Investigating Fully-Connected Layers in Few-Shot Classification

Yu Zhou, Tianyu Shi, Jinhong Xu, Zhiwen Wang
School of Computer Science and Engineering, Sun Yat-sen University, China
{zhouy635,shity3,xujh68,wangzhw28}@mail2.sysu.edu.cn

Abstract

Most of the recent studies on few-shot classification have used pre-trained models and finetuning methods, whereas the fully-connected layer after the pre-trained model is a problem that is usually overlooked. Previous research has shown that using one fully-connected layer and freezing most of the parameters of the pre-trained model can effectively generate the model on the finetuning dataset. In this paper, we investigate the effect of the number of fully-connected layers on the model generalization performance, and find that using more fully-connected layers can obtain better result when the finetuning dataset differs too much from the original pre-trained dataset images. In the experiments, our method obtains 76.8 Top-1 accuracy using CNN for backbone and 84.79 Top-1 accuracy using Swin Transformer for backbone on the Skin40 dataset, which shows a significant improvement compared with previous approaches.

1. Introduction

Few-Shot-Classification [1] is a subset of few-shot-learning in which a classifier must learn to distinguish classes that have never been seen from a small set of labeled samples. A FSC task is a self-contained instance that contains both labeled and unlabeled support and query items.

The Sinkhorn Algorithm [2] has been used in a number of works as a parameterless unsupervised classifier that computes fractional matches between query embeddings and class centers. This method is used by many major FSC efforts, including Laplacian-Shot [3], CentroidNet [4], and PT-MAP [5]. The current state-of-the-art is set by Sill-Net [6], which augments training samples with illumination features that are separated from the images in feature space, and PT-MAP-sf [7], which proposes a DCT-based feature embedding network that encodes detailed frequency-domain information that complements the standard spatial domain features.

In addition, some of the more prevalent FSC methods use the meta-learning principle, where the training data is divided into tasks that are similar to the test time tasks that the learner must generalize. MAML [8] "learns to fine-tune" by learning a network initialization from which it can adapt to a new set of classes with only a few gradient update steps on labeled examples. ProtoNet [9] uses distances from support (labeled) class-prototypes in the embedding space to meta-train a learner to predict query feature classes.

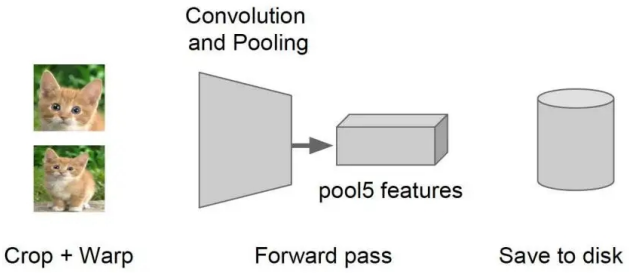


Figure 1. A standard finetuning process that can use different pre-trained models and extract features.

The advantages of meta-learning have been questioned in subsequent works [10, 11], with advocating the usual transfer learning technique of fine-tuning pre-trained networks. The fine-tuning process is illustrated in Figure 1, which shows the benefits of adopting larger and more sophisticated feature-encoding architectures, as well as the use of transductive inference, which fully uses the inference task's data, including unlabeled images.

The limitation of the previous finetuning approach is that only one fully-connected layer is generally used and only that layer is trained. However, if the new dataset is too different from the original pre-trained dataset, training only one fully-connected layer cannot fully learn all the features of the new dataset, which can seriously affect the performance of the model.

To solve this problem, we investigate the effect of multiple fully-connected layers after pre-training the model. The reason for using this approach is that multiple fully-connected layers should theoretically enable the model to learn high-level abstract features of the new dataset while maintaining the original pre-trained features. In addition, we also use a number of data augmentation and regulariza-

tion methods, including RandomCrop, RandomFlip, RandAugment, L2 Regulization, etc. We try various models based on CNN and Vit backbones, including ResNet, VGG, DenseNet, Swin Transformer, etc. As a result, applying our method on several model structures was able to demonstrate the generalizability of our method.

To evaluate the proposed method, we build our model a variety of backbones, and evaluate on the Skin40 dataset. The experimental results show that our method brings significant improvements to few-shot classification. In conclusion, the main contributions of our research can be summarized as follows:

- We investigate the effect of different number of fully-connected layers on few-shot classification and find that using more than one fully-connected layer can effectively improve the performance of the model.
- We find that using pre-trained models without freezing parameters can obtain better results when finetuning dataset is too different from original dataset.
- Experimental results show that our method outperforms our baseline, which achieves 84.79 Top-1 accuracy on the Skin40 dataset.

2. Related Work

2.1. AlexNet and VGG

LeCun [12] designed LeNet5 for the recognition of handwritten numbers, which marked the real introduction of CNN. However, this model did not catch on for a long time due to the limitations of the machine equipment. Later, Alex [13] improved LeNet5 in three main ways: using ReLU as the activation function of the CNN; adding Dropout to the training to deactivate some of the neurons randomly, and using overlapping maximum pooling in the CNN. This resulted in a substantial improvement in the performance of the model. On the imageNet2012 image classification task, AlexNet topped the list with an error rate of 15.3%, and was more than ten percentage points ahead of the second place. The VGG proposed by Karen Simonyan and Andrew Zisserman [14] uses several consecutive 3x3 convolutional kernels instead of the larger convolutions in AlexNet, which boosted the depth of the network and to some extent the effectiveness of the neural network while ensuring the same perceptual field. VGG placed second in the 2014 ILSVRC competition.

2.2. ResNet and DenseNet

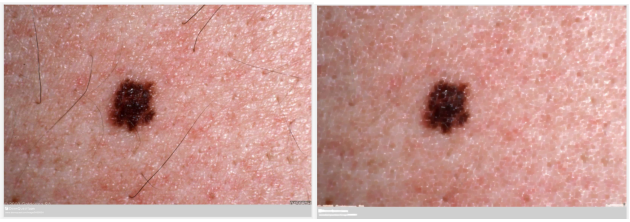
Starting with AlexNet, the network structure has gradually developed in the direction of deeper and deeper networks. It was intuitively thought that as the depth of the network increases, the network's feature fitting ability becomes stronger, and therefore deeper models should achieve better results. The problem of vanishing and exploding gradients has been largely solved by the introduction of batch normalization. Kaiming He [15] believes that this is an optimization problem and that as the network deepens, the network becomes more difficult to optimize. He introduced a short connection into the neural network, allowing the stack layer to learn the residuals rather than the underlying mappings, which would make it easier to optimize, and this was ResNet. Kaiming He [16] further adjusted the order of the components in ResNet to make the network easier to optimize, and the deeper the network, the more effective it became. Based on ResNet, ResNeXt [17] , SENet [18], ResNeSt [19] and other models were later proposed, all with very good results. With the development of ResNet and related network structures, many models expand in width and depth, such as fractal network[20], and achieve competitive results. On the contrary, DenseNet improves the potential of the network through feature reuse and obtains an efficient model which is easy to train. Compared to Inception networks[21], which also concatenate features from different layers, DenseNet are simpler and more efficient.

2.3. Swin Transformer

Transformers are proposed by Vaswani et al. [22] for machine translation, and have since become state-of-the-art method in many NLP tasks. The transformer has been gradually introduced into the field of computer vision in recent years. Minghang Zheng and Peng Gao [**?**] proposed the DERT model, which introduces the Transformer for target detection tasks. Vision Transformer [23] completely reuses the model structure of Transformer in NLP to solve image problems. However, ViTs have corresponding drawbacks. Large training datasets are required. Memory access is expensive and latency is high. And the number of parameters is large and computational complexity is high. For the first problem, Hugo Touvron and Matthieu Cord [24] introduces several training strategies that make it possible to obtain a better result without additional training datasets. Ze Liu and Yutong Lin [25] propose a moving window and hierarchical design to solve the problem of computational complexity and expensive memory access. It reduces the complexity from quadratic to linear in image size and also performs local operations, which proves beneficial in modeling the high correlation of visual signals. Swin Transformer achieves very high accuracy in both COCO object detection and ADE20K semantic segmentation. Xiaoyi Dong and Jianmin Bao [26] develop the Cross-Shaped Window self-attention mechanism for parallel computation of self-attention in horizontal and vertical stripes that form a cross-shaped window, which can achieve 85 Top-1 on ImageNet-1K without any additional training data, showing competitive performance on common vision tasks.

Table 1. Mean and Std for images in Skin40 dataset.

	R	G	B
Mean	0.6075306	0.4911691	0.4606611
Std	0.2260388	0.2162352	0.2191065



(a) Raw image (b) Processed image

Figure 2. Image comparison before and after processing.

3. Method

In this section, we first introduce our Normalization and Data Augmentation methods to preprocess skin images. Then, we present our approach of multiple fully-connected layers. Finally, we describe the overall architecture and training details of our CNN and Swin Transformer backbones.

3.1. Normalization

Normalization is an important step in preprocessing an image dataset. Normalization requires the mean and standard deviation of each channel in the RGB image, and if these values are incorrect, it can cause a bad impact on the final performance of the model.

As indicated in Table 1, we start by writing code to calculate the mean and standard deviation of the Skin40 dataset, which may be used as prior knowledge to help with Normalization during training. Because the images in the Skin40 dataset are pathology images, the mean value of the red channel is much higher than the mean values of the green and blue channels. In addition, the standard deviations of all three channels are comparable.

3.2. Morphological Processing

After observing the images in the data set, we found a problem with hair covering the affected area in many images. We suspect that this problem may affect the effectiveness of training. Therefore, we use morphological closing operation to remove hair.

The close operation eliminates hairs of varying size by setting the size of the structural element. However, we found that the structure element was too small to remove hair, but the structure element was too large to cause the image to become blurred. Therefore, we adopt a structure element of intermediate size to close the operation. The close operation removes small hairs. Comparing the original figure2(a) with the processed figure2(b), the morphologic operation did remove the hair.

However, this kind of preprocessing method also make the image become blurred. As a result, the training was slightly less effective than not removing the hair.

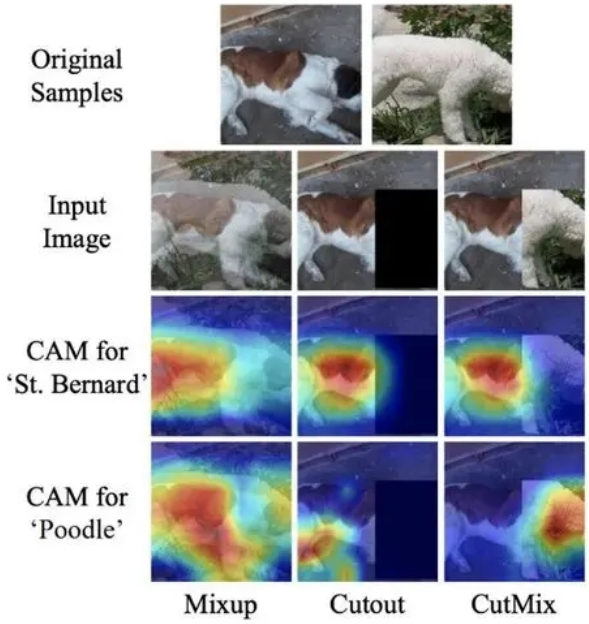


Figure 3. Mixup and CutMix.

3.3. Data Augmentation

RandomFlip Each image in the training set is divided into multiple parts, and each image is rotated randomly at first, then flipped horizontally and vertically with a probability of 50%, to obtain different transformation forms of the original image.

BatchMixup Mixup is a blending algorithm for data augmentation that expands the training dataset by blending images between different classes. Suppose $batch_{x1}$ is a batch sample and $batch_{y1}$ is the label corresponding to this batch sample. $batch_{x2}$ is another batch sample and $batch_{y2}$ is the label corresponding to this batch sample. λ is the mixing factor calculated from the *Beta* mixing distribution with parameters α and β, the corresponding formulas are as follows:

$$\lambda = \mathrm{Beta}(\alpha, \beta) \tag{1}$$

$$mixed\_batch_x = \lambda * batch_{x1} + (1 - \lambda) * batch_{x2} \tag{2}$$

$$mixed\_batch_y = \lambda * batch_{y1} + (1 - \lambda) * batch_{y2} \tag{3}$$

BatchCutMix CutMix is a combination of CutOut and Mixup. As shown in Figure 3, CutMix, compared to Cutout,

Original Image Image after cropping

Figure 4. RandomResizedCrop

is a region deletion operation that takes a region of the same size as the other image and fills it, while changing the label of the new image. Mixup soft-fuses the two images and also soft-fuses the labels of the two images, thus utilizing the information of the whole image. CutMix, on the other hand, hard-fuses the two images and adopts the soft-fusion strategy of the label of Mixup. Such processing makes CutMix does not change the distribution of the whole dataset.

RandomResizedCrop Crop the given image to random size and aspect ratio. As shown in Figure 4, a crop of random size (default: of 0.08 to 1.0) of the original size and a random aspect ratio of the original aspect ratio is made. This crop is finally resized to given size, such as 224 and 384, which is suitable for the pre-trained models.

Colorjitter Colorjitter randomly adjusts common image information such as brightness, contrast, saturation, and hue within a certain range. This is a common data augmentation method, which can generate color-level noise in training. In this way, it can avoid interference to the model due to the shooting problems of equipment and light.

RandAugment RandAugment builds on AutoAugment with a significantly reduced search space, which allows it to train on the target task and significantly improve the generalization ability of the model. Moreover, due to the parameterization, the regularization strength can be tuned to different models and dataset sizes. In our approach, we choose an enhanced magnitude level of 0.9 and a deviation of 0.5 for the magnitude noise.

RandomErasing We randomly select a rectangular area in the image and erase the pixels. This method is a lightweight approach that does not require any additional parameters or memory consumption, and it can be integrated with various models without changing the learning strategy and improving the robustness of the model.

CenterCrop We randomly take a region in the center of the image. This method can be regarded as a special case of RandomErasing, and can ensure to retain the important content of the image center.

As a result, it can be found that mixup can improve several points of accuracy of the model, because the method expands the dataset, increases the disturbance to suppress over-fitting, and enhance the generalization ability of the model. On the other hand, such a method as Colorjitter will significantly reduce the convergence speed and accuracy of the model. It may be that the color of the lesion area is more important for the classification of diseases in medical images. Moreover, the skin color in the picture is primarily yellow. The skin color will change after Colorjitter, which increases the difficulty of model fitting. We also found that the model cannot distinguish different kinds of nevus, and CenterCrop is able to solve this problem.

3.4. Fully-Connected Layers

We apply different fully-connected layers after pre-trained models. In order to first evaluate the performance of each common baseline model on the skin40 test set, we freeze all parameters of the pre-trained model except the last fully-connected layers during training, and only update the parameters of the last full-connected layer to test the transferability of pre-trained models. For CNN backbone model, we discover that only AlexNet achieves an accuracy of nearly 50%, and the accuracy of other models could only be at around 10%. This is the exact opposite of the often poor performance of AlexNet on most datasets. After comparison, we find that this is because the fully-connected layer of AlexNet has two layers, and the fully-connected layer of other models is only set to one, which leads to the performance gap of nearly forty percent. If the experiment is performed without freezing all parameters, AlexNet is still the worst, probably because the performance gap of the AlexNet model is difficult to make up for the performance gap brought by the last fully-connected layers.

Inspired by this, we evaluate the performance of different models when modifying all parameters on the skin40 dataset. For details, please refer to Section 4. The following properties are found in the experiment:

- The changing trend of the performance of frozen and unfrozen parameter models when changing the number of FC layers is the same, but the unfrozen model has a better performance. Because the pre-trained backbone network is trained on Imagenet, the features that need to be extracted from medical images are different. After pre-training, it can better adapt to the current task. If the number of FC layers increases when

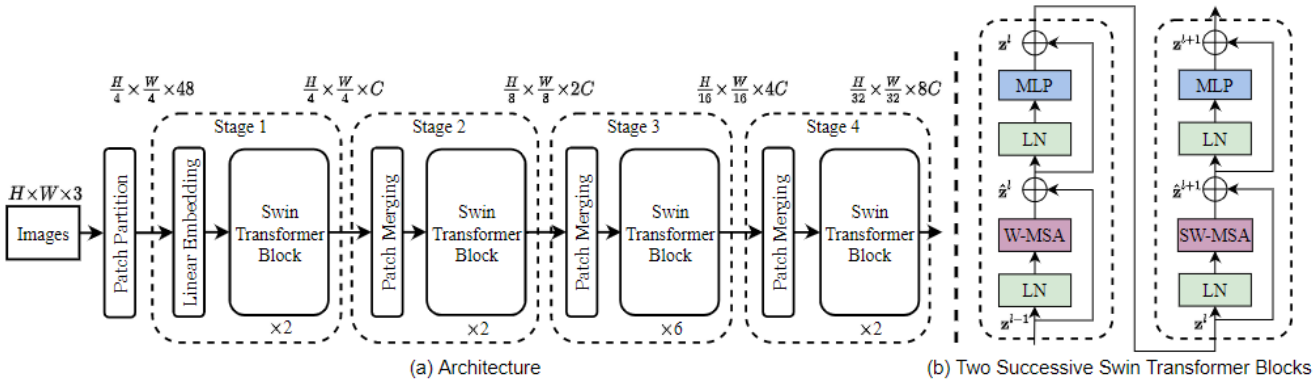


Figure 5. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks. W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

the parameters are fixed, the learnable parameters of the model will be improved, which will bring greater performance improvement than unfixed parameters.

- Compared with one fully-connected layer, the two fully-connected layers setting of other models generally brings a little performance improvement. Increasing the number of number of FC layer can enhance the ability of the model to predict through graphic features. One FC layer is a simple linear combination, and the two-layer FC layer network contains a nonlinear layer. After training, the two-layer FC layer network can fit a nonlinear function, so it can bring significant improvement.

- Compared with the three and four FC layers settings, the two FC layers can guarantee better results. This may be related to the small number of training epochs due to limited computational resources. In the experiment, the accuracy of the model corresponding to the four FC layers setting has dropped by more than a dozen percent. It is likely that the FC layers does not converge or the generalization performance is poor, and the performance of the three FC layer setting also declines for the same reason.

- Through experiments, we discover that increasing the number of fully-connected layers has different effects on different models. The performance of complex models decreases when the number of FC layers changes from 1 to 2 such as DenseNet-161, while the performance of simple models increases such as AlexNet, VGG-11 and so on. It is because the ability to extract image features of the simple model is weak, but the number of epoches required for convergence is few. Therefore, the addition of FC layers brings the improvement of classification ability. Complex models have a strong ability to extract features, and too many FC layers will lead to the disappearance of gradient and slow convergence speed, which lead to poor performance. However, Resnet makes the model easy to converge through skip connection, so it can still converge at the same epoch after increasing the number of layers from 1 to 4, and because the complexity of the model increases, it brings a better performance.

  Complex models have a strong ability to extract features, and too many FC layers will lead to the disappearance of gradient and slow convergence speed. There is no similar phenomenon on ResNet, which may be because the structure of ResNet is much simpler than DenseNet, so its performance can still be improved.

In summary, the number of FC layers in few-shot learning plays an important role, especially for complex pretrained models. Taking into account the convergence speed and model performance, the two FC layers setting is a more appropriate choice.

3.5. Swin Transformer

Figure 5 shows a high-level overview of the Swin Transformer architecture. A patch splitting module splits an input RGB image into non-overlapping patches first. Each patch is considered as a "token", with its feature set to the raw pixel RGB values concatenated. The patch size is 4×4, hence each patch's feature dimension is $4 \times 4 \times 3 = 48$. This raw-valued feature is projected to an arbitrary dimension using a linear embedding layer (denoted as C).

On these patch tokens, many Transformer blocks with modified self-attention computation (Swin Transformer blocks) are applied. The quantity of tokens is maintained by the Transformer blocks ($\frac{H}{4} \times \frac{W}{4}$). As the network goes deeper, the number of tokens is reduced via patch merging layers to provide a hierarchical representation. The first patch merging layer concatenates the features of each set of

Table 2. Top-1 accuracy for CNN models with different full-connected layers on the test set.

Model structure	1-FC layer	2-FC layers	3-FC layers	4-FC layers
AlexNet	64.4	67.5	66.0	64.1
VGG-11	63.6	66.5	64.3	54.0
VGG-16	65.1	66.2	64.6	49.5
ResNet-50	73.3	75.2	73.5	73.2
ResNet-101	72.6	74.5	73.3	72.2
DenseNet-121	71.7	73.5	74.7	61
DenseNet-161	76.8	75.4	73.2	64.5

two adjacent patches and applies a linear layer to the concatenated 4C-dimensional features. The number of tokens is reduced by a factor of $2 \times 2 = 4$, and the output dimension is set to $2C$. After that, Swin Transformer blocks are used to alter the features, with the resolution kept at $\frac{H}{8} \times \frac{W}{8}$.

The technique is done twice more, with $\frac{H}{16} \times \frac{W}{16}$ and $\frac{H}{32} \times \frac{W}{32}$ output resolutions, respectively. These steps operate together to provide a hierarchical representation with the same feature map resolutions as traditional convolutional networks, such as VGG [14] and ResNet [15]. As a result, Swin Transformer may easily replace existing approaches' backbone networks for diverse visual tasks with better performance.

After the last self-attention layer of Swin Transformer, we set several fully-connected layers. Note that the number of in-channels of the first fully connected layer must be the number of out-channels of self-attention layers (1536), and the number of out-channels of the last fully-connected layer is 40. Since the model freezes most of the parameters of the model on the finetune dataset, and it is difficult to extract all new features with only one fully-connected layer, using multiple fully connected layers can obviously achieve better performance.

4. Experiments

To evaluate our proposed method, we follow a standard five-fold cross-validation strategy on the Skin40 dataset. Skin40 is a classification dataset containing 40 skin diseases with 60 images for each category. If we use five-fold cross-validation, there are 1920 (48*40) images for training, and the test set has 480 (12*40) images. In the Skin40 dataset, the images have different resolution sizes, such as 1440*1080 and 260*480, which is a problem we need to address carefully when preprocessing the images.

During our experiments, we try CNN backbone models such as ResNet and VGG, and also try ViT backbone models such as Swin Transformer. We use Top-1 Accuracy as our performance metric.

Table 3. Top-1 accuracy for CNN models whether using data augmentation on the test set.

Network structure	No data augumentation	data augumentation
AlexNet	55.8	63.0
VGG-11	51.9	63.9
ResNet-50	68.4	73.3
DenseNet-121	61.2	71.5

4.1. Setup

CNN In order to apply CNN model to the current 40-category task scenario, we need to change the last layer of different models so that the output becomes a 40-dimensional vector. Therefore, we replace the last fully-connected layer of the different models with the Fully- Connected Layer we designed. During the design process, we wonder if replacing it with more fully connected layers would produce a better result. Therefore, we replace the last layer of the model with one to four fully-connected layers respectively, and evaluate the performance of the model. As illustrated in Table 2, different layers have certain impacts on experimental performance.

Swin Transformer We use Swin Transformer as the backbone model to train four pre-training models with different parametric sizes: Swin-tiny, Swin-small, Swin-base, and Swin-large. For Swin-tiny, Swin-small, Swin-base, and Swin-large pre-trained models with different sizes, we set batch size to 512, 256, 128, and 64, respectively. Additionally, we employ CosineAnnealing to dynamically adjust the learning rate during training, as well as Warmup to begin training with a small learning rate. In addition, we train all of our Swin Transformer models on NVIDIA GeForce RTX 3090 GPU.

4.2. Results of CNN models

By comparing the results of different networks whether using data augmentation, we find that the accuracy of different network structures on the test set has been significantly improved after data augmentation. As illustrated in Table 3, the accuracy of VGG-11 is even improved by 12 points, indicating that the use of data augmentation is very important for the current small training set.

There is a phenomenon of hair occlusion in some data. We think it might have some impact on the network. Therefore, we use morphological closure operation to remove hair. However, this operation resulted in a degree of blurring in the image. As shown in Table 4, it has no positive

Table 4. Top-1 accuracy for CNN models whether using hair removal on the test set.

Network structure	Hair removal	No Hair removal
AlexNet	61.3	63.0
VGG-11	60.3	63.9
ResNet-50	71.6	73.3
DenseNet-121	69.7	71.5

Table 5. Top-1 accuracy for CNN models fixed and unfixed fully-connected layers on the test set.

Network structure	fixed	unfixed
AlexNet	49.5	63
VGG-11	6.7	63.9
ResNet-50	14.3	73.3
DenseNet-121	8.5	71.5

effect on the final result.

We try to freeze the network layer except for the last layer of the network, compared with the unfrozen case. As illustrated in Table 5, we find that several network models generally performe poorly after freezing, but only AlexNet obtains a higher accuracy. We speculate that the reason is the last layer of Alex is composed of multiple fully-connected layers and ReLU activation layers, so the model has better fitting ability. To prove this idea, we made the same change on several other models, replacing the last fully connected layer with a multi-layer Fully Connected Layer, and find that the general effect is improved, reaching about 50% accuracy.

However, the results of the above attempts are still worse than the results of changing all network layer parameters of the model. We speculate that there is a large gap between the image used in pre-trained model and the current task image, which makes the effect of pre-training for the current task poor.

4.3. Results of Swin Transformer

Although all of our experiments with Swin Transformer as the backbone achieve 100% Top-1 accuracy on the training set, they are extremely prone to overfitting due to its extremely stringent training requirements. As illustrated in Table 6, we use four different parameter sizes models as our baseline, respectively. And there is a slight improvement in finetuning as the number of model parameters grows.

We obtain the best result using Swin-L model with 384 resolution, achieving 83.58 Top-1 accuracy on Skin40 test set. Since most of the images in the dataset are over 1000 resolution, it is obvious that using a 384 resolution model can get better result.

Table 6. Top-1 accuracy for Swin Transformer models with different fully-connected layers on Skin40 test set.

Model Structure	1-FC layer	2-FC layers	3-FC layers	4-FC layers
Swin-T 224	70.65	72.53	**72.88**	55.12
Swin-S 224	74.95	**77.44**	73.07	55.81
Swin-B 224	77.21	80.45	73.19	57.32
Swin-B 384	79.88	**82.22**	74.87	57.82
Swin-L 224	80.91	81.82	75.29	57.35
Swin-L 384	82.43	**83.58**	76.43	58.27

Table 7. Top-1 accuracy for Swin Transformer with ensemble learning.

Model	Single	Ensemble
Swin-L 384 with 2-FC layers	83.58	**84.79**

Table 8. Top-1 accuracy for Swin Transformer models compared with freezing parameters.

Model Structure	1-FC layer	2-FC layers	3-FC layers	4-FC layers
Swin-B 384 (freezing)	67.28	70.32	63.77	61.23
Swin-B 384	79.88	**82.22**	74.87	57.82
Swin-L 384 (freezing)	68.93	70.87	65.47	62.31
Swin-L 384	82.43	**83.58**	76.43	58.27

In addition, we obtain the best result on Swin-T with 3 fully-connected layers and on Swin-S, Swin-B and Swin-L with 2 fully-connected layers. It shows that using more than one fully-connected layer can exactly help to improve the model performance. However, using too many fully-connected layers can also significantly degrade the performance.

Furthermore, we discover that two fully-connected layers on Swin Transformer are not as effective as CNN, and we speculate that it may be the influence of MLP in Swin Transformer itself.

We also try to train 5 Swin Transformers for ensemble learning. As shown in Table 7, the result using ensemble learning is improved by 1 point compared with a single Swin Transformer, reaching 84.79 Top-1 accuracy.

Figure 6. features on image's right edge

Figure 7. features too tiny to be catch

Figure 8. Blue-nevus sample.

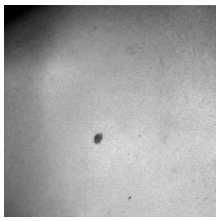


Figure 9. Compound-nevus sample.

Figure 10. Interference left.

Figure 11. Interference Right.

4.4. Ablation Study

Effect of freezing parameters Due to the limited number of samples in the training set, previous research suggest that freezing most of the network layers and train only the last fully-connected layer is supposed to obtain better result. However, out experimental result show that without freezing parameters of pre-trained models can achieve higher accuracy. The compared results are shown in Table 8.

Effect of CutMix A common problem in small sample learning of medical images is that the improvement of some classes may lead to the decrease of the number of other classes. By combining multiple images and generating soft labels, we can follow the training distribution on the data set as much as possible in the training process, so as to avoid the influence of data distribution error on the experimental results in the training process. Through experiments, it is found that it can improve the performance of one point, and the features brought on the original lower label are more obvious, up to 3 points.

Effect of CenterCrop In the experiment, we find that the performance is poor in the classes of nevus. It may be that the area of nevus is relatively small and the area of skin is relatively large, so the model cannot extract the important information about nevus. Then we find that CenterCrop can greatly improve the classes related to nevus. By comparing the images, we also discover that this is because the image background is complex, which has some irrelevant information, and the area of the nevus is relatively small. Considering the good quality of the captured image, most of the lesions are distributed in the center of the image, CenterCrop can extract important areas in the image, ignoring the background. The study shows that the convergence speed and performance of these classes have been greatly improved with CenterCrop, which can improve the accuracy of 5 in these categories to about 40. And it will not have a great impact on other classes.

4.5. Error Analysis

Blurred Features For instance, in the following two images from the sample Blue nevus, the center of Figure 6 must be focused on the image's right edge, and details in Figure 7 are too tiny to be recorded adequately by CNN backbones.

Blurred Classification Boundaries The visual characteristics of the Blue nevus sample and the compound nevus sample are extremely comparable. It is difficult to discern these two images based on their visual characteristics, and other classes in Skin40 dataset have similar difficulties. In the illustration in Figure 8 is the Blue nevus sample, while the illustration in Figure 9 is the Compound nevus sample.

Interference in Feature Some images do not effectively depict the visual features of disease, and there is an excess of irrelevant interference information. For instance, the model may capture image aspects of hands Figure 10 and feet Figure 11 rather than pathogenic abnormalities.

5. Conclusion

In this paper, we first try both Normalization and Data Augmentation methods to preprocess skin images. We also use morphological closing operation to remove hair. These operations facilitate better training. Then we use CNN and Swin Transformer for backbone, based on which we further investigate the effect of the number of fully-connected layers on the the model generalization performance. Our experiments show that using multiple fully-connected layers can obtain better results when the finetuning dataset differs too much from original dataset, which achieves significant improvements compared with previous approaches.

References

[1] Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *neural information processing systems*, 2016. 1

[2] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *neural information processing systems*, 2013. 1

[3] Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. *international conference on machine learning*, 2020. 1

[4] Gabriel Huang, Hugo Larochelle, and Simon Lacoste-Julien. Are few-shot learning benchmarks too simple ? solving them without task supervision at test-time. *arXiv: Learning*, 2019. 1

[5] Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Leveraging the feature distribution in transfer-based few-shot learning. *arXiv: Learning*, 2020. 1

[6] Haipeng Zhang, Zhong Cao, Ziang Yan, and Changshui Zhang. Sill-net: Feature augmentation with separated illumination representation. *arXiv: Computer Vision and Pattern Recognition*, 2021. 1

[7] Xiangyu Chen and Guanghui Wang. Few-shot learning by integrating spatial and frequency representation. *canadian conference on computer and robot vision*, 2021. 1

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *international conference on machine learning*, 2017. 1

[9] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *neural information processing systems*, 2017. 1

[10] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *Learning*, 2019. 1

[11] Guneet S. Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. *Learning*, 2019. 1

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 2

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of The ACM*, 2012. 2

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015. 2, 6

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv: Computer Vision and Pattern Recognition*, 2015. 2, 6

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *arXiv: Computer Vision and Pattern Recognition*, 2016. 2

[17] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *arXiv: Computer Vision and Pattern Recognition*, 2016. 2

[18] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks. *computer vision and pattern recognition*, 2018. 2

[19] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Mueller, R. Manmatha, Mu Li, and Alexander J. Smola. Resnest: Split-attention networks. *arXiv: Computer Vision and Pattern Recognition*, 2020. 2

[20] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *ArXiv*, abs/1605.07648, 2017. 2

[21] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 2

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *neural information processing systems*, 2017. 2

[23] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv: Computer Vision and Pattern Recognition*, 2020. 2

[24] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv: Computer Vision and Pattern Recognition*, 2020. 2

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv: Computer Vision and Pattern Recognition*, 2021. 2

[26] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *arXiv: Computer Vision and Pattern Recognition*, 2021. 2