

# Speech Recognition and Synthesis - Lab setup

This document contains instructions for setting your laptop for the labs of the Speech Recognition and Synthesis (ASR-TTS) course, which rely on Jupyter notebooks.

- Section 1 guides you throughout a “manual” install procedure, so that you can work with notebooks directly in your web browser.
- If you prefer to work with an IDE, Section 2 guides you to install and setup VS Code.

**Note for Windows users** Working with VS Code is **strongly recommended**. Nonetheless, it is possible to pursue with the manual install, by first setting up the Windows Subsystem for Linux (see section 3) before moving to the steps described in section 1.

**Installing Python** Make sure that Python 3 is installed. If you use Linux or Mac OS, this should not be necessary (Ubuntu 22 ships with Python 3.10 by default). On Windows, the easiest way to install it is via the Microsoft app store.

**Working folder** Create a working folder, e.g., `ASR_TTS_Labs`. There, you can place the material downloaded from the Arche platform of the course(s) (e.g., unzip the `dataset.zip` and `setup.zip` files). This is also where you will create your virtual environment and put all the lab sub-folders / notebooks, thus it will look like:

```
ASR_TTS_Labs
├── .venv
├── dataset
│   ├── speech
│   └── transcription
├── lab1
│   └── lab1_asr_intro.ipynb
├── lab2
├── ...
└── setup
```

## 1 Manual setup

We assume to work under Ubuntu 22.04, but this procedure should work up to some minor adjustments under other Ubuntu / Linux versions or Mac OS.

## Prerequisites

Open a terminal and make sure that everything is up-to-date (in particular, the packages and virtual virtual environment<sup>1</sup> managers) by running the following commands:

```
sudo apt-get update
sudo apt-get upgrade
sudo apt install python3-pip
pip install --upgrade pip
sudo apt install python3-venv
```

## Virtual environment

In your working folder, create a virtual environment via:

```
python3 -m venv .venv
```

and activate it by running:

```
source .venv/bin/activate
```

When you're done, you can close it by simply running `deactivate`.

## Packages

For simplicity, the list of all needed Python packages are assembled in a single file. Activate your virtual environment and run the following commands to install everything (first clear the cache to avoid version conflict):

```
pip cache purge
pip install -r setup/requirements.txt
```

## Crash test

You can check if everything went well as follows.

1. Go to your working folder and activate your virtual environment in a terminal.
2. Open a notebook in your browser by running `jupyter-lab` or `jupyter-notebook`.
3. Open the provided test file `test_setup.ipynb`.
4. Execute the cell. If there are no error message, then you're good to go.

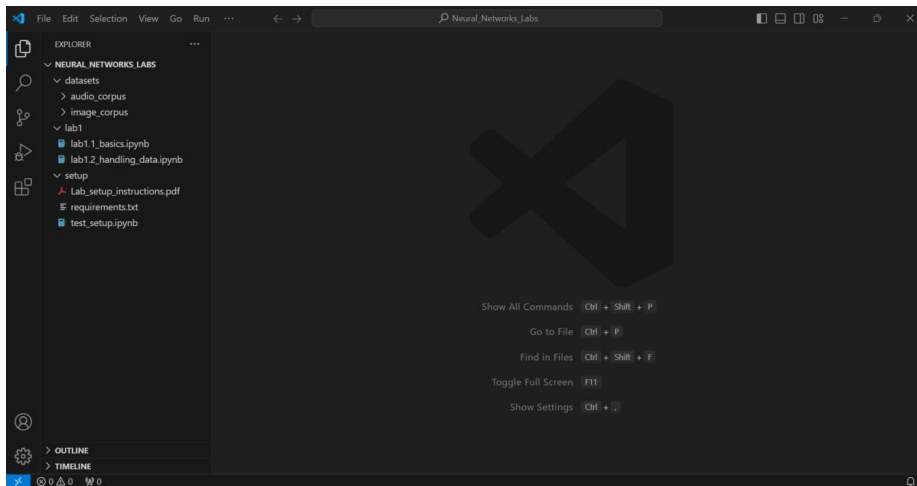
## 2 VS Code setup

### Install and config of VS Code

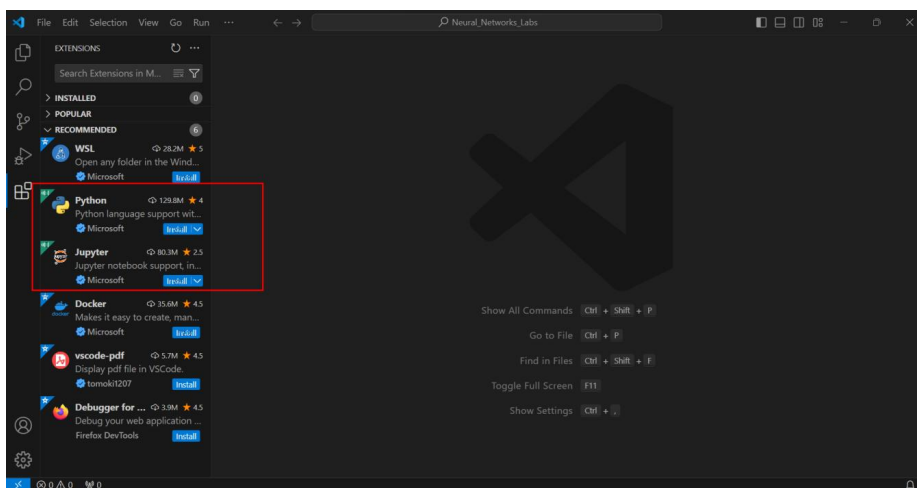
First, install VS Code by following the instructions here depending on your OS / platform (here we use Windows 11). When installed, open VS Code and select your working folder; your screen should look like this (here we use screenshot from the `Neural_Networks_Labs` folder, but the process is the same).

---

<sup>1</sup>Virtual environments allow you to manage package requirements for different projects specifically, so that there are no conflicting package versions between projects.

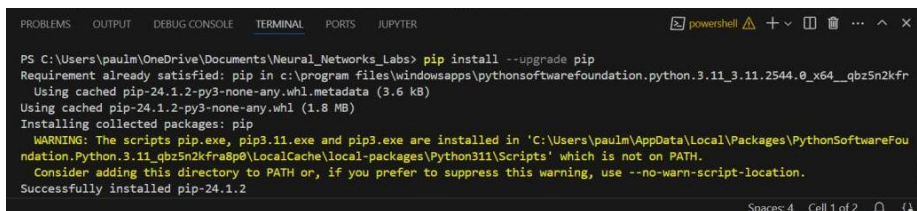


Go to the “Extensions” tab (on the left panel), and install the “Python” and “Jupyter” extensions. These should appear among the recommended extensions, as VS Code detects there is a notebook inside your working folder. Otherwise, search for them manually.



Open a terminal in VS Code (“View → Terminal”) and run the following command to update the “pip” package manager:

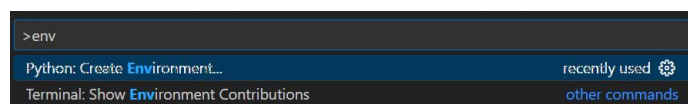
```
pip install --upgrade pip
```



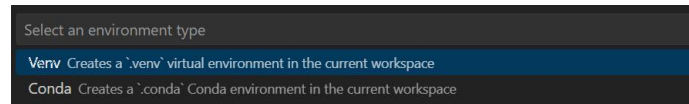
## Environment and packages

Open the command palette (the search bar located at the top of the window), and

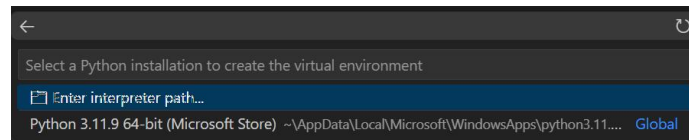
- Simply type 'env' to find the command that initiates the virtual environment creation:



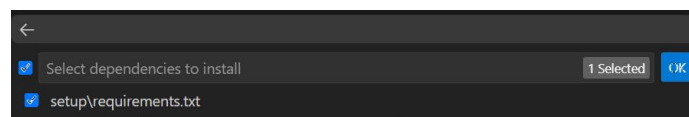
- Select the first option (“Create a .venv virtual environment [...]”):



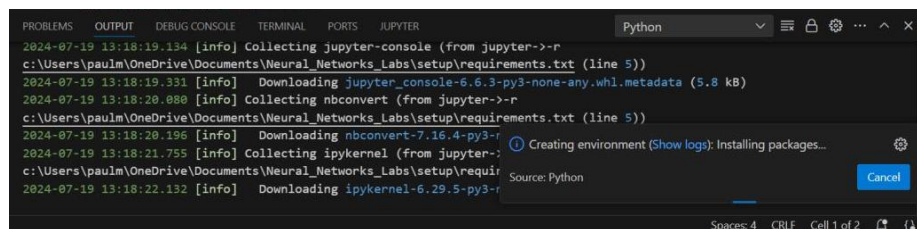
- Choose the installed Python distribution (here, Python 3.11.9) as a reference:



- Select the “setup/requirements.txt” file to install all the needed packages.



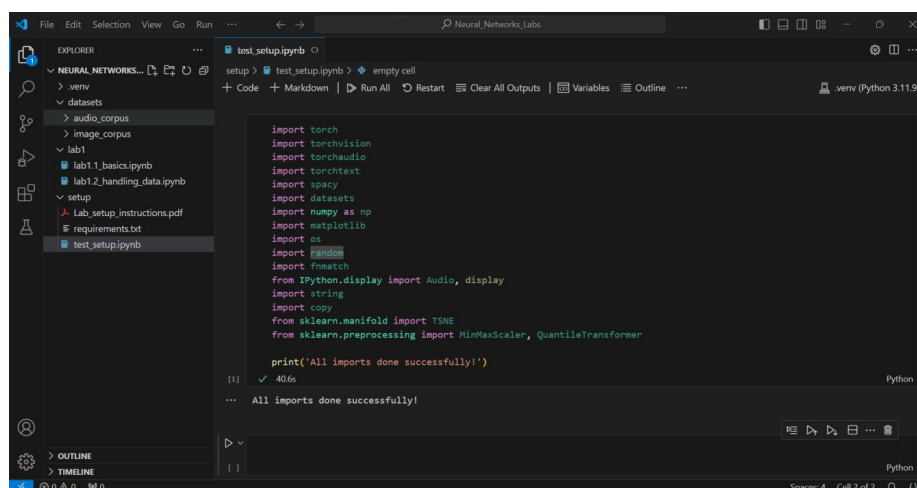
The installation then starts, which might take a while.



## Crash test

You can check if everything went well by opening the notebook `test_setup.ipynb`, and executing the cell (shit+enter).

Note that VS Code might ask you to select the interpreter/kernel, in which case you can select the `.venv` you have just installed. The kernel may take some time to launch and run the first time, but then it is faster.



### 3 Windows Subsystem for Linux

The Windows Subsystem for Linux (WSL) is a tool that allows to work under Linux environments on a Windows PC.

**Install** The simplest method to install it along with all dependencies is to open a Powershell and enter:

```
wsl --install
```

Reboot your computer to finalize installation, run WSL, and setup your Ubuntu user name and password (by default, WSL comes with Ubuntu, but you can change it if you like).

**Working folder** Instead of placing your working folder anywhere in the classical Windows folder structure (“Desktop”, “Downloads”, “Documents”...), we recommend to put it in the specific Linux folder structure that was created upon installing WSL. Indeed, it is faster when working with Python, and it is easier to navigate there in the WSL terminal.

To do so, click on “Linux” in your file explorer (at the bottom of the left panel), then “Ubuntu/home/<UbuntuUserName>” and place your working folder here. You can then navigate to this location in the WSL terminal via:

```
cd /home/<UbuntuUserName>/
```

