ECE 285 – MLIP – Project D
Open-ended project

Written by Charles Deledalle. Last updated on October 18, 2019.

This project is open-ended meaning that you will choose the topic of your project. This topic must be about **image processing** or **computer vision**. Your approach will have to be based on **machine learning** and must involve (at least) one **deep network architecture** and a **training set**.

Do not assume we will grade you better or be more tolerant than if you had chosen one of the other projects. The other projects are suggested to increase your chance of success, if you decide to choose an open-ended project it is at your own risk!

Note that additional information may be posted on Piazza by the Instructor or the TAs. This document is also subject to be updated. Most recent instructions will always prevail over the older ones. So look at the posting/updating dates and make sure to stay updated.

1 Proposal and approval

Please, read attentively the following instructions.

1. First of all, make sure your project does not coincide with Project A, B or C.
2. **Group creation**. When filling the form regarding group creation, please provide in addition of the required fields:
   - A short description of the task (200 words max). Just describe what is the problem you want to solve, not how you are planning to solve it. The task must be about **image processing or computer vision**. (Note that at this step we do not require you to have a solution in mind.)
   - The name and URL of the dataset you want to use. If using a Kaggle dataset, indicate instead the name of the Kaggle dataset.
   - The name and URL of one deep learning reference paper related to the task you want to solve. It does not have to be the method you want to implement. This reference is for us to judge about the feasibility of your project in the scope of a student project.
   - A second choice, among project A, B, C, in case your proposal will be rejected.
3. **Project approval**. In about one week after creating your group, you'll be notified of either:
   - **Accept**. The project seems to be challenging enough and feasible in the scope of a student project.
   - **Revise**. The task seems too easy, too difficult, not well defined, or the dataset is not suitable for that task. We will ask you to solve a simpler but related task or to use another dataset.
   - **Reject**. The project does not seem to be feasible in the scope of a student project. It requires too much training time, it is too risky, the dataset is inappropriate or there are no (labeled) dataset to learn to solve the targeted task, etc. In this case, we will assign you to your second choice (among project A, B, C).
4. After approval, You will be assigned a mentor. All questions regarding the project must be sent to your mentor, via Piazza or email, with potentially Professor Deledalle in CC. Email titles must start by `[ECE285-MLIP][Project][TeamName]`.

2 Guidelines

- You can pick any method of your choice and implement it and try to get decent results.
- You can also make use of any pre-trained models and fine-tune them.
- After you get decent results by implementing an existing technique, you can try out any novel modifications in the method to get improved results to maximize your project grade.
- Always remember the main motto is to learn, there are many freely available codes online to do this, but you should write your own code.
- Before selecting a method, please find out how long does it take for the network to train if you implement it.
- Towards the end of the quarter, the `DSMLP` cluster will become very busy, slow at times and there might be connectivity issues. Please keep these things in mind and start early and also explore other alternatives like google co-lab (12 hours free GPU) etc.
- You are encouraged to implement classes similar to ones introduced in Assignment 3 (`nntools.py`) to structure and manage your project. Make sure you use checkpoints to save your model after every epoch so as to easily resume training in case of any issues.

3 Deliverables

You will have to provide the following

1. **A 10 page final report**:
   - 10 pages **MAX** including figures, tables and bibliography.
   - One column, font size: 10 points minimum, **PDF format**.
   - Use of Latex highly recommended (e.g., NIPS template).
   - Quality of figures matter (*Graph without caption or legend is void.)*
   - The report should contain at least the following:
     - Introduction. What is the targeted task? What are the challenges?
     - Description of the method: algorithm, architecture, equations, etc.
     - Experimental setting: dataset, training parameters, validation and testing procedure (data split, evolution of loss with number of iterations etc.)
     - Results: figures, tables, comparisons, successful cases and failures.
     - Discussion: What did you learn? What were the difficulties? What could be improved?
     - Bibliography.
2. **Link to a Git repository** (such as GitHub, BitBucket, etc) containing at least:
   - Python codes (using Python 3). You can use PyTorch, TensorFlow, Keras, etc.
   - A jupyter notebook file to rerun the training (if any),

     → We will look at it but we will probably not run this code (running time is not restricted).
   - Jupyter notebook file for demonstration,

     → We will run this on UCSD DSMLP (running time 3min max).

This is a demo that must produce at least one illustration showing how well your model solved the target task. For example, if your task is classification, this notebook can just load one single testing image, load the learned model, display the image, and print the predicted class label. This notebook does not have to reproduce all experiments/illustrations of the report. This does not have to evaluate your model on a large testing set.

- As many jupyter notebook file(s) for whatever experiments (optional but recommended)

  → We will probably not run these codes, but we may (running time is not restricted).
  These notebooks can be used to reproduce any of the experiments described in the report, to evaluate your model on a large testing set, etc.
- Data: learned networks, assets, . . . (**5Gb max**)
- **README file** describing:
  - the organization of the code (all of the above), and
  - if any packages need to be `pip` installed.
  - Example:

```
Description
===========
This is project FOO developed by team BAR composed of John Doe, ...

Requirements
============
Install package 'imageio' as follow:

        $ pip install --user imageio

Code organization
=================
demo.ipynb        --  Run a demo of our code (reproduces Figure 3 of our report)
train.ipynb       --  Run the training of our model (as described in Section 2)
attack.ipynb      --  Run the adversarial attack as described in Section 3
code/backprop.py  --  Module implementing backprop
code/visu.py      --  Module for visualizing our dataset
assets/model.dat  --  Our model trained as described in Section 4
```

4 Grading and submission

The grading policy and submission procedure will be detailed later.