Deep Learning

Assignment 3, CNN Classification

朱劭璿

B073040018

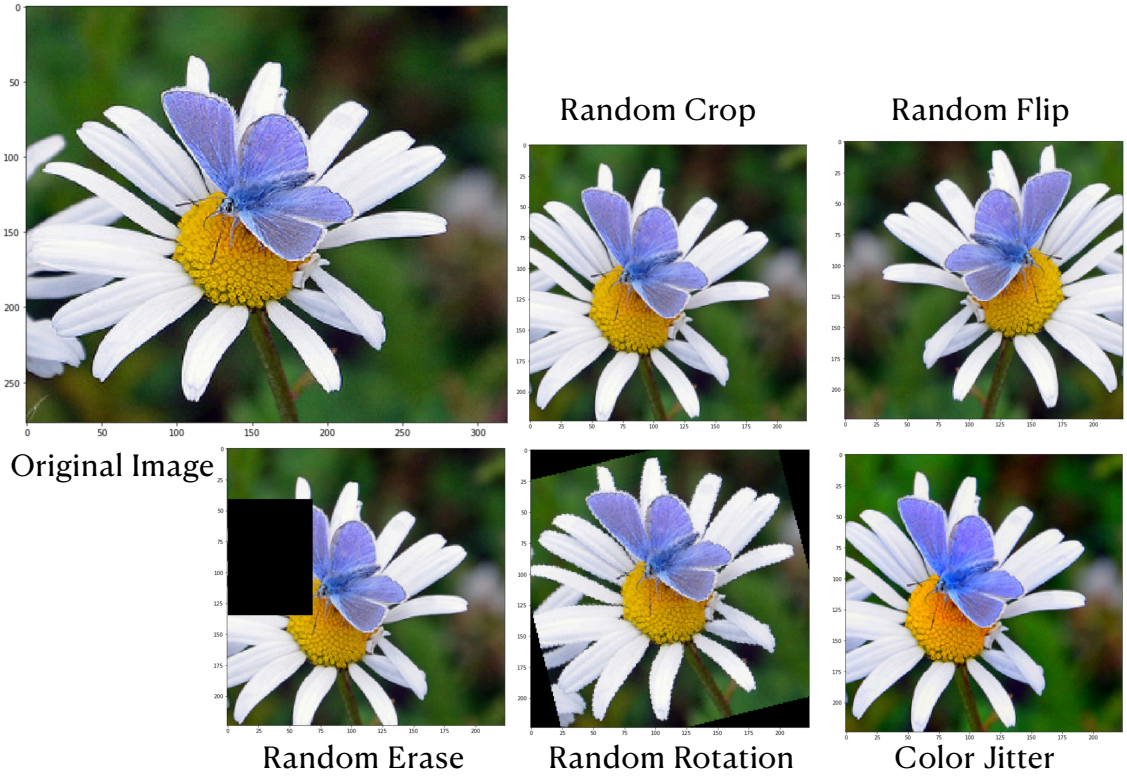
November 10, 2020

Abstract

In this report, we will learn how to train a convolution neural network for image classification from the simplest model to the state-of-art one. CNN is a simple concept itself, and it's composed of convolution layers and fully connected layers. However, building a good CNN model can be quite complicated and challenging. Not only the accuracy is important, but the cost to train a model and the time for inference are also key indicators. Especially in an era where light weight devices, SoCs and embedded systems, are so popular, the performance and the size are absolutely crucial. To build a model that take care of all of the above, the topic can be divided into data augmentation, model selection and training strategy.

Data Augmentation

I performed data augmentation on the dataset including resizing, cropping, random flipping and rotation, random erasing and color jitter.



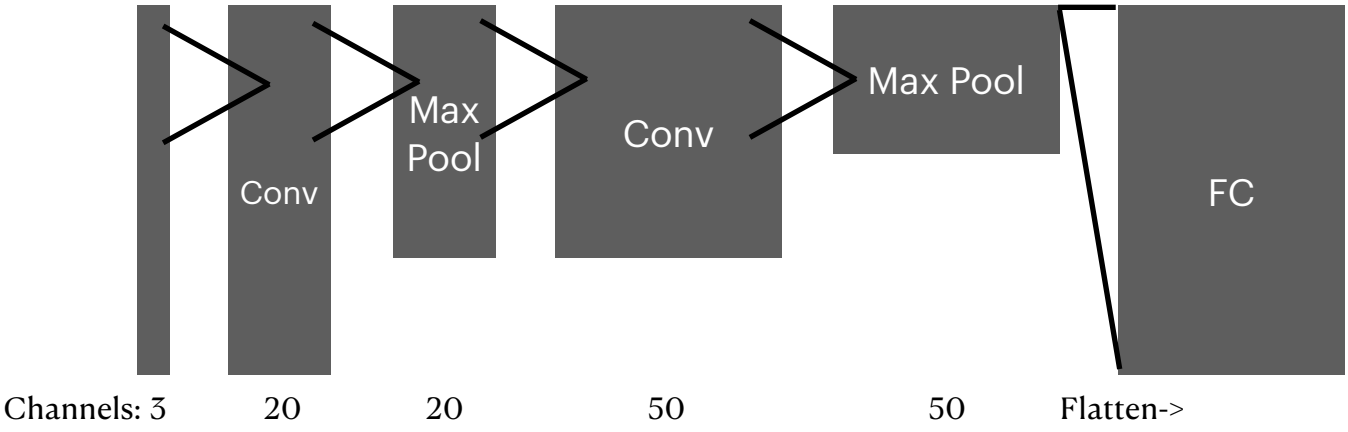
Another notable process is about concatenating the transformed data with the original data. Since with some data transformed, we'll lose the original dataset. So we can do the concatenation to keep the original dataset.

Model Selection

Since LeNet[1] had been introduced by Yann LeCun et al. in 1998, which opened the gate toward the world of image classification using CNN model, lots of researchers made remarkable progress on the CNN model. They improved the accuracy while constraining the model in a modest size. In this assignment, I've implement four models. They are LeNet, AlexNet[2] and ResNet[3].

LeNet

LeNet is a simple yet effective CNN model, it only contains two convolution layers, two pooling layers and two fully connected layers as the diagram.



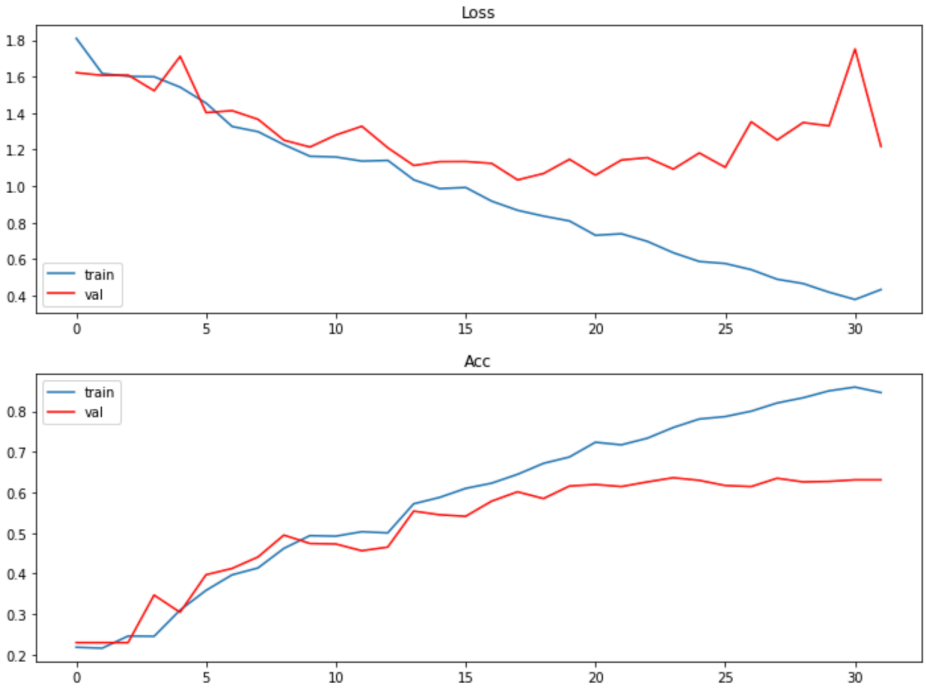
LeNet is good at one-channel simple digit classification, but when it comes to three-channel images of flower, the performance is not ideal.

AlexNet

AlexNet is a model that is more complex in comparison to LeNet. Its has bigger kernels, or say more parameters, in the convolution layers. It introduced the concept of grouping in order to train the model on two GPU cards due to the limitation on the performance.

Grouping is a concept that reduce the connection between channels and kernels. We can imagine the connection is the one in the fully connected layer. Assume we have six channels ready to perform convolution with six kernels, if all of six channels convolve together we have $6 \times 6 = 36$ connections. But if three of channels convolve with three of kernels in a group, we have $3 \times 3 = 9$ connections for each, 18 in total.

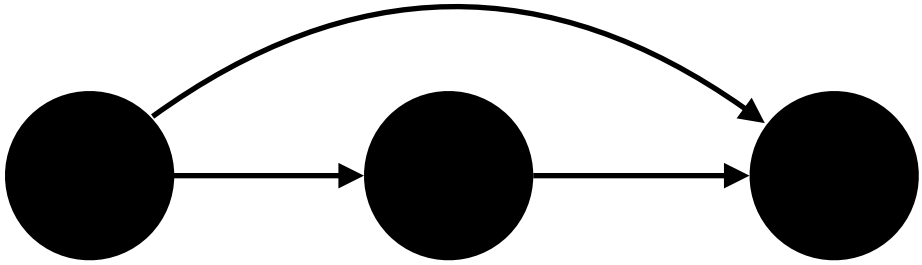
The performance of AlexNet is better than LeNet, here is the accuracy and loss.



ResNet

ResNest, which stands for residual net, was proposed after the trend of big models. We generally think that with deeper and wider neural network, we can get better performance. However, one of the problem is the deeper network cost more to train. The other one, the vanishing gradients. The vanishing gradient happen when the network is too deep that the gradient cannot backpropagate to the front layers. The learning would become very slow. To address that, ResNet came up with a solution. When encounter a

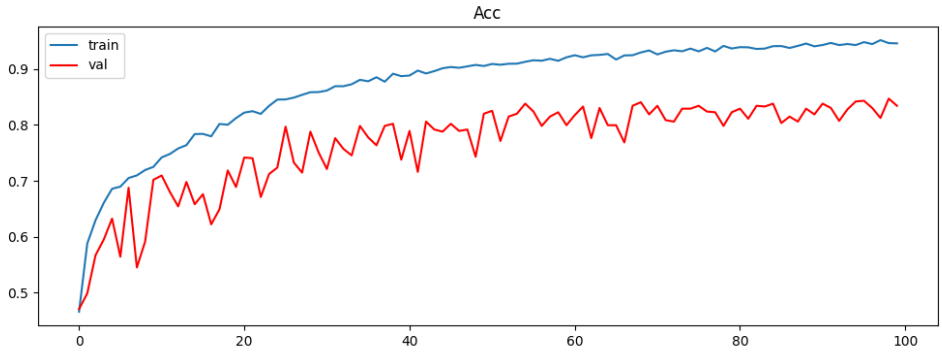
vanishing gradient, pass the former gradient to add to it. As a result, gradients are not easy to vanish through the backpropagation.

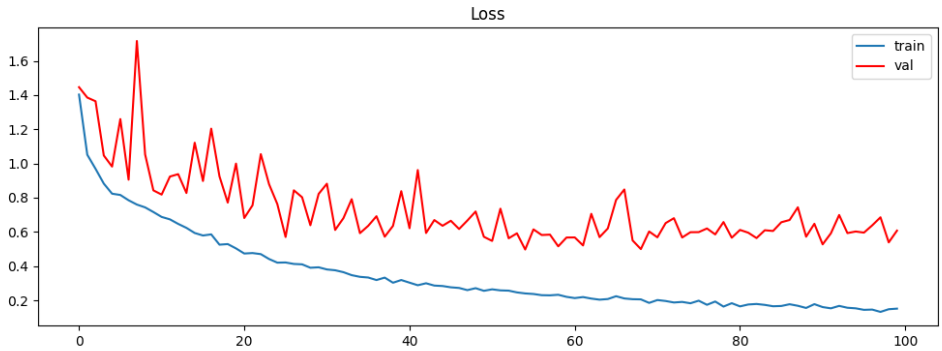


An illustration

In this small block, it also uses a technique call bottleneck, proposed in GoogLeNet[4], to reduce the number of parameters. It works like its name, think of the middle node is the neck of a bottle. It uses convolution with kernel size being one to downsample the input node, then using same technique to upsample to the output node.

The performance is phenomenal, I used this model to achieve accuracy of 87% on testing set.





Training strategy

There are some technique I used in this assignment that helped me doing well on the final score, here are some of them.

Saving parameters

It’t not easy to make the model converged. It takes a very long time. However, I need to see the changes frequently, especially in the trial phase. So saving the parameters that I’v trained really shorten the time to wait for the result.

Moving in to a stable environment

Google Colab that we use in the class is not stable. It delete some of my work sometimes, and I cannot run a big training without keep the browser open(sometimes it still fails even I do that). So moving into a stable environment is crucial for me. I used my personal computer and the cloud computing center provided by the college of management.

Reference

[1] LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; Jackel, L. D. (December 1989). "Backpropagation Applied to Handwritten Zip Code Recognition". Neural Computation. 1 (4): 541–551. doi:10.1162/neco.1989.1.4.541. ISSN 0899-7667.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. DOI:https://doi.org/10.1145/3065386

[3] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-12-10). "Deep Residual Learning for Image Recognition". arXiv:1512.03385

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going deeper with convolutions" in CVPR, pp. 1-9, 2015.