AstroNet: When Astrocyte Meets Artificial Neural Network

Mengqiao Han* Liyuan Pan*,† Xiabi Liu†
Beijing Institute of Technology
{hmq, liyuan.pan}@bit.edu.cn

Abstract

Network structure learning aims to optimize network architectures and make them more efficient without compromising performance. In this paper, we first study the astrocytes, a new mechanism to regulate connections in the classic M-P neuron. Then, with the astrocytes, we propose an AstroNet that can adaptively optimize neuron connections and therefore achieves structure learning to achieve higher accuracy and efficiency. AstroNet is based on our built Astrocyte-Neuron model, with a temporal regulation mechanism and a global connection mechanism, which is inspired by the bidirectional communication property of astrocytes. With the model, the proposed AstroNet uses a neural network (NN) for performing tasks, and an astrocyte network (AN) to continuously optimize the connections of NN, i.e.*, assigning weight to the neuron units in the NN adaptively. Experiments on the classification task demonstrate that our AstroNet can efficiently optimize the network structure while achieving state-of-the-art (SOTA) accuracy.*

1. Introduction

Neural networks have made remarkable success in visual tasks by leveraging a large number of learnable parameters. Deployment of such big models, however, may lead to overfitting and unnecessarily increase the computational of the network [15]. Neural network structure learning is a new learning paradigm to train neural networks by leveraging structured signals in addition to feature inputs.

Existing works can be generally divided into Learning Sparse Networks (LSN) and Neural Architecture Search (NAS). LSN methods obtain sub-networks from a fixed architecture by minimizing the sum of the loss term, and the penalty term [51, 63]. Though efficiency, this strategy generally sacrifices accuracy [2, 32, 44], especially for networks with more capacity [27]. NAS methods sample and combine different units in a defined search space to form an architecture, then evaluate the architecture to determine the optimal output [5, 29, 37]. This strategy requires huge time and computing resources. Though recent NAS works attempt to improve efficiency, their computational costs are still expensive [30, 53, 62, 65].

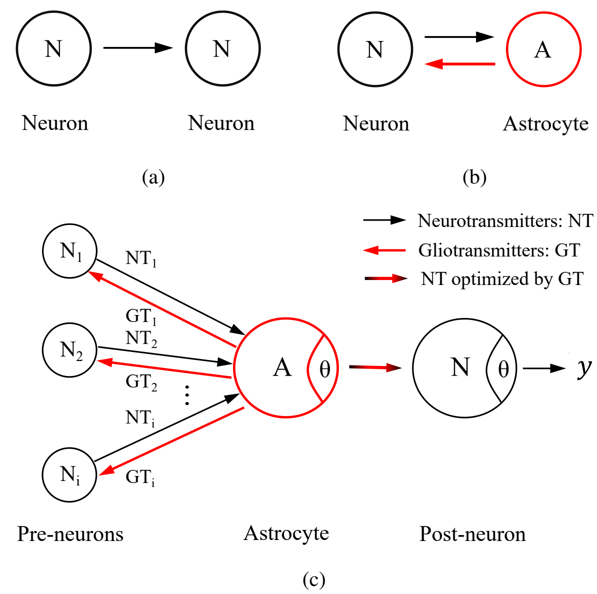


Figure 1. *Illustration of the M-P model and our Astrocyte-Neuron model. (a) A basic unit of the M-P model. The connection between neurons propagates in one direction. (b) A basic unit of the Astrocyte-Neuron model. The connection between astrocyte and neuron is propagating bidirectionally. (c) Our Astrocyte-Neuron model. The astrocyte communicates with neurons bidirectionally as key supportive elements in neuronal function. (Best viewed in color on the screen)*

Inspired by the learning activity in mammalian brains, we propose an AstroNet to effectively optimize network structure while preserving accuracy. Considering the basic units in artificial neural networks are neurons, the strength of connections between neurons can potentially reflect neuron activity [14, 17]. We, therefore, re-assigning connections (weight) for a given network by regulating the neuron connections adaptively to achieve structure learning. Different from NAS has a huge search space that may introduce

*Equal contribution, †corresponding authors human bias [12], we obtain sub-networks from a fixed architecture to reduce the search space, while not relying on the sparse regularization from LSN.

To enable the adaptive connection regulation ability of neurons, we introduce astrocytes [13] to the M-P model [33]. The previous M-P model (Fig. 1a) connect neurons using neurotransmitters in one direction, *i.e.*, from multiple pre-neurons to the post-neuron. According to the new tripartite synapse concept [11], astrocytes communicate with neurons bidirectionally and are recognized as key supportive elements in neuronal function (Fig. 1b). Specifically, astrocytes are stimulated by neuron-released neurotransmitters. Then, astrocytes generate gliotransmitters to regulate neuron connections [13]. In particular, astrocytes have a similar ability to integrate information as neurons [36], which allows us to model astrocytes as neurons. Therefore, we extend the M-P model to the Astrocyte-Neuron model for regulating neuron communications, *i.e.*, connections (Fig. 1c).

We explore the bidirectional communication property of astrocytes, and then formulate the Astrocyte-Neuron model by defining a temporal regulation mechanism and a global connection mechanism. Specifically, the astrocyte temporally regulates the connections with pre-neurons, based on the received weights from all pre-neurons. When the regulation tends to be stable (the re-assigned weights are approximate to the received weights), the astrocyte then propagates the updated weight to the post-neuron. With the model, we construct our AstroNet that can regulate network architectures to achieve connection optimization, *i.e.*, structure learning. Our AstroNet consists of a one-direction propagating neural network (NN), and a bidirectional astrocyte network (AN). The NN constitutes the network for performing tasks, and the AN follows the Astrocyte-Neuron model to regulate the connections/weights of the NN adaptively.

The AstroNet can be utilized with multiple backbones (NN), *e.g.*, ResNet18, ResNet34, DenseNet-BC, VggNet, and MLP, and we demonstrate our efficiency and accuracy in the classification task with three public datasets. Compared to LSN methods, our AstroNet improves the accuracy by $0.17\% \sim 2.81\%$. While for NAS methods, the relative improvements are $0.22\% \sim 2.79\%$ in accuracy, and $3 \sim 70+$ times in efficiency.

Our main contributions are:

- We introduce astrocyte as a new neural unit to the M-P model to solve the structure learning efficiently without compromising network performance.
- By exploring the bidirectional connection of astrocytes, we build the Astrocyte-Neuron model with a temporal regulation mechanism and a global connection mechanism.
- With the Astrocyte-Neuron model, we conduct our AstroNet, which requires few computational resources and exhibits excellent accuracy improvement.

2. Related Work

Learning sparse networks methods perform feature selection on fixed networks. One trend is the element-wise sparsity that uses ℓ_1 norm. The strategy easily results in an accuracy drop in deep networks [27, 28, 51]. Another trend is structured sparsity which includes group sparsity and gate constraint. Wen *et al*. [51] apply group LASSO to the filters, channels and residual blocks as a sparsity regularization, which makes the parameters in some groups all zero. Zhu *et al*. [63] observe that even with strong sparsity regularization applied to group LASSO, there still exists a correlation between filters. Other branches are combinations with other regularizers [1, 61], and better group sparse regularizers [25, 45]. Louizos *et al*. [32] use a collection of non-negative stochastic gates to approximate the ℓ_0 norm. However, the penalty terms are complex for networks with more capacity and may lead to an accuracy drop, especially sensitivity to hyper-parameters [27]. In addition, Ramanujan *et al*. [38] find a sub-network on the initialized network by training a mask as a connection selector. However, it performs worse on smaller networks.

Neural architecture search methods contain network parameters optimization and architecture optimization.

The network parameters optimization includes independent optimization and sharing optimization. Independent optimization learns each network separately [39]. Sharing optimization [3] accelerates training by sharing all the parameters for different architectures within one Super-Net.

The architecture optimization is to search the network architectures, which include: 1) search space defines which architectures can be represented. Global search spaces are to search in a whole space [4]. Motivated by handcrafted architectures consisting of repeated motifs, cell-based search spaces are proposed [10,62]; 2) search strategy includes RL-based, EA-based and gradient-based methods. RL-based methods [64, 65] use the recurrent network as the architecture controller, and the performances of the generated architectures are utilized as the rewards for training the controller. EA-based methods [39, 55] search architectures with evolutionary algorithms. The validation accuracy of each individual is utilized as the fitness to evolve the next generation. Gradient-based methods [9, 56] regard network architecture as a group of learnable parameters; 3) estimation strategy uses the intermediate training accuracy to represent the true accuracy to improve efficiency [26, 58] or focus on the ranking of architectures [57].

Although a lot of work has attempted to improve the efficiency, such as reducing the size of the search space [60,62], decreasing search time cost [52,59], adopting early stopping in the evaluation phase [34, 39], the time to find a proper

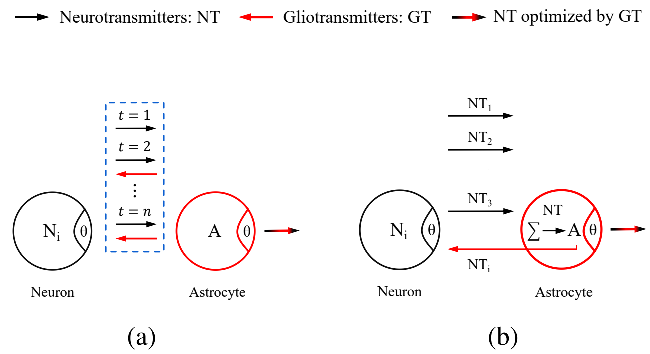


Figure 2. *The temporal regulation mechanism and global connection mechanism between neurons and the astrocyte. (a) The neurotransmitter produced by the i^{th} neuron is optimized t times by the bidirectional propagation between it and astrocytes. (b) The optimization of the i^{th} neuron connection is related to neurotransmitters $\{\mathrm{NT}_i\}_1^N$ from all neurons.*

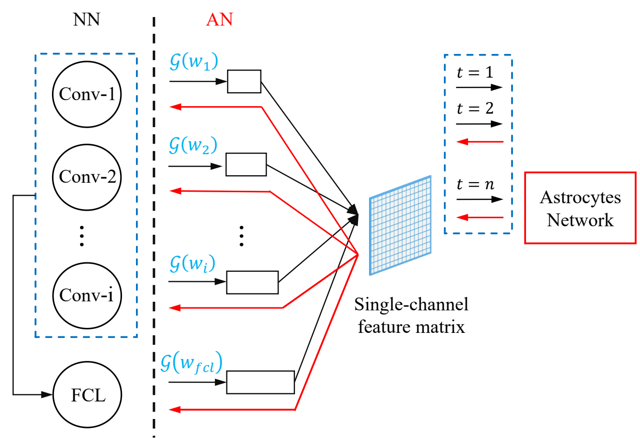


Figure 3. *The architecture of our AstroNet. The NN is used to perform the task, and the AN iteratively optimizes the connections of the NN to optimize the structure of the NN adaptively.*

architecture is still measured in days with the stacking of GPUs.

3. Proposed Method

In this section, we first introduce our Astrocyte-Neuron model in Sec. 3.1. Then, we elaborate on our AstroNet architecture (Sec. 3.2) and the training scheme (Sec. 3.3).

3.1. Astrocyte-Neuron Model

Considering a traditional M-P model [33], without astrocyte, the connection between pre-neurons and the post-neurons is modeled as:

$$y = \phi\left(\sum_i^N x_i w_i\right), \tag{1}$$

where N is the number of neurons, x_i is the input of i^{th} pre-neuron, w_i is the connecting weight from the i^{th} pre-neuron to the post-neuron, y denotes the output signal of the post-neuron, and $\phi(\cdot)$ is the activation function. With the concept of astrocytes in Fig. 1c, we have

$$y = \phi\left(h\left(\sum_i^N x_i f(w_i)\right)\right), \tag{2}$$

where $f(\cdot)$ denotes the regulative function of astrocytes. As the astrocytes can be treated as a kind of neuron [36], we use the $h(\cdot)$ as the activation function of astrocytes. We further formulate the Astrocyte-Neuron model in two aspects by exploring the bidirectional communication property of astrocytes [11, 35].

First, the bidirectional connection leads to a temporal regulation mechanism. We take one unit '$\mathrm{N}_i - \mathrm{A}$' from Fig. 1c to get Fig. 2a as an example. The neuron N_i releases neurotransmitters to stimulate the astrocyte, and then the astrocyte provides gliotransmitters as feedback. The gliotransmitters regulate N_i to re-release neurotransmitters that can also re-stimulate the astrocyte. Therefore, the weight w_i (neurotransmitters) is regulated by the gliotransmitters iteratively. We have

$$f(w_i^t) = f(w_i^{t-1} f(w_i^{t-1})) = f^{t-1}(w_i^{t-1}), \tag{3}$$

where $f^{t-1}(\cdot)$ is specifically a temporal regulation function, $t \in [1, T]$ is the iteration time, w_i^t is the connecting weights of i^{th} pre-neuron at time t.

Second, according to the tripartite synapse concept [11, 36], the astrocyte integrals weights $\{w_i\}_1^N$ from pre-neurons $\{\mathbf{N_i}\}_1^N$ to release gliotransmitters. Therefore, we can build our Astrocyte-Neuron model with the global connections, *i.e.*, $w_i^t = \sum_1^N g(w_j^t)$. Taking Eq. (2) and Eq. (3), we have our Astrocyte-Neuron model:

$$\begin{aligned} y &= \phi\left(h\left(\sum_i^N x_i f^{t-1}(w_i^{t-1})\right)\right) \\ &= \phi\left(h\left(\sum_i^N x_i f^{t-1}\left(\sum_1^N g(w_j^{t-1})\right)\right)\right), \end{aligned} \tag{4}$$

where $g(\cdot)$ is the global connection function. Fig. 2a and Fig. 2b illustrates the temporal regulation mechanism and global connection mechanism between neurons and astrocytes. In summary, the astrocyte integrates neurotransmitters $\{w_i^{t-1}\}_1^N$ released by pre-neurons and outputs gliotransmitters that regulate the connections of these pre-neurons. Then, pre-neurons update their neurotransmitters to $\{w_i^t\}_1^N$ and stimulate the astrocyte again. When the differences between neurotransmitters from time $t-1$ to t are

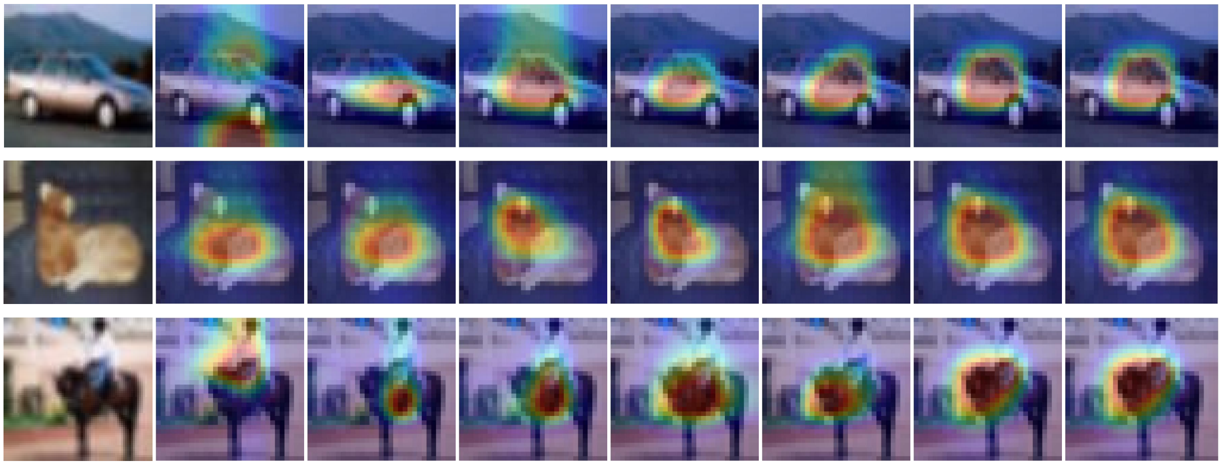


Figure 4. *We display the NN's (ResNet18 on CIFAR10) attention, optimized by our AN, on the input image (a) with the iteration number varying from* 1 *to* 7 *(b)-(h). The attention map adaptively focuses on the target region.*

slight (the regulation provided by astrocytes is light), the neurotransmitters at the time t are finally regulated by the astrocyte and then fed to the post-neuron. With the explored Astrocyte-Neuron model, we then conduct our AstroNet to optimize the connections of neurons.

3.2. AstroNet

Our AstroNet consists of a one-direction propagating neural network (NN) and a bidirectional propagating astrocyte network (AN). As shown in Fig. 3, NN constitutes the network for performing tasks, and AN regulates the connections in NN based on our Astrocyte-Neuron model, *i.e.*, assigns weight to the structural units in the NN, such as filters in convolutional layers (CLs) and neurons in fully connected layers (FCLs). We first formulate our AstroNet, and then give details on designing the two mechanisms in AN that regulate the NN's connections.

Let $X = [x_1, x_2, \cdots, x_N]^{\mathrm{T}}$ be the data. The traditional neural network is modeled as,

$$y = \phi(XW) \,, \tag{5}$$

where $W = [w_1, w_2, \cdots, w_N]^{\mathrm{T}}$ are weights of structural units in the NN, y denotes the output signal of a NN, and $\phi(\cdot)$ is the activation function. Based on our Astrocyte-Neuron model in Eq. (4), AN integrates the weights of NN at the latest timestamp W^{t-1}, then outputs probabilities P^t to regulate the connections of neurons by updating weights of NN W^{t-1} to W^t. Hence, Eq. (5) is rewritten into,

$$\begin{aligned} y &= \mathcal{H}(XW^t) \\ W^t &= W^{t-1} \odot P^t \\ P^t &= \mathcal{F}(\mathcal{G}(W^{t-1}), W^A) \,, \end{aligned} \tag{6}$$

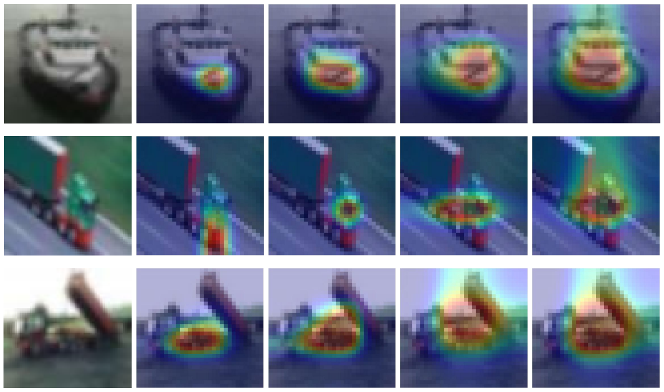
where $\mathcal{H}(\cdot)$ denotes the function of AstroNet that is a combination of the activation function of the astrocyte and post-neuron, $\odot$ denotes the element-wise product, W^A and P denote the weights and output of AN, respectively. Note, t is the iteration timestamp, $\mathcal{F}(\cdot,\cdot)$ and $\mathcal{G}(\cdot)$ are the temporal regulation function and the global connection function.

The temporal regulation mechanism. Based on bidirectional propagation, astrocytes satisfy a temporal regulation mechanism through the interaction between neurotransmitters and gliotransmitters. Based on Eq. (3) and Eq. (6), the temporal regulation mechanism in AstroNet is expressed as

$$\begin{aligned} &W^1 = W \odot P^1, \quad W^2 = W^1 \odot P^2 \\ &\cdots, \qquad\qquad\quad W^t = W^{t-1} \odot P^t \,. \end{aligned} \tag{7}$$

Fig. 4 shows the transfer of attention regions on the input image with NN (ResNet18) under AN's regulation by iterations. The visualization follows the Grad-CAM [41]. By iterations, the valuable information that the NN focuses on is further enhanced. Meanwhile, the interfering information is also weakened, *i.e.*, the NN pays more attention to the target region on the input image rather than the surrounding environment. For example, the 4^{th} row in Fig. 4 labeled as 'horse', contains a rider and a horse. In the first iteration, the NN pays more attention to the rider. Then, the AN gradually guides the NN to shift its attention to the horse instead of the rider. Note that, by iterations, the astrocytes will gradually dwindle the influence on pre-neurons until completing the optimization of the communication. Hence, the maximum iteration time T can be estimated by minimizing the difference between the output of AN P^T, *i.e.*, $\min ||e^T - e^{T-1}||$, where $e^T = ||P^T - P^{T-1}||$ (Sec. 4.1).

The global connection mechanism. The AN integral weights of NN when updating their weights. Based on Eq. (6) and Eq. (4), the updating of the iteration is expressed



(a) Input (b) 25% (c) 50% (d) 75% (e) 100%

Figure 5. *We display the NN's (ResNet18 on CIFAR10) attention on the input image (a) with different percentages of NN connections that are involved varying from* 25% *to* 100% *(b)-(e).*

as follows:

$$P^1 = \mathcal{F}(\mathcal{G}_{avg}(W), W^A), \quad P^2 = \mathcal{F}(\mathcal{G}_{max}(W^1), W^A)$$
$$\cdots, \quad P^t = \mathcal{F}(\mathcal{G}_{max}(W^{t-1}), W^A)\ , \tag{8}$$

where $\mathcal{G}_{avg}(W)$ and $\mathcal{G}_{max}(W)$ is the feature matrix of W with the average and maximum connection intensity of the NN (see Sec. 4.1), respectively. The purpose of the $\mathcal{G}_{max}(\cdot)$ operation is to make AN focus on learning the connection that has the greatest impact on NN.

To illustrate the benefit of using the global connection mechanism, we randomly select 25%, 50%, 75% and 100% of connections in NN to establish bidirectional propagation with AN, and the rest connections remain in one-direction propagation. We take ResNet18 [19] as the NN on CIFAR10 dataset and visualize the results in Fig. 5. With more NN connections participating in bidirectional propagation with AN, the attention of the NN covers the target regions on the input images gradually.

However, using AN to integrate large amounts of weights in the NN is still time-consuming. For example, a 152-layer ResNet [19] has more than 60 million parameters. To this end, we need to compress the weights while preserving the connection information. We first squeeze the local receptive field into a channel descriptor. This is achieved by using the global average pooling or represented as the maximum connection of the neuron to generate channel-wise statistics [20]. Then, we use a simple single FCL to fuse multiple local receptive fields to obtain the inter-layer global receptive field (see Fig. 3). Finally, AN integrates global receptive fields from different layers, aiming to capture the correlations among all neurons fully. More details can be found in our supplementary material for different AN structures, such as UNet [40], fully convolutional network [31], and convolutional neural network [16].

3.3. Training Scheme

Inspired by the bilevel learning scheme [22], we first optimize the AN with the output of NN and then use the optimal AN to guide NN optimization.

3.3.1 Find the Optimal AN

By optimizing AN, we can get its optimal output: $\widetilde{P}$, then the optimal sub-network is found in the neural network N through $\tilde{P}$: $\tilde{\mathrm{N}} = \mathrm{N}\tilde{P}$, $\tilde{P} = k \times P + \sigma$, where k is the scaling factor and σ is the offset.

Inspired by [49], that optimize network weights and the network architecture by alternating gradient descent steps on the training set for weights and on the validation set for architectural parameters. We adopt an alternate optimization strategy for AN and NN in AstroNet to find the optimal AN. We optimize the task network NN on the training set and AN on the validation set. Note for a fair comparison, we split the original training set into a small training and validation set. The latter aims to make the optimized NN connections have good predictive ability for unseen data through bidirectional propagation between AN and NN. Furthermore, to simulate the temporal regulation mechanism of astrocytes and neurons, an optimization round of AN consists of multiple iterations. The optimization method is summarised in Algorithm 1, where T denotes the iteration time of bidirectional propagation.

Algorithm 1: Find the optimal AN.

Input: Training set $D^{Training}$, validation set $D^{Validation}$, Ground-truth label y_g
Output: The output of AstroNet: $\widetilde{y}$
Params: NN's params: W and AN's params: W^A
Setting: $\mathcal{L}$ corresponds to a loss function.

```
for r = 0 to R:
    if r%2 == 0:
        ỹ = H(D^Training(W ⊙ P))
        Optimize L_nn(ỹ, y_g) → Update W
    elif r%2! = 0:
        for t = 0 to T:
            if t < T:
                P^{t+1} = F(G(W^t), W^A)
                ỹ = H(D^Validation(W^t ⊙ P^{t+1}))
                Optimize L_an(ỹ, y_g) → Update W^A
            else:
                Break
```

Optimizing the NN stage. The learning objective of NN on the training set is the performance evaluation function $E(\cdot,\cdot)$ (*e.g.*, cross-entropy) of AstroNet, which is expressed as,

$$\mathcal{L}_{nn} = E(\mathcal{H}(x, W \odot P), y_g)\ , \tag{9}$$

Optimizing the AN stage. The objective of AN (*i.e.*, k and σ) on the validation set is the same as the one for NN, except for a constraint term that reduces the difference between e^*. This constraint term regularises our AN to gradually optimize NN steadily.

$$\begin{aligned} \mathcal{L}_{an} &= E(\mathcal{H}(x, W \odot P^t), y_g) + \lambda ||e^t - e^{t-1}||_2 \,, \\ e^t &= ||P_h^t - P_h^{t-1}||_2 \,, \\ P_h^t &= \mathcal{U}(p^t), \quad P_h^{t-1} = \mathcal{U}(p^{t-1}) \,, \end{aligned} \tag{10}$$

where $||\cdot||_2$ is the ℓ_2 norm, λ is the weight parameter, and $\mathcal{U}(\cdot)$ denotes the normalization function.

3.3.2 Use the Optimal AN to Guide NN Optimization

After obtaining the optimal AN, we fix the parameters of the AN and train the NN on the 'training + validation' set (the original training set). To this end, the optimal AN has the ability to guide the optimization of the NN connections toward generalization and compactness.

For NN training, our target is still the prediction performance of AstroNet. Therefore, the training objective of this stage is consistent with Eq. (9). After NN training, we remove the connection whose probability is less than the pruned threshold δ, and obtain the final sub-network weights W_f from the NN.

$$W_f = \begin{cases} w_i = 0, & \text{if } w_i \le \delta \\ w_i, & \text{otherwise} \,. \end{cases} \tag{11}$$

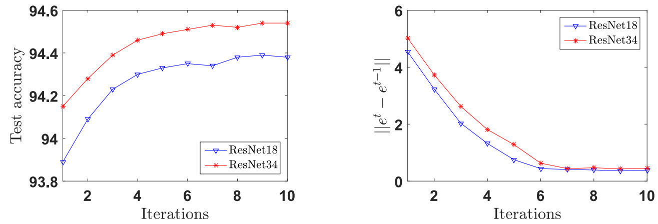
4. Experiments

We compare our AstroNet in the classification task on three datasets, ImageNet-1k [6], CIFAR [24], and MNIST [7] with state-of-the-art (SOTA) NAS and LSN methods. We also provide comparisons on the segmentation task to further demonstrate the efficiency of our AstroNet.

4.1. Experimental Setup

Implementation Details. Our AstroNet is implemented in PyTorch [23] and is trained via Adam optimizer with a batch size of 128 and a learning rate of $1e$-3. The $\lambda = 1e$-3 in Eq. (10). For the random initialization, we use Kaiming normal distribution [18] for CLs and fill FCLs with values drawn from the normal distribution ($\text{mean} = 0$, $\text{std}^2 = 0.01$). Our code and models will be made available to facilitate reproducible research.

Baselines. The NAS baselines include: 1) random search methods, RS and ReNAS [57]; 2) EA methods, REA [39], NPENAS [50] and Shapley-NAS [54]; 3) Reinforcement learning, REINFORCE [52] and ENAS [37]; 4) Differentiable methods, GDAS [9], SETN [8] and SDARTS-RS+PT [47]. For LSN, we adopte the ℓ_0 [32] and edge-



(a) (b)

Figure 6. *(a) Variation of test accuracy with iterations. (b) Variation of the difference between AN outputs with iterations.*

Table 1. *Comparison of global connection function $\mathcal{G}(\cdot)$ in iterations on CIFAR10, where NN is set to ResNet18.*

$\mathcal{G}(\cdot)$	Acc (%)	Params (M)
Global average pooling	94.32	**7.6**
Largest connection	94.31	7.8
Our (Mixture)	**94.35**	**7.6**

Table 2. *The accuracy of our AstroNet with respect to the pruned thresholds δ. The connections of the NN (ResNet18) in AstroNet are trained on CIFAR10.*

Threshold δ	0	1e-1	1e-2	1e-3	1e-4
Acc (%)	94.35	93.89	94.23	**94.35**	94.34
Params (M)	11.17	**4.3**	5.7	7.6	9.4

popup [38]. All NAS conduct experiments on NAS-Bench-201 [10], which provides a cell-based search space including 15625 different architectures. For LSN and our proposed AstroNet, the selection of fixed network architecture (or NN), follows [32, 43], which depends on the parameter requirements. The fixed architectures include ResNet [19], DenseNet-BC [21], MLP [48], and VGG-Conv6 [42].

Setups. First, we discuss the maximum iteration number T. In our AstroNet, the AN regulates weights in NN iteratively to find the optimal NN connection and achieve the best performance on the targeting task. To find a proper iteration number, we minimize the difference between the output of AN P^t with respect to the iteration times, *i.e.*, $\min ||e^t - e^{t-1}||$. In other words, when the changes between P^t and P^{t-1} become slight and stable, we suppose the AN finishing regulation and find the optimal sub-network of NN. Take NN as ResNet18 and ResNet34 as examples, Fig. 6a shows the test accuracy with respect to iterations and Fig. 6b shows the difference $||e^t - e^{t-1}||$ with respect to iterations. The results indicated that our AstroNet gradually optimizes the network connections and achieves accurate performance against the iteration times from 1 to 10. As shown in Fig. 6, the differences become stable when $t \ge 6$. Therefore, we set $T = 6$ for all our experiments.

Second, we discuss the selection in Tab. 1 global connection function $\mathcal{G}(\cdot)$ (see Sec. 3.2). Compared to only using $\mathcal{G}_{avg}(\cdot)$ or $\mathcal{G}_{max}(\cdot)$, our settings achieve the best performance. The results indicated that it is more reasonable

Table 3. *Comparison with LSN and NAS methods on CIFAR10 and CIFAR100. All baselines are trained on the 'training + validation' set of CIFAR10. Following [10], the time cost (time budget) of all baselines is first set to 3.7h. In contrast, our methods can find the optimal AN with a less time cost to achieve SOTA performance. Our (ReNet18) and Our (ReNet34) denote that the NN in AstroNet is set to ReNet18 and ReNet34, respectively. C-10 and C-100 denote results on CIFAR10 and CIFAR100. We highlight the best and the second-best numbers in* **bold** *and* underline*, respectively.*

Architecture	Top-1 Acc%		Params (M)		Time Cost (GPU hour)	Architecture Search method
	CIFAR10	CIFAR100	CIFAR10	CIFAR100		
ResNet18 [19]	92.80	72.22	11.17	11.22	-	Manual
ResNet34 [19]	93.56	72.86	21.28	21.33	-	
L_0 [32] (ResNet34)	93.37	71.69	10.4	10.8	-	LSN
edge-popup [38] (ResNet34)	93.52	71.97	9.4	9.7	-	
RS	93.70 ± 0.36	71.04 ± 1.07	4.1	4.3	3.7	Random Search
ReNAS [57]	93.99 ± 0.25	72.12 ± 0.79	4.5	4.9	3.7	
REA [39]	93.92 ± 0.30	71.84 ± 0.99	4.3	4.6	3.7	Evolution
NPENAS [50]	91.52 ± 0.16	-	3.8	4.2	3.7	
REINFORCE [52]	93.85 ± 0.37	71.71 ± 1.09	4.6	4.2	3.7	Reinforcement learning
ENAS [37]	54.30 ± 0.00	15.61 ± 0.00	4.8	5.1	3.7	
GDAS [9]	93.51 ± 0.13	70.61 ± 0.26	3.7	3.9	3.7	Gradient
SETN [8]	86.19 ± 4.63	56.87 ± 7.77	3.5	3.8	3.7	
Our (ResNet18)	94.49 ± 0.07	73.02 ± 0.27	7.3	7.4	3.7	Astrocyte-Neuron
Our (ResNet34)	$\mathbf{94.68} \pm 0.10$	$\mathbf{74.50} \pm 0.38$	9.2	9.2	3.7	
Our (ResNet18)	94.35 ± 0.14	72.77 ± 0.39	7.6	7.8	1.2 (C-10), 1.3 (C-100)	Astrocyte-Neuron
Our (ResNet34)	$\underline{94.51} \pm 0.13$	$\underline{74.26} \pm 0.47$	9.7	9.8	1.6 (C-10), 1.8 (C-100)	

Table 4. *Comparison results of ours, LSN, and NAS that are fully trained on CIFAR10. The fixed architecture of LSN and the NN of our AstroNet is set to DenseNet-BC.*

Architecture	Acc (%)	Params (M)	Time Cost (GPU days)
DenseNet-BC [21]	96.54	25.6	-
ℓ_0 [32]	96.41	11.3	-
edge-popup [38]	96.49	10.5	-
ReNAS [57]	<u>97.04</u>	3.8	4.5
REA [39]	96.66	3.4	7 (450 GPUs)
Our (DenseNet-BC)	**97.26**	10.4	**0.1**

first to learn the global features of each neuron and then gradually regulate the important connection.

Third, we study the threshold δ in Eq. (11). If δ is set to zero, the connections in the NN are not pruned. When $\delta > 0$, our method prunes low-probability connections ($0 < w_i \leq \delta$) in the trained NN. Tab. 2 reports the results of using a different δ with ResNet18 on the CIFAR10 dataset. When $\delta = 1e$-3, our method achieves the best accuracy, and we use this setting for all our experiments.

4.2. Results

Compare with Neural Architecture Search. We compare with SOTA NAS methods on CIFAR10 and CIFAR100. The search results on the testing set are shown in Tab. 3. The experiment settings follow commonly used standards [10, 57], *i.e.*, the total time budget is set to 15000s (3.7h).

Experimental results show that the proposed AstroNet achieves SOTA accuracy. Our optimization of AN needs only $\approx$ 1h, which outperforms the NAS in efficiency. Compare the top three NAS methods, ReNAS, REA, and REINFORCE in Tab. 3, our AstroNet achieves a relative improvement in accuracy by 0.52%, 0.59%, 0.66% on CIFAR10, and 2.14%, 2.42%, 2.55% on CIFAR100, respectively.

Meanwhile, we increase the searching time for optimizing AN to be consistent with the one of NAS, which is achieved by increasing the round for each learning rate and setting the cutoff learning rate to $1e$-6. In Tab. 3, our accuracy is additionally improved by 0.14% $\sim$ 0.25%. It demonstrates our AstroNet enables further refining subnetworks and accuracy improvement, similar to NAS methods, on the NN with the time cost increases. Moreover, an interesting observation is that by searching on the same NN, AstroNet can find a larger network structure for complex datasets, indicating that AstroNet can adaptively learn structure. This is reasonable as the network needs more connections to represent the complex features.

To further demonstrate the efficiency of our AstroNet, we choose the top-two rank NAS methods (ReNAS and REA) in Tab. 3. They are allowed to be fully trained to get higher-accuracy network architectures, and omit the limitation of time budget (3.7h). For a fair comparison, we also search in a larger NN architecture, DenseNet-BC. Results are shown in Tab. 4, where the efficiency of our method is significantly better than NAS. Our method achieves the best accuracy with a lower time cost, especially for a classical network of a larger size, and has more possibilities to discover its optimal combination of connections.

We trained the best-found architectures (both AstroNet, [47] and [54]) on CIFAR10 to evaluate their transferability on ImageNet-1k. Tab. 5 reports that the transferability of

Table 5. *Comparison with NAS methods for trasferability. AstroNet and NAS architectures are searched on CIFAR10 and then evaluated on ImageNet-1k. Our (ResNet50) denotes that the NN in AstroNet is set to ResNet50.*

Architecture	Acc (%)	Params (M)	Time Cost (GPU days)
ResNet50 [19]	75.3	25.6	-
SDARTS-RS+PT [47]	75.5	**4.7**	0.8
Shapley-NAS [54]	75.7	5.1	0.3
Ours (ResNet50)	**76.5**	7.6	**0.05**

Table 6. *Comparison of using AN to optimize NN with different sizes of parameters based on ResNet18 in CIFAR10. Taking '64-128-256-512' as an example, it denotes the output of the four residual blocks, respectively.*

Pattern	Architecture	Acc (%)	Params (M)	Time Cost (GPU hour)
Tiny-ResNet18	64-128-128-256	94.04	**4.8**	**0.9**
Orgn-ResNet18	64-128-256-512	94.35	7.6	1.2
Large-ResNet18	64-256-256-512	94.38	7.9	1.4

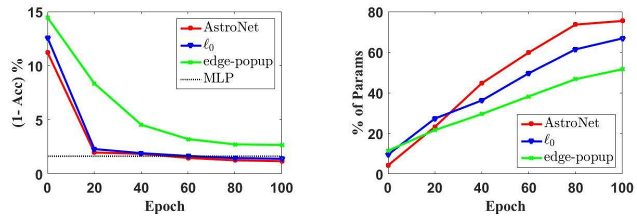
our AstroNet leads to a relative improvement in accuracy by 1.0% and 0.8% on the ImageNet-1k, respectively.

For the varying parameters, 1) our method searches the sub-network from a fixed NN, and the capacity of the sub-network is related to the size of the chosen NN. To this end, we show the results with different numbers of parameters (Tiny, Organ, and Large) based on ResNet18 in Tab. 6. Our AstroNet still achieves competitive performance on the tiny network, and the capacity of the obtained sub-network is also significantly reduced; 2) we focus on searching the optimal sub-network. The δ can be increased to decrease sub-network parameters while preserving accuracy (see Tab. 2).

Compare with the Learning Sparse Networks. We compared our AstroNet with SOTA baselines ℓ_0 and edge-popup with the fixed architecture of ResNet34 on CIFAR10 in Tab. 3, MLP on MNIST in Fig. 7, VGG-Conv6 on CIFAR10 and ResNet50 on ImageNet-1k in Tab. 7.

In Tab. 3, our AstroNet achieves the best accuracy. Fig. 7a shows the comparison of (1-Acc) between ℓ_0, edge-popup, and AstroNet, by using MLP as the fixed architecture (NN) along with epochs on MNIST. Our accuracy (98.78%) surpasses ℓ_0 (98.60%) by 0.18%, and edge-popup (97.32%) by 1.46%. Meanwhile, we report the pruning rate for (% of Params) in Fig. 7b, our method can prune $\approx$ 75.4% of the NN's parameters, which is higher than 66.8% for ℓ_0 and 51.6% for edge-popup.

For ImageNet-1k, our AstroNet outperforms the baseline and the GraSP by 1.2% and 2.48% in accuracy and reduces the capacity of the model by 70.31%. The edge-popup finds a sub-network on a fixed initialization architecture, and its results depend on the abundance of connections in the architecture [38]. Therefore, for a fair comparison, we compare with edge-popup on a larger network VGG-Conv6 in CIFAR10. Specifically, we set the threshold to $\delta = 1e\text{-}3$ to find the sub-network with 1.47% higher accuracy than edge-popup. When we continue to increase the threshold to $\delta = 1e\text{-}1$, we achieve higher compression than edge-popup and still have 0.46% better accuracy.



(a) (b)

Figure 7. *Comparisons of accuracy and pruning rate concerning the epochs for LSN baselines and ours with MLP as the fixed architecture on the MNIST dataset. (a) Results of* (1-Acc)*. (b) The percentage of pruning parameters (% of Params).*

Table 7. *Comparison on CIFAR10 and ImageNet-1k datasets. '% of Params' refers to the percentage of network parameters we pruned of the total number of parameters. Our (ReNet50) and Our (VGG-Conv6) denote that the NN in AstroNet is set to ReNet50 and VGG-Conv6, respectively.*

Method	$\delta = 1e\text{-}3$	Acc (%)	% of Params	Dataset
ResNet50 [19]	-	75.3	25.6	
GraSP [46]	-	74.02	8.7	ImageNet-1k
Ours (ResNet50)	$1e$-3	**76.5**	7.6	
VGG-Conv6 [42]	-	87.10	-	
edge-popup [38]	-	88.25	47.37	
ℓ_0 [32]	-	87.46	45.32	CIFAR10
Ours (VGG-Conv6)	$1e$-3	$\mathbf{89.72 \pm 0.11}$	31.80	
Ours (VGG-Conv6)	$1e$-1	$\underline{88.71} \pm 0.14$	**53.45**	

More. Please refer to our supplementary materials for more details about our AstroNet, *e.g.*, different values of λ and the attention map with different $\mathcal{G}$, experimental results on the Vision Transformer (ViT) model, and downstream task (segmentation).

5. Conclusions

In this paper, we propose an AstroNet to achieve structure learning by studying the astrocytes. By analyzing its bidirectional connection property, we formulate a temporal regulation mechanism and a global connection mechanism that allows AstroNet to regulate neuron connections adaptively. Experiments on the classification task demonstrate that our AstroNet can efficiently optimize the network structure while achieving state-of-the-art accuracy.

Acknowledgment. This research was supported in part by the National Natural Science Foundation of China [grant number 82171965]; Liyuan Pan's work was supported in part by the Beijing Institute of Technology Research Fund Program for Young Scholars.

References

[1] Jose M Alvarez and Mathieu Salzmann. Compression-aware training of deep networks. *Advances in neural information processing systems*, 30:856–867, 2017. 2

[2] M. Barlaud and F. Guyard. Learning sparse deep neural networks using efficient structured projections on convex constraints for green ai. In *International Conference on Pattern Recognition*, 2021. 1

[3] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *International conference on machine learning*, pages 550–559. PMLR, 2018. 2

[4] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Smash: one-shot model architecture search through hypernetworks. *arXiv preprint arXiv:1708.05344*, 2017. 2

[5] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018. 1

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[7] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 6

[8] Xuanyi Dong and Yi Yang. One-shot neural architecture search via self-evaluated template network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3681–3690, 2019. 6, 7

[9] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1761–1770, 2019. 2, 6, 7

[10] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv preprint arXiv:2001.00326*, 2020. 2, 6, 7

[11] Caitlin A Durkee and Alfonso Araque. Diversity and specificity of astrocyte–neuron communication. *Neuroscience*, 396:73–78, 2019. 2, 3

[12] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019. 2

[13] Giuliana Fossati, Michela Matteoli, and Elisabetta Menna. Astrocytic factors controlling synaptogenesis: a team play. *Cells*, 9(10):2173, 2020. 2

[14] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 1

[15] Aidan N. Gomez, Ivan Zhang, Siddhartha Rao Kamalakara, Divyam Madaan, Kevin Swersky, Yarin Gal, and Geoffrey E. Hinton. Learning Sparse Networks Using Targeted Dropout. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 1

[16] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018. 5

[17] Song Han, Jeff Pool, John Tran, and William J Dally. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*, 2015. 1

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 6

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5, 6, 7, 8

[20] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 5

[21] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 6, 7

[22] Simon Jenni and Paolo Favaro. Deep bilevel learning. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 632–648, Cham, 2018. Springer International Publishing. 5

[23] Nikhil Ketkar and Eder Santana. *Deep learning with Python*, volume 1. Springer, 2017. 6

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[25] Jiashi Li, Qi Qi, Jingyu Wang, Ce Ge, Yujian Li, Zhangzhang Yue, and Haifeng Sun. Oicsr: Out-in-channel sparsity regularization for compact deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7046–7055, 2019. 2

[26] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, pages 367–377. PMLR, 2020. 2

[27] Xiaoping Li, Yadi Wang, and Rubén Ruiz. A survey on sparse learning models for feature selection. *IEEE transactions on cybernetics*, 2020. 1, 2

[28] Yong Liang, Cheng Liu, Xin-Ze Luan, Kwong-Sak Leung, Tak-Ming Chan, Zong-Ben Xu, and Hai Zhang. Sparse logistic regression with a l1/2 penalty for gene selection in cancer classification. *BMC bioinformatics*, 14(1):1–12, 2013. 2

[29] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018. 1

[30] Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical representations for efficient architecture search. *arXiv preprint arXiv:1711.00436*, 2017. 1

[31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 5

[32] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017. 1, 2, 6, 7, 8

[33] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943. 2, 3

[34] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. Evolving deep neural networks. In *Artificial intelligence in the age of neural networks and brain computing*, pages 293–312. Elsevier, 2019. 2

[35] Marta Navarrete and Alfonso Araque. Endocannabinoids mediate neuron-astrocyte communication. *Neuron*, 57(6):883–893, 2008. 3

[36] Gertrudis Perea and Alfonso Araque. Glia modulates synaptic transmission. *Brain research reviews*, 63(1-2):93–102, 2010. 2, 3

[37] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, pages 4095–4104. PMLR, 2018. 1, 6, 7

[38] Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11893–11902, 2020. 2, 6, 7, 8

[39] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019. 2, 6, 7

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 5

[41] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 4

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6, 8

[43] Suraj Srinivas, Akshayvarun Subramanya, and R Venkatesh Babu. Training sparse neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 138–145, 2017. 6

[44] A. Subramaniam and A. Sharma. N2nskip: Learning highly sparse networks using neuron-to-neuron skip connections. In *British Machine Vision Conference*, 2020. 1

[45] Amirsina Torfi, Rouzbeh A Shirvani, Sobhan Soleymani, and Naser M Nasrabadi. Gasl: Guided attention for sparsity learning in deep neural networks. *arXiv preprint arXiv:1901.01939*, 2019. 2

[46] C. Wang, G. Zhang, and R. Grosse. Picking winning tickets before training by preserving gradient flow. In *Int. Conf. Learn. Represent.*, 2020. 8

[47] R. Wang, M. Cheng, X. Chen, X. Tang, and C. Hsieh. Rethinking architecture selection in differentiable nas. In *Int. Conf. Learn. Represent.*, 2021. 6, 7, 8

[48] Shengjie Wang, Haoran Cai, Jeff Bilmes, and William Noble. Training compressed fully-connected networks with a density-diversity penalty. 2016. 6

[49] Ye Wang, Amy KY Fu, and Nancy Y Ip. Instructive roles of astrocytes in hippocampal synaptic plasticity: neuronal activity-dependent regulatory mechanisms. *The FEBS Journal*, 289(8):2202–2218, 2022. 5

[50] Chen Wei, Chuang Niu, Yiping Tang, Yue Wang, Haihong Hu, and Jimin Liang. Npenas: Neural predictor guided evolution for neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 6, 7

[51] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29:2074–2082, 2016. 1, 2

[52] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992. 2, 6, 7

[53] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019. 1

[54] H. Xiao, Z. Wang, Z. Zhu, J. Zhou, and J. Lu. Shapleynas: Discovering operation contribution for neural architecture search. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11892–11901, 2022. 6, 7, 8

[55] Lingxi Xie and Alan Yuille. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388, 2017. 2

[56] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018. 2

[57] Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. Renas: Relativistic evaluation of neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4411–4420, 2021. 2, 6, 7

[58] Antoine Yang, Pedro M Esperança, and Fabio M Carlucci. Nas evaluation is frustratingly hard. *arXiv preprint arXiv:1912.12522*, 2019. 2

[59] Zhaohui Yang, Yunhe Wang, Xinghao Chen, Boxin Shi, Chao Xu, Chunjing Xu, Qi Tian, and Chang Xu. Cars: Continuous evolution for efficient neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1829–1838, 2020. 2

[60] 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. 2

[61] Jaehong Yoon and Sung Ju Hwang. Combined group and exclusive sparsity for deep neural networks. In *International Conference on Machine Learning*, pages 3958–3966. PMLR, 2017. 2

[62] Zhao Zhong, Zichen Yang, Boyang Deng, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Blockqnn: Efficient block-wise neural network architecture generation. *IEEE transactions on pattern analysis and machine intelligence*, 43(7):2314–2328, 2020. 1, 2

[63] Xiaotian Zhu, Wengang Zhou, and Houqiang Li. Improving deep neural network sparsity through decorrelation regularization. In *Ijcai*, pages 3264–3270, 2018. 1, 2

[64] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 2

[65] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018. 1, 2