Report for Project 1: Skin disease diagnosis

Renjie Lu
Sun Yat-Sen University
lurj3@mail2.sysu.edu.cn

Long Chen
Sun Yat-Sen University
chenlong36@mail2.sysu.edu.cn

Abstract

The project that we choose is Project 1, in which we need to develop a novel model to solve the 40-class skin disease classification task. The neural network we use is ResNet. Since the data set is limited, we also use some data augmentation methods to improve the performance of the neural network, such as some basic transforms accessible in torchvision ,CutMix and so on. We also tried to use SinGAN to generate more data to improve the performance. Besides, we improved the models performance by assign different weight on each class according to the recall of each class. We also tried to analyse confusion matrix to find the low-accuracy classes and confused classes. However, the confused classes is hard to solve, we have tried fined-grained classification methods but it does not help. Finally we trained a vision transformer which obtained a higher accuracy.

1. Introduction

Statistics indicate that the skin diseases share a large quantity among all the diseases in our daily life and there are hundreds of common skin diseases in the clinical. Consequently, diagnosing the skin diseases correctly is a troublesome burden on doctors' shoulder.

Based on the above situation, we decided to choose project 1 in which we need to develop a neural model to solve the 40-class skin diseases classification task on Skin40 data set, with the hope that this model could be put into practical application to reduce the burden on doctors. Since the ResNet [7] performs well in this kind of classification task, we adopt ResNet as our baseline.

The main challenge in this project is that only 60 images are available for each skin disease class, which is far from enough for the training of a neural model. So this is a typical small-sample recognition problem. Additionally, there are several skin disease classes in Skin40 with similar pathological features. Some of them is quite confused even for human to distinguish, which is undoubtedly a disturbance for the neural model on classification task.

To solve these problem, we mainly make the improvement on these direction. The first one is data augmentation. We not only adopt some basic transforms accessible in torchvision ,CutMix, MoEx and so on, but also try some GAN-based data augmentation methods. The second one is some training techniques, such as 5-fold cross-validation, triplet loss, label smoothing, etc. We also improve the neural model by assigning different weight on each class according to the recall of each class. Finally we train a vision transformer which obtained a higher accuracy. We will introduce these approaches we try in Section 4 in detailed. In short, all these approaches focus on solve a specific defect of the baseline model, and most of them could improve the accuracy and recall rate of the neural model performance.

The baseline neural model achieves the performance at the highest average accuracy 75.25%, which is reach by ResNet101. And after the improvement approaches are adopted, the neural model can achieves the performance with the highest average accuracy 78.67%.

In this paper, we introduce an improved neural model, which perform well on the classification task on Skin40 date set. The neural model we obtained on the base of ResNet can overcome limitation of Skin40 data set to an extent, such as the lack of training data and the confusion between some skin diseases, and learn a relatively well-perform neural model.

2. Related work

Network backbone Convolutional neural network [23] is a powerful model for image classification which pushes the accuracy of image classification to new heights. However, very deep CNN is very difficult to train. ResNet [6] solves this problem and becomes one of the most successful and widely used CNN architectures. Beside traditional Convolutional neural network. Stronger convolutional-free model Transformer [20] is proposed in natural language processing. Vision Transformer [3] is a pioneer work that directly adopt Transformer architecture on image-patches for image classification and outperforms most CNN models.

Model	Average Accuracy
ResNet18	69.21 ± 1.488
ResNet50	74.17 ± 0.755
ResNet101	75.25 ± 1.828

Table 1. Classification accuracy(%) on Skin40.

Data augmentation Data augmentation [17] is an important step to avoid deep model being overfiting and generize better. Regional dropout based methods [26] like CutOut [2] have been proved to enhance the performance of Convolutional neural network by guiding the model to attend on less discriminative parts. Interpolation based methods like Mixup [25] use linear interpolation to generate new samples and labels to minimize vicinal risk and enhance generization performance. CutMix [24] combines the advantages of Mixup and Cutout and becomes a stronger data augmentation methods.

Contrastive learning Contrastive learning is a training technique that make model focus more on differences between classes and extract more distinguishing features. Triplet loss [15] is an importance contrastive learning loss which minimize the distance between intra-class features and maximize the distance between inter-class features. Here we use the triplet loss to supervise the model extracting more discriminatvie features and improve classification accuracy.

3. Baseline

We select ResNet pretrained on ImageNet as the baseline model beacuse of it's strong performance. Due to the limit number of this dataset, we conduct 5-fold [4] cross-validation for a more reliable evaluation of the model. On each fold, 48 images will be sampled to train and the rest 12 images will be used as test data. Therefore, there are 1920 train images and 480 test images. For all the following experiments in this report, We use Adam [9] optimizer to train the model for 60 epochs with initial learning rate 1e-4 and weight decay 1e-3. The learning rate drops to it's 0.8 times every 2 epochs.

To investegate which resnet performances the best, we use ResNet18, ResNet50, ResNet101 [22] as the classifier and record there accuracy of 5 folds in Table 1. As the table shows, Resnet101 achieves the highest average accuracy 75.25%. Therefore, we use the Resnet101 as the baseline model. In the following sections, we will use resnet101 as the backbone to improve it's performance.

4. Improvements

4.1. CutMix

CutMix [24] is a data augmentation technique that addresses the issue of information loss and inefficiency present in regional dropout strategies. Instead of removing pixels and filling them with black or grey pixels or Gaussian noise, it replace the removed regions with a patch from another image, while the ground truth labels are mixed proportionally to the number of pixels of combined images. It's implemented via the following formulas:

$$\begin{aligned} \tilde{x} &= M \odot x_A + (1 - M) \odot x_B \\ \tilde{y} &= \lambda y_A + (1 - \lambda) y_B, \end{aligned} \tag{1}$$

where M is the binary mask which indicates the cutout and the fill-in regions from the two randomly drawn images and λ (in [0, 1]) is drawn from a Beta(α , α) distribution.

The coordinates of bounding boxes are: $B = (r_x, r_y, r_w, r_h)$,which indicates the cutout and fill-in regions in case of the images. The bounding box sampling is represented by:

$$\begin{aligned} r_x &\sim Unif(0, W), \quad r_w = W\sqrt{1-\lambda}, \\ r_y &\sim Unif(0, H), \quad r_h = H\sqrt{1-\lambda}, \end{aligned} \tag{2}$$

where r_x, r_y are randomly drawn from a uniform distribution with upper bound. The experiment results is shown in Table 2. When CutMix is adopted, the accuracy of model could improve for around 0.5% compared to original 75.25%.

4.2. Triplet Loss and Label Smoothing

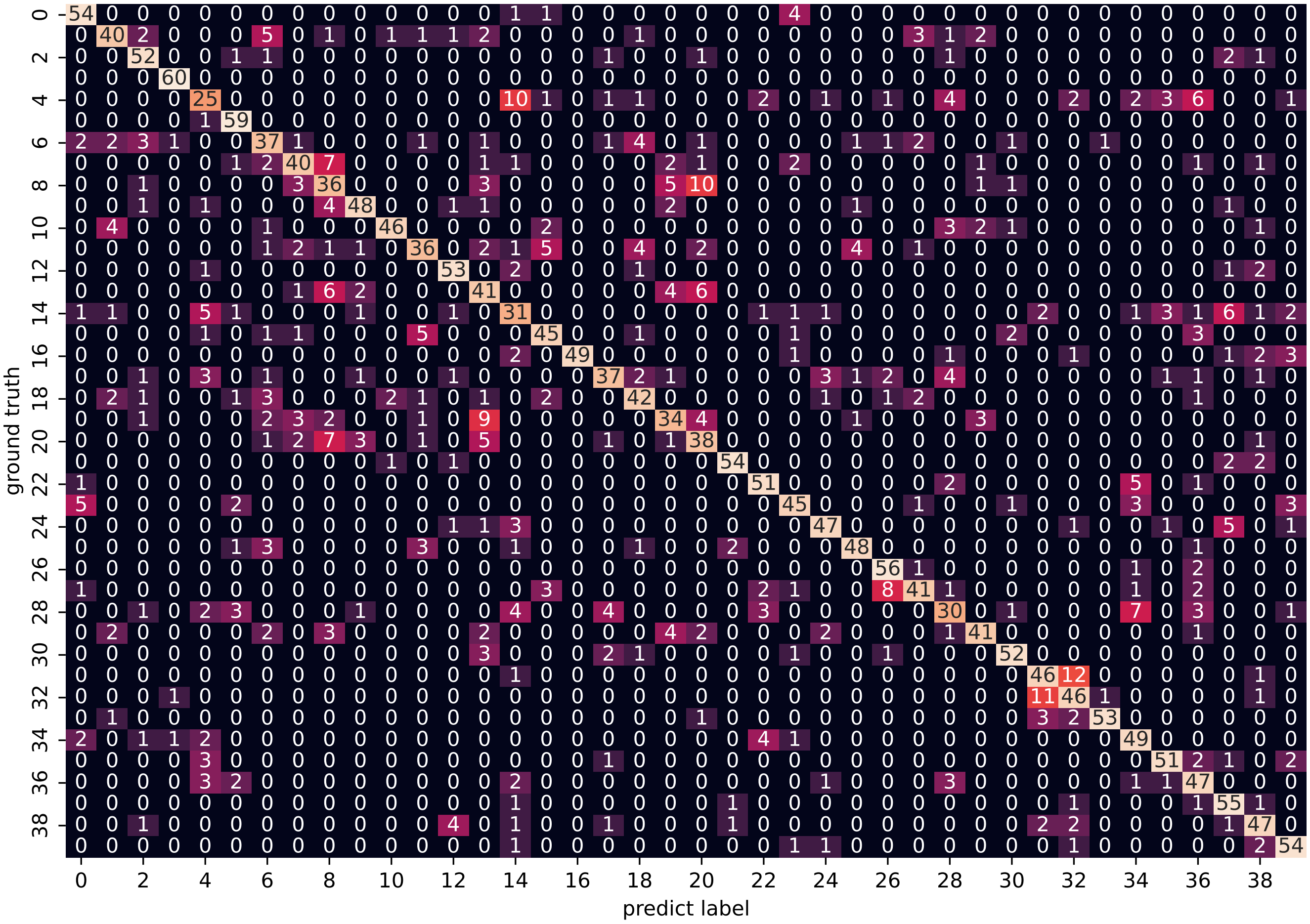
Triplet Loss [15] is a widely used loss in Contrastive Learning. The goal of triplet loss is to make features of images with the same label closer and features of images with different labels more distant. Specifically, the dissimilarity between images with different labels should outperforms images with same label by a margin α. With triplet loss, the model can extract features that are easier to classify. The triplet loss is defined as follows:

$$L_{triplet}(A, P, N) = \max(\text{dissim}(A, P) - \text{dissim}(A, N) + \alpha, 0)$$

where dissim is function that compute the dissimilarity between two features, A refers to anchor image and P, N represents positive image(same label as anchor) and negative image(different label as anchor), respectively. At each training step, we sampled a triplet (A, P, N) from a batch and compute the triplet loss. We do hard sample mining by selecting (A, P, N) that $\text{sim}(A, P)$ is closest to $\text{sim}(A, N)$, which are the most difficult samples to distinguish.

The triplet loss is used with the cross entropy loss(CE loss), so the total loss is defined as:

$$L = (1 - \lambda) L_{CE} + \lambda L_{triplet},$$



	0		2		4		6		8		10		12		14		16		18		20		22		24		26		28		30		32		34		36		38	
0	54	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	40	2	0	0	0	5	0	1	0	1	1	1	2	0	0	0	0	1	0	0	0	0	0	0	0	0	3	1	2	0	0	0	0	0	0	0	0	0	0
2	0	0	52	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	1	0
	0	0	0	60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	25	0	0	0	0	0	0	0	0	0	10	1	0	1	1	0	0	0	2	0	1	0	1	0	4	0	0	0	2	0	2	3	6	0	0	1
	0	0	0	0	1	59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	2	3	1	0	0	37	1	0	0	0	1	0	1	0	0	0	1	4	0	1	0	0	0	0	1	1	2	0	0	1	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	1	2	40	7	0	0	0	0	1	1	0	0	0	0	2	1	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0
8	0	0	1	0	0	0	0	3	36	0	0	0	0	3	0	0	0	0	0	5	10	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
	0	0	1	0	1	0	0	0	4	48	0	0	1	1	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10	0	4	0	0	0	0	1	0	0	0	46	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	3	2	1	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	1	2	1	1	0	36	0	2	1	5	0	0	4	0	2	0	0	0	0	4	0	1	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	1	0	0	0	0	0	0	0	53	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0
	0	0	0	0	0	0	0	1	6	2	0	0	0	41	0	0	0	0	0	4	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	1	0	0	5	1	0	0	0	1	0	0	1	0	31	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	2	0	0	1	3	1	6	1	2
	0	0	0	0	1	0	1	1	0	0	0	5	0	0	0	45	0	0	1	0	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	3	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	49	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	2	3
	0	0	1	0	3	0	1	0	0	1	0	0	1	0	0	0	0	37	2	1	0	0	0	0	3	1	2	0	4	0	0	0	0	0	0	1	1	0	1	0
18	0	2	1	0	0	1	3	0	0	0	2	1	0	1	0	2	0	0	42	0	0	0	0	0	1	0	1	2	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	1	0	0	0	2	3	2	0	0	1	0	9	0	0	0	0	0	34	4	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	1	2	7	3	0	1	0	5	0	0	0	1	0	1	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0
22	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51	0	0	0	0	0	2	0	0	0	0	0	5	0	1	0	0	0
	5	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	45	0	0	0	1	0	0	1	0	0	0	3	0	0	0	0	3
24	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	0	0	0	0	0	0	0	0	0	47	0	0	0	0	0	0	0	1	0	0	1	0	5	0	1
	0	0	0	0	0	1	3	0	0	0	0	3	0	0	1	0	0	0	1	0	0	2	0	0	0	48	0	0	0	0	0	0	0	0	0	0	1	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56	1	0	0	0	0	0	0	1	0	2	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	2	1	0	0	8	41	1	0	0	0	0	0	1	0	2	0	0	0
28	0	0	1	0	2	3	0	0	0	1	0	0	0	0	4	0	0	4	0	0	0	0	3	0	0	0	0	0	30	0	1	0	0	0	7	0	3	0	0	1
	0	2	0	0	0	0	2	0	3	0	0	0	0	2	0	0	0	0	0	4	2	0	0	0	2	0	0	0	1	41	0	0	0	0	0	0	1	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	2	1	0	0	0	0	1	0	0	1	0	0	0	52	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	46	12	0	0	0	0	0	1	0
32	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	46	1	0	0	0	0	1	0
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	3	2	53	0	0	0	0	0	0
34	2	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	0	0	0	0	0	0	0	0	0	0	49	0	0	0	0	0
	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51	2	1	0	2
36	0	0	0	0	3	2	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	1	1	47	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	55	1	0
38	0	0	1	0	0	0	0	0	0	0	0	0	4	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	2	2	0	0	0	0	1	47	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	2	54

Figure 1. Confusion matrix. Each row of the confusion matrix represents the ground truth label and each column of the confusion matrix represents the label predicted by classifier

where λ is a hyperparameter to balance the two losses.

Apart from Triplet loss, we also use label smoothing [13] [18] to improve the performance of the classifier. Namely, we change the one-hot label to

$$P_i = \begin{cases} 1-\epsilon & i = y \\ \frac{\epsilon}{K-1} & i \neq y \end{cases}$$

P_i is the smoothed label and K is the class number. Label smoothing is a regularization method that can avoid the model be too confident and improve model's performance.

The experiment results is shown in Table 2. With label smoothing and triplet loss, the average accuracy is furtherly improved to 76.15%.

4.3. Analysis Confusion Matrix

Using resnet101 as the baseline, we achieved the accuracy of 75.25%. To further improve the accuracy, we tried to analysis the confusion matrix shown in Figure 1. Each row of the confusion matrix represents the ground truth label and each column of the confusion matrix represents the label predicted by classifier. With the confusion matrix [21], we can analysis which classes are tend to be misclassified by the classifier and improve the performance targetedly. For example, it is obevious that class 31 and 32 are confused classes beacuse the model misclassified 12 samples of class 31 into class 32 and 11 samples of 32 into 31, Which indicate that the model can not distinguish these two classes

Model	CutMix	LS	TL	Accuracy
ResNet101	✓			75.67 ± 0.885
ResNet101	✓	✓		75.75 ± 2.255
ResNet101	✓	✓	✓	76.17 ± 0.881

Table 2. Comparison of performance with different methods on Skin40. LS represents labels smoothing. TL represents Triplet Loss.

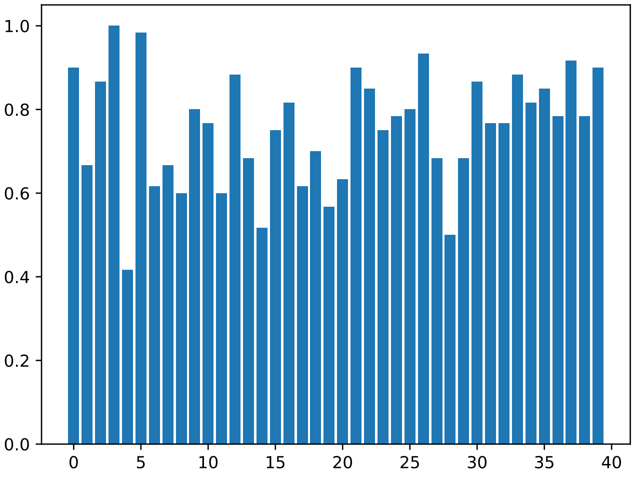


Figure 2. Accuracy of each class.

well.

To solve this problem, I have tried use fine-grained classification method like bilinear pooling [12]. But it does not improve the classification accuracy. We will investigate more methods to solve this problem.

Based on the confusion matrix, we can compute the accuracy of each class which is shown in Figure 2. From this figure we can observe that each class's accuracy differs a lot. Some class are perfectly classified while some class's accuracy is only 40%. Our goal is to improve the model's performance on those classes with low accuracy. As a preliminary improvement, we tried to apply different weight on each class. Specifically, we use reciprocal of accuracy on each class as the weight, which can ensure class with low accuracy will have a higher weight and therefore have a larger loss. For those classes, the classifier will be punished more heavily due to the higher loss. With these simple method, the classification accuracy is improved from 75.25% to 76.88%.

Due to the improvement brings by weighted loss, we also tried another method to compute the loss. We use weights on samples rather than classes. Considering that some categories are easily misclassified into other categories, we set higher weight for those misclassified samples. Specifically, if a sample of label A is always misclassified into label B, we will give more heavily punishment on this sample. In order to achieve this, we use the value of confusion matrix as the weight because the value of confusion matrix reflects how much a category is likely to misclassified into another category. However, the diagonal elements in confusion matrix are very large and are correctly classified samples number, hence we remove the diagonal elements. There are also a lot of zero values in the confusion matrix, we tend to use following matrix as the weight(the confusion matrix is denoted as C):

$$W[A][B] = \ln(C - \text{diag}(C) + I)$$

where $\text{diag}(C)$ represents the diagonal matrix with the same diagonal values in C. The $\ln$ function is used to smooth the values in the matrix. With these method, the average accuracy is 76.50%. Although the accuracy lower than above simple weighted loss, the accuracy of easily misclassified classes are improved as shown in Figure 3. Compare to the baseline ResNet101, classes with low accuracy is improved a lot though classes with high accuracy is dropped. We have tried to combine the advantages of these two models to improve the performance but failed, which is introduced in the appendix.

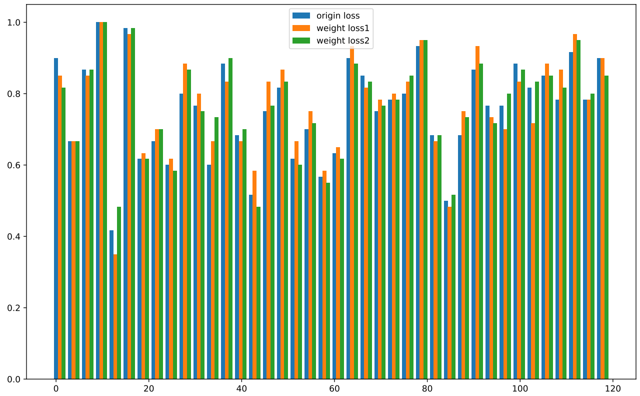


Figure 3. Comparison of accuracy of each class with different weighting method. Use the confusion matrix as the weight can improve the accuracy of easily misclassified classes.

4.4. Vision transformer

Apart from convolutional neural network, we also tried vision transformer(ViT) as the backbone network. ViT [5] consists of several stacked multi-head self-attention module(MSA) and MLP. Besides, layer normalization [] is used and the activation fuction is GELU. Besides, an additional class token is used to represent the whole image's token. The whole network can be formulated as follows:

$$\begin{aligned} z_0 &= [x_{class}; x_1; x_2; \cdots; x_N] + E_{pos} \\ z'_l &= \text{MSA}(\text{LN}(z_{l-1})) + z_{l-1} \\ z_l &= \text{MLP}(\text{LN}(z'_l)) + z'_l \\ y &= \text{LN}(z^0_L) \end{aligned}$$

where x_{class} is the class token, x_i is a patch of the input image after linear projection and E_{pos} is the positional encoding. The classification accuracy of vision transformer on the Skin40 dataset is shown in table3, from which we can observe that ViT provides high accuracy and outperforms ResNet101 by a large margin. Besides, the result is

Fold	Accuracy
1	78.54
2	78.75
3	78.75
4	78.75
5	78.54
average	78.67 ± 0.10

Table 3. Each fold's classification accuracy(%) of Vision Transformer on Skin40.

stable and has a low variance on each fold. This is an interesting result because usually we believe only on very large datasets like ImageNet can ViT have a better performance than CNN. But this results indicate that with pretrained weights, transformer can also be successfully adopted on small datasets.

5. Conclusion

In this project, we select ResNet101 as the baseline method and achieved preliminary results. Then, we tried the data augmentation method such as some basic transforms accessible in torchvision, CutMix and MoEx to push the model extract better features from images. These approaches bring performance improvements. We also attempted to use SinGAN as an augmentation method to generate more data, but the generated images does not differ from original data. Besides, we improved the models performance by using triplet loss and label smoothing. We also improve the neural model by assigning different weight on each class according to the recall of each class. Additionally, we analysed the confusion matrix to find confused classes and easily misclassified classes and obtained some improvements. Finally we trained a vision transformer which obtained a higher accuracy.

References

[1] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018. 6

[2] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[4] Tadayoshi Fushiki. Estimation of prediction error by using k-fold cross-validation. *Statistics and Computing*, 21(2):137–146, 2011. 2

[5] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 2022. 4

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 1

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. pages 770–778, 2016. 1

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. 7

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[10] Boyi Li, Felix Wu, Ser-Nam Lim, Serge Belongie, and Kilian Q Weinberger. On feature normalization and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12383–12392, 2021. 7

[11] Boyi Li, Felix Wu, Kilian Q Weinberger, and Serge Belongie. Positional normalization. *Advances in Neural Information Processing Systems*, 32, 2019. 7

[12] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015. 4

[13] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019. 3

[14] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 6

[15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015. 2

[16] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. pages 4570–4580, 2019. 6

[17] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019. 2

[18] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. 3

[19] C-H Teh and Roland T Chin. On image analysis by the methods of moments. *IEEE Transactions on pattern analysis and machine intelligence*, 10(4):496–513, 1988. 7

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and

Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 1

[21] Sofia Visa, Brian Ramsay, Anca L Ralescu, and Esther Van Der Knaap. Confusion matrix-based feature selection. *MAICS*, 710(1):120–127, 2011. 3

[22] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133, 2019. 2

[23] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018. 1

[24] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, pages 6023–6032, 2019. 2

[25] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. 2

[26] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 2

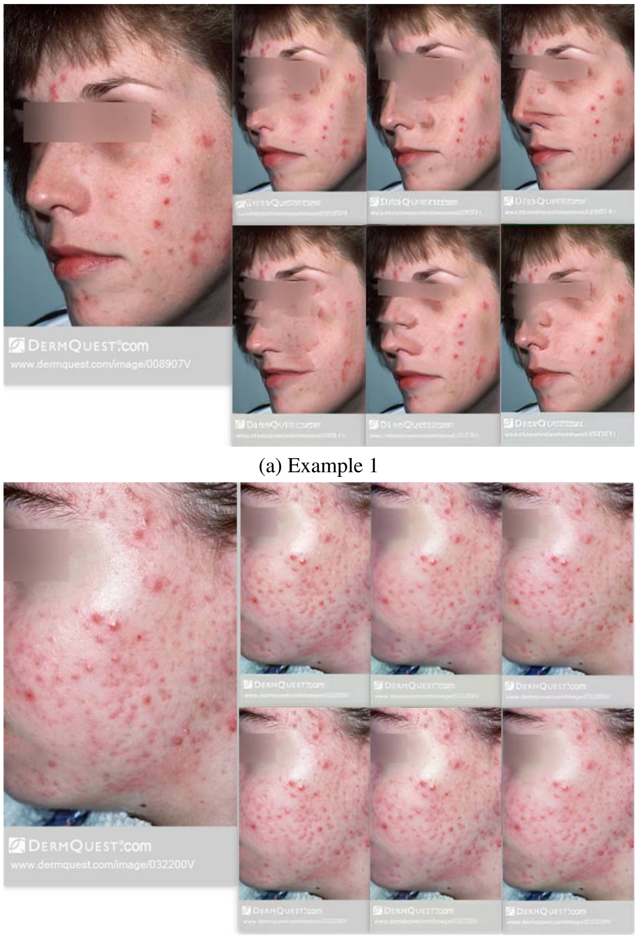
A. Appendix

Here are other methods that we have tried but didn't work well on the neural network.

A.1. SinGAN

SinGAN [16] is short for single generative adversarial network, which is an unconditional generative model that can be learned from a single image. Its main idead is learning the patch distribution at a different scale of the image with the help of a pytramid of fully convolution GANs [1]. This structure allows generating new samples of while maintaining both the global structure and the fine textures of the training image.

However, when adopted in Skin40 dataset [14], the performance of SinGAN doesn't meet our expectations. As Figure 4 shows, the importance part of the image – the pathological features of skin, don't manifest obvious changes. Actually, the significant features look quite similar to the original image in most of the generative images, which means the generative images will be less helpful or almost no help to the improvement of the skin disease classification model.


DERMQUEST.com
www.dermquest.com/image/008907V
DERMQUEST.com
www.dermquest.com/image/032200V

(a) Example 1

(b) Example 2

Figure 4. Experiment results of SinGAN. The image on the far left in each example is the original image, while the others are generated by SinGAN.

After searching for the relative documents, the reason that SinGAN could not perform well in the generation of Skin40 data set can be summarized as follows. Firstly, for the lack of directive information, the SinGAN couldn't detect the global structure and the fine textures of the images. Since the pathological features and skin occupy quite a lot of region of the original image, the SinGAN mistakenly recognizes them as the global structure of image, and maintains them as much as possible, which leads to little changes of the pathological features in the generated images. Meanwhile, the true global structures, such as the faces, arms, legs and so forth, are recognized as the texture since they occupy a relatively small area. Consequently, SinGAN keeps the main pathological features unchanged and tries to change the true global structures, which can be found in Example 1 of Figure 4, in which the generated images' changes concentrate upon the nose, brow and so on, while the pathological features – the red dots on the skin almost stay the same.

A.2. distillation and ensemble

As illustrated in section 4.3, we obtained several different models which have different accuracies on each class. We tried two methods to combine the advantages of these models. The first method is using these models as teacher networks to train a student network. This is similar to multiteacher network distillation [8]. During each step, we select a model from these models according to the accuracy of a certain label. Specifically, we use the model t with the highest accuracy on class c when the input sample is class c:

$$L = \mathrm{CE}(y, \hat{y}) + \mathrm{CE}(\hat{y}, \tilde{y})$$

where y is the ground truth label, $\hat{y}$ is the output of the student network and $\tilde{y}$ is the output of the selected model. However, these method does not bring improvement on performance compared to these teacher networks. The second method we tried is to ensemble these models. For this image classification task, we use two strategies to ensemble these models:vote and take the prediciton with highest softmax output. Both these two strategies lead to slight performance drop. This is reasonable because bagging is a method used to reduce variance and can not ensure to reduce bias.

A.3. Moment Exchange

Moment Exchange [10], which is call MoEx for short, is an implicit data augmentation method. The main idea of MoEx is augmenting the data by utilize the moment information. Moments [19] are attributes of a data instance, which describe the distribution of the features. Consequently, it can roughly capture the shape and style information of the data instance. When using MoEx, the neural model will combine the normalized features of one instance with the feature moments of another to generate a new data.

The structure of MoEx can be summarized as follows. For two randomly chosen data x_A and x_B, when they are send into the neural network, they will generate a series of hidden features $h^0, h^1, ..., h^n$. For each hidden feature h^l, we denote the function by F, which could calculate the normalized features $\hat{h^l}$, the first moment μ^l, and the second moment σ^l:

$$\begin{aligned}(\hat{h^l}, \mu^l, \sigma^l) &= F(h^l) \\ h^l &= F^{-1}(\hat{h^l}, \mu^l, \sigma^l)\end{aligned} \tag{3}$$

Actually, there different method to implement the function F, for example PONO [11].

After implementing the function F, we can use it to generate the moments of data A and B, $\hat{h_A}, \mu_A, \sigma_A, \hat{h_B}, \mu_B, \sigma_B$. Then we can exchange the moments information with the following formulas to generate the new data.

$$h_A^{(B)} = F^{-1}(\hat{h_A}, \mu_B, \sigma_B)$$

In the case of PONO, the formula will turn to be $h_A^{(B)} = \sigma_B \frac{h_A - \mu_A}{\sigma_A} + \mu_B$.

When adding MoEx into the neural network, we don't found obvious improvement. After comparing the accuracy before and after using MoEx, we reckon that the probable reasons can be listed as follows. As we mention above, the information that Moments describe is mainly the shape and style information of the data. But for the Skin40 data set, the style of the image is the people skin of different area, which don't have large differences and isn't quite helpful for the improvement of the neural network. Additionally, we find that the diseases that can be distinguish by shape have already reach a high accuracy before using MoEx and the shape information isn't a distinguishable feature for the rest diseases that is in a low accuracy. Consequently, the accuracy hasn't obvious improvement when MoEx is adopted.