Tiliter Candidate Problems

Contents

1. Purpose

The purpose of this document is to provide a set of problems to be solved by Tiliter engineering candidates as a means of skill assessment.

2. Instructions

Solve the problems as directed and set out in section 3.

2.1. Code base

- Python3.5 or 3.6
- Include requirements.txt file in each repository.
- (Optional IDE) We like PyCharm, the community version is free.

2.2. Submission

Each solution should be uploaded as a GitHub repository that will be deleted after assessment.

Provide the links to repos in an email to christopher.sampson@tiliter.com.au.

2.3. Data

The data for the problems can be found here:

https://drive.google.com/drive/folders/1xrP3TeL7IKOTi6SFtx0Mb12RoZ9akN1I?usp=sharing

2.4. Guidelines

General solutions to these problems may be found available on the internet, feel free to leverage these, however keep in mind we are looking for out-of-the-box thinking as well as neat and scalable code.

You will be asked to explain your code in full detail.

This is an opportunity to show your skills, as much as pass a test, we do not have perfect solutions in mind.

3. Video processing

3.1. Video playback

The task here is to build a script that plays videos (at least .mp4 files) at a configurable framerate, resolution and colour.

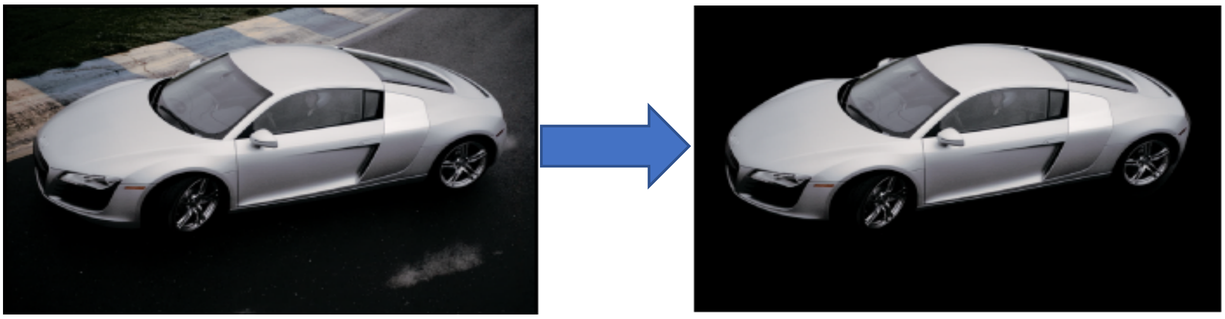
Requirements:

- Script arguments should include: video_file_path, fps, display_resolution, monochrome
- The script must be able to run as single executable .py file without command line arguments by configuring script arguments in the script itself.
- The script should be able to run in the command line by passing command line arguments.
- If monochrome is True, display video as monochrome.
- Pressing ‘p’ should pause the video. While paused, pressing ‘b’ should step back a frame.

3.2. Segmentation

The task here is to extract a masked version of the video files [video_1.mp4, video_2.mp4] where only the fresh produce is shown without the surface.

See the following frame example.



Original frame

Desired frame

Requirements:

- Solution should be constructed in a scalable way that allows different methods to be applied in the future or alternative methods to be selected.
- The script should be configurable using a path to the video and a target path. If the target path is not set the resulting video should be stored in the same folder as the source with the name <source_video_name>_processed.mp4

4. GUI and Web

4.1. Video playback GUI

Build the solution to 3.1 into a Python GUI framework (e.g. TkInter, PyQt) where “Pause” and “Reverse Frame” are push buttons.

4.2. Web server

Build the solution to 3.2 into a Django webserver with a modern front-end.

Requirements:

- Provide a file browser pop-up to select a video file to process.
- Show information processing is in progress.
- Deliver the processed file to the default local Downloads folder.

5. Machine learning

For the following problems a single script that prints a test set score is sufficient.

You must train using only the training set.

5.1. Build a digit recognition script

Using the MNIST data set in digits.zip develop a script that achieves state of the art accuracy (>98%) on the testing data.

5.2. Flower identification

Using the data set in flowers.zip, develop a script that maximizes the test set score using any available modern machine learning method.

Create test set using: *sklearn.model_selection.train_test_split(random_state=1)*