Report of Task 2

MoeKid101

1. The Basic Model

On the original dataset with 5000 training samples and 1000 test examples for each class, we develop our first baseline models. Inspired partially by VGG-16, I chose 3×3 convolution kernels for all convolution layers to reduce the number of model parameters as much as possible. Therefore, the basic convolutional units in my model is the VGG block (as is shown in Fig.1).

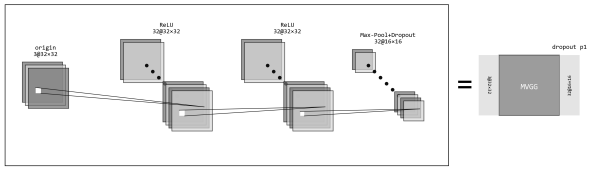


Figure 1. VGG block used in my models

Using the VGG block shown above, I got two models stacking two and three VGG blocks together respectively. Let's simply call them *CNN-2L* (Fig.2) and *CNN-3L* (Fig.3). Both models were trained with data augmentation, and they achieved a similar accuracy of 84.3% and 84.8%.

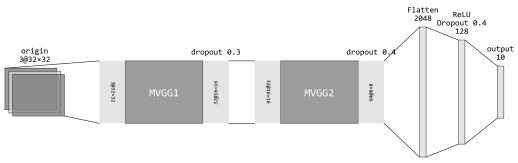


Figure 2. Basic CNN stacking two VGG blocks

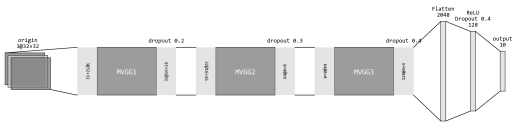


Figure 3. Basic CNN stacking three VGG blocks

1.1. Data Augmentation

Among the several popular means of data augmentation, we adopted *horizontal flipping* and *random erasing*, which have proven useful on CIFAR-10.

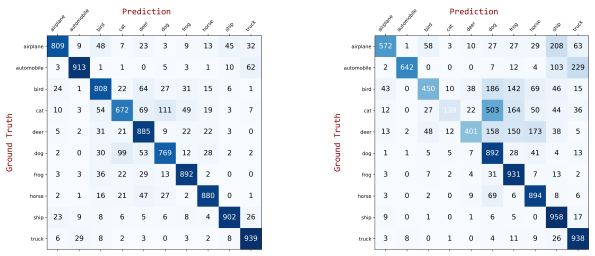
2. Class Imbalance Problem

We would dive into class imbalance problem in this section, which is defined as the phenomenon that collected training data of some labels have significantly more samples than that of other labels. Such a phenomenon might cause the training data of one type to overwhelm another. Before hanging out, we define a value ρ which quantifies exactly how imbalanced the training dataset is.

$$\rho = \frac{\max_i |C_i|}{\max_i |C_i|} \tag{1}$$

2.1. Effect of Class Imbalance on Classification

By introducing class imbalance with $\rho \approx 10$ to CIFAR-10, we witness considerable regression in both model (84.3% to 66.4% in *CNN-2L*; 84.8% to 68.3% in *CNN-3L*), and the evidence of "overwhelming" is clearly shown in the confusion matrices in Fig.4, the most significant one of which is that almost all cat images are misclassified as something else.



(a) Full training set. (b) Masked training set.

Figure 4. Constrast between model performance with and without class imbalance.

2.2. Random Over Sampling

The currently most popular solutions to class imbalance problems drop into three types: data-level, algorithm-level and hybrid methods. Among the common data-level methods, Buda *et al*. stated that *random over sampling* (ROS) was the best one, so I decided to take ROS and explore its influences.

As is implied by its name, *random over sampling* generates replicate samples in minority class at random to reduce the value of ρ till a certain level. It causes samples in minority classes to appear many times, instead of only once, so that the overall data distribution is altered towards balance between majority classes and minority classes.

2.3. Experiments and Analyses

2.3.1 Trial One: Failure on *CNN-3L*

Among the models mentioned in section 1, *CNN-3L* was originally chosen as my baseline model, so I had been working on it at first. I conducted ROS on the masked dataset (with $\rho = 5000/487 \approx 10.27$) till $\rho = 1.0$, and trained it on *CNN-3L* with augmentation. However, I got completely no improvement on the performance of my model, with the accuracy even dropped slightly from 68.3% to 67.2%. The details are shown in Fig.5.

(a) Baseline *CNN-3L* (b) *CNN-3L* after ROS

Figure 5. ROS implemented on *CNN-3L*, no improvement!

This is very strange given the wide usage of ROS in solving class imbalance. But after I checked the model performance on training set, I had some idea. Both the baseline model and the ROS model achieved $> 99.9\%$ accuracy on training set despite the extremely low accuracy on minority classes of test set. It suggested that the problem was with generalization instead of class imbalance, and obviously, a "weaker" model is needed to enable better generalization.

2.3.2 Trial Two: Generalization Helps ROS Work

Following the idea above, I came up with the *CNN-2L* model. Still using the over-sampled dataset with $\rho = 1.0$ and training with augmentation, we saw some progress. It achieved 71.1% accuracy on the test set, given that the baseline accuracy was 66.4%.

To step further, we know that dropout is a generalization technique understood as simultaneously training $C_n^{\lceil (1-p)n \rceil}$ models (where n is the number of neurons in the current layer while p is the dropout rate) and average them together, and the generalization ability it brings about is positively related to the number of models it was training.

Therefore, I set the dropout rate to be $p = 0.5$ to obtain a maximum value of $C_n^{\lceil (1-p)n \rceil}$. It turns out that it helps. On *CNN-3L*, a 0.5 dropout rate improves the performance by 1.0%, while on *CNN-2L*, it improves the performance by 2.0%, leading to a best accuracy of 73.1%. The comparison between confusion matrices is shown in Fig.6

(a) Baseline *CNN-2L* (b) *CNN-2L* after ROS

Figure 6. ROS implemented on *CNN-2L*, there is improvement!

3. Understanding ROS and Over-fitting

We have found that the performance of ROS directly depends on the baseline model we choose, or the capability of generalization of the baseline model. Then the problem arises about how to understand ROS.

From my intuition built during the experiments, I interpret the overfitting problem with ROS as follows: in the extracted feature space of our model, the distribution of minority classes is too sparse so that samples belonging to other classes could invade into the gaps between samples of minority class. Such sparsity is presented because we choose a model with too large capacity. When this happens, ROS itself can't help us more. Instead, the SMOTE algorithm which creates synthetic data samples in the feature space directly could possibly help us.

Then we can develop our conclusion as: ROS works only if we have a moderate model that is not too powerful. When the model complexity reaches a certain limit, ROS only serves as an aggravation of over-fitting.

4. Conclusion

In the task above, we have explored the effects of class imbalance and one vanilla solution to the problem, namely *random over sampling*. During the experiments I gained some insights into the training process of neural networks, and I really benefited a lot from this task.