NPGR035 Python Setup

Mgr. Matúš Goliaš, Doc. RNDr. Elena Šikudová PhD.

October 4, 2023

This document describes Python environment used in the NPGR035 (Machine Learning in Computer Vision) practicals and necessary libraries as well as instructions on how to set it all up. Most of what is written here is very basic and (with the exception of package installation) can be skipped but it is useful to have a common setup when working with Python. We are using standard Python installation with `venv` and `pip`, so command translation to Anaconda or other Python management application is left to the student.

University computers

University computers should have recent python installation with all required packages. Therefore, it is not necessary to do anything to start working on university computers.

Python version

The assignments for this course will be tested in Python 3.10.11 and 3.11.5. Version 3.10 is the last one where Tensorflow natively supports GPU computation on Windows and 3.11.5 is the newest at the time of writing.

Necessary packages

We will use the following list of packages:

- `numpy`
- `scipy`
- `matplotlib`
- `scikit-learn`
- `scikit-image`
- `tensorflow`

A requirements file for `pip` installation with the name `requirements_general.txt` can be found next to this document. The file contains just the list of package names. Versions, as long as they are not too old, shouldn't matter.

Virtual environment setup

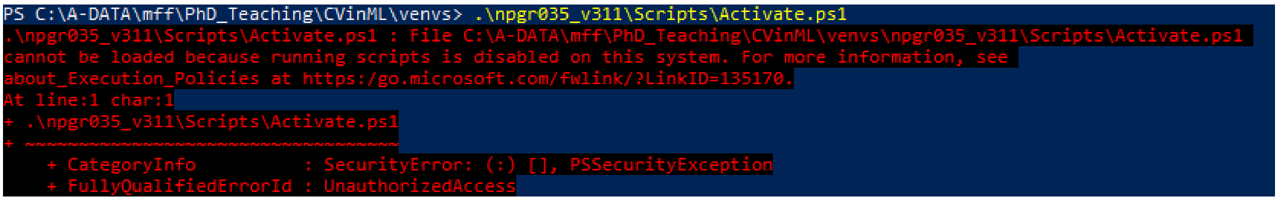
The best way of working on specific Python projects is to create a separate virtual environment which ensures code compatibility between the project members. Therefore, we provide a short tutorial on setting up and using virtual environment for this course. We use standard Python installation and command line to make it simple. Detailed information can be found at https://docs.python.org/3/tutorial/venv.html.

First, we need to navigate to some directory where we want to set up the environment. The location doesn't matter but it makes sense to put it close to the course files. Then, we create the virtual environment using `venv` utility, activate the environment and install the necessary libraries. We can deactivate active virtual environment simply by writing `deactivate` in the command line.

Windows provides two options of system interaction through command line: PowerShell and Command prompt. We recommend using PowerShell but the following instructions include both options for compatibility.

- Windows PowerShell
  - `cd path\to\some\directory`
  - `python -m venv {name of the venv, e.g. npgr035env}`
  - ⋆ `set-executionpolicy RemoteSigned -Scope Process`
  - `.\{name of the venv, e.g. npgr035env}\Scripts\Activate.ps1`
  - `python -m pip install --upgrade pip`
  - `python -m pip install -r requirements_general.txt`
  - `deactivate` *for deactivation*
- Windows CMD
  - `cd path\to\some\directory`
  - `python -m venv {name of the venv, e.g. npgr035env}`
  - `.\{name of the venv, e.g. npgr035env}\Scripts\activate.bat`
  - `python -m pip install --upgrade pip`
  - `python -m pip install -r requirements_general.txt`
  - `deactivate` *for deactivation*

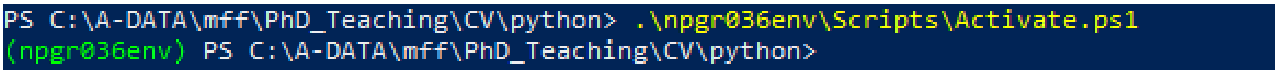
It is possible to run into the error shown in Figure 1 during virtual environment activation in PowerShell. If this happens, we need to change the PowerShell execution policy to something more permissive, see ⋆. This change will have an impact only on the current PowerShell window due to its scope limitation and it will have to be used again after closing the window. It is possible to set it globally or for a single user as well but limiting the scope is probably better.



```
PS C:\A-DATA\mff\PhD_Teaching\CVinML\venvs> .\npgr035_v311\Scripts\Activate.ps1
.\npgr035_v311\Scripts\Activate.ps1 : File C:\A-DATA\mff\PhD_Teaching\CVinML\venvs\npgr035_v311\Scripts\Activate.ps1
cannot be loaded because running scripts is disabled on this system. For more information, see
about_Execution_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\npgr035_v311\Scripts\Activate.ps1
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : SecurityError: (:) [], PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
```

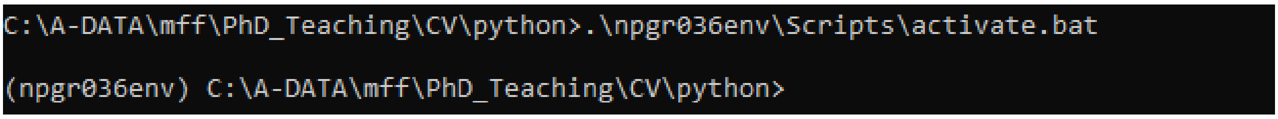
Figure 1: PowerShell error during virtual environment activation.

Successful activation will change the command line text to include the name of the environment, see Figures 2 and 3. Any python commands issued after the activation will be executed using the currently active environment. Virtual environments can be activated in IDEs as well but it is usually



```
PS C:\A-DATA\mff\PhD_Teaching\CV\python> .\npgr036env\Scripts\Activate.ps1
(npgr036env) PS C:\A-DATA\mff\PhD_Teaching\CV\python>
```

Figure 2: Virtual environment activation in Windows PowerShell.



```
C:\A-DATA\mff\PhD_Teaching\CV\python>.\npgr036env\Scripts\activate.bat

(npgr036env) C:\A-DATA\mff\PhD_Teaching\CV\python>
```

Figure 3: Virtual environment activation in Windows Command Prompt.

required to point the IDE towards the newly created python executable within the environment directory.

Unix and MacOS installation is very similar and it can be used on the university computers to work with packages which are not installed in the system installation of Python.

- Unix/MacOS
  - `cd path/to/some/directory`
  - `python -m venv {name of the venv, e.g. npgr035env}`
  - `source {name of the venv, e.g. npgr035env}/bin/activate`
  - `python -m pip install --upgrade pip`
  - `python -m pip install -r requirements_{exact/simple}.txt`
  - `deactivate` *for deactivation*