Mini-Project 1: Residual Network Design

ECE-GY-7123 Teaching Team
ECE, NYU Tandon
Spring 2022

Abstract

In this project, you will design a ResNet architecture to maximize accuracy on the CIFAR-10 dataset under a constraint on parameter count.

1 Introduction

ResNets (or Residual Networks) are one of the most commonly used models for image classification tasks. In this project, you will design and train your own ResNet model for CIFAR-10 image classification. In particular, your goal will be to maximize accuracy on the CIFAR-10 benchmark *while* keeping the size of your ResNet model under some specified fix budget. Model size, typically measured as the number or trainable parameters, is important when models need to be stored on devices with limited storage capacity (e.g. IoT/edge devices).

2 Problem Statement

2.1 ResNet Description

Figure 1 shows a ResNet-18 model, so-called because it has 18 layers. ResNet-18 is just, however, one example of a ResNet architecture. In general, any model that uses residual connections can be classified as a ResNet. First, let us establish some terminology.

Residual Layer. The ResNet consists of N Residual Layers ($N = 4$ for ResNet-18). Each residual layers contains one or more residual blocks. Layer i has B_i blocks (for ResNet-18, $B_i = 4$ for $i = \{1, 2, 3, 4\}$). Residual layers are preceded by a fixed convolutional block and followed by an average pooling and fully connected layer.

Residual Block. A residual block contains two convolutional layers with a skip connection from the block's input to the block's output. Let `F` implement the following sequence of operations: `conv`→`bn`→`relu`→`conv`→`bn` (`bn` is some batch normalization). A residual block implements:

$$\texttt{relu}(\texttt{F}(x) + \texttt{conv}(x)),$$

where the $\texttt{conv}(x)$ term corresponds to the skip connection.

The `conv` layers in any residual block of residual layer i have C_i channels. As a rule, $C_{i+1} = 2C_i$, but you get to select C_1. The `conv` layers and skip connections in any residual block of residual layer i have filters of size $K_i \times K_i$ and $K_i \times K_i$, respectively.

The *first* `conv` in the first residual block of residual layer $i \geq 2$ is special because it has a stride of 2. For this reason, the skip connection of the first residual block of residual layer $i \geq 2$ also has a stride of 2. In all other cases the stride is 1.

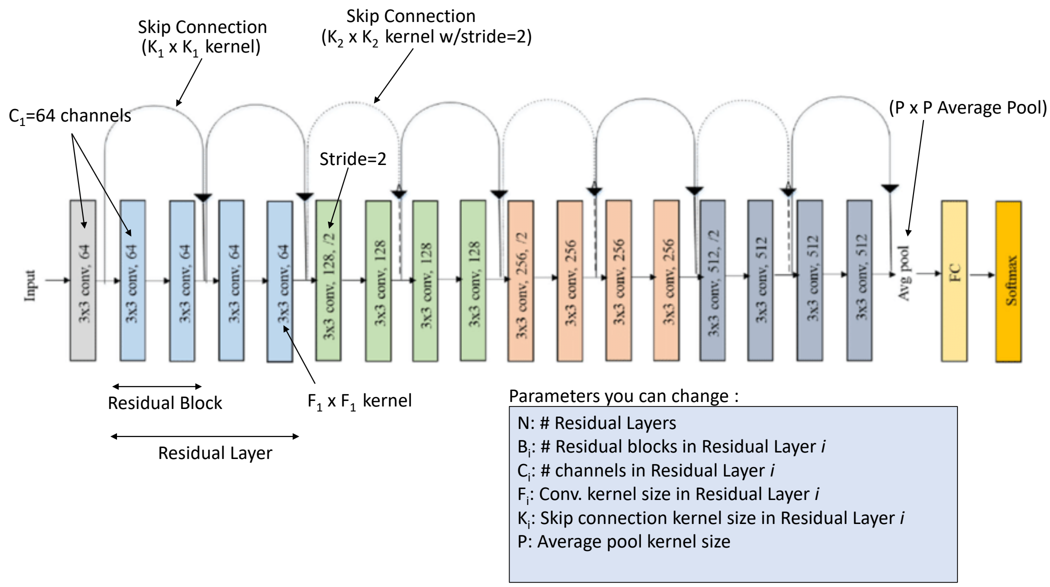

Figure 1: ResNet description for mini-project 1.

Input Layers. The $32 \times 32 \times 3$ input image is processed by `conv` layer $F_1 \times F_1$ sized filters and C_1 output channels. This is followed by `bn` and `relu`.

Average Pool and Output Layers. The last residual layer's output feeds into a $P \times P$ average pooling layer, followed by an appropriately sized fully connected layer. Finally, the output of dimension 10 that are passed through a softmax layer.

2.2 Your Goal

Your goal is to select the ResNet hyper-parameters, N, B_i, C_1, F_i, K_i and P so as to: (1) maximize test accuracy on CIFAR-10 while ensuring (2) your ResNet model has less than 5M trainable parameters. Other than the hyper-parameters above you are not allowed to make any other changes to the ResNet achitecture.

You have broad flexibility in how to train your model. In particular, you are allowed to use:

- Any data augmentation strategies you like (this is highly encouraged!).
- Any optimizer, including but not limited to SGD, ADAM, etc.
- Any regularization scheme.
- Any choice of batch size and epochs of training, or even new/fancy training procedures.

However, you must train your models from scratch. *You are NOT allowed to use any pre-trained models, or any dataset other than CIFAR-10.* If in doubt, we will make sure that your results are reproducible (see Section 3.3).

You can use/adapt methods augmentation, optimization, regularization methods proposed in prior work, and if you like, you can use their code. However, you must cite all papers you relied, and importantly, references/links to any repositories you borrow code from. **Borrowed/adapted code from an online source without a reference will be considered plagiarism and lead to an automatic zero.**

3 What to Submit

Your submissions are due by **midnight March 13** (EST) on Brightspace. Complete submissions must include the following components.

3.1 Report

An up to **8 page** report describing your methodology and results. The report **must be in the Neurips 2015 paper format** (similar format as this document). Style file for the Neurips paper format in Latex and Word formats can be found here (`https://nips.cc/Conferences/2015/PaperInformation/StyleFiles`).

Your report should be structured as a typical paper with an abstract, introduction, methodology, results and conclusion along with a list of citation. It must contain:

- A "Methodology" section that describes how you went about designing and training your ResNet model. Describe your architecture search process, and how you selected data augmentation methods, choice of optimizer etc. Wherever possible, provide data in the form of graphs/tables to justify your choices.
- Any "Results" section that describes the final architecture you arrived at, the number of trainable parameters of this architecture details of how this model was trained, curves plotting the training and test loss/error versus epochs, and the final test error achieved.
- Detailed citations to any paper or code you relied upon as references.
- A link to a GitHub repository containing your code and a README file. The code in the GitHub repo must allow us to retrain the final architecture you selected using your training procedure, which should be clearly described in your README file.

3.2 Trained Model Files

You must also submit a two files, `project1_model.py` that contains a PyTorch description of your ResNet model architecture and `project1_model.pt` which contains the trained weights of this architecture. We will release an examples of these files and also details on how save your trained weights in a `.pt` file on Brightspace.

3.3 Evaluation

We will evaluate your submissions

- Results (30%): we will evaluate your submitted models on the CIFAR-10 test dataset, and on our own held-out test set. Every team that submits a valid model with <5M parameters and >80% accuracy on the CIFAR-10 test set will get 15 points. The remaining 15 points will be allocated competitively based on where you stand with respect to the accuracy of all other submissions.
- Comprehensiveness and originality (40%): we will evaluate the comprehensiveness and originality of your methodology to select the right model hyper-parameters and training procedures, as described in your report.
- Report (20%): your report will be evaluated based on its quality, clearness and readability as well as thoroughness in describing the methodology, use of graphs and tables, citing appropriate references etc.
- Reproducability (10%): we will check if your GitHub repos are sufficient to reproduce your final results, i.e., contain the final model architecture and training scripts to train the model to reproduce the results reported in the report. Note, we realize that running the same training process twice can produce slightly different results, so no need to worry about reproducing the *exact* test accuracy your reported.