Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

A River Flooding Detection System Based on Deep Learning and Computer Vision

Francisco E. Fernandes Jr. · Luis Gustavo Nonato · Jó Ueyama

Received: 19 January 2021 / Revised: 7 February 2022 / Accepted: 9 March 2022

Abstract Although floods cause millions of dollars in economic and social losses each year, many people living in developing countries, such as Brazil, do not have access to a flooding alert system because of its cost. To address this issue, we propose a cheap and robust River Flooding Detection System, which can be easily deployed in any river with a flat surface at its bedside. The novelty of our system is the use of raw images from off-the-shelf cameras with no preprocessing required. Hence, our methodology can be deployed using existing surveillance cameras in urban environments. The proposed system measures the river level by first performing a semantic segmentation of the river water blade using Deep Neural Networks (DNNs). Then, it uses Computer Vision (CV) to estimate the water level. If the water level is near or above a dangerous threshold, it sends alerts automatically without human intervention. Moreover, our system can successfully measure a river's water level with a Mean Absolute Error (MAE) of 3.32 cm, which is enough to detect when a river is about to overflow. The system is also reliable in measuring a river's water level from different camera viewpoints and lighting conditions. We show our approach's viability and evaluate our prototype's

Francisco E. Fernandes Jr.
Institute of Mathematical and Computer Sciences
University of São Paulo
São Carlos, SP, Brazil
E-mail: feferna@icmc.usp.br
ORCID: 0000-0003-2301-8820

Luis Gustavo Nonato
Institute of Mathematical and Computer Sciences
University of São Paulo
São Carlos, SP, Brazil
E-mail: gnonato@icmc.usp.br
ORCID: 0000-0002-8514-8033

Jó Ueyama
Institute of Mathematical and Computer Sciences
University of São Paulo
São Carlos, SP, Brazil
E-mail: joueyama@icmc.usp.br
ORCID: 0000-0002-5591-3750

performance and overhead by deploying it to monitor two urban rivers in the city of São Carlos, SP, Brazil.

Keywords River Flooding Detection · Semantic Segmentation · Deep Learning · Computer Vision

1 Declarations

1.1 Funding

The authors would like to thank the São Paulo Research Foundation (FAPESP), grant 2020/05426-0, for the financial support used to complete this work. The views expressed are those of the authors and do not reflect the official policy or position of the São Paulo Research Foundation.

1.2 Conflicts of interest

The authors declare that there is no conflict of interest.

1.3 Availability of data and material

Data will be made available upon request.

1.4 Code availability

The code will be made available at the following url: https://github.com/feferna/river-flooding-detection

1.5 Authors' contributions

Francisco E. Fernandes Jr. is responsible for the proposed method investigation and methodology, and experimental comparison and validation.

Luis Gustavo Nonato is responsible for structuring part of the experiments and comparisons, while reviewing the manuscript to ensure the work is reproducible.

Jo Ueyama is responsible for the overall idea conceptualization, and ensuring that the descriptions are accurate and agreed by all authors.

2 Introduction

From all the natural disasters that afflict populations worldwide, floods are the most common one [1]. Despite being so prevailing, it does not mean they are less dangerous than other types of natural disasters. For example, more than 74 million people worldwide have been somehow affected by some flooding event with significant economic loss, reaching 57 million dollars in 2016 alone [2]. Floods also

affect the populations' health by increasing the risk of water-borne and vector-borne diseases [3]. Besides, urban areas can experience the so-called river floodings when urban rivers cannot drain the waters from quick and excessive precipitations. Furthermore, river floodings are generally responsible for flash floodings in urban areas, which are among the most dangerous types of flooding events because of how fast they can happen and the complexity of forecasting them [4].

Even the most developed regions in a country are not immune to the risks posed by flooding. For instance, one of Brazil's wealthiest states, São Paulo (SP), is continually facing social and economic uncertainties caused by these natural events. Its capital, São Paulo City, has at its disposal the SAISP alert system, consisting of sensors and weather radars developed to forecast flooding disasters. Authorities and news organizations use this alert system to prepare the population and mitigate the damages caused by flooding events [5]. However, the city still loses around 42 million dollars every year due to flood damages [6], even with the SAISP system in place.

In contrast to the city of São Paulo, most cities located in São Paulo's countryside do not have an alert system available, despite the prevalence of flooding events and the associated social and economic losses caused by them. Specifically, São Carlos, SP, is a city prone to river flooding events during its rainy season. In one recent event, the city lost more than 8 million dollars due to property damages[1]. Hence, the present work focuses on monitoring São Carlos' rivers. Previous works have already developed and deployed a Wireless Sensor Network (WSN), called E-Noe, to monitor the city's rivers [7, 8, 9]. However, the E-Noe system's monitoring relies on pressure and flow sensors installed on the riverbed. These sensors require specialized labor for installation and maintenance and are also prone to problems due to oxidation. With this problem in mind, Ortigossa et al. [10] proposed using video cameras to detect the level of rivers through computer vision, creating a portable and cheaper version of the E-Noe system. Despite successfully detecting the river level, this method demands constant lighting conditions to work correctly and many recalibrations to detect the river level from different camera viewpoints.

Ortigossa et al. [10] show us that it is not trivial to extract usable knowledge from high-resolution raw images. One way to tackle this problem is to use Deep Learning models, such as Deep Neural Networks (DNNs). They are becoming increasingly popular in many fields due to their capabilities of learning from raw data without the need to model the problem at hand mathematically and have been applied in problems ranging from robotics applications [11, 12] to medical imaging diagnostics [13, 14, 15]. DNNs are composed of simple mathematical operations with adjustable parameters stacked in multiple layers. The error signal produced when comparing the ground truth result with the DNN predicted one is used to adjust their parameters. This learning procedure is called backpropagation and allows DNNs to be used as universal approximators if enough training data is available [16]. Furthermore, in recent years, DNN models perform consistently better than pure Computer Vision (CV) methods. For example, in the ImageNet Large Scale Visual Recognition Challenge, where a model needs to classify more

[1] https://g1.globo.com/sp/sao-carlos-regiao/noticia/2020/12/01/prejuizos-causados-pela-enchente-ultrapassam-r-43-milhoes-diz-defesa-civil-de-sao-carlos.ghtml

than 3 million images into 1,000 categories [17], DNN models are consistently better than CV methods since 2013 [18].

Therefore, methods based on Deep Learning or a mix of Deep Learning and CV methods are preferable nowadays and allows experts and researchers to develop robust and straightforward approaches to complex problems. With that in mind, we propose a generic River Flooding Detection System based on Deep Learning and CV. Moreover, the proposed system was tested and deployed in rivers located in São Carlos, SP, Brazil. The proposed system works as follows:

1. The cameras, installed in two nodes of the E-Noe system, capture real-time images from the city's rivers.
2. The images are transmitted to a cloud server on the internet through a 4G connection.
3. The cloud server processes the images to determine the level of the river.
4. Suppose the river level is near or above a dangerous preset threshold. In that case, a River Flooding Warning is sent to the authorities and population to reduce possible flooding damages. Otherwise, no alert is sent.

Our main contribution is an algorithm that automatically detects the water level in urban rivers using only raw and unprocessed images. Our work does not require the deployment of ultrasonic sensors or aerial/satellite imagery as the ones found in [30, 31, 32, 33, 34, 35, 36, 37, 38, 42, 43, 44]. Besides, our methodology uses the floodwall or a flat surface near the river bedside to perform the level measurement. Therefore, it does not require complex Computer Vision (CV) modeling such as image subtraction [20], Otsu thresholding [10], or the extraction of Regions of Interest (RoI) [19]. Moreover, our methodology is robust to changes in image conditions, such as different lighting conditions, weather conditions, or camera positioning. We first use a Deep Neural Network (DNN) model to perform Semantic Segmentation, which detects the river's water surface. Then, we use the segmented water surface to measure the river level using a flat surface as a reference. Since our algorithm relies only on images obtained from video cameras, it allows the construction of a cheap and robust system. Specifically, floodwaters cannot easily damage video cameras, which are also cheaper to deploy than aerial/satellite imaging devices. Furthermore, we can use our proposed methodology to monitor any river containing some flat surface.

The remainder of this work details the proposed River Flooding Detection System, and it is organized as follows. Section 3 presents related works to the present one. The details of the proposed alert system emphasizing the river level detection algorithm are found in Section 4. The experimental design, such as the algorithm parameters used in our tests, is presented in Section 5. In contrast, Section 6 presents and discusses the results we obtained. For last, Section 7 concludes the present work with suggestions for future work.

3 Related Works

The present section provides an overview of Deep Learning and Flood Alert Systems found in the scientific literature. We first describe works related to the general use of Deep Learning tools and models in solving challenging real-world problems. Second, we discuss works related to river monitoring and flood detection

mechanisms that do not use cameras and images. Third, Image-based systems are discussed. Finally, we conclude this section describing methodologies that rely on Deep Learning methods to perform flood detection, independent of the type of sensors used.

Although Deep Learning methods may not guarantee good generalization [22], they are among the most powerful ones we have available to solve challenging real-world problems. Some examples of tasks in which Deep Learning is successfully employed are image classification [23], object localization [24], intelligent game playing [25], machine translation [26], and many others. Likewise, Deep Learning methods are suitable for identifying natural formations from satellite images, such as identifying wetlands [27] and dead forest covers [28] and discovering new archeological sites [29]. Fog computing and edge devices are also taking advantage of Deep Learning methods. For example, smart cities are using them to detect crowd and traffic flows in smart cities [**? ? ?**]. These methods are also powerful enough to be used in disaster monitoring and prevention in general [30, 31, 32]. The variety of successful applications of Deep Learning methods proves that these methods are suitable for flood monitoring and detection.

Multiple types of sensors can be used to measure the level of a river. For example, ultrasonic and pressure sensors have been successfully employed to perform direct measurements of a river's level and flow to predict flooding events [33, 34, 35]. Some works gauge indirect quantities, such as humidity and carbon dioxide of the soil to anticipate whether a flash flooding event will happen [36, 37]. However, although non-imaging devices have advantages, such as simple algorithmic implementation, they require constant calibration to be deployed in distinct locations. They can also be easily damaged when they need to stay submerged.

To overcome some of the limitations discussed above, researchers use imaging devices to detect rivers' water blades. There are mainly two approaches to detect water levels using images: standard computer vision algorithms and machine learning algorithms. The works that use only computer vision first preprocess an image and then detect the river level. For example, Popescu et al. [38] used texture processing from images obtained with Unmanned Aerial Vehicles (UAVs) to detect flooding events in difficult to access areas. On the other hand, Kim et al. [20] used cameras installed at the river bedside and measured the river's level by computing the difference between a given image and a reference image, thus extracting the river water blade. Likewise, Ortigossa et al. [10] rely on the Otsu thresholding of the river image to extract lines of interest that are then used to compute the river level.

Regarding the use of Deep Learning models to process color images, there is the work developed by Pan et al. [19] in which they applied a Deep Convolutional Neural Networks (DCNN) to a Region of Interest (ROI) to classify if a given window inside the ROI corresponds to the water surface or a ruler. From the classifications, the algorithm computes the river's level in the image. Besides, Kang et al. [21] also used DCNNs to perform flood prediction, but, in this case, they make use of satellite images instead of river images taken from the ground. On the other hand, Lopez-Fuentes et al. [39] demonstrated that Deep Neural Networks (DNNs) could be used to perform semantic segmentation of water bodies with high accuracy. However, the work in [39] did not perform any flooding detection with the segmented images.

Table 1: Comparative of related works.

Author	Methodology	Investigated Problem	Sensor Data
Mahdianpari *et al.* [27]	Deep Convolutional Neural Networks	Mapping of wetlands	Satellite images
Sylvain *et al.* [28]	Deep Convolutional Neural Networks	Mapping of dead forests	Aerial images
Roccetti *et al.* [29]	Deep Convolutional Neural Networks	Discovery of new archaeological sites	Satellite images
Kamilaris and Prenafeta-Boldú [30]	Deep Convolutional Neural Networks	General disaster monitoring	Aerial images
Liu and Wu [31]	Deep Auto-Encoders	Geological disaster recognition	Satellite images
Liu and Wu [32]	Joint Auto-Encoders with Neural Networks and Support-Vector Machine	Geological disaster recognition	Satellite images
Wirawan *et al.* [33]	Wireless Sensor Network	Flooding detection	Ultrasonic sensor data
Noar *et al.* [34]	Wireless Sensor Network	Flood detection	Ultrasonic sensor data
Bączyk *et al.* [35]	Wireless Sensor Network	Flood detection	Pressure Transducers
Kafli and Isa [36]	Wireless Sensor Network and Internet-of-Things	Flood detection using water surface platforms	Ultrasonic, temperature, pH, carbon monoxide, GPS and water level sensors
Khan *et al.* [37]	Artificial Neural Networks	Flash flood detection	CO_2 and humidity sensors
Popescu *et al.* [38]	Texture processing using the sliding window method	Flood detection	Aerial images
Kim et al. [20]	Image processing through image subtraction	River water level monitoring	Camera images
Ortigossa et al. [10]	Image processing using Otsu thresholding	River water level monitoring	Camera images
Pan et al. [19]	Deep Convolutional Neural Networks using a Region of Interest	River water level monitoring	Camera images
Kang et al. [21]	Deep Convolutional Neural Networks	Flood detection	Satellite images
Lopez-Fuentes et al. [39]	Semantic Segmentation using Deep Neural Networks	Water body identification	Camera images
Krzhizhanovskaya et al. [41]	Committee of Artificial Neural Networks and dike simulations	Flood monitoring and alert	Pressure sensors
Guo et al. [42]	Deep Convolutional Neural Networks	Flood prediction through simulations	Multi-channel terrain images
Yang and Chang [43]	Recurrent Neural Networks	Flood prediction	Flood sensors
Al Quntus et al. [44]	Wireless Sensor Network and Support Vector Machine	Flood detection	Flood sensors
Our Work	**Semantic Segmentation using Deep Neural Networks and Computer Vision methods**	**River water level monitoring and flood detection and flood alert**	**Camera images**

Additionally, many works employ Deep Learning and Machine Learning tools to process and predict flooding events from different types of data [40, 41, 42]. For example, Yang and Chang [43] used IoT sensor data to train a Recurrent Neural Network to predict the river's water level three hours in the future. Al Quntus et al. [44] applied a Support Vector Machine (SVM) algorithm with a collection of quantities, such as wind speed, water level, humidity, and temperature, to make binary decisions of flooding or not flooding events. A more comprehensive survey on the use of Machine Learning, Computer Vision, and Wireless Sensor Network to detect and predict flooding events can be found in [45].

From the related works in the literature, it is easy to see that detecting floods is an active research field with many open questions. Many works make use of satellite or aerial images to monitor and predict flooding events [30, 31, 32, 38, 42]. The main drawback of this strategy is that they require expensive imaging technology, such as satellites and drones, which are not available in developing countries and poor cities. This type of image is also more suitable for use when dealing with large-scale flooding events, which can happen in a matter of days or weeks.

Simultaneously, some works rely heavily on sensor networks using pressure and ultrasonic sensors to detect a river's water level and predict when a flood will happen [33, 34, 35, 36, 37, 43, 44]. The use of these sensors has the advantage of

requiring a simple algorithmic implementation because they can direct output the current water level. The use of this type of data can be framed as a time-series problem, where the algorithm can predict the water level many hours in the future. However, this type of sensor can be easily damaged due to oxidation and silting. Moreover, once these sensors are damaged, they require specialized machinery to remove them from the river's bottom and install a new one, which can be both time-consuming and expensive.

Researchers are starting to use Deep Learning methods instead of pure Computer Vision ones, but with complex and hard-to-replicate methodologies. In that sense, the works of Kim et al. [20], Ortigossa et al. [10], Pan et al. [19], and Lopez-Fuentes et al. [39] are the ones most closely related to our proposed methodology. Kim et al. [20] use the Image Subtraction technique where the current river image is subtracted from a reference image, but it requires a complex calibration phase using a pattern installed in the field. Ortigossa et al. [10] employs Otsu Thresholding to extract lines from a given image, and the water level is computed by finding the water line in an image. Pan et al. [19] use Deep Convolutional Neural Networks (DCNN) to search a grid image extracted from Region of Interest (RoI). Pan et al. [19] used a sliding window in the RoI to classify if the texture comes from water or something else. Based on the number of water grids found by the DCNN, the algorithm can determine the water level. The problems with this method are that the dataset creation becomes non-trivial, and it also requires recalibrations when dealing with different angles and river images. Lopez-Fuentes et al. [39] is closely related to ours because they used DNN to perform semantic segmentation of water surfaces. They showed that this type of segmentation is possible, but no water level detection was investigated.

Hence, we propose a more straightforward methodology with competitive results. Table 1 presents a comparison of the works discussed in this section. The main contribution of our methodology is the detection of a river's water level from a single and raw image without preprocessing it. The river level is estimated by measuring the distance between a reference line and the segmented river water surface. Furthermore, we detect the river's water surface using a Deep Neural Network, which is robust to changes in viewpoints and lighting conditions, avoiding the need for constant recalibration and submersion of sensors.

4 Proposed Framework for a River Flooding Detection System

This section presents a detailed description of the proposed generic River Flooding Detection System. An overview of the system is presented first, followed by each of its core components.

4.1 Overview

The proposed system is comprised of four main steps, as illustrated in Figure 1. The system works by first collecting images in real-time of the rivers being monitored and then sending these images to a cloud server through a 4G internet connection. When a new image is received, the cloud server processes it by segmenting the water surface, detecting the river level, and, if necessary, it triggers a river flooding

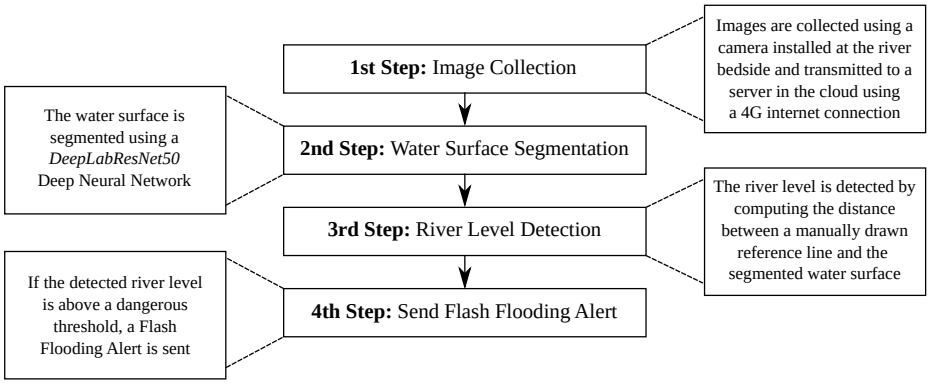


Fig. 1: Overview of the proposed River Flooding Detection System.

Fig. 2: Camera node used in our River Flooding Detection System.

alert to registered emails. The images are currently collected using 8 Mega Pixels (MP) cameras connected to Raspberry Pi mini-computers, as shown in Figure 2. The image processing steps are detailed in the following subsections.

4.2 Water Surface Segmentation

The water surface segmentation is the first processing step performed in the cloud server. The segmentation is performed with a Deep Neural Network (DNN). Specifically, a DeepLabv3 DNN model is used to perform the segmentation [46]. This chosen DNN architecture is illustrated in Figure 3. Compared to classical models, the main difference of the DeepLabv3 is the Atrous Spatial Pyramid Pooling (ASPP), which allows for better upsampling operations than traditional transposed convolution operations. This DNN model was chosen because it has one of the highest performances in semantic segmentation tasks [46]. In our tests, this model was able to segment the water surface of different rivers with various lighting conditions and camera positions. We trained the DeepLabv3 model to classify each pixel in a given image into two classes: water surface and background. Once

the water surface is segmented, the river level is detected using a simple computer vision algorithm, detailed in the following subsection.

4.3 Water Level Detection

A reference line perpendicular to the water surface must be manually drawn for a given camera viewpoint to detect the river water surface level. We draw this reference line from the highest point, called point $\mathbf{p}$, of the floodwall to the lowest point, called point $\mathbf{q}$, using an image from the river at its lowest water level, as shown in Figure 4a. This step is a calibration one. However, it avoids the need to find the parameters of each new camera installed in the system. Thus, with the proposed method, it is possible to use existing cameras installed throughout the city to monitor river floodings.

Once we define the reference line's location for a given camera viewpoint, the proposed method performs the following set of calculations to determine the river's water level:

1. Calculate the Euclidean distance, $d_r(\mathbf{p}, \mathbf{q})$, in pixels, between the two extreme points ($\mathbf{p} = [p_x, p_y]$ and $\mathbf{q} = [q_x, q_y]$) of the reference line.
2. Compute the slope of the reference line: $m_r = (q_y - p_y)/(q_x - q_y)$.
3. Use the slope, m_r, to find the point in the reference line that intersects the segmented water surface: $\mathbf{r} = [r_x, r_y]$.
4. Compute the Euclidean distance, $d_i(\mathbf{p}, \mathbf{r})$, in pixels, between the upmost point in the reference line and the point intersecting the segmented water surface.
5. The river's water level is computed from the following linear equation:

$$h = h_{max} - \left((h_{max} - h_{empty}) \cdot \frac{d_i(\mathbf{p}, \mathbf{r})}{d_r(\mathbf{p}, \mathbf{q})} \right), \tag{1}$$

where h is the river water level, h_{max} is the maximum height of the floodwall or flat surface used for reference, and h_{empty} is the height of the lowest water level from the reference image. Moreover, h, h_{max}, and h_{empty} are measured in meters, while d_r and d_i are measured in pixels. The quantities in Eq. 1 are illustrated in Figure 4b.

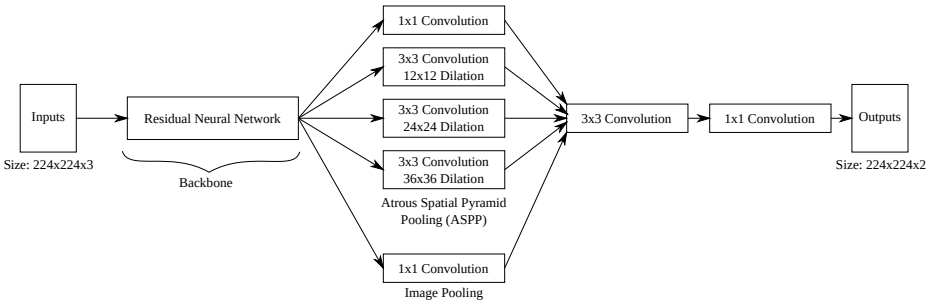


Fig. 3: Architecture of the DeepLabv3 Neural Network.

There is also an edge case when computing the water level: when the water surface, point **q**, is above the floodwall, point **p**. In this case, we consider that the river is already overflown, and we do not compute the water level anymore. Furthermore, when the river is overflown, a River Flooding Alert is sent independently of the actual water level.

4.4 River Flooding Alert

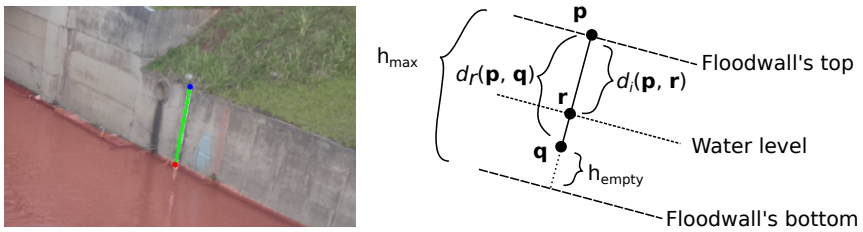
Once the river level is known, the system can decide to send a River Flooding Alert. There are currently two ways an alert can be sent: directly from the project's website and by email. The alert is sent once either monitored river has reached a dangerous preset level. Once the alert is sent, the population has between five to ten minutes to take shelter. This alert aims to reduce the probability of economic and social loss in the area near the river. The proposed system can also send an email alert to the city's civil defense authorities, allowing them to assist the population close to the affected area and block drivers' access to flooded roads. With the alert, the latest available image is also sent, allowing the authorities to understand the river's surroundings better.

5 Experimental Design

This section presents the parameters and the experimental setting used to validate the proposed algorithm.

5.1 Deep Neural Network Fine-Tuning Parameters

We chose to test Fully Convolutional Networks (FCN) and DeepLabv3 architectures. Both architectures have backbones composed of Residual Neural Networks (ResNets) with 50 layers (ResNet50) and 101 layers (ResNet101). These models were trained using Transfer Learning, where a model pre-trained in a different dataset is fine-tuned to the new task [47]. In our case, the chosen models were pre-trained in the Microsoft COCO dataset [48], and we fine-tuned them to segment the water surface of river images. The FCN and DeepLabv3 models were then fine-tuned for a total of 50 epochs with this dataset. Furthermore, we fine-tuned the



(a) Manually drawn reference line. (b) How the water level is measured.

Fig. 4: Water level measurement.

Table 2: Parameters used to fine-tune the DNNs for water surface segmentation.

Parameter	Value
DNN Models	FCN ResNet50 FCN ResNet101 DeepLabv3 ResNet50 DeepLabv3 ResNet101
Pretained Dataset	Microsoft COCO
Optimizer	Stochastic Gradient Descent
Loss Function	Cross-entropy Loss
Number of Epochs	50
Weight Decay	5×10^{-4}
Momentum	0.9
Learning rate	1×10^{-5}
Learning rate decay factor	0.5
Learning rate decay schedule	7 epochs
Batch Size	8

chosen DeepLabv3 DNN models using the Stochastic Gradient Descent optimizer with a Batch Size of eight images, an initial Learning Rate equal to 1×10^{-5}, Momentum equal to 0.9, and Weight Decay equal to 5×10^{-4}. The Learning Rate also decays by a factor of 0.5 every seven epochs during training. Table 2 summarizes the parameters used in the fine-tuning process.

5.2 Monitored Rivers and Training Dataset

Our proposed alert system currently monitors two rivers, called rivers A and B, in São Carlos, SP, Brazil. Figure 6a shows the physical locations of the cameras installed in the city, while Figures 6b and 6c show sample images from rivers A and B, respectively. We chose these two points because they are located in an important commercial district and are also notoriously known for being susceptible to flash floodings during the city's rainy season. Thus, being able to detect flooding events can reduce the economic loss caused by them.

We have 167,897 raw and unedited images from these two rivers spanning a total of years of river monitoring from 2016 to 2020. We built the dataset used to train the DeepLabv3 models from these images, where we manually annotated the water surface of 968 images used for training and 30 images used for testing. Some examples of training images are shown in Figure 5. The annotated images were chosen randomly, but we tried to keep the dataset balanced with images of empty and flooded rivers. Furthermore, the DeepLabv3 models only accept input images with a resolution of 224 by 224 pixels. Thus, we downsized and center-cropped our images from 1280 by 720 pixels to match this resolution.

5.3 Implementation Details and Algorithm Parameters

We trained the DeepLabv3 models used in this work in a machine equipped with an Nvidia RTX 3070 GPU with 8GB of video memory, an AMD Ryzen 7 3700x processor, and 32 GB of Random Access Memory (RAM). On the software side, we used Python 3.8 with the PyTorch 1.4.0 Deep Learning library and OpenCV

Fig. 5: Examples of manually segmented water surfaces used to train the DeepLabv3 DNN.

4 to run, train and deploy our algorithm. Once trained, the DNN model and river level detecting algorithms are deployed in a server running on an Nvidia GTX 1650 GPU with 4 GB of video memory, an Intel Core i7-7700 processor, and 16 GB of RAM. Furthermore, all software was developed and deployed using the Ubuntu 20.04 Linux Operating System. For anyone who desires to reproduce or improve our algorithm, the source code is available on GitHub[2].

Due to processing power and internet bandwidth constraint, the proposed alert system receives images from our sensor nodes every five minutes. Moreover, for river A, the quantities represented in Eq. 1 are $h_{max} = 2.7$ meters, $h_{empty} = 0.45$ meters, while for river B h_{max} is equal to 2.7 meters and $h_{empty} = 0.2$ meters. We manually measured these quantities for each river. For last, an alert will be sent if the computed river A's water level reaches 2.40 meters, which is close to the height of the floodwall, or if river B's level reaches 1.50 meters.

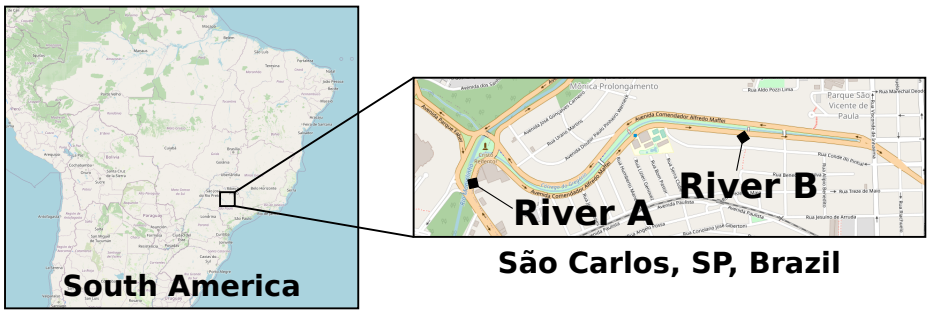
6 Experimental Results

This section details the results obtained with the proposed river flooding detection system. First, we present the overall results we obtained with the proposed system, and then we discuss these results in detail.

6.1 Overall Results

The resulting training curves of the FCN and DeepLabv3 DNN models can be visualized in Figure 7, where the Cross-Entropy Loss and the Intersection over the Union (IoU) of the river's water surface, two of the most used metrics in semantic

[2] https://github.com/feferna/river-flooding-detection



(a) Physical location of the rivers being monitored.

(b) Sample image of river A.

(c) Sample image of river B.

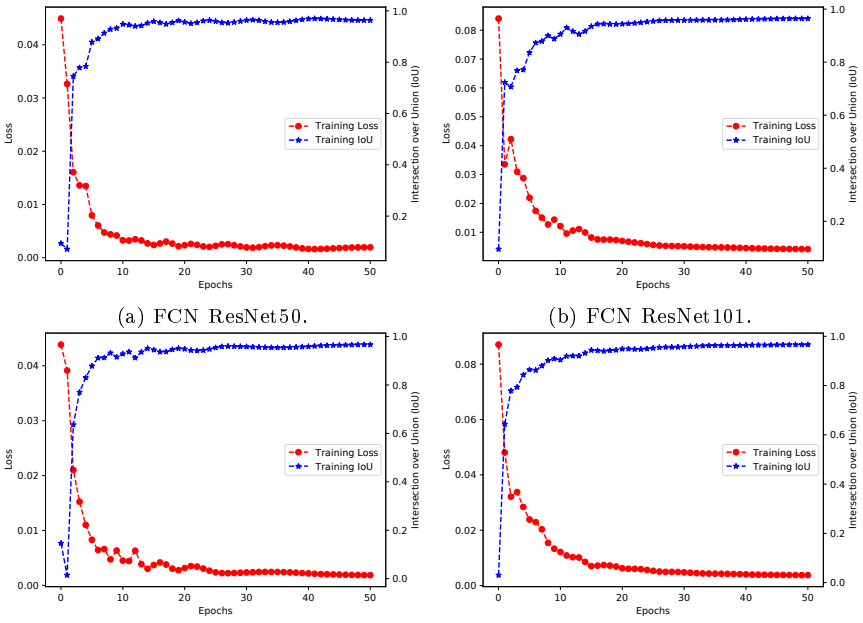
Fig. 6: Location and sample images of the rivers being monitored. The water flows from river B to river A.

Table 3: Fine-tuning results of the chosen DNN models on our custom water segmentation dataset.

Model	Training Loss	Training IoU	Test Loss	Test IoU
FCN ResNet50	0.0019	0.9634	0.0043	0.9372
FCN ResNet101	0.0042	0.9641	0.0078	0.9402
DeepLabv3 ResNet50	0.0019	0.9666	0.0037	0.9434
DeepLabv3 ResNet101	0.0038	0.9661	0.0072	0.9422

segmentation, are depicted. Moreover, at the end of the training session, the models were tested with 30 images, not from the training set, to verify their generalization capabilities. Table 3 also presents the training results of the four DNN models we tested. The test loss and test IoU for the FCN ResNet50 model are equal to 0.0043 and 0.9372, respectively, while for the FCN ResNet101, they are equal to 0.0078 and 0.9402, respectively. The loss and the IoU for the DeepLabv3 ResNet50 model's test set were equal to 0.0037 and 0.9434, respectively. The DeepLabv3 ResNet101 model obtained a test loss equal to 0.0072 and a test IoU equal to 0.9422. These results show that all four DNN models can learn how to segment the water surface in a river's image with very high accuracy. Furthermore, the training curves in Figure 7 and the loss and IoU in the test set show that the models did not overfit the training set. The test IoU is very close to the training IoU showing no signs of loss of generalization.

To verify the accuracy of our proposed water level algorithm, we used a set of eight images taken from a single river but with different angles, as exemplified in Figure 8. These images were also used by Ortigossa et al. [10] to verify the accuracy of their algorithm. Hence, these eight images can help us compare our



(a) FCN ResNet50.

(b) FCN ResNet101.

(c) DeepLabv3 ResNet50.

(d) DeepLabv3 ResNet101.

Fig. 7: Fine-tuning curves of the chosen DNN models.

results with their results, shown in Table 4. The water level in these eight images was manually measured to be equal to 51.22 cm. The Mean Absolute Error (MAE) from the Ortigossa et al. [10] method was equal to 0.46 cm, while the MAE using our method was equal to 3.35 cm. Although our method has a worse MAE than the one in [10], it is still satisfactory because our main objective is to detect when a given river level has reached a dangerous preset threshold. Thus, we consider an absolute error of 10 cm enough to detect when the rivers being monitored in this study are about to overflow. Furthermore, our proposed method is robust to changes in illumination and camera viewpoints, which is not the case of the method in [10].

We also tested our proposed system with images of a flooding event in São Carlos, SP, caused by heavy rains. Figure 9 shows the evolution of this flooding event in roughly 20 minutes, and each image was taken with five minutes of difference. River A starts with 1.17 meters, Figure 9a, and the water level rises 2.40 meters, Figure 9c, where the system sends a Flood Alert. The water level continues to rise until the river has overflown, Figure 9d. Additionally, Figure 9d shows that our algorithm can detect if the river level is above a manually chosen threshold by comparing if the chosen reference point is below the segmented water surface border. The images of river B in Figure 10 depict a similar history, but river B did not overflow and kept its water level constant. However, as shown in Figure 6a, the water flows from river B to river A. Hence, even though the water level in river B is below the floodwall, it contributes to the water level in river A. Because of this, the water level threshold in river B needs to be lower than in river A. Additionally, we compared the water level computed by our algorithm and shown

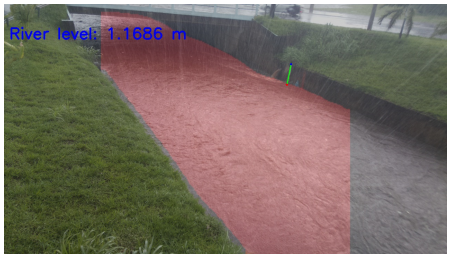
Table 4: Water level results in a set of images taken from a single river with different camera angles.

Image	Ground Truth (y)	Ortigossa et al. [10] ($\tilde{y}$)	Abs. Error ($\lvert y - \tilde{y} \rvert$)	**Ours ($\hat{y}$)**	**Abs. Error ($\lvert y - \hat{y} \rvert$)**
1	51.22 cm	50.24 cm	0.98 cm	**49.72 cm**	**0.46 cm**
2		50.74 cm	0.48 cm	**46.85 cm**	**4.37 cm**
3		50.71 cm	0.51 cm	**52.16 cm**	**0.94 cm**
4		50.34 cm	0.88 cm	**44.98 cm**	**6.24 cm**
5		51.41 cm	0.19 cm	**44.17 cm**	**7.05 cm**
6		51.55 cm	0.33 cm	**47.94 cm**	**3.28 cm**
7		50.93 cm	0.29 cm	**52.22 cm**	**1.00 cm**
8		51.21 cm	0.01 cm	**47.80 cm**	**3.42 cm**
Mean		50.89 cm	0.46 cm	**48.16 cm**	**3.35 cm**

in Figures 9 and 10 to a ruler installed at the river's bedside, shown in Figure 11, because we do not have any pressure or ultrasonic sensors installed.

We have also tested the segmented water surface's quality using images with different lighting conditions. We select four images from different days and times, but with the river in a low water level, as shown in Figure 12. We can see that the DeepLabv3 DNN model can very well segment the water surface in each image. These results show that the proposed water segmentation approach is suitable for dealing with lighting variation, avoiding the need to deal with such variations algorithmically.

Fig. 8: Water segmentation and water level detection results using images of a single river from different viewpoints.



(a) Beginning of heavy rain. River level is computed as 1.17 meters.

(b) Water level starts to rise. River level is computed as 1.82 meters.

(c) The water level reaches the highest point of the floodwall. River level is computed as 2.40 meters.

(d) The heavy rain overflows the river. Our system detects that the river has overflown. The system does not compute the water level once it is above the floodwall.

Fig. 9: River A's water level results during a flooding event.

(a)

(b)

(c)

Fig. 10: River B's water level results during a flooding event. In all images, river B has reached 1.50 meters, which is considered dangerous for this river.

6.2 Results Discussion

The results presented here show that the proposed generic River Flooding Detection System can satisfactorily segment the water surface and detect the river level using only raw images, allowing for portable and cheap sensors for detect-

Fig. 11: Ruler installed at river A's bedside. We use this ruler to verify how well our algorithm performs.

Fig. 12: Water level detection results under different lighting conditions. Our proposed algorithm is robust to changes in lighting conditions.

ing flooding situations. It is also important to note that we are not interested in have millimetric precision with our proposed river water detection algorithm. Our main goal is to detect when an urban river is close to overflowing or has already overflown due to a heavy rain. Furthermore, we intended to create a simple algorithm that could be easily deployed with different rivers. Thus, although our proposed method obtained a higher average MAE of 3.35 cm than the 0.46 cm from [10], it could still detect when the rivers' water level reached the dangerous preset thresholds.

The results also show that the proposed system is robust to changes in camera viewpoints and illumination, making it easier to deploy in many urban rivers. Moreover, the system is already available for use by the population[3], and it is capable of sending flooding alerts through the project's website and registered emails. The system sends a flooding alert once either one of the monitored rivers reaches a predetermined threshold deemed dangerous, of 2.40 meters for river A and 1.50 meters for river B, as illustrated in Figure 13. The city's authorities, business owners, and people living closer to these rivers can use the alerts to take shelter and move expensive items out of the water's way, reducing the economic and social losses caused by these disasters[4]. Although the proposed alert system currently monitors only two rivers, it can be easily extended to work with many more rivers.

However, there are some limitations to the proposed River Flooding Detection System as follows:

1. The system needs a flat surface orthogonal to the water blade in which a reference line can be manually drawn. For different camera viewpoints, this reference line needs to be redrawn.
2. The system does not account for slight variations in the camera viewpoint caused by the wind and temperature dilation. For example, a strong wind can increase the error between the computed water level and the real one.
3. Currently, the proposed system can only perform water surface segmentation during the daytime due to the lack of a night vision camera.

[3] The proposed flooding detection system can be accessed through the URL: http://agora.icmc.usp.br:5001

[4] https://g1.globo.com/sp/sao-carlos-regiao/noticia/2020/12/17/inteligencia-artificial-que-monitora-e-alerta-sobre-enchentes-e-criada-pela-usp-de-sao-carlos.ghtml

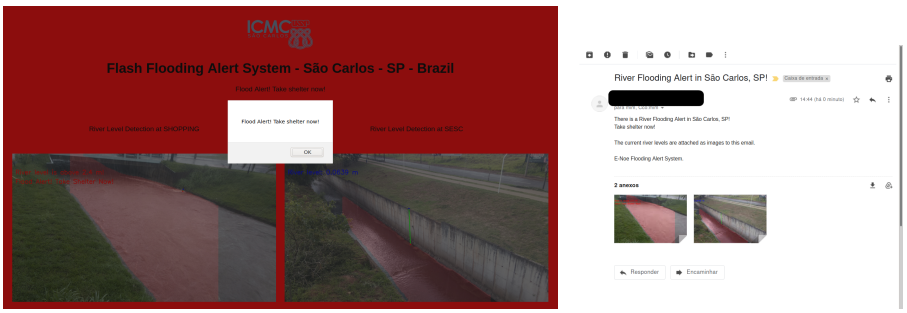


Fig. 13: River flooding detection system. On the left, alert sent through the project's website. On the right, alert sent through email.

Nevertheless, due to the current work scope, we believe these limitations are more suitable to be addressed in future works.

7 Conclusion and Future Work

This work presents the development of a generic River Flooding Detection System. Moreover, the proposed detection system can be used with different rivers, viewpoints, and lighting conditions thanks to the semantic segmentation performed by Deep Neural Networks (DNNs). Once the segmentation is performed, the river level can be easily determined by computing the distance from a manually drawn reference line to the water surface's border.

The proposed detection system was deployed in two rivers located in São Carlos, SP, Brazil, where alerts are sent once either one of the monitored rivers' water level reaches a preset threshold. The population can receive alerts by visiting the project's website or by registering their email. The city's authorities can also use the alerts to reduce property and human losses in the affected area.

In this document, we presented only the beginning of our plans to develop a robust river flooding detection system using only images from video cameras. There are still several improvements that we will be addressing in future work:

1. We plan to develop a smartphone app to allow for the quickest delivery of flood alerts.
2. We plan the development of portable hardware to enclosure the whole sensor for easy installation.
3. We will deploy a sensor network in multiple rivers throughout the city to detect flood events as early as possible.
4. We intend to develop a machine learning algorithm capable of automatically finding the reference lines in rivers images using visual clues, such as bar codes installed at the river bedside.

For last, we will use the future sensor network nodes' own processing power to perform the river flooding detection algorithm, allowing for a fault-tolerant system.

8 Acknowledgments

The authors would like to thank the São Paulo Research Foundation (FAPESP), grant 2020/05426-0, for the financial support used to complete this work. The views expressed are those of the authors and do not reflect the official policy or position of the São Paulo Research Foundation.

References

1. Du W, FitzGerald GJ, Clark M, Hou XY (2010) Health Impacts of Floods. Prehosp Disaster med 25(3):265–272, DOI 10.1017/S1049023X00008141, URL https://www.cambridge.org/core/product/identifier/S1049023X00008141/type/journal_article

2. Paterson DL, Wright H, Harris PNA (2018) Health Risks of Flood Disasters. Clinical Infectious Diseases 67(9):1450–1454, DOI 10.1093/cid/ciy227, URL https://academic.oup.com/cid/article/67/9/1450/4945455
3. Alderman K, Turner LR, Tong S (2012) Floods and human health: A systematic review. Environment International 47:37–47, DOI 10.1016/j.envint.2012.06.003, URL https://linkinghub.elsevier.com/retrieve/pii/S0160412012001237
4. Hapuarachchi HAP, Wang QJ, Pagano TC (2011) A review of advances in flash flood forecasting. Hydrol Process 25(18):2771–2784, DOI 10.1002/hyp.8040, URL http://doi.wiley.com/10.1002/hyp.8040
5. Barros MTL, Conde F (2017) Urban Flood Warning System Social Benefits. In: World Environmental and Water Resources Congress 2017, American Society of Civil Engineers, Sacramento, California, pp 642–653, DOI 10.1061/9780784480601.054, URL http://ascelibrary.org/doi/10.1061/9780784480601.054
6. Haddad EA, Teixeira E (2015) Economic impacts of natural disasters in megacities: The case of floods in São Paulo, Brazil. Habitat International 45:106–113, DOI 10.1016/j.habitatint.2014.06.023, URL https://linkinghub.elsevier.com/retrieve/pii/S019739751400099X
7. Hughes D, Ueyama J, Mendiondo E, Matthys N, Horré W, Michiels S, Huygens C, Joosen W, Man KL, Guan SU (2011) A middleware platform to support river monitoring using wireless sensor networks. J Braz Comput Soc 17(2):85–102, DOI 10.1007/s13173-011-0029-3, URL https://journal-bcs.springeropen.com/articles/10.1007/s13173-011-0029-3
8. Furquim G, Pessin G, Gomes PH, Mendiondo EM, Ueyama J (2015) A Distributed Approach to Flood Prediction Using a WSN and ML: A Comparative Study of ML Techniques in a WSN Deployed in Brazil. In: Jackowski K, Burduk R, Walkowiak K, Wozniak M, Yin H (eds) Intelligent Data Engineering and Automated Learning – IDEAL 2015, vol 9375, Springer International Publishing, Cham, pp 485–492, DOI 10.1007/978-3-319-24834-9_56, URL http://link.springer.com/10.1007/978-3-319-24834-9_56, series Title: Lecture Notes in Computer Science
9. Furquim G, Filho G, Jalali R, Pessin G, Pazzi R, Ueyama J (2018) How to Improve Fault Tolerance in Disaster Predictions: A Case Study about Flash Floods Using IoT, ML and Real Data. Sensors 18(3):907, DOI 10.3390/s18030907, URL http://www.mdpi.com/1424-8220/18/3/907
10. Ortigossa ES, Dias F, Ueyama J, Nonato LG (2015) Using digital image processing to estimate the depth of urban streams. In: Workshop of undergraduate works in conjunction with conference on graphics, patterns and images (SIBGRAPI), Salvador, BA, Brazil
11. Polydoros AS, Nalpantidis L, Kruger V (2015) Real-time deep learning of robotic manipulator inverse dynamics. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Hamburg, Germany, pp 3442–3448, DOI 10.1109/IROS.2015.7353857, URL http://ieeexplore.ieee.org/document/7353857/
12. Levine S, Pastor P, Krizhevsky A, Ibarz J, Quillen D (2018) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research 37(4-5):421–436, DOI 10.1177/0278364917710318, URL http://journals.sagepub.com/doi/

10.1177/0278364917710318
13. Ronneberger O, Fischer P, Brox T (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab N, Hornegger J, Wells WM, Frangi AF (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, vol 9351, Springer International Publishing, Cham, pp 234–241, DOI 10.1007/978-3-319-24574-4_28, URL http://link.springer.com/10.1007/978-3-319-24574-4_28, series Title: Lecture Notes in Computer Science
14. Fernandes FE, Yen GG (2020) Automatic Searching and Pruning of Deep Neural Networks for Medical Imaging Diagnostic. IEEE Trans Neural Netw Learning Syst pp 1–11, DOI 10.1109/TNNLS.2020.3027308, URL https://ieeexplore.ieee.org/document/9222548/
15. Fernandes FE, Yen GG (2021) Pruning of generative adversarial neural networks for medical imaging diagnostics with evolution strategy. Information Sciences 558:91–102, DOI 10.1016/j.ins.2020.12.086, URL https://linkinghub.elsevier.com/retrieve/pii/S0020025521000189
16. Hagan MT, Demuth HB, Beale MH, De Jesus O (2014) Neural network design, 2nd edn. Amazon Fulfillment Poland Sp. z o.o, Wrocław
17. Deng J, Dong W, Socher R, Li LJ, Kai Li, Li Fei-Fei (2009) ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, Miami, FL, pp 248–255, DOI 10.1109/CVPR.2009.5206848, URL http://ieeexplore.ieee.org/document/5206848/
18. (2021) ImageNet Benchmark (Image Classification). URL https://paperswithcode.com/sota/image-classification-on-imagenet
19. Pan J, Yin Y, Xiong J, Luo W, Gui G, Sari H (2018) Deep Learning-Based Unmanned Surveillance Systems for Observing Water Levels. IEEE Access 6:73561–73571, DOI 10.1109/ACCESS.2018.2883702, URL https://ieeexplore.ieee.org/document/8550626/
20. Kim K, Lee NK, Han Y, Hahn H (2007) Remote Detection and Monitoring of a Water Level Using Narrow Band Channel. In: Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, ISPRA'07, pp 25–30, event-place: Corfu Island, Greece
21. Kang W, Xiang Y, Wang F, Wan L, You H (2018) Flood Detection in Gaofen-3 SAR Images via Fully Convolutional Networks. Sensors 18(9):2915, DOI 10.3390/s18092915, URL http://www.mdpi.com/1424-8220/18/9/2915
22. Geirhos R, Temme CRM, Rauber J, Schütt HH, Bethge M, Wichmann FA (2018) Generalisation in humans and deep neural networks. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R (eds) Advances in Neural Information Processing Systems, Curran Associates, Inc., vol 31, URL https://proceedings.neurips.cc/paper/2018/file/0937fb5864ed06ffb59ae5f9b5ed67a9-Paper.pdf
23. Simonyan K, Zisserman A (2014) Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:14091556 URL http://arxiv.org/abs/1409.1556, arXiv: 1409.1556
24. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You Only Look Once: Unified, Real-Time Object Detection. In: 2016 IEEE Conference on Computer

Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, pp 779–788, DOI 10.1109/CVPR.2016.91, URL http://ieeexplore.ieee.org/document/7780460/
25. Barratt J, Pan C (2019) Playing Go without Game Tree Search Using Convolutional Neural Networks. arXiv:190704658 [cs] URL http://arxiv.org/abs/1907.04658, arXiv: 1907.04658
26. Singh SP, Kumar A, Darbari H, Singh L, Rastogi A, Jain S (2017) Machine translation using deep learning: An overview. In: 2017 International Conference on Computer, Communications and Electronics (Comptelix), IEEE, Jaipur, India, pp 162–167, DOI 10.1109/COMPTELIX.2017.8003957, URL http://ieeexplore.ieee.org/document/8003957/
27. Mahdianpari M, Salehi B, Rezaee M, Mohammadimanesh F, Zhang Y (2018) Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery. Remote Sensing 10(7):1119, DOI 10.3390/rs10071119, URL http://www.mdpi.com/2072-4292/10/7/1119
28. Sylvain JD, Drolet G, Brown N (2019) Mapping dead forest cover using a deep convolutional neural network and digital aerial photography. ISPRS Journal of Photogrammetry and Remote Sensing 156:14–26, DOI 10.1016/j.isprsjprs.2019.07.010, URL https://linkinghub.elsevier.com/retrieve/pii/S0924271619301777
29. Roccetti M, Casini L, Delnevo G, Orrù V, Marchetti N (2020) Potential and Limitations of Designing a Deep Learning Model for Discovering New Archaeological Sites: A Case with the Mesopotamian Floodplain. In: Proceedings of the 6th EAI International Conference on Smart Objects and Technologies for Social Good, ACM, Antwerp Belgium, pp 216–221, DOI 10.1145/3411170.3411254, URL https://dl.acm.org/doi/10.1145/3411170.3411254
30. Kamilaris A, Prenafeta-Boldú FX (2018) Disaster Monitoring using Unmanned Aerial Vehicles and Deep Learning. arXiv:180711805 [cs, stat] URL http://arxiv.org/abs/1807.11805, arXiv: 1807.11805
31. Liu Y, Wu L (2016) Geological Disaster Recognition on Optical Remote Sensing Images Using Deep Learning. Procedia Computer Science 91:566–575, DOI 10.1016/j.procs.2016.07.144, URL https://linkinghub.elsevier.com/retrieve/pii/S1877050916313370
32. Liu Y, Wu L (2018) High Performance Geological Disaster Recognition using Deep Learning. Procedia Computer Science 139:529–536, DOI 10.1016/j.procs.2018.10.237, URL https://linkinghub.elsevier.com/retrieve/pii/S1877050918319069
33. Wirawan W, Rachman S, Pratomo I, Mita N (2008) Design of low cost wireless sensor networks-based environmental monitoring system for developing country. In: 2008 14th Asia-Pacific Conference on Communications, IEEE, Tokyo, Japan, pp 1–5
34. Noar NAZM, Kamal MM (2017) The development of smart flood monitoring system using ultrasonic sensor with blynk applications. In: 2017 IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA), IEEE, Putrajaya, pp 1–6, DOI 10.1109/ICSIMA.2017.8312009, URL http://ieeexplore.ieee.org/document/8312009/
35. Bączyk A, Piwiński J, Kłoda R, Grygoruk M (2017) Survey on River Water Level Measuring Technologies: Case Study for Flood Management Purposes of the C2-SENSE Project. In: Szewczyk R, Kaliczyńska M (eds) Recent Ad-

vances in Systems, Control and Information Technology, vol 543, Springer International Publishing, Cham, pp 610–623, DOI 10.1007/978-3-319-48923-0_65, URL http://link.springer.com/10.1007/978-3-319-48923-0_65, series Title: Advances in Intelligent Systems and Computing

36. Kafli N, Isa K (2017) Internet of Things (IoT) for measuring and monitoring sensors data of water surface platform. In: 2017 IEEE 7th International Conference on Underwater System Technology: Theory and Applications (USYS), IEEE, Kuala Lumpur, pp 1–6, DOI 10.1109/USYS.2017.8309441, URL http://ieeexplore.ieee.org/document/8309441/
37. Khan TA, Alam M, Kadir K, Shahid Z, Mazliham M (2018) A Novel Approach for the Investigation of Flash Floods using Soil Flux and CO $_2$: An Implementation of MLP with Less False Alarm Rate. In: 2018 2nd International Conference on Smart Sensors and Application (ICSSA), IEEE, Kuching, pp 130–134, DOI 10.1109/ICSSA.2018.8535606, URL https://ieeexplore.ieee.org/document/8535606/
38. Popescu D, Ichim L, Caramihale T (2015) Flood areas detection based on UAV surveillance system. In: 2015 19th International Conference on System Theory, Control and Computing (ICSTCC), IEEE, Cheile Gradistei, Romania, pp 753–758, DOI 10.1109/ICSTCC.2015.7321384, URL http://ieeexplore.ieee.org/document/7321384/
39. Lopez-Fuentes L, Rossi C, Skinnemoen H (2017) River segmentation for flood monitoring. In: 2017 IEEE International Conference on Big Data (Big Data), IEEE, Boston, MA, pp 3746–3749, DOI 10.1109/BigData.2017.8258373, URL http://ieeexplore.ieee.org/document/8258373/
40. Subeesh A, Kumar P, Chauhan N (2019) Flood Early Detection System Using Internet of Things and Artificial Neural Networks. In: Bhattacharyya S, Hassanien AE, Gupta D, Khanna A, Pan I (eds) International Conference on Innovative Computing and Communications, vol 55, Springer Singapore, Singapore, pp 297–305, DOI 10.1007/978-981-13-2324-9_30, URL http://link.springer.com/10.1007/978-981-13-2324-9_30, series Title: Lecture Notes in Networks and Systems
41. Krzhizhanovskaya V, Shirshov G, Melnikova N, Belleman R, Rusadi F, Broekhuijsen B, Gouldby B, Lhomme J, Balis B, Bubak M, Pyayt A, Mokhov I, Ozhigin A, Lang B, Meijer R (2011) Flood early warning system: design, implementation and computational modules. Procedia Computer Science 4:106–115, DOI 10.1016/j.procs.2011.04.012, URL https://linkinghub.elsevier.com/retrieve/pii/S1877050911000706
42. Guo Z, Leitao JP, Simoes NE, Moosavi V (2020) Data-driven Flood Emulation: Speeding up Urban Flood Predictions by Deep Convolutional Neural Networks. arXiv:200408340 [cs] URL http://arxiv.org/abs/2004.08340, arXiv: 2004.08340
43. Yang SN, Chang LC (2020) Regional Inundation Forecasting Using Machine Learning Techniques with the Internet of Things. Water 12(6):1578, DOI 10.3390/w12061578, URL https://www.mdpi.com/2073-4441/12/6/1578
44. Al Qundus J, Dabbour K, Gupta S, Meissonier R, Paschke A (2020) Wireless sensor network for AI-based flood disaster detection. Ann Oper Res DOI 10.1007/s10479-020-03754-x, URL http://link.springer.com/10.1007/s10479-020-03754-x

45. Arshad B, Ogie R, Barthelemy J, Pradhan B, Verstaevel N, Perez P (2019) Computer Vision and IoT-Based Sensors in Flood Monitoring and Mapping: A Systematic Review. Sensors 19(22):5012, DOI 10.3390/s19225012, URL https://www.mdpi.com/1424-8220/19/22/5012
46. Chen LC, Papandreou G, Schroff F, Adam H (2017) Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv:170605587 [cs] URL http://arxiv.org/abs/1706.05587, arXiv: 1706.05587
47. Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? In: Ghahramani Z, Welling M, Cortes C, Lawrence N, Weinberger KQ (eds) Advances in Neural Information Processing Systems, Curran Associates, Inc., vol 27, pp 3320–3328, URL https://proceedings.neurips.cc/paper/2014/file/375c71349b295fbe2dcdca9206f20a06-Paper.pdf
48. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft COCO: Common Objects in Context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T (eds) Computer Vision – ECCV 2014, vol 8693, Springer International Publishing, Cham, pp 740–755, DOI 10.1007/978-3-319-10602-1_48, URL http://link.springer.com/10.1007/978-3-319-10602-1_48, series Title: Lecture Notes in Computer Science