MLAI - HOMEWORK 2

Deep Learning

Teaching assistants: Frattin Fabio and Lorenzo Bonasera

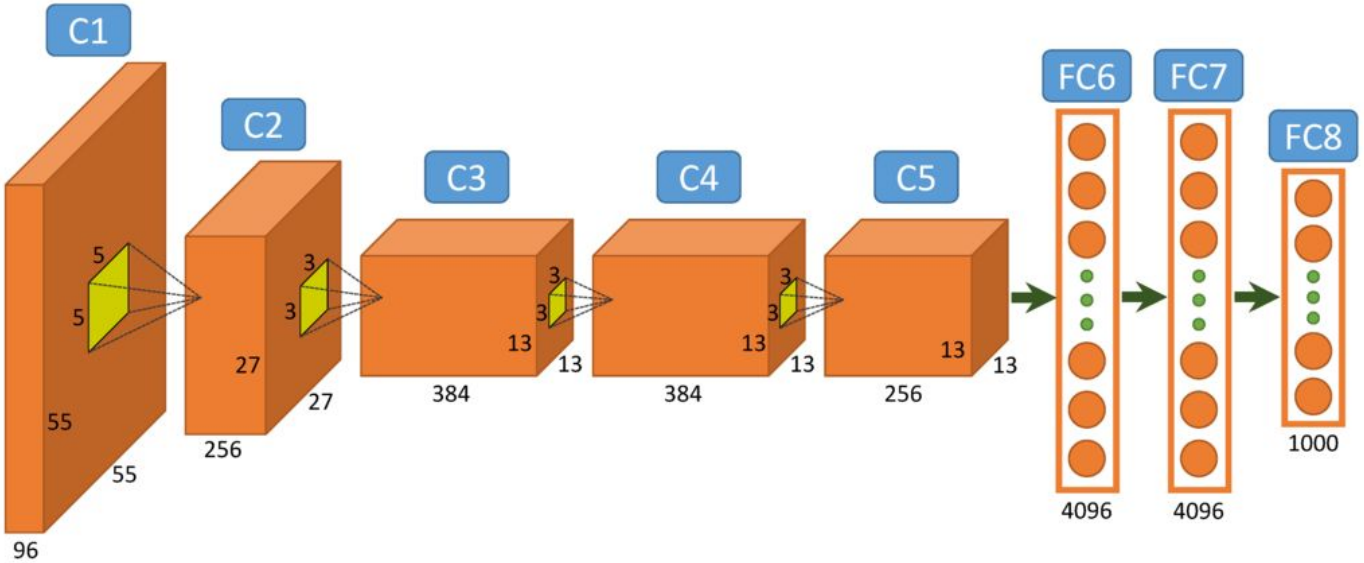
STEP 0 - OVERVIEW

HOMEWORK PURPOSE

- You are going to train a Convolutional Neural Network for image classification
  - CNN: Alexnet
  - Data: Caltech-101

- You are required to fill a provided code template and perform guided experiments
  - Data Preparation
  - Train from scratch
  - Transfer Learning
  - Data Augmentation

ALEXNET


C1
C2
C3
C4
C5
FC6
FC7
FC8
5
5
55
55
96
3
3
27
27
256
3
3
13
13
384
3
3
13
13
384
3
3
13
13
256
4096
4096
1000

CALTECH-101 (link)

- 101 Classes (+ background)
  - Airplanes
  - Faces
  - ...

GPUs and GOOGLE COLAB

- Deep Learning requires GPU acceleration
- Google Colab comes in handy (link)

CODE TEMPLATE

- Code template is available at this link
  - Save a copy on your own drive, do not directly modify the one provided
    - "File -> Save a Copy on Drive"
  - **"Edit -> Notebook Settings" to switch to GPU acceleration**
  - If you prefer to work and debug locally with your CPU, remember to set the `DEVICE` parameter to `'cpu'`
- Dataset and dataset class template available via github repository:
  - suggestions:
    - clone the provided repository for the DATA
    - make your own repository for the dataset CLASS and clone it as well

BEFORE YOU START

1. Study code and data:
    a. read the comments, understand what everything is doing
    b. make sure you know the meaning of all the variables defined since later you will be asked to change their values to achieve different results
    c. also explore the ImageFolder class code to understand how a dataset class should be implemented in PyTorch

2. Run the code
    a. training should take less than 10 minutes
    b. you have to stay connected

STEP 1 - DATA PREPARATION

WHAT YOU SHOULD DO

1. Create your own dataset class for Caltech-101, following the code provided in the GitHub repository and taking as reference the ImageFolder dataset:
    a. train-test split is already provided in the github repository: **test.txt** and **train.txt** contain the relative paths for all the images they include
    b. There is also a BACKGROUND folder that you are required to **filter out**
    c. the class should read and store only the images belonging to the corresponding split (see the `split` parameter of the `Caltech` class)

STEP 2 - TRAIN FROM SCRATCH

RECAP

- For training, we need a validation set for hyperparameter tuning and model selection
- In deep learning, it is often prohibitive to perform a K-fold Cross Validation since training takes a lot of time
- The most common procedure is the hold-out, having one single validation set both for hyperparameter tuning and model selection

WHAT YOU SHOULD DO

1. Split the training set in training and validation sets
    a. The validation set must be **half** the size of the training set
    b. Be careful to not filter out entire classes from the sets!
    c. Train and validation must be **balanced**: aim for half samples of each class in train and the other half in val

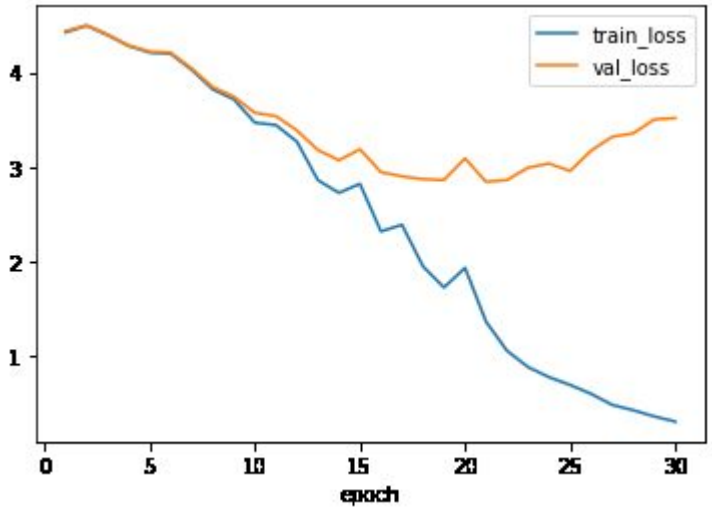
2. Implement model selection with validation
    a. After each training epoch, evaluate (= test) the model on the validation set
    b. Use only the best performing model on the validation set for test

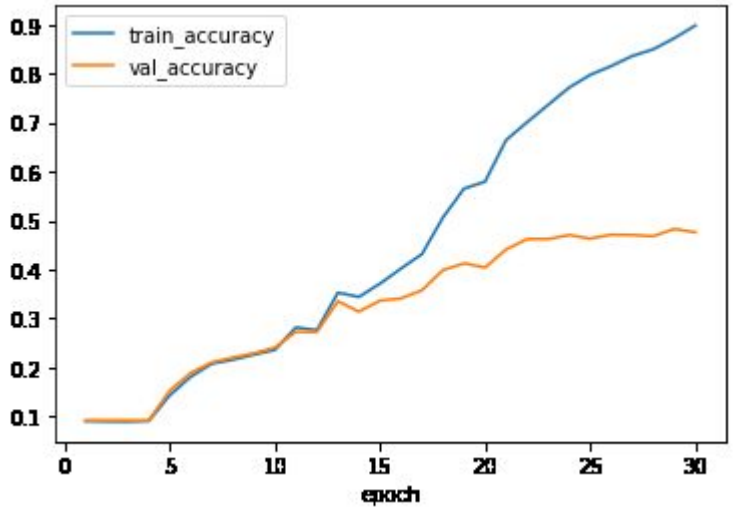
WHAT YOU SHOULD DO

The current implementation trains using SGD with momentum for 30 epochs with an initial learning rate (LR) of 0.001 and a decaying policy (STEP_SIZE) after 20 epochs.

3. Experiment with at least two different sets of hyperparameters (suggestion: experiment changing **LR** and one among: **decaying policy, optimiser, epochs**)
    a. The first hyperparameter to optimize is the **learning rate**
    b. Low learning rate = the model converges too slowly (loss decreases slowly)
    c. High learning rate = the model converges fast but accuracy is sub-optimal
    d. Too high learning rate = the model diverges (loss increases)

WHAT YOU SHOULD GET





STEP 3 - TRANSFER LEARNING

RECAP

- Deep Learning needs very large datasets to train good features
- Caltech-101 is very small dataset
- Solution 1: **Transfer Learning**
    - Use the weights learned by training on a large related dataset as a starting point for training on the small dataset
    - We can also choose to “freeze” part of the network and oly train certain layers

WHAT YOU SHOULD DO

1. Load AlexNet with weights **trained** on the ImageNet dataset
    a. (read the documentation)
2. Change the Normalize function of Data Preprocessing to **Normalize** using ImageNet's mean and standard deviation
    a. (also search for this)
3. Run experiments with at least three different sets of hyperparameters
4. Experiment by training only the fully connected layers (**freeze** other layers)
5. Experiment by training only the convolutional layers
    a. Compare all results you obtained, reasoning about what does it mean **logically** to freeze some layers

STEP 4 - DATA AUGMENTATION

RECAP

- Deep Learning needs very large datasets to train good features
- Caltech-101 is very small dataset
- Solution 2: **Data Augmentation**
    - Artificially increase the dataset size by applying some transformations at training time, preserving the label
    - Transformations applied should be the ones we are expected to see at test time, otherwise the overall accuracy could be negatively affected

WHAT YOU SHOULD DO

1. Apply at least **three different sets** of preprocessing for training images
    a. link
    b. no need to use complex transformations, such as RandomAffine: try with the simple ones, such as flipping or changing the brightness and **combine** them
    c. if test data is very similar to training data, some data augmentation policies may worsen accuracy
    d. if the loss keeps decreasing, increase training epochs or increase learning rate

2. Compare the results

STEP 5 - BEYOND ALEXNET (EXTRA)

WHAT YOU SHOULD DO

1. Try with different models (**ResNets** for example)
    a. check their requirements in terms of
        i. input image **size**
        ii. GPU memory consumed, may be needed to **reduce batch size**
        iii. some require much more time, check it before trying since Cloab's GPU may not be suited to support such an expansive training

2. Compare the results
    a. spoiler: more complex networks generally lead to better results

NEED HELP?

USEFUL LINKS

- Caltech 101 dataset
- Google Colab overview
- Colab notebook to be copied
- Github repository with dataset class to be implemented and data
- Alexnet PyTorch implementation
- PyTorch transforms documentation

CONTACT US!

- Live assistance
  - **WHEN**: LUN 16 NOV 2020, 9.30 - 11.30
  - **WHERE**: Virtual Classroom
  - **WHO**: assistant Lorenzo Bonasera
- **Slack** channel
  - 2 groups based on the surname (A-M and N-Z)
  - UNTIL **30 NOV 2020** (15 days from today). On that date a possible solution will be uploaded.
- **DO NOT WRITE EMAILS (please),** keep the discussion on Slack so that also other students can see and maybe help before we do

LET'S DO IT!