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].

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% ~ 2.81%. While for NAS methods, the relative improvements are 0.22% ~ 2.79% in accuracy, and 3 ~ 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 \( \ell_1 \) 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 \( \ell_0 \) 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, 59], the time to find a proper
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:

$$ y = \phi \left( \sum_{i}^{N} x_i w_i \right), $$

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

$$ y = \phi \left( h \left( \sum_{i}^{N} x_i f(w_i) \right) \right), $$

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 ‘$N_i - A$’ from

![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 {NT}$^{l}_{1}$.](image)

Then, pre-neurons update their neurotransmitters to {NT}$^{f}_{t+1}$ and stimulate the astrocyte again. When the differences between neurotransmitters from time $t - 1$ to $t$ are

![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.](image)
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 \( X = [x_1, x_2, \ldots, x_N]^T \) be the data. The traditional neural network is modeled as,

\[
y = \phi(XW),
\]

where \( W = [w_1, w_2, \ldots, w_N]^T \) 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,

\[
\begin{align*}
    y &= \mathcal{H}(XW^t), \\
    W^t &= W^{t-1} \odot P^t, \\
    P^t &= \mathcal{F}(\mathcal{G}(W^{t-1}), W^A),
\end{align*}
\]

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

\[
W^1 = W \odot P^1, \quad W^2 = W^1 \odot P^2, \quad \ldots, \quad W^t = W^{t-1} \odot P^t.
\]

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 as

\[
\begin{align*}
    W^1 &= W \odot P^1, \\
    W^2 &= W^1 \odot P^2, \\
    \ldots, \\
    W^t &= W^{t-1} \odot P^t.
\end{align*}
\]
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:

\[ P^1 = F(G_{avg}(W), W^A), \quad P^2 = F(G_{max}(W^1), W^A) \]
\[ \cdots, \quad P^t = F(G_{max}(W^{t-1}), W^A), \quad (8) \]

where \( G_{avg}(W) \) and \( G_{max}(W) \) 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 \( G_{max}(\cdot) \) 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: \( \tilde{P} \), then the optimal sub-network is found in the neural network \( N \) through \( \tilde{P} \): \( \tilde{N} = N\tilde{P} \), \( \tilde{P} = k \times P + \sigma \), where \( k \) is the scaling factor and \( \sigma \) 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 summarized 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: \( y \)

**Params:** NN’s params: \( W \) and AN’s params: \( W^A \)

**Setting:** \( \mathcal{L} \) corresponds to a loss function.

1. for \( r = 0 \) to \( R \):
   2. if \( r \% 2 == 0 \):
      3. \( \bar{y} = \mathcal{H}(D^{Training}(W \odot P)) \)
      4. Optimize \( \mathcal{L}_{nn}(\bar{y}, y_g) \) → Update \( W \)
   5. else \( r \% 2! == 0 \):
      6. for \( t = 0 \) to \( T \):
         7. if \( t < T \):
            8. \( P^{t+1} = F(G(W^t), W^A) \)
            9. \( \bar{y} = \mathcal{H}(D^{Validation}(W^t \odot P^{t+1})) \)
         10. Optimize \( \mathcal{L}_{an}(\bar{y}, y_g) \) → Update \( W^A \)
      11. else:
         12. 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,

\[
\mathcal{L}_{nn} = E(\mathcal{H}(x, W \odot P), y_g),
\]
Optimizing the AN stage. The objective of AN (i.e., $k$ and $\sigma$) 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.

$$L_{an} = E(H(x,W \odot P^t), y_g) + \lambda \| e^t - e^{t-1} \|_2,$$

$$e^t = \| P^t_h - P^{t-1}_h \|_2,$$

$$P^t_h = U(p^t), P^{t-1}_h = U(p^{t-1}),$$

where $\| \cdot \|_2$ is the $\ell_2$ norm, $\lambda$ is the weight parameter, and $U(\cdot)$ 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 AN connections toward generalization and compactness.

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

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

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 (mean = 0, std = 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 adopt the $\ell_0$ [32] and edge-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 $t \geq 6$. Therefore, we set $T = 6$ for all our experiments.

Second, we discuss the selection in Tab. 1 global connection function $G(\cdot)$ (see Sec. 3.2). Compared to only using $G_{avg}(\cdot)$ or $G_{max}(\cdot)$, our settings achieve the best performance. The results indicated that it is more reasonable
Our optimization of AN needs \( i.e. [10, 57], \) and achieves SOTA accuracy. The experiment settings follow commonly used standards with SOTA NAS methods on CIFAR10 and CIFAR100. Compare with Neural Architecture Search.

### 4.2. Results

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>
<thead>
<tr>
<th>Architecture</th>
<th>Top-1 Acc%</th>
<th>Params (M)</th>
<th>Time Cost (GPU hour)</th>
<th>Architecture Search method</th>
</tr>
</thead>
<tbody>
<tr>
<td>L_{0} [32] (ResNet34)</td>
<td>93.37</td>
<td>71.69</td>
<td>10.4</td>
<td>10.8</td>
</tr>
<tr>
<td>edge-popup [38] (ResNet34)</td>
<td>93.52</td>
<td>71.97</td>
<td>9.4</td>
<td>9.7</td>
</tr>
<tr>
<td>RS</td>
<td>93.70 ± 0.36</td>
<td>71.04 ± 1.07</td>
<td>4.1</td>
<td>4.3</td>
</tr>
<tr>
<td>REA [39]</td>
<td>93.92 ± 0.30</td>
<td>71.84 ± 0.99</td>
<td>4.3</td>
<td>4.6</td>
</tr>
<tr>
<td>NPNAS [50]</td>
<td>91.52 ± 0.16</td>
<td>-</td>
<td>3.8</td>
<td>4.2</td>
</tr>
<tr>
<td>REINFORCE [32]</td>
<td>93.85 ± 0.37</td>
<td>71.71 ± 1.09</td>
<td>4.6</td>
<td>4.2</td>
</tr>
<tr>
<td>ENAS [37]</td>
<td>54.30 ± 0.00</td>
<td>15.61 ± 0.00</td>
<td>4.8</td>
<td>5.1</td>
</tr>
<tr>
<td>GDAS [9]</td>
<td>93.51 ± 0.13</td>
<td>70.61 ± 0.26</td>
<td>3.7</td>
<td>3.9</td>
</tr>
<tr>
<td>SETN [8]</td>
<td>86.19 ± 4.63</td>
<td>56.87 ± 7.77</td>
<td>3.5</td>
<td>3.8</td>
</tr>
<tr>
<td>Our (ResNet18)</td>
<td>94.19 ± 0.07</td>
<td>73.02 ± 0.27</td>
<td>7.3</td>
<td>7.4</td>
</tr>
<tr>
<td>Our (ResNet34)</td>
<td>94.68 ± 0.10</td>
<td>74.50 ± 0.38</td>
<td>9.2</td>
<td>9.2</td>
</tr>
<tr>
<td>Our (ResNet18)</td>
<td>94.35 ± 0.14</td>
<td>72.77 ± 0.39</td>
<td>7.6</td>
<td>7.8</td>
</tr>
<tr>
<td>Our (ResNet34)</td>
<td>94.51 ± 0.13</td>
<td>71.26 ± 0.47</td>
<td>9.7</td>
<td>9.8</td>
</tr>
</tbody>
</table>

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

Third, we study the threshold \( \delta \) in Eq. (11). If \( \delta \) 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 \( \delta \) 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

#### 4.2.1. 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. Comparing 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 sub-networks 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 transferability. 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.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Acc (%)</th>
<th>Params (M)</th>
<th>Time Cost (GPU days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet50 [19]</td>
<td>75.3</td>
<td>25.6</td>
<td>-</td>
</tr>
<tr>
<td>SDARTS-RS+PT [47]</td>
<td>75.5</td>
<td>4.7</td>
<td>0.8</td>
</tr>
<tr>
<td>Shapley-NAS [54]</td>
<td>75.7</td>
<td>5.1</td>
<td>0.3</td>
</tr>
<tr>
<td>Ours (ResNet50)</td>
<td>76.5</td>
<td>7.6</td>
<td>0.05</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Architecture</th>
<th>Acc (%)</th>
<th>Params (M)</th>
<th>Time Cost (GPU hour)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tiny-ResNet18</td>
<td>64-128-256-256</td>
<td>94.04</td>
<td>4.8</td>
<td>0.9</td>
</tr>
<tr>
<td>Orgn-ResNet18</td>
<td>64-128-256-512</td>
<td>94.35</td>
<td>7.6</td>
<td>1.2</td>
</tr>
<tr>
<td>Large-ResNet18</td>
<td>64-256-256-512</td>
<td>94.38</td>
<td>7.9</td>
<td>1.4</td>
</tr>
</tbody>
</table>

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 $\delta$ 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 $\ell_0$ 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 $\ell_0$, edge-popup, and AstroNet, by using MLP as the fixed architecture (NN) along with epochs on MNIST. Our accuracy (98.78%) surpasses $\ell_0$ (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 $\ell_0$ 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 $\delta = 1e-3$ to find the sub-network with 1.47% higher accuracy than edge-popup. When we continue to increase the threshold to $\delta = 1e-1$, we achieve higher compression than edge-popup and still have 0.46% better accuracy.

More. Please refer to our supplementary materials for more details about our AstroNet, e.g., different values of $\lambda$ and the attention map with different $G$, 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


[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


[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


