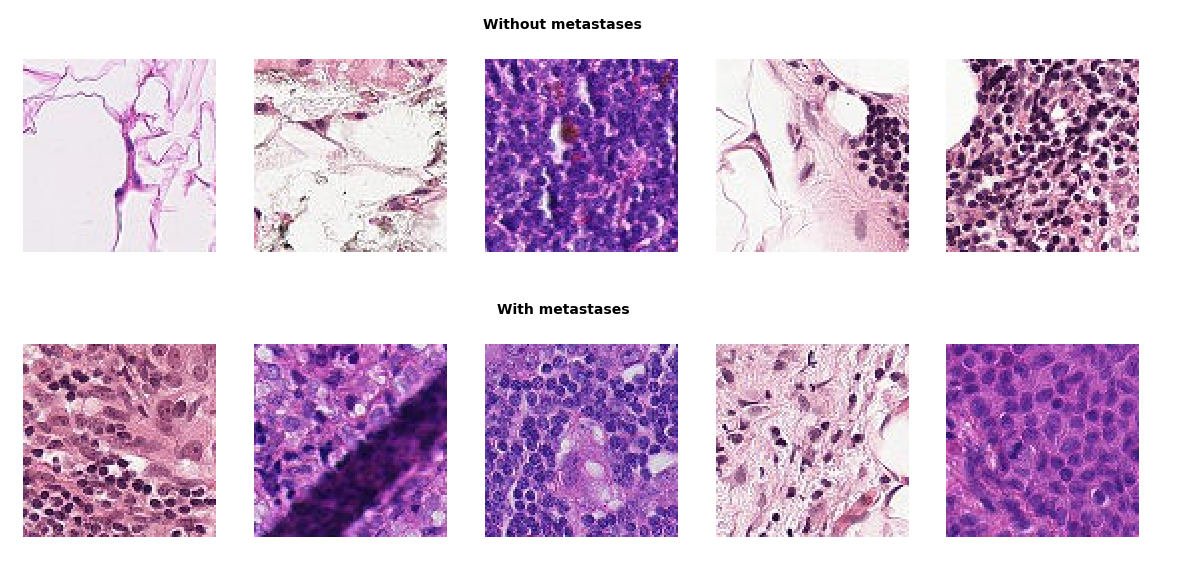
## Assignment 1, Exercise 4

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.

The python script written for this exercise displays a set of 5 images with metastases, and a set of 5 images without metastases, as can be seen in the figure below. Both sets of images show cells, cell nuclei, vessels (lymph or blood) and other tissue parts.

There is no clear distinction visible between the images with and images without metastases, because images in one set differ as much as images from different sets. A neural network should be made to determine if tissue in an image contains metastases.

**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 script has been run with the following combinations of number of layers and number of neurons per layer, which resulted in different losses and accuracies. From these results, it seems as if more layers and more neurons per layers cause a higher accuracy.

|  |  |  |
| --- | --- | --- |
|  | 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 per layer | 0.1258 | 0.9609 |
| 4 layers, 64 neurons per layer | 0.1133 | 0.9655 |
| 5 layers, 64 neurons per layer | 0.1306 | 0.9578 |
| 10 layers, 64 neurons per layer | 0.1189 | 0.9651 |
| 20 layers, 64 neurons per layer | 0.1775 | 0.9574 |
| 5 layers, 128 neurons per layer | 0.0993 | 0.9693 |
| 5 layers, 256 neurons per layer | 0.0935 | 0.9716 |
| 10 layers, 256 neurons per layer | 0.0905 | 0.975 |

A formula[[1]](#footnote-1) to determine the optimal number of hidden neurons could be

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. In case of the data that is used, Nin is 28x28 and Np is 5400. The formula has now changed to

𝑁ℎ = 1016/𝐿

To determine which combination of Nh and L results in the lowest loss and the highest accuracy, trial and error was used.

|  |  |  |  |
| --- | --- | --- | --- |
| Number of hidden layers | Numbers of hidden neurons per layer | Loss | Accuracy |
| 1 | 1016 | 0.15356 | 0.9547 |
| 2 | 508 | 0.1143 | 0.9649 |
| 3 | 339 | 0.0905 | 0.9723 |
| 4 | 254 | 0.0916 | 0.9717 |
| 5 | 203 | 0.0953 | 0.969 |
| 6 | 169 | 0.0902 | 0.9723 |
| 7 | 145 | 0.0945 | 0.972 |
| 8 | 127 | 0.1052 | 0.9714 |
| 9 | 113 | 0.1065 | 0.9704 |
| 10 | 102 | 0.1272 | 0.9641 |
| 11 | 92 | 0.1441 | 0.96 |
| 12 | 85 | 0.1244 | 0.9657 |
| 13 | 78 | 0.1324 | 0.9632 |

The optimal number of hidden layers is 6 with 169 neurons per layer, which resulted in an accuracy of 0.9723. Three hidden layers with 339 neurons per layers also results in an accuracy of 0.9723, but it generates a higher loss.

1. (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>) [↑](#footnote-ref-1)