Universiteit Leiden

Master Computer Science

Prediction of overtopping volume of waves approaching a crest using a sequence of frames

Name:	Muhammet Özer
Student ID:	1311093
Date:	April 25, 2019
Specialisation:	Data Science
1st supervisor:	Dr. Wojtek Kowalczyk
2nd supervisor:	Dr. Bas van Stein

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

The correct assessment of coastal structures is important as these should meet some requirements. One of these requirements is the average overtopping discharge rate (in $m^3/s/m$). This is the amount of water that passes the crest per meter (of width) per second during the test duration. In a recent study, the problem is first decomposed into several parts. Consequently, deep learning is applied. When decomposing the problem, the final results are more sensitive to the output of the intermediate steps. In this research, the focus is on the prediction of the volume per meter in one overtopping wave (m^3/m) by only using a sequence of frames. The advantage of this end-to-end approach is that the decomposition of the problem is not needed. The network automatically extracts the features that are important for the prediction. Furthermore, many papers about coastal structures are in combination with neural networks with a multi-layer perceptron architecture, while in this research, the focus is on deep learning. Two deep learning architectures are applied to predict the overtopping volume per wave. These waves are created in the Scheldt Flume at Deltares. We trained a convolutional neural network (CNN) and a convolutional neural network in combination with a long short term memory network (CNN LSTM). A sequence of frames of a wave are used as input to a network to predict the overtopping volume. The CNN architecture achieves the MSE between 5 and 7 m^3/m, whereas the CNN LSTM achieves the MSE between 2.5 and 3.5 m^3/m. Unfortunately, the MSE can not be compared as there are no similar studies with this experimental setup. These results are obtained by using data from two cameras: the top camera for preprocessing the dataset and the side camera for making the prediction. With this research, we have shown that it is possible to predict the overtopping volume by using raw videos of waves.

Acknowledgement

This master thesis was done at LIACS in collaboration with Deltares in Delft. I would like to thank Dr. Wojtek Kowalczyk for his great knowledge about deep learning, supervision and involvement by visiting Deltares. In addition, I would like to thank my second supervisor Dr. Bas van Stein for his valuable feedback.

I would also like to thank Alex Capel, Matthijs Gawehn, Menno de Ridder and Joost den Bieman from Deltares for their great supervision and guided tour at the Hydro Hall. The Scheldt Flume which is used in this research is also located in the Hydro Hall and gave me great insight about how to approach the problem. Thank you for giving me the opportunity to do an internship at Deltares.

I would like to thank my family for their support. Last but not least, I would like to express my special thanks to my girlfriend Sehriban Bayram for always motivating me throughout my master.

Muhammet Özer

Contents

1 Introduction 5
- 1.1 Motivation . . . 5
- 1.2 Problem Definition . . . 5
- 1.3 Research Questions . . . 6
- 1.4 Overview . . . 7

2 Related Work 7
- 2.1 Coastal structures with overtopping (discharge) prediction . . . 8
- 2.2 CNN LSTM with similar data in other context . . . 12

3 Experimental Setup 14
- 3.1 Dataset Description . . . 14
- 3.2 Preprocessing . . . 16
- 3.3 Coding Environment . . . 21

4 Network Architectures 22
- 4.1 Convolutional Neural Network . . . 23
  - 4.1.1 Experiments . . . 24
- 4.2 CNN combined with LSTM . . . 26
  - 4.2.1 Experiments . . . 27

5 Results 30
- 5.1 Results CNN . . . 31
- 5.2 Results CNN LSTM . . . 32
- 5.3 Comparison of the two architectures . . . 34
- 5.4 Outlier Analysis . . . 34
- 5.5 Comparison with the existing neural network . . . 37
- 5.6 Limitations . . . 37

6 Conclusion 38
- 6.1 Future Work . . . 39

References 41

Appendices 43
- Parameters used in CLASH coastal structure . . . 43

1 Introduction

1.1 Motivation

The significant increase of the temperature in the last century led to the rise of the sea level. Because of this, the difference of the water level between the sea and crest decreased. With this, the wave conditions also changed. These developments are potential threats to the coastal structures. Adjustments of the crests might be needed in order to prevent flood. Decisions regarding the adjustments are based on the overtopping discharge rate (in $m^3/s/m$): the amount of water that passes the crest per meter (of width) per second. In other words, the risk and safety assessments are mainly based on this overtopping discharge, which makes the assessment of this measure important.

It is important to build robust coastal structures according, for example, the EurOtop manual [van der Meer et al., 2018]. The EurOtop manual provides guidance to coastal engineers to analyze overtopping performance. An example of a safety measure in the EurOtop manual is the overtopping discharge rate. An overestimation of the overtopping will lead into wrong decisions about the adjustment of a crest. There are many studies regarding the prediction of the overtopping discharge. An overview will be provided in section 2.

1.2 Problem Definition

The current algorithm of [Hydralab, 2018], is not robust in tests with a rubble mound breakwater and has difficulties with splashes and drops as the wave plunges. The research shows that it is possible to extract several parameters of a wave (in 2D basin) by applying deep learning to high-speed (150 Hz) video stream. Deep learning is one of the machine learning methods that learns from data to model an expected outcome. These parameters are the $\bar{v}$, $\bar{h}$ and T. Respectively: the time averaged bore speed (in m/s), time averaged flow depth per wave (in m) and the overtopping duration per wave (in s). By combining these parameters, the overtopped volume in (m^3/m) can be calculated with the following formula:

$$q_v \approx \overline{v \cdot h} \cdot T \tag{1}$$

The overtopped volume (in m^3/m) is predicted by first extracting the $\bar{v}$, $\bar{h}$ and T. In the second step, these parameters are multiplied to extract q_v. Waves often plunge at steep and/or rough slopes. The algorithm overestimates the boundaries of the wave. It assumes the water drops and splash as a part of the wave. However splashes and water drops are mainly air. An example of such a wave can be seen in Figure 1.

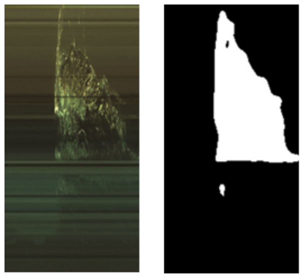


Figure 1: Example of a wave with splash of water and water drops in the air. Obtained from [Hydralab, 2018].

The consequence of this overestimation is that other parameters (such as q_v, $\bar{v}$ and $\bar{h}$) are overestimated as well. The latter is of significant importance as these parameters are considered during the construction of a dike. Note that it is also unclear to what extent a water drop or splash is significant for the deduction of wave impact. In terms of density, water is denser than air. As water is denser, it absorbs more light than a bubble. This is also the reason that bubbles appear whiter than the denser/lower part of the wave. Based on this, the bubble region can be approximated. However this approximation is subjective as there is no clear boundary between the real and the bubble part of the wave.

1.3 Research Questions

In this research the main focus will be on predicting the overtopped volume per wave with the help of deep learning. As training data, we will use videos, taken from a series of laboratory experiments which are held in the Scheldt Flume (which will be called flume in the remaining part of this research) at Deltares. Note that in many studies related to coastal structures and waves, the overtopping discharge is measured over the entire duration of the experiment in $m^3/s/m$. During this research, the overtopping volume in m^3/m is used instead of the overtopping discharge. This is the amount of water that is passing the crest, but normalized against the width of the metal funnel. We divide overtopping volume by the width of the metal funnel as this is different per experiment. These waves are being recorded by a top and side camera. This leads us to the following research question:

Can we get a good approximation of the overtopping volume q_v of (plunging) waves in a flume, by applying deep learning algorithm(s) on these video streams?

The quality of the approximation is measured by comparing the predicted value with the actual values measured by the physical devices, which is considered as the ground truth. The difference is expressed with the help of the mean squared error(MSE):

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(y_i - \hat{y}_i)^2 \tag{2}$$

[Hydralab, 2018] predicted the q_v after the extraction of the parameters $\bar{v}$, $\bar{h}$ and T. The aim of this research is to apply deep learning algorithm to **directly** predict the overtopping volume q_v. Deep learning is one of the machine learning techniques that can be applied. There are many neural networks. One of these networks is the convolutional neural network

(CNN) or convolutional neural network in combination with a long short term memory network (CNN LSTM). During this research the focus will be mainly on these two networks. However, other networks might also be applicable to this context. Therefore, the following sub research questions arise:

- *How does the CNN perform in the prediction of the overtopping volume q_v?*
- *How does the CNN LSTM perform in the prediction of the overtopping volume q_v?*

Various wave conditions are created by a wave board in the flume. Wave characteristics involve different parameters such as water level, wave height, speed and flow velocities. However, a deep neural network which predicts the overtopping volume q_v does not necessarily provide us other parameters like the time averaged bore speed ($\bar{v}$ in m/s), time averaged flow depth per wave ($\bar{h}$ in m) and overtopping duration per wave (T in s). During this research the focus will be the prediction of the overtopping volume q_v without the decomposition of the problem in the above mentioned parameters ($\bar{v}, \bar{h}, T$). Regarding the interpretability, only a small study will be done without results.

1.4 Overview

This thesis will be structured in the following way. Section 2 contains an overview of the related work regarding coastal structures, CNN and CNN LSTM. Section 3 contains a graphical overview of the experimental setup at Deltares. The preprocessing steps and descriptive statistics are also described in this section. Section 4 provides a brief overview of how a simple neural network works. It also contains a description of the network architectures and experiments with different (hyper)parameters that are used during this research. The results and the comparison of the CNN and LSTM are discussed in section 5. Conclusions, answers to the research questions, and future work are discussed in section 6.

2 Related Work

The literature about coastal structures in combination with neural networks is mainly based on the CLASH (Crest Level Assessment of Coastal Structures by Full-scale Monitoring, Neural Network Prediction and Hazard Analysis on Permissible Wave Overtopping) project and EurOtop (European Overtopping) manual. The CLASH project provides a database with more than 13000 tests about wave overtopping. These wave overtopping events are occurring under different circumstances and have more than 30 parameters and is now called the EurOtop-database. The artificial neural network (ANN) that is being used in the CLASH project is updated by the EurOtop ANN, [Zanuttigh et al., 2017]. The CLASH ANN is trained on a smaller number of tests than the EurOtop ANN. The main difference is that the EurOtop is also able to predict small discharges. The EurOtop neural network can be seen in Figure 2.

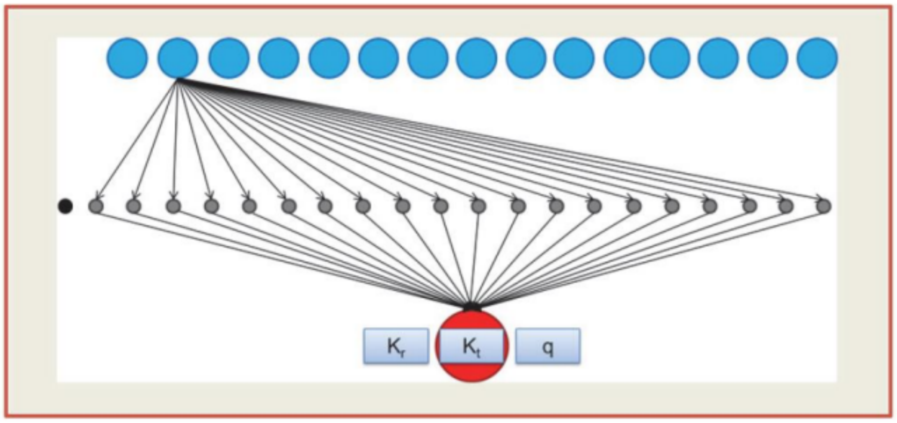


Figure 2: The ANN used in [van der Meer et al., 2018]. It consists 15 input parameters, 1 hidden layer(which consists of 20 neurons and 1 bias) and 1 output. The output can be q_d, K_r or K_t.

This architecture, also known as multi-layer perceptron (MLP) is trained by using the q_d, K_r or K_t. These are respectively the overtopping discharge, the wave reflection and the wave transmission coefficients. In other words, three ANNs are trained by using the same input parameters.

However (except of [Hydralab, 2018]), there were no studies found about videos of coastal structures in combination with CNN and/or LSTM. In order to address the overtopping volume prediction problem, the related work is separated into two parts. The first part is about coastal structures in combination with the overtopping discharge rate. The second part is about CNN and LSTM which also uses a similar dataset (with temporal information between frames).

2.1 Coastal structures with overtopping (discharge) prediction

[van Gent et al., 2007] performed a research to estimate wave overtopping discharges for wide range of coastal structures. These coastal structures are taken from a large number of physical model test within the European project CLASH. The parameters can be seen in the scheme of the coastal structure in Figure 3.

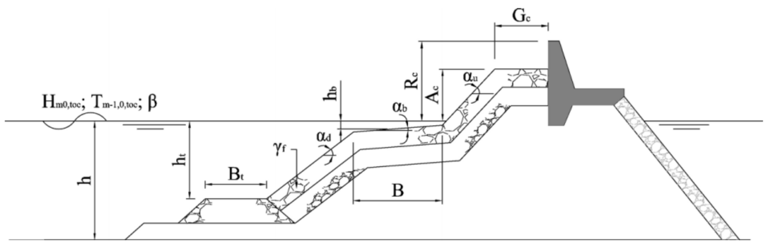


Figure 3: Scheme of the clash coastal structure [van Gent et al., 2007].

Figure 3 was taken from the research of van Gent. This includes (some of) the parameters that is used for training the neural network to predict the overtopping discharge rate of waves. Table 3 contains the parameters used in the CLASH project. This table is taken from [Zanuttigh et al., 2016b] as [van Gent et al., 2007] does not provide a clear overview about these parameters. This is the reason why some of the parameters in Figure 3 and Table 3 might slightly change in notation. Furthermore, some of the data is deleted as these were considered as inconsistent. This resulted into a dataset of 8372 tests. The neural network used in that research is similar to the one presented in Figure 2. The network is trained to estimate the logarithm of the observed overtopping discharges scaled with Froude's law ($\log q_{obs}\prime$). The root mean squared error (RMSE) in their paper is thus measured by:

$$RMSE_{train} = \sqrt{\frac{1}{N_{train}} \sum_{n=1}^{N_{train}} ((\log q_{obs}\prime)_n - (\log q_{NN}\prime)_n)^2} \tag{3}$$

They divided the dataset into a training and test set. It is crucial to have a good test set as this will provide and object measure of the performance of the neural network. In their research they trained multiple networks by using different splits of the training and test. The training and test sets are created by using Resampling Techniques [Boogaard, 2000]. As mentioned above, the network has 1 hidden layer with 20 neurons. According to their research, using more than 20 neurons in the hidden layer did not significantly decrease the RMSE. 500 resamples of the original dataset and corresponding trained neural networks were created. The prediction of the overtopping discharge is based on the mean of these 500 neural networks. The trained networks were mainly good in predicting the high overtopping discharges.

The neural network trained by van Gent had greater errors regarding prediction of low overtopping data. Van Gent used a threshold to determine whether a wave is significant or not. This threshold was set to $10^{-6} m^3/s/m$. [Zanuttigh et al., 2016b] extended the CLASH database by adding a few more parameters. Furthermore, small overtopping discharges $< 10^{-6} m^3/s/m$ were also included in the dataset. This is done on purpose as the elimination of overtopping discharge $< 10^{-6} m^3/s/m$ can cause bias. To minimize this bias, all records with values between 0 and $10^{-6} m^3/s/m$ are also included in the dataset for training the neural network. Scaling effects are largest for low overtopping results. This effect is however not been considered. This led to significant reduction of the the bias of the neural network predictions regarding low overtopping discharges. However, the network was not able to predict overtopping values of zero as this resulted into a continuous relationship between input and output parameters. Having a zero-value causes a discontinuity in this relationship.

More research with the same neural network architecture as in Figure 2 is done by [Zanuttigh et al., 2016a]. There were no studies about overtopping prediction in combination with deep neural networks such as CNN and/or LSTM. The only research that we found is done by [Hydralab, 2018]. According to their recent research, deep image segmentation algorithm is used to extract the pattern of the waves in high speed video streams. These videos are obtained by using a 150 Hz high speed camera placed 1 meter from the flume. An example of a frame can be seen in Figure 4.

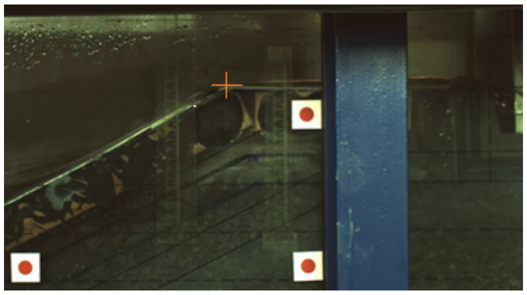


Figure 4: Point of view from the high speed camera of case with smooth slope. Red points are for calibrating the camera. Obtained from [Hydralab, 2018].

This high speed camera records the wave 1s before and 1s after the impacting wave (over a period of 2 seconds). The impacting wave is occurring at the orange marker in Figure 4. Thus the video footage of the waves is 2 seconds which results into 300 frames. Subsequently, these frames are used to create timestacks. Each timestack consists of 600 x n pixels with in the vertical and horizontal axis respectively space (600 pixels) and frames (n) at a certain position along the horizontal axis. To clarify, a wave consists of s $\times$ 150 frames (s seconds). By taking a vertical slice of a frame in for example the orange position, an image of 600 $\times$ 1 pixels is obtained. By doing this for multiple consecutive frames an image is obtained as in the left image in Figure 5.

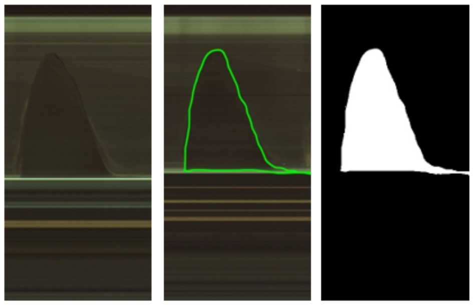


Figure 5: Examples of a 600 x 300 timestack. Vertical axis is in space, horizontal axis is in time. Left panel: original timestack. Middle panel: manual annotation. Right panel: resulting binary mask. Obtained from [Hydralab, 2018].

They used the above technique along every 10 pixels along the flume axis. The border of the wave (green line) is manually annotated in order to create a binary mask. The binary mask indicates the wave of a timestack. The manual annotation is applied to 225 timestacks. 200 and 25 timestacks are used respectively for training and testing the deep segmentation algorithm (neural network). A schematic overview of this network can be seen in Figure 6. The binary mask is also created to validate the performance of the network. With validation, the shape of the wave is meant. Once the shapes of the waves are correctly extracted, it is also possible to derive other parameters such as:

- Time averaged bore speed ($\bar{v}$ in m/s),
- Time averaged flow depth per wave ($\bar{h}$ in m) and
- Overtopping duration per wave (T in s).

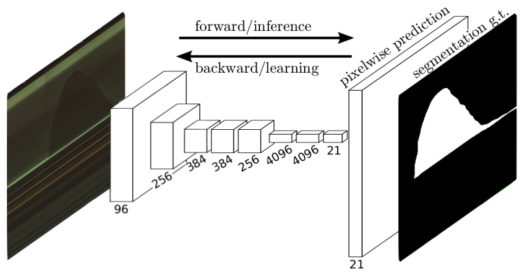


Figure 6: Schematic of deep image segmentation. From the original image (left) a set of neural network layers describe the image in a decreasing number of functions with increasing dimensionality. The pixel wise prediction of the segmentation classes is obtained using one or more densely connected layers (right). Obtained from [Hydralab, 2018].

By using these timestacks, a spatiotemporal graph is obtained, see Figure 7. By extracting the pattern of a wave, it is also possible to calculate the distance between the wave and the crest. The distance can be read from the vertical axis (with the crest at 0).

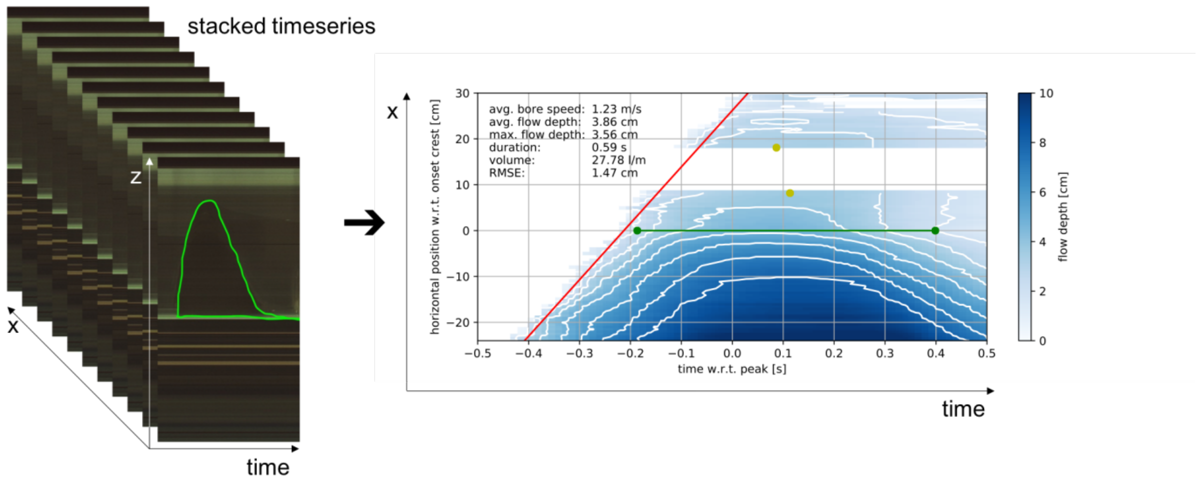


Figure 7: Spatiotemporal variations in flow depth (blue shading and white contours) of a single wave (B3W0T101, wave No. 9) and derived quantities bore speed (red line slope: linear fit through 2 cm flow depth contour), overtopping duration (green line length: maximum distance between up- and down-crossing of 2 cm flow depth contour through the origin of the vertical axis) and maximum flow depth in the vicinity of the physical measurement location (yellow dots). Obtained from [Hydralab, 2018].

Furthermore, the color intensity indicates the flow depth. The flow depth gets lower as the wave approaches the crest. Since the shape of a wave is known for each time, it is also possible to extract the peak of a wave. Doing so along several locations of the flume, wave height (m) as well as position (m) are known as a function of time (s). This can be used to calculate the velocity (m/s). The green line is associated with the crest of the dike (0 on vertical axis) and indicates the timespan (horizontal axis) along which a significant amount (at least 2 cm flow depth) of water has overtopped. The amount of water (m^3/s) passing the crest follows directly from the depth of the water column (blue shading) at a certain time (horizontal axis). To clarify, the white space between the two yellow dots is due to the a blue metal frame which can be seen in Figure 4. These parameters are used to calculate the overtopped volume per wave (q_v in m^3/m):

$$q_v \approx \overline{v \cdot h} \cdot T \tag{4}$$

The total overtopping volume is calculated by integrating the overtopping volume per wave. This can be compared to the overtopping volume measured by the physical device. They were able to predict the overtopping reasonable well. Waves are exposed to different circumstances. This ranges from different breakwater design, slope roughness and water level to wave steepness and wave height. But as mentioned in the section Problem Definition, it is unclear to which extent bubbles belong the wave. This leads to wrong boundaries of a wave like in Figure 1. This will finally lead into wrong calculations of the overtopped volume q_v. By decomposing the problem as in [Hydralab, 2018], the final results are more sensitive the the output of the intermediate steps.

2.2 CNN LSTM with similar data in other context

During this research, videos are used as input to predict one single number: overtopped volume q_v. Similar research but in another context, namely self driving cars is done by [Pomerleau, 1989]. Pomerleau used a 3-layer back-propagation network for the task of road following and showed that the so called ALVINN architecture is suitable to some extent for autonomous navigation. A more recent study is done by [Bojarski et al., 2016]. Bojarski placed a cameras in front of a car. The goal was to train a CNN by using the left, center and right camera. After random shifting and rotation of these frames, the network is backpropagated by only using the steering wheel angle. At the end, the trained network is applied on only the center camera to generate steering commands. Most research are separating the problem into for example lane detection or path planning. Bojarski was training the CNN by only using one single number: steering wheel angle. This trained CNN was able to drive approximately 98% autonomously from Holmdel to Atlantic Highlands.

A research about CNN LSTM was made by [Donahue et al., 2015]. They have used a Long-term Recurrent Convolutional Network (LRCN) for activity recognition, image and video description. For activity recognition, they used the LRCN architecture for a sequence of frames to predict the activity. Image description is done by only using a single frame to give a description of the image. Video description is done in a similar way: a video (sequence of frames) is used as input to give a description of the video. The LRCN network is flexible enough to apply it on a variety of vision tasks as described above.

[Sainath et al., 2015] did also a research by combining CNN, LSTM and deep neural networks (DNN) with each other. The CNN is for reducing the frequency variation of the input while the LSTM is used for modeling the temporal information. The DNN is used to map the features into a space where the features are separable. This network is called CLDNN in their research. Among the three separate models (CNN, LSTM and DNN), the LSTM performed the best regarding the word error rate (WER):

$$WER = \frac{S + D + I}{S + D + C} \tag{5}$$

Where S, D, I and C are respectively the number of substitutions, deletions, insertions and correct words. WER is mainly used as metric in the context of natural language processing (NLP). The CLDNN improved the WER of the LSTM by 4-6%.

Similar algorithm is used by [Molchanov et al., 2016]. A recurrent 3d convolutional neural network (R3DCNN) is used in order to online detect and classify hand gestures. A video of a gesture (recorded with depth color and stereo-IR sensors) is presented as a series of frames, called clips in the paper. Each clip C_t is used as input for the convolutional layers. These are extracting the local spatial-temporal features f_t for a given C_t. Consequently f_t is used by the recurrent network to finally estimate the class conditional probabilities s_t of various gestures. The network is trained by using the connectionist temporal classification (CTC) cost function. By using this approach the network was able to classify gestures online with zero or even negative lag (classification before gesture is finished) with an accuracy of 83.8%. This was close to the human accuracy of 88.4%.

[McLaughlin et al., 2016] used a recurrent convolutional neural network in videos to re-identify persons. In their research, they used a so called Siamese network architecture. A video of a person is first converted into a sequence of frames. This is used as input together with the optical flow information for the CNN to extract the features. The output of the convolutional layers is used as input for the recurrent neural network. By doing this for multiple time-steps, overall appearance feature of the complete sequence is obtained. In other words, the RNN is used to deal with temporal information: a sequence of video(frames). This approach was applied on the iLIDS-VID and PRID-2011 datasets and was able to outperform the existing methods of video-based re-identification.

3 Experimental Setup

As mentioned in the introduction, the dataset is created by using a series of laboratory experiments which are held in the (2D basin) Scheldt Flume at Deltares. A simplified version of this experimental setup can be seen in Figure 8.

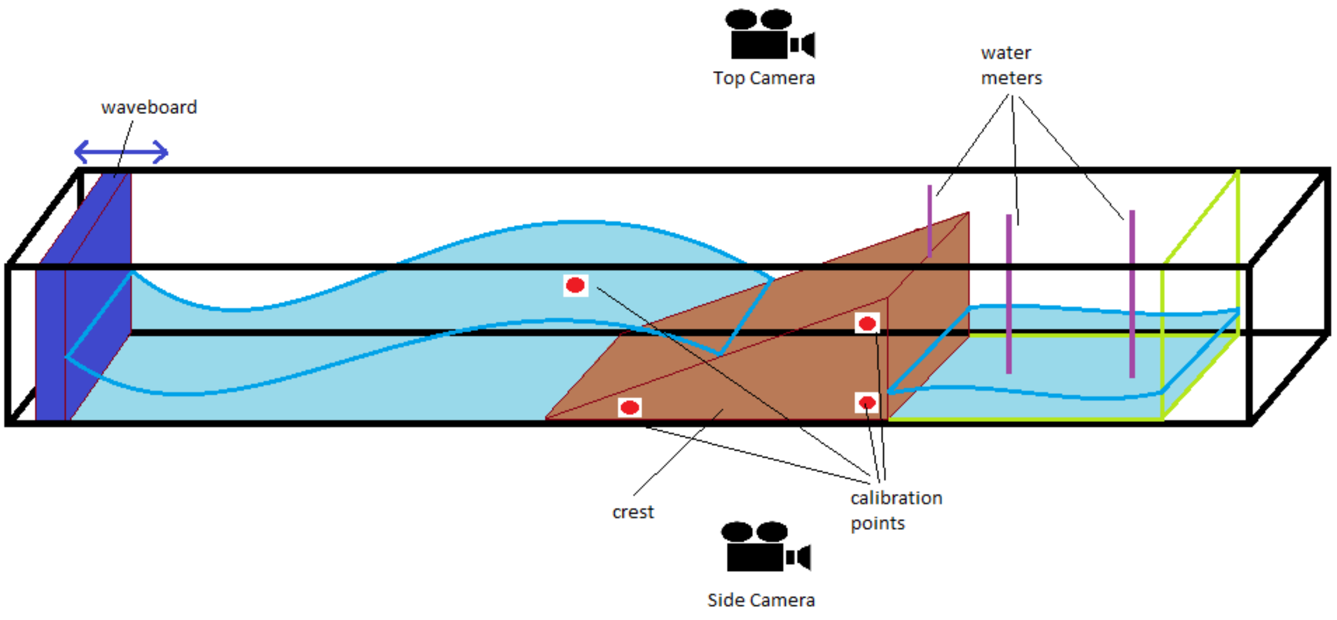


Figure 8: Simplified version of the experimental setup.

The 2D basin is filled with water. By moving the waveboard horizontally, waves are created. These waves are approaching the coastal structure, which has during this research a smooth slope. When the water passes the crest, it will fall into the box (in green). The water in this box is measured by the two independent water meters. The rightmost water meter is used to measure the level of the water as this is further away from the coastal structure which results into less volatile measurements. The leftmost water meter is used as a trigger. When a wave touches this water meter, the regarding wave can be extracted as this wave will result into overtopping in the box. The water meter in the middle is not used during this research. Waves that are approaching the crest are being recorded by two cameras. The side camera is recording with 25 Hz and the top camera records with 20 Hz.

3.1 Dataset Description

There were multiple experiments done in the flume. The duration of each experiment is roughly 30 minutes. In other words, the waveboard creates waves during this 30 minutes. Each experiment is recorded with the side and top camera. An example of a frame of the side and top camera can be seen in Figure 9.

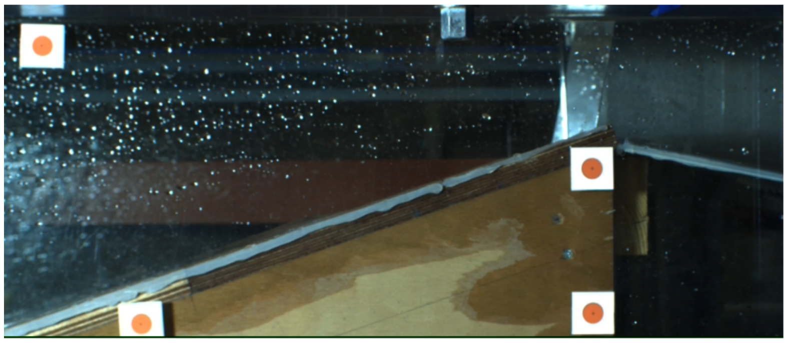
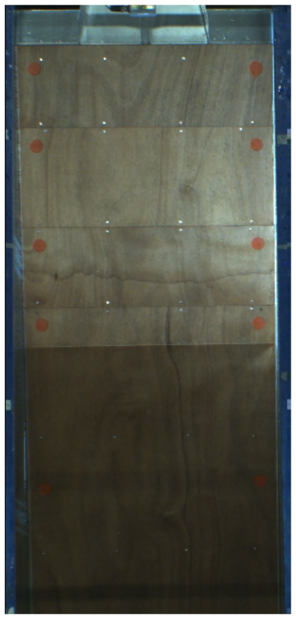



Figure 9: Left: frame from side camera. Right: frame from top camera.

The dimensions of a frame of the side camera is 850 by 2000 pixels. For the top camera, this is 1200 by 560 pixels. The middle and right water meters are measuring the water level in the box in voltage. Each wave causes a difference in voltage. Each meter has a different liter/volt ratio. The ratio of the left meter is unknown as this is not calibrated. The middle and right water meter are calibrated and are respectively 19.39 l/v and 18.84 l/v. By multiplying this voltage difference with 18.84, the overtopping in liter is obtained. A metal funnel can be seen in the upper part of the frame from the top camera. Water which is passing from the left or right of this funnel is not being measured. The amount of overtopping is dependent on the width of this funnel. As this is different per experiment, the overtopping is normalized against the width of this funnel, hence the overtopped volume in m^3/m. One might ask the question why this is done as we expect the deep learning algorithm to learn the width of the funnel by itself. The reason for this is that the prediction of the overtopping will only be done by using the side camera and that the labeling of the wave with the regarding overtopping will be wrong if this width is not considered. More about the procedure can be read in the next subsection. Furthermore, the top and side camera and the water meters are started at the same time and running/measuring simultaneously. An overview of the experiments is provided in Table 1. In each experiment, the wave board creates 1000 waves. The amplitude differs per experiment, resulting into different durations of the experiment. For example, T102 is the experiment with bigger waves while T103 and T104 contain smaller waves.

Experiment	Duration (seconds)	Funnel width (meters)
T102	2096	0.5
T103	1793	0.8
T104	1690	0.8
T105	2596	0.5
T106	2117	0.5
T109	2599	0.5

Table 1: Overview of the experiments.

3.2 Preprocessing

During these experiments, there are many waves which are passing the crest and causing overtopping. In this section, the steps will be described to extract these waves with their regarding labels.

As mentioned in the previous section, the leftmost water meter was used as a trigger. When the value of this trigger changes, we can say that water is passing by. The side camera is used to record the wave 1 second before and 1 second after the *activation* of the water meter (over a period between 2 and 3 seconds). However, there might be cases where waves are passing from the left or the right of the water meter, see Figure 10. To clarify, water which is passing from the left or right of the yellow line are not being measured.

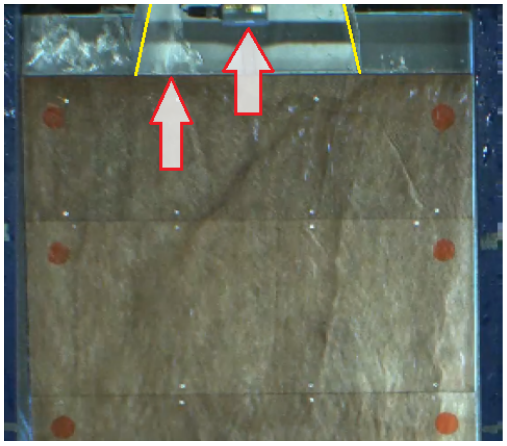


Figure 10: Yellow lines are the boundaries of the funnel. Only this water is measured by the water meters. Right arrow indicates the water meter (used as trigger). Left arrow shows us that water is passing from the left of the water meter.

By only using the water meter as trigger, mainly weak waves can not be detected, thus not extracted. This will result into a skewed dataset which consists of only strong waves. In order to avoid this, the waves are extracted in the following way. An experiment of roughly 30 minutes is initially converted to the maximal amount of frames by using the top camera. As the top camera is recording with 20 Hz, this resulted into 41935 frames for experiment T102. To extract waves, the funnel of each consecutive top frame is compared with each other, see Figure 11.

Figure 11: Only the funnel of each consecutive top frame is compared with each other.

If the mean squared error (MSE) between two consecutive frames is above certain threshold, we can say that the wave is passing the funnel. This threshold is decided by looking into the histogram of the MSE of all consecutive top frames of the funnel. As the MSE is a pixelwise comparison, this approach is sensitive to the vibrations of the camera and light conditions of the building. For simplicity, the comparison was made in grayscale instead of RGB. The histogram of experiment T102 can be seen in Figure 12.

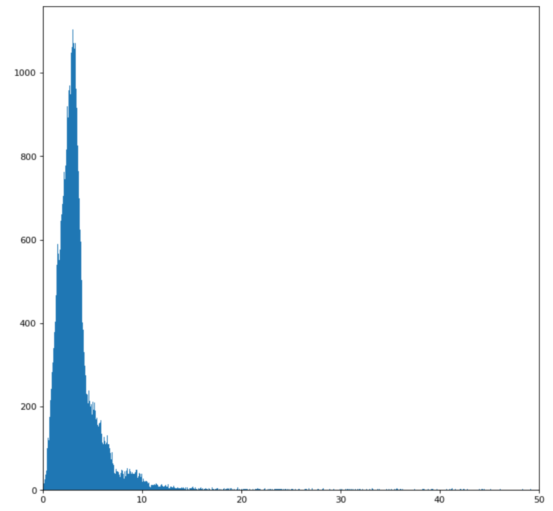


Figure 12: Histogram of the MSE of the grayscale top frame of the funnel.

Figure 12 shows that the MSE of the majority of consecutive pairs of frames are below 10. This indicated the threshold must be above 10. For T102, 15 was chosen as threshold. This method worked good for all experiments, except T103. During the experiment of T103, the light intensity was changing during the experiment which resulted into relative high false positives. In other words, the changes in light conditions resulted into the detection of a wave. But in reality, there was no wave occurring. Such cases were deleted manually. Furthermore, experiment T105 was not suitable for this research. The reason for this is that the top and side camera suddenly freezes at 20:36. Because of this, the cameras and water meter were not running on parallel anymore. Meaning that we could not see an increase in the box when a wave was occurring at a given timestamp. Thus extracting labels was not possible. Table 2 shows the number of waves per experiment passing the crest. In total there are 494 waves.

Experiment	Number of Waves
T102	101
T103	79
T104	60
T105	Camera and water meter not recording parallel
T106	96
T109	158

Table 2: Number of waves passing the crest per experiment.

These waves are extracted from the regarding videos. Each wave is a video of roughly 2 to 3 seconds. This video is converted into frames, ranging from 10 to 20. This amount can be increased, but this will result into similar images and will not provide significantly new information. These frames are resized by 80 percent resulting into 170 by 400 pixels. This resizing is done in order to make it computationally feasible. To rephrase, these frames together are causing the overtopping. This overtopping is measured by the (right) water meter in the box, see Figure 13. The measurements done by the water meter is fluctuating as the water is not standing still in the box. Especially during the occurrence of a wave.

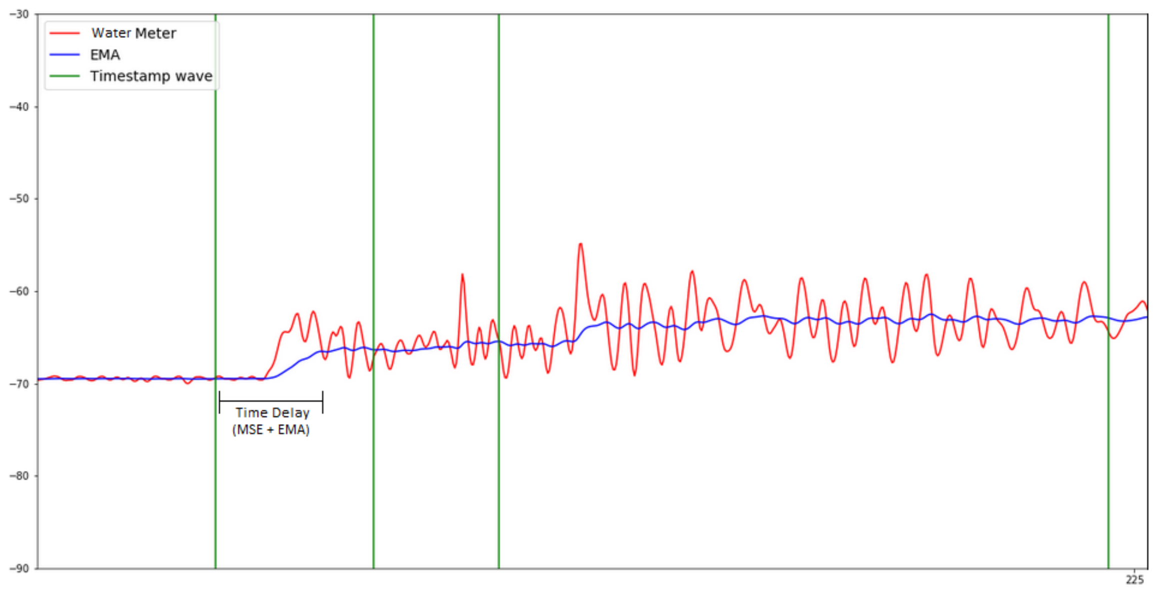


Figure 13: Red: overtopping measured by the water meter. Green: timestamp of a wave detected by the MSE method. Blue: EMA applied to the water meter.

Each green line represents the timestamp of a wave. These timestamps are used for the extraction of the video of 2 to 3 seconds. The observation indicates that the measurements done by the water meter increases later than the timestamp of a wave. This is due to the time delay between the wave detected by the MSE method and the water meter, see Figure 14.

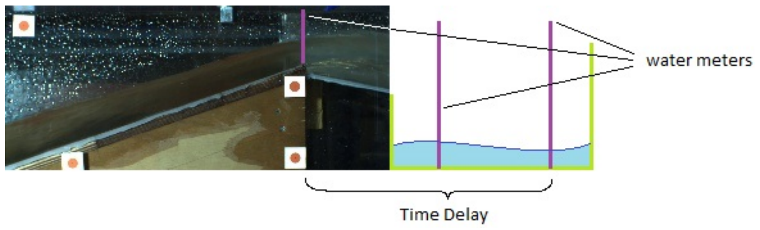


Figure 14: Time delay between the detection left (by the MSE) and right water meter.

To create the training set, it is important to have reliable labels. As mentioned above, a series of frames will cause a certain amount of overtopping. This certain amount of overtopping is not reliable as the measurements done by the water meter is fluctuating. To smooth the measurements done by the water meter, the exponential moving average (EMA) method is used. This is indicated with the blue line. The EMA at a timestamp is calculated with the formula below.

$$y = \frac{\sum_{i=0}^{t} w_i x_{t-i}}{\sum_{i=0}^{t} w_i} \tag{6}$$

Where y_t, x_t and w_t are respectively the result, input and its regarding weight. The weights are calculated by $w_i = (1-\alpha)^i$. α is the *span* and is also known as "N-day exponential weighted moving average" and is calculated by $\alpha = 2/(span + 1)$. During this research, $span = 80$. The higher the value for $span$, the smoother the (blue) line is. To be more precise, the result is calculated in the following way:

$$y_t = \frac{x_t + (1-\alpha)x_{t-1} + (1-\alpha)^2 x_{t-2} + ... + (1-\alpha)^t x_0}{1 + (1-\alpha) + (1-\alpha)^2 + ... + (1-\alpha)^t} \tag{7}$$

The distribution of the weights is exponential, see Figure 15. This means that the closer a data point is to a timestamp t (where we want to know the average from), the more it will contribute to the average.

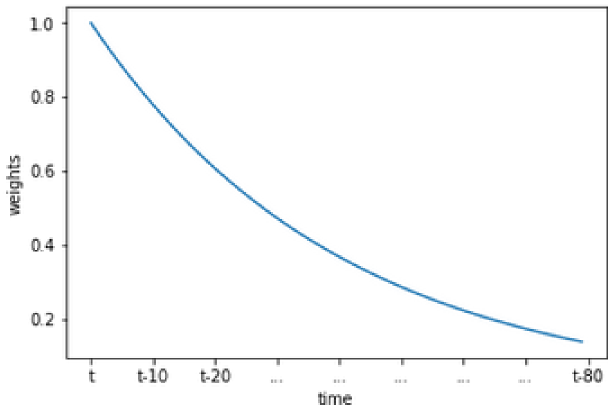


Figure 15: The distribution of the weights of 80 data points (taken over a period of 2 seconds).

The EMA and the MSE method are both causing a delay. This is shown in Figure 13. Note that this delay is until the EMA is stabilized to some extent and is thus definitely not the same for all waves. But to increase the reliability of the labels to some extent, this delay (for the green line) is set to 1 second, resulting into Figure 16. Important note is that in some experiments, e.g. T109, the green box in Figure 8 is being emptied for approximately 50 seconds when this is full. During this emptying, there were also waves occurring. These waves are not used in the training or test set as their labels are not reliable.

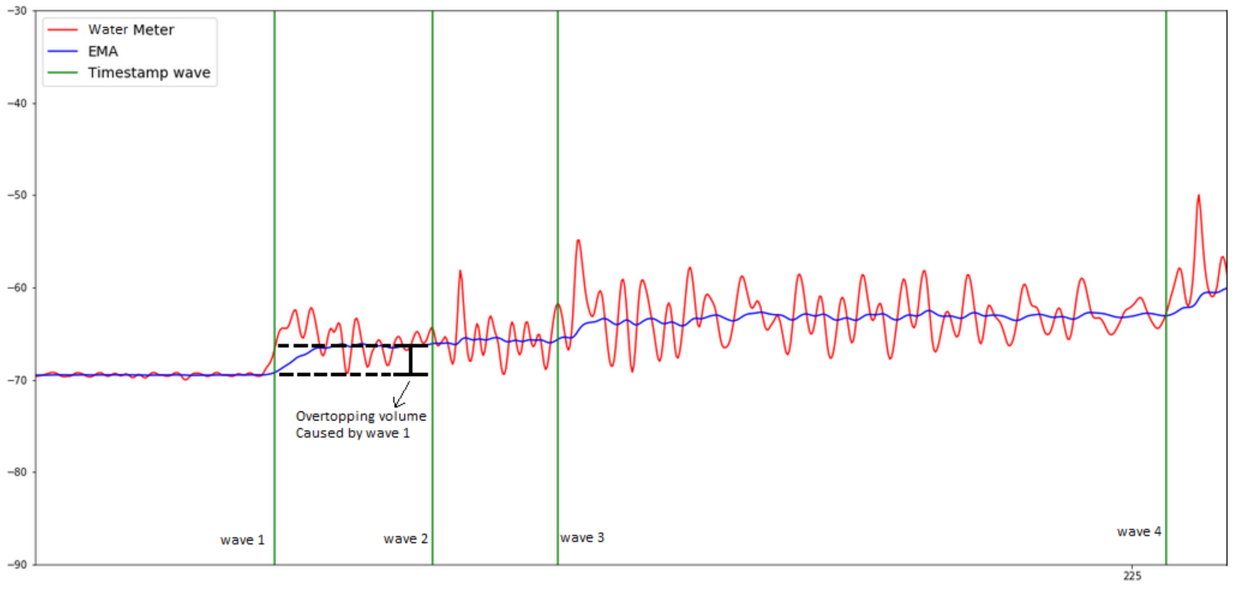


Figure 16: After adding the delay of 1 second to the timestamp of a wave, the EMA line increases short after this. The EMA is used to extract the labels for the waves.

One attempt to correctly calculate the time delay in Figure 14 was with the use of cross correlation between the MSE of the funnel and the right water meter. The right water meter can also be used to extract waves. A change in this water meter above certain threshold means that a wave is occurring. However, small waves which are passing the funnel with a low speed and are not triggering the water meter. Such waves are dripping down from the wall into the box. This increases the water in the box, but does not trigger the right water meter, resulting into wrong output of the cross correlation function.

Furthermore, the EMA method is chosen instead of the simple moving average (SMA) as recent events are more important for the average. In Figure 16, three consecutive waves can be seen. The first wave is a strong wave. The second one is a particularly weak one. However, the first wave might influence the second wave in terms of overtopping. By using the EMA method, the overtopping of a specific wave gets less influenced by the previous overtopping of a wave as these data points will count exponentially less in the average. The preprocessing steps above are done for all experiments. These data is stacked on top of each other resulting into the format as in Table 17.

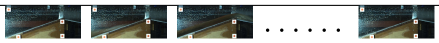
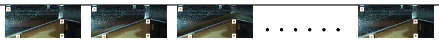
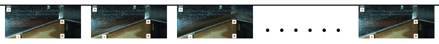
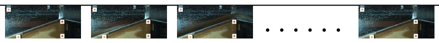
Input (length=2/3 seconds)	Waves converted into 20 frames	overtopping (in m^3/m)
path/wave1.avi		13.2
path/wave2.avi		10.9
path/wave3.avi		5.2
⋮	⋮	⋮
⋮	⋮	⋮
path/waven.avi		8.2

Figure 17: Videos of the waves with their regarding frames and labels (overtopping).

The column overtopping in Figure 17 is considered as the labels for the (frames of the) waves.

The labels of these waves depends on the method that is used to extract the overtopping for the waves. During this research, this overtopping is calculated with the EMA method and is considered as the ground truth. The data is split randomly to 70% training and 30% test set. The distribution of the labels (overtopping volume) can be seen in Figure 18.

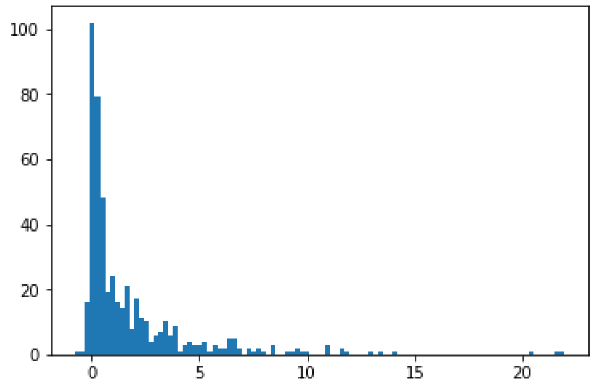


Figure 18: Histogram of the labels of 494 waves.

Most waves are causing an overtopping volume between 0 and 2.5. The distribution is right skewed. Most labels are between 0 and 5. There are also labels between 0 and -1 due to the fluctuations of the water. In order to train the network, the data is split in such a way that the distribution is similar to the complete data set, see Figure 19.

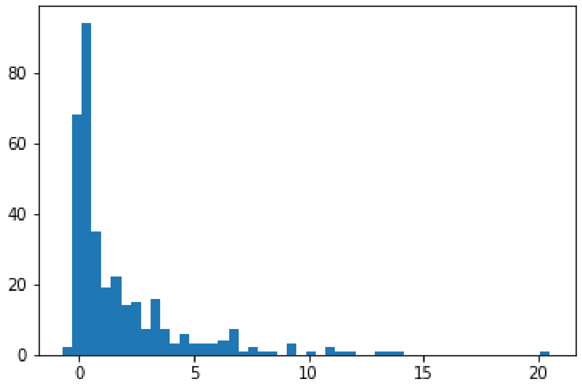


Figure 19: Histogram of the labels used during the training of the network.

3.3 Coding Environment

The preprocessing and the training of the network is done by transferring all the data to GPU enabled servers. Regarding the GPU, Nvidia 980 TI was used with CUDA 9.2.148 for training the neural network. On this machine, Jupyter Notebook with Python 3.4.8 was hosted. Via port forwarding, this Jupyter Notebook was accessible via the web browser on the local machine. This step was necessary as training the network took hours on the CPU.

4 Network Architectures

Once the training set is created, this is used as input for the neural network. A neural network works similar like a human brain. Neural networks are able to learn by looking to examples. These examples can range from images to voice recordings. A sample neural network can be seen in Figure 20.

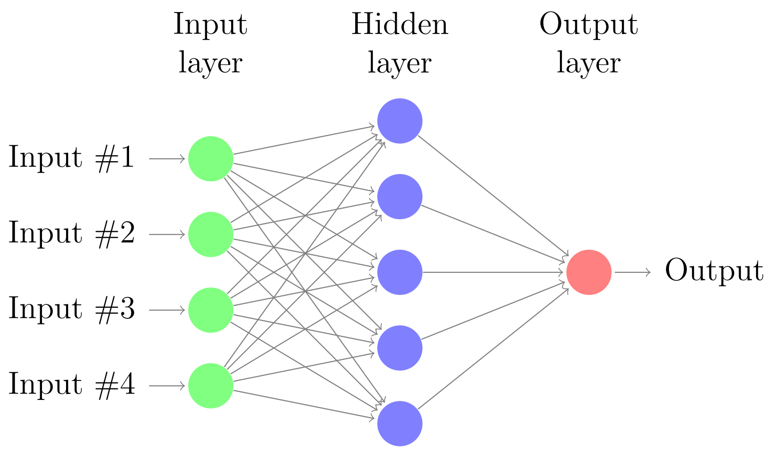


Figure 20: Sample (fully connected) neural network with input, hidden and output layer.

This network consists of input layer, hidden layer and output layer. The hidden layers is necessary to train a network which can learn non-linear patterns. Each circle (except of input layer) is a neuron. The edges are the weights of the network. Each neuron takes as input the multiplication of the output of the previous layer with the regarding weights, see In Figure 21.

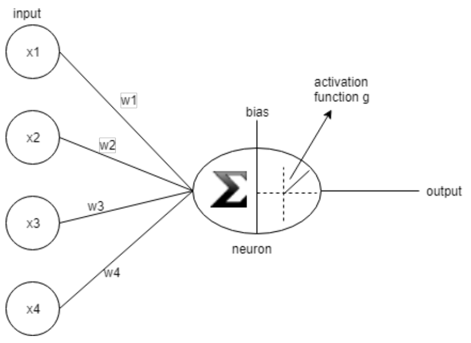


Figure 21: Single Neuron.

In matrix notation, we will have the following:

$$z = x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4 + b = X \cdot W + b \tag{8}$$

Where x_i, w_i and b are respectively the input, weight and bias. The sum of this is used as input for the activation function:

$$a(z) = \frac{1}{1 + e^{-z}} \tag{9}$$

The activation function above is the Sigmoid. But can also be a ReLu [Nair and Hinton, 2010] or Tanh [LeCun et al., 1998] function. This process is done for each neuron until the output layer. Once the output is calculated, this will be compared against the ground truth. By using a loss function (for example MSE), the error can be calculated. This error is used to update the weights of the network by using the backpropagation algorithm. During this backpropagation, the gradients are computed with respect to the weights. These weights can be stuck into a local optimum. This local optimum can be avoided by for example using higher learning rates. High learning rate will lead into bigger update of the weights. However, these gradients can get smaller and smaller when moving to the input layers. These gradients will eventually get close to zero which will lead to small updates of the weights. This means that the network will learn slowlythus converge slowly. This problem is also known as the vanishing gradient. One way to solve this problem is to use the rectified linear unit (ReLu) activation function:

$$g(z) = max(0, z) \tag{10}$$

According to the research of [Krizhevsky et al., 2012], the ReLu activation functions converges six times faster (until 25% training error rate) than an equivalent network with Tanh activation functions. Beside this, the ReLu is computationally faster than the Sigmoid as the ReLu only selects the maximum between two numbers whereas the Sigmoid is performing exponential calculations. Furthermore, the weights can be updated by using for example Stochastic Gradient Descent (SGD) optimiser. During this research, two architectures (CNN and CNN LSTM) are compared with each other. During this research, different setups will be compared. This can be read in the next section.

4.1 Convolutional Neural Network

A CNN is a neural network as described in the previous section, but it has preceding convolutional layers. More about this can be read in [LeCun et al., 1990]. CNNs are mainly used for image recognition or classification tasks, but is also used for natural language processing. In most studies about CNN, single frames are used as input for the network. In this research the dataset consist of a time series of frames. This can be represented as a cube. The architecture of the CNN that is being used, is provided in Figure 22.

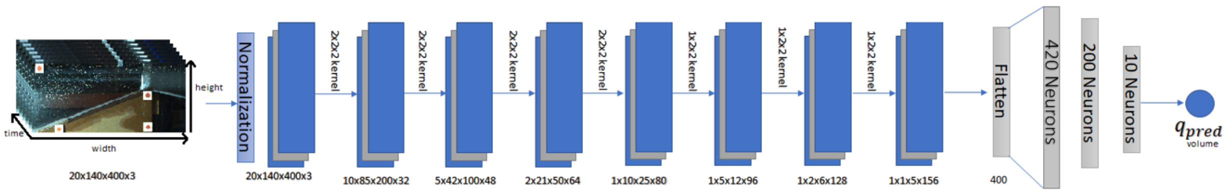


Figure 22: Architecture of the CNN with 3 dimensional kernels.

The input is a cube with 4 dimensions: [frames, height, width, channel]. In this case, this is [20,170,400,3]. The input is normalized by dividing it by 255. The weights of the kernels are randomly initialized. As [Bengio, 2012] mentions, weights need to be initialized carefully to break symmetry between hidden units of the same layer. If these hidden units have the same input and weights, it will compute the same output and gradient which results into the same update. This will not lead to any changes and is only a waste of capacity. The weight of both networks (CNN and CNNLSTM) is initialized by using the truncated normal distribution. Furthermore, we are interested in the boundary of the wave. As a human, the boundary of a wave is important as this is used to approximate the overtopped volume. In order to preserve this important information (boundary of the wave), kernels and strides are chosen to be below 2 by 2 by 2. The dimensions in the next layer is calculated with the formulas below:

$$output\,width = \frac{W - F_w + 2P}{S_w} + 1 \tag{11}$$

$$output\,height = \frac{W - F_h + 2P}{S_h} + 1 \tag{12}$$

Where W and H are the width and height. S_w and S_h are the horizontal and vertical slide. P is the zero padding that is used. The same formula is applied for the third dimension: time. Bigger kernels and strides are considered, but not used. The reason for this is that the network might lose a lot of information in the layers with big kernel or striding sizes. After each convolutional layer, batch normalization is applied before the activation function. This has several benefits as mentioned in [Ioffe and Szegedy, 2015]. Higher learning rates can results into the vanishing or exploding gradient problem. By using batch normalization, the network is normalized after each convolutional layer. This prevents gradients to vanish or explode. Furthermore, it prevents overfitting by regularizing the model. After the convolutional layers, the data is flattened into an array with a length of 400. This is used as input for the fully connected layers. The fully connected layer consists of input, hidden and output layer (with 420, 200 and 1 neurons). According to the research of [Srivastava et al., 2014], dropout effectively prevents overfitting. Dropout is a technique where a neuron and its incoming and outgoing connections are temporarily removed from the network. This forces the network to learn to produce the outcome by using other neurons. The results of this architecture can be seen in the next subsection.

4.1.1 Experiments

The architecture above is used as a baseline. On top of the architecture, different (hyper)parameters are used to find the lowest error for the test set. The (hyper)parameters that are fixed during this research are:

- Batch size: 20
- Epoch: 100
- Learning rate: 0.005
- Training and test set resp. 345 and 149 video segments of 2 to 3 seconds
- Linear activation function in the output layer
- ReLu activation function in the remaining (convolutional and fully connected) layers.
- Weight initialization by using the truncated normal distribution
- Size of kernels and strides

Furthermore, [Ruder, 2016] did a research about the application of different optimization algorithms. The conclusion is that the adam optimizer is best overall choice. Ruder also mentions that many studies are using the vanilla stochastic gradient descent (SGD) without momentum and a simple learning rate annealing schedule. According to his research, SGD might perform better. But the main drawback is that it can take significantly longer as it is relies heavily on a robust initialization. The bottleneck of the SGD is that it can get stuck in a saddle point. Therefore, an adaptive learning rate (adadelta, rmsprop, adam) is advised if a fast converging network is desired. Figure 23 shows how the MSE evolves during the training of the network by using the training set. The abbreviations are in the following way:

- LR: learning rate
- adam: adam optimizer
- BN: batch normalization
- D: dropout rate
- FC: fully connected layer
- SGD: stochastic gradient descent
- NM: Nesterov momentum

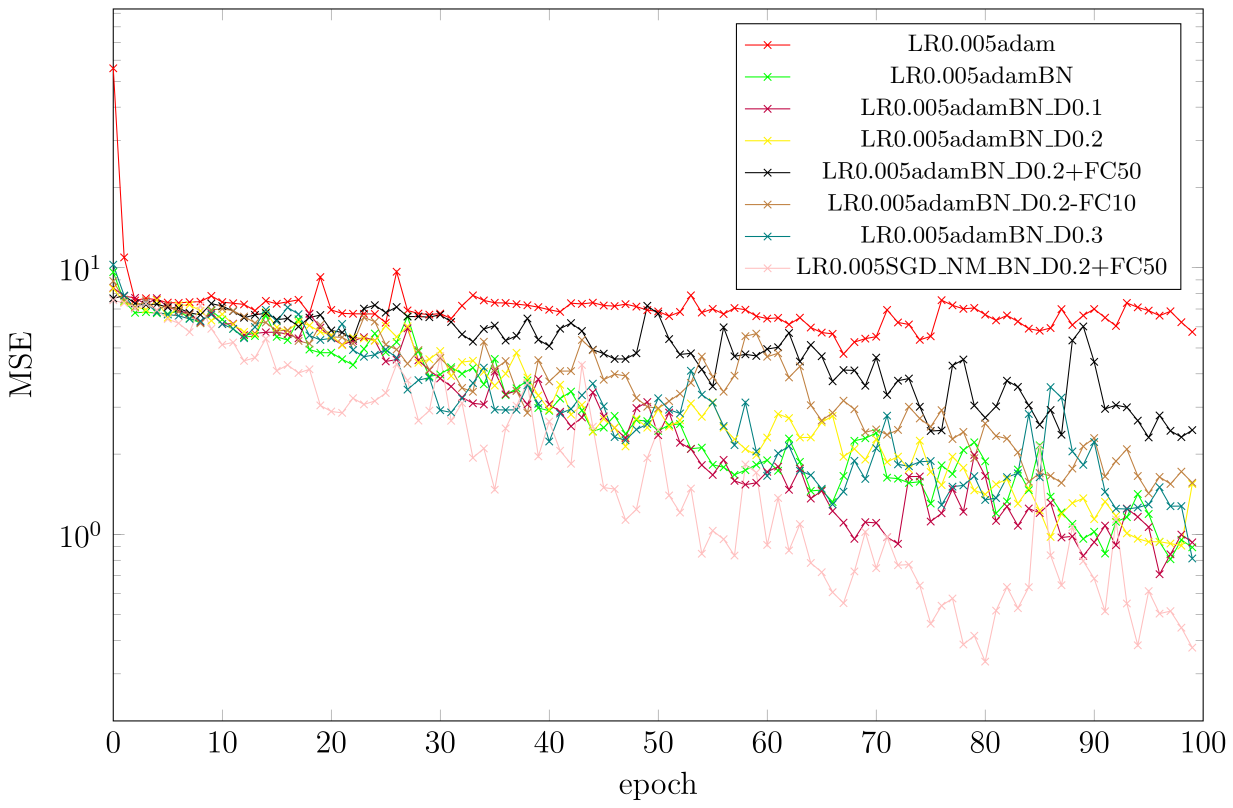


Figure 23: MSE of the training set during the training of the network in Figure 22.

During the experiment, mainly a learning rate of 0.005 is used. Higher learning rate caused high fluctuations in the loss functions whereas a lower learning rate was converging slowly. The graph above shows that batch normalization was necessary to get a lower error on the training set. The dropouts for the other models is only used in the fully connected layers, except the hidden layer with 10 nodes. Furthermore, a model with an additional hidden layer of 50 neurons is tried. But also a model without the last hidden layer of 10 neurons. This is mentioned with respectively *+FC50* and *-FC10* in Figure 23. The graph shows that the model, which uses the SGD optimizer in combination with Nesterov momentum with value of 0.9, is converging the fastest. Note that this is for the training set. The performance of the models applied on the test set is provided in Figure 24.

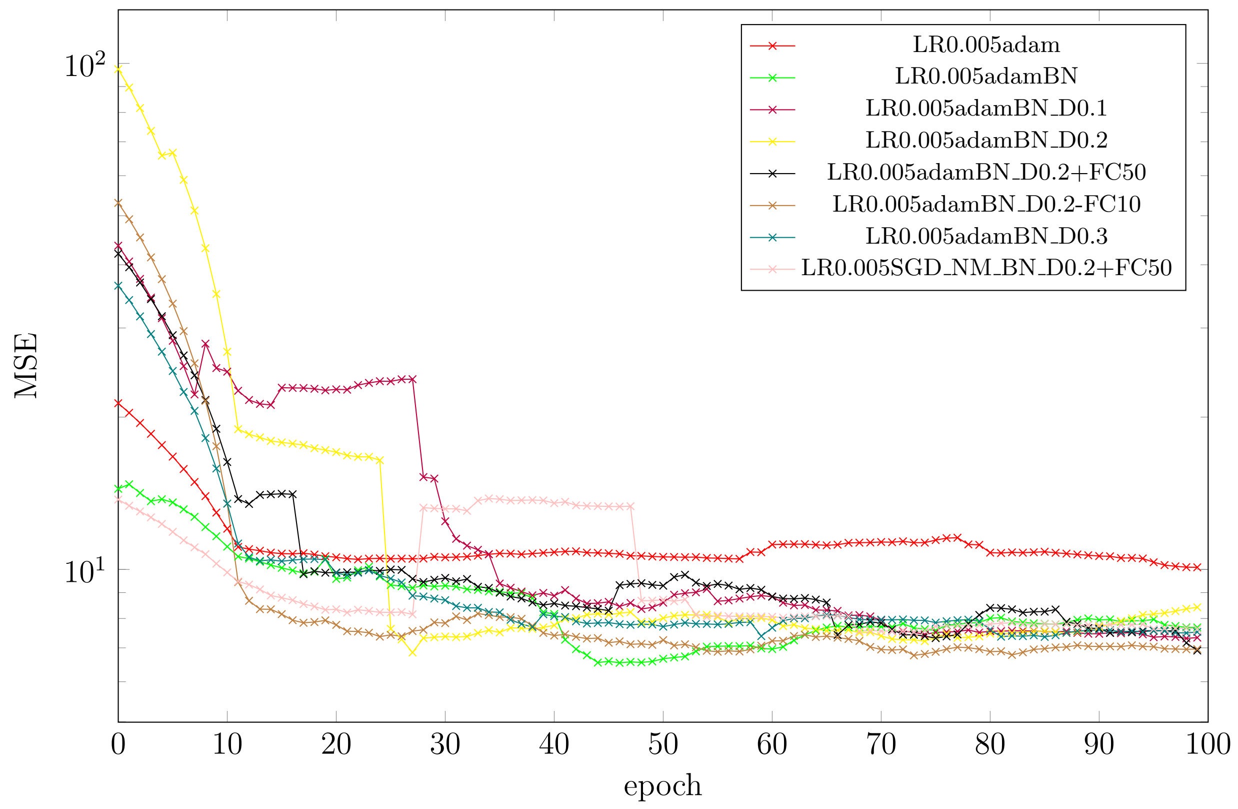


Figure 24: MSE (with SMA of 20 epochs) of the test set during the training of the network in Figure 22.

The graph above contains the performance of the models applied on the test set. The models are only run once, meaning that there is no strict order between them. However, it still provides us a nice overview about the capabilities of these models. To make them comparable with each other, the simple moving average of 20 epochs is used (as the values of the MSE were fluctuating and making the graph visually less appealing). We can again see that batch normalization is necessary to have a low MSE. Furthermore, a dropout rate of 0.2 with an additional hidden layer of 50 neurons (*+FC50*) was found to be the best for this architecture. However, the model with a dropout rate of 0.2 without the hidden layer of 10 neurons, did perform approximately the same. The best models were achieving a MSE between 5 and 7 m^3/m. The latter is that the adam optimizer is outperforming the SGD significantly within 100 epochs.

4.2 CNN combined with LSTM

The architecture of a neural network depends on the context of the problem and the type of the data. During this research the data consists of a series of frames, resulting into some overtopping volume. It might be important to consider the temporal information between these frames. The way a wave evolves, can have an impact on the overtopping volume. Furthermore, as mentioned in the beginning, some the bubble parts are contributing less to this overtopping volume than the remaining part. These bubbles appear white on a image and disappear faster than the *core* part of the wave in the next frame. A network that might be suitable to this context is the LSTM network. Some of the applications of LSTM are time series prediction, speech or hand writing recognition. A LSTM network is suitable when there is a dependency between the input sequence. In other words, it can model the temporal aspect of the data. The LSTM architecture was developed in order to deal with the vanishing and exploding gradient problem, which is mainly occurring in recurrent neural networks (RNN). More about the (first) LSTM network can be read in [Hochreiter and Schmidhuber, 1997]. In this section, we analyze the effect of adding the LSTM to the output of the convolutional layers. At the end a fully connected layer is used. This architecture is shown in Figure 25.

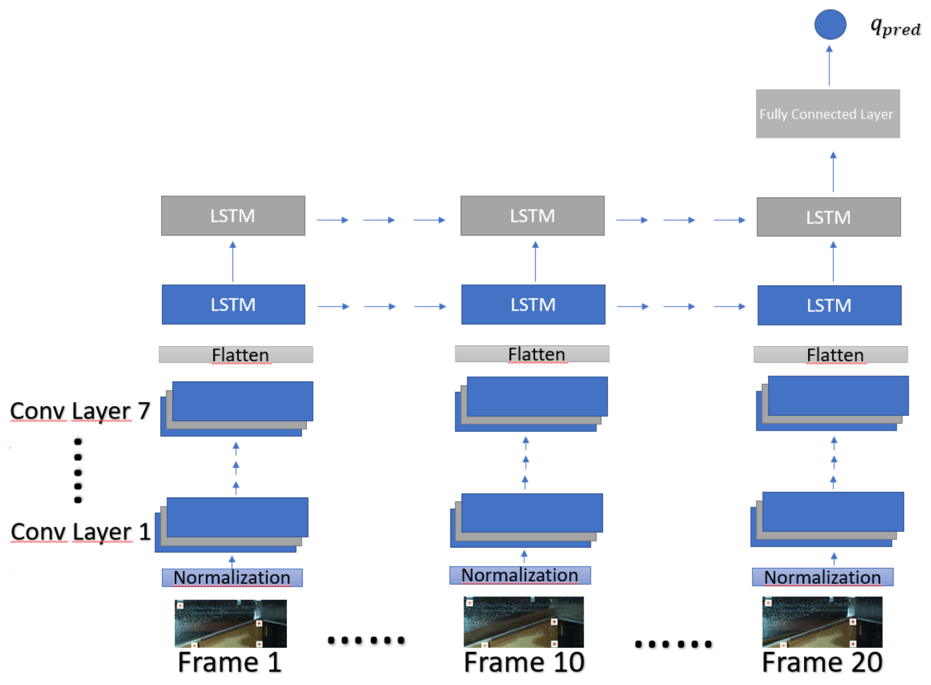


Figure 25: The architecture of the CNN LSTM model. LSTM is applied to the output of the convolutional layers.

Each frame of a wave is first normalized and then used as input for the convolutional layers. These layers are using a kernel size smaller than 3. The output of the convolutional layers is flattened and used as input for the LSTM layer. The LSTM layer consist of two layers of 64 cells. After the LSTM layer, a fully connected layer (of 500 neurons) is used to predict the overtopping volume. All layers are using the ReLu activation functions, except the last neuron. This does not use an activation function.

4.2.1 Experiments

The architecture above is used as a baseline. On top of the architecture, different (hyper)parameters are used to find the lowest error for the test set. The (hyper)parameters that are fixed (unless stated otherwise) are:

- Batch size: 20
- Epoch: 100

- Learning rate: 0.1
- Adadelta optimizer
- Training and test set resp. 345 and 149 video segments of 2 to 3 seconds
- ReLu activation function in all layers, except the output layer.
- Weight initialization by using the truncated normal distribution
- Two LSTM layers each consisting of 64 cells
- Fully connected layer with 500 neurons

The size of the kernels used in this architecture is chosen to be 3 or less. Batch normalization is applied before the activation function. After the convolutional layers, two layers of 64 LSTM cells are used. This approach is applied to the 20 timesteps (1 wave = 20 frames). The output of the LSTM layer is used as input for the fully connected layers. The fully connected layer consists of 500 neurons and calculates one output: overtopping volume. To clarify, the network is back-propagating by only using the overtopping volume of the wave. Intermediate overtopping volumes during the occurrence of a wave (from $t = 1$ to $t = 20$) is not used for back-propagation as it was not possible to extract these from this experimental setup. Figure 26 shows how the MSE evolves for the training set. The abbreviations are in the following way:

- LR: learning rate
- BN: batch normalization
- D: dropout rate fully connected layer
- RD: recurrent dropout rate
- DLSTM: dropout rate of the lstm cell
- FC: fully connected layer
- SGD: stochastic gradient descent
- NM: Nesterov momentum

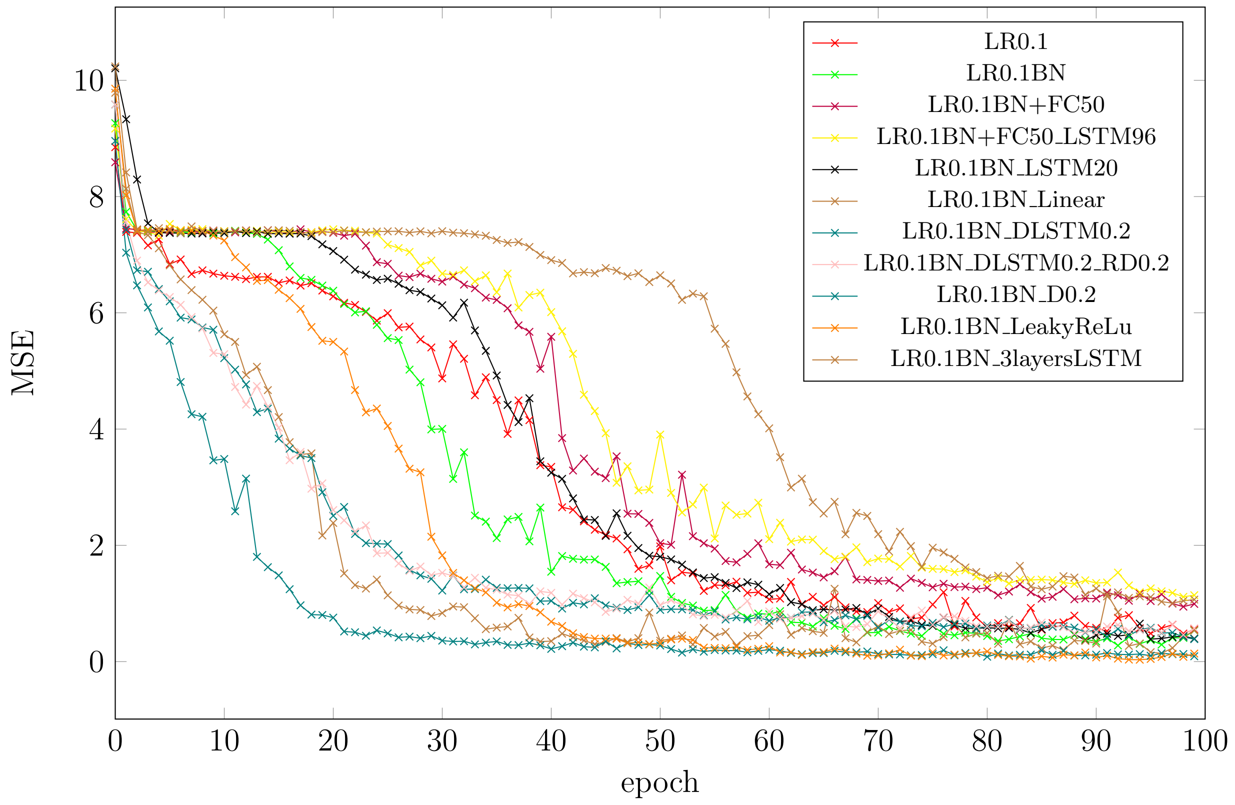


Figure 26: MSE of the training set during the training of the network in Figure 25 by using CNN LSTM architecture.

During this experiment a learning rate is used of 0.1 in combination with the adadelta optimizer. This model is still converging without the batch normalization. Figure 26 shows that the model with the Leaky ReLu activation function, and the model with a dropout rate of 20% in the fully connected layer, is performing the best regarding the training set. Furthermore, three layers of 64 LSTM cells is converging slowly and obtains one of the highest MSE for the training set. This is also the case for using an additional fully connected layer of 50 neurons in combination with two layered 96 LSTM cells. Note that this is for the training set. Figure 27 shows the performance of the same models, in the same color, for the test set.

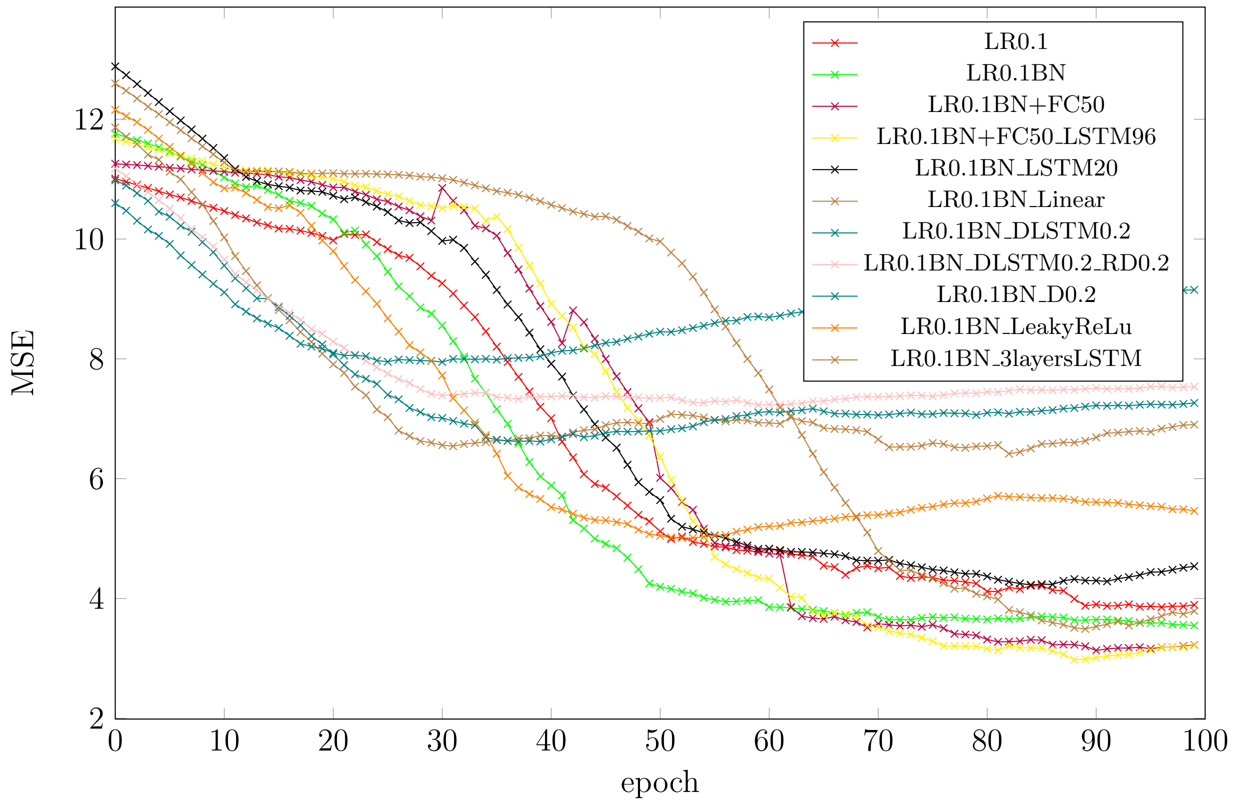


Figure 27: MSE (with SMA of 20 epochs) of the test set during the training of the network in Figure 25 by using CNN LSTM architecture.

Figure 27 shows that the baseline architecture without batch normalization achieves a MSE of approximately 4 m^3/m. However, using batch normalization in the convolutional layers is increasing the performance significantly. The application of the dropout technique (whether this is in the recurrent state, input of the LSTM cell or the fully connected layers after the LSTM layers) does not improve the MSE of the models. We can also see that Leaky ReLu activation function is performing significantly worse compared to the ReLu activation function, despite the fact that Leaky ReLu was (over)fitting the training set. Using two layers of 20 LSTM cells caused a slightly higher MSE while using two layers of 96 LSTM cells did not increased the performance. One of the best models was with batch normalization and an additional fully connected layer of 50 neurons. But in general, the best models were achieving the MSE between 2.5 and 3.5 m^3/m.

5 Results

In this section the results of the best CNN and CNN LSTM network will be compared with each other. To clarify, the test set is the same for both networks. An overview will be shown how the models fit to the training and test set. We will also compare the fit of the models on waves with a small and big overtopping volume. On the end, the outliers of the best model will be analyzed and some technical limitations are addressed.

5.1 Results CNN

One of the "best" CNN models with a learning rate of 0.005, batch normalization, dropout rate of 0.2 and an additional hidden layer of 50 neurons, is shown in Figure 23. "Best", because this models is not strictly the best as the models are only ran once. The results of a model with a MSE of 6.768 m^3/m, is provided in Figure 28.

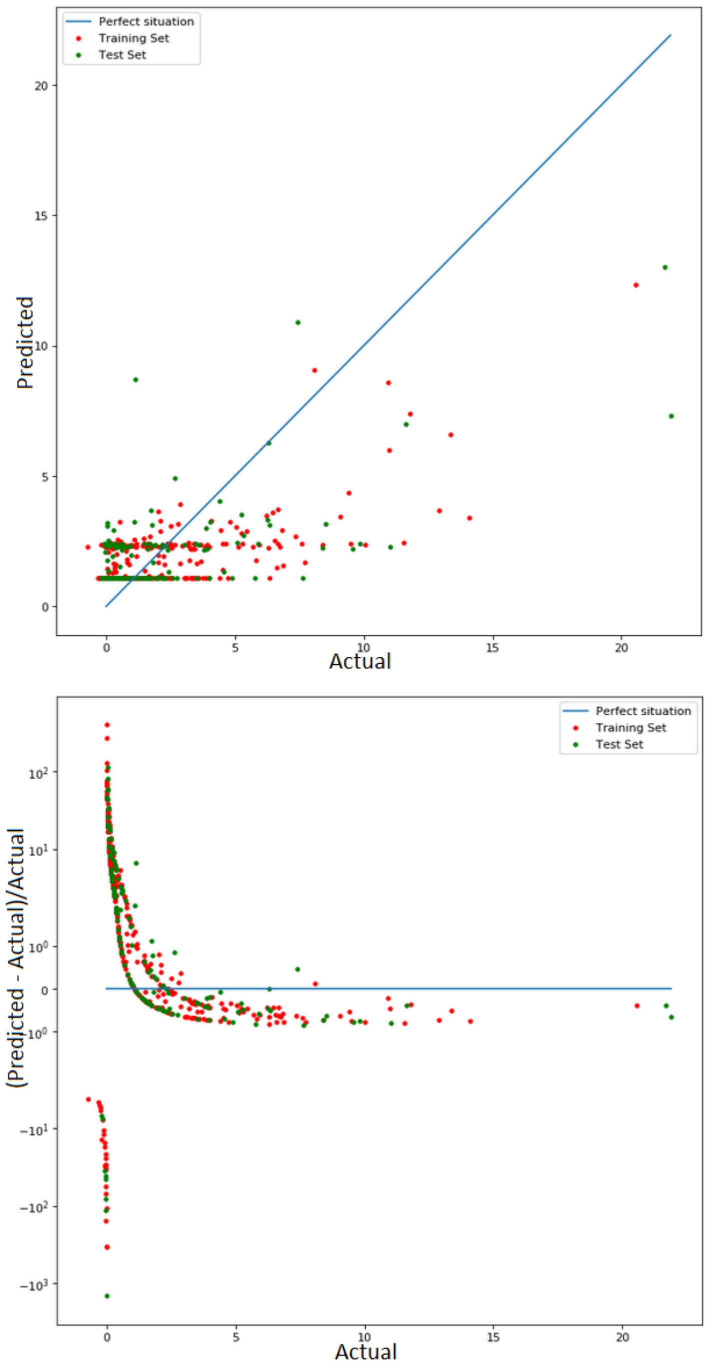


Figure 28: Each point represents a wave of 2 to 3 seconds. Red and green points are respectively the training and test set. Above: Actual vs Predicted plot. Below: Actual vs (Predicted - Actual)/Actual. Results are obtained by using a model with a MSE of 6.768 m^3/m, learning rate of 0.005, batch normalization, dropout rate of 0.2 and additional hidden layer of 50 neurons.

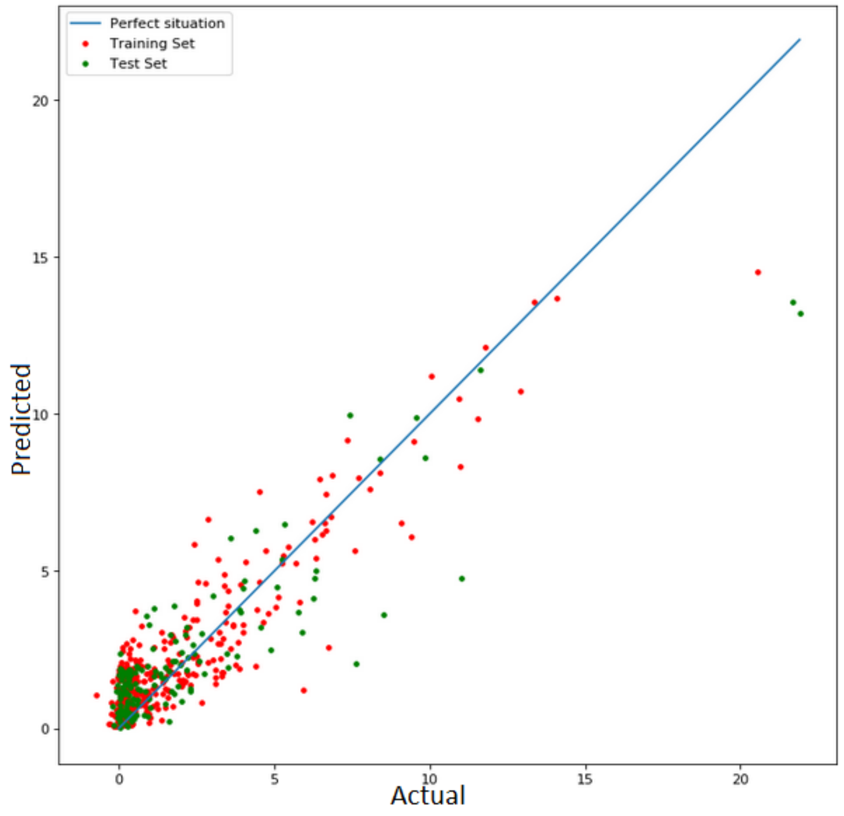
Plot above: ideally, we would like to be as close as possible to the blue line. However, the model is overfitting when all the red points are on the blue line. The overfitting is avoided to some extent by using batch normalization, ReLu activation function and dropouts. The results show that both waves (training and test set) are not fitting the blue line. There are many waves which are over- or underpredicted. Another note is that the two rightmost green points are underpredicted as well. But this is quite normal as these type of waves (resulting into big overtopped volume) is only occurring once in the training set. To clarify, the results above are just an example of a trained network with a MSE of 6.768 m^3/m. This network can give different results every time when its trained from the beginning. However, the MSE will be approximately the same. The top 3 performing CNN models have a MSE between 5 and 7 m^3/m. The plot below shows how much the predicted value deviates from the actual value, normalized by its actual value. The observation indicates that this model is not suited for the prediction of the overtopped volume of waves. Especially waves with a small overtopped volume are over- and underpredicted compared to its actual value. The CNN model approaches the horizontal and vertical asymptote of -1 and 0. The bigger the overtopped volume of a wave is, the closer it is to the asymptotes. The observations indicate that this model is not able to extract the features to predict the overtopped volume.

5.2 Results CNN LSTM

Figure 29 shows the predictions of one of the best trained CNN LSTM models. This model has an additional fully connected layer of 50 neurons and has a MSE of 2.758 m^3/m. There are multiple models which are performing approximately the same. These are the following:

- Batch normalization, two layers of 64 LSTM cells,
- Batch normalization. two layers of 64 LSTM cells and additional fully connected layer of 50 neurons, see Figure 29 and
- Batch normalization. two layers of 96 LSTM cells and additional fully connected layer of 50 neurons.

These models are achieving a MSE between 2.5 and 3.5 m^3/m. The plot above shows that waves with a low overtopped volume are close to the blue line. However, these labels are small and look extraordinary close to the blue line as the range of the axes increases. The plot below provides how much the predicted value deviates from the actual value, divided by the actual value. Especially waves with a small overtopped volume are over- and underpredicted. The higher the overtopped label of a wave is, the closer its prediction is to the blue line. The results indicate that this experimantal setup and CNN LSTM model is more suited for waves with a significant amount of overtopping volume. Some of the outliers in Figure 29 will be discussed in subsection 5.4.



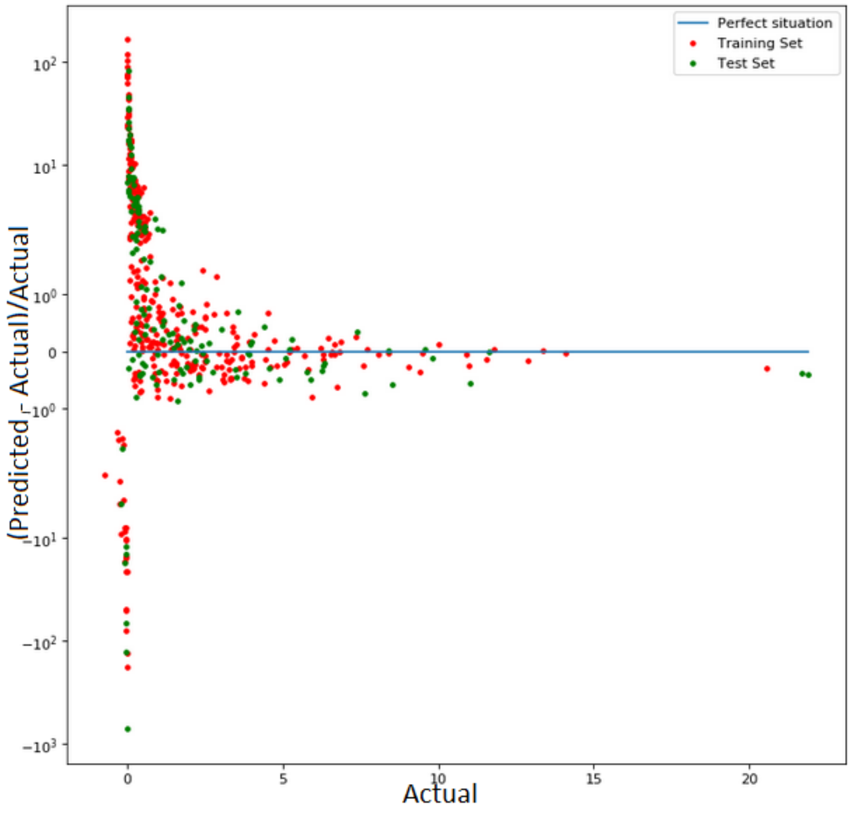


Figure 29: Each point represents a wave of 2 to 3 seconds. Red and green points are respectively the waves in training and test set. Above: Actual vs Predicted plot. Below: Actual vs (Predicted - Actual)/Actual. The results are obtained from a model (MSE of 2.758 m^3/m) with an additional hidden layer of 50 neurons and two LSTM layers of 64.

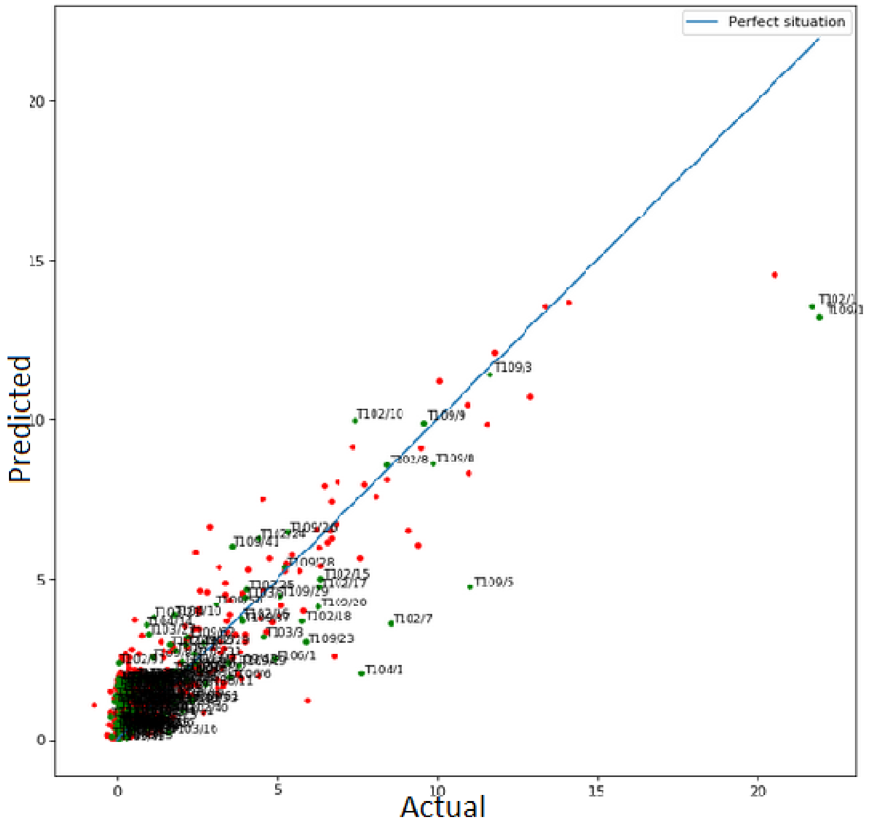
5.3 Comparison of the two architectures

The top 3 CNN models mentioned in the section above resulted into a MSE between 5 and 7 m^3/m. While the CNN LSTM models achieved a MSE between 2.5 and 3.5 m^3/m. This can also be seen from the plots above in Figure 28 and 29: the green points are closer to the blue diagonal. There are also less outliers for the CNN LSTM. The plots below in Figure 28 and 29 indicates that both models (CNN and CNN LSTM) or that the experimental setup is less suitable on waves with a small overtopped volume. The CNN model approaches the horizontal and vertical asymptote of -1 and 0 as the overtopped volume increases. Regarding the CNN LSTM model, the point above and under the blue line are better distributed. It also gets closer to the blue line as the overtopped volume of a wave increases. The CNN LSTM clearly outperforms the CNN, especially for waves with a bigger overtopped volume. Note that higher performance regarding the MSE could be achieved for both models by trying out different architectures, parameters, adding more waves or modifying existing waves by rotating and shifting.

Another important point is the interpretation of the trained networks. The two trained networks can be analyzed further by using for example layer-wise Relevance Propagation (LRP) [Bach et al., 2015]. Both networks can perform the same in terms of MSE. But the approach can be different. The CNN might use other pixels to calculate the MSE than the CNN LSTM network.

5.4 Outlier Analysis

In this subsection the outliers are analyzed of the best fitting model. This is the CNN LSTM model with batch normalization, two layers of 64 LSTM cells and additional fully connected layer with 50 neurons. Figure 30 provides two plots with outliers and the name of the experiments where they occur. The plot above is the same model as in subsection 5.2 with a MSE of 2.758 m^3/m. When rerunning the same model, it is possible to obtain other outliers. This is shown in the plot below in Figure 30 and has a MSE of 3.02 m^3/m.



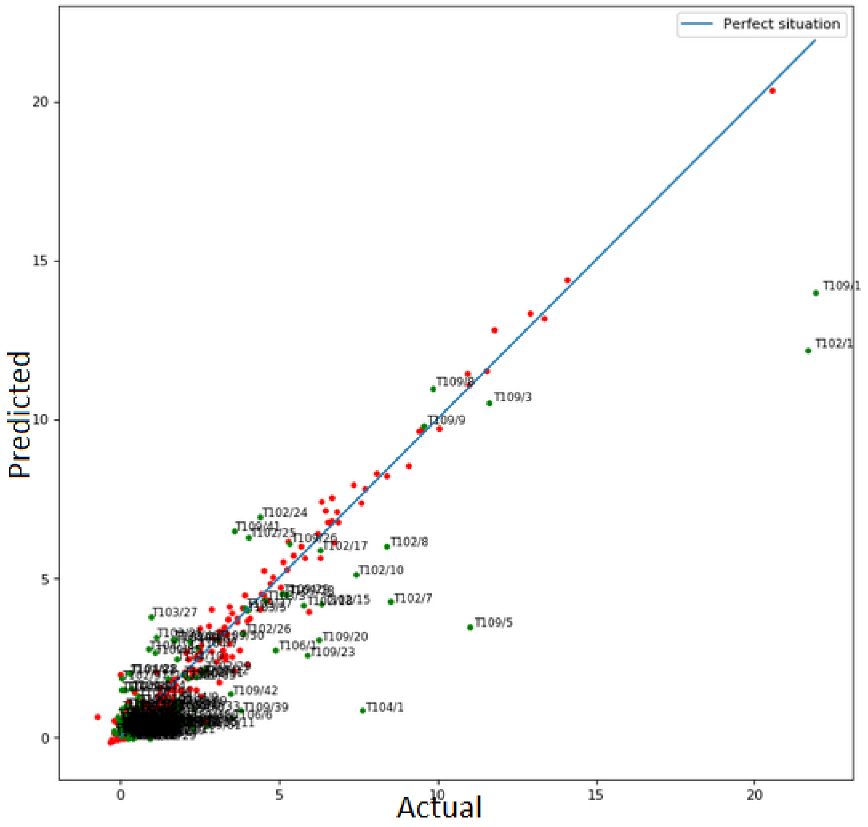


Figure 30: Each wave marked with the experiment they occur with their regarding rank. The plot above (from subsection 5.2) and below have a MSE of respectively 2.758 and 3.02 m^3/m. Both plots are obtained by using the same model with an additional hidden layer of 50 neurons and two LSTM layers of 64.

Each wave is followed with the name of the experiment which they occur with their regarding *rank*. We can for example see wave T109/5. This is the 5-th strongest wave of experiment T109. A couple of outliers can be derived by looking to both plots:

- T102/7,
- T102/24,
- T103/21,
- T103/27,
- T104/1,
- T109/5,
- T109/20 and
- T109/23.

T102/1 and T109/1 are not really outliers as there is only one other wave in training set which has a similar overtopping volume. However, the rest are outliers in this case. Regarding the outliers below the blue line, these might be cases where the wave is weak from the side camera. While, when looking from the top camera, it might be the case that it is a strong wave. Thus the width of a wave can also contain useful information. However, there might be other features that are missed by only using the side camera. An example of frames of two outlier waves approaching the crest is provided in Figure 31.

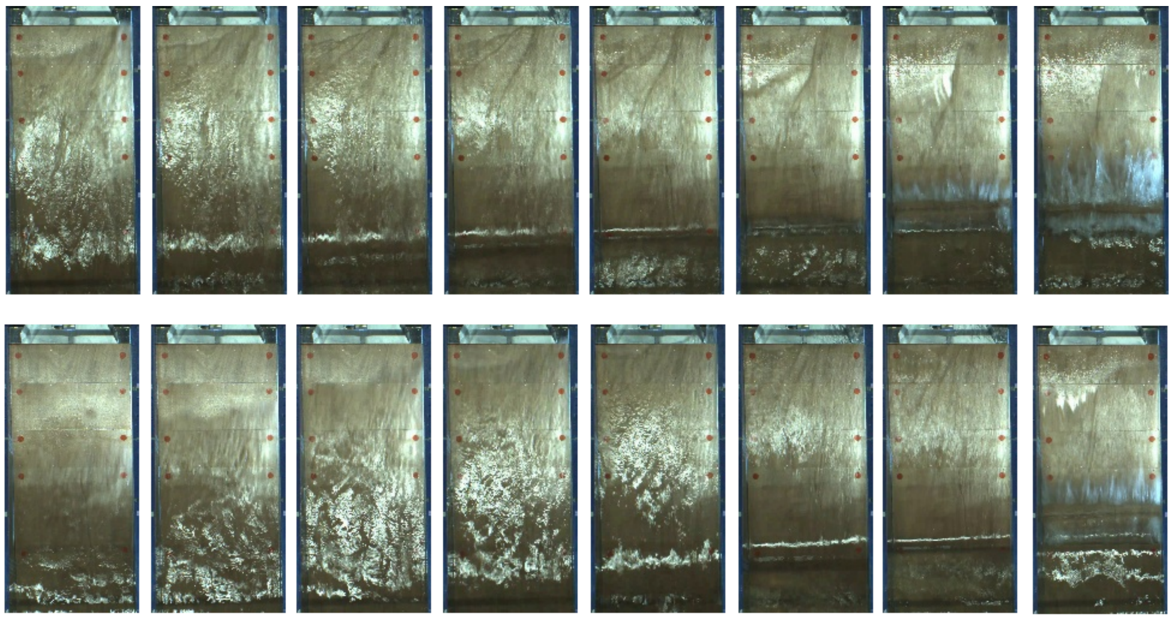


Figure 31: Row 1: 21st-strongest wave from experiment T103. Row 2: 27th-strongest wave from experiment T103.

In some situations, the right side of the wave is stronger than the left side. As the camera is on the right side, the left side (when looking from the top) is not being recorded. Thus from the side camera, this wave looks strong. This is also the reason why this is significantly overpredicted. Another outlier can be seen in Figure 32.

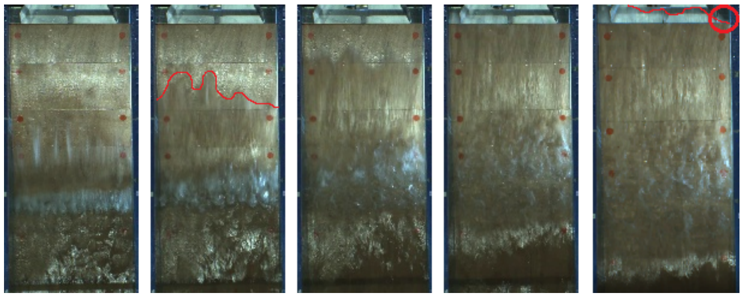


Figure 32: The approach of strongest wave from experiment T104.

In Figure 32, the left part of the wave is stronger. In the second frame, the border of the wave is shown. The wave has a higher speed on the left side. In the last frame the right side of the wave just reaches the red circle while on the left side, it already passed the crest. The cases above are waves which are strong on the right or left side. There are also waves which are strong on the left side, but still being overpredicted. Said otherwise, there might also be other features causing the wrong predictions. These outliers can distort the trained model and cause a high MSE. Rerunning the network without these outliers is considered, but this would not represent the reality. Because of this reason, outliers are kept in the dataset.

5.5 Comparison with the existing neural network

The neural network used in the EurOtop manual is using a multi-layer perceptron model. As input, some of the parameters are used from Table 3 (in total 15). It would be useful to compare the CNN LSTM results with the results of the EurOtop manual. Unfortunately, this can not be done during this research. The first reason is that they are calculating the average overtopping discharge rate over the entire duration of the experiment. While in this research, the overtopped volume per wave is calculated. Secondly, the EurOtop neural network uses the parameters in Figure 3 as input, while in this research, the inputs consists of raw frames or pixels. The last reason is that the side camera only records a part of the total crest. The location of all of the parameters (used in EurOtop manual) are not visible in the video. These are the reasons why the comparison is not possible.

5.6 Limitations

As can be seen from Figure 13, the measurements (thus labels) done by the water meter is noised by these fluctuations. The EMA method is applied to obtain more reliable labels. These labels are considered as the ground truth. Furthermore, the CNN and CNN LSTM network used in this research are reproducing an expected outcome. Thus the predictions will be limited by the capabilities of these devices. The network will not be able to predict better than the devices and methods used in this research. To improve the predictions significantly, it is advised to use different methods and/or devices to extract the labels of the waves.

The second limitation is that the current network is trained in an environment where there is no wind. However, when there is wind, this is not visible at all from the side or top camera. Meaning that it can potentially distort the current model.

6 Conclusion

As mentioned in the section Related Work, a high degree of literature studies solely considered coastal structures in combination with neural networks with a multi-layer perceptron model. The only literature about coastal structures and deep learning that could be found, was from [Hydralab, 2018]. In this study, problems with the boundaries of a wave were addressed. It assumed the splash, bubbles and water drops as a part of the wave. As this is not a binary problem, we approached this by first creating a deep neural network that was able to understand what is happening in these sequence of frames. Using this approach, an answer is provided to the main research question:

Can we get a good approximation of the overtopping volume q_v of (plunging) waves in a flume, by applying deep learning algorithm(s) on these video streams?

Two deep learning architectures are the convolutional neural network (CNN) and the convolutional neural network in combination with the long short term memory network (CNN LSTM). The difference between the two networks is that the CNN LSTM captures the temporal sequential information. The observation indicates that the CNN LSTM network can predict the overtopping volume considerably better than the CNN. The results are satisfactory. Nevertheless, improvements can be achieved in this area. For example, by using a bigger dataset, crest with a rubble mount, other deep learning architectures or even running more epochs. Also, the preprocessing can be done differently. During this research MSE and EMA methods are used to extract the labels. However, other preprocessing steps might show considerable improvement regarding label reliability. By measuring the performance of the CNN and CNN LSTM architecture, we were able to answer the sub research questions:

How does the CNN perform in the prediction of the overtopping volume q_v?

The top three models in this research are achieving a MSE between 5 and 7 m^3/m, including outliers. Many outliers were observed in the actual vs. predicted plot in Figure 28. Many waves were significantly over- or underpredicted. The model did not perform well on the training and test set. Subtracting the actual value from the predicted value, and normalizing by its actual value, we derive that this model is not suited for making the predictions. Especially for waves with a low overtopped volume.

How does the CNN LSTM perform in the prediction of the overtopping volume q_v?

CNN LSTM performs significantly better than the CNN. This model achieves a MSE between 2.5 and 3.5 m^3/m, including outliers. When looking to the actual vs. predicted graph (Figure 29) and comparing this with the CNN, we can see that this model has fewer outliers. From the view of the top camera, the CNN LSTM outliers were, in some cases, stronger on either the left or right side. This is not always the case, however. Other features can also cause wrong predictions. In some situations, a wave is strong on the left side, weak on the right side, and still overpredicted. Subtracting the actual value from the predicted value, and normalizing by its actual value, we derive that this model is not suited for the prediction of waves with a low overtopped volume. The predictions are closer to the blue line as the overtopped volume of the waves increases.

In a recent study by [Hydralab, 2018], the problem is first decomposed into time averaged bore speed, time averaged flow depth and overtopping duration per wave. In this research, a deep learning algorithm is used to predict the overtopped volume. The advantage of this end-to-end approach is that a decomposition of the problem is not needed. The observations indicate that it is possible to predict the overtopping volume by using pixels as input for a CNN LSTM network. As the network understands what is happening on these series of frames, pixelwise sensitivity analysis can be applied to get an overview, which pixels are contributing the most to predicting the overtopping volume. Using this approach, we can address one of the problems mentioned in [Hydralab, 2018]: to what extent the bubbles or foam belongs to the boundary of a wave.

6.1 Future Work

As mentioned in the section Related Work, many studies are related to overtopping discharge rate in combination with neural networks with a multi-layer perceptron architecture. Interesting addition considers comparison of this research with the existing neural network mentioned in the EurOtop manual. In order to have a reliable comparison, the video must be captured in such a way that all the parameters used in the EurOtop manual are represented in the video.

The preprocessing was performed by using the top camera, whilst prediction was performed using the side camera. The side camera captures the height of the wave. However, we determined that the width also plays an important role in the prediction of the overtopping volume. In other words, the top camera can also contain useful information and contribute to the prediction of the overtopping volume. Future work can incorporate these elements.

In this research, a network is obtained which can predict the overtopping based on only pixels. Unfortunately, deep learning algorithms (e.g. CNN LSTM) are characterized by not being transparent. It is hard to derive which part of the wave is used to predict the overtopped volume. The pattern of the wave can be extracted by using pixel wise sensitivity analysis, for example Layer-wise Relevance Propagation or Deep Taylor Decomposition. This algorithm basically changes the value of each pixel one by one and measures the difference on the output which, in our case, is the overtopped volume. In other words, every neuron gets its share of relevance depending on activation and strength of connection. By doing this, a heatmap of the wave can be constructed. The heatmap contains an overview of the most relevant pixels which are used by the deep learning network to predict the overtopped volume. Using this approach, we can identify the pixels which are contributing the most to predicting the overtopping volume. These pixels can also be the boundary of the real and bubble part of the wave.

Furthermore, it takes more time for the water meter to capture the effect of weak waves. This problem can be simplified by binning the labels into several categories. This will change the complete research into a classification problem. The aim would be than the correct classification of the wave. We can, for example, range the waves from 1 (weakest) to 5 (strongest), and simplify the problem.

We can see many hyperparameters from the architecture of both models. This is the case for every machine learning system. As the performance of a system is dependent on these hyperparameters, it is important to choose them in such a way that we achieve the highest performance. However, the latter takes too much effort, time and money. In the recent years, studies were performed on this so called automated hyperparameter optimization (HPO). More about this can be read in [van Stein et al., 2018] and [Hutter et al., 2018]. The goal of HPO is to reduce the human effort and increase the performance of the machine learning algorithm. In this book, they show that these automated machine learning approaches already perform good and in some situations even better than human machine learning experts. In other words, this book can be used as a guide for the algorithms to determine how to learn. This approach can also be used to predict overtopping volumes.

References

[Bach et al., 2015] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):1–46.

[Bengio, 2012] Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. *CoRR*, abs/1206.5533.

[Bojarski et al., 2016] Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to end learning for self-driving cars. *CoRR*, abs/1604.07316.

[Boogaard, 2000] Boogaard, H.F.P. van den; Mynett, A. H. T. (2000). Resampling techniques for the assessment of uncertainties in parameters and predictions of calibrated models. *Hydroinformatics : 4th International Conference held from 23-27 July 2000 in Iowa, USA. Proceedings*.

[Donahue et al., 2015] Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

[Hutter et al., 2018] Hutter, F., Kotthoff, L., and Vanschoren, J., editors (2018). *Automatic Machine Learning: Methods, Systems, Challenges*. Springer. In press, available at http://automl.org/book.

[Hydralab, 2018] Hydralab (2018). Hydralab+ adaptation for climate change. protocols for representing variability and unsteadiness in flume facilities, deliverable 8.2, final version 3.3.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA. Curran Associates Inc.

[LeCun et al., 1990] LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In Touretzky, D. S., editor, *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan-Kaufmann.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Orr, G. B., and Müller, K.-R. (1998). Efficient backprop. pages 9–50.

[McLaughlin et al., 2016] McLaughlin, N., Martinez-del Rincon, J., and Miller, P. (2016). Recurrent convolutional network for video-based person re-identification. pages 1325–1334.

[Molchanov et al., 2016] Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., and Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network.

[Nair and Hinton, 2010] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. pages 807–814.

[Pomerleau, 1989] Pomerleau, D. A. (1989). Advances in neural information processing systems 1. chapter ALVINN: An Autonomous Land Vehicle in a Neural Network, pages 305–313. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Ruder, 2016] Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

[Sainath et al., 2015] Sainath, T., Vinyals, O., Senior, A., and Sak, H. (2015). Convolutional, long short-term memory, fully connected deep neural networks. pages 4580–4584.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

[van der Meer et al., 2018] van der Meer, J., Allsop, W., Bruce, T., De Rouck, J., Kortenhaus, A., Pullen, T., Schüttrumpf, H., Troch, P., and Zanuttigh, B. (2018). *EurOtop: Manual on wave overtopping of sea defences and related sturctures - An overtopping manual largely based on European research, but for worlwide application (2nd edition)*.

[van Gent et al., 2007] van Gent, M. R., van den Boogaard, H. F., Pozueta, B., and Medina, J. R. (2007). Neural network modelling of wave overtopping at coastal structures. *Coastal Engineering*, 54(8):586 – 593.

[van Stein et al., 2018] van Stein, B., Wang, H., and Bäck, T. (2018). Automatic configuration of deep neural networks with EGO. *CoRR*, abs/1810.05526.

[Zanuttigh et al., 2016a] Zanuttigh, B., Formentin, S., and van der Meer, J. (2016a). A neural network tool for predicting wave reflection, overtopping and transmission. *Coastal Engineering Journal*.

[Zanuttigh et al., 2017] Zanuttigh, B., Formentin, S., and van der Meer, J. (2017). Update of the eurotop neural network tool: Improved prediction of wave overtopping. *Coastal Engineering Proceedings*, 1(35):2.

[Zanuttigh et al., 2016b] Zanuttigh, B., Formentin, S. M., and van der Meer, J. W. (2016b). Prediction of extreme and tolerable wave overtopping discharges through an advanced neural network. *Ocean Engineering*, 127:7 – 22.

Appendices

Parameters used in CLASH coastal structure

#	Parameter	Unit	Type	Definition of the parameter
1	Name	[-]	general	
2	$H_{m,0deep}$	[m]	hydraulic	Off-shore significant wave height
3	$T_{p,deep}$	[s]	hydraulic	Off-shore peak wave period p
4	$T_{m,deep}$	[s]	hydraulic	Off-shore average wave period
5	$T_{m-1,deep}$	[s]	hydraulic	Off-shore spectral wave period
6	h_{deep}	[m]	structural	Off-shore water depth
7	m	[-]	structural	Foreshore slope
8	β	[°]	hydraulic	Wave obliquity
9	h	[m]	structural	Water depth at the structure toe
10	$H_{m,0,t}$	[m]	hydraulic	Significant wave height at the structure toe
11	$T_{p,t}$	[s]	hydraulic	Peak wave period at the structure toe
12	$T_{m,t}$	[s]	hydraulic	Average wave period at the structure toe
13	$T_{m-1,t}$	[s]	hydraulic	Spectral wave period at the structure toe
14	h_t	[m]	structural	Toe submergence
15	B_t	[m]	structural	Toe width
16	Type	[-]	structural	Type of structure and armour unit
17	$\cot\alpha_d$	[-]	structural	Cotangent of the angle that the structure part below/above the berm makes with a horizontal
18	$\cot\alpha_u$	[-]	structural	
19	$\cot\alpha_excl$	[-]	structural	Cotangent of the mean angle that the structure makes with a horizontal, excluding/including the berm, in the run
20	$\cot\alpha_incl$	[-]	structural	up/run-down zone
21	γ_f	[-]	structural	Roughness factor (average in the run-up/down area in the new DB)
22	B	[m]	structural	Berm width
23	h_b	[m]	structural	Berm submergence
24	$\tan\alpha_b$	[-]	structural	Berm slope
25	B_h	[m]	structural	Horizontal berm width
26	A_c	[m]	structural	Crest height with respect to swl
27	R_c	[m]	structural	Wall height with respect to swl
28	G_c	[m]	structural	Crest width
29	RF	[-]	general	Reliability Factor
30	CF	[-]	general	Complexity Factor
31	Pow	[-]	hydraulic	Overtopping probability
32	q	$[m^3/s/m]$	output	Wave overtopping discharge per unit width

Table 3: Parameters used in the CLASH project [Zanuttigh et al., 2016b].