

Enunciado

Description and objectives:

The goal of PBD's project is to confront students with a typical "data science" problem to apply state-of-the-art "data science" methodologies as well as standard techniques learned in the course and on foundational courses such as linear algebra, programming, calculus or optimization. We will do this through a specific application to video analytics of cycling races that we will describe below.

The project is structured such that students are required to practice all steps of the data processing "pipeline" : Exploratory Data Analysis, Data Representation, Visualization, Modeling, Algorithm Design and Performance analysis. Furthermore, the project will show the reality of "big data processing": lack of a clear problem structure, heterogeneity of the data, huge dimension and unreliable data (outliers).

In short, the project does not have a single clear solution or approach. Students are encouraged to explore and autonomously test multiple approaches/solutions though the context is a bit constrained to make it realizable. The application scenario is extracted from the Italian bicycle competition.

The Giro D'Italia: Analysis of RAI's Media Coverage

Large outdoor events like the bicycle race "Giro D'Italia" follow complex but very well planned "scripts" by the producer/director of the coverage.



Fig1: skeleton extracted by openpose

In particular the director is managing the viewpoints originated from a multitude of video sources such as motorbikes, fixed cameras, drones or helicopters and at each time instant, he/she selects the best video feed or the most meaningful to the viewer.

Objectives and data

The original data is comprised of a set of videos taped from RAI's media coverage. These videos were processed and the available data is the following:

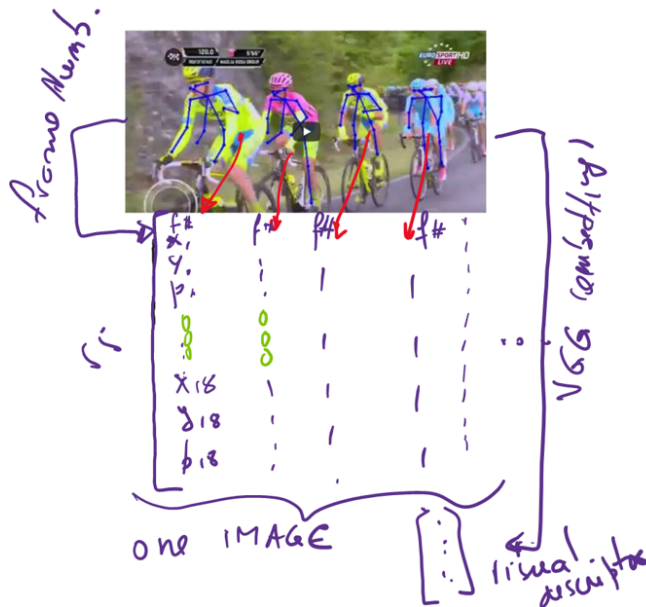
- **Visual embeddings:** A high dimensional vector (1024) for each image of the video. This is a representation provided by a Neuronal Network that encodes each image. This is to say that images can be compared in terms of similarity using these vectors. Details will be provided in the practical classes.
- **Skeletons :** Each image was processed and skeletons of visible people were extracted using the "open pose" algorithm. As fig 1 shows, depending on the number of visible persons and the algorithm reliability, a variable number of skeletons are extracted. A skeleton is a set of 25 2D coordinates encoding the image location of each joint. Details will be provided and the original representation is here <https://github.com/CMU-Perceptual-Computing-Lab/openpose>.
- **The original images** in case you want to extract further information from them.

In short, the minimum set of data is represented by a set of matrices as described below.



<https://youtu.be/DEZ8yw78Y94>

Keypoints of detected poses.



The main data is comprised of a $55 \times \text{\#detectedposes}$ matrix where the first element in each column is the frame number and the remaining elements are triplets (x_i, y_i, p_i) with the x, y coordinate of each point and the score ("probability") of the detection.

The goal of the project is to segment the whole video by "type of shot" which is indexed by the pose of the cyclists and the contents of the image. The figure below illustrates the main idea



There will be three levels of difficulty which involve three different types of data.

- 1 - Complete data: the skeleton data is complete, that is, the whole set of skeleton points is available for each detected person.
- 2 - Incomplete data: skeleton detectors fail and sometimes only part of a body is visible therefore it happens that some points are missing. In fact, the previous dataset (completed) was created from this one using one particular data completion algorithm !
- 3 - Extra data: any data extracted from the images or crawled from the internet or any other source. This is left to each group initiative, students are allowed (and encouraged) to extract further data.

The project will have 3 main parts that will be detailed further as the course develops along the period (it's the first time it runs!). :

- Data Analysis and Preparation. During the practical classes and documented in the presentation, groups must do and show data analysis performed in one dataset or highlight specificities among the different datasets (if used more than one).
- Code that produces the output and that all groups will submit. In due time we will specify what are the inputs and outputs of the final submission.
- Analysis of the results. Also mostly documented in the final presentation.

The project illustrates a typical application with “big data” where the goal involves many unsupervised learning tasks and where the performance is not evaluated by one single parameter. For example, we resist the temptation of providing an “objective” performance indicator such as accuracy in estimating missing points (which is not at all “objective” since there is no ground truth!).

However we can set the following list of “general” that each figures of merit that each project should exhibit:

- Segment the video in parts that show a similar content, indexed by the “type” of pose data. The visual contents (scenario) is a secondary factor.
- The main clusters and the outliers (of the pose data) must be identified and it’s one of the outputs
- Several statistics (to be defined later) must be supplied for each video.
- We reward rigour, depth and creativity ... in this order !

Implementation

Students are encouraged to use python and any library available, but projects can be implemented in any language (matlab, R, C++, Julia ...). Except for matlab, all projects must run in a “typical” docker image that we will supply.

Remember: To sign for this course students are required to master some programming language and have all the skills necessary to implement algorithms.

DOCKERs: for the first time we will introduce and encourage using docker containers to deploy the final code and use existing tools. In fact, part of the data producing tools (skeleton detector, embedding encoder, object detector) are already available as docker containers. Since it is a first time for everybody we will not make it mandatory (yet) but this will be the preferred project delivery tool.

So, you can start by installing docker on your computer and we will help in the first practical class where the technology will be introduced. See <http://www.docker.com>

The skeletons’ keypoints (open pose output) are described below


[Openpose](#)

Evaluation

The project suggests the application of a “canonical” approach using textbook techniques and following the several steps of data preparation, data analysis and unsupervised learning. Applying this methodology and the adequate techniques will certainly grant a good grade but will not necessarily achieve excellent grades.

Resources

Key articles

 https://people.orie.cornell.edu/mru8/doc/udell16_glrn.pdf

Several computational resources and code snippets will be released in PB classes and posted on the course website.

Try the python package on “low rank models” based on Madeleine Udell’s paper :
<https://github.com/udellgroup/pyglrm>

Docker containers

Datasets :

Datasets will be shared through googledrive: The link is either on fenix or will be sent by email.

All data is encoded in .mat files. In python you must install numpy and use the savemat package to read the data. Of course any other universal format (csv for example) may or will be used if necessary.

<https://youtu.be/Cpz3NzrsjpM>

This is the full video of the EUROSPTALL dataset

To explore - <https://www.di.ens.fr/willow/research/surreal/data/>

gRPC docker components: existing components and how to build one.

We will make available a set of tools that can help you extract data and enrich your model. Also, container technologies are here to stay and are a fundamental tool to develop and deploy software.

[Tutorial on how to create your gRPC-enabled component and its own Docker container, for building pipelines \(compatible with AI4Europe platform\)_\(1\).](#)


<https://github.com/andrejfsantos4/DIY-gRPCcomponent>

[DuarteMRAves/open-pose-grpc-service: Deep Pose Estimation implemented using Tensorflow with Custom Architectures for fast inference. \(1\).](#)

<https://github.com/DuarteMRAves/open-pose-grpc-service>

open-pose-grpc-service/test_image.py at master · DuarteMRAIves/open-pose-grpc-service

This file contains bidirectional Unicode text that may be interpreted or compiled differently than what appears below. To review, open the file in an editor that reveals hidden Unicode characters. Learn more about bidirectional Unicode characters You can't perform that action at this time. You signed in

 https://github.com/DuarteMRAIves/open-pose-grpc-service/blob/master/tests/test_image.py

DuarteMRAIves/**open-pose-grpc-service**

Deep Pose Estimation implemented using TensorFlow with Custom Architectures for fast inference.

Rx 0 Contributors 0 Issues ☆ 1 Star ▼ 1 Fork

- DockerHub - andrejfsantos/img_embedding:latest E também no github (com instruções)
- <https://github.com/andrejfsantos4/ImgEmbeddingComponent>

DuarteMRAIves/yolov5-grpc (1).


<https://github.com/DuarteMRAIves/yolov5-grpc>

DuarteMRAIves/Pipeline-Orchestrator (1).

<https://github.com/DuarteMRAIves/Pipeline-Orchestrator>

Quick start

This guide gets you started with gRPC in Python with a simple working example. This guide gets you started with gRPC in Python with a simple working example.

 <https://www.grpc.io/docs/languages/python/quickstart/>



<https://www.grpc.io/docs/languages/python/quickstart/>