electronics

MDPI

Article

INTS-Net: Improved Navigator-Teacher-Scrutinizer Network for Fine-Grained Visual Categorization

Huilong Jin, Jiangfan Xie, Jia Zhao *, Shuang Zhang, Tian Wen, Song Liu and Ziteng Li

College of Engineering, Hebei Normal University, Shijiazhuang 050024, China
* Correspondence: zhaojia2021@hebtu.edu.cn

Abstract: Fine-grained image recognition, as a significant branch of computer vision, has become prevalent in various applications in the real world. However, this image recognition is more challenging than general image recognition due to the highly localized and subtle differences in special parts. Up to now, many classic models, including Bilinear Convolutional Neural Networks (Bilinear CNNs), Destruction and Construction Learning (DCL), etc., have emerged to make corresponding improvements. This paper focuses on optimizing the Navigator-Teacher-Scrutinizer Network (NTS-Net). The structure of NTS-Net determines its strong ability to capture subtle information areas. However, research finds that this advantage will lead to a bottleneck of the model's learning ability. During the training process, the loss value of the training set approaches zero prematurely, which is not conducive to later model learning. Therefore, this paper proposes the INTS-Net model, in which the Stochastic Partial Swap (SPS) method is flexibly added to the feature extractor of NTS-Net. By injecting noise into the model during training, neurons are activated in a more balanced and efficient manner. In addition, we obtain a speedup of about 4.5% in test time by fusing batch normalization and convolution. Experiments conducted on CUB-200-2011 and Stanford cars demonstrate the superiority of INTS-Net.

Keywords: computer vision; convolution neural network; fine-grained imaged recognition

Citation: Jin, H.; Xie, J.; Zhao, J.; Zhang, S.; Wen, T.; Liu, S.; Li, Z. INTS-Net: Improved Navigator-Teacher-Scrutinizer Network for Fine-Grained Visual Categorization. *Electronics* **2023**, *12*, 1709. https://doi.org/10.3390/electronics12071709

Academic Editors: Jungong Han and Guiguang Ding

Received: 28 February 2023
Revised: 27 March 2023
Accepted: 28 March 2023
Published: 4 April 2023

1. Introduction

Computer vision is an interdisciplinary field in artificial intelligence. The focus of the research is to enable computers to extract information from digital images or videos that rivals or surpasses human comprehension, including image processing, image analysis, image understanding, etc. Image recognition, including both general and fine-grained image recognition, is an important task in computer vision. With the vigorous development of deep learning, fine-grained image recognition, a research hotspot in the visual field, has made remarkable progress in the application of deep learning technology. Existing works [1–15] attempt to identify under subcategories of given images by exploring various techniques. More recent part-based methods [2,9,10,12,13] and sampling-based methods [1,3,4] can discover regions in a weakly supervised learning manner. In order to enrich feature representations with high-order information, examples such as [16–18] perform element-wise swapping or mixing for partial features between samples to inject noise during training.

Twenty years ago, Irving Biederman et al. [19] put forward the concept of fine-grained image classification (FGVC) for the first time. They introduced a critical issue largely disregarded at the time: whether machines could recognize objects as precisely as humans could at a fine-grained level. This idea has aroused the interest of researchers, and since then many classic fine-grained image recognition models have been proposed, substantially advancing this discipline. Traditional image recognition methods mainly learn advanced features. Despite the high accuracy achieved in general image recognition tasks, the defect of ignoring subtle details makes it impossible for them to achieve satisfactory results in

fine-grained recognition. Compared with general image recognition, fine-grained image recognition is more challenging, in that the inter-class differences in objects are subtle. The model should be able to extract object features from the whole picture, and should also be more sensitive to local features with discriminant information, which raises the threshold for the feature extraction ability of the neural network model.

Many researchers have focused on gaining delicate feature representation which plays a key role in fine-grained image recognition. Some methods [20–22] complete the detection of the foreground object with the help of the annotation box and eliminate the interference of the background. Due to the low labeling cost, other methods [11,23–26] that only need label information have received more attention in research and application.

NTS-Net belongs to the latter method. Consisting of three networks sharing weights, this model adopts a self-monitoring mechanism and combines multi-scale feature extraction. Moreover, NTS-Net combines multiple networks to build an overall network structure, processes features at multiple scales, and learns the features of the whole and parts of the original image so that the critical features in the images can be fully extracted and selected. Therefore, it can efficiently locate the information area without using any bounding-box or part annotations, achieving the state-of-the-art performance of various public datasets at that time. To some extent, the learning ability of a model determines its dataset recognition accuracy. If used improperly, the excellent learning ability of image features will act as a barrier to further promotion rather than a stepping stone for a successful model. Similarly, the NTS-Net model can quickly improve the ability to capture detailed features, but this kind of ability may cause bottlenecks in the process of model learning. That is, the loss values may decrease rapidly and approach zero in the process of model training. This phenomenon is not conducive to the weight update of the model. Methods such as Mixup [27], Cutmix [28], and SPS [17] can be understood as a form of data augmentation, which improve the performance of the models by injecting or reducing noise into the model. These methods give us some possibilities for further exploration. Therefore, in order to further improve the recognition effect of NTS-Net, this paper makes two improvements. First, the SPS method is added to the backbone network, which not only suppresses over-active neurons but also fully enhances the learning ability of each neuron in network layers, thereby improving its ability to obtain discriminant information. Second, inspired by module fusion methods which could merge adjacent modules to improve computational efficiency, the fusion of a convolutional layer and a batch normalization layer is applied to the feature extractor. During runtime, this strategy accelerates the inference time.

2. Related Works

2.1. Fine-Grained Image Recognition

Compared with general image recognition tasks, fine-grained image recognition is more challenging. Due to the subtle differences among inter-classes, fine-grained image recognition requires more specialized techniques, such as discriminative feature learning and object parts localization. With the exploration of deep learning, fine-grained image recognition models have developed rapidly, and the development process has also witnessed some important progress in computer vision. At present, the research on fine-grained image recognition is mainly carried out in two directions, that is, learning better visual features from original images and using local or attention-based methods to obtain vital features.

Before deep learning was widely applied, fine-grained image recognition methods used multi-stage manual feature extraction methods for image recognition. Research results of this stage showed that feature description methods and feature coding methods have a significant impact on classification accuracy. Stronger feature processing methods can improve the accuracy, which is further proved by subsequent studies on convolutional features. Regarding the study on convolution features, Yu et al. proposed a bilinear pooling method of end-to-end training [29], which interacted with local features from two CNN branches. Despite the impressive performance it achieved, the method's high

dimensionality of bilinear features makes it challenging to optimize. The bilinear pooling method gives researchers a lot of inspiration, so they study for the optimization of the bilinear method on this basis. Using the compact bilinear representation kernel method [30] or low-rank bilinear pooling [31] to represent covariance features as a matrix, and applying the low-rank bilinear classifier, can greatly reduce the calculation time and reduce the effective number of learning parameters.

Another research dimension lays emphasis on the extraction of discriminative local features. The method can be further subdivided into fine-grained image classification methods with strong supervision and weak supervision depending on whether additional manual labeling information, such as labeling boxes and local area positions other than the category labels of the images, is used. Although the extra manual labeling information fosters the network in obtaining the local features with discriminant information and improves the network performance to a certain extent, the practicability of this type of algorithm is constrained by the labor cost of labeling information to a great extent. At present, much research focuses on weakly supervised fine-grained image recognition, that is, single reliance on category labels to complete model training. Xiao et al. [32] combine three types of attention to train domain-specific deep nets, then use them to improve performance. Sun et al. [5] propose a novel attention-based convolutional neural network composed of two parts, and the model regulates multiple part regions among different images. These methods introduce the attention mechanism into image recognition, making the deep learning model pay more attention to certain local information. NTS-Net takes advantage of the learning ability to pay attention to the details, so it is critical to fully and evenly invoke neurons.

2.2. Noise Injection Methods

The model designed in deep learning should not only perform well on training data but also achieve satisfactory results on new data sets, which requires a good generalization ability of the model. For this reason, researchers have put forward many strategies to explicitly reduce test errors, which are collectively called regularization. Deep neural networks often have over-fitting because of the data set or network structure. Regularization is a common method to avoid this phenomenon. The method we choose belongs to the noise regularization technology in regularization [33–35], where classic methods include Dropout and its variants. These methods mainly inject noise by adding or increasing noise during training. For example, Dropout randomly discards neurons in a certain ratio during training, and Gaussian Dropout multiplies characteristic units by Gaussian random noise. Compared with these methods mentioned before, Stochastic Partial Swap (SPS) [17] method uses some elements of other samples as noise sources to generate noise features, which can effectively suppress some neurons' over-confidence in specific categories. It provides a more reasonable method of simulating real data noise to improve the robustness of the classifier. In this paper, SPS fully demonstrates its advantages in training.

3. Methods

In this section, firstly, the classic fine-grained image recognition model NTS-Net and the SPS method are presented, then the improved model termed INTS-Net is introduced in the following subsection.

3.1. NTS-Net

NTS-Net is a fine-grained image recognition method with weak supervision. It adopts a novel self-supervision mechanism to locate the information area and uses ResNet50 as the feature extractor. The model accurately identifies the information areas in the image through a multi-agent cooperative learning scheme. This scheme includes the Navigator network, Teacher Network, and Scrutinizer network. Agents share the parameters of feature extractors, cooperate with each other, promote each other, and make progress together, so as to improve the model's ability to obtain discriminant information. At the

same time, NTS-Net uses a three-part loss function, including Navigator loss, Teaching loss, and Scrutinizing loss, to improve the probability of selecting an area containing more semantic information about object features, thus providing a more accurate prediction.

As shown in Figure 1, the structure of NTS-Net consists of three networks: Navigator network, Teacher Network, and Scrutinizer network. Firstly, the original image is input to the Navigator network, which includes a top-down architecture with horizontal connections to detect areas at multiple scales, and then the multi-scale feature map is used to generate areas with different scales and proportions. The first network generates several regions, then uses NMS to select M areas and send them to the Teacher network. The teacher network outputs confidence as teaching signals which help the Navigator network learn. The Scrutinizer network receives op-K candidate regions from the Navigator network and resizes these regions. Finally, the Scrutinizer network concatenates K regions with the full image feature and predicts the label of the image. The L_{total} is the loss function composed of three parts for the total model, which is defined as:

$$L_{total} = L_{\mathcal{I}}(I, C) + \lambda L_C(R, X) + \mu L_S(R, X) \tag{1}$$

$$\begin{aligned} L_{total} = \sum_{(i,s):c_i<c_j} f(I_s - I_i) + \left(- \sum_{i=1}^{M} \log C(R_i) - \log C(X) \right) \\ + (-\log S(X, R_1, R_2, \cdots, R_k)) \end{aligned} \tag{2}$$

where the parameter X stands for the full image, I and C separately denote the informativeness of candidate regions R predicted by Navigator network and corresponding confidence predicted by the Teacher network. In the experiments, $\lambda = \mu = 1$.

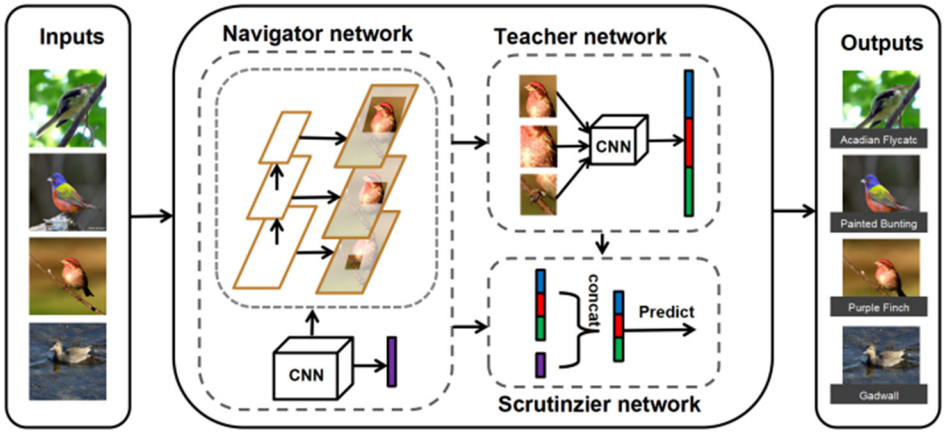


Figure 1. The framework of NTS-Net which consists of three parts.

3.2. SPS

When a model performs well in the training set but fails to achieve the desired effect on the test set, the model is overfitting. In the face of insufficient samples, the model will often over-fit, but this does not imply that the deep neural network will not over-fit in the presence of enough samples. The deep neural network is a very powerful machine learning system. However, as the depth increases, over-fitting becomes a serious problem.

In 2014, Srivastavae et al. put forward the dropout method to solve this problem. During the training process, they suggested injecting noise into each layer of the network before calculating the subsequent layers. Because when training a deep network, noise injection will enhance the smoothness of the input-output mapping. Therefore, injecting noise into each inner layer of forward propagation computation has become a common technique to train neural networks.

Although dropout activates more neurons in the process of model training, the predicted result will still be controlled by highly activated neurons, which is not conducive

to model training and generalization. Therefore, in this paper, SPS is adopted to alleviate this phenomenon. As one of the noise injection methods, SPS exchanges elements of partial features between samples, injecting noise when neural networks are trained in this way. For each sample (feature vector) in the mini-batch, we first randomly select another sample from the same mini-batch as a noise source and swap their partial feature elements element-wise. The function of SPS is summarized in Algorithm 1. Figure 2 illustrates how SPS works, which is similar to the regularization effect of Dropout. The noise-injecting operation can be expressed as:

$$\widetilde{f}^{m}_{\rho\sim U(\alpha,\beta)}(x_i) = M \odot f^{m}(x_i) + (1 - M) \odot f^{m}(x_j) \tag{3}$$

where x_i and x_j are samples from the same mini-batch, U is a uniform distribution between α and β, $\odot$ denotes element-wise multiplication, the function f^m stands for feature extractor of the model, and $M \in R^{dim(f^m(x))}$ is a binary mask acting on samples. To define SPS, we need a random number generation function rand to regenerate a value ρ between 0 and 1 for every epoch, that is

$$M[k] = \begin{cases} 1, & rand(0,1) \leq \rho \\ 0, & rand(0,1) > \rho \end{cases} \tag{4}$$

where M is based on the generated value ρ, and the parameter $k \in [0, dim(f^m(x)) - 1]$ is the dimension index.

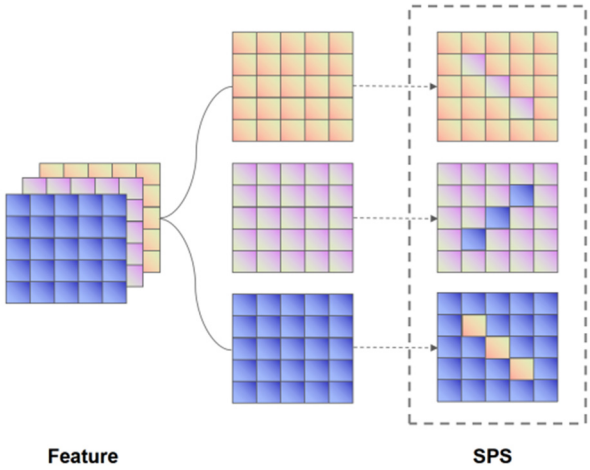


Figure 2. An illustration of the SPS, which shows how SPS exploits samples as a source for noise injection.

Algorithm 1. The function of SPS

Input: A two-demensional matrix x, which represents mini batches of data. The two dimension respectively represents batch size and feature vectors.
Output: x is the processed output.

```
maxp, minp = 0.3, 0.5 //Initialized two parameters
lam = random.beta(1, 1) //A random value that obeys the beta distribution
sp = minp + lam * (maxp − minp) //The parameter sp is bigger than 0 and less than 1
rp = random.permutation(x.size(0)) //The parameter is a randomly arranged array
actidx = rand(x.size(1)) //Randomly generate a list based on the size of the feature vectors
sidx = actidx < sp //The value sidx is a binary mask acting on samples
x[:, sidx] = x[rp[:, None], sidx] // This step is to inject noise into the samples
```

This training strategy has more advantages compared with the existing noise injection methods. Firstly, SPS provides a method that allows gradient to suppress overconfident neurons in the process of back propagation. For example, when a certain neuron in the model is highly activated by one kind of sample but negatively affects the prediction of another kind of sample, it is necessary to punish the over-activation of neurons in order

to make the neuron reach the balance of category prediction, which also promotes the expression of more neurons. At the same time, SPS is also a better way to simulate the real noise data to train the classifier. For instance, swapping partial features between the intra-class samples will increase the likelihood that the model will learn the relationship between features and classes, in large part because it injects the real activation value of one sample into another sample rather than artificial noise.

3.3. Improvement

NTS-Net uses ResNet50 as the backbone network to extract features from images, and on this basis, it adds a module to extract multi-scale features. Therefore, NTS-Net can acquire detailed information from images very well, but this also leads to the rapid convergence of NTS-Net in the training set. When reproducing NTS-Net, we used an Nvidia 2080 Ti graphics card and changed the original batch size from 16 to 8 based on the experimental environment. After 30 epochs of training, the loss dropped below 0.009, and the accuracy reached over 99%. At this time, the loss of the test set was about 1.000, and the accuracy was 83.2%. As shown in Figure 2, after training 100 epochs, the loss value and accuracy of the training set were 0.000 and 1.000, while the loss value of the test set was maintained between 0.831 and 0.893, with the highest accuracy of 86.7%. We also tried to continue to increase the number of training epochs, which proved to be unsatisfactory, so we finally selected 140 rounds as the final number of training rounds.

Therefore, the idea of improving NTS-Net was born; this required us to slow down the decline in the loss function in the training set, and further enhance its learning ability. The network feature extractor was effective, and the design of the loss function also involves the whole picture, local features, and the combination of both. Therefore, the noise injection method is a suitable way to optimize the model in one step. We fuse SPS into NTS-Net, which can effectively suppress the neurons that have a great influence on the prediction results and encourage more neurons to perform feature representation. SPS has the ability of data enhancement and can improve the robustness of the classifier. This improved method proved to be effective through our experiments.

The original ResNet network contains dropout, we use SPS in replacement of Dropout in the backbone network of NTS-Net because SPS has the same effect as Dropout. Considering that the feature extractor of NTS-Net has required the multi-scale feature extraction network and the training time cost, this paper only uses the SPS method with a single branch. Without excessive impact on the feature extraction of NTS-Net, this method is only added to the feature extractor of the Scrutinizer network but not used in the other two networks. Experiments show that this design has the best effect on improving network performance, and the improvement of the INTS-Net model is also significant.

As mentioned above, we inject noise by exchanging some feature units, which helps prevent the model from excessive attention on active neurons. Thus, the training loss for a single training instance X is defined as:

$$\widetilde{L}_{tatol} = L_{\mathcal{I}}(I, C) + \widetilde{L}_C\left(\widetilde{R}, X\right) + \widetilde{L}_S\left(\widetilde{R}, X\right) \tag{5}$$

where $\widetilde{R}$ is the informative regions after treatment by the SPS method, Teacher network and Scrutinizer network both use this method in the same feature extractor

Batch Normalization (*BN*) [36] allows the use of much higher learning rates and accelerates network convergence. It is widely used for training convolutional network models. NTS-Net has a large number of *BN* layers in its deep architecture and takes advantage of *BN* to normalize activation. The input feature maps f_{pre} go through the convolutional layers to obtain more advanced features f_{conv}, W_{conv}, and b_{conv} denote Weights and biases in convolutional layers,

$$f_{conv} = W_{conv} \cdot f_{pre} + b_{conv} \tag{6}$$

Consider a mini-batch $A = \{x_1, x_2, x_3, \ldots, x_n\}$ from the previous convolutional layer, n is the batch size and represents values of this activation in the mini-batch. The parameters γ and β are to be learned in training. BN requires mean μ_A and variance σ_A calculations over each mini-batch during training,

$$\widetilde{x_i} \leftarrow \frac{x_i - \mu_A}{\sqrt{\sigma_A^2 + \epsilon}} \tag{7}$$

$$f_i \leftarrow \gamma \widetilde{x_i} + \beta \tag{8}$$

Many studies accelerate inference by fusing adjacent network layers. In this paper, the fusion of a convolutional layer and a batch normalization layer is applied to the feature extractor except for the downsampling block. To save time, the calculation in network runtime is reduced through the combination of parameters between layers. Figure 3 shows the detailed structure of INTS-Net. The fusion process is shown below,

$$\begin{gathered} \hat{f}_{conv} = W_{BN}\left(W_{conv} \cdot f_{pre} + b_{conv}\right) + b_{BN} \\ \hat{f}_{conv} = W_{BN} \cdot W_{conv} \cdot f_{pre} + W_{BN} \cdot b_{conv} + b_{BN} \\ \hat{f}_{conv} = W \cdot f_{pre} + b \end{gathered} \tag{9}$$

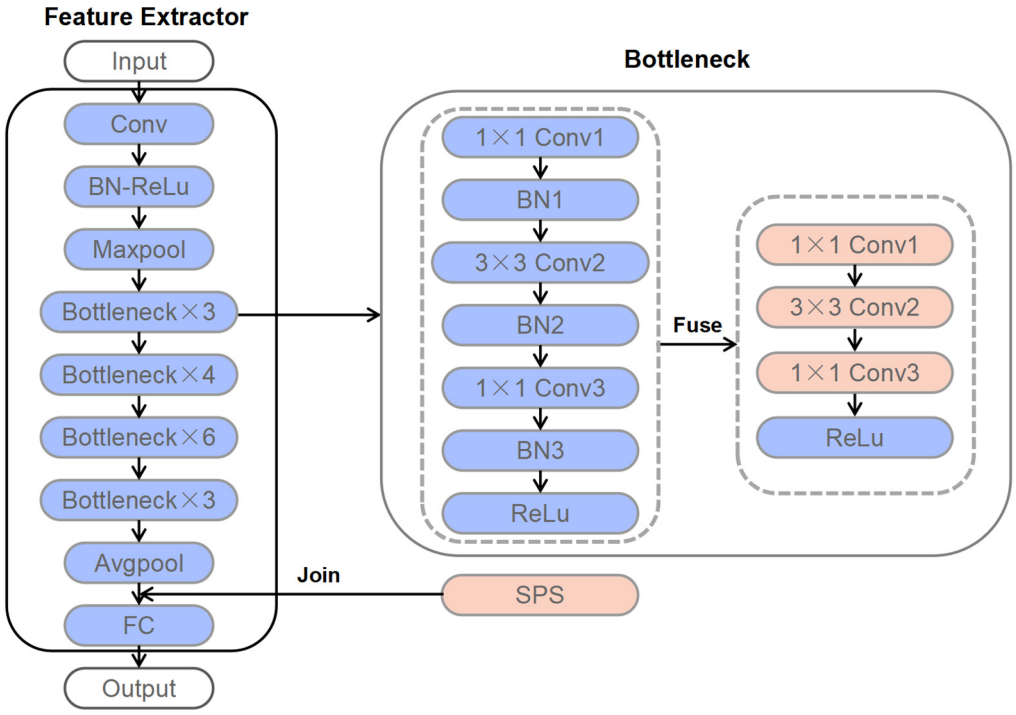


Figure 3. The network structure of ResNet50 shows where changes exist. The SPS is added after the average pooling layer. These two layers are replaced with a single convolutional layer.

4. Experiments

4.1. Dataset

The dataset used in the experiment is Caltech-UCSD Birds (CUB-200-2011) [37], which is one of the most classic and common databases in fine-grained image recognition. There are 11,788 images from 200 wild bird species. Figure 4 shows four kinds of birds. The number of training data and test data are 5994 and 5794 respectively. Meanwhile, the dataset provides abundant manual annotation data, and each image contains 15 local area positions, 312 binary attributes, one annotation box, and semantic segmentation images. The other dataset is Stanford Cars [38], which contains 16,185 images over 196 classes,

and each class has a roughly 50-50 split. The cars in the images are taken from many angles, and the classes are typically at the level of production year and model. NTS-Net is a weakly supervised fine-grained image recognition method, so any bounding boxes or part annotations are not used in all experiments based on our improved method.

Figure 4. Images we used for visual recognition. From left to right, each column contains examples from CUB Bird.

4.2. Implementation Details

The hardware environment configured in the experiment includes a GPU with 8 G video memory, Windows10 operating system, and CUDA version 1.11.0. We use Pytorch to implement our algorithm. NTS-Net used no pre-trained detection model. When reproducing the model, we use a pre-trained detection model and reduce batch size because of the hardware constraints. Table 1 shows detailed comparison information. The weight of pre-trained on ILSCRC2012 is used to initialize the backbone network, and the initial learning rate of the INTS-Net model is 0.001. The public datasets have abundant images, but image sizes are not uniform. In all our experiments, we preprocess images to size 600×600. Then each image is randomly cropped and randomly flipped, which could augment data. During the 140 epochs of training, the learning rate multiplied by 0.1 in the 60th and 100th epochs, so that the goal could continue to decline along the gradient. The hyperparameters M and K in the model respectively act on the number of candidate regions in the NMS method and the selection of K regions from M candidate regions for future prediction. Experiments show that the value of k has a slight effect on the model results and the model works best when K is 4. Considering that the SGDM can be closer to the real gradient, thus increasing the stability of the optimizer, the Stochastic Gradient Momentum (SGDM) was applied to function as the optimizer of the model.

Table 1. The comparison of partial hyperparameters.

	NTS-Net	INTS-Net
pre-trained	False	True
epoch	500	140
batch size	16	8
learning rate	0.001	0.001
momentum	0.9	0.9

4.3. Quantitative Results

After the second convolution layer becomes the group convolution in every bottleneck, different runs in CUB-200-2011 are designed and the results are reported in Table 1. It is discovered that the performance of object recognition can be further demonstrated in terms of recognition accuracy and model parameter. The number of parameters is reduced to 61.4% while the model performance is basically maintained.

We reproduce NTS-Net and SPS, and train the INTS-Net in the same environment based on the aforementioned experiment to verify the performance of the improved algorithm proposed in this paper, as illustrated in Table 2. In order to help the model converge faster, ResNet50′s pre-training parameter on the ImageNet dataset is used. This operation greatly reduces the number of training rounds and saves the training time. In the repeated experiment, the accuracy of NTS-Net is slightly lower than the result mentioned in the original article. There is a small loss of accuracy, and small changes in pre-training, batch size, random disruption of data sets, and the process of model training all had an impact on the final result of the model. On the whole, the results of the two models are not much different from those in the original text. During the training process, the changes in the loss values of NTS-Net and INTS-Net are compared as shown in Figure 5. From the training, it can be seen that the SPS method further increases the difficulty of learning image features in the initial stage of training, and its effect of suppressing overactive neurons is obvious. As shown in Table 3, INTS-Net is tested several times in order to reduce the training error. The average value and corresponding standard deviation are 92.16% and 0.32. On the whole, it promotes the model's learning ability. The final loss value of the improved NTS-Net on the test set of CUB-200-2011 is 0.410, which is nearly 0.440 lower than that of NTS-Net, and the highest accuracy rate is 92.6%, achieving a 5.1% growth. Overall, the effect of group convolution and the SPS method is remarkable.

Table 2. Comparisons on CUB-200-2011 dataset. * refers to the model we reproduce.

Method	Accuracy
NTS-Net	87.5%
NTS-Net *	86.5%
SPS	87.29%
SPS *	87.95%
INTS-Net	92.16%

In the training, the information areas of top-K combined with the whole picture are used as the input of the Scrutinizer network, which promotes fine-grained image recognition. Therefore, apart from making the comparison with the original NTS-Net results, the influence of the K value on the experimental results is also explored. When the value of k is 4 and 6, the final accuracy rate is above 92%. When K = 4, the training effect of the model is the best, and the influence of the value of k on NTS-Net and the improved NTS-Net is basically the same. In order to get a more intuitive understanding, we use two classical network visualization methods, Guided backpropagation, and Class Activation Mapping (CAM), to visualize the last layer of ResNet50. As shown in Figure 6, INTS-Net pays more attention to the head, wings, and other parts with category characteristics when learning bird images. To verify the effectiveness of our methods, we tested our method on

two datasets. A comparative evaluation of the CUB-200-2011 and Stanford Cars datasets is reported in Table 4. Obviously, it can be observed that the accuracy improvement is more significant on the CUB dataset.

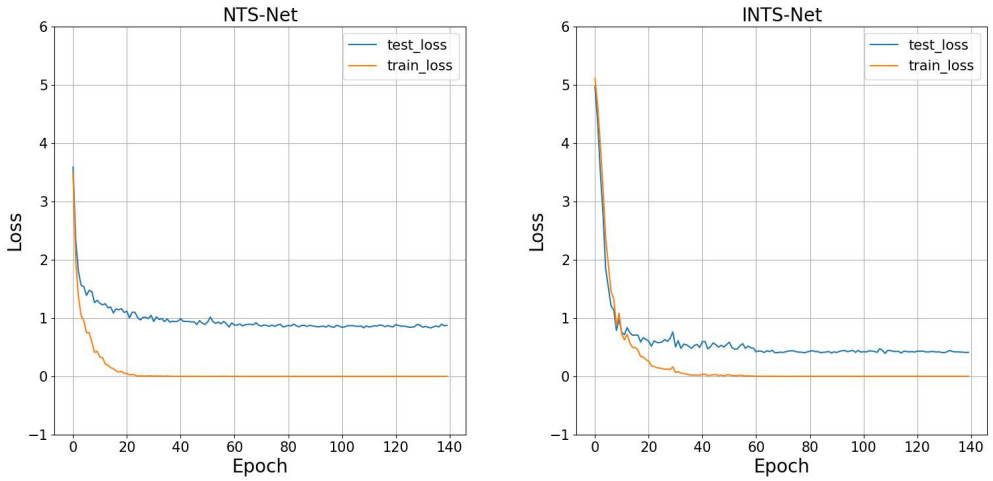


Figure 5. The curves of loss value changes of NTS-Net and INTS-Net on the CUB_200_2011 data set after 140 rounds of training.

Table 3. Repeated Experimental of INTS-Net on CUB-200-2011 dataset.

Number	Accuracy
1	92.6%
2	92.2%
3	91.9%
4	91.8%
5	92.3%

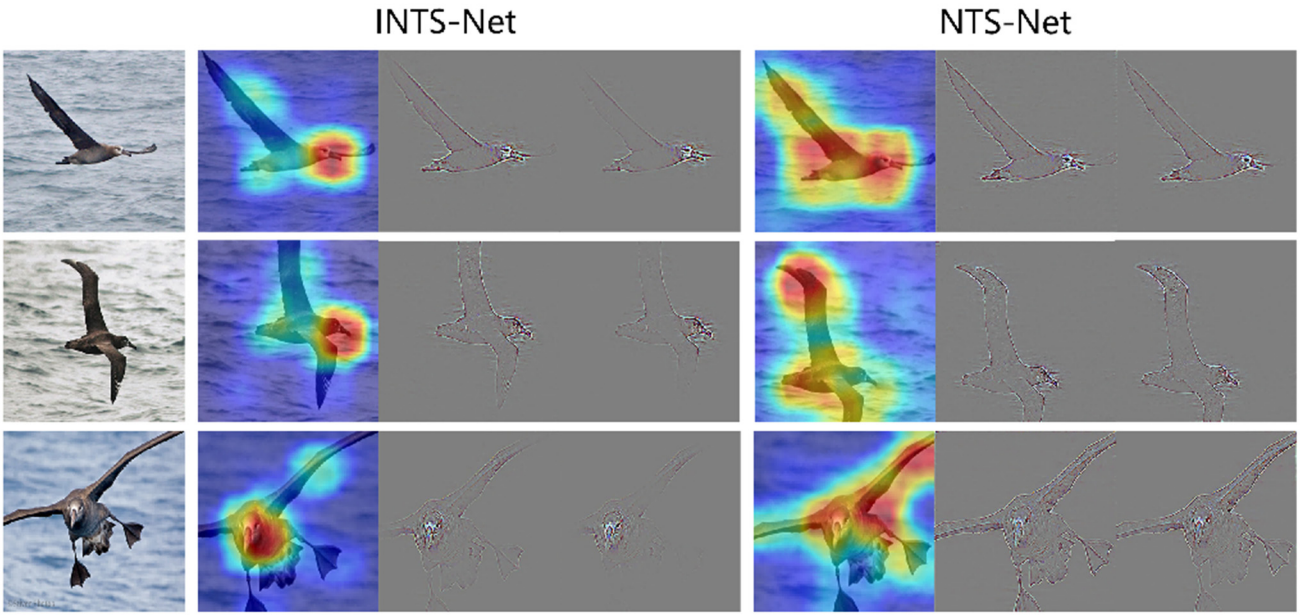


Figure 6. Visualize the features of ResNet50.

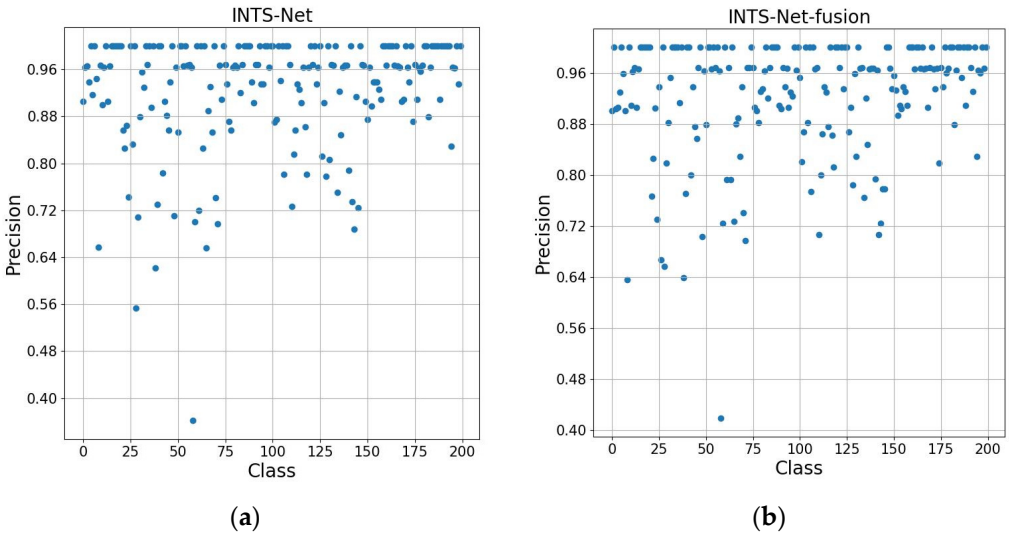
Table 4. Comparison with state-of-the-art methods on classification accuracy with the CUB, the CAR.

Method	Backbone	Year	Accuracy(%)	
			CUB	CAR
Bilinear-CNN [4]	VGG16	2015	84.1%	91.3%
RA-CNN [5]	VGG19	2017	85.3%	92.5%
MA-CNN [15]	VGG19	2017	86.5%	92.8%
Cross-X [21]	ResNet50	2019	87.7%	94.6%
MGE-CNN [39]	ResNet-50	2019	88.5%	93.9%
SPS	ResNet50	2021	88.7%	94.35%
DTRG [7]	ResNet50	2022	88.8%	95.2%
INTS-Net	ResNet50	-	92.16%	94.8%

The feature extractor of NTS-Net consists of a certain number of bottleneck modules that contain many combinations of batch normalization and convolution. After module fusion, the comparison experiment of INTS-Net is run in the same setting, as shown in Table 5. The accuracy of the model marginally decreased by 1.4%, but the accuracy is still much improved compared with NTS-Net. In previous training, due to BN's similar function to bias, bias is not set in the convolutional layer. After fusing the network layer, the weights, and bias in the *BN* are added to the previous convolutional layer, so the overall parameter number of the network hardly changes. INTS-Net improves inference speed by nearly 4.5% on the test set. We added precision and recall of each category as evaluation index and drawn four scatter plots in Figure 7. It can be seen that the model basically maintains all indexes.

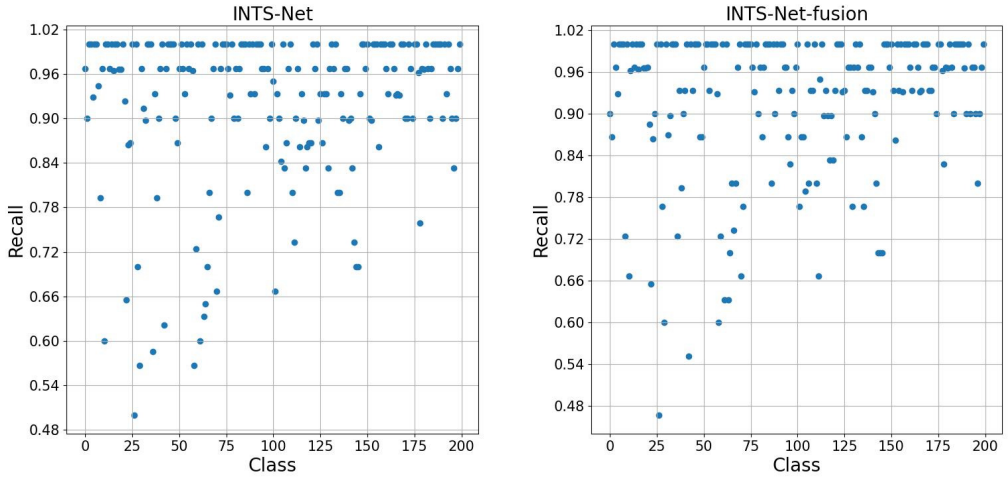
Table 5. Experimental Results in CUB-200-2011. This table shows the comparison with the accuracy, parameters, size, flops, and time.

Method	Accuracy	Size	Flops	Params	Time
NTS-Net	87.5%	116.48 MB	167.19 G	29.03 M	2 min 56 s
INTS-Net	92.6%	117.77 MB	167.19 G	29.03 M	2 min 58 s
INTS-Net-light	91.2%	116.46 MB	165.77 G	29.01 M	2 min 50 s



(**a**) (**b**)

Figure 7. *Cont.*



(**c**) (**d**)

Figure 7. Precision and recall in each category. (**a**,**c**) are the results before fusing *BN* and CNN. (**b**,**d**) are the results after fusing BN and CNN.

5. Conclusions

In this paper, we improved the NTS-Net and proposed the INTS-Net model, which adds the noise injection method SPS and fuses batch normalization and convolution in runtime. We also made flexible adjustments to make the two methods more compatible with the NTS-Net network. The method gives full play to its role in suppressing over-active neurons and enhances the balance. Experiments against state of-the-art methods exhibit the superior performance of our method on various fine-grained recognition tasks. Also, our proposed method is lightweight, easy to train, and agile for inference.

Author Contributions: Conceptualization, J.X. and H.J.; methodology, J.X. and J.Z.; software, J.X.; validation, H.J., J.Z. and S.Z.; formal analysis, H.J.; investigation, H.J. and J.Z.; resources, J.X.; data curation, J.X.; writing—original draft preparation, J.X.; writing—review and editing, J.X., S.Z., T.W., S.L. and Z.L.; visualization, J.X.; supervision, H.J., J.Z. and S.Z.; project administration, H.J. and J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was financially supported by Industry-University-Research Innovation Foundation of Chinese University (2021LDA06003) and Science and Technology Project of Hebei Education Department (QN2020423).

Data Availability Statement: Not applicable.

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

References

1. Fu, J.; Zheng, H.; Mei, T. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4438–4446.
2. Huang, S.; Xu, Z.; Tao, D.; Zhang, Y. Part-stacked cnn for fine-grained visual categorization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1173–1182.
3. Reed, S.; Akata, Z.; Lee, H.; Schiele, B. Learning deep representations of fine-grained visual descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 49–58.
4. Lin, T.Y.; RoyChowdhury, A.; Maji, S. Bilinear CNN models for fine-grained visual recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1449–1457.
5. Sun, M.; Yuan, Y.; Zhou, F.; Ding, E. Multi-attention multi-class constraint for fine-grained image recognition. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 805–821.
6. Zheng, H.; Fu, J.; Zha, Z.J.; Luo, J. Looking for the devil in the details: Learning trilinear attention sampling network for fine-grained image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5012–5021.

7. Liu, K.; Chen, K.; Jia, K. Convolutional fine-grained classification with self-Supervised target relation regularization. *IEEE Trans. Image Process.* **2022**, *31*, 5570–5584. [CrossRef] [PubMed]
8. Shu, X.; Tang, J.; Qi, G.J.; Li, Z.; Jiang, Y.G.; Yan, S. Image classification with tailored fine-grained dictionaries. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *28*, 454–467. [CrossRef]
9. Peng, Y.; He, X.; Zhao, J. Object-part attention model for fine-grained image classification. *IEEE Trans. Image Process.* **2017**, *27*, 1487–1500. [CrossRef] [PubMed]
10. Zhuang, P.; Wang, Y.; Qiao, Y. Learning attentive pairwise interaction for fine-grained classification. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 13130–13137.
11. Ding, Y.; Ma, Z.; Wen, S.; Xie, J.; Chang, D.; Si, Z.; Wu, M.; Ling, H. AP-CNN: Weakly supervised attention pyramid convolutional neural network for fine-grained visual classification. *IEEE Trans. Image Process.* **2021**, *30*, 2826–2836. [CrossRef] [PubMed]
12. Ding, Y.; Zhou, Y.; Zhu, Y.; Ye, Q.; Jiao, J. Selective sparse sampling for fine-grained image recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6599–6608.
13. Chen, Y.; Bai, Y.; Zhang, W.; Mei, T. Destruction and construction learning for fine-grained image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5157–5166.
14. Zheng, H.; Fu, J.; Zha, Z.J.; Luo, J.; Mei, T. Learning rich part hierarchies with progressive attention networks for fine-grained image recognition. *IEEE Trans. Image Process.* **2019**, *29*, 476–488. [CrossRef] [PubMed]
15. Zheng, H.; Fu, J.; Mei, T.; Luo, J. Learning multi-attention convolutional neural network for fine-grained image recognition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5209–5217.
16. Zhang, L.; Huang, S.; Liu, W. Intra-class part swapping for fine-grained image classification. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Virtual, 3–8 January 2021; pp. 3209–3218.
17. Huang, S.; Wang, X.; Tao, D. Stochastic partial swap: Enhanced model generalization and interpretability for fine-grained recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 620–629.
18. Luo, W.; Yang, X.; Mo, X.; Lu, Y.; Davis, L.S.; Li, J.; Yang, J.; Lim, S.N. Cross-x learning for fine-grained visual categorization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8242–8251.
19. Biederman, I.; Subramaniam, S.; Bar, M.; Kalocsai, P.; Fiser, J. Subordinate-level object classification reexamined. *Psychol. Res.* **1999**, *62*, 131–153. [CrossRef] [PubMed]
20. Wei, X.S.; Xie, C.W.; Wu, J.; Shen, C. Mask-CNN: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognit.* **2018**, *76*, 704–714. [CrossRef]
21. Cui, Y.; Zhou, F.; Lin, Y.; Belongie, S. Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1153–1162.
22. Xu, Z.; Huang, S.; Zhang, Y.; Tao, D. Augmenting strong supervision using web data for fine-grained categorization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2524–2532.
23. Yang, Z.; Luo, T.; Wang, D.; Hu, Z.; Gao, J.; Wang, L. Learning to navigate for fine-grained classification. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 420–435.
24. Chang, D.; Ding, Y.; Xie, J.; Bhunia, A.K.; Li, X.; Ma, Z.; Wu, M.; Guo, J.; Song, Y.-Z. The devil is in the channels: Mutual-channel loss for fine-grained image classification. *IEEE Trans. Image Process.* **2020**, *29*, 4683–4695. [CrossRef] [PubMed]
25. Du, R.; Chang, D.; Bhunia, A.K.; Xie, J.; Ma, Z.; Song, Y.Z.; Guo, J. Fine-grained visual classification via progressive multi-granularity training of jigsaw patches. In Proceedings of the Europen Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 153–168.
26. Zhang, X.; Xiong, H.; Zhou, W.; Lin, W.; Tian, Q. Picking deep filter responses for fine-grained image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1134–1142.
27. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. Mixup: Beyond Empirical Risk Minimization. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
28. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6023–6032.
29. Yu, C.; Zhao, X.; Zheng, Q.; Zhang, P.; You, X. Hierarchical bilinear pooling for fine-grained visual recognition. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 574–589.
30. Gao, Y.; Beijbom, O.; Zhang, N.; Darrell, T. Compact bilinear pooling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 317–326.
31. Kong, S.; Fowlkes, C. Low-rank bilinear pooling for fine-grained classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 365–374.
32. Xiao, T.; Xu, Y.; Yang, K.; Zhang, J.; Peng, Y.; Zhang, Z. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 842–850.

33. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
34. Molchanov, D.; Ashukha, A.; Vetrov, D. Variational dropout sparsifies deep neural networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 2498–2507.
35. Li, Z.; Gong, B.; Yang, T. Improved dropout for shallow and deep learning. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2016; pp. 2523–2531.
36. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 6–11 July 2015.
37. Wah, C.; Branson, S.; Welinder, P.; Perona, P.; Belongie, S. *The Caltech-Ucsd Birds-200-2011 Dataset*; Technical Report CNS-TR-2011-001; California Institute of Technology: Pasadena, CA, USA, 2011.
38. Krause, J.; Stark, M.; Deng, J.; Li, F. 3d object representations for fine-grained categorization. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 1–8 December 2013; pp. 554–561.
39. Zhang, L.; Huang, S.; Liu, W.; Tao, D. Learning a mixture of granularity-specific experts for fine-grained categorization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8331–8340.