Deep Learning and Object Detection for Water Level Measurement using Patterned Visual Markers

G. M. Domingues Filho ⬤, C. M. Ranieri ⬤, S. N. Matos ⬤, R. I. Meneguette ⬤, and J. Ueyama ⬤

Abstract—Flooding is one of the most impactful natural disasters, causing significant losses and prompting extensive research into monitoring water levels in urban streams. Current technologies rely on pressure and ultrasonic sensors, which, while accurate, can be susceptible to damage from floods and are often costly. As an alternative, ground camera approaches offer a low-cost solution; however, most of these methods use raw images from the water stream and are sensitive to environmental factors. This study addressed this gap with a dataset comprising a visual marker with black bars indicating the water level, hereby referred to as the "barcode panel". The method employed various deep-learning algorithms to predict the water level and compared their performance. The proposed approach was evaluated using classic classification and error metrics. The models demonstrated accuracy in detecting the water level. These promising results provide important insights for practical applications and future studies.

Link to graphical and video abstracts, and to code: https://latamt.ieeer9.org/index.php/transactions/article/view/9046

Index Terms—Deep learning, computer vision, flood management, visual marker.

I. Introduction

In terms of natural disasters, floods are among the most impactful, causing significant losses of human lives, financial assets, land degradation, and infrastructure damage [1], [2]. Floods have severely affected Brazil recently [3], with notable events in Rio Grande do Sul State in 2024, Acre State in 2024 and 2023, and Petrópolis City in Rio de Janeiro State in 2022. According to the Brazilian National Water Agency (ANA), floods affected approximately 800,000 individuals in 2020. This number exceeded 1 million people in 2021, and the latest report indicates 1.5 million people affected by flood events in 2022 [4], [5].

While deploying sensors in contact with the measurand to measure water levels may provide accurate results, maintaining such devices requires constant interventions that may be costly in terms of financial resources and time. Furthermore, contact sensors are prone to failure and damage in flooding situations due to direct contact with debris and sand [6]. On the other hand, a computer vision-based system that relies on fixed cameras placed beside a water course is cheaper to deploy in

terms of financial and human resources. The placement of the camera and the embedded computing platform is not limited to the immediate vicinity of the water; it does not need to be in contact with the liquid. This means the equipment can be placed on a nearby pole close to a power source without requiring long wired connections. Essentially, the camera can be deployed just like a simple surveillance camera in an urban environment.

However, the existing research has not adequately studied the use of visual markers for measuring water levels. Previous studies mostly relied on traditional image processing methods, which can be affected by lighting, camera positions, and weather conditions [7]. Recognizing this gap, this study has chosen to investigate different deep learning techniques combined with a stainless steel panel featuring evenly spaced rectangular bars, hereby referred to as a "barcode panel". The number of bars above the water surface is designed to be inversely proportional to the water level.

The barcode panel, a mechanical plate with no need for a power supply and minimal maintenance requirements (as long as it remains fixed to the creek wall), adds little complexity to this ground camera approach. As a result, this setup can be deployed to more points to monitor flood risk in a given area, including remote locations that are difficult to access.

The patent BR 10 2023 004081 0 [8] describes a similar panel as the one used in this study. The invention describes an elongated rigid body with markings to create high-contrast patterns. The barcode panel was also described by Domingues et al. [9]. The present work used it along with a dataset of images of this panel immersed in a pool under several different conditions. It employed classification, regression, object detection, and Siamese network algorithms to evaluate how each type of model would handle the task and compare the results between them. The main idea is to determine the number of black bars or the region of the panel that is above the water level. This information would enable a damage control system to respond appropriately to the water level. The results demonstrated that the approaches yielded similar outcomes.

II. Related Work

Many studies and approaches have addressed water level detection and flood prediction. This section presents, reviews and discusses studies on Machine Learning, Deep Learning, and Visual Markers.

Lo et al. [6] presented a method for automatically monitoring floods using image processing and remote cyber surveillance systems, achieving instant flood feedback. The authors

considered a flood as an invasion object in the intrusion detection of the surveillance systems and concluded that the system can provide reference for warning actions in small areas, making them more effective. Another approach based on surveillance systems was developed by Pan et al. [10], aiming to create a low-cost system with remote measurement stations and a monitoring center. This system uses cameras, water level analyzers, wireless routers, and a ruler. They evaluated the system using the difference between frames to analyze water variation, Dictionary Learning, and Deep Learning.

Park et al. [11] proposed a system based on multiclass transformers using SpaceNet8 as a dataset to predict flood occurrence while classifying roads and buildings. The results surpassed those of classic CNN models. Wu et al. [12] collected SAR images from 16 flood events in the Yangtze River Basin and divided them for training, testing, and application. They evaluated the use of several CNN models on this dataset and concluded that these models have great potential for near-real-time flood prediction.

Lin et al. [13] proposed a method for automatic water-level detection using a single camera and a water gauge. They employed image processing techniques and co-linearity equations to determine the water level, minimize noise, and used photogrammetric techniques to track camera movements. The results showed that this approach can overcome changing weather conditions and unexpected camera movements to accurately identify the water level. However, too extreme changes in camera position and weather conditions can compromise the detection.

Chen et al. [14] presented a novel method for recognizing water levels from water gauge images. The studies were conducted in Wuyuan City, Jiangxi Province, China. They used Fully Convolutional One-Stage Object Detection (FCOS) in combination with a contextual adjustment to meet the requirements of edge computing and ensure considerable detection accuracy. This contextual adjustment was used with DeepLabv3+ to segment the water gauge area above the water level line. The method achieved an error margin of 1 cm and was capable of dealing with complex scenarios.

Zhang et al. [15] developed a method utilizing surveillance cameras. The authors presumed that the water level line is commonly located where the greatest change in gray color occurs in the water gauge. Utilizing image processing techniques, such as the maximum mean difference between gray and edges, they demonstrated an accuracy of 90%. They encountered challenges with water vibration and floating debris during floods in front of the water gauge.

Vandaele et al. [16] developed an approach based on Deep Learning for automated semantic water segmentation to estimate the water level on a river from camera angles. They acquired the dataset from the Severn and Avon rivers in the United Kingdom. They compared the results with water gauge markers near the cameras and concluded that this approach can be easy and low cost, especially for environments where water gauges are not available.

Sabbatini et al. [17] proposed a solution based on automatic Computer Vision, processing images from a staff gauge. The authors acquired the images from an IoT device, and the solution consists of two modules. The first module deals with power consumption at the edge, selecting frames with good quality, while the second module is implemented on a Cloud server, where the water level is extracted. The model performed well in nighttime environments but had difficulties dealing with sunlight during the day.

The previously mentioned works emphasize the potential of using cameras, deep learning, and visual markers. However, approaches that primarily rely on raw images of water bodies encounter issues with weather variations and lighting conditions. Additionally, some of these methods are limited to specific environments and are not easily generalizable. Other approaches yield good results but come with high implementation costs and are not straightforward to use in various locations, such as those based on SAR images. Consequently, this work integrated the concept of visual markers with various deep-learning techniques to tackle these existing issues and reduce the solution's cost. This study aims to advance research in the field of water level detection and flood prediction, contributing to solutions that are more robust, more generalizable, and easier to implement than the current ones.

III. ARTIFICIAL INTELLIGENCE APPROACHES

As already stated, this study aimed to assess the effectiveness of distinct deep learning algorithms on a dataset for water level measurement. It examined: (i) a classification task, treating the bars of the barcode as categories; (ii) a regression task, estimating the number of bars above the water as a number; (iii) an object detection task, detecting the barcode area above the water; and (iv) two distinct Siamese networks, classifying the number of bars above the water based on a distance metric.

A. Dataset

The dataset consisted of images of a barcode-patterned panel [9] consisting of 7 evenly spaced black bars printed on it, resembling a "zebra" pattern. This panel was constructed from stainless steel to maintain its properties, as stainless steel is not susceptible to degradation effects such as oxidation. The dataset was collected outdoors with exposure to sunlight, which could cause reflectance issues. However, as demonstrated further, the suggested techniques successfully addressed this challenge.

The dataset contains 214 images of the barplate in a controlled environment (i.e., a swimming pool), with various angles and position variations. The Barcode was also adjusted in the water to create different scenarios, and some images include artificial rain, although all the images were taken during daylight. Within this dataset, the number of visible bars above the water ranges from 3 to 7, representing the classes in the dataset. Fig. 1 shows a sample of the dataset, and Fig. 2 shows the class distribution.

B. Classification and Regression

One approach utilized was categorizing the problem into classification and regression. For this, the images from the
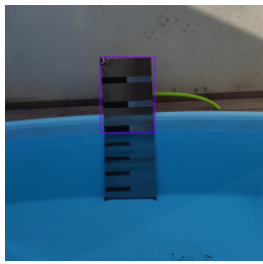
Fig. 1. Barcode panel sample with bounding box highlighting the plate area above the water, indicating class 3.
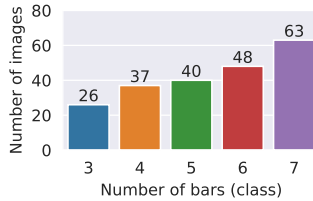

Fig. 2. Classes distribution over the dataset.

dataset were used as inputs. In the classification problem, the results were values belonging to classes 3 to 7, whereas for Regression, the values were continuous numbers in the range of 3 to 7. These numbers denote the number of black bars above the water level. The base network was a ResNet 50 [18] model pre-trained with weights from ImageNet. The backbone was frozen, and additional layers were added: a Global Average Pooling layer added, followed by a Fully Connected Layer with 256 units, 20% Dropout, ReLU activation, and an Output Layer with 5 units and softmax activation. The model was then fine-tuned on the dataset. Fig. 3 presents the model and the newly added layers.

To train the model, the method used the Stratified K-Fold technique, ensuring equal representation of each class across the folds, which is important for handling overfitting and bias. This technique divides the dataset into K folds, where one fold is used to validate the model, and the remaining K-1 folds are used to train the model. This training process is repeated for K iterations, changing each iteration's validation fold so that each fold is used once for validation. This approach allowed us to utilize the available data better and make more confident estimates, as the average performance across all K iterations
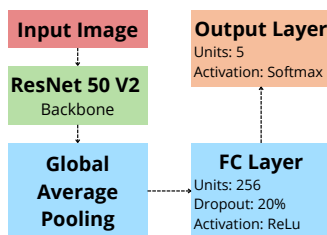

Fig. 3. Classification model architecture: The input image is processed through a frozen ResNet 50 V2 backbone, followed by global average pooling, a fully connected layer, and finally, the output layer with softmax activation.

determines the model's performance. Compared to a hold-out approach, this produces more reliable and better-estimated results.

The number of folds was 5. All the models were trained for 100 epochs. Images were preprocessed and resized to 224×224 pixels, and were loaded in batches of 32. The Adam optimizer was employed with a learning rate of 10^{-4}. Categorical Crossentropy was utilized as the loss function for classification tasks, while Mean Squared Error (MSE) was used for regression tasks. The preprocessing and augmentation techniques applied were pixel normalization to the range $[0, 1]$, random horizontal flipping, random zoom variations up to 40%, random rotations up to 40 degrees, random vertical and horizontal shifts up to 30% of the image dimensions, and the nearest method for filling pixels outside the image boundaries.

C. Object Detection

Another approach employed was object detection, which aimed at identifying the region of the image containing the plate. This region was then extracted from the image and passed through the classification algorithm again. This approach is expected to achieve better generalization, allowing the barcode to be inserted in any region, cropped, and classified.

In the object detection literature, there is a trade-off between two-stage detectors with higher average precision but slower, and single-stage detectors that compute faster at the expense of lower average precision. The most noticeable difference is that two-stage detectors comprise a separate architecture for the proposal of regions of interest that result in the bounding boxes identifying the location and scale of the objects. In contrast, single-stage detectors employ a single neural network for object detection and classification. The most widely used two-stage object detection architecture is the Faster-RCNN and its variants [19]. At the same time, single-stage detectors are represented by the single-shot detectors (SSD) [20], the YOLO family [21], the RetinaNet [22] and others.

The approach hereby presented aimed at understanding how different deep learning techniques could tackle the problem of water level identification based on the barcode panel monitored by a video camera. To provide an overview of the performance of deep learning-based object detectors in identifying the panel, the Faster-RCNN 101 was chosen to represent the two-stage detectors, and the RetinaNet was selected to represent the single-stage detectors. RetinaNet was favored over YOLO architectures because post YOLO-v3 models were released as YOLO without being a direct upgrade over the earlier models of the series. Additionally, the most recent YOLO models are developed by private entities that do not disclose the architecture with scientifically rigorous analysis of their results. Furthermore, RetinaNet has shown promising results for detecting distant objects, making it particularly useful for drone imagery. This capability could be valuable for specific ground camera and barcode panel configurations.

Moreover, although the level of a water stream in a real-world environment can change in a matter of minutes during heavy rainfall events, any changes occurring in intervals at

the scale of milliseconds are negligible. Hence, the issue at hand does not necessitate rapid computation, as is provided by single-shot detectors designed for real-time object detection at several frames per second. Conversely, damage control measures based on flood risk alerts benefit from more accurate predictions and are not affected by delays in scales lower than a few dozen seconds. Therefore, a two-stage detector such as Faster-RCNN can be the most suitable for the task. As a single-stage detector, RetinaNet was included as a reference for comparison purposes.

In both cases, the networks were previously trained on the COCO dataset and then fine-tuned on the barcode dataset. To ensure a direct comparison, all parameters used were identical between the two networks, as were the training and testing data. In this case, a simple train-test split approach was used for testing, as the computational cost for detection is relatively higher than for simple classification or regression. After training both models, the barcodes in all dataset images were detected and cropped, generating two pre-processed datasets, which were used to train new models identical to those in section III-B.

Both models were trained for 250 epochs. Images were preprocessed and resized to 384x384 pixels and were loaded in batches of 16. The learning rate was set to 10^{-3} using the Stochastic Gradient Descent (SGD) optimizer. For the Faster R-CNN, the loss functions employed were Softmax Cross-Entropy for the classification component and Smooth L1 Loss for the bounding-box regression. RetinaNet's classification loss function was Sigmoid Focal Loss, and the bounding-box regression loss was Smooth L1 Loss. These classification and bounding-box regression loss functions are the default settings in the Detectron2 framework. The classification components of these models were not used. In both models, the following augmentation techniques were applied: random horizontal flip, random rotation up to 25 degrees, random zoom variation between 10% and 30%, random noise with an intensity of 0.12, random contrast and brightness variation between 0.1 and 0.5, elastic transformation with an alpha of 1.2 and a sigma of 0.2, and crop and pad up to 40%.

D. Siamese Network I

Siamese networks perform metric learning, a branch of few-shot learning aimed at problems with a small number of samples from each class. Besides, it accounts for class imbalance for these problems [23], which is the situation posed by the class distribution illustrated in Fig. 2. This is why we considered using this technique, as more common methods, such as oversampling, have not shown any improvements in preliminary experiments.

The first approach, called **SNI**, involved adapting a neural network for classification, and the training method followed exactly those of the network described in Section III-B. However, to evaluate the model, the classification head is removed. Train images were processed, and their feature vectors were stored. Then, the test images were processed, and their vectors were compared with the stored ones. To compare the images, this approach used the Euclidean distance, defined

as $d(b,p) = \sqrt{\sum_{i=1}^{n}(b_i - p_i)^2}$, where b represents the stored feature vector, p represents the newly processed vector and n it the dimension of the vector. The pair with the smallest distance to the new feature vector is considered to be from the same class.

The architecture of the SNI Network and the parameters used are the same as those described in section III-B. The test images were resized to 224x224 and normalized to the range $[0,1]$. The augmentation techniques applied were the same as the model in section III-B. To evaluate the model, several tests were conducted. In addition to assessing standard metrics, these tests aimed to evaluate the quantity of images required in the reference database to achieve good results. The tests are as follows:

- **Test 1:** This test consisted of randomly placing in the reference database only one image from each class from the training set, totaling 5 images. Additionally, since the model was trained using the K-Fold method with 5 folds and 5 different sets of training and test data, this test was performed for each K. For each K, the method consisted of selecting images and calculating the metrics 20 times. Thus, the final results were the averages of the 100 times the model was evaluated;
- **Test 2:** This test consisted of repeating Test 1, but this time randomly placing 3 images from each class in the reference database, totaling 15 images.
- **Test 3:** This test consisted of repeating Test 1, but this time placing 5 images from each class in the reference database, totaling 25 images.
- **Test 4:** The final test consisted of placing all training images in the reference database, and evaluating the test only once for each K, resulting in an average over the 5 folds.

E. Siamese Network II

The second implementation, called **SNII**, consists of a classical Siamese Network, in contrast with SNI, which is an adapted version of a classification network into a Siamese network model. In SNII, two identical networks with shared weights provide and process a pair of inputs. At the end, the feature vectors of the images are generated and can be compared. In this network, the loss function must be able to distinguish between pairs of identical images (positives) and different images (negatives). For this, the method employed Contrastive Loss with margin and Euclidean distance, as proposed by [24] and given by:

$$L(Y,D) = \frac{1}{2}YD^2 + \frac{1}{2}(1-Y)\{max(0, m-D)\}^2 \quad (1)$$

where Y is the binary label indicating positive ($Y = 1$) or negative ($Y = 0$) pairs, m is the margin parameter for dissimilar pairs, and D is the Euclidean distance between the feature vectors [25]. The base network was a ResNet 50 [18] model pre-trained with weights from ImageNet. The backbone was frozen, and additional layers were added: a Fully Connected Layer with 256 units, 20% Dropout, ReLU activation, and a final Linear Layer with 128 units. The model
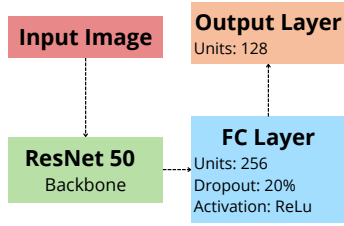
Fig. 4. SNII model architecture: The input image is processed through a frozen ResNet 50 V2 backbone, then a fully connected layer, and finally, the output layer with 128 units.
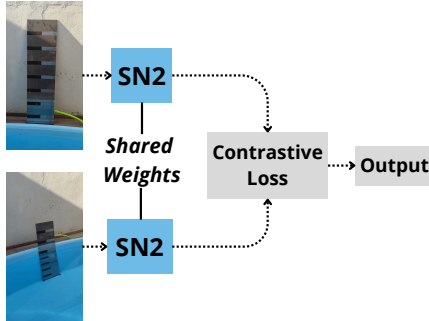

Fig. 5. Training pipeline of the SNII model: Two images are processed by a twin model with shared weights. The resulting vectors are then processed and compared using Contrastive Loss, producing an output. This output is utilized to predict whether the pair is similar or dissimilar, and the weights are updated accordingly.

was then fine-tuned on the barcode dataset. Fig. 4 shows the model architecture and Fig. 5 illustrates the proposal.

The model was trained for 250 epochs. Images were resized to 224x224, normalized to the range [0,1], and loaded in batches of 16. The SGD optimizer, with a learning rate of 10^{-3} and momentum of 0.9, was employed. The loss function was Contrastive Loss, as previously mentioned. The margin parameter was set to 1.4. The model was trained with a stratified train-test split, using 80% of the data for training and 20% for testing. With this data, training pairs were created. It is important to carefully create pairs, ensuring an equal number of positive and negative pairs. Additionally, the class that creates the fewest positive pairs should limit the number of positive pairs from other classes to maintain balance. The augmentation techniques applied were: pixel normalization to the range [0,1], random horizontal flipping, random zoom variations up to 40%, random rotations up to 40 degrees, random vertical and horizontal shifts up to 30% of the image dimensions, mean equal to [0.485, 0.456, 0.406] e standard deviation equal to [0.229, 0.224, 0.225], and the nearest method for filling pixels outside the image boundaries.

The test step followed the SN1 model, using Euclidean distance to compare the test images with the images in the reference database. The same tests were conducted with a slight difference: this model was trained five different times. Therefore, Tests 1, 2, and 3 were conducted 50 times instead of 20, and Test 4 was conducted one time. All the results of the tests are the average results of the five models.

F. Experimental Setup

The design of the experimental setup aimed to facilitate direct comparisons between the models presented in this paper and similar studies reported in the literature. For classification and regression algorithms, this study used the Keras framework with a TensorFlow backend. For object detection, this study employed the Detectron2 framework. Siamese Networks were implemented using both PyTorch and Keras frameworks. All training processes were conducted on Google Colaboratory, which offers a free environment with access to a free GPU.

IV. RESULTS

The results of the experiments performed can be shown in terms of classification metrics (i.e., whether the number of bars above the water surface was correctly classified or not) or regression metrics (i.e., how large was the error between the predicted and the actual number of bars above the water surface). Table I showcases the classification metrics for the classification models and Siamese Networks in the experiments. This section considers the precision, the recall, and the F1-score. Besides being more adequate than the accuracy for imbalanced datasets, these metrics are among the most commonly used in related work, allowing for comparisons with other models in the literature.

TABLE I
CLASSIFICATION METRICS FOR CLASSIFICATION MODELS
AND SIAMESE NETWORKS TESTS

	Precision	Recall	F1-Score
Classi. Original Images	0.85	0.84	0.84
Classi. Faster-RCNN crop	0.87	0.88	0.87
Classi. RetinaNet crop	0.85	0.85	0.85
SNI Test 1	0.70	0.70	0.66
SNI Test 2	0.77	0.76	0.73
SNI Test 3	0.82	0.82	0.81
SNI Test 4	0.93	0.92	0.92
SNII Test 1	0.71	0.73	0.68
SNII Test 2	0.75	0.76	0.74
SNII Test 3	0.78	0.79	0.77
SNII Test 4	0.82	0.85	0.82

Table II presents the regression metrics for the classification and regression models, as well as for the Siamese Networks. The table shows the Mean Average Error (MAE), the Mean Squared Error (MSE), the Coefficient of Determination or R^2, and Root Mean Squared Error (RMSE). The evaluations are with respect to the number of bars above the water surface. For instance, the MAE of a given model corresponds to the average difference between the predicted and the actual number of bars above the water surface.

Table III shows the results for the object detection models, comparing the Faster-RCNN and RetinaNet approaches. The metrics considered were the Average precision (AP), Average Recall (AR) and Intersection over Union (IoU). Fig. 6 shows the confusion matrix corresponding to the best-performing model in the results, SNI Test 4.

The classification models using original images, images cropped after Faster R-CNN detection, and images cropped after RetinaNet detection show very similar results in terms

TABLE II

ERROR METRICS FOR CLASSIFICATION MODELS,
REGRESSION MODELS AND SIAMESE NETWORKS

	MAE	MSE	R2	RMSE
Classification Orig. Images	0.192	0.277	0.848	0.526
Classification RCNN Crop	0.177	0.262	0.858	0.512
Classification RNet Crop	0.168	0.205	0.894	0.453
Regression Orig. Images	0.502	0.474	0.614	0.688
Regression RCNN Crop	0.373	0.234	0.842	0.484
Regression RNet Crop	0.416	0.284	0.802	0.533
SNI Test 1	0.603	1.349	0.273	1.161
SNI Test 2	0.411	0.898	0.552	0.948
SNI Test 3	0.331	0.656	0.653	0.810
SNI Test 4	0.103	0.168	0.906	0.410
SNII Test 1	0.452	0.971	0.435	0.985
SNII Test 2	0.383	0.847	0.528	0.904
SNII Test 3	0.323	0.656	0.630	0.810
SNII Test 4	0.274	0.619	0.651	0.787

TABLE III

OBJECT DETECTION METRICS FOR FASTER-RCNN AND
RETINANET MODELS

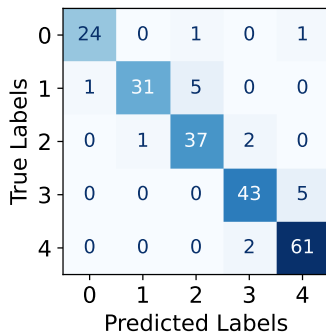|	Faster R-CNN	RetinaNet
AP @ IoU=0.5:0.95	0.837	0.898
AP @ IoU=0.75	0.982	0.983
AP @ IoU=0.5	0.996	0.983
AR @ IoU=0.5:0.95	0.871	0.922
IoU	0.909	0.936

Fig. 6. Confusion Matrix for SNI Test 4

of precision, recall, and F1-score. The results after Faster R-CNN cropping are marginally superior, as shown in Table I. Analyzing the errors of these models reveals that the errors are slightly lower after cropping with RetinaNet. Overall, these models achieved precision greater than 85%, and the RMSE was below 0.526, indicating less than one error bar on average per model. This demonstrates the accuracy of the models.

Regarding the object detection models, Table III shows that RetinaNet delivers the best results across all metrics. Both models achieve an IoU greater than 0.909, with an Average Recall above 0.871, indicating that the models are effectively cropping the regions containing the bars. The Average Precision exceeds 0.837. However, even though RetinaNet outperforms in detection metrics, this difference is not as evident in the classification results after image cropping. This suggests that precise cropping may not be crucial for achieving good classification performance. Considering the complete solution, it would be more advantageous to use RetinaNet for

real-time applications, as it is a single-shot detector, meaning it has a shorter execution time compared to Faster R-CNN.

Another interesting point is the performance of the complete SN1, as evidenced by Test 4 in Tables I and II. Despite using the same base architecture as the classification models, transforming it into a Siamese Network increased classification metrics by approximately 8%. The error metrics also decreased. When comparing the performance of SN2 in Test 4, SN2 is closer to the classification networks and performs approximately 10% lower than SN1. Evaluating Tests 1, 2, and 3 of the Siamese Networks, it can be observed that with the available image dataset, the results improve over time, requiring 5 images per class to achieve 80% accuracy with an RMSE of 0.810. In situations with limited images, the model could start with a few images and gradually expand the dataset to achieve good results. It is noteworthy that all models used images with clear water. Notably, especially in detection, the models can distinguish between what is above and below the water, detecting only the region above the threshold.

V. CONCLUSIONS

This paper introduces the utilization of various deep learning algorithms for water level measurement in conjunction with the barcode panel. All algorithms were evaluated using the same metrics. In the case of the object detection algorithm, the image cropping and classification steps were added to create a more comprehensive model. For the Siamese networks, tests on the quantity of images in the database were conducted to assess how this quantity influences the results.

Using an object detection model followed by a classification network after image cropping is a more robust solution and would likely yield better results in terms of generalization. However, transforming the network into an SN1 type yields better metric results but requires a database for comparison, thus increasing the complexity of the solution. An SN2-type network is faster, but its implementation is more complex, and the results are slightly worse.

In most cases, errors occur between closely related numbers of bars above the water surface. Consequently, the magnitude of the error impact in a real-world scenario is smaller. While a misclassification to a distant class (e.g., actual class is 3, predicted class is 7) could result in an actual error of up to 85 cm, potentially indicating a false flooding event, a misclassification between classes corresponding to adjacent bars in the panel (e.g., actual class is 3, predicted class is 4) yields an error around 15-25 cm in water level.

Future work will involve using images that are not from this dataset, such as images taken from rivers. This will allow testing the models in real-world conditions, with a variety of image types and lighting conditions.

REFERENCES

[1] C. K. Seigerman, N. S. Leite, E. S. P. Martins, and D. R. Nelson, "At the extremes: Assessing interrelations among the impacts of and responses to extreme hydroclimatic events in ceará, northeast brazil," *Journal of Hydrology*, vol. 632, p. 130850, 2024. 10.1016/j.jhydrol.2024.130850.

[2] M. H. K. Valentini, S. Beskow, T. L. C. Beskow, C. R. de Mello, F. Cassalho, and M. E. S. da Silva, "At-site flood frequency analysis in brazil," *Natural Hazards*, vol. 120, no. 1, pp. 601–618, 2024. 10.1007/s11069-023-06231-3.

[3] M. P. de Oliveira Roza, R. A. Cecílio, S. S. Zanetti, M. C. Abreu, G. B. Lyra, and G. B. Reis, "Natural disasters related to rainfall trends in espírito santo, southeastern brazil," *Theoretical and Applied Climatology*, vol. 155, no. 2, pp. 1451–1466, 2024. 10.1007/s00704-023-04703-x.

[4] Brazilian National Water and Basic Sanitation Agency - ANA, *Conjuntura dos recursos hídricos no Brasil 2022: informe anual*. 2023.

[5] Brazilian National Water and Basic Sanitation Agency - ANA, *Conjuntura dos recursos hídricos no Brasil 2023: informe anual*. 2024.

[6] S.-W. Lo, J.-H. Wu, F.-P. Lin, and C.-H. Hsu, "Cyber surveillance for flood disasters," *Sensors*, vol. 15, no. 2, pp. 2369–2387, 2015. 10.3390/s150202369.

[7] C. M. Ranieri, T. L. D. e. Souza, M. Nishijima, B. Krishnamachari, and J. Ueyama, "A deep learning workflow enhanced with optical flow fields for flood risk estimation," *Applied Intelligence*, vol. 54, pp. 1–22, 2024. 10.1007/s10489-024-05466-2.

[8] J. Ueyama, A. R. da Silva, C. M. Ranieri, D. Q. G. Batista, I. F. da Silva, and P. H. Fini, "Dispositivo de indicação de nível de Água e método para aferição de nível de Água." Patent BR 10 2023 004081, mar 2023. Filed in March 2023.

[9] G. M. Domingues Filho, C. M. Ranieri, D. Q. G. Batista, and J. Ueyama, "Enhancing water level identification with a barcode-patterned panel and machine learning," in *Anais do XX Encontro Nacional de Inteligência Artificial e Computacional*, pp. 668–682, SBC, 2023. 10.5753/eniac.2023.234311.

[10] J. Pan, Y. Yin, J. Xiong, W. Luo, G. Gui, and H. Sari, "Deep learning-based unmanned surveillance systems for observing water levels," *Ieee Access*, vol. 6, pp. 73561–73571, 2018. 10.1109/ACCESS.2018.2883702.

[11] J.-C. Park, D.-G. Kim, J.-R. Yang, and K.-S. Kang, "Transformer-based flood detection using multiclass segmentation," in *2023 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 291–292, IEEE, 2023. 10.1109/BigComp57234.2023.00056.

[12] X. Wu, Z. Zhang, S. Xiong, W. Zhang, J. Tang, Z. Li, B. An, and R. Li, "A near-real-time flood detection method based on deep learning and sar images," *Remote Sensing*, vol. 15, no. 8, p. 2046, 2023. 10.3390/rs15082046.

[13] Y.-T. Lin, Y.-C. Lin, and J.-Y. Han, "Automatic water-level detection using single-camera images with varied poses," *Measurement*, vol. 127, pp. 167–174, 2018. 10.1016/j.measurement.2018.05.100.

[14] C. Chen, R. Fu, X. Ai, C. Huang, L. Cong, X. Li, J. Jiang, and Q. Pei, "An integrated method for river water level recognition from surveillance images using convolution neural networks," *Remote Sensing*, vol. 14, no. 23, p. 6023, 2022. 10.3390/rs14236023.

[15] Z. Zhang, Y. Zhou, H. Liu, L. Zhang, and H. Wang, "Visual measurement of water level under complex illumination conditions," *Sensors*, vol. 19, no. 19, p. 4141, 2019. 10.3390/s19194141.

[16] R. Vandaele, S. L. Dance, and V. Ojha, "Deep learning for automated river-level monitoring through river-camera images: an approach based on water segmentation and transfer learning," *Hydrology and Earth System Sciences*, vol. 25, no. 8, pp. 4435–4453, 2021. 10.5194/hess-25-4435-2021.

[17] L. Sabbatini, L. Palma, A. Belli, F. Sini, and P. Pierleoni, "A computer vision system for staff gauge in river flood monitoring," *Inventions*, vol. 6, no. 4, p. 79, 2021. 10.3390/inventions6040079.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 10.1109/CVPR.2016.90.

[19] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37, Springer, 2016.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017. 10.1109/ICCV.2017.324.

[23] P. Kumari and K. Seeja, "One shot learning approach for cross spectrum periocular verification," *Multimedia Tools and Applications*, vol. 82, no. 13, pp. 20589–20604, 2023.

[24] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, vol. 2, pp. 1735–1742, IEEE, 2006. 10.1109/CVPR.2006.100.

[25] I. Melekhov, J. Kannala, and E. Rahtu, "Siamese network features for image matching," in *2016 23rd international conference on pattern recognition (ICPR)*, pp. 378–383, IEEE, 2016. 10.1109/ICPR.2016.7899663.

Gabriel M. Domingues Filho is an undergraduate student in Electrical Engineering with an emphasis on electronics at the São Carlos School of Engineering, University of São Paulo. His research interests lie in Artificial Intelligence, particularly in Deep Learning and Computer Vision. Gabriel has participated in various research projects, primarily focused on water level detection for flood events. His career goals include advancing the field of AI and contributing to innovative technological solutions.

Caetano M. Ranieri is an Assistant Professor at the Institute of Geosciences and Exact Scientes of the Sao Paulo State University (IGCE-UNESP). He was a postdoctoral research fellow at the University of Sao Paulo (USP), with research focused on Artificial Intelligence in the context of the Internet of Things. He graduated in Computer Science at UNESP (2013) and did his Master's degree (2016) and Ph.D. (2021) at USP. During his Ph.D., he worked as a visiting scholar at Heriot-Watt University, Scotland (2020).

Saulo N. Matos is a Ph.D. student at the University of São Paulo (USP), with research focused on Machine Learning, Instrumentation, and Internet of Things. He holds a master's degree in Instrumentation, Control, and Automation of Mining Processes from the University of Ouro Preto and the Vale Technological Institute (2022). He graduated in Control and Automation Engineering from the Federal University of Ouro Preto (2020). He has published papers and patents on instrumentation, embedded systems, machine learning, and control theory.

Rodolfo I. Meneguette is a professor at University of São Paulo (USP). He received his Bachelor's degree in Computer Science from the Paulista University (UNIP) in 2006. He received his master's degree in 2009 from the Federal University of São Carlos (UFSCar). He received his doctorate from the University of Campinas (Unicamp) in 2013. In 2017, he did his post-doctorate at the PARADISE Research Laboratory, University of Ottawa, Canada. His research interests are in vehicular networks, resources management, flow of mobility, and vehicular clouds.

Jó Ueyama is a Full Professor at the Institute of Mathematics and Computer Science (ICMC) of the University of São Paulo (USP). He has been a Brazilian Research Council (CNPq) fellow since 2014. He received his Ph.D. in computer science from the University of Lancaster in 2006 and was a research fellow at the University of Kent at Canterbury before joining USP. Jó has a publication record with 72 journal articles and over 100 conference papers. His research interests are focused on Computer Networks, Security, and Blockchain.