CAMELYON17 CHALLENGE: RESULTS FROM 4TH-IR

Jonathan Kaufman, on behalf of the 4th-IR team

4th-IR
Brooklyn, New York
and
Lucerne, Switzerland
j.kaufman@4th-ir.com

ABSTRACT
This paper serves as a compendium to the CAMELYON17 competition results for the 4th-IR team. We utilized the VGG-16 convolutional neural network to analyze digital pathology images of stained breast lymph-node biopsies for cancerous cells. Geometric features were extracted from tumor regions and used to train a Random Forest Classifier for whole-slide image diagnoses. These were then aggregated to determine full-patient stage diagnoses. Our analysis found 20 pN0, 1 pN0(i+), 28 pN1mi, 21 pN1, and 30 pN2 patients out of the 100 in the test data set.

1. INTRODUCTION
Although the Artificial Intelligence (AI) shown in most sci-fi tends to be more fiction than science, recent breakthroughs in computer vision and natural language processing are allowing computers to see, hear, and (most importantly) understand as well as or better than their human counterparts for specific tasks. The CAMELYON16 challenge demonstrated this for classifying breast cancer lymph node biopsies [1].

4th-IR was founded under the concept described by Professor Klaus Schwab (Founder and Executive Chairman of the World Economic Forum) in which a fourth Industrial Revolution, brought on by the advent of technologies such as the IoT and Artificial Intelligence will likely significantly overshadow the economic growth of all other periods of technological breakthrough. As a company, our goal is to utilize machine intelligence to engineer practical solutions that are easy to understand and simple to use. We decided to enter this competition as a fun side project in our field, as well as to gauge how our methodology compares to other groups’ from around the world.

The CAMELYON17 challenge asks participants to design algorithms for classifying the cancer stage of breast lymph node biopsies [2]. These categories are shown in Table 1.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pN0:</td>
<td>No micro-metastases or macro-metastases or ITCs found.</td>
</tr>
<tr>
<td>pN0(i+):</td>
<td>Only ITCs found.</td>
</tr>
<tr>
<td>pN1mi:</td>
<td>Micro-metastases found, but no macro-metastases found.</td>
</tr>
<tr>
<td>pN1:</td>
<td>Metastases found in 1 – 3 lymph nodes, of which at least one is a macro-metastasis.</td>
</tr>
<tr>
<td>pN2:</td>
<td>Metastases found in 4 – 9 lymph nodes, of which at least one is a macro-metastasis.</td>
</tr>
</tbody>
</table>

Table 1. Patient-level stage label definitions.

The paper is organized as follows: we describe the data sets in Section 2, followed by a description of our methodology in Section 3. We then present the results of our algorithms in Section 4.

2. DATA SETS
The CAMELYON17 data consists of 200 simulated patients with five whole-slide images (WSI) per patient for a total of 1000 WSIs, in addition to the complete CAMELYON16 data set. The WSIs originate from five treatment centers in the Netherlands.

Both the CAMELYON16 and CAMELYON17 data sets consist of multilayered WSIs where each layer represents a lower resolution version of the image. The full multilayered images are typically a few gigabytes, but in some cases can take up more than 20 GB of disk space. Images are around 100,000 × 100,000 pixels-squared with a resolution of 0.25 μm-per-pixel in both the x− and y−axes.

CAMELYON16 data consist of 270 WSIs for training (110 positive and 160 negative slides) and another 130 WSIs for testing. The training data came with detailed tumor annotations, which were used for training the deep learning network. The testing data did not have tumor annotations, but whole-slide classifications were provided. These were split between negative, macro-metastases, and micro-metastases.

...
The CAMELYON17 data set had a few tumor annotations, but ground truth was provided at the slide- and patient-level. In addition to negative, micro-metastases, and macro-metastases, the 2017 dataset contained the isolated tumor cell (itc) classification.

3. METHOD

We approached the challenge as three distinct steps: 1) analyze the WSIs for tumor cells, 2) diagnose the tumors at the WSI-level, and 3) aggregate metastases for a patient-level stage diagnosis.

For the first step, we utilized the deep learning algorithm described in Section 3.1 to generate probability heat-maps for normal/abnormal classification at the pixel-level. Geometric features from these heat-maps were calculated and used to train a Random Forest Classifier for WSI-level classification, as described in Section 3.2. These WSI classes were aggregated using logic derived from the definitions of the stages, as shown in Section 3.3.

For both datasets, we used the openslide-python library, which is a set of python bindings for the open-source OpenSlide C library used for opening medical multilevel tiff files [3].

Image preprocessing, deep learning training, and WSI analysis were performed on Amazon’s EC2 p2.xlarge instances running ubuntu 16.04. Each consisted of a four-core Xeon E5-2686v4 (Broadwell) processor with 61 GB of RAM, and an NVIDIA K80 GPU with 12 GB of dedicated memory. We utilized up to five of these running in parallel for the most computational intensive tasks. Data transfer was done on the “free-tier” t2.micro and approximately 4 TB of SSDs were attached on-demand. It cannot be overstated how much the flexibility of the AWS EC2 environment allowed for easy scalability, which increased efficiency for data transfer and analysis.

All analyses were implemented in python 2.7 using either the jupyter notebook or spyder environments.

3.1. Deep Learning for analyzing WSIs

Convolutional neural networks (ConvNets or CNNs) have proven to be exceptional at the task of image recognition [4]. We utilized the open-source VGG-16 architecture and weights [5], which we re-trained using our data set. VGG-16 is a deep CNN with thirteen 2-D convolution layers and two fully connected layers, followed by a fully-connected output layer. VGG-16 was created for the ImageNet competition, which included millions of every-day images (not cancer cells), however the architecture and filters have proven to be adaptable to different imaging applications [4, 5]. It’s worth taking pause and considering that a deep learning architecture built to distinguish cats, dogs, airplanes, cars, plants, etc. can easily and quickly be taught to distinguish cancer cells from healthy cells with high accuracy, and the technology is only improving. There are several other successful open-source CNN architectures [6, 7], but VGG-16 was chosen due to its performance and the group’s prior familiarity with the network.

We implemented VGG-16 using the keras library, which acts as a high-level interface for both the theano and tensorflow libraries [8].

Only CAMELYON16 data was used for re-training VGG-16. Since the VGG-16 input is $224 \times 224$ pixels-squared, the large WSIs were pre-processed and tiled. We determined that a simple threshold cut was sufficient for tissue segmentation. The tissue region was then tiled into $224 \times 224$ pixel-squared sections (without overlap). For WSIs without tumors, 1500 tiles were taken at random from within the tissue region, defined as at least 99% overlap between the tile and the tissue mask and an additional 500 tiles were taken from the edge of the tissue mask, defined as tiles with between 10% and 50% overlap with the tissue mask. These constituted the bulk of the “negative” labels for training.

For WSIs with known tumors, 2000 “positive” tiles were taken from within the tumor regions at random (or as many tiles as the region contains, if less than 2000). These were defined as tiles that have at least 90% of the pixels overlapping with the annotation mask. Another 500 “positive” tiles were taken at random at the edge of the tumor, defined as between 10% and 50% pixels in the tile overlapping with the annotation mask. In addition to these “positive” tiles, another 500 tiles were taken at random from the non-tumor regions within the same WSI and added to the “negative” labels. An example result of this tiling procedure can be seen in Figure 1.

This tiling resulted in nearly 300,000 images for training the classifier. All layers of VGG-16 were re-trained first using a 20,000 tile subset and a relatively coarse learning rate. The learning rate was tuned and the data set expanded until the accuracy of the training set and a separate validation set of 60,000 tiles plateaued, at 97%. Data augmentation was utilized in the form of random horizontal and vertical flipping, as well as random rotations up to 45°. Although rotations required re-sampling, which could introduce artifacts, it was found empirically that the benefits outweighed the potential harm.

To get the WSI probability heat-map, tissue regions were segmented as before and tiled into $224 \times 224$ pixel-squared regions. These tiles were passed to the re-trained VGG-16 classifier and probabilities for each tile were collected. An example of the probability heat-map is shown in Figure 2.

The full CAMELYON17 data set was processed with the re-trained network to generate probability heat-maps. In addition, the CAMELYON16 test-set images were processed. This added 130 more images, but lacked any slides with the ‘itc’ classification.
3.2. Classifying WSIs

Arguably the most challenging part of the CAMELYON17 competition is the WSI classification. This is partly because there is a relatively small amount of data and partly due to limitations with the technology, though the two reasons are related. We investigated several independent methodologies for classifying the WSIs but settled on a Random Forest Classifier for this work.

For each WSI probability we processed the heat-map by first creating a binary mask using a threshold cut on the probability. We then extracted feature vectors from the contiguous regions of the binary masks, as well as relevant metrics for the full WSI mask. Our goal was to keep the feature vector both simple and reasonable. We chose 1) the total pixel count for the positive regions in the mask, 2) the sum of the pixel count for the twenty largest positive regions (ranked by area) in the mask, 3) the number of positive regions in the mask that had an area of only one pixel, and 4) the variance of the mask.

The first element in the vector is fairly self-explanatory, as the metastasis level is primarily based on size. Although this is defined in literature by the radius of the region, doctors also look at less quantifiable measures, such as the extent. This makes sense intuitively, as the shapes are often quite jagged and elongated, making the radius difficult to define and measure in a consistent manner. The second element of the feature vector provides context for the total area. For a classifier trained on area alone, a noisy heat-map with 500 sparse ‘positive’ pixels would be classified the same as an image with one 500-pixel large tumor. Summing the area of a limited number of regions favors larger contiguous regions. The third element provides additional context on the area and acts an estimate for the noise. The final element is essentially a measure of spread. Large, compact tumors will have a very different variance than small, geographically sparse tumors. We investigated using more geometric features of the tumors like major-axis, minor-axis, ratio of the area to a box with the same outer dimensions, etc. but found no major improvement in our classification. It was decided to use the simpler model. Models should be no more complex than they need to be for desired results.

Before training the classifier, we removed 129 marginal WSIs to create a “golden” training set. Marginal WSIs were e.g. negative slides with a large area or macro slides with a small area, etc.

The Random Forest Classifier is a natural choice for challenges like this, as the classification problem can be logically structured as a decision tree, and the limited dataset size leaves the data susceptible to overfitting. Initially, we trained the algorithm using a training/validation split of 85%/15%. We generated 100 random realizations of the training/validation split and calculated the accuracy for each realization. Although this split had a high maximum accuracy, it also had a fairly large spread of accuracies. In a real-world scenario, it is not a safe assumption that the testing data statistically resemble the training data. However, in a challenge scenario, such as CAMELYON17, it is possible (and perhaps even likely) that the testing data resemble the training data and thus a reliance on overfitting may not be a significant risk. For example, the CAMELYON16 data had an approximately 60%/40% negative/positive split for both the training data and the testing data.

However, we approached this as we would a real-world problem by “starving” the classifier. For this, we systematically reduced the size of the training data set and re-ran 100 realization of each training/validation split. We tracked $\mu \pm 2\sigma$ of the classifier’s Cohen’s kappa score, where $\mu$ is the mean...
and $\sigma$ is the standard deviation of the kappa score for the set of 100 realizations. As expected, the spread of the kappa score shrinks as we starve the classifier, until the classifier is over-starved at which point the kappa score error increases again. This kappa spread behavior can be seen in Figure 3. We used a 35%/65% training/validation split for the final classifier, as the error is relatively stable at this split. The final classifier was trained using one realization of this split.

![Validation kappa score](image)

**Fig. 3.** The $\pm 2\sigma$ contours (red) and the mean (black) of the Cohen’s kappa score from 100 realizations of each training/validation split of the Random Forest Classifier.

### 3.3. Patient-level diagnosis

For the patient-level diagnosis, we resisted the urge to build another classifier, opting instead to use the technical definitions outlined in Table 1. Each patient-level classification was a simple aggregation of the slide-level classifications for all five WSIs per patient.

### 4. RESULTS

The results of our analysis are in the attached `submission.csv`. Table 2 shows the individual counts of each category for all WSIs in the test data set. These were then aggregated to determine full patient stages, shown in Table 3.

<table>
<thead>
<tr>
<th>negative</th>
<th>itc</th>
<th>micro</th>
<th>macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counts:</td>
<td>261</td>
<td>5</td>
<td>98</td>
</tr>
</tbody>
</table>

**Table 2.** WSI diagnoses from the Random Forest Classifier (RFC). These are aggregated to determine full-patient diagnoses.

<table>
<thead>
<tr>
<th>pN0</th>
<th>pN0(i+)</th>
<th>pN1mi</th>
<th>pN1</th>
<th>pN2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counts:</td>
<td>20</td>
<td>1</td>
<td>28</td>
<td>21</td>
</tr>
</tbody>
</table>

**Table 3.** Full-patient diagnoses for the CAMELYON17 test data set.

### Acknowledgements

The 4th-IR team would like to thank the Trestle team, in particular Casey Conger and Krisztián Posza for technical and intellectual support. We would also like to thank Nyq Kabelev for sharing his valuable expertise with us. 4th-IR would like to thank the organizers of the CAMELYON17 challenge. We thoroughly enjoyed working on this project.

### 5. REFERENCES


