Deep Learning for Data Science Midterm Report

Chen Yu (Erioe), Liu
Boston University
erioe@bu.edu

1. AI Disclosure

I used ChatGPT to help debugging, report writing, and some model architecture suggestion.

2. Model Description

For the model, I tried different popular architectures, including ResNet, DenseNet, and ConvNext. Also, I tried different type of classification.

2.1. Simple Convolution Network

Because the image resolution in Cifar-100 dataset is relatively small, only 32*32 pixel. Therefore, for the simple convolution network, I used four Convolution layers, one Maxpooling layer, a Convolution Block Attention Module (CBAM), and two fully connected layers at the end. Using the augumented dataset with traditional classification learning, this simple cnn achieved about 0.6 accuracy with the testing data, and 0.45 in the public out-of-distribution data.

2.2. Better Convolution Network

I used both ResNet and ConvNext to train the model. Without modification, and using both medium size of the model, ConvNext Base achieved better performance then ResNet34. Because the Cifar100 image are small, with only 32 * 32 pixel, so I tried to smaller the kernel size and stride of the first convolution layer and remove the maxpolling layer in the ResNet50, ResNet101, and ResNet152 .

2.3. Contrastive Learning

I set the base model as the image encoder, and used CLIP to obtain the text encoding for all categories. Following the principles of Open-Vocabulary Detection and Contrastive Learning, the image encoder outputs a 512-dimensional feature vector. This can be achieved by modifying the original model to output 512 classes, followed by a softmax layer. This vector is then compared with each category's text encoding using a dot product operation. The model objective is that the dot product between the image encoding and its corresponding category's text encoding should be large, while the dot product with other categories' text encodings should be small.

During inference, the image is passed through the image encoder to obtain its feature vector. This vector is then compared with all category text encodings using the dot product, and the category with the highest score is selected as the prediction.

2.4. Hierarchical Learning

The CIFAR-100 dataset contains many similar categories, such as "girl" and "woman" or "bus" and "streetcar." To address this, I organized the data into coarse and fine categories. The coarse categories represent broader, general classes, while each coarse category contains several related fine categories. For example, the coarse category

”people” includes the fine categories ”man”, ”woman”, ”boy”, ”girl”, and ”baby”. Using Hierarchical Learning, I first trained a model to classify images into coarse categories. Then, for each coarse category, I trained a separate model to classify its corresponding fine categories.

During inference, the image is first passed through the coarse classification model to obtain its coarse category. The image is then forwarded to the corresponding fine classification model to produce the final prediction.

One advantage of Hierarchical Learning is that different model architectures can be designed for the coarse and fine classifiers. Since coarse categories are generally easier to distinguish, a simpler model may suffice, while fine categories often require a more complex model for accurate classification. Additionally, this approach can be effectively combined with contrastive learning to further enhance performance.

3. Hyperparameter Tuning

For batch size, I experimented with values of 64, 128, and 256. Since the performance across these settings was similar, I selected 256 to accelerate training. The loss function used was Cross Entropy with label smoothing, and the optimizer was AdamW.

4. Regularization Techniques

To improve model generalization and prevent overfitting, I employed several regularization techniques. These include Dropout, which randomly deactivates neurons during training; Data Augmentation, to increase data diversity; and Label-Smoothed Cross-Entropy loss, which mitigates overconfidence by distributing a small portion of the probability mass across all classes. Additionally, during training, I applied MixUp and Random Erasing transformations to further enhance the model's robustness.

5. Data Augmentation Strategy

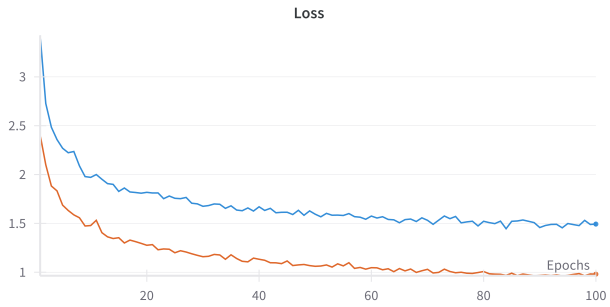
For data augmentation, I applied various techniques to generate various image variations. These include horizontal flipping, random rotation, random cropping, and random brightness adjustments to create different visual states. Additionally, I added noise to the training data using methods such as blurring and adding random noise, enhancing the model's robustness. Below is the setting of augmentation.

```
import albumentations as A
TRANSFORMS = A.Compose([
    A.RandomCrop(width=int(IMAGE_SIZE*0.8), height=int(IMAGE_SIZE*0.8), p=0.2),
    A.HorizontalFlip(p=0.5),
    A.RandomBrightnessContrast(p=0.2),
    A.RGBShift(p=0.2),
    A.Rotate(limit=15, p=0.2),
    A.AdvancedBlur(blur_limit=5, p=0.3),
    A.ISONoise(color_shift=(0.01, 0.05), intensity=(0.1, 0.5), p=0.3),
    A.MotionBlur(blur_limit=5, p=0.2),
    A.ImageCompression(p=0.1),
    A.Resize(IMAGE_SIZE, IMAGE_SIZE),
])
```

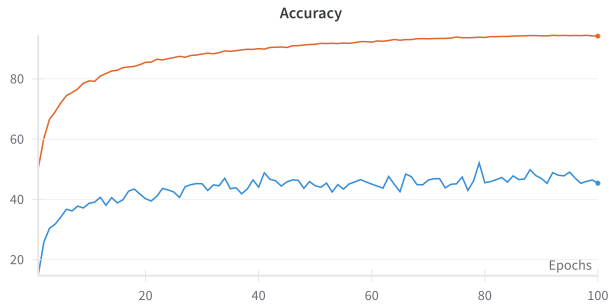
6. Experiment Tracking Summary

After training all architectures with their base configurations, the modified version of ResNet achieved the highest accuracy. Specifically, reducing the kernel size in the first convolutional layer and removing the max pooling layer improved accuracy from approximately 60% to 80%. Additionally, incorporating MixUp enhanced model robustness, as reflected by a smaller gap between validation and test accuracy.

Contrastive Learning yielded performance comparable to traditional classification methods, while Hierarchical Learning resulted in lower accuracy. Therefore, for the final model, I adopted a modified ResNet152 with MixUp and Random Erasing augmentations, achieving 83.27% accuracy on the test set and a 74.25% public score on Kaggle.



(a) Training & Validation Loss



(b) Training & Validation Accuracy

Figure 1. Comparison between Model A and Model B on validation and test accuracy. The blue line represents the training accuracy, while the orange line represents the validation accuracy. Since MixUp was applied to the training data, the training accuracy appears significantly lower than the validation accuracy.

Table 1. Misclassified Classes. Most of the misclassifications occur between visually similar classes, such as cloud and sea, girl and woman, or bus and streetcar. Since CIFAR-100 images are relatively small and low-resolution, it is challenging for the model to accurately distinguish between such closely related categories.

Ground Truth	Predicted	Proportion
cloud	sea	0.0281
plain	sea	0.0261
oak_tree	maple_tree	0.0259
maple_tree	oak_tree	0.0240
baby	girl	0.0230
boy	girl	0.0219
girl	woman	0.0212
bus	streetcar	0.0207
willow_tree	oak_tree	0.0204