CS172 assignment1 ResNet

Shouchen Zhou
Student ID: 2021533042
zhoushch@shanghaitech.edu.cn

Abstract

In this assignment, after reading the ResNet paper, we learned the structure and features of ResNet. The origin task of ResNet: the image classification on CIFAR10.

We tested the performance difference between the model trained from scratch and finetuned from a pretrained weights on ImageNet.

Then we used the ResNet to train and test on the Shanghaitech Crowd Counting Dataset, the see the processes during the training.

And the following tasks were finished.

- *Task1: Use ResNet for image classification on CIFAR10*
- *Task2: Use ResNet for crowd counting task on Shanghaitech crowd Counting Dataset*

1. paper reading

The work [1] proposed a way to train a network when the network is deep. As the era of deep learning, people would like to increase the depth of the network, but this usually cause a degradation. By the experiments in the work, we can see that the loss and error of the deeper network is always larger than the loss and error of shallow network, which is also defined as the degradation problem. Then the ResNet are design geniusly.

To train the network with deeper structure, a residual learning was designed with shortcut by identity mapping.

The relation between the input $\mathbf{x}$ *and output* $\mathbf{y}$ *is that*

$$\mathbf{y} = \mathcal{F}\left(\mathbf{x}, \{W_i\}\right) + \mathbf{x}$$

The shape of $\mathcal{F}\left(\mathbf{x}, \{W_i\}\right)$ *should be same with* $\mathbf{x}$.

This might to difficult to maintain as the intermediate block increase within the forword process.

So another shortcut was proposed

$$\mathbf{y} = \mathcal{F}\left(\mathbf{x}, \{W_i\}\right) + W_s\mathbf{x}$$

Where W' *is the mapping the maintain the same shape.*

With the short cuttings, the degradation are solved, so the deeper network could be better performanced.

2. Task1: Use ResNet for image classification on CIFAR10

We could inplement the ResNet with torch structure, and train the network with different parameters.

The model are trained in two ways:

1. from scratch

2. using pretrained weights from ImageNet

2.1. basic settings

The hyparameters are used as followed.

- The ResNet with depth $18, 34, 50, 101, 152$ *are implemented to compare the effects.*

- The last layer of the network are set to be linear mapping to 10 nerual, this is because the CIFAR-10 are used for classification on 10 *different classes.*

- the criterion is set to be the CrossEntropyLoss

- the optimizer is set to be SGD optimizer, the momentum 0.9

- the batch size is set to be 128*, and the epoch number is set to be* 200

As for the learning rate of the optimizer, the learning rate are initially set to be 0.1*, and after* 32000 *iterations' training, reduce the learning rate* 10%*, i.e.* 0.01*. And after* 48000 *iterations' training, reduce the learning rate* 10% *again, i.e.* 0.01*.*

The ResNet structure are import from torch, and the last layer is set to be a $10-FC$ *layer, this is because the CIFAR-10 dataset has* 10 *classes. So the last layer is the one-hot to do the classification.*

And other settings are by default.

2.2. way to run the code

In the submitted package, including two jupyter notebooks named 'task1_scratch' and 'task1_finetune'. Just run all the module in it could generate all the results.

2.3. trained from scratch

After constructing the net, we do not load the pretrained weights from torch package. Then just do the training.

2.4. trained using pretrained weights from ImageNet

Similarly to training from scratch, but the difference is that the torch package has already trained on the ImageNet, so we could just load in the pretained weight, and then training on the pretrained model. i.e. finetuning on the CIFAR-10.

2.5. comparison the difference between the two training modes

We could train from scratch and finetune the network with different hyper-parameters. In this task, I am more likely to test the behavior of ResNet with different depth.

So ResNet18, ResNet34, ResNet50, ResNet101, ResNet152 are trained and evoluated.

The CIFAR-10 data are divided into two groups with ratio $9:1$, the bigger group are used as training data, and the smaller group are used as validation data. In this method, we could avoid the overfitting that makes the test loss and test accuracy looks much better.

The following forms are the testing result of training from scratch and finetune on pretrained on ImageNet with different network depth. The test acc in the form represents test accuracy.

The followuing for is trained from scratch:

ResNet depth	*18*	*34*	*50*	*101*	*152*
test loss $\downarrow$	*0.9872*	*0.9670*	*0.8518*	*0.8283*	***0.8425***
test acc $\uparrow$	*0.6608*	*0.6626*	*0.7842*	*0.8410*	***0.8460***

The followuing for is finetuned from pretrained on ImageNet:

ResNet depth	18	34	50	101	152
test loss $\downarrow$	1.0972	1.1825	***0.8175***	1.4900	1.0870
test acc $\uparrow$	0.7448	0.8110	***0.8458***	0.7374	0.8128

And the details of training loss, training accuracy, testing loss, testing accuracy as the epoch grows are shown in Figure 1.

The figure 2. is the testing accuracy with different network depth that trained from scratch.

The figure 3. is the testing loss with different network depth that trained from scratch.

2.6. discussion on results

We could discover that as it was mentioned in He.'s paper [1]. The training loss and training accuracy have obvious sharp changes. And the testing loss can also be seen as a obvious phonomenon that the testing loss decrease at first, but increase a little later(In Figure 3). [1] pointed that this is not because of the overfitting of the network, and we could also see this from Figure 2 that the testing accuracy tends to increase. Also, the paper [1] mentioned that the ResNet's biggest advantage is that it will not lead to a degradiant as the depth of the network increasing. From Figure 1. we could discover this, with a long terms training, the traing loss, traing accuracy, testing accuracy all has a sudden change. And the deeper network does not behave worse, the ResNet does not lead to degradiant, so we can see in Figure 1. the deeper network has a better behavior in the training from scratch.

However, in the training from pretrained on ImageNet, the finetuning process initially has a better testing loss and testing accuracy, but as the training process going, the behavior graduately become worse than the training from scratch.

And a noticable phenomenon is that the best behavior on the validation dataset is ResNet50, instead of the deeper one. I am quite curious about this phenomenon, due to the time limit, I would like to pay more effort in discovering this later.

3. Task2: Use ResNet for crowd counting task on Shanghaitech Crowd Counting Dataset

The ResNet are modified from doing classification task into crowd counting task.

We want to use the network to learn the distribute of where the human heads are. But the δ function the undifferentiable, which is difficult for the network training, so we can transfer the groundtruth coordinates into a heatmap with Gaussion kernel.

The groundtruth of each image are the coordinate of each human head on the image coordinate system, so with the given coordinates, it is easy for us to generate a heatmap of gaussian. However, some of the annotated points are out of boundary(actually, those out of x boundary are in the part_A, those out of y boundary are in the part_B). Ignore those negative coodinate, we will generate a heatmap.

And for considering the effect, we have evaluation metric:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|z_i - \hat{z}_i| \tag{1}$$

and

$$MSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i - \hat{z}_i)^2} \tag{2}$$

where N is the number of test images, z_i is the actual number of people in the ith image, and $\hat{z}_i$ is the estimated number of people in the ith image. Roughly speaking, MAE indicates the accuracy of the estimates, and MSE indicates the robustness of the estimates. The settings of evaluation metric are from [2].

In the Figure 4. from left to right are the origin image, gaussian heatmap generate from the ground truth, and the heatmap generated by the trained network.

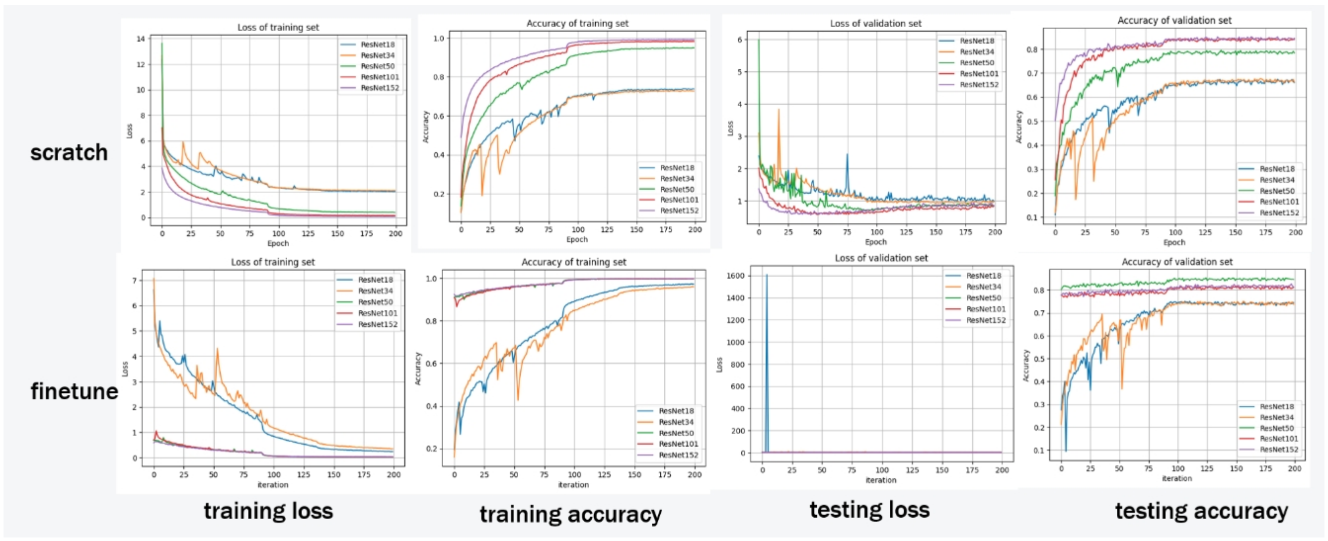

Figure 1. training and testing loss and accuracy line chart of training from scratch and finetuning

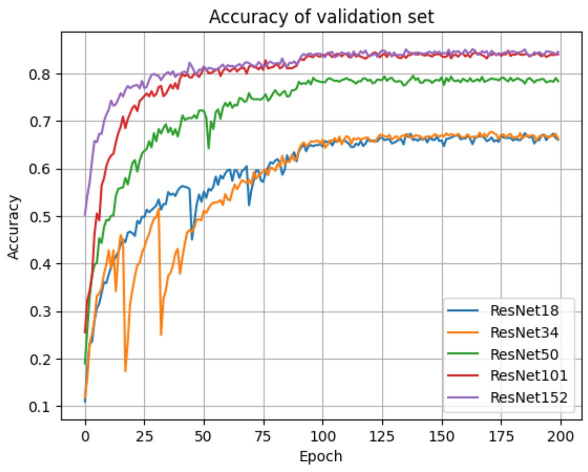

Figure 2. testing accuracy trained from scratch

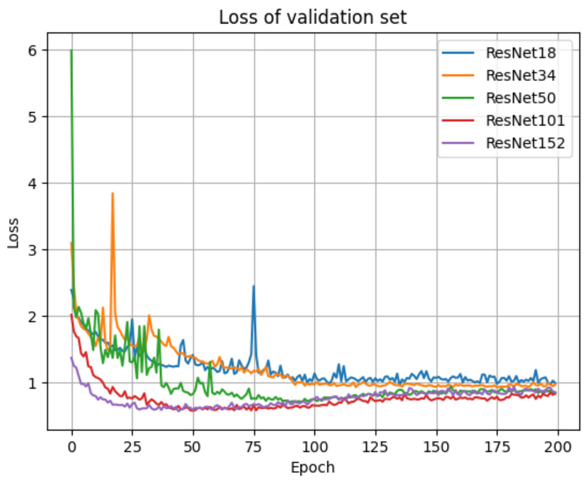

Figure 3. testing loss trained from scratch

3.1. basic settings

The hyparameters are used as followed.

we could construct a network, similarly to task1, with ResNet. But the last layor is set to a 512×512*-FC layer. This is because all images are cropped into* 512×512*, and so does the heatmap. We want to net to learn from the image and its heatmap.*

- The last layer of the network are set to be linear mapping to 512×512 *nerual*
- the criterion is set to be the MSELoss
- the optimizer is set to be the Adam optimizer, the learning rate is defaultly set to be 0.01*, and it will be tested to find a better one in the later experiments.*
- the batch size is set to be 1, and the epoch number is set to be 50.

And other settings are by default.

3.2. way to run the code

We can just "python crowd_counting .py" to run the code.

data class, batch size, epoch number, these hyparameters can be modified with command line parameter.

Without given command line parameter, other settings are by default.

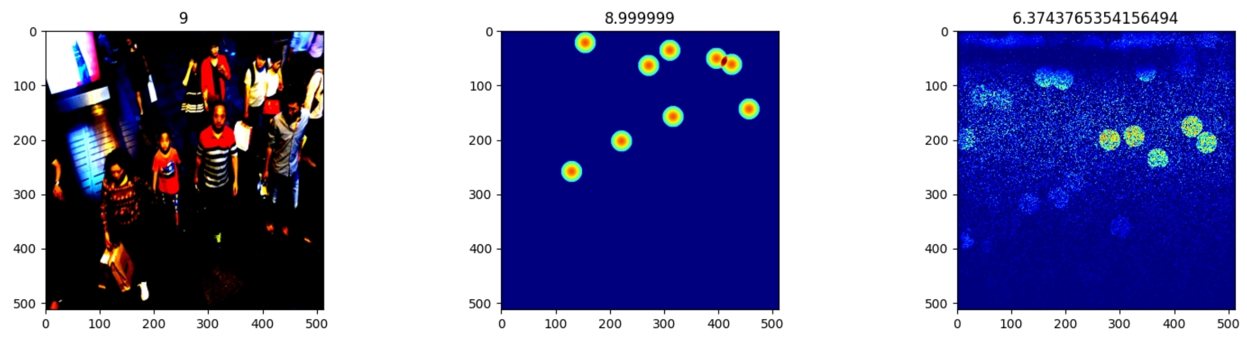

Figure 4. From left to right are : origin image, gaussian heatmap of the ground truth, the learned prediction heatmap

3.3. results

From Figure 4. we could see the

although the position of the head's point is not quite corrct, but the net somehow learned the gaussian density, we could clearly see the circle which is quite conforming to the characteristics of gaussian: circle, and the centers of the circle are the brightest, the value decearses from inside out.

We could compare the ResNet with different hyper-parameters, to find the sitable settings for the crowd counding task with ResNet.

We can test with the fixed ResNet depth, which learning rate could get a better performance. In the following forms, the lr represents learning rate.

ResNet18	Part_A		Part_B	
lr	MAE ↓	MSE↓	MAE↓	MSE↓
0.1	401.3516	531.8209	123.8861	156.2428
0.01	**321.9121**	**476.1688**	**122.4778**	**155.0796**
0.001	324.5824	503.7461	277.9780	433.7102

From the tests above, we could discover that when epoch=50*, we can set learning rate* = 0.01 *to chase a better performance. With the fixed learning rate, we test the differnet net depth.*

lr=0.01	Part_A		Part_B	
net depth	MAE↓	MSE↓	MAE↓	MSE↓
ResNet18	**321.9121**	**476.1688**	122.4778	**155.0796**
ResNet34	360.4066	498.0374	125.0759	**154.7871**
ResNet50	386.8956	524.2342	**123.1994**	155.6714

3.4. discussion on results

From the testing result, we could discover that when the epoch=50*, learning rate is set to be* 0.01 *could chase a better performance. And after a fixed learning rate, we could see that ResNet18 has a btter performance.*

From our intuition, maybe the deeper net could have a better preformance, but the result does not. I thought this may have two reasons, maybe the deeper network truly has a worse behavior, or the epoch number 50 *is too small for this task.*

To varify my thoughts, I tried to train the network with more time. On ResNet18 with 500 epoch, the MAE and MSE improves a lot comparing with the epoch=50.

So the network is highly probability that is not convergence. Due to the time limitation, the experiment is not finished yet. But I would like to pay more efforts on this to check which is the probobly reason.

And the loss function is chosen for MSELoss. Actually, I initially picked the dice-correlation, because all values are range in $(0, 1)$ *in a heatmap. But the effect is not too good. And then I tried the MSELoss, due to the* 512×512 *large image, and the* $(0, 1)$ *small numbers, the MSELoss could be really small. So I multiple the groudtruth heatmap with* 512*, and then let the learned, predicted heatmap divided to* 512*. This would not to much change the heatmap, but greatly enlarge the MSELoss, and the effects seems much better.*

4. conclusion

From the implements above, I discovered the importance of the paramaters' settings could greatly influence the results. And the structure of network has a great testing , knowledge inside it.

This assignments still leave us a lot of things to discover, we should doubt every opinion. And work on it, practice to varify. Think further on every phenomenon during the experimnets.

References

[1] Kaiming He, Xiangyu Zhang abd Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CVPR, 2016. 1, 2

[2] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. CVPR, 2016. 2