Applied Deep Learning
Final Report: The shape of the urine stream

Baris Kalayci
Vienna University of Technology
baris.kalayci@student.tuwien.ac.at

***Abstract*—The shape of the urine stream of males is an indicator for the healthiness of the prostate and the bladder. The goal in this work is to classify different shapes of urine streams which should provide the basis for a tool which can be used as a non-invasive method for measuring the healthiness of the prostate and the bladder. To achieve this, water streams with two different shapes were created with a plastic bag and a piping tip and images were taken and data sets were prepared. Therefore, the defined task is a binary classification problem where two different (basic and complex) Convolutional Neural Networks (CNN) were trained with the machine learning framework PyTorch[1]. Both of the models performed similar and an accuracy of 99.9% after 10 epochs could be achieved.**

Keywords – urine stream, prostate, bladder

I. Problem Description

Measuring the healthiness of the prostate or the bladder is today combined with a visit to a urologist. Although in 2017 the most common type of cancer for men in Austria was the prostate cancer, an examination appointment is very unpleasant for males [1]. One of the indicators of healthiness is the flow rate which can be measured with a clinical urine flow meter. From the flow rate the amount of pressure can be derived and a statement can be made. In addition to that, in a novel study [2] it could be shown that the shape of the urine stream of males is also a good indicator for the urine flow rate and orifice geometry of the penis (urethral meatus). Hence, the shape of the urine stream is a useful indicator for monitoring the health status of the prostate and the bladder. The maximum flow rate is used as an indication of an obstruction to urine flow which may be caused by prostate enlargement or various other conditions. In addition, low flow rate may be caused by problems in the contraction of the bladder, hence flow rate and associated urine stream shape along are not clear-cut diagnostic markers but are useful enough so that a patient or rather healthy male can monitor himself at home and can visit the urologist if necessary. That means also that the shape and flow rate of urine can be used as a tool since it provides a non-invasive method to monitor the health of the prostate and bladder. A clip of a typical simulated urine stream created with the help of the equipment of Wheeler et al. can be found on YouTube [2], the measurements are done with a ruler.

II. Project Idea

The project tackles a classification problem. The idea is to distinguish healthy urine streams from atypical streams with a CNN [3] programmed with the machine learning (ML) framework PyTorch. As the indicator, the wavelength (with wavelengths we mean the oval shapes of the stream shown in figure 2) of the streams will be measured. The project was designed to predict two classes: typical stream and streams with small wavelengths. Because the data set, the images with different classes, had to be created with limited or rather poor equipment, the decision felt on two classes. At the beginning of the project, the idea was to create three classes (a third class with long wavelengths). There was already a mail traffic with Prof. Knight before the project started who didn't mention an approach for classifying the wavelength of the stream (or other indicators of the urine stream) with Computer Vision (CV) or deep networks.

III. Data

Unfortunately, there are no real or artificial data samples which are provided by Wheeler et al. Therefore, the data sets for training, testing, and validating were created by myself. To simulate the stream and orifice of the penis a plastic piping tip of a cake decorating set and a plastic bag as the bladder was used.

The forms of the water streams serve as artificial urine streams. Water streams generated without building up pressure on the plastic bag belong to the **bad** category, whereas water streams generated with pressure belong to the **good** category. See examples for the both classes in figure 3.

In figure 1 the piping tip is shown which was prepared and formed to look alike an orifice of the penis.

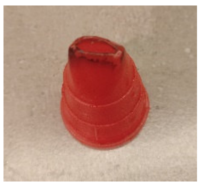

Figure 1 Piping tip to produce artificial urine streams.

The opening of the piping tip in figure 1 was formed to produce the desired shape of the stream which is shown in figure 2.

[1] https://pytorch.org/

[2] YouTube Link of the simulated urine stream: https://www.youtube.com/watch?v=BXezbw_xWoE

©2020 IPPP

Figure 2 It is clear that the prepared piping tip creates nice oval shaped water streams which is the desired stream shape. Due to less pressure, the oval forms are small.

Around 100k images were taken, 44% of category good and 56% of category bad. These images were split in train (60%), validation (20%), and test set (20%). To gain knowledge about constructing networks with different resolutions, the images are provided and used during the project in different resolutions (e.g. 300x300, 100x100, etc.). Although labels are provided, the prefixes of the image names give the indication to which group the image belongs to:

- Images starting with **g** belong to category good.
- Images starting with **b** belong to category bad.

Furthermore, the images were taken in different environments, outside with no specific (predefined) background and inside with a dark background. The second letter in the image name therefore indicates where the image was taken:

- If the second letter is equal to **d** the image was taken with a dark background.
- If the second letter is equal to **o** the image was taken without a predefined background. (Around 80% of the images have the dark background because it was much easier to generate the images under these conditions.)

In figure 4, the environment in which the images with the dark background were taken, is shown.

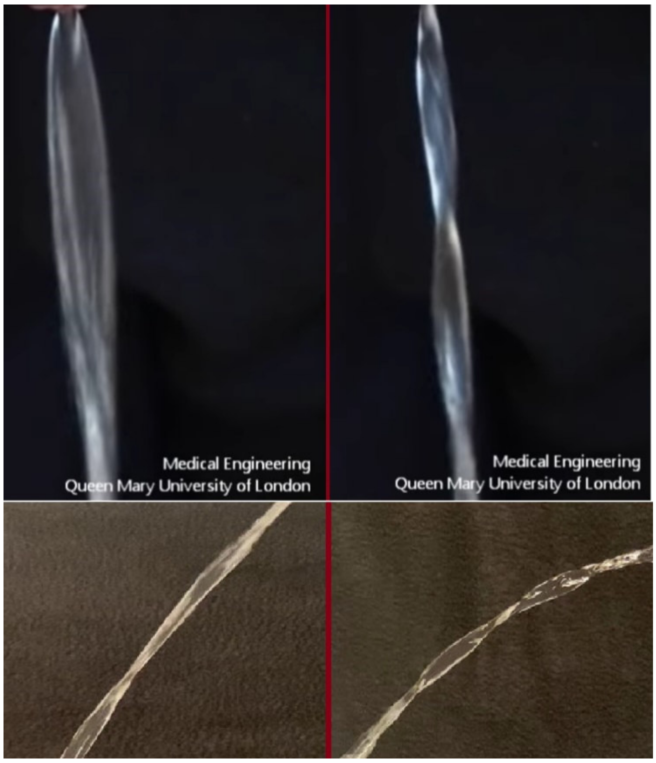

Figure 3 The left images show an example of a good stream, whereas the right images show a bad stream. The different oval shapes are good indicators for the pressure which was built up on the plastic bag. Compared with the water streams created by Queen Mary University, the results look similar.

Figure 4 Capturing images at home.

IV. Approach and Results

The problem is described as a binary classification problem which was solved with PyTorch. Two models were programmed to see the effects of more complex CNN's on the task. The first model is rather a simple model with only two convolutional layers. Whereas, the second model provides more convolutional layers so a deeper network can be trained. For the definition of the models, please refer to Appendix I.

Both of the models were trained and tested with 96x96 images. The highest available resolution of the images is 300x300 which can be downloaded here[3].

Interesting to see was the effect of dropout in the complex model which was used with a probability of 0.5. The size of the basic model is around 150 MB whereas for the complex model with dropout only 20 MB. We assume that the model size for the prepared task can be reduced much more to be more compact.
From the accuracy and losses plots in figure 5, we can see that an accuracy of 99% with train and validation losses < 0.1 can be achieved in few epochs for both models. From that we conclude that also with a linear classifier, comparable results regarding accuracy would have been possible to achieve. Therefore, the conclusion is that for this task a CNN wasn't necessary in our opinion.

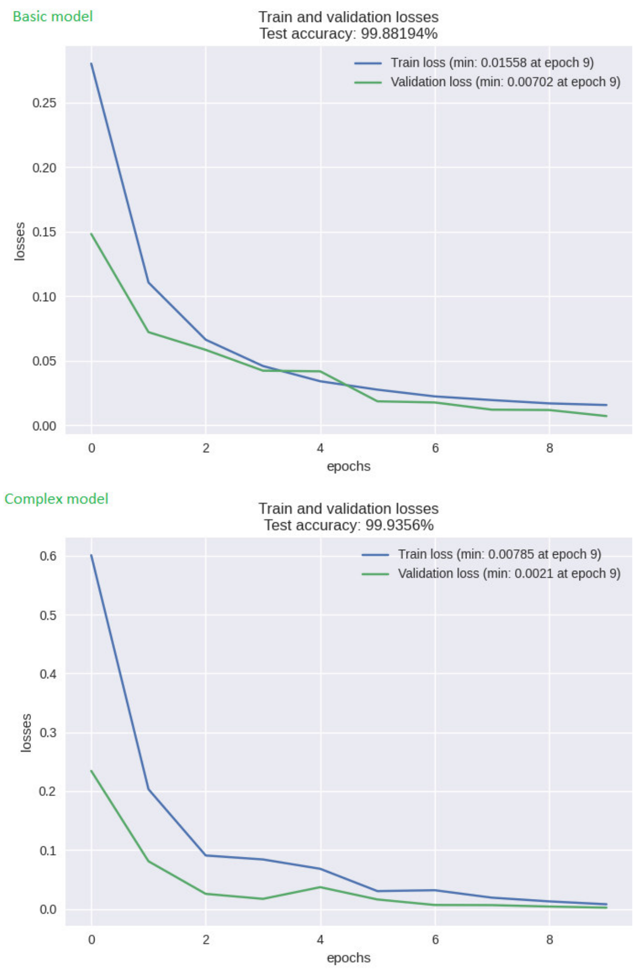

Figure 5 Train and validation losses for 10 epochs for both models. First plot shows the basic model and the second plot shows the results of the complex model.

V. Remarks on Parameters

During training and testing phase, different parameters were tried out. In this subchapter we want to explain them shortly and give an idea what was tried out:

- Normalization of channels: The performance of prediction was very much improved by normalizing the channels by mean and standard deviation of the test data set. At the beginning we used the values proposed by Imagenet with mean=[0.485,0.456,0.406] and std=[0.229,0.224,0.225]
- Batch normalization: Although, batch normalization improved the results, there were problems when the trained models were used inside web application. The model converted to ONNX and used within the web application performed very bad, it predicted always the same class (also when training samples were used). See: https://discuss.pytorch.org/t/performance-highly-degraded-when-eval-is-activated-in-the-test-phase/3323

  https://discuss.pytorch.org/t/low-accuracy-when-loading-the-model-and-testing/44991/6

- Optimizer:
  Stochastic gradient descent (SGD) worked quite well for this task. At the beginning we started with a learning rate of 1e-4 but the same results with a learning rate of 1e-1 could be achieved and the training time was shorter.
  AdamW was very sensitive to the weight decay. Large values for weight decays moved weights quickly towards zero. Adam behaved similar.

VI. Final Remarks on Project

Taking images, collecting data sets, and preparing them for training took almost more time than programming and training the models. Also, the environment and the same camera perspective while taking the images could be changed in future projects. Further, more images from different perspectives and angles are needed and with different backgrounds. The idea with the black background was to cover "negative" effects during training phase but the texture combined with the lighting maybe mislead the model in a way to predict the background rather the water stream. This could be also improved in future projects.

References

[1] STATISTICS AUSTRIA. 2020. Cancer incidence, Overview. Last visited: 19.01.2021. URL: https://www.statistik.at/web_en/statistics/PeopleSociety/health/cancer_incidence/cancer_incidence_overview/index.html

[2] Wheeler AP, Morad S, Buchholz N, Knight MM. The shape of the urine stream--from biophysics to diagnostics. PLoS One. 2012;7(10):e47133. doi:10.1371/journal.pone.0047133

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. DOI:https://doi.org/10.1145/3065386

[3] Images with the highest resolution: https://mega.nz/file/bIR2mbKT#ZtVDEW0-N8CzKuxjtt4thHh94hLPZWIRUQmAu8T0B-U

Appendix I

Summary of basic model

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 64, 96, 96]           1,792
       BatchNorm2d-2           [-1, 64, 96, 96]             128
         MaxPool2d-3           [-1, 64, 48, 48]               0
            Conv2d-4          [-1, 128, 48, 48]          73,856
       BatchNorm2d-5          [-1, 128, 48, 48]             256
         MaxPool2d-6          [-1, 128, 24, 24]               0
            Linear-7                  [-1, 512]      37,749,248
            Linear-8                  [-1, 256]         131,328
            Linear-9                    [-1, 2]             514
================================================================
Total params: 37,957,122
Trainable params: 37,957,122
Non-trainable params: 0
```

Summary of complex model

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 96, 96, 96]           2,688
              ReLU-2           [-1, 96, 96, 96]               0
         ConvLayer-3           [-1, 96, 96, 96]               0
            Conv2d-4           [-1, 96, 96, 96]          83,040
              ReLU-5           [-1, 96, 96, 96]               0
         ConvLayer-6           [-1, 96, 96, 96]               0
         MaxPool2d-7           [-1, 96, 48, 48]               0
            Conv2d-8          [-1, 192, 48, 48]         166,080
              ReLU-9          [-1, 192, 48, 48]               0
        ConvLayer-10          [-1, 192, 48, 48]               0
           Conv2d-11          [-1, 192, 48, 48]         331,968
             ReLU-12          [-1, 192, 48, 48]               0
        ConvLayer-13          [-1, 192, 48, 48]               0
           Conv2d-14          [-1, 192, 48, 48]         331,968
             ReLU-15          [-1, 192, 48, 48]               0
        ConvLayer-16          [-1, 192, 48, 48]               0
        MaxPool2d-17          [-1, 192, 24, 24]               0
           Conv2d-18          [-1, 384, 24, 24]         663,936
             ReLU-19          [-1, 384, 24, 24]               0
        ConvLayer-20          [-1, 384, 24, 24]               0
           Conv2d-21          [-1, 384, 24, 24]       1,327,488
             ReLU-22          [-1, 384, 24, 24]               0
        ConvLayer-23          [-1, 384, 24, 24]               0
           Conv2d-24          [-1, 384, 24, 24]       1,327,488
             ReLU-25          [-1, 384, 24, 24]               0
        ConvLayer-26          [-1, 384, 24, 24]               0
        MaxPool2d-27            [-1, 384, 3, 3]               0
           Linear-28                  [-1, 192]         663,744
          Dropout-29                  [-1, 192]               0
           Linear-30                    [-1, 2]             386
================================================================
Total params: 4,898,786
Trainable params: 4,898,786
Non-trainable params: 0
```