## Assignment 1, Exercise 4

Download and unzip the [training and validation (4 GB)](https://drive.google.com/file/d/1nK9H2ldvGQL9LAIOYk3E8nD9kUri_Ly4/view?usp=sharing), and [testing (1 GB)](https://drive.google.com/file/d/13i66bWie00tt6LuhLPzoZ8pBPJ9ZFIXf/view?usp=sharing) subsets of the Patch-CAMELYON dataset. Note that the unzipping process might take a while due to the large number of files in the archives.  
Write a small Python script that reads and displays a few images from the two classes. Visually describe and compare the appearance of the tissue in the patches with and without metastases.

## Show 5 images of tissues without metastases and 5 images of tissues with metastases

names0 = ['',

'86de9372bb997cd6035632409db41d6e86af9956.jpg',

'86dfaa84c46733a5dcd66c0f9b4617f737c5101e.jpg',

'86e1d2e0b588aeb77481a59e19b08ff5913341e5.jpg',

'86e4ab0936529351589ef3e218aad1918799fcc4.jpg',

'86e6e628e6b881f4f8d00ead3f550815d1897fa0.jpg']

#insert 5 image names including .jpg or .tif

names1 = ['',

'7518323a26fe0368a6245e4821ba8741ee0d566e.jpg',

'7520311b93c0f597e032006b3b5e4b34f229c5fb.jpg',

'7523674dbc4aeae1c52ede35e5cc03b7a5d3db12.jpg',

'07530776f753d8d26c04eaaff29e9776fd89131b.jpg',

'7534162eff11bd83be4bfae4034ee5423ddf695c.jpg']

#insert 5 image names including .jpg or .tif

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

columns = 5

rows = 1

#without metastase

fig=plt.figure(figsize=(20, 4))

fig.suptitle('Without metastases', fontsize=14, fontweight='bold')

for i in range(1,6):

img\_name = 'train/0/' + names0[i]

img = mpimg.imread(img\_name)

fig.add\_subplot(rows, columns, i)

plt.imshow(img)

plt.axis('off')

#with metastase

fig=plt.figure(figsize=(20, 4))

fig.suptitle('With metastases', fontsize=14, fontweight='bold')

for i in range(1,6):

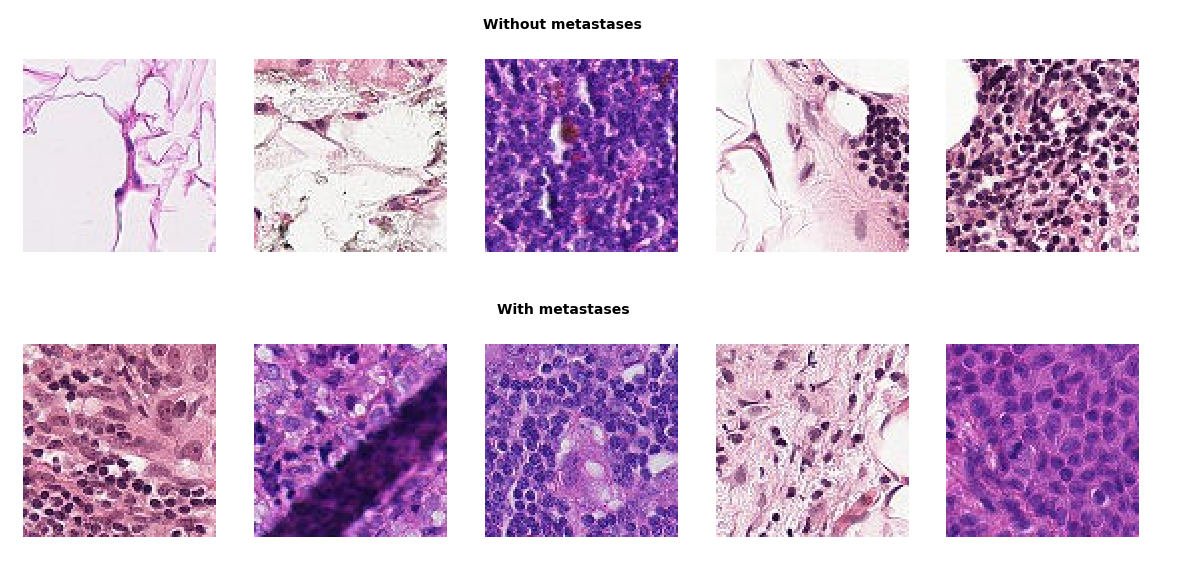
img\_name = 'train/1/' + names1[i]

img = mpimg.imread(img\_name)

fig.add\_subplot(rows, columns, i)

plt.imshow(img)

plt.axis('off')

I made the above script to show 5 images with metastases and 5 images without metastases. The script works and give 10 images as an output, but I cannot really see a clear difference or distinction between them.

**Assignment 2, Exercise 1**

The example neural network classification model in this assignment is relatively simple — it contains a single hidden layer of 64 neurons.  
Perform a set of experiments with more complex models, e.g. with more layers (deeper models), more neurons per layer or a combination.  
Describe the set of experiments that you have performed. What is the accuracy of the best model? How did you determine which model is the best?

(The values of the loss and accuracy differ each time I run the program, but these are some values as an example)

A layer is created by the following code line:  
model.add(Dense(64, activation='relu'))

Which creates a layer with 64 neurons. This results in a loss of 0.1885 and a accuracy of 0.945.  
Another layer can be created by copying the code line and adding it another time to the script. The number of neurons can be altered by changing the number 64.

|  |  |  |
| --- | --- | --- |
|  | Loss | accuracy |
| 1 layer, 64 neurons | 0.1885 | 0.945 |
| 1 layer, 128 neurons | 0.1756 | 0.9484 |
| 1 layer, 256 neurons | 0.1700 | 0.952 |
| 2 layers, 64 neurons per layer | 0.1317 | 0.9601 |
| 3 layers, 64 neurons ‘’ | 0.1258 | 0.9609 |
| 4 layers, 64 neurons ‘’ | 0.1133 | 0.9655 |
| 5 layers, 64 neurons ‘’ | 0.1306 | 0.9578 |
| 10 layers, 64 neurons ‘’ | 0.1189 | 0.9651 |
| 20 layers, 64 neurons ‘’ | 0.1775 | 0.9574 |
| 5 layers, 128 neurons | 0.0993 | 0.9693 |
| 5 layers, 256 neurons | 0.0935 | 0.9716 |
| 10 layers, 256 neurons | 0.0905 | 0.975 |

It seems as if the loss decreases and the accuracy increases as the number of layer and the number of neurons increases (separately or combined).

To find the best model:  
Jinchuan and Xinzhe [3] investigated a formula tested on 40 cases: 𝑁ℎ = (𝑁in + √𝑁𝑝)/𝐿 where 𝐿 is the number of hidden layer, 𝑁in is the number of input neuron and 𝑁𝑝 is the number of input sample. The optimumnumber of hidden layers and hidden units depends on the complexity of

network architecture, the number of input and output units, the number of training samples, the degree of the noise in the sample data set, and the training algorithm  
(K. Gnana Sheela and S. N. Deepa, “Review on Methods to Fix Number of Hidden Neurons in Neural Networks,” Mathematical Problems in Engineering, vol. 2013, Article ID 425740, 11 pages, 2013. <https://doi.org/10.1155/2013/425740>)

Using 1 hidden layer, 28x28 input neurons and 54000 input samples would result in 1016 hidden neurons. After running the script with this amount of neurons, a loss of 0.15356 and an accuracy of 0.9547, which are definitely not the most optimal values.