MICCAI 2018 - ENDOSCOPIC VISION CHALLENGE: CATARACTS - Team IBT -

A PREPRINT

Andreas Wachter Institute of Biomedical Engineering (IBT) Karlsruher Institut of Technologie (KIT) Karlsruhe, Germany Andreas.Wachter@kit.edu Werner Nahm Institute of Biomedical Engineering (IBT) Karlsruher Institut of Technologie (KIT) Karlsruhe, Germany Werner.Nahm@kit.edu

September 10, 2018

1 Introduction

The main idea to determine whether a tool is in tissue contact with the eye, is to use a pretrained neuronal network for classification. This has the advantage that we do not need as much data to train the network from scratch. Being trained with thousands of other images, the pretrained net is already able to extract features in images like e.g. edges or shapes. The disadvantage of this approach is that the neuronal net is losing the information about time dependencies, because the Cataract surgery follows in the most cases a straight temporal workflow. We chose to use Matlab's [1] neuronal network toolbox [2].

2 Methods

The Matlab neuronal network toolbox has several advantages. The first is that you do not have to take care of CPU an GPU calculations. If available, it first uses the GPU. Another advantage is that Matlab can automatically build up an image datastore with the classification names extracted from the image folders.

For our approach, we only used the microscope video frames. The steps for the classification of the CATARACTS frames are:

1. Sorting all frames of the cataract videos into different folders according to the given classification. Frames which are not clearly classified by 1 are moved in the folder not_clear_classified. If there is more than one tool in tissue contact in a frame, this was handled as a new class. So in the end, we come up with 44 classes. A mistake that had happened is that also the data in the classes not_clear_classified and not_to_be_evaluated were included in the training process.

A problem after the frame sorting was that the numbers of images are not equally-distributed between classes. Figure 1 shows the distribution of images over classes.

2. The next step was to increase the training data. The number of training images should be equally distributed, so that the net is not biases towards specific classes. We were balancing the training images. For each class, we created 100000 images. We followed the approach from Jin, et al. [3].

Therefore, we randomly selected 100000 images from each class and decided randomly whether the image should be rotated, cropped, flipped or a combination of these three transformations should be applied.

Rotations were within the range $[1^{\circ}, 356^{\circ}]$. Bi-cubic interpolation was used and the images were cropped to the original format after rotation.



Figure 1: Image distribution after sorting the videos by the given classification

For cropping, the starting pixel in the left upper corner was randomly chosen within the ranges [1, 480] and [1, 270] for x- and y-coordinates, respectively. The width and height of the window was also randomly chosen within [961, 1440] and [521, 790], respectively.

At last, the image could be flipped horizontally, vertically or both. The decision was also randomly taken.

- 3. After increasing the image training data, the images were resized to 224×224 pixel, depended on the input layer of the pretrained net.
- 4. After preparing the images, the pretrained net must be adapted to the classification task. We decided to take the Google net form the Matlab neuronal network tool box. We replaced the last 3 layers, which are for the classification. We were added a fully connected layer, a softmax layer and a new classification layer with 44 nodes each.
- 5. Now the neuronal network was ready for training. The images were randomly divided into 75% training data and 25% validation data. The following training parameters were used:
 - sgdm (use the stochastic gradient descent with momentum (SGDM) optimizer)
 - MiniBatchSize 10 (how many file would load simultaneous)
 - MaxEpochs 15 (maximum number of training epochs)
 - InitialLearnRat 0.001 (learning rate)
 - ValidationFrequency 3 (frequency of network validation in number of iterations)
 - ValidationPatience Inf (no automatic validation stopping)
 - Verbose true (indicator to display training progress information)
 - plots training-progress

The other parameters were set to the default values. Matlab automatically determined to use 330000 iterations per each epoch.

6. Finally, automatic classification of the test data, frame by frame and only using the microscopic video data.



Figure 2: Intermediate snapshot of the training process and performance of the neuronal network.

3 Results

Because of time constraints, we were only able to train our neuronal net for 762 iterations and 1 epoch. This took 326 h 07 min and 42 sec. The validation accuracy was 53.8% at that point. A snapshot of the training performance will shown in figure 2.

The combined classes representing 2 or 3 tools within one image were reduced to the single probability using formula 1 in case of 2 tools and formula 2 in case of 3 tools. This way, we obtained only the given 21 classes in the table.

$$p_{1}' = p_{1} + p_{12} * \frac{p_{1}}{p_{1} + p_{2}}$$

$$p_{2}' = p_{2} + p_{12} * \frac{p_{2}}{p_{1} + p_{2}}$$

$$p_{1}' = p_{1} + p_{123} * \frac{p_{1}}{p_{1} + p_{2} + p_{3}}$$

$$p_{2}' = p_{2} + p_{123} * \frac{p_{2}}{p_{1} + p_{2} + p_{3}}$$

$$p_{3}' = p_{3} + p_{123} * \frac{p_{3}}{p_{1} + p_{2} + p_{3}}$$
(1)
(2)

It took about 0.089 seconds to classify one frame of the test data. The complete Matlab code can be found in the folder code with an meaningful file name. The tables can be found in the folder results and are named as the test videos.

References

- Inc. The Mathworks. MATLAB Version 9.4 (R2018a). https://de.mathworks.com/, Natick, Massachusetts, 2018.
- [2] MATLAB: Neural Network Toolbox, Version 11.1 (R2018a). https://de.mathworks.com/products/ neural-network.html, Natick, Massachusetts, 2018.
- [3] Y. Jin, Q. Dou, H. Chen, L. Yu, J. Qin, C. Fu, and P. Heng. Sv-renet: Workflow recognition from surgical videos using recurrent convolutional network. *IEEE Transactions on Medical Imaging*, 37(5):1114–1126, May 2018.