Image Recognition 2020 – Final Report

Nguyen Thuy Vy – 20M12174

I proclaim myself as “we” due to scientific habit. It’s just extremely awkward for me to write a technical report while proclaiming “I”. The work is done by myself alone.
I used “validation accuracy” instead of “test accuracy” as stated in the sample code, yet the implication is the same.

Outline of the cifar-10 (train1000)

CIFAR-10 (train1000) contains 1,000 training samples (100 samples per class) and 10,000 testing samples. The samples are of very small size (32 x 32). Therefore, it is a very challenging dataset.

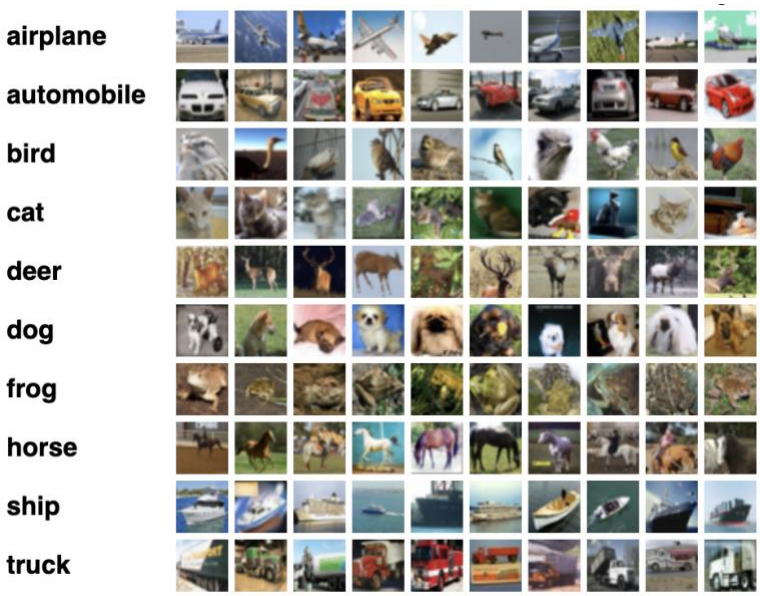


Figure 1 10 classes in cifar-10

To the best of our knowledge, the state-of-the-art result is as below:

Accuracy ▼	Cross entropy ▼	Name ▼	Code ▼	Date ▼	Paper ▼
0.5295	1.5017	Train1000 wig ensemble	https://github.com/m...	2018/11/15	[01]
0.5170	1.6784	Train1000 sample	https://github.com/m...	2018/11/09	

Figure 2 Yes, I copy this from http://www.ok.sc.e.titech.ac.jp/~mtanaka/proj/train1000/cifar10.html

We also found a github repository claims the accuracy up to 81.119%.

Your network architecture and the reason

Our network architecture is improved from the sample code and heavily inspired by VGG [2], and we do not consider any other filter sizes other than 3 x 3 based on the VGG's authors' argument: a stack of three 3 x 3 layers with non-linear activation function in between is better than a single 7 x 7 layer in terms of regularization (less number of parameters) and discriminative power (non-linearity in between).

The choice of dropout parameters is influenced by [3]. We decrease the retention probability when going from the input layer to higher levels to enforce a more generalized representation of deep features. Though, we find that having a dropout layer directly after input layer only hampers the performance, which is likely due to the low resolution of the training samples.

We use He initialization as the default initialization, Glorot, assumes that activation functions are linear [4].

We tried to replace max pooling layers with average pooling layers and convolution layers, their performance is a little bit less than max pooling layers. We think max pooling layers can detect edge and that is helpful toward image classification.

Considering that the input samples are very small, we decided to make shortcuts from shallow features to prediction layer. Information from low-level layers is less distorted and shallow features are also enforced to be semantic. This technique is very common among computer vision problems under various names with various variants. Our implementation for this technique is done without any references, it's hard to declare our method is original as we have not yet done a comprehensive survey regarding this technique.

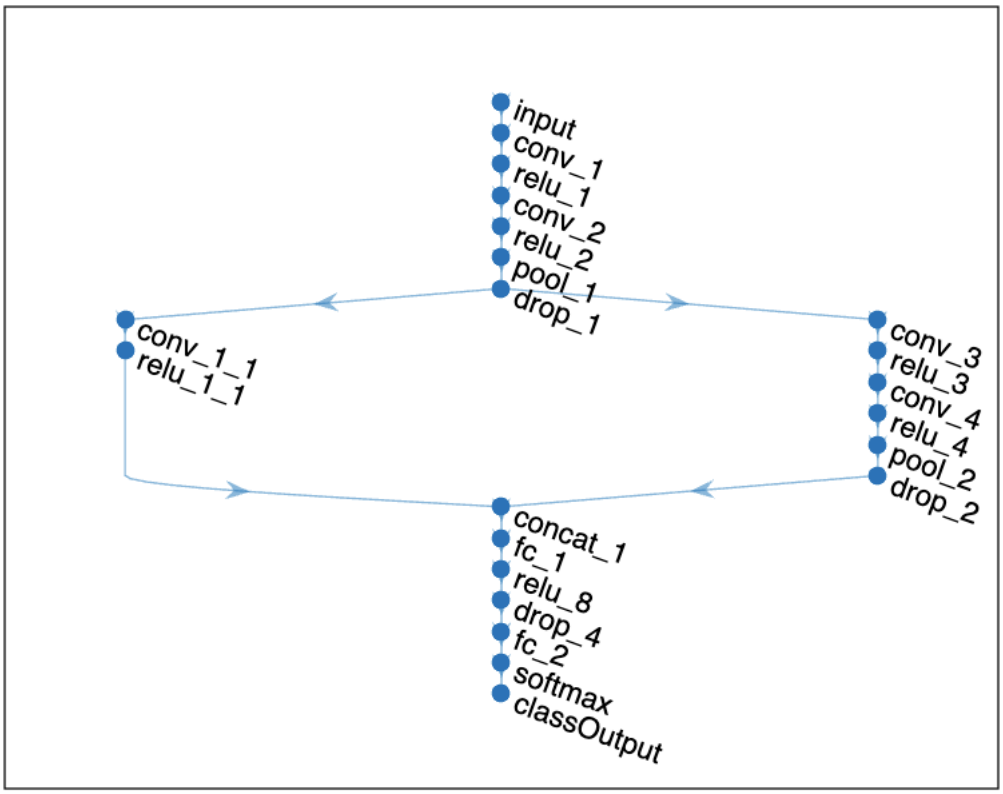


Figure 3 Our network connection

Name	Type	Dimension	Stride	Retention probability
input	imageInputLayer	32 x 32 x 3		
conv_1	Convolution2DLayer	3 x 3 x 32		
relu_1	ReLULayer			
conv_2	Convolution2DLayer	3 x 3 x 32		
relu_2	ReLULayer			
pool_1	MaxPooling2DLayer	2 x 2	2 x 2	
drop_1	DropoutLayer			0.75
conv_1_1	Convolution2DLayer	2 x 2 x 32	2 x 2	
relu_1_1	ReLULayer			
conv_3	Convolution2DLayer	3 x 3 x 64		
relu_3	ReLULayer			
conv_4	Convolution2DLayer	3 x 3 x 64		
relu_4	ReLULayer			
pool_2	MaxPooling2DLayer	2 x 2	2 x 2	
drop_2	DropoutLayer			0.75
fc_1	FullyConnectedLayer	128		
relu_8	ReLULayer			
drop_4	DropoutLayer			0.5
fc_2	FullyConnectedLayer	10		
softmax	SoftmaxLayer			

classOutput	ClassificationOutputLayer			
concat_1	DepthConcatenationLayer			

Your training parameters and the reason

- Number of training iterations: The default number of training iterations 100 cannot reflect full potential of the network. We do not remember how many iterations it needs for our network to get this performance to be honest.
- Other training parameters are set by 田中先生.

Optimizer	Adam
Shuffle	Every-epoch
MiniBatchSize	100
ValidationFrequency	10

Your key technique to improve the test accuracy

We realize that the original network architecture given by the course already overfits the data. Therefore, our first mission is to mitigate that overfitting effect using data augmentation up to the point that exceeds the overfitting capability of the original model. We then try to improve the complexity of the network architecture to deal with the complexity of the augmented data, just to find it overfits again. We start the loop of fighting overfitting (using various approaches) and increasing network architecture's complexity to further improve the test accuracy. It turns out we got tired before our network architecture can get complicated.

After running several tries, we believe the key technique to improve the test accuracy is data augmentation. 1000 training images does not contain enough intra-class variances, hence, data augmentation helps the network to learn invariant deep features by increasing variances via rotation, shear, reflection, translation, random cropping, color augmentation, synthetic noise and synthetic blur. Mathematically, it improves the VC bound [1]. Of course, a little of them may help, too much of them may be redundant (all the augmented data coming from the same image are significantly correlated) or noisy. Though, it is very hard to tell if the data is noisy or the model's architecture is not well-aligned. In our observation, the validation accuracy goes up when the training batch accuracy goes down and vice versa. We suspect that a combination of data augmentation in a given image can enhance our performance, and it did. **We expect more data augmentation will help improve our network's performance.**

We believe GAN is a promising method in dealing with this challenge as it has a generator acting as data augmentation inherently. Yet, due to computational limitation, we decide not to try this approach as more data implies more iterations.

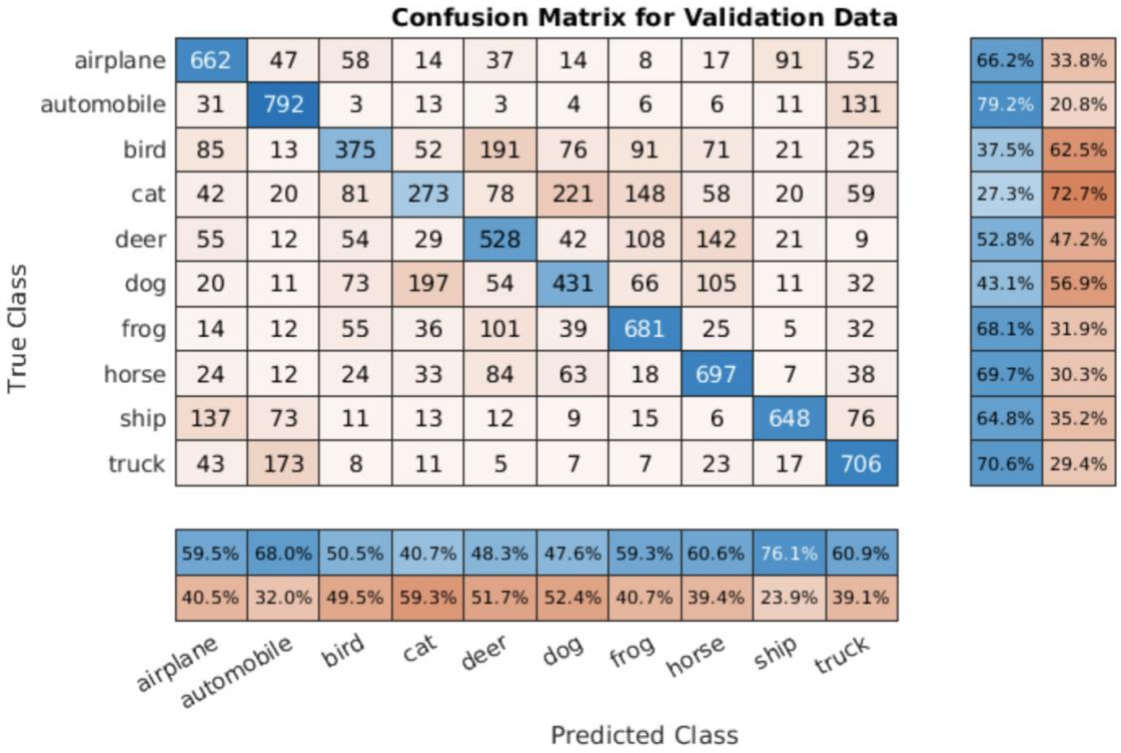
Evaluation results and discussion

We employed ResNet implementation from [MATLAB tutorial on CIFAR-10](). The performance was around 40% validation accuracy despite the effort of data augmentation. Though, our main focus is still data augmentation. No augmented data, nothing can be in consideration.

Our network architecture is of very small size which allows very fast inference. If there are anything that we are not happy, then that is its slow convergence. We applied batch normalization to speed up the network convergence and for better regularization [5]. It just worsens our network's performance, perhaps due to the fact that our network is very shallow.

Our best validation accuracy is 58.09% with validation cross-entropy loss 1.6958 at some iteration. Yet, MATLAB only automatically saves models by epoch, so here we present the best model we can obtain and save:

Validation accuracy	Validation cross-entropy	Train accuracy	Train cross-entropy
57.93%	1.76408	82.35%	0.5327



Our model does not work well with cat, bird and dog. Our model easily mistook automobile for truck, bird for deer, cat for dog and frog, dog for cat and horse, deer for frog and horse, frog for deer, ship for airplane, truck for automobile. It's easy to see that airplane, automobile, ship, truck, frog, horse, and deer can greatly benefit from data augmentation (slight rotation, sheer). Bird is a challenging class as the appearance greatly varies across species. It's also not so easy to distinguish between cat and dog given such low resolution.

Reference

[1] Vapnik, Vladimir. “The nature of statistical learning theory.” *Springer science & business media* (2013).
[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).
[3] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958.
[4] He, Kaiming, et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." *Proceedings of the IEEE international conference on computer vision*. 2015.
[5] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).