AI3607 Deep Learning: Assignment 2

Xiangyuan Xue (521030910387)
School of Electronic Information and Electrical Engineering

I. Dataset Construction

Download CIFAR10 dataset from the source. The dataset contains 60000 32×32 3-channel images of 10 classes, divided into 50000 training images and 10000 test images.

TABLE I
CIFAR10 COMPOSITION

#Train	#Test	#Total	#Class
50000	10000	60000	10

Override the `Dataset` class to construct our customized dataset for easier loading in training.

II. Neural Network Model

We prefer CNN-based models for image classification. Both simple CNN and pretrained VGG will be tried.

A. CNN Model

Here we refer to LeNet and design a simple CNN model for CIFAR10 classification.

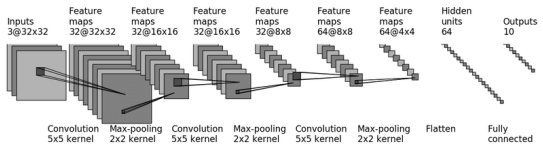

Fig. 1. simple CNN model

This model contains 3 convolutional layers, 3 max pooling layers, 2 fully connected layers and 1 output layer.

B. VGG Model

With a little modification, the pretrained VGG model can be utilized for CIFAR10 classification. [1]

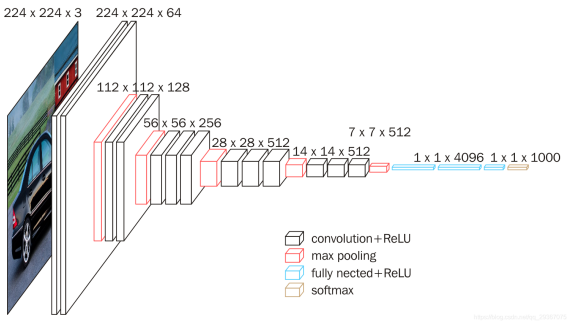

Fig. 2. original VGG model

Note that only the feature extractor is reserved, and the classifier is redesigned for CIFAR10 classification.

III. Experiment Result

Run the experiment script and we can find that the simple CNN model trains much faster than the pretrained VGG model. However, the total accuracy of the CNN model is approximately 70%, compared to 85% of the VGG model.

TABLE II
EXPERIMENT RESULT

Network	Loss	Accuracy
CNN	135.83	71.94%
VGG	10.87	85.34%

Considering the better performance of the VGG model, we utilize the pretrained VGG model for all the following experiments. Hence, we only show the training curve of the VGG model below.

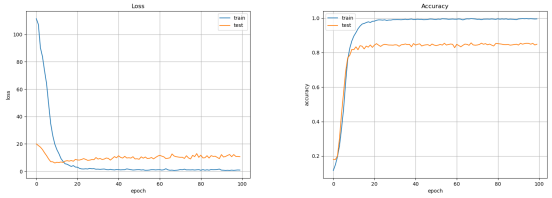

Fig. 3. training curve

The training curve shows that the pretrained VGG model converges quickly and performs high accuracy. However, the overfitting problem is still obvious, where the accuracy differs a lot between the training and test set.

IV. Unbalanced Dataset

The unbalanced dataset is created with a little modification, where the images with label $(0, 1, 2, 3, 4)$ are removed by 90%, while the images with label $(5, 6, 7, 8, 9)$ remain unchanged. Hence, the size of the training set is reduced from 50000 to 27500, which can lead to performance degradation.

An identical VGG model is trained on the unbalanced dataset, where the experiment result is shown below.

TABLE III
UNBALANCED RESULT

Set	Loss	Accuracy
Train	10.87	85.34%
Test	22.05	70.18%

The training curve is also shown below.

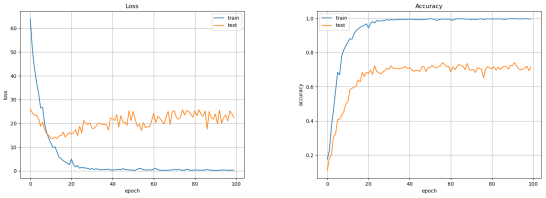

Fig. 4. training curve

We can see that the accuracy dropped by about 15% on the unbalanced dataset, and the overfitting problem is more serious. This problem comes from the imbalance of training, where the model is trained too much on the images with label $(5, 6, 7, 8, 9)$. Hence, the model works terribly on the images with label $(0, 1, 2, 3, 4)$. We will try to handle this problem in the following sections.

V. Model Improvement

A. Data Augmentation

Data augmentation is a good choice to rebalance the dataset. Here we use a simple method to augment the training set, where Guassian noise is added into the images with label $(0, 1, 2, 3, 4)$ and the variances of the noise are respectively $0.1, 0.2, \ldots, 0.9$, namely

$$\boldsymbol{N} \leftarrow \mathcal{N}(0, \sigma^2) \tag{1}$$

$$\boldsymbol{X} \leftarrow \boldsymbol{X} + \boldsymbol{N} \tag{2}$$

Data augmentation restores the size of the training set and make the dataset balanced, which may improve the performance of the VGG model. [2]

B. Model Generalization

The imbalance is essentially an out-of-distribution generalization problem. Invariant risk minimization (IRM) is a popular method to generalize the model. [3] The idea is to add a penalty term into the loss function specified as

$$\min_{\Phi:\boldsymbol{X}\to\boldsymbol{y}} \sum_{e\in E} R^e(\Phi) + \lambda \cdot \left\|\nabla_{\omega|\omega=\omega_0} R^e(\omega\cdot\Phi)\right\|^2 \tag{3}$$

where Φ is a predictor, ω is a scalar and λ is a regularizer. It has been proved that IRM can effectively generalize the model. In addition, this algorithm is quite easy to implement using the automatic derivation function.

VI. Experiment Result

We make a little modification to the dataset class to support data augmentation. The training function is also updated to implement the IRM algorithm. Then an identical VGG model is trained respectively on the augmented dataset, with the IRM algorithm, and with two methods combined together. The experiment result is shown below.

Note that comparing the total accuracy is meaningless because the model trained on the unbalanced dataset tends to predict the images with label $(5, 6, 7, 8, 9)$, which raises the total accuracy unreasonably. Hence, we provide the detailed accuracy of each class as follows.

TABLE IV
IMPROVED RESULT

Group	Loss	Accuracy
Augmented	23.66	71.09%
Generalized	19.75	69.49%
Combined	22.44	72.14%

TABLE V
DETAILED RESULT

Group	Unbalanced	Augmented	Generalized	Combined
0	65.60%	66.60%	67.70%	59.00%
1	77.40%	69.30%	69.70%	73.30%
2	38.10%	50.30%	38.30%	55.30%
3	16.20%	16.80%	29.30%	35.50%
4	44.70%	51.70%	41.40%	47.90%
5	88.10%	89.40%	84.70%	84.80%
6	92.70%	92.90%	91.80%	92.40%
7	91.90%	87.10%	89.30%	86.10%
8	93.90%	93.80%	90.70%	95.40%
9	93.20%	93.00%	92.00%	91.70%
Total	70.18%	71.09%	69.49%	72.14%

We can see that data augmentation improves the performance slightly and model generalization brings relatively larger improvement, where the images with label $(0, 1, 2, 3, 4)$ are predicted more accurately, despite the fact that the total accuracy is almost the same. Combining the two methods brings the best result, where both the detailed accuracy and the total accuracy are significantly improved.

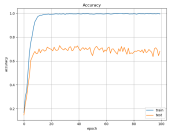

Fig. 5. augmented

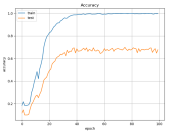

Fig. 6. generalized

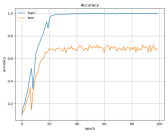

Fig. 7. combined

We can also see that the IRM algorithm leads to slower convergence, and such phenomenon can be alleviated when combined with data augmentation.

Appendix
Training Parameter

All the datasets are loaded with 1024 batch size and random shuffling enabled. We use cross entropy loss function and Adam optimizer with learning rate 10^{-3}. All the VGG models are trained for 100 epochs and trained and tested on a single NVIDIA A100 80G Tensor Core GPU.

References

[1] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[2] Vasconcelos C N, Vasconcelos B N. Convolutional neural network committees for melanoma classification with classical and expert knowledge based image transforms data augmentation[J]. arXiv preprint arXiv:1702.07025, 2017.

[3] Arjovsky M, Bottou L, Gulrajani I, et al. Invariant risk minimization[J]. arXiv preprint arXiv:1907.02893, 2019.