biomimetics

MDPI

Article

Deep Learning and Neural Architecture Search for Optimizing Binary Neural Network Image Super Resolution

Yuanxin Su [1,2], Li-minn Ang [3,*], Kah Phooi Seng [1,3] and Jeremy Smith [2]

[1] XJTLU Entrepreneur College (Taicang), Xi'an Jiaotong Liverpool University, Taicang 215400, China
[2] Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, UK
[3] School of Science, Technology and Engineering, University of the Sunshine Coast,
Moreton Bay, QLD 4502, Australia
* Correspondence: lang@usc.edu.au

Abstract: The evolution of super-resolution (SR) technology has seen significant advancements through the adoption of deep learning methods. However, the deployment of such models by resource-constrained devices necessitates models that not only perform efficiently, but also conserve computational resources. Binary neural networks (BNNs) offer a promising solution by minimizing the data precision to binary levels, thus reducing the computational complexity and memory requirements. However, for BNNs, an effective architecture is essential due to their inherent limitations in representing information. Designing such architectures traditionally requires extensive computational resources and time. With the advancement in neural architecture search (NAS), differentiable NAS has emerged as an attractive solution for efficiently crafting network structures. In this paper, we introduce a novel and efficient binary network search method tailored for image super-resolution tasks. We adapt the search space specifically for super resolution to ensure it is optimally suited for the requirements of such tasks. Furthermore, we incorporate Libra Parameter Binarization (Libra-PB) to maximize information retention during forward propagation. Our experimental results demonstrate that the network structures generated by our method require only a third of the parameters, compared to conventional methods, and yet deliver comparable performance.

Keywords: deep learning; neural architecture search; binary neural network; image super resolution

check for updates

Citation: Su, Y.; Ang, L.-m.; Seng, K.P.; Smith, J. Deep Learning and Neural Architecture Search for Optimizing Binary Neural Network Image Super Resolution. *Biomimetics* **2024**, *9*, 369. https://doi.org/10.3390/biomimetics9060369

Academic Editor: Yongquan Zhou

Received: 6 May 2024
Revised: 11 June 2024
Accepted: 15 June 2024
Published: 18 June 2024

1. Introduction

Super resolution (SR) is an essential task in computer vision, aimed at designing effective models to reconstruct high-resolution (HR) images from low-resolution (LR) ones. It plays a vital role across various domains, such as medical imaging [1,2], biometric information identification [3–5], and astronomical images [6].

Traditional SR interpolation methods [7] offer rapid processing, but fall short in terms of accuracy. The field has thus evolved from these methods to adopting deep learning techniques, such as [8–12], which provide enhanced performance. Nevertheless, the growing complexity and escalated computational demands of deep neural networks make it difficult to deploy these models on devices with limited resources. As network architecture search has advanced, the introduction of DARTS [13] has significantly lowered the barrier to entry for this technology. This development makes differentiable NAS an attractive option for achieving lightweight super resolution (SR), particularly because manually designing lightweight methods often involves considerable time spent on trial and error. DARTS can relieve designers from the cumbersome process of manual design. This method has enabled efficient searches for neural network architectures within a continuous search space. DARTS models neural architecture search (NAS) as a bi-level optimization problem, employing alternate optimization through gradient descent to derive the optimal network architecture. DARTS is also categorized under one-shot NAS methods, which involve constructing a super net and then extracting the best sub-network from it. This approach addresses the traditional black box nature of network structure searches, making them more transparent and detailed. The key lies in its use of soft functions to mix candidate operations. The gradient optimization approach pioneered by DARTS demonstrates remarkable results, evident from the minimal GPU days required to see outcomes [14]. This method stands out from other early NAS techniques [15–17], particularly those based on reinforcement learning or evolutionary algorithms, because it is not constrained by the intrinsic discrete optimization nature of these methods. Fundamentally, it addresses the issue that previous NAS methods required extensive computational resources.

For lightweight models, an optimal network structure is just one part of the equation. Introducing specific data types can also make networks more efficient and lightweight. Binary neural networks (BNNs), which map full-precision data to binary values $\{-1, 1\}$ [18], are particularly hardware friendly. This attribute not only speeds up processing, but also decreases memory pressure, significantly easing the load on hardware resources during model computation. However, at present, most research on binary networks focuses on adaptations of established full-precision architectures, including ResNet [19] and DenseNet [20]. Notable examples include Bi-Real net [21], binary DenseNet [22], and the Binarized Ghost Module (BGM) [23]. Although Bethge et al. [22] proposed architectural design principles through extensive experimentation and demonstrated their effectiveness in creating efficient new architectures, manually designing an appropriate architecture remains a task that rapidly depletes both resources and time. To address the complexities of designing the architecture of BNNs, researchers have begun to explore the application of NAS in BNNs. Nevertheless, nowadays, most of the existing work on NAS only focuses on real-valued architecture design. Using real-valued NAS strategies in binary domains often results in inadequate convergence. Prior research has validated this issue, highlighting the following principal reasons behind it:

1. An unsuitable search space for BNNs;
2. Conducting architecture searches with binary weights and activations can result in topological degeneration or the training merely converging to an extremely low accuracy;
3. Mismatches between the search process and evaluation;
4. The imbalanced selection of operations.

Although they each proposed solutions designed to enhance model performance in image classification, these methods might not be equally effective for super-resolution tasks and could potentially degrade the performance. Therefore, we have revised the original solutions to better suit the search for optimal binary network structures specifically for super-resolution tasks.

In summary, our main contributions include the following:

1. We define a friendly search space for BNNs for the task of super resolution;
2. We further stabilize the search process by applying L1 and L2 regularization, making the selection of operations more equitable during the search process. Additionally, we explore the effects of different combinations of L1 and L2 regularization;
3. We modify the basic architecture of the model to preserve image information through a hierarchical basic architecture, as much as possible;
4. Given the unique demands of super-resolution (SR) tasks, which differ from image classification in that SR models need to preserve information to the greatest extent possible, we introduce Libra Parameter Binarization (Libra-PB) [24] to maximize the retention of information during forward propagation. For backward propagation, the introduction of the Error Decay Estimator (EDE) helps the model effectively manage the loss of information caused by the reduced parameter update capability outside the truncation range and the approximate errors within it. These measures are designed to minimize the mismatch between the search and evaluation;
5. Our approach is capable of generating superior BNN architectures for super-resolution tasks, with relatively low computational expenditure.

The paper is structured as follows. The earlier sections provide background information on the domain areas of the research. Section 2 presents the discussions on image super resolution, Section 3 presents the discussions on binary neural networks, and Section 4 presents the discussions on neural architecture search. Section 5 discusses the methodology used in the paper to develop a deep learning and neural architecture search that is targeted towards optimizing binary neural network image super-resolution tasks. Section 6 discusses the experimental results to validate the proposed approaches. This section also provides comparisons with other approaches to image super resolution (SR). Some concluding remarks are given in Section 7.

2. Image Super Resolution

The SRCNN [8] marks the pioneering work of deep learning in super-resolution reconstruction. The network structure of SRCNN is very simple, utilizing just three convolutional layers. The network architecture is illustrated in the figure below. SRCNN first enlarges the low-resolution image to the target size using bicubic interpolation, then it applies a three-layer convolutional network to approximate the nonlinear mapping, and finally outputs the high-resolution image result. In this paper, the author interprets the three-layer convolutional structure as three steps: the extraction and feature representation of the image blocks, the nonlinear mapping of the features, and the final reconstruction.

The FSRCNN [25] and SRCNN are works by Dong Chao, Xiaoou Tang, and others from the Chinese University of Hong Kong. FSRCNN is an improvement on the previous SRCNN in regard to three main aspects. First, it uses a deconvolution layer at the end to scale up the size, allowing the original low-resolution image to be directly input into the network, rather than requiring enlargement through the bicubic method as in SRCNN. Second, it modifies the feature dimensions, uses smaller convolutional kernels, and employs more mapping layers. Third, it allows for the sharing of mapping layers; models trained for different upsampling rates can simply fine tune the final deconvolution layer.

Methods like SRCNN require low-resolution images to be upscaled via interpolation, to match the size of high-resolution images before being input into the model. This necessitates performing convolution operations at a higher resolution, thereby increasing the computational complexity. Shi et al. [26] proposed an efficient method that extracts features directly at the low-resolution image size and computes the high-resolution image. The core concept of ESPCN is the sub-pixel convolutional layer. The input into the model is the original low-resolution image, which, after passing through three convolutional layers, produces a feature map with the same size as the input image but with a channel count of r^2. These features are then rearranged via the sub-pixel convolutional layer to form a high-resolution image.

The DRCN [9] was the first to apply an existing recursive neural network structure to super-resolution processing. It also utilized the concept of residual learning to deepen the network, which enlarged the layers' receptive field and improved the performance. The authors of the DRRN [10] utilized a deeper network structure to achieve performance enhancements. In the DRRN, each residual unit shares the same input, which is the output from the first convolutional layer in the recursive block. Each residual unit contains two convolutional layers. Within a recursive block, the convolutional layers at corresponding positions in each residual unit share parameters (represented by the light green or light red blocks in the diagram).

The model proposed by Tong is called SRDenseNet [27], which utilizes dense blocks as the basic structure and employs skip connections to combine low-level feature information with high-level feature information. Subsequently, image reconstruction is carried out through a deconvolution network, facilitating the transformation from low resolution to high resolution. Furthermore, it is also highlighted that the information contained in the features across different depth layers is complementary.

Xin et al. [28] designed a Bit Accumulation Mechanism (BAM), using a value accumulation scheme to approximate full-precision convolution, refining the quantization precision progressively along the direction of the model inference. They also proposed an efficient model architecture based on BAM, named the Binary Super-Resolution Network (BSRN), to reduce the computational complexity and parameters. In experiments, they implemented BAM into VDSR and SRResNet to demonstrate the effectiveness of their method and compared it with the BSRN.

3. Binary Neural Networks

Binary neural networks (BNNs) refer to neural networks that use only two values, $\{+1, -1\}$, to represent weights and activations. Compared to full-precision neural networks, BNNs can replace the 32-bit float multiplication and accumulation, typically used in convolution operations with a much simpler combination of XNOR and popcount operations. Significantly, this substitution saves memory and computational resources, facilitating the deployment of models by resource-constrained devices. However, due to the limited amount of information that binary values can represent, BNNs have historically exhibited much lower accuracy than full-precision models. Nevertheless, recent studies such as those on MeliusNet [29], IRNet [24], and ReActNet [30], have made substantial efforts to improve BNNs, achieving over 70% top-1 accuracy on the ImageNet dataset.

The concept of binary neural networks (BNNs) originated from BinaryConnect [18], and was proposed by Courbariaux et al. To address the issue of gradient propagation in binarized weights, the authors propose maintaining a set of real-valued weights during training and then using the sign function to obtain the binarized weights. The binarization can be given by:

$$B_w = Sign(R_w) \tag{1}$$

where B_w and R_w denote the binarized and real-valued weights, respectively.

To overcome the issue where the sign function is non-differentiable at zero and has a derivative of zero elsewhere, which hinders effective gradient propagation, the authors designed the Straight-Through Estimator (STE).

XNOR-Net [31] builds upon the original BNN framework by accounting for quantization errors and introduces the use of scaling factors. Each output channel direction of the real-valued weights is associated with a scaling factor to restore the information lost in binarized weights. Similarly, each pixel in the height and width direction of the activation is associated with a scaling factor to recover the information lost in the binarized activations. These scaling factors do not require learning; they can be directly determined by calculating the corresponding L-1 norm. This method does not compromise the efficiency of binary convolution operations. Experimental results show significant improvements over the original BNN and, for the first time, demonstrate the performance of BNNs on a large dataset like ImageNet. In convolution operations, XNOR-Net achieves a speed-up of up to 58 times and saves 32 times the memory compared to traditional methods.

Liu et al. [30] initially adopted the concept from Bi-Real Net [21], which involves incorporating shortcut layers into the original network, to modify MobileNetV1. Through extensive experimentation, they found that the performance of BNNs is particularly sensitive to changes in the distribution of activations, specifically noting that shifts in and the scaling of activations have a significant impact on BNN performance. Consequently, the authors considered that each layer's activations might have an optimal offset and scaling value that would maximize the model's overall performance. This led to the proposal to modify the sign and PReLU functions to include learnable parameter variables, allowing the model to automatically learn the best offset and scaling values for each layer. These were named the ReAct-Sign (abbreviated as RSign) and ReAct-PReLU (abbreviated as RPReLU).

4. Neural Architecture Search

The architecture of a neural network, a critical component of deep learning models, plays a pivotal role in determining model performance. For instance, ResNet [19] (residual network) has greatly advanced image processing technology by addressing the issue of degradation during deep network training through the introduction of residual connections.

However, as models become increasingly complicated, the performance enhancement also demands precise adjustments of the hyperparameters. Minor variations in the hyperparameters can significantly impact model performance, making experimental results difficult to replicate. This not only adds to the burden of researchers, but also raises the barrier to entry into the field. So, just as steam engines were gradually replaced by electric motors, the design of neural network architecture is transitioning from manual to automated design by machines. A landmark event in this process occurred in 2016, when Zoph and Le [15] utilized reinforcement learning for neural architecture search (NAS) and achieved superior performance on image classification and language modelling tasks compared to previous manually designed networks. This development highlights a significant shift towards automating the design process in the field of neural networks, promising enhancements in the efficiency and effectiveness of model architectures. With the development of NAS, automatic design using the NAS method for specific tasks has begun to outperform the finest architectures designed by humans in several domains, such as object detection [32,33], semantic segmentation [34,35], protein folding [36,37], and weather prediction.

As previously mentioned, network architectures can relieve designers of the cumbersome process of manual design. Particularly, the recently introduced gradient-based methods, such as DARTS [13], have enabled efficient searches for neural network architectures within a continuous search space. DARTS models neural architecture search (NAS) as a bi-level optimization problem, employing alternate optimization through gradient descent to derive the optimal network architecture.

DARTS is also categorized under one-shot NAS methods, which involve constructing a super net and then extracting the best sub-network from it. This approach addresses the traditional black box nature of network structure searches, making them more transparent and detailed. The key lies in its use of soft functions to mix candidate operations. The gradient optimization approach pioneered by DARTS demonstrates remarkable results, evident from the minimal GPU days required to see outcomes [14]. This method stands out from other early NAS techniques [15–17], particularly those based on reinforcement learning or evolutionary algorithms, because it is not constrained by the intrinsic discrete optimization nature of these methods. Fundamentally, it addresses the issue that previous NAS methods required extensive computational resources. Figure 1 presents the flow of DARTS.

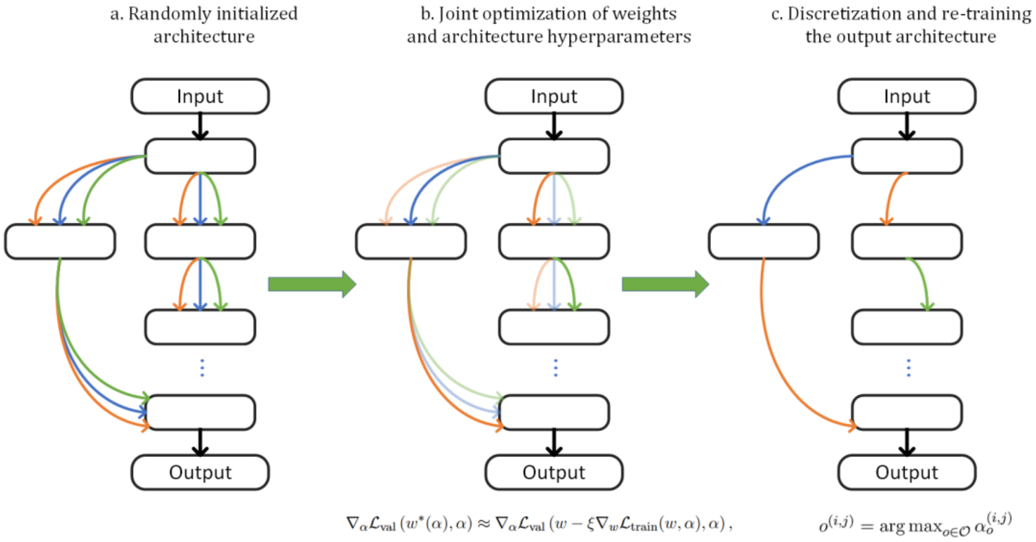

a. Randomly initialized architecture
b. Joint optimization of weights and architecture hyperparameters
c. Discretization and re-training the output architecture
Input
Input
Input
Output
Output
Output
$\nabla_\alpha \mathcal{L}_{\text{val}}(w^*(\alpha),\alpha) \approx \nabla_\alpha \mathcal{L}_{\text{val}}(w - \xi \nabla_w \mathcal{L}_{\text{train}}(w,\alpha),\alpha)$,
$o^{(i,j)} = \arg\max_{o\in\mathcal{O}} \alpha_o^{(i,j)}$

Figure 1. DARTS flow.

Since the introduction of DARTS, numerous researchers have proposed improvements, aiming to achieve better network architectures. In P-DARTS [38], researchers proposed a progressive approach to alleviate the depth gap in DARTS, and they also improved the stability of the architectures derived from the search by implementing skip-connect regularization. PC-DARTS [39] aims to significantly reduce the computational requirements and memory usage, allowing for faster searches with larger batch sizes. The authors designed a channel-based sampling mechanism, where only a small fraction of 1/K of the channels in a node are used for the operation search, reducing the memory usage by (K − 1)/K, thereby enabling an increase in the batch size by a factor of K. To address the instability caused by channel sampling, they introduced edge normalization, which reduces uncertainty during the search by learning the edge-level hyperparameters of the super net. Xue and Qin introduced a method called ADARTS [40], a differentiable neural architecture search based on channel attention, which utilizes partial channel connections. By using an attention mechanism, it selects channels with higher importance to be sent to the operation space, whereas the rest of the channels are directly connected with the processed channels. This approach has been shown to greatly improve search efficiency and memory usage, and to reduce instability in network structures that typically arises from random channel selection.

Courbariaux proposed a method using binary values to represent weights and activations in neural networks, which was later defined as the first binary neural network. In Courbariaux's approach, through (1), full-precision weights and activation values are converted to binary values {−1, 1}, significantly reducing the hardware burden. Subsequently, multiplication operations within the model can be replaced with XNOR-popcount operations, which are described as follows:

$$X * W \approx sign(X) \odot sign(W) = B_x \odot B_w \tag{2}$$

where X and W denote inputs and weights of the convolutional layer, respectively. And $*$ represents the convolution operation. The XNOR-popcount operation is denoted by $\odot$.

A crucial consideration is that while binary neural networks (BNNs) offer advantages in terms of storage and computation speed, it must be acknowledged that model performance inevitably declines due to insufficient information representation of binary values. This issue can be mitigated through well-designed network architectures. However, to ascertain whether a structure can enhance neural network performance, extensive experimentation is required. Hence, the relevance of binary network architecture search techniques. The BNAS model [41] introduced a new binary search space and cell template, rediscovered the utility of the Zeroise layer, and implemented diversity regularizers to search for binary structures with improved performance. In BATS [42], in addition to designing a new binary-oriented search space, it also introduced a softmax with a temperature coefficient to foster more discriminative NAS.

5. Method

In the overall network framework and the basic structure of cells, we have made specific adjustments for super-resolution tasks to ensure the model's suitability for such applications. A basic architecture is shown in Figure 2.

The fundamental concept of DARTS is to learn the model structure within a differentiable search space, seeking the best combination of operations by optimizing a parameterized search space. Within DARTS, operation selection is accomplished via the softmax function, which converts operation weights into a probability distribution. The operation selection can be expressed as:

$$P^o = \frac{\exp({\alpha^o}_{i,j})}{\sum_{o' \in O} \exp({\alpha^{o'}}_{i,j})} \tag{3}$$

where ${\alpha^o}_{i,j}$ is the architecture parameter associated with the weight operation o. P^o denotes the probability of operation o, which is based on the operation weights ${\alpha^o}_{i,j}$ calculated by the softmax function.

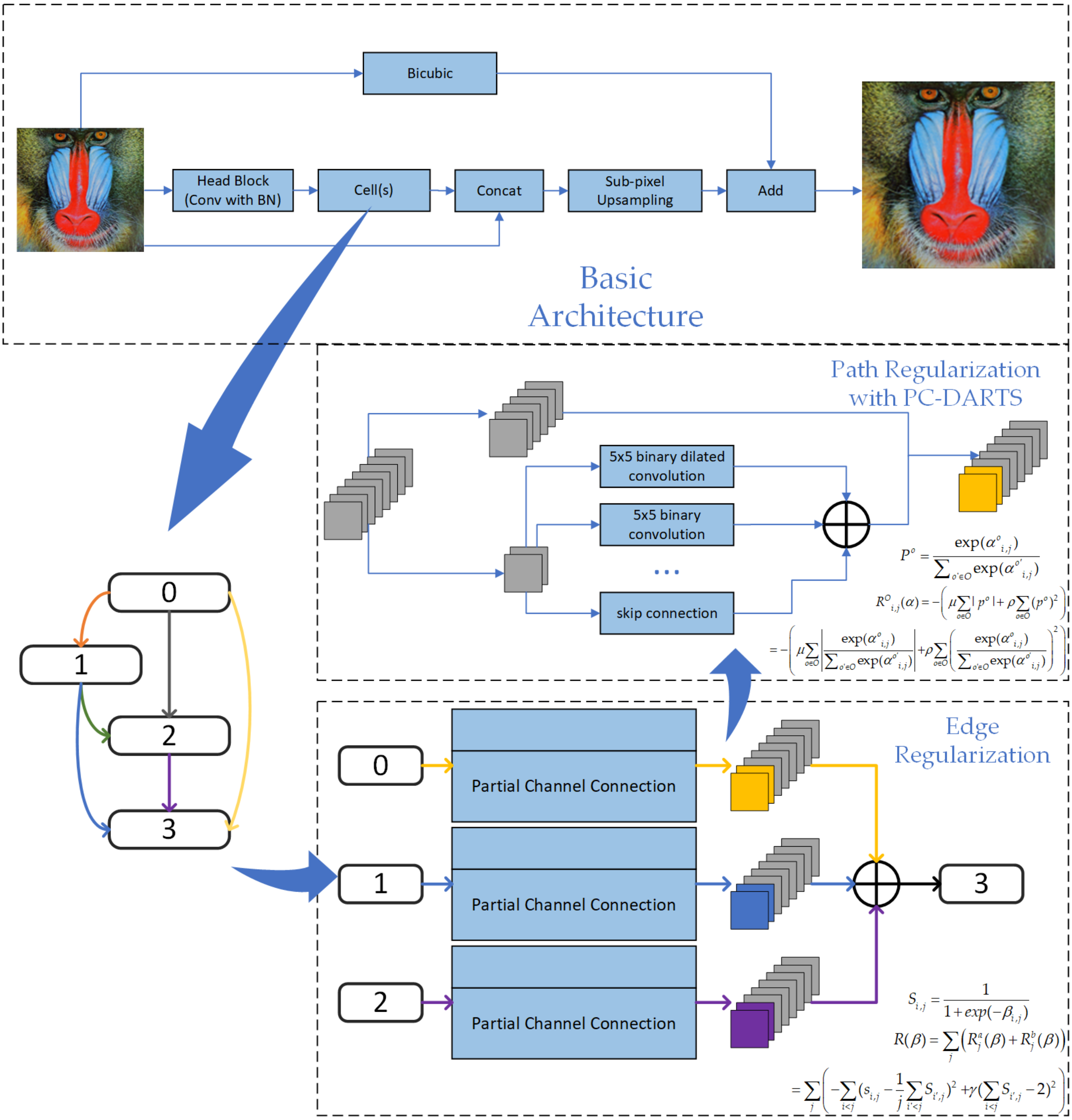

Bicubic
Head Block (Conv with BN)
Cell(s)
Concat
Sub-pixel Upsampling
Add
Basic Architecture
Path Regularization with PC-DARTS
5x5 binary dilated convolution
5x5 binary convolution
…
skip connection
$P^o = \frac{\exp(\alpha^o{}_{i,j})}{\sum_{o'\in O}\exp(\alpha^{o'}{}_{i,j})}$
$R^O{}_{i,j}(\alpha) = -\left(\mu\sum_{o\in O}|p^o| + \rho\sum_{o\in O}(p^o)^2\right)$
$= -\left(\mu\sum_{o\in O}\left|\frac{\exp(\alpha^o{}_{i,j})}{\sum_{o'\in O}\exp(\alpha^{o'}{}_{i,j})}\right| + \rho\sum_{o\in O}\left(\frac{\exp(\alpha^o{}_{i,j})}{\sum_{o'\in O}\exp(\alpha^{o'}{}_{i,j})}\right)^2\right)$
0
1
2
3
Edge Regularization
Partial Channel Connection
Partial Channel Connection
Partial Channel Connection
$S_{i,j} = \frac{1}{1+exp(-\beta_{i,j})}$
$R(\beta) = \sum_j \left(R^a_j(\beta) + R^b_j(\beta)\right)$
$= \sum_j \left(-\sum_{i<j}(s_{i,j} - \frac{1}{j}\sum_{i'<j} S_{i',j})^2 + \gamma(\sum_{i<j} S_{i',j} - 2)^2\right)$

Figure 2. Search framework and architecture.

In DARTS, the architectural parameters associated with different operations exhibit minimal changes, and even slight fluctuations can induce alterations in the cells. Therefore, in our framework, we introduce L1 and L2 regularization to enhance the robustness of the entire search framework. The regularization term can be expressed as:

$$
\begin{aligned}
R^O{}_{i,j}(\alpha) &= -\left(\mu \sum_{o\in O} |p^o| + \rho \sum_{o\in O} (p^o)^2\right) \\
&= -\left(\mu \sum_{o\in O} \left|\frac{\exp(\alpha^o{}_{i,j})}{\sum_{o'\in O}\exp(\alpha^{o'}{}_{i,j})}\right| + \rho \sum_{o\in O} \left(\frac{\exp(\alpha^o{}_{i,j})}{\sum_{o'\in O}\exp(\alpha^{o'}{}_{i,j})}\right)^2\right)
\end{aligned}
\tag{4}
$$

For L1 regularization in operation selection, it can be used to encourage the search framework to produce sparse operation selections, thereby achieving more efficient and stable operation selection. By adding the L1 regularization term to the loss function, the weights of some unimportant operations tend toward zero, thereby reducing the model complexity and improving the generalization ability. But applying only L1 regularization may result in the weight of some operations becoming very close to zero, which means that these operations are ignored during the search process, eventually leading to skip connect for most operations. Hence, we also introduce L2 regularization with balancing factor ρ to avoid the negative effect of L1 regularization. By controlling the strength of L2 regularization, one can effectively manage the complexity of the model and help it generalize better to unseen data. This method can help stabilize the search process, preventing search instabilities and, thus, better enabling the search for suitable model structures. The search object with regularization is given by:

$$\mathcal{L}(\theta,\alpha) = \mathcal{L}^{\ell 1}(\theta,\alpha) + \lambda\ Linear(R(\alpha)) \tag{5}$$

where $\mathcal{L}^{\ell 1}$ is the search object for the super-resolution task, which is an L1 loss. L1 loss is popular for SR tasks [43,44]. Where λ is a balancing factor. The introduction of linear interpolation is to avoid adding strong regularization in the early stages of the architecture search, which would cause the normalized weights of dominant operations to be further enlarged, making it difficult for other operations to stand out, even if they have better performance, in the later search stage. Hence, it is necessary to gradually increase the impact of regularization through linear interpolation, with the formula being:

$$Linear = \frac{e}{E} \tag{6}$$

where e is the current epoch and E presents the expected epoch that has the maximum effect on the regularization. To further enhance the stability of the architecture search process, we also introduce edge normalization to balance the importance of each edge. This normalization method adopts the variance-based edge regularization in EBNAS [45]. The regularization is given by:

$$\begin{aligned} R(\beta) &= \sum_j \Big(R_j^a(\beta) + R_j^b(\beta) \Big) \\ &= \sum_j \left(-\sum_{i<j} (s_{i,j} - \frac{1}{j} \sum_{i'<j} S_{i',j})^2 + \gamma (\sum_{i<j} S_{i',j} - 2)^2 \right) \end{aligned} \tag{7}$$

Finally, the object with two instances of regularization can be expressed as:

$$\mathcal{L}(\theta,\alpha) = \mathcal{L}^{\ddagger 1}(\theta,\alpha) + \lambda\ Linear(R(\alpha)) + \mu\ Linear(R(\beta)) \tag{8}$$

Regarding search spaces, maxpool and avgpool operations contribute to learning positional invariance in image classification tasks, effectively recognizing features even if they shift or distort slightly in the image. This effectiveness is due to pooling operations that select certain values to reduce the spatial dimensions, thus deriving more robust representations of the features. However, in super-resolution tasks, using maxpool and avgpool is generally not recommended, as these pooling techniques can lead to some information loss during processing. The essence of super-resolution tasks lies in extracting high-resolution details from low-resolution images, which requires preserving as much information as possible, rather than reducing it. Hence, removing all of the pooling operations is necessary to avoid the model suffering from a large amount of information loss. Additionally, larger convolutional kernels can cover a broader area, capturing more extensive contextual information. In super-resolution tasks, a larger receptive field helps the network understand the relationships between pixels over a greater area, which is crucial for reconstructing high-quality, high-resolution images. This is the reason we maintained the dilated group convolution operation and introduced a 7 × 7 binary group convolution. Table 1 presents our search space.

Table 1. Search space for BNNs in regard to SR.

3 × 3 binary dilated group convolution
3 × 3 binary group convolution
5 × 5 binary dilated group convolution
5 × 5 binary group convolution
7 × 7 binary group convolution
skip connection

In the search space, we made detailed structural modifications to each operation to adapt them for super-resolution tasks. For image super resolution, the model's output image should match the input in terms of color, contrast, and brightness. The changes are primarily made in regard to the resolution and some details. Batch normalization (batch norm) acts as a contrast stretcher for images. When an image undergoes batch norm, its color distribution is normalized, which causes a loss of the original contrast information of the image. Therefore, batch norm actually impacts the quality of the model's output. Figure 3 shows the basic structure of this operation in the search space.

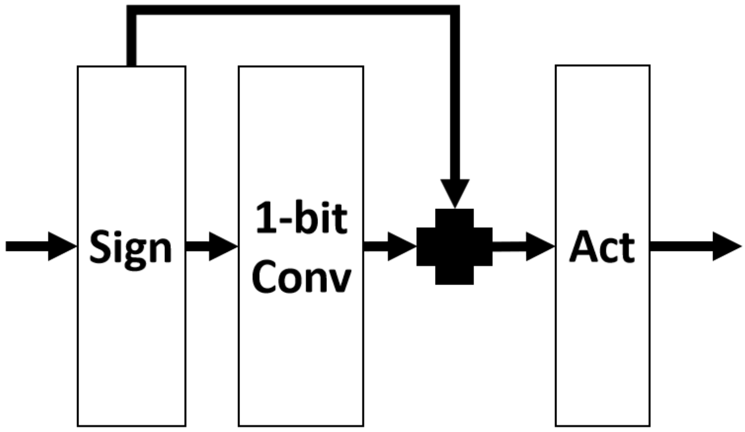


Figure 3. Basic structure of the operation in the search space.

A cell consists of several different candidate operations in the search space. The Figure 4 shows an example of a cell, with each edge representing an operation.

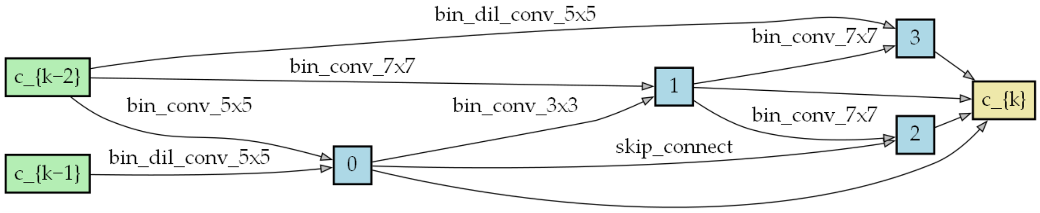


Figure 4. Example of a cell.

The performance degradation in binary neural networks is primarily due to their limited representational capacity and the discrete nature of binarization, which leads to significant information loss during both forward and backward propagation. In forward propagation, when activations and weights are restricted to just two values, the diversity of the model sharply decreases. During backward propagation, accurate gradients are essential for providing correct optimization directions; however, binary networks often produce inaccurate gradients and incorrect optimization directions during training due to their discrete binary values. To address this problem, we introduce the quantization method from IR-Net [24].

During forward propagation, quantization operations lead to information loss. In many binarized convolutional neural networks, minimizing the quantization error is adopted as the objective function for optimization. The quantization can be formulated as

$$Q_x(x) = B_x \tag{9}$$

where B_x is the binarized activation or inputs according to the sign function. The objective function is given by:

$$\min J(Q_x(x)) = \|x - Q_x(x)\|^2 \tag{10}$$

The information entropy is performed for the binarized result $Q_x(x)$, which is in fact the information entropy of B_x, which follows a Bernoulli distribution. Thus, the formula can be expressed as:

$$f(B_x) = \begin{cases} p, & if B_x = +1 \\ 1-p, & if B_x = -1 \end{cases} \tag{11}$$

$$H(Q_x(x)) = H(B_x) = -p\ln(p) - (1-p)\ln(1-p) \tag{12}$$

Moreover, the information entropy reaches its maximum when $p = 0.5$, indicating that the quantized values are uniformly distributed. The objective function of Libra-PB is defined as follows:

$$\min J(Q_x(x)) - \lambda H(Q_x(x)) \tag{13}$$

Additionally, to ensure more stable training and mitigate adverse effects caused by weights and gradients, further normalization was applied to balance the weights as follows:

$$\hat{W}_{std} = \frac{\hat{W}}{\sigma(\hat{W})}, \quad \hat{W} = W - \overline{W} \tag{14}$$

where $\overline{W}$ is the mean of the weight and $\sigma(\bullet)$ denotes the standard deviation. Through Equation (6), we gain the maximum information entropy of the weights, which makes the full-precision weight involved in binarization more spread out.

Moreover, in order to bypass expensive floating-point operations, while boosting the representational capability of binary weights, this approach incorporates integer scaling factors instead of floating-point ones. This adjustment allows binary calculations involving scaling factors to be simplified to:

$$\begin{gathered} Q_w(\hat{W}_{std}) = B_W <<>> s = sign(\hat{W}_{std}) <<>> S \\ S^* = round(log_2(||W_{std}||_1/n)) \end{gathered} \tag{15}$$

where <<>> denotes the left or right bit-shift operation. Finally, the binary convolution operation can be expressed as follow:

$$Z = (B_w \odot B_a) <<>> S \tag{16}$$

Due to the discontinuous nature of binarization, gradient approximation is an inevitable aspect of backward propagation. This makes it challenging to accurately model the effects of quantization, leading to significant information loss. To preserve the information derived from the loss function during backward propagation, we introduced a progressive two-stage approximation gradient method using EDE. In the first stage, we maintain the updating capability of the backward propagation algorithm by keeping the derivative values of the gradient estimation function close to 1, then gradually reducing the truncation value from a large number to 1. This rule allows our approximation function to evolve from an identity function to a clip function, ensuring early training updates. In the second stage, we keep the truncation value at 1 and gradually evolve the derivative curve to the shape of a step function. Using this rule, our approximation function transitions from a clip function to a sign function, thus ensuring consistency between forward and backward propagation.

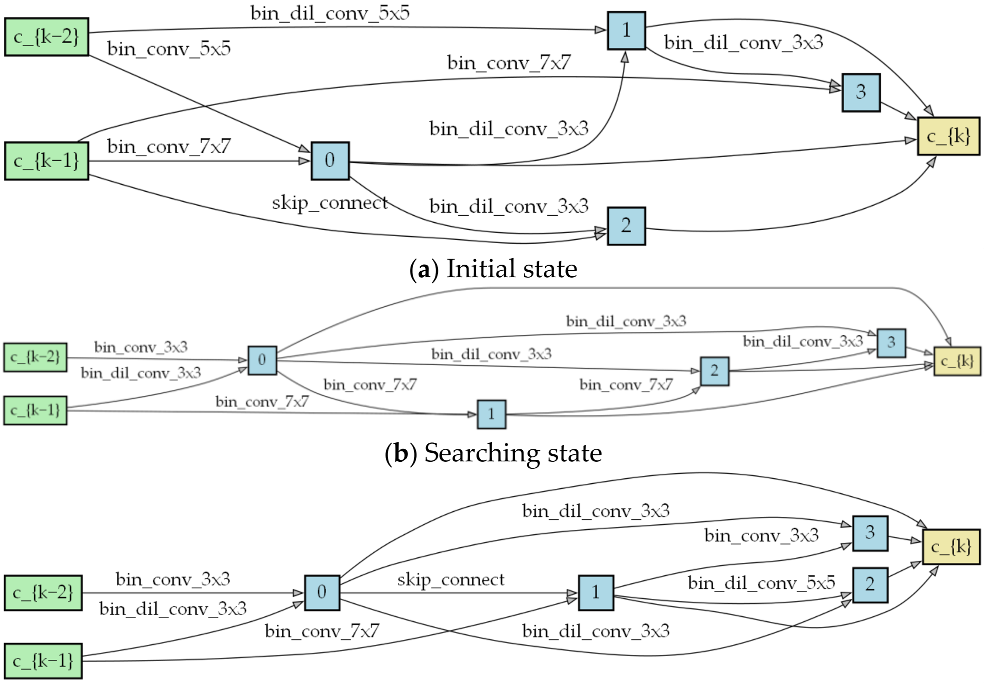
Of course, in addition to the improvements above, the basic architecture of DARTS is also unsuitable for SR tasks. Therefore, for the application of a binary neural network in SR, we propose a hierarchical basic architecture, as shown at the top of Figure 2. Through two full-precision shortcuts, we attempt to pass image information from different stages as much as possible without adding a large amount of computation, in order to further compensate for the information loss caused by binary neural networks.

6. Experiments

In terms of model training, to ensure the fairness of ablation studies, all models are trained on the publicly available Timofte dataset [46], which comprises 91 images designed for training purposes. For testing, the Set5 [47] and Set14 [48] datasets are used, containing 5 and 14 images, respectively. Additionally, the Berkeley Segmentation Dataset, consisting of 100 images (BSD100), is utilized for model evaluation. The training and testing were performed on the RTX4080 GPU. In the experiments, we configured each search to have 100 epochs, with a search framework learning rate of 0.0006. Upon completion of the search, model training was conducted using a learning rate of 0.001.

To simplify the comparison, we adopted the parameters from EBNAS for the balancing factors λ and γ, which are 1.5 and 0.8, respectively. The experiments primarily focused on the impact of the balancing factors in the alpha regularization term. We conducted three sets of experiments in which the balancing factors μ and ρ are 0.2 and 0.8, 0.5 and 0.5, and 0.8 and 0.2, respectively.

We trained a lightweight network comprised of three cells and a larger network consisting of eight cells, with scaling factors equal to 3 and 4. For the three-cell lightweight network, each layer has 64 channels, and all parameters are optimized using Adam with a channel sampling factor of k = 8. This model was trained over 100 epochs, with a batch size of 256. Figures 5 and 6 show the cell structure after the search process.

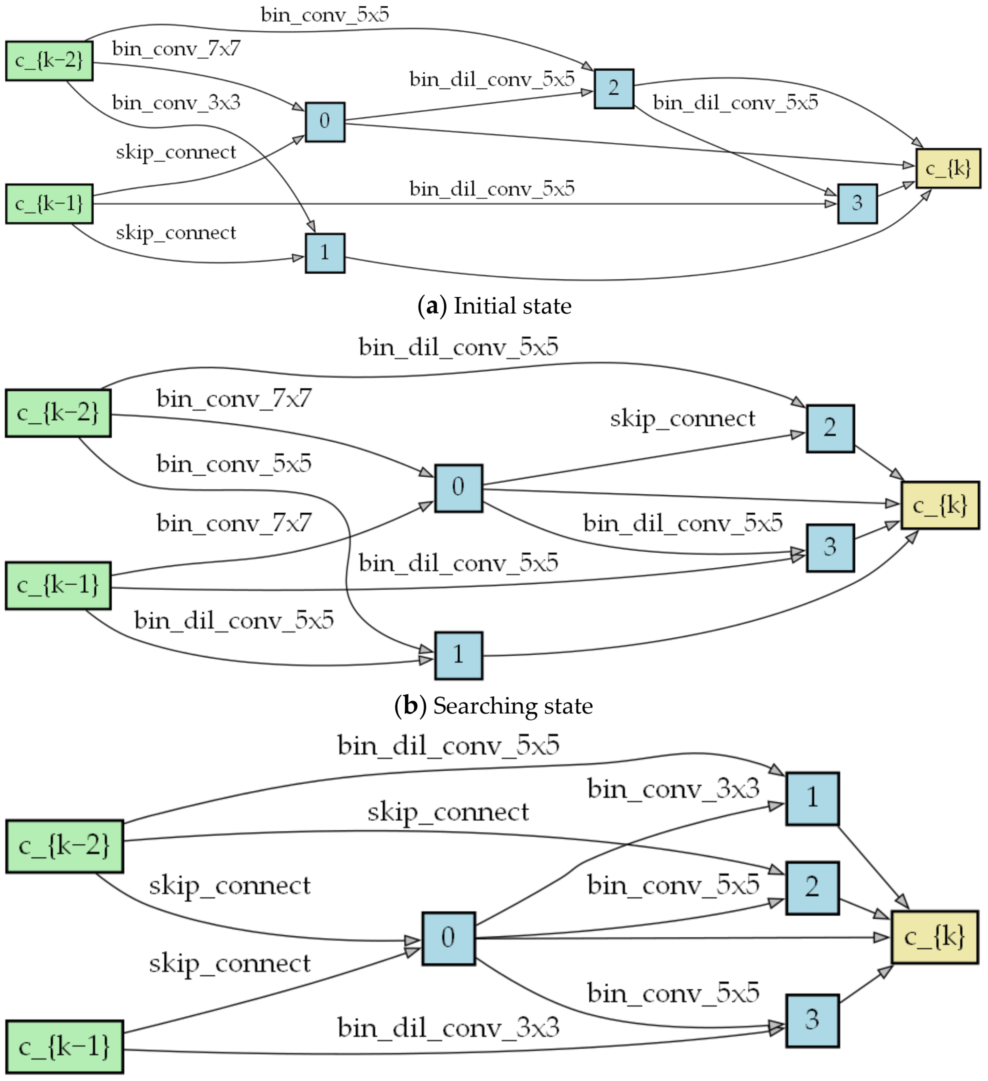


(**a**) Initial state

(**b**) Searching state

(**c**) Selected architecture

Figure 5. Cell structure, with scaling factor 3, after the search process.



(**a**) Initial state

(**b**) Searching state

(**c**) Selected architecture

Figure 6. Cell structure, with scaling factor 4, after the search process.

For the evaluation, the peak signal-to-noise ratio (PSNR) is a widely used metric in super-resolution tasks, used to evaluate the quality of high-resolution images obtained through models compared to ground truth images. It measures the similarity between two images, while considering the pixel values and image size.

The structural similarity (SSIM) metric is a metric used to measure the degree of similarity between two images and is commonly used in image quality evaluation.

Table 2 presents the performance comparison using the manual method on the three datasets, where our SRBNAS represents our 3-cell model searched by super net. When compared with manually designed models, the models derived from our network architecture search method show performance that rivals those based on the three evaluation datasets. The table shows that the SRBNAS results have very good performance. The results are shown for scaling factors 3 and 4.

Table 2. The mean PSNR/SSIM of different methods evaluated using different datasets.

Dataset	Scale	Bicubic	ESPCN (Full Precision)	ResBinESPCN-A2 [49]	SRResNet (BNN)	VDSR (BNN)	Ours: SRBNAS
Set5	×3	30.46/0.87	32.29/-	29.82/-	31.18/0.88	31.01/0.87	31.80/0.90
Set14		27.59/0.78	28.90/-	27.33/-	28.29/0.80	28.15/0.79	28.57/0.81
BSDS100		27.26/0.75	28.16/-	27.03/-	27.73/0.77	27.57/0.76	27.95/0.78
Set5	×4	28.48/0.82	28.80/-	28.11/-	29.33/0.82	29.02/0.82	29.34/0.84
Set14		25.92/0.72	26.16/-	25.78/-	26.72/0.72	26.55/0.72	26.47/0.74
BSDS100		26.02/0.67	26.21/-	25.87/-	26.45/0.69	26.29/0.69	26.45/0.70

Additionally, as depicted in Table 3, it becomes apparent that the model structures used are particularly efficient. Our SRBNAS model requires only about 156 K number of parameters to achieve an image reconstruction performance comparable to other methods. These results clearly demonstrate the high efficiency of our method. Compared to manual design approaches, NAS can save a significant amount of trial-and-error design time.

Table 3. Number of parameters for each model with a scale factor of 3.

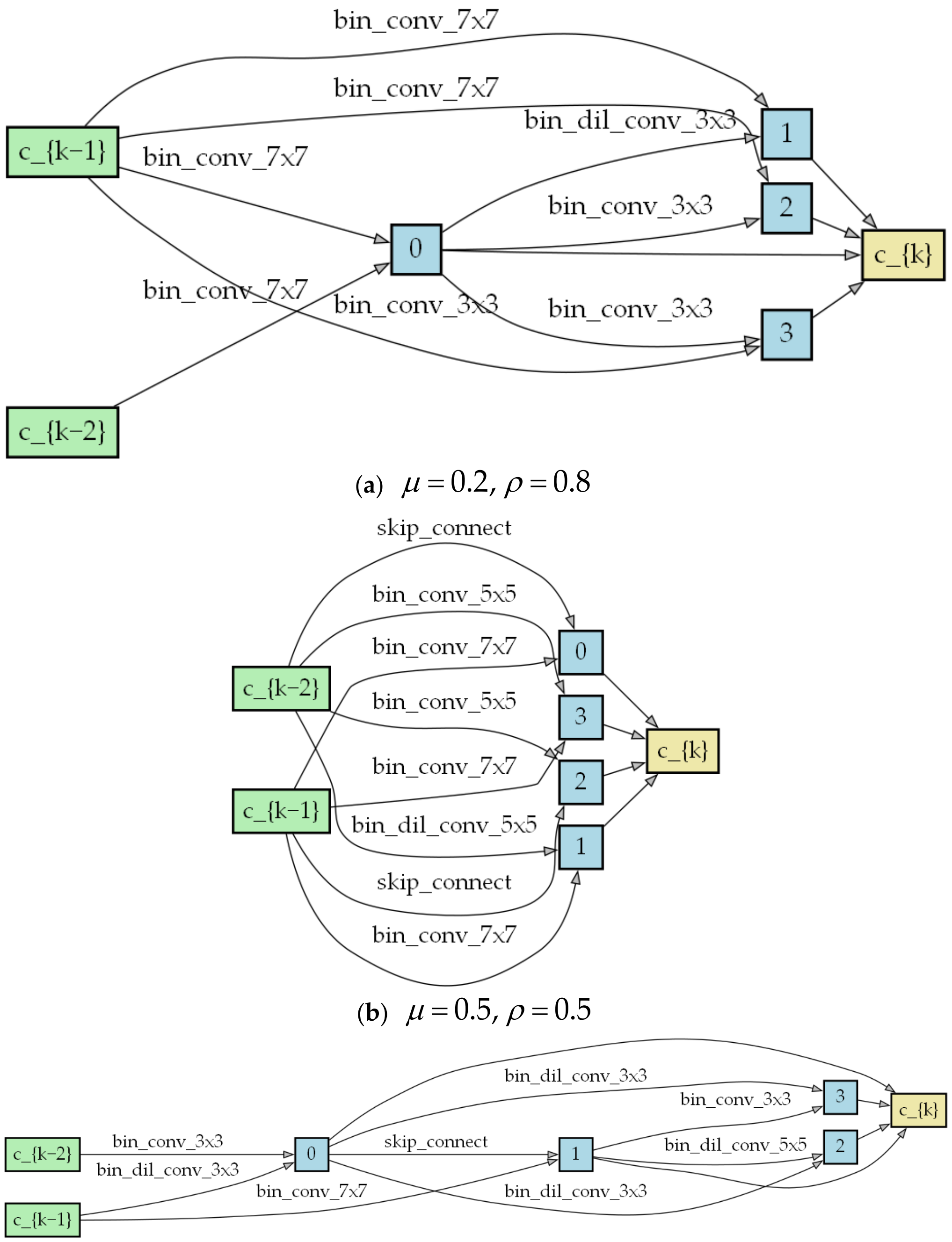
Model	Parameters
VDSR_BAM	668 K
SRResNet BAM	1547 K
BSRN	1216 K
ResBinESPCN-A2	349 K
Ours: SRBNAS	192 K

First, in Table 4, μ represents the balancing factor of L1 regularization, while ρ represents the balancing factor of L2 regularization. From the results of the PSNR, there is no significant difference in the performance of the model structures obtained from the search. However, as the regularization weight tends toward the L1 regularization term, the model size begins to drop sharply. Yet, in terms of model performance, the decrease in model size makes the model more efficient and lightweight. Specifically, L1 regularization tends to produce sparse weights, i.e., to make some weights tend toward zero, thus reducing the complexity of the model, while L2 regularization makes the weight distribution smoother and avoids the weights being too large or too small. Therefore, combining these two types of regularization and adjusting the weights of the L1 regularization appropriately can achieve the effect of simplifying the model and reducing the amount of computation.

Table 4. A comparison of different balancing factors with a scaling factor of 3.

Factor	Parameters	PSNR
$\mu = 0.2,\ \rho = 0.8$	366 K	32.17
$\mu = 0.5,\ \rho = 0.5$	350 K	32.02
$\mu = 0.8,\ \rho = 0.2$	192 K	31.84

Figure 7 shows the cell structure results according to the different combinations of balancing factors. From the figure, as ρ increases, high-computation operations in the model also increase accordingly, such as 5 × 5 and 7 × 7 convolutions and a lack of skip connections. The number of parameters increases significantly, but there is not much improvement in terms of the performance. However, as μ increases, the operations in the cell structure are more of a lower computational complexity, such as 3 × 3 and 5 × 5 dilation convolutions, or 3 × 3 convolutions.



(**a**) $\mu = 0.2, \rho = 0.8$

(**b**) $\mu = 0.5, \rho = 0.5$

(**c**) $\mu = 0.8, \rho = 0.2$

Figure 7. Cell structure with different combinations of regularization.

After a comparison with the manual method, for a fair comparison Table 5 presents the computational comparison with other gradient-based architecture search method for super resolution. The table shows the reduced computational requirements for our SRBNAS method compared with two other methods (DLSR [48] and DNAS-EASR [49]).

Table 5. Computational comparison with other gradient-based architecture search methods for SR.

Method	Scale	Param	Set5	Set14	BSDS100	Search Cost
DLSR [50]	×4	338 K	32.33	27.85	27.61	2 GPU days (RTX3080)
DNAS-EASR [51]		555 K	32.18	28.64	27.61	21 GPU hours (RTX3090)
Ours SRBNAS		156 K	29.34	26.47	26.45	1.2121 GPU hours (RTX4080)

To visualize the model performance, Figures 8 and 9 display three different HR images. We also zoomed in on the images to facilitate the observation of changes in the image details.

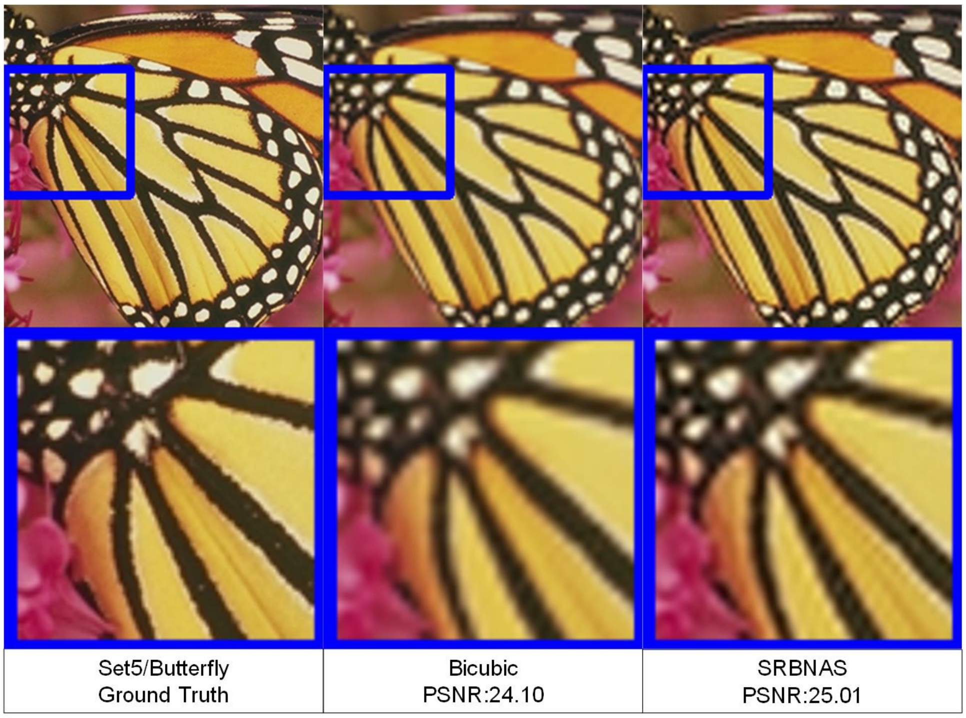


Figure 8. Visual quality comparisons of a HR image of a butterfly from Set5, with an upscaling factor of 3.

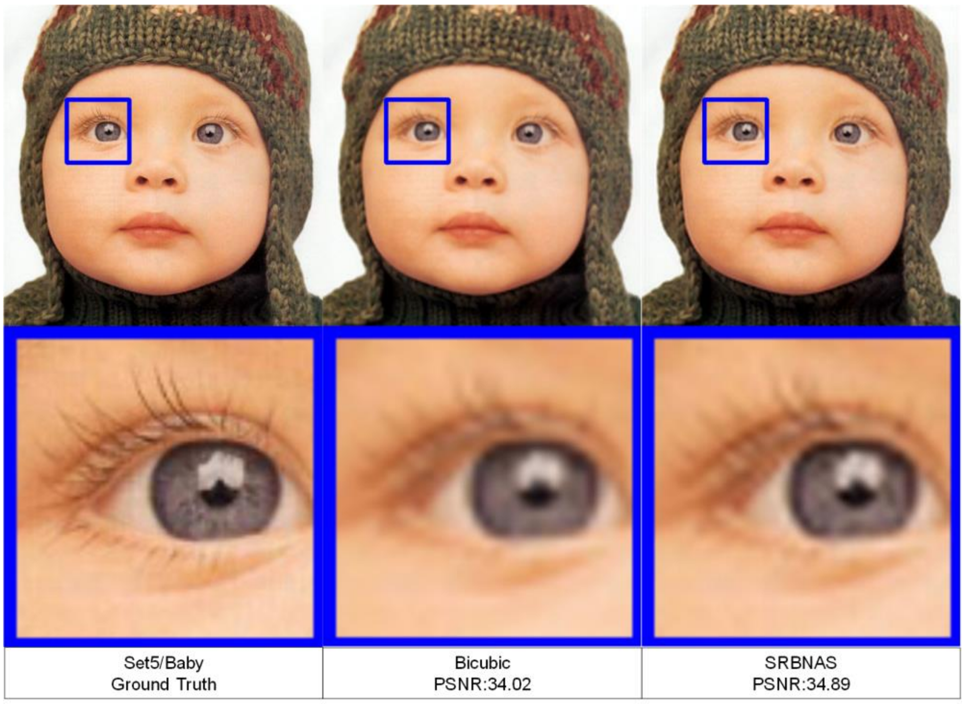


Figure 9. Visual quality comparisons of a HR image of a baby from Set14, with an upscaling factor of 3.

7. Conclusions

Based on binary neural networks (BNNs), this paper proposes a network architecture search method called SRBNAS, specifically for super-resolution tasks. The method utilizes several techniques to achieve performance and computational improvements. By integrating Libra Parameter Binarization (Libra-PB), the method aims to preserve as much information as possible during forward propagation. The inclusion of an Error Decay Estimator (EDE) during backward propagation assists in effectively handling the reduction in the parameter update capabilities outside the truncation range and mitigating information loss due to approximate errors within that range. These strategies effectively lessen the discrepancies between the search phase and the evaluation phase. The experimental results demonstrate that the network structures identified by this search method have a good level of performance and are computationally efficient compared to other approaches and manually designed alternatives.

8. Patents

This section is not mandatory but may be added if there are patents resulting from the work reported in this manuscript.

Author Contributions: Conceptualization, K.P.S., Y.S. and L.-m.A.; methodology, Y.S. and K.P.S.; resources, K.P.S.; data curation, Y.S., K.P.S. and L.-m.A.; writing—original draft preparation, K.P.S., Y.S., L.-m.A. and J.S.; writing—review and editing, K.P.S., Y.S. and L.-m.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author/s.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Umehara, K.; Ota, J.; Ishida, T. Application of Super-Resolution Convolutional Neural Network for Enhancing Image Resolution in Chest CT. *J. Digit. Imaging* **2018**, *31*, 441–450. [CrossRef] [PubMed]
2. You, C.; Li, G.; Zhang, Y.; Zhang, X.; Shan, H.; Ju, S.; Zhao, Z.; Zhang, Z.; Cong, W.; Vannier, M.W.; et al. CT Super-Resolution GAN Constrained by the Identical, Residual, and Cycle Learning Ensemble(GAN-CIRCLE). *IEEE Trans. Med. Imaging* **2020**, *39*, 188–203. [CrossRef]
3. Rasti, P.; Uiboupin, T.; Escalera, S.; Anbarjafari, G. Convolutional Neural Network Super Resolution for Face Recognition in Surveillance Monitoring. In Proceedings of the Articulated Motion and Deformable Objects, Palma de Mallorca, Spain, 13–15 July 2016; Perales, F.J., Kittler, J., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 175–184.
4. Shen, Z.; Xu, Y.; Lu, G. CNN-Based High-Resolution Fingerprint Image Enhancement for Pore Detection and Matching. In Proceedings of the 2019 IEEE Symposium Series on Computational Intelligence (SSCI), Xiamen, China, 6–9 December 2019; pp. 426–432.
5. Ribeiro, E.; Uhl, A.; Alonso-Fernandez, F.; Farrugia, R.A. Exploring Deep Learning Image Super-Resolution for Iris Recognition. In Proceedings of the 2017 25th European Signal Processing Conference (EUSIPCO), Kos, Greece, 28 August–2 September 2017; pp. 2176–2180.
6. Lobanov, A.P. Resolution Limits in Astronomical Images. *arXiv* **2005**, arXiv:astro-ph/0503225.
7. Keys, R. Cubic Convolution Interpolation for Digital Image Processing. *IEEE Trans. Acoust. Speech Signal Process.* **1981**, *29*, 1153–1160. [CrossRef]
8. Dong, C.; Loy, C.C.; He, K.; Tang, X. Image Super-Resolution Using Deep Convolutional Networks. *arXiv* **2015**, arXiv:1501.00092. [CrossRef] [PubMed]
9. Kim, J.; Lee, J.K.; Lee, K.M. Deeply-Recursive Convolutional Network for Image Super-Resolution. *arXiv* **2016**, arXiv:1511.04491.
10. Tai, Y.; Yang, J.; Liu, X. Image Super-Resolution via Deep Recursive Residual Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2790–2798.
11. Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; Fu, Y. Residual Dense Network for Image Super-Resolution. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2472–2481.
12. Zhang, Y.; Li, K.; Li, K.; Wang, L.; Zhong, B.; Fu, Y. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. *arXiv* **2018**, arXiv:1807.02758.
13. Liu, H.; Simonyan, K.; Yang, Y. DARTS: Differentiable Architecture Search. *arXiv* **2019**, arXiv:1806.09055.
14. Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Chen, X.; Wang, X. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions. *arXiv* **2021**, arXiv:2006.02903.
15. Zoph, B.; Le, Q.V. Neural Architecture Search with Reinforcement Learning. *arXiv* **2017**, arXiv:1611.01578.

16. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning Transferable Architectures for Scalable Image Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
17. Real, E.; Aggarwal, A.; Huang, Y.; Le, Q.V. Regularized Evolution for Image Classifier Architecture Search. *arXiv* **2019**, arXiv:1802.01548.
18. Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. *arXiv* **2016**, arXiv:1602.02830.
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
20. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. *arXiv* **2018**, arXiv:1512.03385.
21. Liu, Z.; Wu, B.; Luo, W.; Yang, X.; Liu, W.; Cheng, K.-T. Bi-Real Net: Enhancing the Performance of 1-Bit CNNs with Improved Representational Capability and Advanced Training Algorithm. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 722–737.
22. Bethge, J.; Yang, H.; Bornstein, M.; Meinel, C. BinaryDenseNet: Developing an Architecture for Binary Neural Networks. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Republic of Korea, 27–28 October 2019; pp. 1951–1960.
23. Sun, R.; Zou, W.; Zhan, Y. "Ghost" and Attention in Binary Neural Network. *IEEE Access* **2022**, *10*, 60550–60557. [CrossRef]
24. Qin, H.; Gong, R.; Liu, X.; Shen, M.; Wei, Z.; Yu, F.; Song, J. Forward and Backward Information Retention for Accurate Binary Neural Networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–18 June 2020; pp. 2247–2256.
25. Dong, C.; Loy, C.C.; Tang, X. Accelerating the Super-Resolution Convolutional Neural Network. *arXiv* **2016**, arXiv:1608.00367.
26. Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.P.; Bishop, R.; Rueckert, D.; Wang, Z. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. *arXiv* **2016**, arXiv:1609.05158.
27. Tong, T.; Li, G.; Liu, X.; Gao, Q. Image Super-Resolution Using Dense Skip Connections. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 4809–4817.
28. Xin, J.; Wang, N.; Jiang, X.; Li, J.; Huang, H.; Gao, X. Binarized Neural Network for Single Image Super Resolution. In *Computer Vision—ECCV 2020, Glasgow, UK, 23–28 August 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2020; Volume 12349, pp. 91–107. ISBN 978-3-030-58547-1.
29. Bethge, J.; Bartz, C.; Yang, H.; Chen, Y.; Meinel, C. MeliusNet: Can Binary Neural Networks Achieve MobileNet-Level Accuracy? *arXiv* **2020**, arXiv:2001.05936v2.
30. Liu, Z.; Shen, Z.; Savvides, M.; Cheng, K.-T. ReActNet: Towards Precise Binary Neural Network with Generalized Activation Functions. In Proceedings of the Computer Vision—ECCV 2020, Glasgow, UK, 23–28 August 2020; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 143–159.
31. Rastegari, M.; Ordonez, V.; Redmon, J.; Farhadi, A. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. *arXiv* **2016**, arXiv:1603.05279.
32. Ghiasi, G.; Lin, T.-Y.; Pang, R.; Le, Q.V. NAS-FPN: Learning Scalable Feature Pyramid Architecture for Object Detection. *arXiv* **2019**, arXiv:1904.07392.
33. Xu, A.; Yao, A.; Li, A.; Liang, A.; Zhang, A. Auto-FPN: Automatic Network Architecture Adaptation for Object Detection Beyond Classification. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27–28 October 2019; pp. 6648–6657.
34. Chen, L.-C.; Collins, M.D.; Zhu, Y.; Papandreou, G.; Zoph, B.; Schroff, F.; Adam, H.; Shlens, J. Searching for Efficient Multi-Scale Architectures for Dense Image Prediction. *arXiv* **2018**, arXiv:1809.04184.
35. Liu, C.; Chen, L.-C.; Schroff, F.; Adam, H.; Hua, W.; Yuille, A.; Fei-Fei, L. Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
36. Roberts, N.; Khodak, M.; Dao, T.; Li, L.; Ré, C.; Talwalkar, A. Rethinking Neural Operations for Diverse Tasks. *arXiv* **2021**, arXiv:2103.15798.
37. Shen, J.; Khodak, M.; Talwalkar, A. Efficient Architecture Search for Diverse Tasks. *arXiv* **2022**, arXiv:2204.07554.
38. Chen, X.; Xie, L.; Wu, J.; Tian, Q. Progressive Differentiable Architecture Search: Bridging the Depth Gap between Search and Evaluation. *arXiv* **2019**, arXiv:1904.12760.
39. Xu, Y.; Xie, L.; Zhang, X.; Chen, X.; Qi, G.-J.; Tian, Q.; Xiong, H. PC-Darts: Partial channel connections for memory-efficient architecture search. *arXiv* **2020**, arXiv:1907.05737.
40. Xue, Y.; Qin, J. Partial Connection Based on Channel Attention for Differentiable Neural Architecture Search. *IEEE Trans. Ind. Inf.* **2023**, *19*, 6804–6813. [CrossRef]
41. Kim, D.; Singh, K.P.; Choi, J. Learning Architectures for Binary Networks. *arXiv* **2020**, arXiv:2002.06963.
42. Bulat, A.; Martinez, B.; Tzimiropoulos, G. BATS: Binary ArchitecTure Search. *arXiv* **2020**, arXiv:2003.01711.
43. Lim, B.; Son, S.; Kim, H.; Nah, S.; Lee, K.M. Enhanced Deep Residual Networks for Single Image Super-Resolution. *arXiv* **2017**, arXiv:1707.02921.

44. Zhao, H.; Gallo, O.; Frosio, I.; Kautz, J. Loss Functions for Neural Networks for Image Processing. *arXiv* **2015**, arXiv:1511.08861.
45. Shi, C.; Hao, Y.; Li, G.; Xu, S. EBNAS: Efficient Binary Network Design for Image Classification via Neural Architecture Search. *Eng. Appl. Artif. Intell.* **2023**, *120*, 105845. [CrossRef]
46. Timofte, R.; De Smet, V.; Van Gool, L. A+: Adjusted Anchored Neighborhood Regression for Fast Super-Resolution. In *Proceedings of the Computer Vision—ACCV 2014*; Cremers, D., Reid, I., Saito, H., Yang, M.-H., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 111–126.
47. Bevilacqua, M.; Roumy, A.; Guillemot, C.; Morel, M.A. Low-Complexity Single-Image Super-Resolution Based on Nonnegative Neighbor Embedding. In Proceedings of the British Machine Vision Conference 2012, Surrey, UK, 3–7 September 2012; British Machine Vision Association: Surrey, UK, 2012; pp. 135.1–135.10.
48. Zeyde, R.; Elad, M.; Protter, M. On Single Image Scale-Up Using Sparse-Representations. In *Proceedings of the Curves and Surfaces*; Boissonnat, J.-D., Chenin, P., Cohen, A., Gout, C., Lyche, T., Mazure, M.-L., Schumaker, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 711–730.
49. Su, Y.; Seng, K.P.; Smith, J.; Ang, L.M. Efficient FPGA Binary Neural Network Architecture for Image Super-Resolution. *Electronics* **2024**, *13*, 266. [CrossRef]
50. Huang, H.; Shen, L.; He, C.; Dong, W.; Liu, W. Differentiable Neural Architecture Search for Extremely Lightweight Image Super-Resolution. *arXiv* **2022**, arXiv:2105.03939. [CrossRef]
51. Sheng, C.; Gao, X.; Hu, X.; Wang, F. Differentiable Neural Architecture Search Based on Efficient Architecture for Lightweight Image Super-Resolution. In *Proceedings of the MultiMedia Modeling*; Rudinac, S., Hanjalic, A., Liem, C., Worring, M., Jónsson, B.Þ., Liu, B., Yamakata, Y., Eds.; Springer Nature: Cham, Switzerland, 2024; pp. 169–183.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.