	0.037	0.050	0.038	0.045	0.024	0.128
Snip	C	-0.015	0.052	0.045	-0.010	0.012	0.027	-0.031	0.085
Zen-NAS	C	-0.052	0.125	0.054	0.003	0.070	0.050	-0.001	0.047
ZiCo	C	-0.032	0.015	0.029	0.109	0.000	0.008	0.032	0.013
MeCo	C	-0.047	0.081	0.048	0.049	0.049	0.040	0.072	0.089
Auto-Prox	C	-0.050	0.023	0.041	0.013	0.050	-0.018	-0.001	0.006
AC	T	-0.112	0.031	0.081	0.077	0.054	0.097	0.103	0.126
HI	T	-0.093	0.078	0.087	0.061	0.093	0.126	0.106	0.096
HC	T	-0.106	0.067	0.078	0.111	0.041	0.024	0.080	0.091
DSS	T	-0.105	0.117	0.104	0.098	0.109	0.095	0.118	0.093
AttnNAS	T	-0.298	0.240	0.181	0.131	0.108	0.222	0.193	0.208
TF-MAS (ours)	M	**-0.655**	**0.684**	**0.325**	**0.442**	**0.396**	**0.424**	**0.501**	**0.520**

- D: 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 (13 values)
- W: 384, 416, 448, 480, 512, 544, 576, 608, 640, 672, 704, 736, 768 (13 values)
- N: 64, 72, 80, 88, 96, 104, 112, 120, 128 (9 values)
- H: 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 (13 values)

For SSMamba2, the size of the search space is given by $13 \times 13 \times 9 \times 13 = 19,773$. For VWSS-Mamba2, the size of the search space is approximately $\sum_{i=12}^{24} 13 \times 9^i \times 13^i \approx 5.68 \times 10^{50}$.

During the search for opt Mamba2 and opt VWMamba2, our search method employs an evolutionary approach with a population size of 50, iterating for 300 generations. In each iteration, there are three methods for generating new individuals: (1) randomly inheriting from an original individual; (2) randomly selecting an original individual for mutation. During mutation, one AH is randomly chosen; if the AH is an integer, it is changed to a neighboring selectable value, while if the AH is a list (only exists in the case where the search space is SSVWMamba2), 1/5 of its elements are randomly selected

Table 10: Ranking correlation on SSMamba2Bench_1.3B. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.426	0.414	0.238	0.315	0.215	0.323	0.331	0.404
Gradnorm	C	-0.082	0.102	0.046	0.028	0.048	0.052	0.043	0.073
Synflow	C	-0.006	0.035	0.037	0.022	0.032	0.019	0.026	0.051
GraSP	C	-0.024	0.071	0.007	0.049	0.021	0.071	0.075	0.007
Fisher	C	-0.085	0.008	0.035	0.017	0.074	0.027	0.041	0.019
Snip	C	-0.023	0.047	0.079	0.053	0.017	0.036	0.034	0.083
Zen-NAS	C	-0.112	0.003	-0.017	0.073	0.015	0.026	0.058	-0.007
ZiCo	C	-0.042	0.048	0.026	-0.037	0.080	0.024	0.034	0.051
MeCo	C	-0.023	0.083	0.042	0.052	0.016	-0.003	0.011	-0.005
Auto-Prox	C	-0.067	0.059	0.007	0.028	0.010	0.037	0.054	0.050
AC	T	-0.099	0.162	0.089	0.087	0.103	0.099	0.083	0.135
HI	T	-0.076	0.133	0.010	0.093	0.033	0.049	0.116	0.086
HC	T	-0.108	0.079	0.040	0.061	0.008	0.036	0.076	0.068
DSS	T	-0.152	0.140	0.094	0.082	0.119	0.111	0.102	0.111
AttnNAS	T	-0.202	0.273	0.130	0.163	0.166	0.197	0.249	0.228
TF-MAS (ours)	M	**-0.667**	**0.654**	**0.316**	**0.467**	**0.379**	**0.440**	**0.513**	**0.544**

Table 11: Ranking correlation on SSMamba2Bench_2.7B. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.449	0.456	0.326	0.329	0.232	0.308	0.285	0.356
Gradnorm	C	-0.035	0.063	0.070	0.032	0.069	0.074	0.034	0.031
Synflow	C	-0.032	0.076	0.048	0.005	-0.002	0.054	0.036	0.074
GraSP	C	-0.043	0.047	0.013	0.023	-0.011	0.041	0.088	0.013
Fisher	C	-0.046	0.082	0.047	0.057	0.034	0.033	0.031	0.092
Snip	C	-0.037	0.023	0.066	0.031	-0.005	0.059	0.027	0.105
Zen-NAS	C	-0.067	0.080	0.091	0.059	0.064	0.038	0.054	0.003
ZiCo	C	-0.071	0.001	0.058	0.037	0.059	0.008	0.017	0.018
MeCo	C	-0.087	0.051	0.010	0.039	0.100	0.091	0.025	0.039
Auto-Prox	C	-0.061	0.041	0.010	0.058	0.047	0.049	0.007	0.005
AC	T	-0.075	0.153	0.070	0.053	0.078	0.094	0.083	0.102
HI	T	-0.071	0.088	0.100	0.120	0.041	0.102	0.071	0.076
HC	T	-0.124	0.087	0.101	0.030	0.110	0.051	0.095	0.113
DSS	T	-0.073	0.154	0.086	0.109	0.073	0.122	0.088	0.111
AttnNAS	T	-0.237	0.257	0.140	0.168	0.124	0.179	0.245	0.215
TF-MAS (ours)	M	**-0.656**	**0.663**	**0.408**	**0.446**	**0.385**	**0.429**	**0.521**	**0.510**

and changed to neighboring selectable values; (3) randomly selecting two original individuals for crossover, where two AHs of the new individual are randomly taken from one original individual, and the remaining two AHs are taken from the other original individual. If the parameter number of the new individual exceeds 130M, a new individual is generated until the parameter number is below 130M.

During the search for opt VWMamba2 w/ bwe using block-wise evolution, we calculate the proxy values for each layer separately, meaning that the outermost summation operation is not performed when applying Equation (6). Once we obtain these proxies, during the mutation process, if a list-type AH is selected for mutation, the elements that need to be changed are no longer chosen randomly but are designated as the elements corresponding to the top a layers with the minimum proxy values, where a is the number of elements to be changed. In the crossover process, the values of N and H are no longer wholly inherited from one original individual. For each element value of the new individual, which original individual it inherits from depends on which original individual has a greater proxy value in the corresponding layer. Thus, the values of N and H in the new individual are hybrids from the two original individuals. To balance the contributions of both original individuals,

Table 12: Ranking correlation on VWSSMamba2Bench_130M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.445	0.437	0.215	0.308	0.242	0.334	0.317	0.336
Gradnorm	C	-0.021	0.037	0.034	0.073	-0.005	0.027	0.012	0.022
Synflow	C	-0.019	0.048	-0.008	0.038	0.059	0.016	0.034	0.051
GraSP	C	-0.004	0.003	0.050	0.075	0.075	0.066	0.049	0.064
Fisher	C	-0.050	0.031	0.033	0.067	0.043	0.022	0.054	0.096
Snip	C	-0.069	0.060	0.062	0.014	0.060	0.066	0.060	0.011
Zen-NAS	C	-0.061	0.027	0.044	0.034	0.052	0.069	0.040	0.093
ZiCo	C	-0.100	0.080	0.032	0.001	0.038	0.049	0.043	0.056
MeCo	C	-0.058	0.042	0.011	0.115	0.068	0.009	0.045	0.075
Auto-Prox	C	-0.027	0.061	0.019	-0.036	0.008	0.011	0.041	0.059
AC	T	-0.060	0.153	0.111	0.095	0.094	0.078	0.096	0.067
HI	T	-0.059	0.099	0.047	0.109	0.039	0.067	0.051	0.109
HC	T	-0.022	0.088	0.073	0.044	0.061	0.041	0.055	0.078
DSS	T	-0.166	0.125	0.139	0.114	0.112	0.128	0.104	0.159
AttnNAS	T	-0.253	0.225	0.097	0.156	0.151	0.204	0.191	0.249
TF-MAS (ours)	M	**-0.675**	**0.667**	**0.357**	**0.450**	**0.329**	**0.467**	**0.485**	**0.556**

Table 13: Ranking correlation on VWSSMamba2Bench_370M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.384	0.429	0.209	0.314	0.277	0.308	0.316	0.372
Gradnorm	C	-0.044	-0.008	0.008	0.066	0.039	0.040	0.037	0.092
Synflow	C	0.005	0.017	0.072	0.005	0.073	0.090	0.051	0.050
GraSP	C	-0.055	0.059	0.087	0.036	0.007	0.028	0.022	0.037
Fisher	C	-0.028	0.069	0.029	0.032	-0.014	0.083	0.062	0.023
Snip	C	-0.032	0.006	0.078	0.077	-0.008	0.075	0.014	0.044
Zen-NAS	C	-0.114	0.035	0.087	0.054	0.062	0.050	0.029	0.106
ZiCo	C	-0.073	0.041	0.064	0.006	0.023	0.075	0.012	0.013
MeCo	C	-0.074	0.076	0.045	-0.011	0.069	0.007	0.006	0.043
Auto-Prox	C	-0.040	0.027	-0.013	0.030	-0.011	0.014	0.036	0.036
AC	T	-0.081	0.080	0.082	0.100	0.092	0.100	0.094	0.152
HI	T	-0.076	0.038	0.026	0.061	0.089	0.076	0.046	0.123
HC	T	-0.112	0.157	0.040	0.035	0.063	0.036	0.107	0.081
DSS	T	-0.127	0.114	0.067	0.088	0.072	0.106	0.152	0.094
AttnNAS	T	-0.238	0.259	0.097	0.169	0.143	0.236	0.185	0.270
TF-MAS (ours)	M	**-0.660**	**0.672**	**0.378**	**0.444**	**0.359**	**0.486**	**0.495**	**0.547**

for the remaining two integer-valued AHs, D and W, one original individual is randomly selected to inherit the value of D, while the other inherits the value of W.

H Search results

In this section, we present the architectures of the search results from Subsection 4.2.

Optimized Mamba2:

- D: 43
- W: 608
- N: 104
- H: 18

Optimized VWMamba2:

Table 14: Ranking correlation on VWSSMamba2Bench_780M. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.435	0.433	0.316	0.260	0.258	0.298	0.292	0.351
Gradnorm	C	-0.084	0.106	-0.040	0.026	0.019	0.053	0.055	0.080
Synflow	C	-0.037	0.006	-0.041	0.056	0.101	0.056	0.021	0.109
GraSP	C	-0.109	0.079	0.032	0.029	0.077	-0.013	0.041	0.097
Fisher	C	-0.053	0.024	-0.000	0.063	0.056	0.089	0.058	-0.015
Snip	C	-0.069	-0.010	0.022	0.065	0.043	0.037	0.040	0.097
Zen-NAS	C	-0.018	0.047	0.043	-0.005	0.070	0.046	-0.017	0.060
ZiCo	C	0.009	-0.025	-0.013	0.036	0.023	0.072	0.039	0.081
MeCo	C	-0.040	0.077	0.050	0.053	0.060	0.036	-0.025	0.061
Auto-Prox	C	-0.038	0.021	-0.011	0.048	0.031	-0.034	0.010	0.042
AC	T	-0.138	0.136	0.050	0.128	0.085	0.091	0.096	0.126
HI	T	-0.109	0.086	0.063	0.136	0.050	0.072	0.050	0.064
HC	T	-0.065	0.059	0.071	0.081	0.041	0.077	0.130	0.099
DSS	T	-0.100	0.111	0.143	0.081	0.094	0.104	0.075	0.108
AttnNAS	T	-0.283	0.249	0.161	0.228	0.141	0.198	0.253	0.204
TF-MAS (ours)	M	**-0.649**	**0.698**	**0.365**	**0.430**	**0.364**	**0.451**	**0.470**	**0.557**

Table 15: Ranking correlation on VWSSMamba2Bench_1.3B. The abbreviations are consistent with Table 1. Best results in bold.

Proxy	Target	LAMBADA (PPL)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
#Param	general	-0.390	0.443	0.309	0.331	0.243	0.331	0.344	0.350
Gradnorm	C	-0.051	0.063	0.019	0.067	0.036	0.042	0.048	0.067
Synflow	C	-0.002	0.028	0.030	0.080	0.041	0.030	-0.012	-0.026
GraSP	C	-0.062	0.002	0.018	0.023	-0.007	0.021	0.022	-0.002
Fisher	C	-0.059	0.087	-0.007	0.054	-0.020	0.040	0.038	-0.024
Snip	C	-0.061	-0.012	0.019	-0.018	0.063	0.088	0.002	0.091
Zen-NAS	C	-0.122	0.030	-0.011	0.010	0.000	0.048	0.010	0.037
ZiCo	C	-0.035	0.052	-0.024	0.042	0.036	0.055	0.042	0.038
MeCo	C	-0.037	0.047	0.019	0.050	-0.010	0.019	0.033	0.047
Auto-Prox	C	-0.106	0.017	0.050	-0.001	0.062	-0.012	0.062	0.032
AC	T	-0.100	0.122	0.099	0.071	0.127	0.114	0.054	0.125
HI	T	-0.100	0.046	0.090	0.064	0.034	0.081	0.076	0.108
HC	T	-0.097	0.133	0.071	0.073	0.029	0.045	0.045	0.082
DSS	T	-0.148	0.162	0.078	0.083	0.079	0.098	0.110	0.139
AttnNAS	T	-0.249	0.247	0.122	0.160	0.177	0.140	0.261	0.206
TF-MAS (ours)	M	**-0.681**	**0.697**	**0.324**	**0.464**	**0.377**	**0.480**	**0.489**	**0.527**

Table 16: Ranking correlation on VMSSMamba2Bench_2.7B using varying numbers of candidates. The abbreviations in datasets are consistent with Table 1.

Candidates	LAMBADA (PPL↓)	LAMBADA (ACC)	HS (ACC)	PIQA (ACC)	Arc-E (ACC)	Arc-C (ACC)	WG (ACC)	OBQA (ACC)
all	-0.685	0.661	0.381	0.468	0.339	0.452	0.483	0.519
top 20%	-0.693	0.683	0.390	0.483	0.347	0.470	0.486	0.527
top 10%	-0.675	0.653	0.374	0.453	0.319	0.455	0.468	0.500
top 5%	-0.662	0.692	0.366	0.421	0.332	0.439	0.489	0.522

- D: 35
- W: 544
- N: [104, 96, 112, 72, 72, 64, 80, 64, 96, 80, 80, 120, 88, 80, 96, 64, 104, 72, 104, 64, 80, 88, 104, 104, 80, 112, 64, 112, 72, 120, 104, 120, 112, 112, 88]
- H: [22, 21, 22, 28, 22, 21, 31, 19, 29, 20, 26, 21, 24, 23, 28, 16, 25, 18, 30, 22, 31, 21, 31, 20, 27, 25, 24, 26, 23, 21, 18, 21, 28, 25, 29]

Optimized VWMamba2 with block wise evolution:

- D: 27
- W: 672
- N: [88, 104, 72, 72, 72, 72, 96, 88, 104, 104, 112, 88, 88, 120, 72, 120, 72, 80, 96, 72, 104, 64, 120, 112, 80, 88, 120]
- H: [31, 25, 31, 29, 18, 18, 20, 25, 30, 24, 17, 27, 29, 25, 17, 19, 25, 30, 25, 19, 30, 21, 29, 19, 25, 21, 28]

I Details of rank collapse metric

In Subsection 4.1, we introduce a metric to evaluate the extent of rank collapse. This subsection will detail the specifics of this metric. For a matrix $\boldsymbol{X}$ under evaluation, the motivation of the metric is to measure its distance to the nearest rank-1 matrix, defined as $\operatorname{argmin}_{\boldsymbol{x}} \left\|\boldsymbol{X} - \mathbf{1}\boldsymbol{x}^\top\right\|_{1,\infty}$. However, this metric may be influenced by the size of $\boldsymbol{X}$ and the scale of its elements. To address this issue, the rank collapse metric is the normalized distance:

$$\text{metric} = \frac{\operatorname{argmin}_{\boldsymbol{x}} \left\|\boldsymbol{X} - \mathbf{1}\boldsymbol{x}^\top\right\|_{1,\infty}}{\operatorname{argmin}_{\boldsymbol{x}} \left\|\boldsymbol{X}\right\|_{1,\infty}}. \tag{17}$$

J Details of AH ranges evaluation metric

In Subsection 4.4, we introduce a metric to evaluate the range of AHs within the search space. This section will elaborate on the details of this metric. Intuitively, an ideal AH range needs to ensure that as many architectures as possible with their parameter numbers fall within the interval $[0.9c_0, c_0]$, while also maintaining diversity among these architectures by maximizing their pairwise distances. Therefore, our metric first selects architectures whose parameter numbers lie within the interval $[0.9c_0, c_0]$ and then considers the AHs as features. We begin by identifying the center of these architectures, followed by calculating the sum of distances from each architecture to the center. The magnitude of this sum reflects the rationality of the AH range. To ensure that the metric is not influenced by the sampling count, we normalize this value by the number of samples, yielding the final metric value.

Specifically, for SSMamba2, let m represent the number of architectures whose parameter numbers fall within the interval $[0.9c_0, c_0]$. Let the AHs for the i-th architecture be denoted as $\mathsf{D}^{(i)}$, $\mathsf{W}^{(i)}$, $\mathsf{N}^{(i)}$, and $\mathsf{H}^{(i)}$. To ensure consistency in feature scaling, we first normalize these parameters as follows:

$$\hat{\mathsf{D}}^{(i)} = \frac{\mathsf{D}^{(i)}}{\mathsf{D}_0}, \quad \hat{\mathsf{W}}^{(i)} = \frac{\mathsf{W}^{(i)}}{\mathsf{W}_0}, \quad \hat{\mathsf{N}}^{(i)} = \frac{\mathsf{N}^{(i)}}{\mathsf{N}_0}, \quad \hat{\mathsf{H}}^{(i)} = \frac{\mathsf{H}^{(i)}}{\mathsf{H}_0}, \tag{18}$$

where $\hat{\mathsf{D}}^{(i)}$, $\hat{\mathsf{W}}^{(i)}$, $\hat{\mathsf{N}}^{(i)}$, and $\hat{\mathsf{H}}^{(i)}$ are the normalized features, and D_0, W_0, N_0, and H_0 are the AHs of the benchmark model. We then calculate the center values for each feature:

$$\bar{\mathsf{D}} = \sum_{i=1}^{m} \hat{\mathsf{D}}^{(i)}, \quad \bar{\mathsf{W}} = \sum_{i=1}^{m} \hat{\mathsf{W}}^{(i)}, \quad \bar{\mathsf{N}} = \sum_{i=1}^{m} \hat{\mathsf{N}}^{(i)}, \quad \bar{\mathsf{H}} = \sum_{i=1}^{m} \hat{\mathsf{H}}^{(i)}. \tag{19}$$

Based on this, we can compute the metric as follows:

$$\text{metric} = \frac{1}{m} \sum_{i=1}^{m} \sqrt{\left(\hat{\mathsf{D}}^{(i)} - \bar{\mathsf{D}}\right)^2 + \left(\hat{\mathsf{W}}^{(i)} - \bar{\mathsf{W}}\right)^2 + \left(\hat{\mathsf{N}}^{(i)} - \bar{\mathsf{N}}\right)^2 + \left(\hat{\mathsf{H}}^{(i)} - \bar{\mathsf{H}}\right)^2}. \tag{20}$$

For the SSVWMamba2 search space, the overall calculation process is similar to that of SSMamba2; however, there is a critical difference: the values of s and h are lists rather than integers. Consequently, the terms $\left(\hat{\mathsf{N}}^{(i)} - \bar{\mathsf{N}}\right)^2$ and $\left(\hat{\mathsf{H}}^{(i)} - \bar{\mathsf{H}}\right)^2$ in Equation (20) require adjustment. Considering that the values of N and H in SSVWMamba2 are lists, we denote the normalized representations of the i-th architecture as $\hat{\mathbf{N}}^{(i)}$ and $\hat{\mathbf{H}}^{(i)}$ (bold letters). Notably, the dimensions of $\hat{\mathbf{N}}^{(i)}$ and $\hat{\mathbf{H}}^{(i)}$ may not be the same across architectures, i.e., $\dim \hat{\mathbf{N}}^{(i)} \neq \dim \hat{\mathbf{N}}^{(j)}$ and $\dim \hat{\mathbf{H}}^{(i)} \neq \dim \hat{\mathbf{H}}^{(j)}$ may not hold. Therefore, we consider performing linear interpolation on $\hat{\mathbf{N}}^{(i)}$ and $\hat{\mathbf{H}}^{(i)}$ to unify their lengths to $\mathsf{D}^{(i)}$. Let the interpolated values be $\hat{\mathbf{N}}^{(i)}_{\text{int}}$ and $\hat{\mathbf{H}}^{(i)}_{\text{int}}$; we then replace the terms $\left(\hat{\mathsf{N}}^{(i)} - \bar{\mathsf{N}}\right)^2$ and $\left(\hat{\mathsf{H}}^{(i)} - \bar{\mathsf{H}}\right)^2$ in Equation (20) with the average squared distances of their components:

$$\text{metric} = \frac{1}{m}\sqrt{\sum_{i=1}^{m}\left[\left(\hat{\mathsf{D}}^{(i)} - \bar{\mathsf{D}}\right)^2 + \left(\hat{\mathsf{W}}^{(i)} - \bar{\mathsf{W}}\right)^2 + \left(\frac{1}{\mathsf{D}^{(i)}}\sum_{j=1}^{\mathsf{D}^{(i)}} \hat{\mathsf{N}}^{(i)}_{\text{int},j} - \bar{\mathsf{N}}_j\right)^2 + \left(\frac{1}{\mathsf{D}^{(i)}}\sum_{j=1}^{\mathsf{D}^{(i)}} \hat{\mathsf{H}}^{(i)}_{\text{int},j} - \bar{\mathsf{H}}_j\right)^2\right]}. \quad (21)$$

Here, $\hat{\mathsf{N}}^{(i)}_{\text{int},j}$ and $\hat{\mathsf{H}}^{(i)}_{\text{int},j}$ denote the j-th components of $\hat{\mathbf{N}}^{(i)}_{\text{int}}$ and $\hat{\mathbf{H}}^{(i)}_{\text{int}}$, while $\bar{\mathsf{N}}_j$ and $\bar{\mathsf{H}}_j$ represent the j-th components of $\bar{\mathbf{N}}$ and $\bar{\mathbf{H}}$, respectively.

Intuitively, as more architecture samples fall within the interval $[0.9c_0, c_0]$, the summation terms in Equations (20) and (21) will increase, which favors a larger computed result. Similarly, if the distances between architectures within this interval are greater, their distances to the center will also increase, further contributing to a larger result. Thus, our metric is considered reasonable.