Adaptive Patch Selection

Code Documentation

Simon Albergel, PRIM 2020, Meero

This Documentation is about my implementation of the Adaptive Patch Selection algorithm, prerequisite to the Multi-Patch Subnet of the paper “A-Lamp: Adaptive Layout-Aware Multi-Patch Deep Convolutional Neural Network for Photo Aesthetic Assessment”, Shuang Ma, Jing Liu, Chang Wen Chen, 2017, arXiv:1704.00248.

I. Modules Dependencies

List of Modules with versions :

```
cloudpickle==1.2.2
numpy==1.17.5
opencv-contrib-python==4.1.2.30
opencv-python==4.1.2.30
pandas==0.25.3
more-itertools==8.2.0
psutil==5.4.8
scipy==1.4.1
```

```
From CPython:
os
tarfile
pdb
time
```

When specified, is used:

```
ray==0.9.0
```

We also depend on “Saliency Detection via Graph-Based Manifold Ranking” implementation by Ruan Xiang :
https://github.com/ruanxiang/mr_saliency
Additional inherited dependencies :

```
wxPython
skimage
matplotlib
```

II. Scripts

Regarding the Adaptive Patch Selection, are included :

- 2 Research Python Notebooks:
  - `Patch_Selection_Research.ipynb` ; which describes every implementation choices and serves as a detailed explanation of how the algorithm works.
  - `Patch_Selection_Prod_Ray.ipynb` ; which tries to use the Ray Module for parallel and distributed computing. Please note that due to computational limitation on our part, we were not able to test the script on a many cores cpu.

- 1 production Python script:
  - `patch_selection.py <input_dir> <output_dir>` ; in this production script, the patch implementation algorithm is implemented as a Python Class `SelectAdaptativePatch` and patch predictions are made using `predict()` method. Every images in input_dir readable by OPenCV's cv2.imread() (jpg, jpeg, png were confirmed to be working) will be processed. Every 1000 images processed, all of the results saved in a Pandas Dataframe were saved in a .pickle file.
    The DataFrame is built as follow : 2 columns "Filename" and "BBboxes". "Filename" stores a string with the name of the image, and "BBoxes" stores the upper left and lower right coordinates of the patch (Bounding Boxe) in a Numpy array of shape (4, 5) (4 coordinates x 5 patches).
    Exemple:

	BBoxes	Filename
0	[[0, 318, 224, 542], [45, 168, 269, 392], [106...	AVA_images/120997.jpg
1	[[4, 2, 228, 226], [5, 153, 229, 377], [75, 17...	AVA_images/132234.jpg
2	[[3, 405, 227, 629], [4, 8, 228, 232], [103, 1...	AVA_images/122605.jpg
3	[[28, 2, 252, 226], [172, 0, 396, 224], [161, ...	AVA_images/10767.jpg
4	[[351, 58, 575, 282], [386, 2, 610, 226], [1, ...	AVA_images/12912.jpg

- 2 utils bash scipts:
  - `tar_split.sh <DIR> <TARDEST> <OUTSIZE>` ; from user roaima on StackOverflow, splits every files in `DIR` in different `.tar` files each made up of `OUTSIZE` number of files. Tarballs are stored in `TARDEST` directory.
  - `run_patchselection_gcp.sh <BUCKET> <TAR>` ; specific script to run patch_selection.py Google Cloud Platform (See usage for more information). `TAR` and `patch_selection.py` are expected to be in `BUCKET`.

III. Usage

General Use: after making sure dependencies are met, just run :

```
./patch_selection.py input_dir/ output_dir/
```

With every images in `input_dir` and expect `.pickle` files in `output_dir`.

Usage using Google Cloud Platform:

1. The AVA Dataset (255 510 images) was split in 30 chunks of 8517 images using : `./tar_split.sh AVA_Dataset/ AVA_Dataset_split/ 8517`

   Each tarball were named from `chunkAA.tar` to `chunkBD.tar` and saved up in a public (read only) Google Cloud bucket :

   `gs://ava-dataset-split/`

   In this bucket are also saved up `run_patchselection_gcp.sh` and `patch_selection.py`

2. After instantiating the GCP virtual machine (be sure to check the “Complete access to the Cloud API” otherwise you will not be able to export the results out of the vm and into another bucket) download `run_patchselection_gcp.sh:`

   ```
   gsutil cp gs://ava-dataset-split/run_patchselection_gcp.sh
   ```

3. Run :

   ```
   ./run_patchselection_gcp.sh gs://ava-dataset-split/ chunkAA.tar
   ```

   To process `chunkAA.tar` from the `gs://ava-dataset-split/` bucket. Expected output files saved up in `output/` directory:

Nom	Taille
1000_bboxes.pickle	720,22 Ko
2000_bboxes.pickle	720,13 Ko
3000_bboxes.pickle	719,69 Ko
4000_bboxes.pickle	720,15 Ko
5000_bboxes.pickle	720,18 Ko
6000_bboxes.pickle	719,96 Ko
7000_bboxes.pickle	719,87 Ko
8000_bboxes.pickle	720,3 Ko
8517_bboxes.pickle	372,42 Ko

4. Eventually, every .pickle files where saved up in another public (read) bucket :

   ```
   gsutil cp output/*.pickle gs://output-split/chunkAA/
   ```

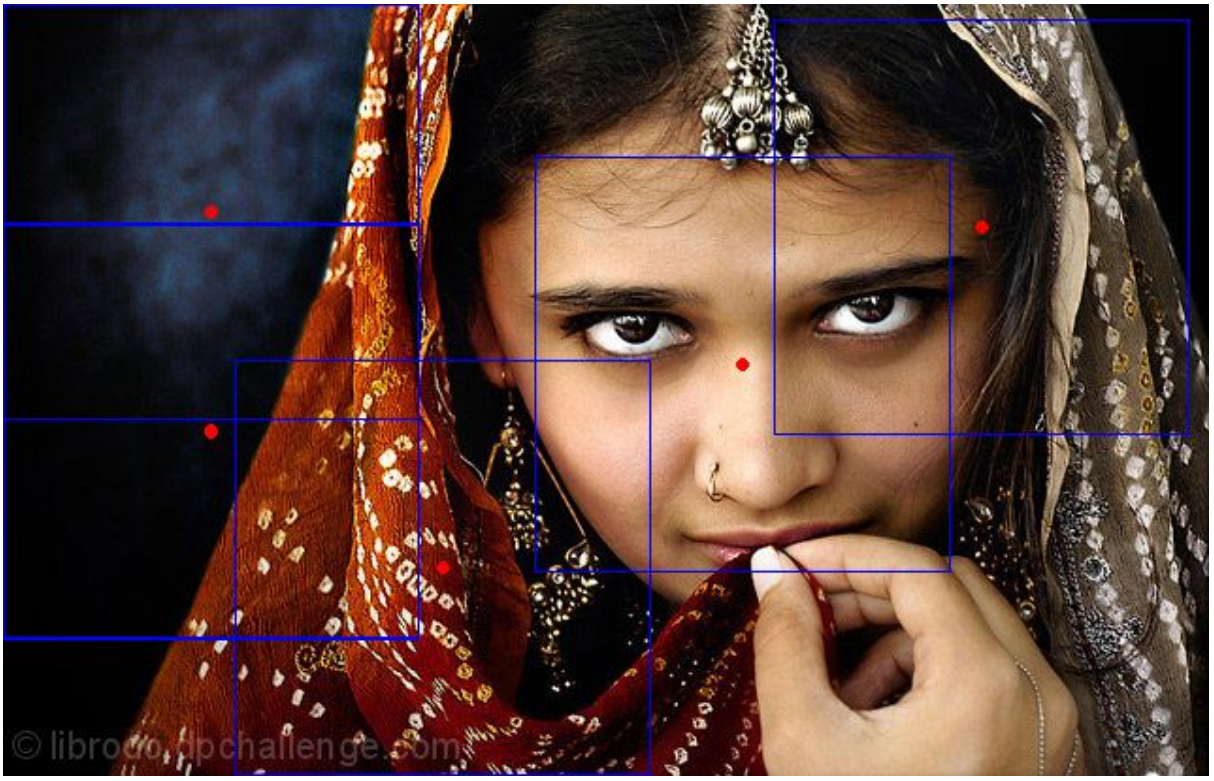

Image from AVA that serves as our reference piece with the A-Lamp paper.