Machine Learning
Università della Svizzera italiana

Assignment 2

May 17, 2021

In this assignment you are asked to:

1. Implement a neural network to classify images from the `CIFAR10` dataset;
2. Fine-tune a pre-trained neural network to classify rock, paper, scissors hand gestures.

Both requests are very similar to what we have seen during the labs. However, you are required to follow **exactly** the assignment's specifications.

1 Follow our recipe

Implement a multi-class classifier to identify the subject of the images from `CIFAR-10` data set. To simply the problem, we restrict the classes to 3: `airplane`, `automobile` and `bird`.

1. Download and load `CIFAR-10` dataset using the following function, and consider only the first three classes. Check `src/utils.py`, there is already a function for this!
2. Preprocess the data:
   - Normalize each pixel of each channel so that the range is [0, 1];
   - Create one-hot encoding of the labels.
3. Build a neural network with the following architecture:
   - Convolutional layer, with 8 filters of size 5×5, stride of 1×1, and ReLU activation;
   - Max pooling layer, with pooling size of 2×2;
   - Convolutional layer, with 16 filters of size 3×3, stride of 2×2, and ReLU activation;

- Average pooling layer, with pooling size of 2×2;
- Layer to convert the 2D feature maps to vectors (Flatten layer);
- Dense layer with 8 neurons and tanh activation;
- Dense output layer with softmax activation;

4. Train the model on the training set from point 1 for 500 epochs:
   - Use the RMSprop optimization algorithm, with a learning rate of 0.003 and a batch size of 128;
   - Use categorical cross-entropy as a loss function;
   - Implement early stopping, monitoring the validation accuracy of the model with a patience of 10 epochs and use 20% of the training data as validation set;
   - When early stopping kicks in, and the training procedure stops, restore the best model found during training.
5. Draw a plot with epochs on the x-axis and with two graphs: the train accuracy and the validation accuracy (remember to add a legend to distinguish the two graphs!).
6. Assess the performances of the network on the test set loaded in point 1, and provide an estimate of the classification accuracy that you expect on new and unseen images.
7. **Bonus** (Optional) Tune the learning rate and the number of neurons in the last dense hidden layer with a **grid search** to improve the performances (if feasible).
   - Consider the following options for the two hyper-parameters (4 models in total):
     - learning rate: [0.01, 0.0001]
     - number of neurons: [16, 64]
   - Keep all the other hyper-parameters as in point 3.
   - Perform a grid search on the chosen ranges based on hold-out cross-validation in the training set and identify the most promising hyper-parameter setup.
   - Compare the accuracy on the test set achieved by the most promising configuration with that of the model obtained in point 4. Are the accuracy levels statistically different?

2 Transfer learning

In this task, we will fine-tune the last layer of a pretrained model in order to build a classifier for the *rock, paper, scissors dataset* that we acquired for the lab. The objective is to make use of the experience collected on a task to bootstrap the performances on a different task. We are going to use the `VGG16` network, pretrained on Imagenet to compete in the ILSVRC-2014 competition.

`VGG16` is very expensive to train from scratch, but luckily the VGG team publicly released the trained weights of the network, so that people could use it for transfer learning. As we discussed during classes, this can be achieved by **removing the last fully connected layers** form the pretrained model and by using the output of the convolutional layers (with freezed weights) as input to a **new fully connected network**. This last part of the model is then trained from scratch on the task of interest.

1. Use `keras` to download a pretrained version of the `vgg16` network. You can start from the snippet of code you find on the repository of the assignment.
2. Download and preprocess the rock, paper, scissor dataset that we collected for the lab. You find the functions to download and build the dataset in `src/utils.py`. Vgg16 provides a function to prepropress the input
   `applications.vgg16.preprocess_input`
   You may decide to use it. Use 224×224 as image dimension.
3. Add a hidden layer (use any number of units and the activation function that you want), then add an output layer suitable for the hand gesture classification problem.
4. Train with and without **data augmentation** and report the learning curves (train and validation accuracy) for both cases.
   - Turn on the GPU environment on Colab, otherwise training will be slow.
   - Train for 50 epochs or until convergence.
   - Comment if using data augmentation led to an improvement or not.