Journal of Hydroinformatics

Journal of Hydroinformatics Vol 00 No 0, 1 doi: 10.2166/hydro.2024.013

Deep learning for automated trash screen blockage detection using cameras: Actionable information for flood risk management

Remy Vandaele[a,b,*], Sarah Lance Dance[a,c,d] and Varun Ojha[e]

[a] Department of Meteorology, University of Reading, Whiteknights Campus, Brian Hoskins building, Reading RG6 6ET, UK
[b] Joint Centre of Excellence in Environmental Intelligence, University of Exeter, Streatham Campus, Laver Building, Exeter EX4 4QE
[c] Department of Mathematics and Statistics, University of Reading, Whiteknights Campus, Mathematics building, Reading RG6 6AX, UK
[d] National Centre for Earth Observation, University of Reading, Whiteknights Campus, Brian Hoskins building, Reading RG6 6ET, UK
[e] School of Computing, Newcastle University, Urban Sciences Building, 1 Science Square, Newcastle upon Tyne NE4 5TG, UK
*Corresponding author. E-mail: r.vandaele@exeter.ac.uk

RV, 0000-0002-0693-8011

ABSTRACT

Trash screens are used to prevent debris from entering critical parts of rivers. However, debris can accumulate on the screen and generate floods. This makes their monitoring critical both for maintenance and flood modeling purposes (e.g., local forecasts may change because the trash screen is blocked). We developed three novel deep learning methods for trash screen maintenance management consisting of automatically detecting trash screen blockage using cameras: a method based on image classification, a method based on image similarity matching, and a method based on anomaly detection. To facilitate their use by end users, these methods are designed so that they can be directly applied to any new trash screen camera installed by the end users. We have built a new dataset of labeled trash screen images to train and evaluate the efficiency of our methods, in terms of both accuracy and implications for end users. This dataset consists of 80,452 trash screen images from 54 cameras installed by the Environment Agency (UK). This work demonstrates that trash screen blockage detection can be automated using trash screen cameras and deep learning, which could have an impact on both trash screen management and flood modeling.

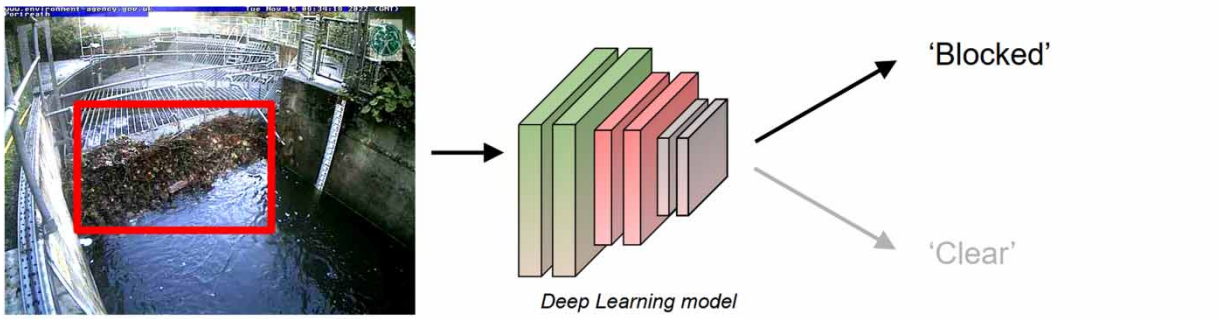
Key words: computer vision, deep learning, flood management, machine learning, river management, trash screen

HIGHLIGHTS

- Development and analysis of three deep learning-based trash screen monitoring methods using cameras, tailored for various investment levels.
- Binary classification: ready-to-use, no additional labeling or retraining required.
- Image similarity matching: requires minimal user annotation for improved accuracy.
- Anomaly detection: able to use unlabeled datasets.
- Network and dataset provided to support further research.

GRAPHICAL ABSTRACT

Automated monitoring of trash screens by processing trash screen camera images with deep learning approaches.



1. INTRODUCTION

Trash screens are structures made of evenly spaced bars or mesh, installed at the inlet of culverts or drainage systems to prevent debris from causing a blockage that could progress further downstream and damage critical assets (e.g., pumping stations or pipes) (Benn *et al.* 2019). The bar spacing is typically designed to only trap debris that could potentially cause damage.

As depicted in Figure 1, the trash screen can get blocked once debris starts bridging across multiple bars and then starts to progressively accumulate, blocking the waterway and potentially causing flood events (Blanc 2013; Benn *et al.* 2019). It is thus of the greatest importance to clear the blocked trash screens, especially before episodes of heavy rainfall (Speight *et al.* 2021). Practically, this means that local authorities need to develop better strategies for the clearing of these assets. Currently, these trash screens are maintained through manual inspection of cameras or regular schedules, but this can prove inefficient in situations where the clearing of a particular trash screen needs to be prioritized.

In addition, while the blockage of trash screens can severely worsen flood events (Streftaris *et al.* 2013), this information has, to our knowledge, never been integrated in flood forecasting systems. Recent advances in the development of simulation libraries for near real-time flood inundation forecasting systems are based on the automated selection of a precomputed flood inundation map, using observed or modeled river discharge to inform the map selection (e.g., Hooker *et al.* 2023). Knowing the locations and states of trash screens could thus be considered valuable information for the automated selection of such flood inundation maps. For example, the simulation library could contain maps computed according to different trash screen blockage scenarios, and the correct map is selected based on knowledge of the trash screen state.

Figure 1 | Midsomer Norton trash screen. On 15 October 2022 (left), the trash screen is clear. On 8 November 2022 (right), debris is blocking the waterway.

Initial attempts to predict trash screen blockage involved examining correlations between blockage occurrences and relevant numerical parameters. However, these methods lack general applicability and, to our knowledge, have not been implemented in real-world scenarios: Wallerstein *et al.* (2010) conducted a study using a dataset from 140 Belfast trash screens, revealing varying likelihoods of blockage in different catchments, with rainfall and time of year identified as significant factors. Building on this, Wallerstein & Arthur (2013) proposed monthly equations that incorporated parameters such as channel slope and rainfall to predict blockage probability. Subsequent research by Streftaris *et al.* (2013) improved the accuracy of these equations by employing a Bayesian model. Despite these advancements, the authors note that the validation of such models is hampered by data limitations, and the urban bias in the dataset raises concerns about their broader applicability.

Over recent years, the price of cameras has dropped and the accessibility of mobile networks has made them a practical tool for monitoring. Many UK organizations use cameras and provide image feeds available online. For example, Highways England has approximately 1,500 traffic cameras distributed over England (Highways England 2023), and, most importantly in our case, the Environment Agency has installed around 350 cameras to manually monitor rivers and trash screens (EA 2023). This makes the use of cameras for automated trash screen monitoring an attractive possibility.

Several studies explored the application of deep learning and computer vision in flood monitoring, as exemplified by works from Vandaele *et al.* (2023) and Moy de Vitry *et al.* (2019). However, these algorithms focus on detecting river levels, which is relevant to operational trash screen maintenance but does not provide all of the information needed. Trash screens can only be effectively cleared when river levels are low. During periods of elevated river levels, safety concerns render screen clearing impractical. In contrast to these flood monitoring approaches, our research centers on trash screen blockage detection. Notably, our method distinguishes itself from existing literature that addresses trash-related tasks. For instance, Jaikumar *et al.* (2021) employed deep learning image segmentation to identify water bottles in camera images, while Tharani *et al.* (2021) and Nguyen & Tran (2022) compared object detection and segmentation methods for detecting floating trash on water surfaces. Nevertheless, the applicability of these works to our trash screen monitoring task is limited due to disparities in image types (trash screen images versus river images), trash characteristics (plastics bottles versus miscellaneous trash including objects such as tree branches), and the fundamental task difference (locating trash versus determining if the trash is obstructing the screen). This underscores the unique challenges of our context, necessitating a tailored approach accounting for the distinct aspects of trash screen blockage detection.

In a more closely related task, efforts have been made to automate the detection of blockages in images of culverts through the application of convolutional neural networks, as demonstrated in studies by Iqbal *et al.* (2021) and Iqbal *et al.* (2022). These studies employed synthetic images created in a hydrology lab rather than images from real culverts. Importantly, both the training and testing phases utilized identical camera fields of view, limiting the assessment of the generalizability of these methods to new fields of view in natural environments. The reliance on synthetic images and the constraint of consistent camera perspectives raise questions about the applicability of these automated detection methods in real-world scenarios with diverse and unpredictable camera setups.

This literature review highlights that, to the best of our knowledge, we are the first to propose an approach to tackling the problem of automated trash screen blockage detection using cameras. Our goal is to fill this gap by developing methods tailored for the detection of trash screen blockage from any camera field of view: these approaches must assign one of two labels (*clear* or *blocked*) to a new trash screen camera image and should be deployed fairly simply during the installation of a new camera, without a large image labeling or retraining effort required. To provide end users with informed choices, we have developed three methods, each aligning with different levels of investment. These investment levels encompass considerations such as network training, availability of data, and effort involved in data labeling.

- The development of three deep learning methods allowing the automatic monitoring of trash screen blockage using cameras of *previously unseen* trash screens. While every method takes a trash screen image as input and outputs its label (*blocked* or *clear*), they address different situations:
  - The *binary classification method* allows end users to directly reuse the network and weights provided with this article without any additional data labeling or retraining required.
  - The *image similarity matching method* is based on the idea that an end user could easily annotate a few samples (5–10 samples) of images from new cameras to improve method's accuracy. Note that this method does not need to be trained specifically for the new camera.

- The *anomaly detection method* does not need a labeled dataset to be trained to estimate trash screen blockage and can thus easily take advantage of growing amounts of unlabeled recorded images stored by end users.
- An in-depth analysis and comparison of the performance of these methods to help end users make the most relevant choice.
- The trained networks and the dataset of 80,452 trash screen images manually annotated with blockage labels to foster the development of new automated methods (see our *Data Availability Statement*).

2. TRASH SCREEN CAMERA DATASET

2.1. Data acquisition

The images were downloaded from the UK EA South West CCTV Camera website (EA 2023). In January 2022, the website provided the feed of 170 CCTV cameras. Among these, we identified 120 cameras that were observing trash screens.

The trash screens are located across the South West of England to protect critical assets from debris. We are not aware of any available metadata on the assets that they are protecting. According to Benn *et al.* (2019), there are no official guidelines to decide on the installation of a trash screen in the United Kingdom. However, we could observe that the trash screens captured by the cameras had different designs, in terms of both sizes and bar spacing. Note that during their study of trash screens in Belfast, Wallerstein *et al.* (2013) observed that these parameters were related to first-hand knowledge (types of debris to block) and the size of the channel on which the screen is located.

The cameras typically produce one image every hour. These images are available online during a week-long period. In consequence, we curated a dataset of trash screen camera images by downloading the available images between February 2022 and December 2022.

During that time, we analyzed the camera feeds and removed those presenting problems (e.g., night-time images, camera fields of view without trash screen, and so on) from the set of selected cameras. Examples are given in Supplementary Material (Figure S1).

2.2. Data labeling

We labeled the images of the cameras with one of three labels:

- *Blocked*, if the trash screen in the image is blocked by debris and prevents the river from flowing normally. As shown in Figure 2, this means that a screen partially blocked by debris can be considered blocked.
- *Clear*, if there is little to no debris.
- *Other*, if the state of the trash screen is hardly visible or if it is unclear whether the debris in the image could cause blockage. We use this category to avoid images representing borderline situations in our training and test set.

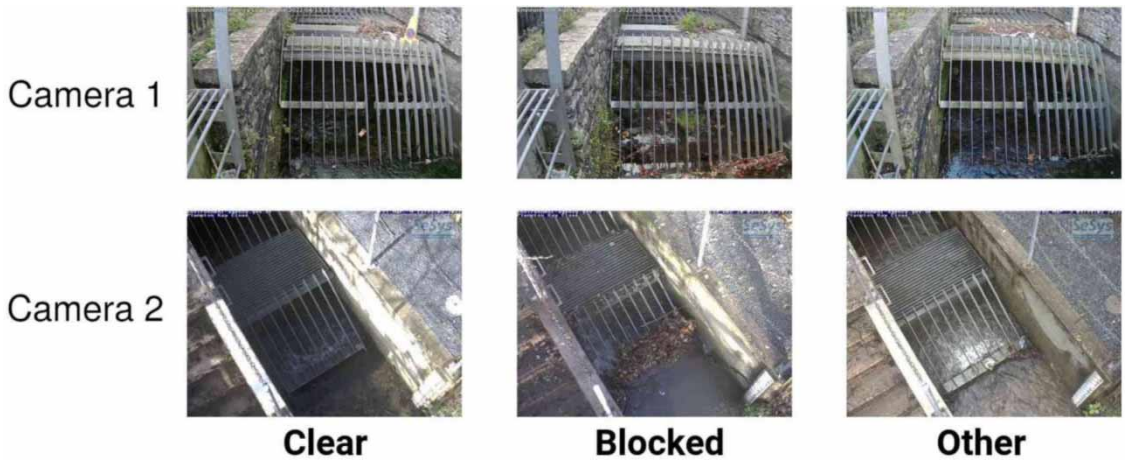


Figure 2 | Examples of manual labels associated with images of two cameras. The trash screens in the clear images contain little to no debris. The debris presented in the blocked images are clearly preventing the water from flowing normally. In the other examples, while we can see relatively high water levels, it is unclear if the debris is actually preventing the water from flowing normally.

Examples of labeled images are shown in Figure 2. We annotated the images of a given camera sorted by their timestamps. As the blockage events typically happened at specific times (e.g., after heavy rain), this allowed us to quickly annotate large sequences of images with the same label and thus reduce the annotation time to approximately 0.5 s of annotation time per image.

2.3. Manual localization of the trash screen

With the heterogeneity of trash screen types and limited availability of camera fields of view, we assumed that the location of the trash screen within the image was known. We think that this assumption has little impact on our experiments. Indeed, the detection of the trash screen location would be a complex task for a deep learning algorithm, while its localization by a human observer during the installation of a fixed trash screen camera requires little effort. Hence, we manually selected the location of the trash screen using a rectangle annotation as shown in Figure 3. The image was then cropped to these rectangles.

2.4. Dataset statistics

The final dataset is made of 54 cameras. In total, 80,452 images were labeled of which 42,782 images were labeled with a 'clear' label, 11,612 with a 'blocked' label, and 26,058 with an 'other' label. While the number of images is large, they come from the same 54 trash screen camera fields of view. As we explained, the images labeled with 'other' are ignored for the training of the network.

3. METHODS

This work explores several methods that we developed through informal discussions with practitioners. These methods address the problem of trash screen blockage detection by requiring different levels of investment (new data labeling, available data, computing power) from the end users. However, once these methods are trained and their requirements are fulfilled, all these methods take a *trash screen camera image as input* and output a blockage score that can then be thresholded to decide whether to assign the image in the 'clear' or 'blocked' category. The method generalizes to any camera and does not need to be retrained when a new camera is installed. This is necessary to minimize the time and workflow needed for manual interventions (e.g., labeling camera images) when setting up a new camera and reducing computing needs. These three methods are summarized in Figure 4 and described further in this section.

Figure 3 | Manual selection of the trash screen locations (rectangles in red) during the preprocessing of the images.

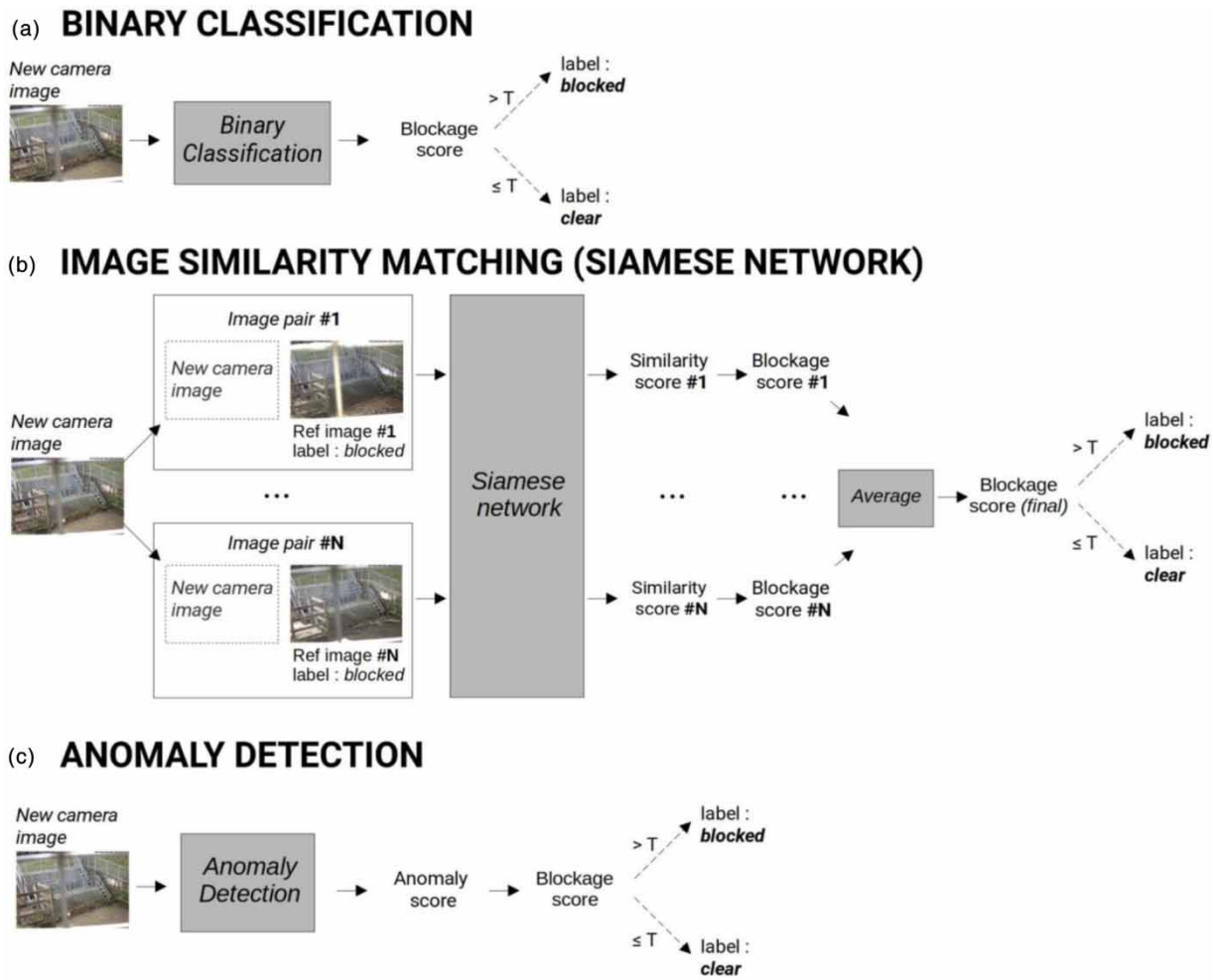


Figure 4 | Estimation process for the automated trash screen blockage detection method. (a) Binary classification is described in Section 3.1. (b) Image similarity matching is described in Section 3.2. (c) Anomaly detection is described in Section 3.3.

3.1. Binary classification method

With this method, we train and produce a new deep learning model that can be directly used without any additional constraint or setup by end users looking for an easy way to integrate an automated trash screen blockage detection with their new cameras.

We implemented this method using a convolutional neural network (CNN). A CNN is a deep learning algorithm designed to process structured grid-like data such as images. CNNs employ convolutional layers to automatically learn hierarchical patterns and features directly from the data (LeCun *et al.* 2015). The network is trained using the labeled images from the training set described in Section 2. As shown in Figure 4(a), this binary classification network takes as input an image from a new camera and outputs a blockage score value between 0 and 1 that can either be thresholded to 'clear' (0) or 'blocked' (1).

A ResNet-50 network architecture (He *et al.* 2016) was chosen as the binary classification network. This architecture is known to perform very well in image classification tasks and its implementation is easily available in various deep learning libraries, such as PyTorch (Paszke *et al.* 2017). Similar to our previous work (Vandaele *et al.* 2021), the last layer of the network was modified to match the needs of our task: indeed, the last layer of ResNet-50 is made of 1,000 neurons to provide 1,000 values for the ImageNet classification task (Deng *et al.* 2009). Given that our classification task is binary, we replaced this layer with a layer of two neurons.

To train the network, we rely on a transfer learning approach (Tan *et al.* 2018), typically used to improve the performance of classification networks: instead of training the network from scratch, we fine-tune the weights of the network that was

optimized over ImageNet. This transfer learning approach proved to consistently outperform training from scratch in many image classification tasks (Sabatelli *et al.* 2018; Vandaele *et al.* 2021).

3.2. Image similarity matching method

With this method, we produce a new model for trash screen blockage detection that considers and takes advantage of the fact that the manual labeling of a handful of images could be a low-cost, high-reward task for end users if it means that it would assist models at obtaining better accuracies than binary classifiers. However, this method also enforces the constraint of a global model that would not need to be retrained for each camera.

The idea of this method is to create a model that can learn the similarity between two images coming from the same camera. The similarity between a new image from the new camera and the few labeled images from this camera can then be used to obtain a blockage score.

We chose to implement this method by using a Siamese network (Bromley *et al.* 1993; Chicco 2021). Siamese network is a type of neural network architecture that consists of two identical subnetworks with shared weights. As shown in Figure 4(b), Siamese networks are designed to compare two images: they take two images as input and learn to output a value related to how similar the images are, the similarity score, typically a known and labeled *reference* image and a new image.

In our case, we use the Siamese network with a labeled image from the new camera as reference and compute the difference with new images from the same camera. The network uses our training dataset to learn that two images of the same camera look similar if they are both '*clear*' or both '*blocked*'. If one image is '*clear*' and the other is '*blocked*', then the two images are not similar.

The estimation process is shown in Figure 4(b). Once trained, the Siamese network takes as input a pair of images from the same camera: the new unlabeled image **I** that we want to label and one *reference* image **R** that is labeled *clear* or *blocked*. The output of the Siamese network is a similarity score ranging from 0 to 1. The value of the score indicates whether the images share the same blockage label. In consequence, knowing the label of an image **R**, the blockage score of image can be computed as follows:

$$\text{Blockage score}\,(I, R) = \begin{cases} \text{Similarity score}\,(I, R) & \text{if label}\,(R) = 'blocked' \\ 1 - \text{Similarity score}\,(I, R) & \text{if label}\,(R) = 'clear' \end{cases} \tag{1}$$

The final blockage score for an unlabeled image can be calculated by averaging over the results from several pairs of images, with N pairs of images made of the same new image paired with N different *reference* images.

We used a ResNet-50 (He *et al.* 2016) network as the identical subnetwork with shared weights in the Siamese network. The two images passing through the Siamese network first go through the ResNet-50, which gives two image outputs. These two outputs are then multiplied together through an element-wise multiplication and then passed through a succession of three fully connected linear layers (1000×512, 512×64; 64×1) each followed by a sigmoid activation function. As noted, the Siamese network outputs a single blockage score value.

3.3. Anomaly detection method

With this method, we produce a new model to investigate if end users could use a large dataset of unlabeled images (unsupervised) to see whether the tedious labeling of the training set could be avoided altogether.

We used an anomaly detection method for this task. In our context, anomaly detection involves identifying patterns or instances that deviate significantly from the norm within a dataset, such as rare events or outliers (see Chandola *et al.* 2009 for more details). As we found in Section 2.4, blocked images are a rare occurrence in the training dataset, with 14% of images labeled as blocked. As anomaly detection methods aim at finding the anomalies in a dataset, in our case, the method should find the blocked trash screens. The estimation process is shown in Figure 4(c). Once trained, the anomaly detection model will take as input an image from a new camera and output an anomaly score, which in our case will be the blockage score.

We implemented this method by using a recent anomaly detection approach developed by Rippel *et al.* (2021). In this work, the authors fit a multivariate Gaussian distribution on a deep representation of the training images. A deep representation of an image refers to a high-level abstract representation of the image that is learned by a deep neural network. It is obtained by passing the image through multiple layers of the neural network, where each layer extracts and refines different features or

patterns from the input image. The deep representation obtained from a deep neural network can be considered as a compressed and abstract representation of the image that preserves important visual characteristics.

Using the same deep representation for a new image, the authors compute its Mahalanobis distance to the multivariate Gaussian distribution to produce an anomaly score. The Mahalanobis distance $D(x)$ is calculated using the equation:

$$D(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{2}$$

where x is the deep representation of the new image, μ is the mean vector of the multivariate Gaussian distribution, and Σ is the covariance matrix of the multivariate Gaussian distribution.

Using the Mahalanobis distance is crucial in this method because it accounts for the correlations between different features in the deep representation space. This ensures that the anomaly score accurately reflects the deviation of a new image from the distribution captured by the training data, considering the underlying covariance structure of the multivariate Gaussian distribution model. Thus, the Mahalanobis distance enhances the sensitivity of anomaly detection, leading to more robust and accurate results.

Similar to Rippel *et al.* (2021), we relied on the EfficientNet-B4 architecture (Tan & Le 2019) to produce a deep representation of the images by applying average pooling (see Chapter 9: Convolutional Neural Networks from Goodfellow *et al.* (2016a) for more details on average pooling) on the output of intermediary convolutional layers of the network.

4. EXPERIMENTS AND RESULTS

4.1. Experimental design

4.1.1. Dataset split

The dataset consists of images coming from 54 cameras. For all of our experiments, the dataset was split by separating the cameras into training, validation, and test cameras.

- The *training set* consists of 46 cameras (67828 images with 9189 'blocked', 36548 'clear', and 22091 'other').
- The *validation set* consists of four cameras (6501 images with 859 'blocked', 3385 'clear', and 2257 'other') chosen randomly, with the constraint that at least 10% of the images should have the blocked label.
- The *test set* consists of four cameras (6123 images with 1543 'blocked', 2870 'clear', and 1710 'other') chosen manually to be representative of different fields of view available within the dataset and to make sure that each test camera had at least 10% of blocked images. The label statistics of the images are presented in Table 1.

The choice to keep relatively small validation and datasets was made to keep a large number of cameras and fields of view in the training set.

4.1.2. Evaluation metrics

The performance of the different methods is reported using two criteria that take into account the imbalance between the number of *clear* and *blocked* images: the balanced accuracy and the receiver operating characteristic (ROC) area under the curve (AUC) criteria.

Table 1 | Statistics regarding the number of labels associated with the images of the test cameras

Location	#Clear	#Blocked	#Other	Total
Crinnis	762	639	412	1,823
Mevagissey	657	541	465	1,663
Barnstaple	669	218	517	1,404
Siston	782	145	306	1,233

The *Balanced Accuracy* criterion is computed as the average of the true positive rates and the true negative rates for a common blockage score threshold allowing to make the difference between images classified as *clear* or *blocked*:

$$Balanced\,accuracy = 0.5 \times \frac{TP}{TP + FN} + 0.5 \times \frac{TN}{TN + FP} \tag{3}$$

where TP is the number of correctly detected *blocked* trash screen images (true positive), FP is the number of *clear* trash screen images incorrectly estimated as *blocked* (false positive), TN is the number of correctly detected *clear* trash screen images, and FN is the number of *blocked* trash screen images incorrectly estimated as *clear* (false negative).

The ROC AUC criterion corresponds to the AUC of the ROC curve. By using multiple score thresholds, the ROC curve plots the true positive rate (TPR) against the false-positive rate (FPR) (Figure 8 for examples).

$$TPR = \frac{TP}{TP + FN},\ FPR = \frac{FP}{TN + FP} \tag{4}$$

The area under the curve can then be computed using a trapezium rule (Süli & Mayers 2003). This criterion ranges from 0 to 1, where a value of 1 indicates a perfect classifier, while a value of 0.5 represents a random binary classifier. More explanations are given in the Supplementary Material (Figure S2).

The balanced accuracy criterion allows us to understand the performance of the models using a blockage score threshold common for all of the test cameras. The ROC AUC criterion uses multiple blockage score thresholds: it allows one to understand how well a model can differentiate between *clear* and *blocked* images over a wide range of blockage thresholds and thus false-positive rates. As these blockage thresholds could be easily set and modified by end users, this score thus provides information less biased towards the optimization of a blockage threshold. Further information regarding these metrics can be found in the study by Hastie *et al.* (2009).

4.2. Comparison of the methods

The goal of this first experiment was to compare the accuracy of the three automated trash screen blockage detection methods, both in terms of balanced accuracy and ROC AUC score, as described in Section 4.1.

4.2.1. Training the networks

We used the same validation methodology for our three approaches. In the first step, we tested different learning parameters to train the model and selected the model obtaining the best ROC AUC averaged over the four validation camera images. In the second step, using the blockage scores generated during the first step, the blockage threshold T was selected to optimize the balanced accuracy results. The performances of models were then assessed on the test set cameras.

For the *binary classification method* implemented with a ResNet-50 network, the parameter values tested on the validation cameras are presented in Table 2. The network was trained using a binary cross-entropy loss function and a stochastic gradient descent optimization algorithm (see Goodfellow *et al.* (2016b) for more details). As indicated in Table 2, we used scheduling to avoid plateaus during training. The network was trained over 50 epochs. At each epoch, the network was tested over the validation set. The network weights returning the highest ROC AUC results over the validation set were chosen.

For the *anomaly detection method* implemented using Rippel *et al.* (2021) approach, the multivariate Gaussian distribution was built using the deep image representations of all the images of the training dataset. Similar to Rippel *et al.* (2021), we

Table 2 | Learning parameters tested during the grid search for the training of the binary classification

Parameter	Values tested
Learning rate	10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Scheduling update factor	0.1, 0.5, 0.9
Scheduling patience	1, 5, 10, 25, 50
Blockage threshold	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9

validated the output of different layers for the deep representation of the image, but the network did not need to be retrained. We then validated the blockage threshold score by considering the 10th, 20th, … , and 90th percentiles of anomaly scores obtained on the validation set images. The training dataset was increased to five times its size using the ImageNet Data Augmentation policy (see Cubuk *et al.* (2019) for more details).

For the *image similarity matching method* implemented with a Siamese network using a ResNet-50 backbone, the network was trained with pairs of images. Each pair of images came from the same camera. They could be either similar (both images are *clear* or both images are *blocked*) or not (one image is *blocked*, the other is *clear*). The network was trained with 50% of similar pairs and 50% of images that were not. The pairs were randomly extracted by first randomly choosing a camera, and then a pair of images in the camera was chosen, to avoid giving more representation to cameras with more images. In total, the network was trained with over 100,000 pairs of images transformed through the ImageNet Data Augmentation policy (Cubuk *et al.* 2019). Similar to the binary classifier, a grid search with tested parameter values presented in Table 2 was performed. The network was also trained over 50 epochs using a binary cross-entropy loss function and a stochastic gradient descent optimizer. At each epoch, the network was tested over the validation set. The network weights returning the best ROC AUC results for the similarity estimation were selected, and the blockage score threshold was validated afterwards. To estimate the blockage score during testing, N reference images and their labels were randomly extracted for each of the test cameras. For our main experiment, we chose a value of $N = 5$. However, in Section 4.3, we will further discuss the impact of N on the performance of the Siamese network.

4.2.2. Analysis of the results

The results of the three different methods on the test cameras are given in Figure 5. The full details of the balanced accuracy results are also provided in the Supplementary Material Table S1. On average, the Siamese network (image similarity matching) obtains the best results with 91% balanced accuracy and 0.98 ROC AUC score. The binary classification network is a close second-best network with 88% balanced accuracy and 0.97 ROC AUC score.

Figure 5 also shows that the anomaly detection approach obtains significantly lower results than the classification and Siamese networks. More specifically, the approach fails at the Mevagissey location where its balanced accuracy is below 50%. We think that the anomaly detection approach has not learned to detect trash as anomalies, especially on fields of view uncommon in the training set, as is the case for Mevagissey. Note that, as we show in the Supplementary Materials (Figure S3), we performed an additional experiment only using *clear* images to train the anomaly detection method, and while its performance improved, it was not able to reach the binary classification and Siamese network performance.

The Siamese network outperforms the binary classification network in three of the four test locations. On average, the Siamese network obtains better results than the binary classifier using both the balanced accuracy criterion and the ROC AUC criterion. However, the gaps in performance between these two networks are smaller using the ROC AUC criterion. As

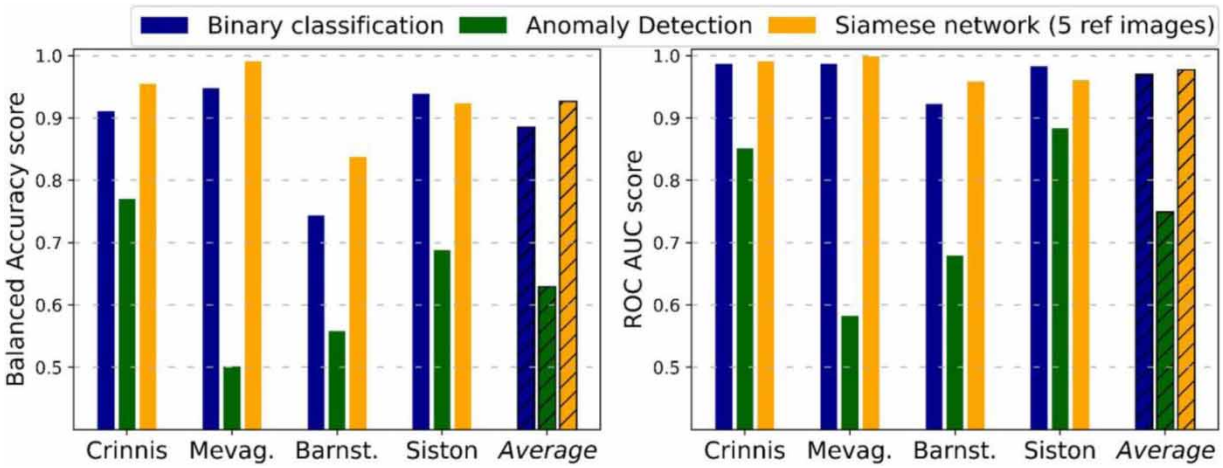


Figure 5 | Comparison between the different methods presented in the article. Left, using the balanced accuracy criterion, and Right, the ROC AUC criterion.

explained in Section 4.1, the balanced accuracy score is obtained by computing an optimal blockage score threshold common for all cameras, while the ROC AUC score considers multiple blockage score thresholds. So, these results show that the classification approach has more difficulties than the Siamese network in providing a common standardized blockage threshold for all cameras. To provide a further understanding of these results, we will discuss the ROC curves and related false alarm rates in Section 4.4.

There is a decrease in performance for both the Siamese and the classification networks at the Barnstaple location. By analyzing the results as provided in Table S1, we noticed that it was due to some images being estimated as blocked with high confidence due to nonblocking debris (e.g., fallen leaves) piling up before the screen. Examples are given in Figure 6.

We could also notice that this problem was present on the other test cameras, but especially at Barnstaple where nonblocking debris tends to accumulate before the screen itself. The problem is more noticeable with the classification network, but this also happens with the Siamese network. Even if the double screen configuration of the site is unique in our dataset, we think the large amounts of false positives are due to the regular presence of large amounts of leaves, which is something that we did not witness on that scale at any other camera locations.

4.3. Influence of the number of reference images on the Siamese network

The goal of this experiment was to test the influence of the number of reference images used to compute the image blockage score. Except for changing the number of reference images, the same protocol as the one presented in Section 4.2 was used. The results are shown in Figure 7.

At Barnstaple, we can observe a progressive increase in the ROC AUC score along with the number of reference images while the balanced accuracy remains stable. We explain this by the fact that the more reference images used, the more clear images containing leaves are used as references. Test images containing leaves are paired with clear reference images

Figure 6 | Samples of images wrongly detected as blocked by the classifier and the Siamese network at the Barnstaple location.

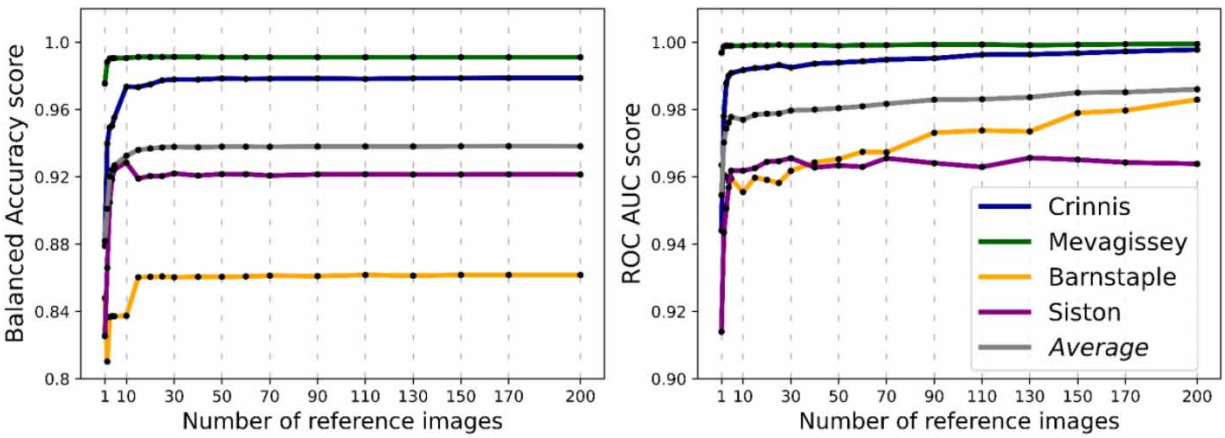


Figure 7 | Influence of the number of reference images on the performance of the Siamese network. Left, using the balanced accuracy criterion, and right, using the ROC AUC criterion.

containing leaves, which allows the blockage score to decrease on these images and thus improve the ROC AUC score. However, given the presence of other reference images, the average blockage score does not reach the blockage threshold found at validation to impact the balanced accuracy score.

We can also observe performance fluctuations at Siston. We explain this as Siston's screen is captured at low resolution with contrast issues related to sun exposure at given times. This makes the labeling by both the network and human observer more complicated and thus generates more mistakes.

The Siamese network needs two labeled images to outperform the classifier with the balanced accuracy criterion (0.9 versus 0.89) and the ROC AUC criterion (0.98 versus 0.97). Increasing the number of reference images also has a more noticeable impact on the ROC AUC results, but the impact on the balanced accuracy seems minimal after 10 labeled images.

It thus seems unnecessary to recommend labeling more than 10 images to assist the Siamese network in its estimations.

4.4. Considerations for end users

For the end user, the false alarm rate is an important consideration. False alarms could trigger unnecessary maintenance visits, wasting time, and the energy cost of a site visit. Furthermore, in scenarios where there is limited time to clear several screens in advance of a heavy rainfall event, false alarms could result in poor prioritization of sites and floods could occur that might otherwise have been prevented. Thus, in Figure 8, we provide the ROC curves associated with the results presented in Section 4.2. As we explained, the ROC curves show the different false-positive (false alarm) rates versus true positive rates trade-offs that are possible to reach by choosing different blockage threshold values. We can, for example, see that with an acceptable false alarm rate of 10% (1 in 10 *clear* image detected as *blocked*, around one false alarm per day when the screen is clean in our case), the Siamese network (five reference images) can detect more than 95% of blocked images at

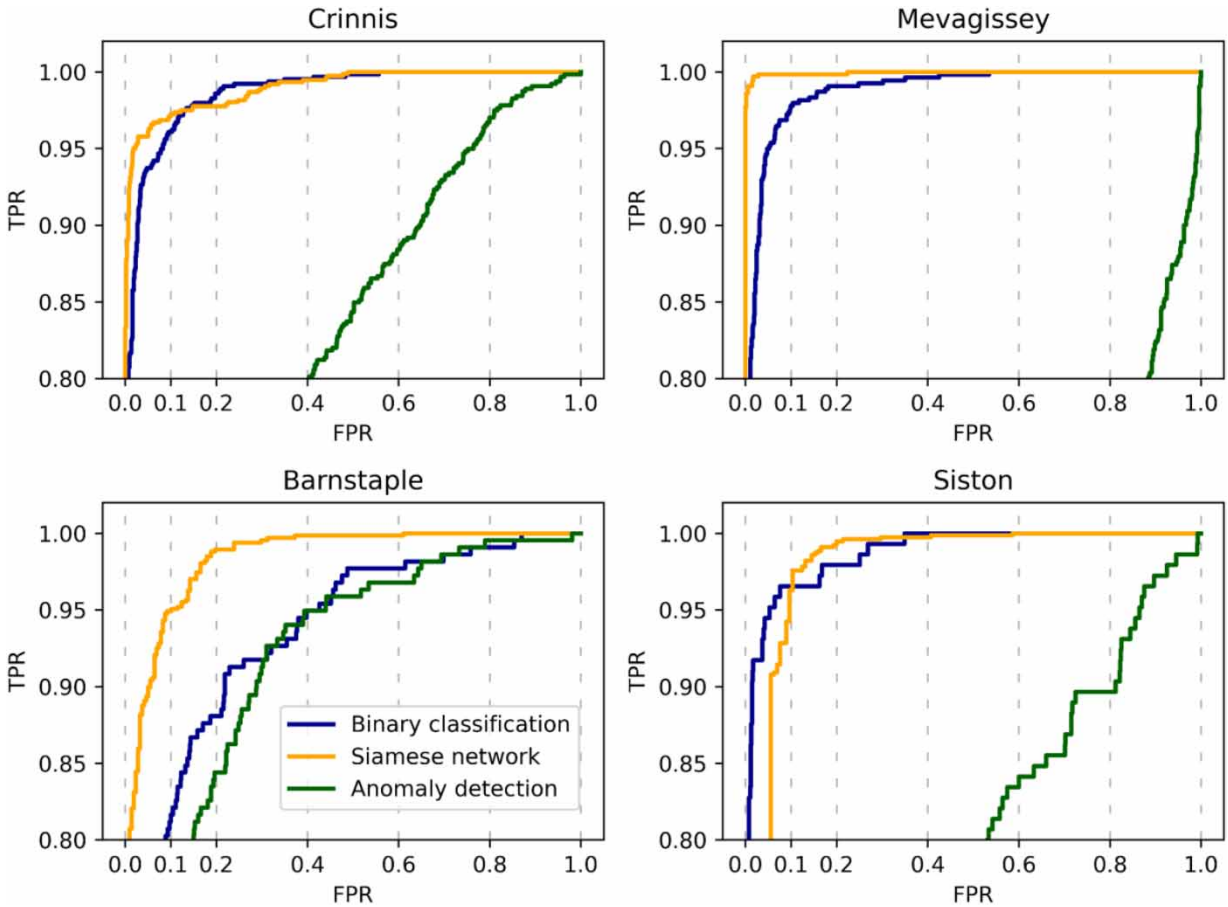


Figure 8 | ROC curves corresponding to the blockage detection scores obtained for each of the methods. See Section 4.1 for more details. Note that for clarity reasons, we limited the plots to show a minimum TPR of 0.8.

every location. The performance of the binary classification network is almost as good as the Siamese network at Crinnis and Siston, but is outperformed at Mevagissey and Barnstaple.

In summary, we saw that both the Siamese network and the binary classification network consistently outperformed the anomaly detection method by large margins, in terms of both balanced accuracy and ROC AUC scores. The Siamese network method outperforms the binary classification method by much smaller margins, but we showed that it is a more interesting method for trade-offs with low false alarm rates.

However, the binary classification method does not require additional intervention, and it is thus the easiest to integrate within a trash screen monitoring infrastructure. Besides, it is also the fastest method (2–3 s per image on an 8×3.4 GHz CPU laptop with no GPU).

At this stage, we cannot recommend the use of the anomaly detection approach.

5. CONCLUSIONS

Trash screens prevent debris from entering critical parts of rivers. However, debris can pile up and block the trash screen to provoke flood events. We investigated the potential of using cameras to automate the detection of blockage. To address the complexity of the challenge (i.e., variety of trash screen types and camera fields of view), and through discussions with practitioners, we decided to investigate deep learning methods for this task.

With the first part of this work, we created a new dataset of trash screen camera images where each image was associated with one of three labels (*blocked, clear*, or *other*). This dataset consists of 80,452 labeled images coming from 54 different cameras and has been made available along with this work (see our Data Availability Statement).

In the second part of this work, we used this dataset to develop three novel deep learning methods to detect the trash screen blockage on new cameras (not used at training). This corresponds to a realistic scenario where end users would want to use the method directly on new trash screen cameras. The first method is a binary classification approach using a ResNet-50 backbone. The second method is an image similarity matching method implemented using a Siamese network that aims at taking advantage of a few labeled images (5–10) for each newly installed camera. The third one is an anomaly detection method that can be trained without needing any labels, thus facilitating the building of a dataset. Note that these networks and their trained weights are also made available along this work so end users can directly reuse these methods (see Data availability statement).

We showed that the binary classifier was able to accurately detect blockage with an average balanced accuracy results of 0.88. The Siamese network was able to improve the results of the binary classifier while only needing a small number of labeled images from the new trash screen camera (5% average improvement in Balanced Accuracy with five annotated images) and is less sensitive to the presence of nonblocking debris. To clarify these results to end users, we also showed that the Siamese network was able to obtain better trade-offs (blockage detection vs false alarms) than the binary classifier, especially at low false alarm rates.

The anomaly detection is significantly outperformed by the other networks. When analyzing the results, we hypothesized that this may be due to an insufficient variety of camera fields of view. However, we cannot evaluate this with the available dataset.

The automated binary trash screen blockage scores that we developed in this article are useful information for practitioners whose job is to monitor and clear trash screens across an area. This information could be integrated into an alerting system, to avoid wasting time manually examining lots of camera images and visiting clear trash screens. The necessity to clear a screen from its trash depends on the amount of trash that has piled up on the screen, as well as the type of screen and local conditions (river level, weather forecast, and so on). For example, it may be urgent to clear a blocked trash screen before an episode of heavy rainfall, but unnecessary during a dry season, and impossible/dangerous if the river levels are high. Thus, this initial alerting system we propose still has a human-in-the-loop to evaluate these wider factors.

However, our future work will focus on the integration of this binary information along with other sources of data (e.g., weather forecasts, rainfall data, and river levels) into a unified platform and smarter alerting system. Furthermore, this information could also be directly used in automated simulation library flood inundation forecasting systems (Hooker *et al.* 2023), to aid in selecting the correct flood maps. There could be a choice of precomputed flood inundation map depending on the trash screen blockage status.

The promising results of this study demonstrate the potential of deep learning methods for the automated detection of trash screen blockage. This represents a flexible, cost-effective, and accurate alternative when compared to more conventional means to maintain trash screens and has the potential to help in the prevention of further losses.

ACKNOWLEDGEMENTS

This work was in part funded by the Environment Agency under the Flood and Coastal Resilience Innovation Programme (FCRIP) and the NERC National Centre for Earth Observation (NCEO). The work has benefited from discussions with members from the FCRIP project *A FAIR (Flood: Aware, Informed, Resilient) approach to community Flood Risk* including local councils from Staffordshire, Stoke, and the Black Country as well as JBA Consulting.

DATA AVAILABILITY STATEMENT

All relevant data are available from an online repository or repositories.

CONFLICT OF INTEREST

The authors declare there is no conflict.

REFERENCES

Benn, J., Hankin, B., Kitchen, A., Lamb, R., van Leeuwen, Z. & Sayers, P. 2019 *Blockage Management Guide*. Available form: https://assets.publishing.service.gov.uk/media/60378f4fd3bf7f03985e1286/Blockage_management_guide_-_report.pdf (accessed 21 November 2023)

Blanc, J. 2013 *An Analysis of the Impact of Trash Screen Design on Debris Related Blockage at Culvert Inlets*. *Doctoral Dissertation*, Heriot-Watt University.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. & Shah, R. 1993 Signature verification using a 'Siamese' time delay neural network. *Advances in Neural Information Processing Systems* **6**.

Chandola, V., Banerjee, A. & Kumar, V. 2009 Anomaly detection: A survey. *ACM Computing Surveys (CSUR)* **41** (3), 1–58.

Chicco, D., 2021 Siamese neural networks: An overview. In: *Artificial Neural Networks. Methods in Molecular Biology*, Vol. 2190 (Cartwright, H., ed.). Humana, New York, NY.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V. & Le, Q. V. 2019 Autoaugment: Learning augmentation strategies from data. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 113–123.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. 2009 Imagenet: A large-scale hierarchical image database. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255.

EA 2023 *EA Web Cams*. Available form: http://www.eathorvertonwebcam.org.uk/Webcammenu/EAFrameset.html (accessed 21 November 2023)

Goodfellow, I., Bengio, Y. & Courville, A. 2016a Convolutional neural networks. In: *Deep Learning*. MIT Press. pp. 335–339.

Goodfellow, I., Bengio, Y. & Courville, A. 2016b Optimization for training deep models. In: *Deep Learning*. MIT Press, pp. 274–279.

Hastie, T., Friedman, J. & Tibshirani, R. 2009 *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer.

He, K., Zhang, X., Ren, S. & Sun, J. 2016 Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Highways England. 2023 *National Highways: Our Latest Traffic Information*. Available form: https://www.trafficengland.com (accessed 21 November 2023).

Hooker, H., Dance, S. L., Mason, D. C., Bevington, J. & Shelton, K. 2023 A multi-system comparison of forecast flooding extent using a scale-selective approach. *Hydrology Research* **54** (10), 1115–1133.

Iqbal, U., Barthelemy, J., Li, W. & Perez, P. 2021 Automating visual blockage classification of culverts with deep learning. *Applied Sciences* **11** (16).

Iqbal, U., Barthelemy, J. & Perez, P. 2022 Prediction of hydraulic blockage at culverts from a single image using deep learning. *Neural Computing and Applications* **34** (23).

Jaikumar, P., Vandaele, R., Ojha, V., 2021 Transfer learning for instance segmentation of waste bottles using mask R-CNN algorithm. In: *Intelligent Systems Design and Applications. ISDA 2020. Advances in Intelligent Systems and Computing* (Abraham, A., Piuri, V., Gandhi, N., Siarry, P., Kaklauskas, A. & Madureira, A., eds). Springer, Cham, Vol. 1351.

LeCun, Y., Bengio, Y. & Hinton, G. 2015 Deep learning. *Nature* **521** (7553), 436–444.

Moy de Vitry, M., Kramer, S., Wegner, J. D. & Leitão, J. P. 2019 Scalable flood level trend monitoring with surveillance cameras using a deep convolutional neural network. *Hydrology and Earth System Sciences* **23** (11), 4621–4634.

Nguyen, T.-T. & Tran, H.-L. 2022 An efficient model for floating trash detection based on YOLOv5s. In: *9th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 230–234.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. & Lerer, A. 2017 *Automatic Differentiation in PyTorch*. Available form: https://pytorch.org/

Rippel, O., Mertens, P. & Merhof, D. 2021 Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In: *25th International Conference on Pattern Recognition (ICPR)*, pp. 6726–6733.

Sabatelli, M., Kestemont, M., Daelemans, W. & Geurts, P. 2018 Deep transfer learning for art classification problems. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*.

Speight, L. J., Cranston, M. D., White, C. J. & Kelly, L. 2021 Operational and emerging capabilities for surface water flood forecasting. *WIREs Water* **8**, e1517.

Streftaris, G., Wallerstein, N., Gibson, G. J. & Arthur, S. 2013 Modeling probability of blockage at culvert trash screens using Bayesian approach. *Journal of Hydraulic Engineering* **476139** (7), 716–726.

Süli, E. & Mayers, D. 2003 Numerical integration – I. In: *An Introduction to Numerical Analysis*. Cambridge University Press, pp. 202–203.

Tan, M. & Le, Q. V. 2019 EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97, pp. 6105–6114.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C., 2018 A survey on deep transfer learning. In: *Artificial Neural Networks and Machine Learning – ICANN 2018. ICANN 2018. Lecture Notes in Computer Science* (Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L. & Maglogiannis, I., eds). Springer, p. 11141

Tharani, M., Amin, A. W., Rasool, F., Maaz, M., Taj, M. & Muhammad, A. 2021 Trash detection on water channels. In *Neural Information Processing, Springer International Publishing*, pp. 379–389.

Vandaele, R., Dance, S. L. & Ojha, V. 2021 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* **25** (8), 4435–4453.

Vandaele, R., Dance, S. L. & Ojha, V. 2023 Calibrated river-level estimation from river cameras using convolutional neural networks. *Environmental Data Science* **2**, e11.

Wallerstein, N. P. & Arthur, S. 2013 A new method for estimating trash screen blockage extent. In: *Proceedings of the Institution of Civil Engineers-Water Management*, Vol. 166, Thomas Telford Ltd., pp. 132–143.

Wallerstein, N., Arthur, S. & Sisinngghi, D. 2010 Towards predicting flood risk associated with debris at structures. In: *Proceedings of the 17th Congress of the Asia and Pacific Division of the International Association for Hydro-Environment Engineering and Research (IAHR-APD)*.

First received 9 January 2024; accepted in revised form 22 March 2024. Available online 1 April 2024