Image search (assignment 2)

Emil Trenckner Jessen

20th May 2021

Contents

1 Introduction

1.1 GitHub repository

Access to the GitHub repository can be found via the link here:

https://github.com/emiltj/cds-visual-exam

The repository contains the individual assignments (including this assignment) a long with READMEs. The READMEs specify cloning the repository, setup (virtual environment installation and data collection). You may also read the information here.

1.2 Getting started

For running my script I'd recommend following the below steps in your bash-terminal (notice the bash scripts are different, depending on your OS). This functions as a setup of the virtual environment, as well as an execution of a bash script that downloads all the data to the data folders respective to the assignments.

Cloning repository and creating virtual environment

Listing 1: bash terminal - MAC/LINUX/WORKER02

```
git clone https://github.com/emiltj/cds-visual-exam.git
cd cds-visual-exam
bash ./create_vis_venv.sh
```

Listing 2: bash terminal - WINDOWS

```
git clone https://github.com/emiltj/cds-visual-exam.git
cd cds-visual-exam
bash ./create_vis_venv_win.sh
```

Retrieving the data

The data is not contained within this repository, considering the sheer size of the data. Using the provided bash script data_download.sh that I have created, the data will be downloaded from a Google Drive folder and automatically placed within the respective assignment directories.

Listing 3: bash terminal

```
bash data_download.sh
```

After cloning the repo, creating the virutal environment and retrieving the data you should be ready to go. Move to the assignment folder and read the README for further instructions (or read a long here).

2 Image search (assignment 2)

2.1 Assignment description

Using the Oxford-17 data set, compare RGB-histograms of a target image and the rest of the image corpus. Utilize the chi-square method to the calculations one-by-one. Furthermore, have the script save a data frame with two column names: "filename" and "distance".

- Make sure to round the number to 2 decimal places
- Also find the image with the shortest distance to target image and print it in the terminal.

2.2 Methods

Specifically for this assignment

For solving the project task I have made use of several functions from the cv2 library; the most important being calcHist(). It was used in this assignment for accessing information about the intensity distribution of all 3 color channels. The histograms of non-target images and the target images were then compared using the chi-square method. As a bonus, to allow for an easy inspection of the results, I included the argument "save" in the script - this gives the ability of saving both the target image and the image closest in distance, although it was not specifically required in the assignment description (see images in directory "out/", or in the section "results and discussion", below.

On a more general level (this applies to all assignments)

I have tried to as accessible and user-friendly as possible. This has been attempted by the use of:

- Smaller functions. These are intended to solve the sub-tasks of the assignment. This is meant to improve readability of the script, as well as simplifying the use of the script.
- Information prints. Information is printed to the terminal to allow the user to know what is being processed in the background.
- Argparsing. Arguments that let the user determine the behaviour and paths of the script.

2.3 Results and discussion

Finding the closest image

By looking at the images above; target image (left) and the image with the smallest histogram distance (right), we see some clear similarities in terms of

Figure 1: Target image (left) and closest image (right)

color and color intensities. The placement of the high intensities of yellow are similar, as is the color intensities of ground around it. This was however, not a given. When reducing the complexity of an image to its RGB histogram, we lose important spatial information about the colors as this method does not take placement of the different color intensities into account. Images with high pixel intensities in the bottom left for a given color will - using this method - be as similar to an image with the same high pixel intensities in the top left for the same given color (given that all other pixels are the same for both images). If we want to find images that are similar in a way that is more in line with the human interpretation of the word "similar", you might want to consider using CNN approaches instead.

Output .csv file (distances_to_image_..csv)

After running the function with the default arguments, a new output .csv file is created. It shows the distances for the images in the image corpus to the target image, and clearly shows filename.

2.4 Usage

Make sure to follow the instructions in the README.md located at the parent level of the repository, for the required installation of the virtual environment as well as the data download.

Subsequently, use the following code (when within the cds-visual-exam folder):

Listing 4: bash terminal

```
cd assignment_2
source ../cv101/bin/activate # If not already activated
python image_search.py
```

2.4.1 Optional arguments

- "-f" "–filepath", type = str, default = os.path.join("data", "*.jpg"), required = False, help= "str - path to image corpus")
- "-t" "–targetpath", type = str, default = os.path.join("data", "image_0002.jpg"), required = False, help = "str - path to target file from which to calculate distance to the other images")