Computing for Medicine: Phase 3, Seminar 4 Project

Rabiya Noori

Based on slides by Michelle Craig

Package Installation

```
> conda activate C4M (Windows)

> source activate C4M (Mac)
```

Packages needed for this project:

- numpy
- scipy
- scikit-learn
- matplotlib
- **scikit-image**
- **joblib**

Package Installation

```
> conda install -c anaconda numpy
> conda install -c anaconda scipy
> conda install -c anaconda scikit-learn
> conda install -c conda-forge matplotlib
```

New for this project:

```
> conda install -c conda-forge scikit-image
> conda install -c anaconda joblib
```

Starter code and data

Starter code

- image_processing_tutorial.py
- nuclei_detection_tutorial.py
- project_helpers.py
- project.py

Data

- 100 H&E stained histology images of colorectal adenocarcinomas
- Sirinukunwattana et al., ‘Locality Sensitive Deep Learning for Detection and Classification of Nuclei in Routine Colon Cancer Histology Images’

Your tasks

Project goal: automatically detect Nuclei centres in histology

images.

- Read and understand the code provided in image_processing_tutorial.py.
- Complete functions from by modelling your solutions after the starter code.
  - project_helpers.py,
  - nuclei_detection_tutorial.py, and
  - project.py

Data path

- The starter code assumes that the data directory (crchistophenotypes_2016_04_28) will be in the same directory as the .py files.
- If that is not the right location, you must set the data_path variable to the right directory.

Viewing plots

- In the starter code, there is a constant named VIEW.
- When VIEW is set to False, no images are shown.
- When VIEW is set to True, the images are displayed using show.
- You may change the value of VIEW as you develop your code.
- Example code snippet from starter code:

```
if VIEW:

        pyplot.show()
```

tuple

- Python has a type tuple, which is used to store ordered collections of data.
- Like lists, tuples can be indexed.
- Unlike lists, tuples are immutable.

Example:

```
>>> t = (1, 2, 3, 4)
>>> len(t)
4
>>> t[1]
2
```

Numpy’s vstack

- Take arrays and stack them vertically to produce a single array.
- Example:

```
>>> a1 = np.array([1, 2, 3])
>>> a2 = np.array([4, 5, 6])
>>> result = np.vstack((a1, a2)) # note: two sets of parentheses; the argument is the tuple (a1, a2)

>>> result
array([[1, 2, 3],
       [4, 5, 6]])
```

Numpy’s dstack

- Take arrays and stack them depthwise to produce a single 3D array.
- Example:

```
>>> a1 = np.array([1, 2, 3])
>>> a2 = np.array([4, 5, 6])
>>> result = np.dstack((a1, a2)) # note: two sets of parentheses; the argument is the tuple (a1, a2)

>>> result
array([[[1, 4],
        [2, 5],
        [3, 6]]])
```