Assignment

Neural Networks and Deep Learning (ECS7026P)

In this assignment, you will implement a neural network that classifies images. Please read this entire document before you start working on the assignment.

1 Dataset

The CIFAR-10 dataset is composed of 60000 small ($3 \times 32 \times 32$) color images, each of which belongs to one of 10 classes. There are 6000 images per class. The images are divided into a training dataset composed of 50000 examples and a testing dataset composed of 10000 examples. This dataset is readily available for *PyTorch*.

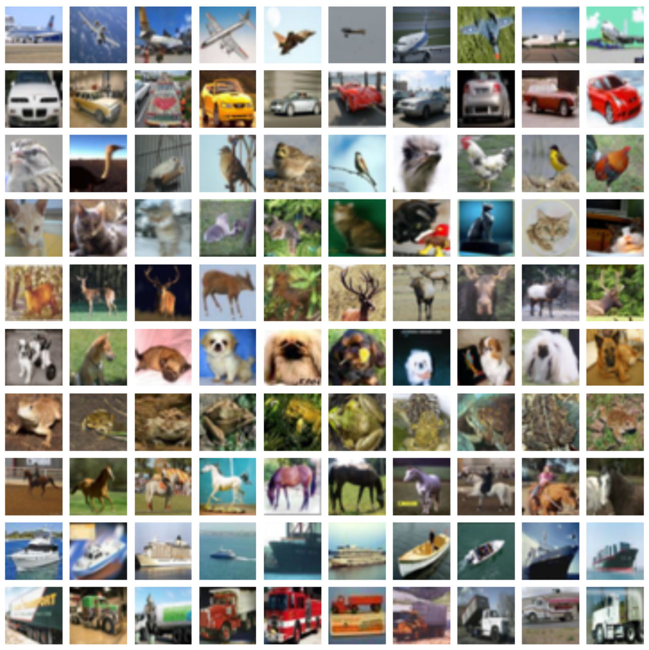

Figure 1: Examples from CIFAR-10 (classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck).

Your first task is to create a *DataLoader* for the training dataset and a *DataLoader* for the testing dataset, which should enable generating batches of examples.

2 Basic architecture

Your next task is to implement the neural network architecture described in this section to classify images from the CIFAR-10 dataset. The architecture is composed of a sequence of *intermediate blocks* $B_1, B_2, \ldots, B_K$ that are followed by an *output block* O, as shown in Figure 2. These blocks are detailed in the following subsections.

2.1 Intermediate block

An intermediate block receives an image x and outputs an image x'. Each block has L independent convolutional layers. Each convolutional layer C_l in a block receives the input image x and outputs an image $C_l(x)$. Each of these images is combined into the single output image x', which is given by

$$x' = a_1 C_1(x) + a_2 C_2(x) + \ldots + a_L C_L(x),$$

where $\mathbf{a} = [a_1, a_2, \ldots, a_L]^T$ is a vector that is also computed by the block. Note that each convolutional layer in a block receives the same input image x (and not the output of another convolutional layer within the block).

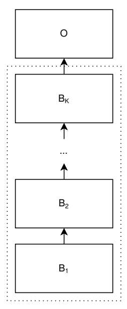

Figure 2: Neural network architecture diagram.

Suppose that the input image x has c channels. In order to compute the vector $\mathbf{a}$, the average value of each channel of x is computed and stored into a c-dimensional vector $\mathbf{m}$. The vector $\mathbf{m}$ is the input to a fully connected layer that outputs the vector $\mathbf{a}$. Note that this fully connected layer should have as many units as there are convolutional layers in the block.

Each block in the basic architecture may have a different number of convolutional layers, and each convolutional layer may have different hyperparameters (within or across blocks). However, every convolutional layer within a block should output an image with the same shape.

2.2 Output block

The output block receives an image x (output of the last intermediate block) and outputs a logits vector $\mathbf{o}$.

Suppose that the input image x has c channels. In order to compute the vector $\mathbf{o}$, the average value of each channel of x is computed and stored into a c-dimensional vector $\mathbf{m}$. The vector $\mathbf{m}$ is the input to a sequence of zero or more fully connected layer(s) that output the vector $\mathbf{o}$.

3 Training and testing

Your next task is to train a neural network with the basic architecture described in the previous section and compute its accuracy in the testing dataset.

For a given batch size b, your network should receive a $b \times 3 \times 32 \times 32$ tensor composed of b images of shape $3 \times 32 \times 32$ and output a $b \times 10$ logits matrix $\mathbf{O}$. You should use a cross entropy loss for training.

You should make the remaining decisions (such as hyperparameter settings) by yourself. You may adapt the code provided in previous lectures to compute evaluation metrics and implement the training procedure.

Hint: Before implementing the basic architecture described in the previous section, you may want to implement a simpler architecture that allows testing whether the remaining components are working. Your experiments may also benefit from using a GPU through Google Colab (Runtime > Change runtime type).

4 Improving the results

Your next task is to improve the results of your initial implementation. You should train different neural networks in the training dataset to find a neural network that achieves the highest possible accuracy in the testing dataset. Your mark will depend on this accuracy.

Note that you are being asked to use the *testing dataset* as if it were a *validation dataset* in order to simplify this assignment. This is generally a methodological mistake, since a testing dataset should only be used to assess generalization after the hyperparameters are chosen (rather than used to choose hyperparameters).

In order to improve the results of your initial implementation, you can try different hyperparameters and employ all the techniques covered during Week 6 and Week 8. You can also make mild changes to the basic architecture, such as including additional layers between intermediate blocks. In order to inspire your solution, you are strongly encouraged to find and study existing code that implements neural networks for (CIFAR-10) image classification.

You are not allowed to radically change the basic architecture described in Section 2. For example, each intermediate block must weight the outputs of independent convolutional layers using coefficients obtained by a fully connected layer that receives the average value of each channel of the input image. As a general rule, reverting your implementation to an implementation of the basic architecture described in Section 2 should be trivial.

If you are not sure whether a change would be considered too radical, please ask. This assignment is intended to assess whether you are able to implement an architecture based on a

high-level description (rather than copying or adapting existing code). Therefore, you will receive no marks if you implement an architecture that is not clearly based on the basic architecture described in Section 2, regardless of its performance.

You should store the following statistics about the neural network that achieves the highest accuracy in the testing dataset: loss for each training batch, training accuracy after each epoch, and testing accuracy after each epoch. You should plot the loss for each training batch. You should also plot the training accuracy and testing accuracy for each training epoch.

5 Submission

The deadline for submitting this assignment can be found on QM+. Penalties for late submissions will be applied in accordance with the School policy. The submission cut-off date is 7 days after the deadline. Submissions should be made through QM+. Submissions by e-mail will be ignored. Please check whether the file was uploaded correctly to QM+. Cases of extenuating circumstances have to go through the proper procedure in accordance with the School policy. Only cases approved by the School in due time will be considered.

You should work individually. Collaboration between students is prohibited, and we will take action if there is evidence of collusion or plagiarism. If you are unsure about what constitutes plagiarism, please ask. You are strongly encouraged to find and study existing code that implements neural networks for (CIFAR-10) image classification. If you borrow ideas from such code, this should be clearly documented in your submission. You should not ask questions through public channels that reveal aspects of your solution. Preferentially, ask your questions in private during lab sessions.

5.1 Marking

This assignment corresponds to 50% of the final mark for this module. This assignment should be submitted as a single compressed (zip) file. This zip file should contain a single folder, which should contain only two files:

- A Jupyter Notebook file (ipynb) that contains your code for this assignment. Your implementation should use PyTorch, and should run on Google Colab without requiring access to Google Drive. If your code does not run on Google Colab without issues, your marks may be severely penalized. Your code should be well documented and organized, so that it is relatively easy to understand what you have implemented. Make sure that the notebook also includes the output of a successful execution. In other words, do not clear the output of the cells before submitting the notebook. Your marks may be severely penalized if your notebook does not contain outputs.

- A portable document file (pdf) that contains your report for this assignment. The report should be at most 3 pages long. If you submit a report in another format (such as doc, docx, odt) or with more than 3 pages, your marks may be severely penalized. The report must be excellently organized and identified with your name, student number, and module identifier. The report should describe your neural network architecture, focusing on how it deviates from the basic architecture described in Section 2. The report should include a list of hyperparameters and techniques employed for training. For the neural network that achieved the highest accuracy in the testing dataset, the report should also include a plot of the loss for each training batch, and a plot of the training accuracy and testing accuracy for each training epoch. Besides including these plots, the report should clearly state the highest accuracy obtained in the testing dataset. Optionally, you may provide a brief history of how you tried to improve the results of your initial implementation.

Based on the correctness and clarity of your code and on the information provided in the report, you will receive the following number of points for accomplishing each of the following tasks:

1. Loading the CIFAR-10 dataset into appropriate *DataLoaders* [5/100]
2. Implementing a neural network based on the architecture described in Section 2 [40/100]
3. Implementing appropriate code to train this neural network [5/100]
4. Training this neural network in the training dataset and computing its accuracy in the testing dataset, as evidenced by the following information provided in the report: [30/100]
   - Plot of the loss for each training batch
   - Plot of the training accuracy and testing accuracy for each training epoch
   - Accuracy obtained in the testing dataset

- Description of hyperparameters and techniques employed for training

Finally, depending on the accuracy a obtained by this neural network in the testing dataset, you will receive the following number of points:

- $a < 70\%$ [0/100]
- $70\% \leq a < 80\%$ [5/100]
- $80\% \leq a < 85\%$ [10/100]
- $85\% \leq a < 92\%$ [15/100]
- $a \geq 92\%$ [20/100]

Hints: Minimal hyperparameter search with a correct implementation should achieve 70% accuracy. Achieving 85% accuracy typically requires some hyperparameter search, some of the techniques covered in Week 6, and borrowing principles from the architectures covered in Week 8. Achieving 92% accuracy typically requires extensive experimentation with hyperparameters, techniques, and architectures. If you intend to achieve $a \geq 92\%$, we recommend borrowing ideas from state-of-the-art implementations.