HW1 Image Classification

Selected Topics in Visual Recognition using Deep Learning

GitHub link

https://github.com/samuelyutt/Selected-Topics-in-Visual-Recognition-using-Deep-Learning-course/tree/hw1/hw1-ImageClassification

Introduction

ResNet is a deep neural network based on residual blocks. The benefit of residual learning is making extremely deep networks to become trainable. Widely known ResNet structures include ResNet50, ResNet101, and ResNet152. The differences between these structure are expansion block size and number of blocks.

ResNet with carnality are called ResNeXt. Rather than Interception network, ResNeXt uses the split-transform-merge technic by applying a simple and efficient method called equivalent cardinal. According to researches, ResNeXt performs better than ResNet under same conditions.

Methodology

Data pre-process

Data transforming is used during data pre-process. Original training images are random resized and cropped to shape of (224, 224), random horizontal flipped, and normalized. I called these basic transforms because they are necessary.

Some advanced and optional transforming such as RandomAffine, RandomPerspective, and add random noise are sometimes used as well.

Original training images are randomly split into two parts: train set and validation set. Train set contains 90% of the original training data, while validation set contains the other 10%.

Both train set and validation set are set to use batch size 16. Only the train set is shuffled.

Model architecture

I trained ResNet18, ResNet50, ResNet152, and ResNeXt101_32x8d models with pertained data.

I tried 2 major transfer learning scenarios as the following:

Finetuning

The output size of the final fully connected layer of each model is set to 200.

Fixed feature extractor

The weights of the whole network except that of the final fully connected layer are frizzed. This technic did not result in better accuracy.

Hyperparameters

The learning rate is set to 0.01 at the beginning of every training, and will be tuned if the validation accuracy maintains at a certain level after many epochs.

Momentum is set to 0.9. Cross entropy loss and stochastic gradient descent are used for back propagation.

A checkpoint of weights are stored for every 10 epochs. Meanwhile, a validation test will be performed as well.

Weighted bagging

Since I trained a lot of networks, I decide to use bagging method instead of depending on a single network. Up to 4 models of highest accuracy on testing data are chosen, and for each input testing image, all 4 outputs are simply summed up to make a final decision. The bagging method increased the testing accuracy by approximately 0.05.

After simply summing up the outputs, I realized that each model has different accuracy. As the result, their outputs importance may be different. I then decided to multiply each outputs by a weight before summing them up. The weight of each model is their accuracy on testing data. Final prediction will be decided by the weighted sum. The weighted bagging method gave a slightly higher accuracy than that of simply bagging method.

Summary

In this homework, I have worked on two types of networks, ResNet and ResNeXt. ResNeXt gave better results, which is unsurprising. Also, multiple technics are used during data pre-processing. If the advanced transforms are used, the results were better. However, whether adding random noise or not did not significantly affect the results. As for transfer learning, tuning the final layer is required, but freezing the other layers' weights resulted in much worse accuracies. Additionally, searching for hyper parameters was extremely time consuming; nevertheless, it was worth to search since some parameters such as learning rate and batch size do affect the efficiency. Last but not least, bagging method do increased the accuracies significantly.