Intelligent Consumer Technologies

Dr. Luigi Celona
a.a. 2024/2025

Tools and Speech processing

Topics: Instructor introduction, Tools.

Learning Objectives

- Learn how to choose the most appropriate Python tool for different purposes
- Understand the usefulness of Google Colab
- Realize the difference among CPU, GPU and TPU

WhoIAm

focus

Dr. Luigi Celona → *Assisted Exercises*

e-mail: luigi.celona@unimib.it → email subject label: **[ICT]**

DISCo (Department of Informatics, Systems and Communication)

University of Milan-Bicocca

Viale Sarca 336, Building U14

Room 1048, tel: +390264487871

My academic path

2011 → Bachelor's Degree in Computer Science (@UniMe)

2014 → Master's Degree in Computer Science (@UniMib)

2017 → PhD in Computer Science (@UniMib)

2018 → PostDoc in Computer Science (@UniMiB)

2023→ Assistant Professor of Computer Science (@UniMiB)

Follow my research activities on: ☞

http://www.ivl.disco.unimib.it/people/luigi-celona/

@luigi_celona

https://scholar.google.it/citations?user=F9vDCKAAAAAJ

http://luigicelona.it/

https://www.linkedin.com/in/luigicelona/

WhoIAm

focus

Computer Vision

Quality assessment

Image aesthetics

ArchedEyebrows — BigNose — BlondHair — HeavyMakeup — HighCheekbones — MouthSlightlyOpen — NarrowEyes — NoBeard — PointyNose — RosyCheeks — Smiling — WavyHair — WearingLipstick — WearingNecklace

Face analysis

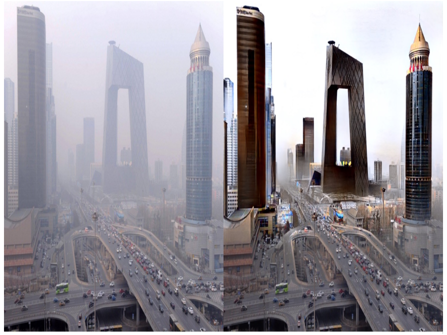
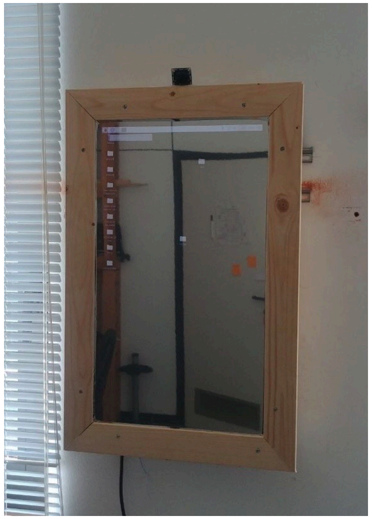


Image enhancement

Human Behavior Monitoring

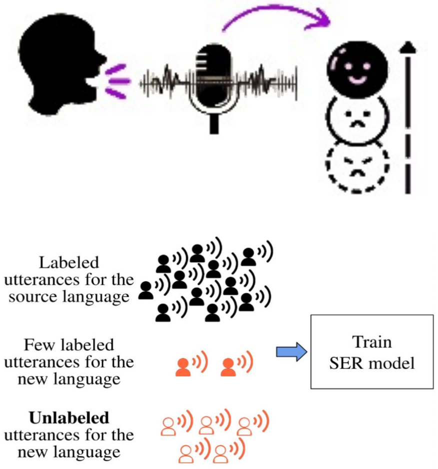


Smart magic mirror

Car driver monitoring

Speaker analysis

Speaker recognition

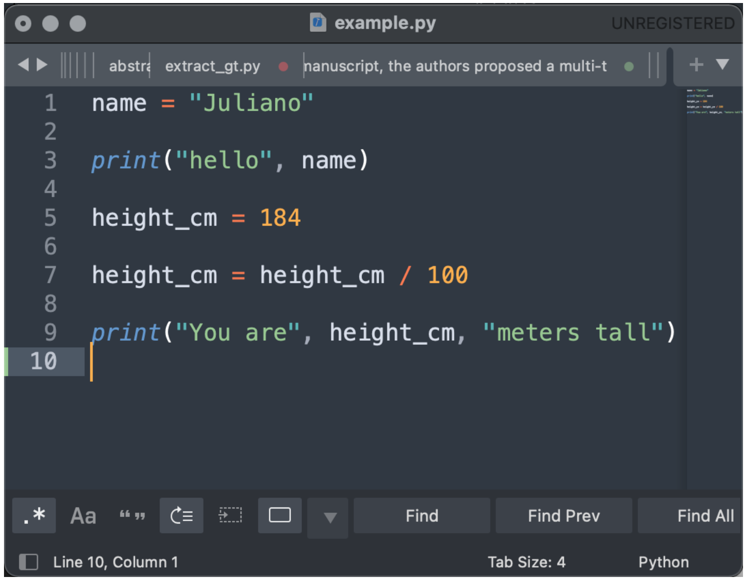

Labeled utterances for the source language
Few labeled utterances for the new language
Unlabeled utterances for the new language
Train SER model

SSL Emotion recognition

Tools

Python scripts vs. Jupyter notebooks

- A **Python script** is a plain text file ending with the *.py* extension
  - Can be edited with text editors such as *nano*, *vi*, notepad, or sublime
  - Integrated Development Environment (IDE) can be also used that include text editor, a debugger, and a terminal window (e.g., VisualStudioCode and PyCharm)



```python
name = "Juliano"

print("hello", name)

height_cm = 184

height_cm = height_cm / 100

print("You are", height_cm, "meters tall")

```

- Python scripts are **executed linearly**, in a top-down fashion
  - To run the script from the terminal, you would type **`python example.py`**
  - In an IDE, there is probably a button in the interface to directly run the script

Tools

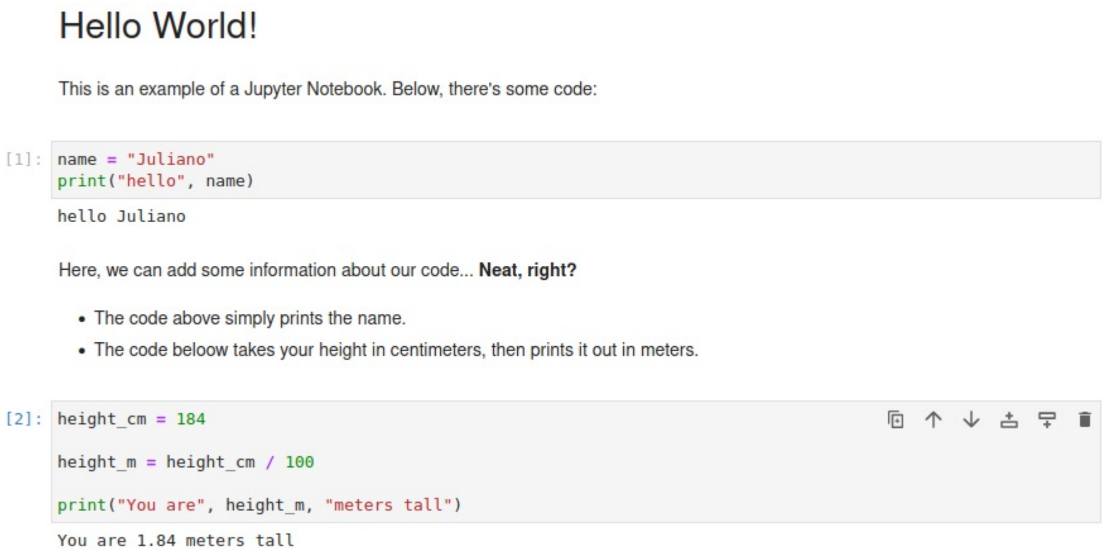
Python scripts vs. Jupyter notebooks

- A **Jupyter notebook** consists of multiple *cells*
  - Each cell can contain either a block of Python code or plain text
  - Bits of code can be surround with useful information, like explanations, links, and images

- Jupyter notebooks are **executed in a non-linear fashion**
  - Code blocks can be executed in an arbitrary order

- Jupyter notebooks are stored into notebook files with extension *.ipynb*

- A Jupyter server let to interact with and edit a Jupyter notebook using a **web browser** like Chrome or Firefox



Hello World!

This is an example of a Jupyter Notebook. Below, there's some code:

[1]:
```
name = "Juliano"
print("hello", name)
```

```
hello Juliano
```

Here, we can add some information about our code... **Neat, right?**

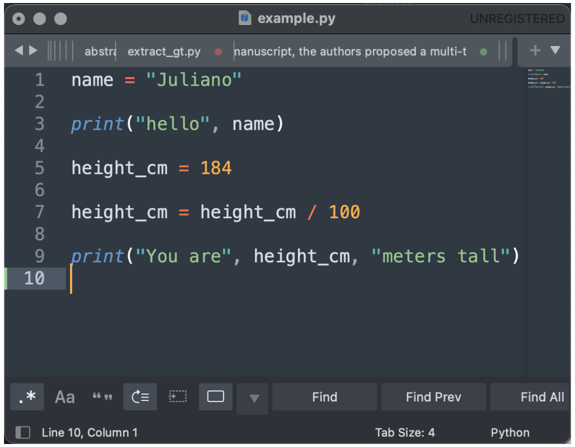
- The code above simply prints the name.
- The code beloow takes your height in centimeters, then prints it out in meters.

[2]:
```
height_cm = 184

height_m = height_cm / 100

print("You are", height_m, "meters tall")
```

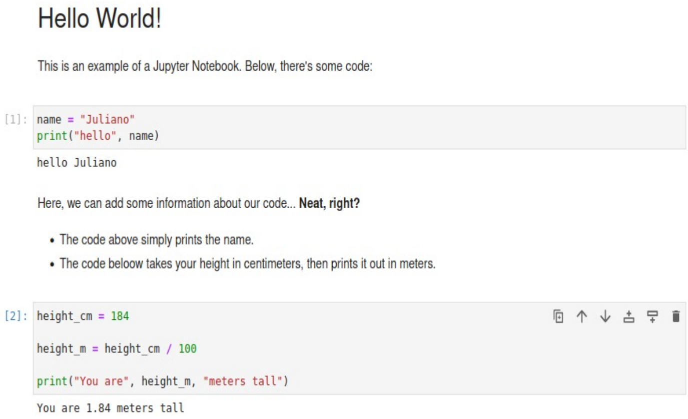
```
You are 1.84 meters tall
```

Tools

Python scripts vs. Jupyter notebooks



```python
name = "Juliano"

print("hello", name)

height_cm = 184

height_cm = height_cm / 100

print("You are", height_cm, "meters tall")

```



Hello World!

This is an example of a Jupyter Notebook. Below, there's some code:

```
[1]: name = "Juliano"
     print("hello", name)
```

```
hello Juliano
```

Here, we can add some information about our code... **Neat, right?**

- The code above simply prints the name.
- The code beloow takes your height in centimeters, then prints it out in meters.

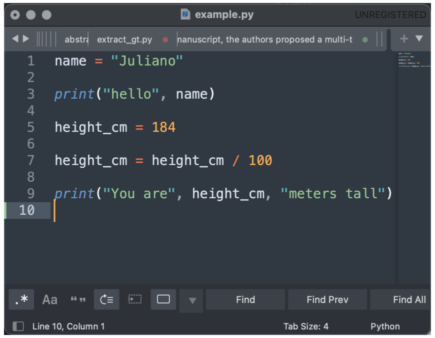
```
[2]: height_cm = 184

     height_m = height_cm / 100

     print("You are", height_m, "meters tall")
```

```
You are 1.84 meters tall
```

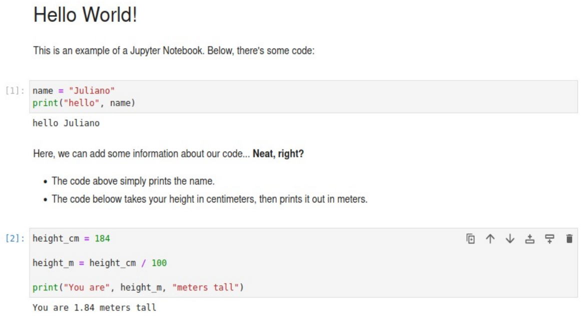
Which is your favorite tool? Python scripts or Jupyter notebooks?

Tools

Python scripts vs. Jupyter notebooks



```python
name = "Juliano"

print("hello", name)

height_cm = 184

height_cm = height_cm / 100

print("You are", height_cm, "meters tall")

```



Hello World!

This is an example of a Jupyter Notebook. Below, there's some code:

```
[1]: name = "Juliano"
     print("hello", name)

hello Juliano
```

Here, we can add some information about our code... **Neat, right?**

- The code above simply prints the name.
- The code beloow takes your height in centimeters, then prints it out in meters.

```
[2]: height_cm = 184

     height_m = height_cm / 100

     print("You are", height_m, "meters tall")

You are 1.84 meters tall
```

Pros:

- Reliable and the most common way to write Python code.
- Minimal setup is required (i.e., only text editor).
- Top-down execution makes it less confusing to debug and reason through the code.
- Support modularity. Variables and functions can be imported from another script.

Cons:

- Must be re-executed to test any changes to the code.
- Are plain text files. Formatted text or figures cannot be added to them.
- By default, no output is saved anywhere. The script must be re-executed to see messages, outputs, and results.

Pros:

- Code blocks can be surrounded by helpful notes, figures, and links.
- Provide nonlinear execution. Code cells can be run independently from one another.
- Output (such as messages, plots, and dataframes) appear automatically under each cell, and look great out-of-the-box.
- Are good for prototyping, data analyses and sharing results with colleagues.

Cons:

- Require installing the jupyter-notebook package into the Python environment.
- Nonlinear execution can make debugging confusing, especially if you lose track of which cells you have executed or not.
- Sharing code or data is not straightforward.

Google Colaboratory

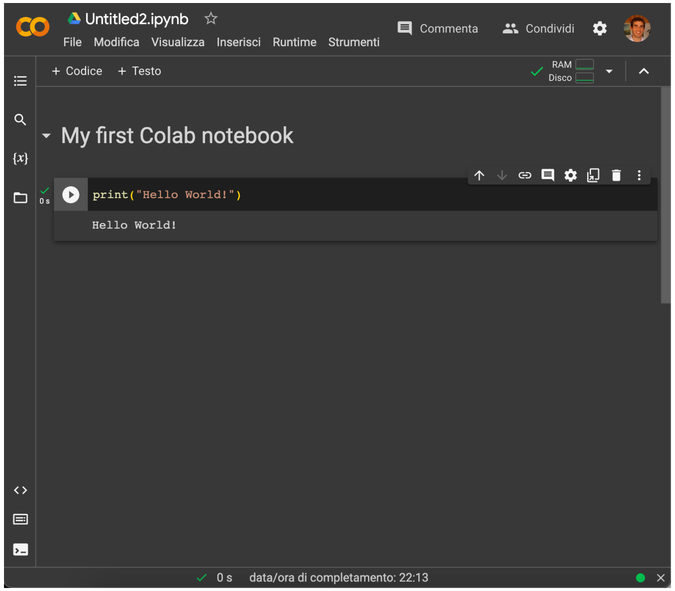
What is Google Colab?

- **Google Colab(oratory)** is a hosted Jupyter notebook service. Meaning you can run your Jupyter Notebook online with no setup and access free computing resources

- With Colab it is possible to **access powerful computing resources** without the need to purchase expensive hardware or set up complex software environments

- Colab notebooks **are stored on Google Drive**, which makes it easy to share your work with others and collaborate in real-time

- Colab notebooks are a great way to explore data, build machine learning models, and document your findings



My first Colab notebook

```
print("Hello World!")
```

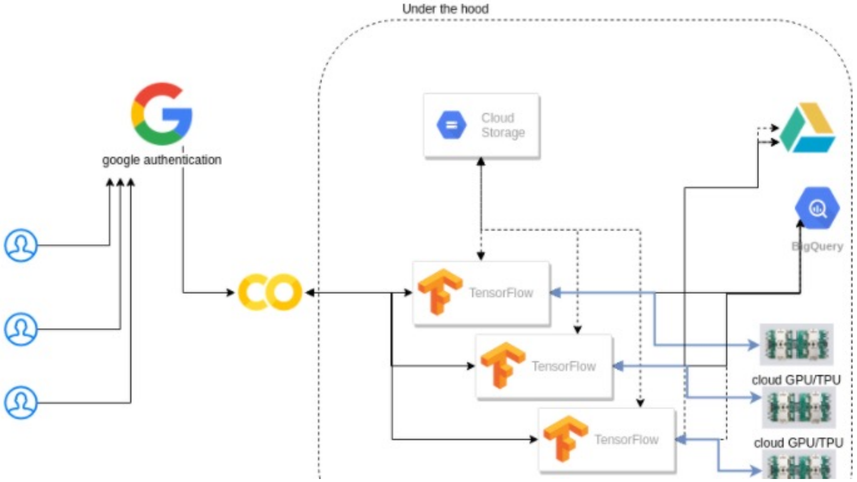
```
Hello World!
```

✓ 0 s data/ora di completamento: 22:13

Google Colaboratory

Characteristics

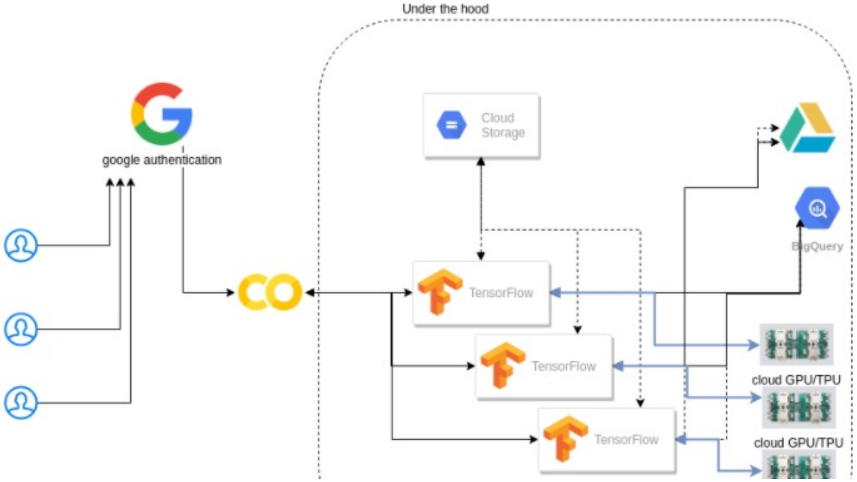
- In Google Colab there are **numerous Python libraries**, including many from Data Science such as Keras and Tensorflow
- Access to several resources, **such as GPUs and TPUs**, to give important computational boosts to our work, for example in implementing neural networks with Tensorflow
- Importing data into Google Colab is very easy
  - Manually uploading the data
  - Connectors provided to access datasets in our Google Drive
  - Integrate with other cloud services such as Big Query

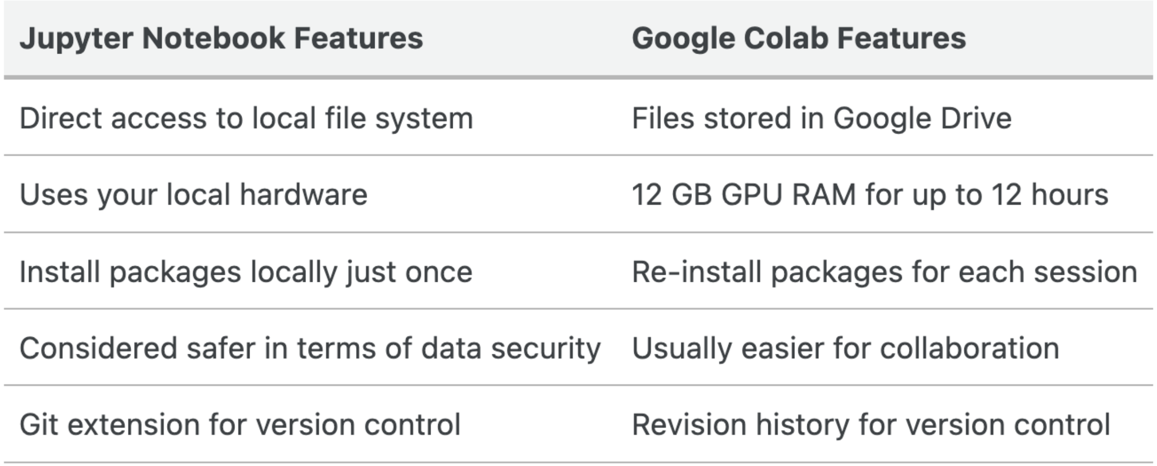


Google Colaboratory

Characteristics

- **Resources are limited and vary** depending on fluctuations in demand
- Higher performance machines or more powerful GPUs and TPUs can be accessed using the **Pro version**
- Google Colab has a Revision history option to help with **version control**





Jupyter Notebook Features	Google Colab Features
Direct access to local file system	Files stored in Google Drive
Uses your local hardware	12 GB GPU RAM for up to 12 hours
Install packages locally just once	Re-install packages for each session
Considered safer in terms of data security	Usually easier for collaboration
Git extension for version control	Revision history for version control

Processing units

CPU vs. GPU vs. TPU

- **CPU:** Central Processing Unit. Manage all the functions of a computer.
- **GPU:** Graphical Processing Unit. Enhance the graphical performance of the computer.
- **TPU:** Tensor Processing Unit. Custom build ASIC (Application Specific Integrated Circuit) to accelerate TensorFlow projects

CPU	GPU	TPU
intel SAMSUNG AMD Qualcomm hp NVIDIA VIA IBM Atmel	NVIDIA AMD BROADCOM Imagination	Google Coral HAILO

Processing units

CPU vs. GPU vs. TPU

- **CPU:** It is the primary hardware of the computer, the «brain of the computer» that **executes the instruction for computer programs**. All the basic arithmetic, logic, controlling, and the CPU handles input/output functions of the program.
- **GPU:** It visually renders the **graphical user interface**. It allows speeding up and parallelization of simple matrix calculations.
- **TPU:** It is an **application-specific integrated circuit**, to accelerate the AI calculations and algorithm. Google develops it specifically for neural network machine learning for the TensorFlow software

CPU	GPU	TPU
Several core	Thousands of Cores	Matrix based workload
Low latency	High data throughput	High latency
Serial processing	Massive parallel computing	High data throughput
Limited simultaneous operations	Limited multitasking	Suited for large batch sizes
Large memory capacity	Low memory	Complex neural network models

Processing units

CPU vs. GPU vs. TPU

- **Which is better TPU or GPU?**
  - A single GPU can process thousands of tasks at once, but GPUs are typically less efficient in the way they work with neural networks than a TPU
  - TPUs are more specialized for machine learning calculations and require more traffic to learn at first, but after that, they are more impactful with less power consumption
- **Is TPU faster than CPU?**
  - TPUs are 3x faster than CPUs
  - TPUs are 3x slower than GPUs for performing a small number of predictions
- **How much faster is TPU vs. GPU?**
  - The TPU is 15 to 30 times faster than current GPUs.

QUESTIONS?