UNIVERSITY OF AMSTERDAM

Breast cancer detection on SurfSARA's Cartesius with the CAMELYON dataset

Final report for Tweedejaarsproject BSc KI, July 1, 2018.

Lecturer:	Ashley Burgoyne	
Tutor:	Liz Verbeek	
Product owners:	Valeriu Codreanu	(SurfSARA)
	Damian Podareanu	(SurfSARA)
Authors:	Bram Otten	(UvA ID 10992456)
	Dennis Holkema	(UvA ID 10323155)
	Imre Fodi	(UvA ID 11407816)
	Sander Brinkhuijsen	(UvA ID 11305517)
	Lennart Beekhuis	(UvA ID 11344873)

Contents

1 Introduction

1.1 The CAMELYON challenge

Breast cancer is an illness that one in seven women in the Netherlands have to deal with in their lifetime. Each year, over three thousand women in the Netherlands die from the consequences of breast cancer (Kankerregistratie, 2018). Metastasis is the development of secondary malignant growths at a distance from a primary site of cancer. Some cancer cells are able to penetrate the walls of lymphatic vessels, after which they spread to other places in the body, settle, and continue to grow. These new tumours are called metastases. The detection of metastases is a time-consuming effort with high clinical relevance: patients with metastases need proper treatment in due time to prevent further spreading of the cancer cells. The CAMELYON16 challenge aimed to evaluate algorithms for automated detection of metastases in whole slide images (WSIs) of lymph nodes. The challenge was hosted by the Radboud UMC and UMC Utrecht, who provided 400 WSIs in total. Every WSI has been evaluated by eleven pathologists and labeled as containing metastases or being healthy. The goal of the challenge was to find an algorithm that could identify metastases correctly. This would reduce the workload of the pathologists, and reduce the subjectivity in diagnosis (Ehteshami Bejnordi, 2016a).

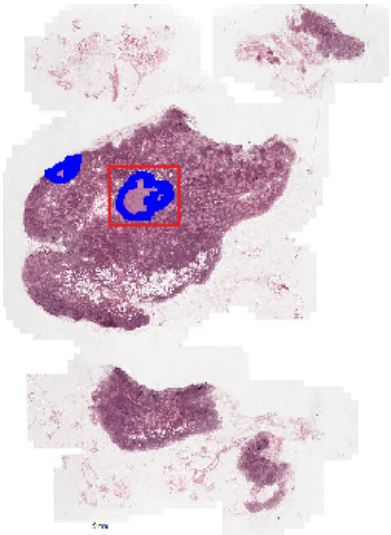

Figure 1: Example of a WSI (Ehteshami Bejnordi, 2016b)

The submitted algorithms were evaluated in two ways, slide-based evaluation and lesion-based evaluation. Slide-based evaluation measured the ability of the algorithm to accurately discriminate WSIs containing metastasis from the WSIs without. This evaluation was given in the form of a confidence score, with zero being fully confident in the absence of metastases and one being fully confident in the presence of at least one metastasis. In order to compare the different algorithms, the area under the receiver operating characteristic (ROC) curve is calculated. The ROC curve is created by calculating the percentage of false positives and true positives on a test set for different classification thresholds, see Figure 2. This threshold denotes how 'sure' the algorithm needs to be before labelling something as containing metastases. The better the classifier, the more the plot will be in the top-left corner, see Figure 3. The area under curve (AUC) is the size, as a percentage, of the area under the curve. The higher the AUC, the more the ROC curve is in the top-left corner, and thus the better the classifier.

Second is the lesion-based evaluation. This evaluation measured the ability of the algorithm to localize the metastases in the WSIs. In order to assess the quality of the localization, free-response receiver operating characteristic (FROC) curves are used. The algorithm assesses the entire image, marks the possible metastases location, and adds a probability that the marked areas contain metastases. Because the number of lesions is known, the true-positive fraction (TPF) can be calculated by dividing the number of marked areas by the total number of areas on the images known to have disease. This TPF is usually denoted as the sensitivity. The number of false-positive marks is counted but since there is no true negative equivalent, the false-positive rate (FPR) is calculated as the mean number of false-positive locations per image (CS, 2017). Then, like in the slide-based evaluation, the area under the curve is calculated.

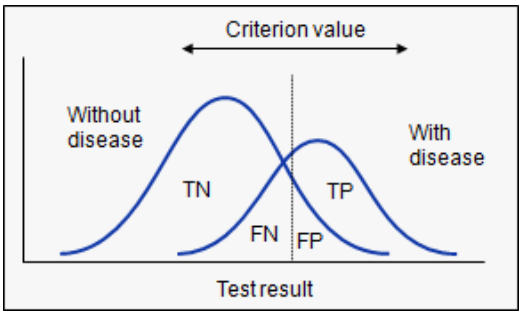

Figure 2: Quality of ROC curves (MedCac, 2018)

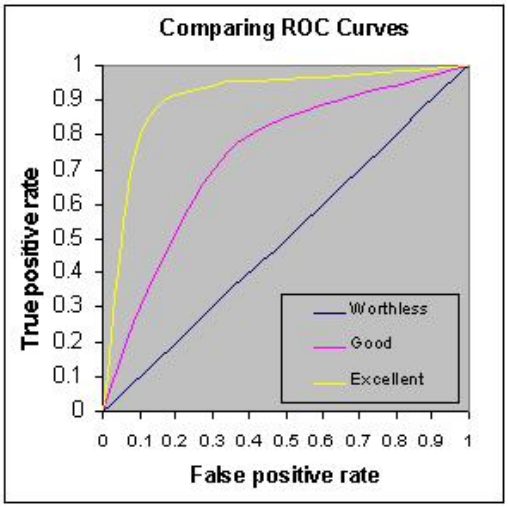

Figure 3: Comparing different ROC curves (Tape)

1.2 Previous solutions

The CAMELYON16 challenge was won by Wang et al. (2016). They achieved an AUC of 0.9250 for the classification task, figure 4, and a FROC score of 0.6933 for the localization task, figure 5 (Dayong Wang and Beck, 2016a). To put these results in perspective with pathologists' performance, two groups of pathologists analysed the same WSIs, under different conditions. The algorithms were subject to these same conditions. In the first test, the algorithm and pathologists did not have a time constraint, in the other test they had two hours to analyse the WSIs. In the time constrained simulation, the average pathologist had an AUC of 0.810. So the algorithm performs significantly better than the pathologists when there is a time constraint. However, when there was no time constraint, the pathologists had an AUC of 0.966 (Ehteshami Bejnordi et al., 2017). Thus, the winning solution only outperformed the pathologists when there was a time constraint. The winning team submitted an updated version of their method after the challenge had been terminated, and improved their AUC by a few percent, from 0.9250 to 0.9935, outperforming even the pathologists without the time constraint. They also managed to improve their FROC score. The FROC of this updated version had increased from 0.6933 to 0.8074 (Dayong Wang and Beck, 2016b).

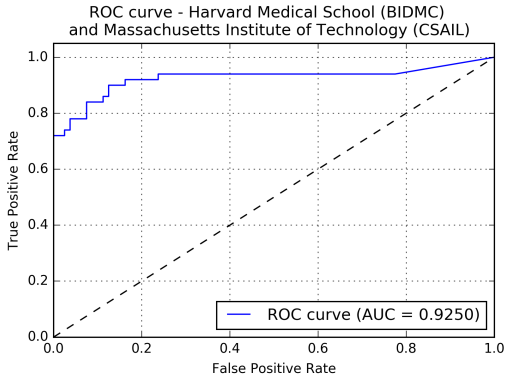

Figure 4: The ROC curve of the Camelyon challenge winner

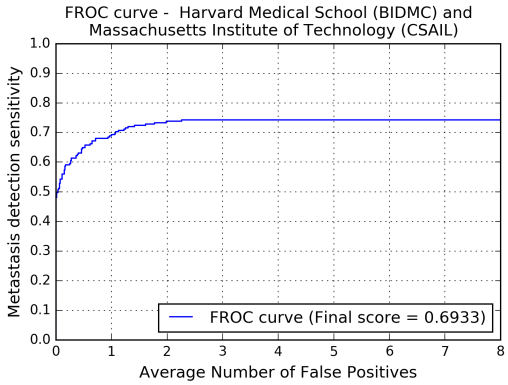

Figure 5: The FROC curve of the Camelyon challenge winner

1.3 Our challenge

Our client, SURFsara, provides students and researchers with advanced ICT facilities and they want us to improve on the existing CAMELYON solutions. These previous solutions split a large whole slide image (WSI) into smaller patches (256x256 pixels) before inputting it into their learning network, due to memory constraints. SURFsara wants us to adapt an existing implementation to use larger patch sizes. Using patches solves the memory issues, but it comes with an information loss. Some regions that contain metastases will be split up between multiple patches, and this might hinder the algorithm from recognizing the metastases. Larger patch sizes will decrease the number of split metastases and therefore decrease the information loss. There could be features that can only be recognized in these larger patches, but couldn't be in the smaller ones, which would improve accuracy too. Larger patches might also decrease the time needed for the algorithm to evaluate the WSI, since the number of patches will decrease. Our goals are to create an implementation that uses patch size 512x512, and if that succeeds, an implementation that uses 1024x1024 patches. In order to do this, SURFsara provides access to a high-performance computing (HPC) environment, which will enable us to address the memory constraints. We had to evaluate whether larger patch sizes increase the quality of the metastases detection and decrease the time needed for diagnosing a patient.

To be clear, the overarching goal still is the slide-based one from the CAMELYON challenge: to accurately predict whether the 130 WSIs in CAMELYONs test set contain metastases or not. Our method will initially be to emulate others on hardware that does not need to divide the input images up into so many different patches.

2 Method

2.1 Proposed solution and developed product

WSIs are large histopathological images (~100000x200000 pixels sized). There are 160 normal WSIs and 110 WSIs with tumours in our training set. First, we would like to classify these images as either 'containing metastases' or 'not containing metastases', but analysing these large images in one go is difficult because of memory constraints. A conventional solution in image processing is to split the image into patches. Most of the challenge submissions used ~256x256 pixel sized overlapping patches. However, this comes at informational cost, since the patches are treated as independent inputs, leading to a degradation of information compared to scanning the full image as done by a human pathologist. Because we have access to a high-performance computing (HPC) infrastructure, we can use larger patch sizes to come closer to the methodology a human pathologist uses. This will hopefully increase performance.

Because of time constraints on the project, we have mostly re-used already available code. We adapted our pre-processing code from a previous CAMELYON report and implementation by Vekariya (2018), available at `https://github.com/arjunvekariyagithub/camelyon16-grand-challenge`. Code has been tested on a subset of the KNL-partition of the Cartesius HPC environment, which has 18 Intel Xeon Phi 7230 processors with 96 gigabytes of RAM each.

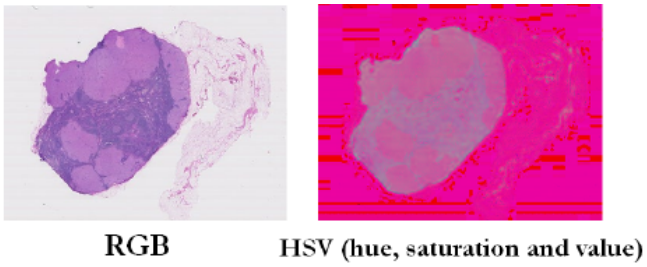

Figure 6: From RGB to HSV (Vekariya, 2018)

The first thing that needs to be done is extracting regions of interest (ROIs) from the WSIs. Because WSIs are very large (~100000x200000 pixels sized) images even at the medium (5X) level of magnification we use ($\frac{1}{2^5} = \frac{1}{32}$ of their original resolution) filtering white space from tissue space will reduce the computing time for other steps significantly. The pipeline is as follows:

First, the images are converted from RGB colour space to HSV (hue, saturation and value) colour space. HSV separates colour information from pixel intensity, this reduces the effect of different illumination levels on the images (Fisher, 1999).

Second, we create a binary mask by filtering pixels with HSV values in the range 20 to 200. The mask contains white pixels (with a value of 1) in areas where the HSV values are in the filtering range, and black pixels (with a value of 0) in areas where they do not.

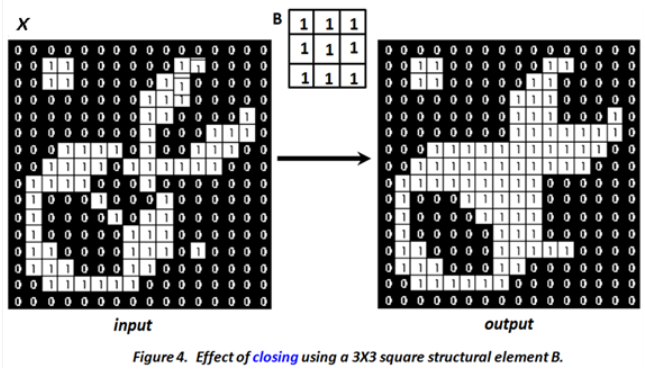

Figure 7: The closing operation (Vekariya, 2018)

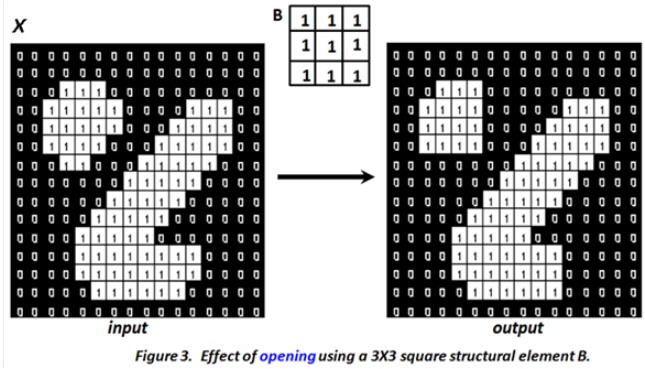

Figure 8: The opening operation (Vekariya, 2018)

Third, we apply a so-called closing operation on the binary mask, which fills small holes and breaks. The closing operation is done by first dilating and then eroding the image. The dilation process slides a grid over the mask and checks whether there is at least one positive pixel: if so, the whole grid will become positive. The eroding process also slides a grid over the mask: if all pixels in the grid are positive then all but the middle pixel will become negative. We preserved the grid size the original code used, which was 20x20.

Fourth, we apply an opening operation on the binary mask, which is the exact opposite of a closing operation. Instead of first dilating and then eroding, we first erode and then dilate the image. A grid size of 5x5 was used for this operation. This opening operation is used to smooth the boundaries in the images.

Lastly, we derive the boundaries of the white areas in the final binary mask, which gives us the contours. We then draw the contours on the original RGB image to highlight ROIs.

After acquiring the ROIs, we randomly extract patches from them. We did this for two patch sizes: 1024x1024 and 512x512. The number of 512x512 patches is in the order of 1000 per slide, although this is a parameter that could be modified. These patches were shuffled and split into a training and a validation set, the ratio of which is once again a parameter scheduled for tuning.

2.2 ResNet

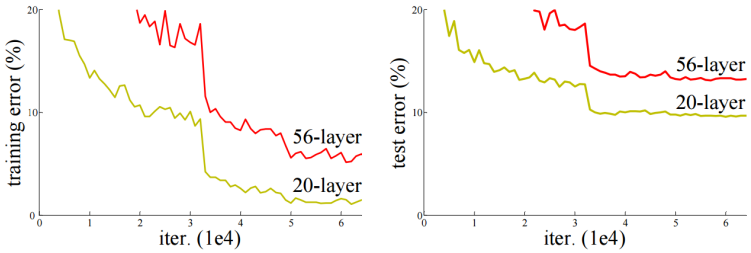

Figure 9: Training/test error of neural networks with different depth (He et al., 2015)

The method used for learning whether patches imply metastases or not is a residual neural network (ResNet). In a ResNet, not only connections between 'neighbouring' layers exist, there are also some connections that skip layers. This deals with the problem of degradation after accuracy saturation which occurs in deeper 'normal' networks (figure 9), where deeper networks converge to a lower accuracy than their shallower counterparts (He et al., 2015). The weights that our ResNet learns are convolution filters, as these tend to work well on images (LeCun et al., 2010).

Our client, SURFsara, provided us with two pre-trained models, trained on the ImageNet dataset. ImageNet is an image database with over 14 million images, which are all categorized. Pre-training is necessary because the CAMELYON dataset is too small to learn basic image features. The pre-trained model achieved an accuracy of around 70% on the ImageNet dataset. The code was written for Python 2.7.12, and primarily makes use of TensorFlow, which is a machine learning library that helps in making distribution over many different kinds of hardware possible. We used a special TensorFlow package optimized for the Cartesius HPC environment, which is based on TensorFlow 1.6.0.

The ResNet used in this project had 70 layers for dealing with 512x512 patches, and 72 layers for 1024x1024 patches. Parameters such as learning rate were initially set to values found to work well for classifying medical imaging data in a similar project by SURFsara, based on `https://github.com/sara-nl/tpu`.

2.3 Evaluation

After the model has been trained and tweaked, we can calculate a ROC curve. Patches are extracted from the test set WSIs and given as input to the model. The model assigns a probability of metastases to each WSI, then we calculate the area under ROC curve and compare it with other solutions. Due to limited time, we made lesion-based evaluation optional and decided to only do it when we had finished the slide-based evaluation.

3 Implementation

Our implementation and some documentation can be found in a GitHub repository on: `https://github.com/bramotten/Tweedejaarsproject-KI`.

In this implementation we, in a sense, have all components working. The ROI extraction and patch extraction from those ROIs definitely works in that it produces some output, but because later stages have not worked successfully their validity is unclear. The first of these later stages, the TensorFlow record building, is where problems began to surface. Either the input patches or the script that does the record building is incorrect, as at least some of the built records are. (A `verify_tfrecords.py` script is in the repository.) Without correct TensorFlow records, the ResNet could not be tuned. It fails on the incorrect input, as is to be expected. Because the ResNet has been used before by SURFsara, it is expected to work when the creation of TensorFlow records is fixed.

4 Discussion

As our end product is not finished, we will be discussing the hurdles and challenges we encountered here. First of all we provided with an implementation that was made available by Vekariya (2018), we were supposed to adapt this code in order to use larger patch sizes. Starting from this codebase seemed to complicate matters instead of simplifying them, as it did not work out of the box. The codebase was a crude amalgation of others' solutions with little to no informative comments. There were also classes that were defined multiple times in multiple ways, and the parameters were scattered over multiple files. After working on the code for a week we wondered if it had ever worked at all, as functions were called while missing important arguments. We had to rearrange or rewrite almost all of the code, which cost us a lot of time we could have spent on e.g. getting the ResNet to work properly or fine-tuning ResNet. If we could start over, we would try to find another repository with functioning code, and start building from there.

A second reason for the product unfinished was getting started on the HPC environment. There was a slight delay in getting access to the HPC and actually making the required software work took some time, as the installation of packages and programs was hindered by our lack of administrator rights. This was compounded by our later than usual start, since another project fell through and communication was stiff. Because of this we effectively lost nearly a week of development time.

Last of all, dividing the task also turned out to be more difficult than we thought. The task is mostly sequential, and because we didn't have any previous experience with TensorFlow or HPC environments, it was not possible to effectively divide coding tasks. This resulted in most of the time being spent just looking at the code, while one to three people were actually coding. In later weeks, this was solved by having some people work on the report, and the others working on the coding part, but especially the first week turned out less productive than we would have liked.

5 Future improvements

Our methodology omitted some basic improvements because of the time constraints we were facing.

One of these improvements is avoiding overfitting by augmenting the data to create more variety. The patches are extracted randomly, so patches are prone to overlap. We can mostly fix this by augmenting the data. We already performed some data augmentation by cropping a sub-region from the patches. Another example of data augmentation is flipping a patch (figure 10).

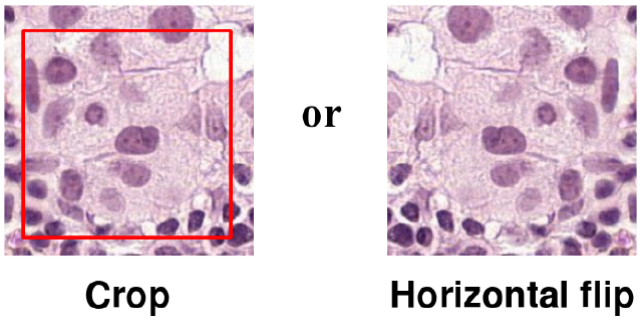

Figure 10: Augmentations: cropping and flipping (Vekariya, 2018)

A second improvement is training on a less random dataset. Right now, patches are extracted randomly, which could cause less frequently appearing features to not be learned properly. Arjun Vekariya describes in his paper exactly such a phenomenon happening: the false positive rate in his first run is high because the training dataset is missing some vital patches containing histological mimics of cancer. He trained a new model with a different dataset, containing 100000 extra patches with false positives. This improved accuracy by more than 10% (Vekariya, 2018).

Our general methodology with the aforementioned improvements seem popular among CAMELYON16 challenge submissions. But we feel there is at least one improvement that can be made upon it. The WSIs contain many different magnifications, and to make computation lighter a medium level of detail is used for finding the ROIs and patches. This apparently works well, but perhaps training two separate networks would be even better. One for the medium level of detail, and another for a greater level of detail. The medium-network can be set to have a relatively low threshold for marking something as 'maybe a tumour,' and these possible tumours can then be forwarded to the detailed-network which returns a conclusion. This could be slower, especially the training stage, but might be worth it in the end.

References

Moskowitz CS. Using free-response receiver operating characteristic curves to assess the accuracy of machine diagnosis of cancer. *JAMA*, 318(22):2250–2251, 2017. doi: 10.1001/jama.2017.18686.

Rishab Gargeya Humayun Irshad Dayong Wang, Aditya Khosla and Andrew Beck. Results of harvard medical school and mit, method 1, 2016a. URL https://camelyon16.grand-challenge.org/PerTeamResult/?id=BIDMC_CSAIL.

Rishab Gargeya Humayun Irshad Dayong Wang, Aditya Khosla and Andrew Beck. Results of harvard medical school and mit, method 1, 2016b. URL https://camelyon16.grand-challenge.org/PerTeamResult/?id=HMS2Up. (Retrieved: July 1, 2018).

Babak Ehteshami Bejnordi, Mitko Veta, and Paul Johannes van Diest. Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer. *JAMA*, 318(22):2199, December 2017. ISSN 0098-7484. doi: 10.1001/jama.2017.14585.

Johannes van Diest van Ginneken Karssemeijer Litjens van der Laak Ehteshami Bejnordi, Veta. Camelyon challenge, automated detection of metastases, 2016a. URL https://camelyon16.grand-challenge.org/background/. (Retrieved: July 1, 2018).

Johannes van Diest van Ginneken Karssemeijer Litjens van der Laak Ehteshami Bejnordi, Veta. Camelyon challenge, automated detection of metastases, 2016b. URL https://camelyon16.grand-challenge.org/data/. (Retrieved: July 1, 2018).

RB Fisher. Change detection in color images. In *Proceedings of 7th IEEE Conference on Computer Vision and Pattern*. Citeseer, 1999.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. doi: 10.1109/CVPR.2016.90. URL http://arxiv.org/abs/1512.03385. arXiv: 1512.03385.

Nederlandse Kankerregistratie. Borstkanker: Sterftecijfers, 2018. URL https://www.cijfersoverkanker.nl/selecties/sterfte_borst/img568ba9c4123de. (Retrieved: July 1, 2018).

Yann LeCun, Koray Kavukcuoglu, Clément Farabet, et al. Convolutional networks and applications in vision. In *ISCAS*, volume 2010, pages 253–256, 2010. doi: 10.1109/ISCAS.2010.5537907.

MedCac. Roc curve analysis, 2018. URL https://www.medcalc.org/manual/roc-curves.php. (Retrieved: July 1, 2018).

Thomas G. Tape. The area under roc curve. URL http://gim.unmc.edu/dxtests/roc3.htm. (Retrieved: July 1, 2018).

Arjun Vekariya. Camelyon16-grand-challenge: Implementation of Camelyon'16 grand challenge, June 2018. URL https://github.com/arjunvekariyagithub/camelyon16-grand-challenge. (Retrieved: July 1, 2018).

Dayong Wang, Aditya Khosla, Rishab Gargeya, Humayun Irshad, and Andrew H. Beck. Deep Learning for Identifying Metastatic Breast Cancer. *arXiv:1606.05718 [cs, q-bio]*, June 2016. URL http://arxiv.org/abs/1606.05718. arXiv: 1606.05718.