Automated water segmentation and river level detection on camera images using transfer learning

Rémy Vandaele[1,2], Sarah L. Dance[1,3], and Varun Ojha[2]

[1] Department of Meteorology, University of Reading, Reading, U.K
[2] Department of Computer Science, University of Reading, Reading, U.K
[3] Department of Mathematics and Statistics, University of Reading, Reading, U.K
{r.a.vandaele,s.l.dance,v.k.ojha}@reading.ac.uk

Abstract. We investigate a deep transfer learning methodology to perform water segmentation and water level prediction on river camera images. Starting from pre-trained segmentation networks that provided state-of-the-art results on general purpose semantic image segmentation datasets ADE20k and COCO-stuff, we show that we can apply transfer learning methods for semantic water segmentation. Our transfer learning approach improves the current segmentation results of two water segmentation datasets available in the literature. We also investigate the usage of the water segmentation networks in combination with on-site ground surveys to automate the process of water level estimation on river camera images. Our methodology has the potential to impact the study and modelling of flood-related events.

1 Introduction

In recent years, the price and accessibility of surveillance cameras has greatly improved. Notably, this progress has allowed many organizations, private and public, to install surveillance cameras along rivers and other water bodies. The availability of these cameras allows the interested parties to monitor the conditions of the river as well as its surroundings for purposes such as boating, fishing, flood monitoring, etc. [14, 27].

For the flood-risk management sector, the use of such cameras brings an unparalleled opportunity for the study and modelling of flood-related events. Indeed, as of now, to measure the water level of rivers, it is necessary to rely on water gauges [25]. Gauges are expensive to install and maintain, and their measurements can be unreliable when the river goes out-of-bank during a flood. Satellite data from Synthetic Aperture Radar (SAR) can provide information when the river goes out-of-bank, but the frequency of satellite overpasses is limited (currently at most once in each 12 hour period)[17, 5].

River cameras offer a new possibility: by using the measurements of the heights of particular landmarks or objects in the field of view of the camera, or by matching the camera image with light detection and ranging (LIDAR) digital

surface model data [10], it becomes possible to directly estimate the water level from a camera. Such an example is given in Fig. 1. This approach is much more flexible in matter of river surveillance location choices as well as budget.

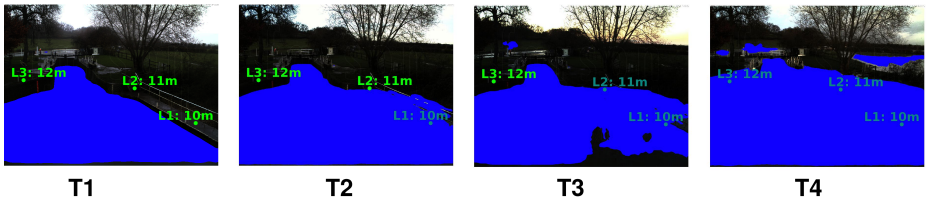

Fig. 1. Sequence of river camera images with annotated landmark heights L1 (10m), L2 (11m) and L3 (12m). At T1, water (segmented in blue) water has not reached any of the landmarks: the water level is below 10m. At T2, L1 is reached by water, but not L2 or L3: the water level is between 10 and 11m. At T3, water has reached L2 but not L3: water level is between 11 and 12m. At T4, water has reached all the landmarks: water level is above 12m.

When considering this approach, the water level measurements must be calculated through a complex workflow: an operator (algorithm or human) has to segment the image to find which areas/landmarks are flooded. Once the operator knows which landmarks were flooded or not, it is possible to estimate the water level: the lower bound will be the height of the highest flooded landmark, and the upper bound will be the height of the lowest not flooded landmark. However, if a human operator is considered, this process makes the water level measurement, time consuming, and possibly an unusable approach since the number of images to study (locations, extent in time, framerate) are typically large.

Our goal is to automate the process of river water segmentation by applying transfer learning (TL) on deep convolutional neural networks, and assess its potential for flood modelling. Specifically, for the datasets at our disposal, we study the relevance of using TL approaches in order to perform water segmentation and possibly use this segmentation to estimate the river levels as accurately as possible. Our paper brings three novel contributions:

1. We develop water segmentation algorithms by using TL, and demonstrate that it outperforms the current methods presented in the literature.
2. We provide an insightful comparison of several TL approaches for water segmentation.
3. We show that it is possible to use our semantic segmentation method in combination with ground survey measurements to estimate water levels for a variety of locations.

In Section 2, we discuss the current related methods that are used to adress the problem of water segmentation on river camera images. In Section 3, we motivate and explain the approach we used to tackle the problem of water segmentation. In Section 4, we show the results of our TL approach. We compare

our method with the current state-of-the-art methods and show that we are able to improve the water segmentation performance. In Section 5, we analyze the efficiency of water segmentation networks to estimate the river levels. We make our final observations and conclusions in Section 6.

2 Related Work

There have been many successful applications of deep learning to images from surveillance cameras. Some examples include deep learning for crowd counting [30, 22], abnormal behavior detection [8], pedestrian detection [26] or even parking occupancy detection [1]. Until now however, most attempts that have tried to tackle the problem of water detection in the context of floods have been realized using hand-crafted features [9]. However, those algorithms remain sensitive to luminosity and water reflection problems [9].

A deep learning approach was applied to flood detection in [16]. The authors perform water detection on a home-made, accessible, dataset of 300 water images that were gathered from the web and annotated manually. The performance of three semantic segmentation networks (FCN-8 [15], Tiramisu [12] and Pix2Pix [11]) are evaluated. By training the networks from scratch, Tiramisu produces the best results, with 90.47% pixel accuracy. It is not clear however if the results are transferable to water level estimation for real cases.

In another work, water detection is performed in the context of autonomous driving for low-cost boats [23]. In this work, a deep learning architecture using a fully convolutional network based on U-Net [20] to perform the water segmentation is proposed. A pixel accuracy of 97.45% is obtained. However, the evaluation protocol used images from the same video streams (therefore very similar images) both for training and test sets, which suggests that the reported results might be overestimated.

In [28], a deep semantic segmentation network is trained from scratch for water segmentation and river level estimation. The biggest originality of this paper lies in their development of the SOFI index (the percentage of segmented water pixels in the image) to evaluate the quality of their results.

A water level estimation model based on voluntary geographic information (VGI), LIDAR data and river camera images is developed in [13]. Notably, random forests are used to develop a waterline detection algorithm [2].

3 Transfer Learning for Water Segmentation

In Section 2, we saw that little research has focused on water segmentation, especially in the context of flooding. Indeed, there are only a few specific water segmentation datasets.

However, semantic segmentation of natural images is an area that has been extensively studied over the past years. State-of-the-art algorithms for multi-purpose semantic segmentation are based on the use of Fully Convolutional Networks (FCNs) [15].

The most well-known datasets used for the comparison of semantic segmentation algorithms are COCO-stuff [3] and ADE20k [31]. These two datasets contain large sets of images semantically annotated with 182 types of labels for COCO-stuff and 150 for ADE20k. As we show in Table 1, some label types of these two datasets correspond to water bodies. These two datasets, among others [6, 7], are widely used for evaluating semantic segmentation algorithms.

Table 1. Water body related images in ADE20k and COCO-stuff datasets.

ADE20k dataset			COCO-stuff dataset		
	Training	Test		Training	Test
water	709	75	river	2113	90
sea	651	57	sea	6598	292
river	320	26	water-other	2453	79
waterfall	80	9			

Given these observations, we decided to tackle the problem of water segmentation using transfer learning (TL).

For a supervised learning problem, the aim is to find a function $f : X \to Y$ from a dataset of N input-output pairs $B = \{(x_i, y_i)_{i=1}^N : x_i \in X, y_i \in Y\}$ such that the function f should be able to predict the output of a new (possibly unseen) input, as accurately as possible. The set X is called the input space, and Y the output space.

With TL, the aim is also to build a function f_t for a *target* problem with input space X_t, output space Y_t and possibly a dataset B_t. However, TL tries to build f_t by *transferring* knowledge from a *source* problem s with input space X_s, output space Y_s and a dataset B_s.

Inductive TL [18] is the branch of TL related to problems where we have datasets of input-output pairs in both source and target domains, and where $X_s = X_t$ and $Y_s \neq Y_t$. Typically, inductive TL is used to repurpose well known, efficient machine learning models trained on large datasets to related problems with smaller training datasets [19, 21].

In our case, we want to use inductive TL where the source problem s will be the segmentation of ADE20K or COCO-stuff images, and the target problem t will be the binary water segmentation of river camera images. We think the problems of segmenting the ADE20K and COCO-stuff datasets are especially relevant in our context given the fact that they contain labels of water bodies, which makes source and target output domains fairly similar.

In the scope of this study, we chose to focus on three TL approaches. With the first TL approach, we use the pre-trained network as such, taking advantadge of the water body labels to directly create binary semantic segmentation masks. With the second approach, we consider model transfer approaches, where we fine-tune semantic segmentation networks pre-trained on either ADE20K or COCO-stuff on water segmentation datasets. We also test a third TL approach related to sample selection, where we fine-tune the pre-trained network on the subset of

ADE20k and COCO-stuff images containing water bodies. While other inductive TL approaches exist and could possibly outperform our current results, we found that our methods are computationally efficient, which will be critical for potential future applications in near-real-time water level estimation.

4 Water Segmentation Experiments

In this section, we discuss the water segmentation experiments that were performed on water segmentation datasets available in the literature. We also compare our results to state-of-the-art water detection results.

4.1 Protocol

Pre-trained networks. The purpose of our experiments is to evaluate the relevance of applying TL for water segmentation. As we explained in Section 3, we chose to consider two datasets for pre-training: ADE20k and COCO-stuff. We chose these datasets as they contain water-labelled images. For each of these two datasets, we study one of its best performing semantic segmentation networks.

For ADE20k, the network we considered is an FCN with a ResNet50 encoder and an UperNet decoder[31]. UperNet [29] is a model that is based on Pyramid Pooling in order to avoid the use of deconvolution layers. During training, the images are rescaled at 5 different sizes: the shorter edge of the image is rescaled to either 300, 375, 450, 525 or 600 pixels, and the bigger edge is rescaled according to the image aspect ratio. We re-used the original implementation as well as the available pre-trained network weights.

For COCO-stuff, we chose the state-of-the-art network referenced by the authors of the dataset, *DeepLab* (v2). It has a ResNet101 encoder, and an atrous spatial pyramid pooling decoder able to robustly segment objects at mutliple scales [4]. We used a pytorch implementation of the model with available pre-trained COCO-stuff weights[4].

First TL approach: pre-trained network use. With this method, we use the pre-trained weights of the networks: we do not tune any layer of the network. We apply the pre-trained networks on our images, and aggregate the predictions of water body labels (lake, river, sea, water, and other similar water related labels) as the output water segmentation. Given that the networks were trained with images of water bodies, this first, simple approach should provide a baseline result for the evaluation of our other approaches. We refer to this approach as **Pre-Trained.**

Second TL approach: networks fine-tuning. As the output of ADE20k semantic segmentation networks is not binary, the last output layers of the semantic segmentation networks could not be directly reused in our binary semantic

[4] https://github.com/kazuto1011/deeplab-pytorch

segmentation problem. This is why we considered three fine-tuning methodologies:

- **HEAD**. With this approach, we only retrain the last output layers of the network. The rationale is that the network has already learned all the necessary filters to perform water segmentation, and it requires only to learn how to perform the binary segmentation.
- **WHOLE.** We fine-tune the entire network, with a random initialization of the last binary output layers.
- **2STEPS**. We first retrain the last layer of the network with all the other layers kept as is. Once the the last layer is retrained, we fine-tune the entire network. This approach can be considered as retraining the entire netwok after having applied the HEAD approach.

Third TL approach: sample selection. As we show in Table 1, the two datasets on which our networks were pre-trained contain images with water related labels. We thus consider a *sample selection* approach algorithm in order to perform TL: we extract all the images containing water labels from the ADE20k and COCO-stuff dataset, and fine-tune the two pre-trained networks on this new dataset with binary masks. In our experiments, we will refer to this approach as **Sample Selection**. We then fine-tuned the network using the WHOLE approach. HEAD and 2STEPS were also tested during our experiments, but for clarity purposes, we chose to only present the results using the approach providing the best results.

Relevance of using TL. In order to understand the relative performance of these TL approaches, we also considered what results could be obtained with the same networks trained from scratch (only using the training images of the dataset). We will refer to this approach as **Scratch**. For the same purpose, we also compared our TL approach with the water semantic segmentation results obtained in the literature [16, 23].

Training. We trained the networks using the parameters recommended by the authors [31, 4]. For the *fine-tuning* and *scratch* approaches, we increased the number of epochs to 300 in order to ensure full convergence for all the networks. For the approaches WHOLE and 2STEPS, we used an initial learning rate value 10 times smaller than its recommended value (0.001) in order to start with less agressive updates.

4.2 Datasets

Our experiments are performed on two datasets used for water segmentation in the literature, and which we use for evaluating the performance of our TL methodology.

- **INTCATCH**, an available dataset of RGB images annotated with binary semantic segmentation water/not-water masks [23]. The images come from a camera positioned on a boat. It was designed for waterline detection for driving autonomous boats. The dataset consists of 144 training images and 39 test images. We noticed that the images in training and test come from two video streams with relatively high frame-rates, which makes training and test datasets look similar.
- **LAGO** (named after the main author[16]), an accessible dataset of RGB images with binary semantic segmentation of water/not-water labelled pixels. The dataset was created through manual collection of camera images having a field-of-view capturing riverbanks. This dataset was used for river segmentation [16]. The dataset is made of 300 images, with 225 used in training, and 75 in test.

Sample images of the datasets are shown in Fig. 2.

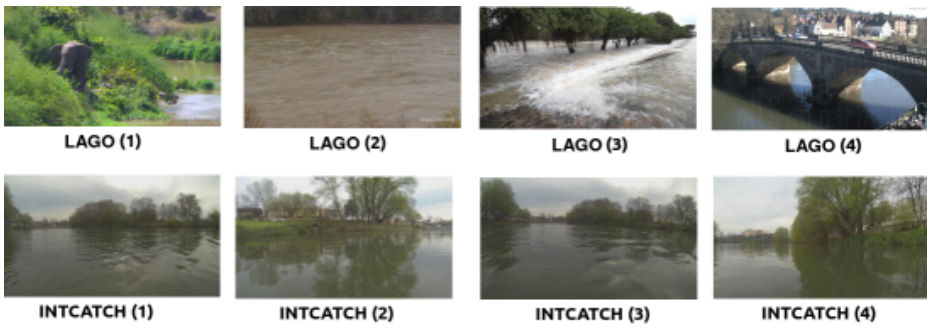

Fig. 2. Sample images from the datasets used for the water segmentation experiments.

4.3 Performance criteria

Let $I \in [0,255]^{H\times W\times 3}$ be a typical 8-bit RGB, image of height H and width W. Let $S \in [0,1]^{H\times W}$ be its corresponding, ground-truth, pixel-wise, semantic segmentation mask, and $\hat{S} \in [0,1]^{H\times W}$ be the estimation (prediction) of this segmentation made by our semantic segmentation algorithm. The two performance criteria used for the evaluation of semantic segmentation methods are defined as follows:

Pixel Accuracy (Acc). In (1), we define the pixel accuracy as the percentage of pixels correctly estimated by our algorithm.

$$Acc = \frac{\sum_{y=1}^{H}\sum_{x=1}^{W} 1 - |(S(y,x) - \hat{S}(y,x))|}{H \times W} \tag{1}$$

Mean Intersection over Union (MIoU). The Intersection over Union (IoU) represents the percentage of overlap between the ground truth and its estimation. The $MIoU$ criteria defined in (2) is the average of the IoU over all the pixel labels. In our case, the pixel label types are water and background (not-water). Thus, we need to consider the binary case:

$$MIoU = \frac{1}{2}\sum_{y=1}^{H}\sum_{x=1}^{W}\frac{S(y,x)\hat{S}(y,x)}{S(y,x)+\hat{S}(y,x)-S(y,x)\hat{S}(y,x)} + \frac{1}{2}\sum_{y=1}^{H}\sum_{x=1}^{W}\frac{(1-S(y,x))(1-\hat{S}(y,x))}{(1-S(y,x))+(1-\hat{S}(y,x))-(1-S(y,x))(1-\hat{S}(y,x))} \tag{2}$$

The advantage of using $MIoU$ over Acc is that it is less sensitive to class imbalance within the image. However, one of the works we are comparing with provide their results for Acc only. For the sake of transparency, we provide our results using both criteria.

4.4 Results and analysis

The results of the water segmentation approaches are presented in Table 2.

Table 2. Results of the water segmentation approaches on LAGO and INTCATCH test datasets.

			LAGO		**INTCATCH**	
			MIoU	*Acc*	*MIoU*	*Acc*
Gonzalez et al.[16]			81.91	90.2	-	-
Steccanella et al.[23]			-	-	-	97.5
ResNet50-UperNet	*Pre-Trained*		90.2	95.4	97.4	98.7
Pre-trained on ADE20k	*Fine-Tuning*	HEAD	89.06	94.37	98.06	99.03
		WHOLE	93.32	96.50	98.94	99.47
		2STEPS	93.09	96.44	99	99.5
	Sample Selection		92.2	96.95	98.95	99.48
	Scratch		83.41	91.74	96.09	98.02
DeepLab	*Pre-Trained*		90.34	95.52	97.70	98.85
Pre-trained on COCO-stuff	*Fine-Tuning*	HEAD	92.19	96.04	99.07	99.54
		WHOLE	93.74	96.76	99.19	99.59
		2STEPS	93.72	96.75	99.16	99.56
	Sample Selection		91.69	96.31	98.59	99.3
	Scratch		80.70	89.95	98.73	99.36

As explained in Section 4.2, we noticed that the images contained in the INTCATCH training and test sets are largely similar to each other as they are

frames randomly sampled from the same two videos. This is how we explain the excellent performance of the different networks tested on these images.

On both LAGO and INTCATCH datasets, we can observe that, for both networks, the pre-trained networks, and TL approaches provide better results than the methods presented in the literature [16, 23]. We can also see that the networks retrained from *scratch* obtain results similar to the ones of the state-of-the-art on the respective datasets. This shows that the use of semantic segmentation networks that are first trained on large multi-purpose datasets can improve the performance. Indeed, even without any kind of fine-tuning, the pre-trained networks already outperform the state-of-the-art.

For the three datasets and both networks, fine-tuning the entire networks (WHOLE and 2STEPS) or using *sample selection* always provides better results than the *pre-trained* networks. This shows that it is possible to further improve the performance of the segmentation by fine-tuning the networks weights. Between *sample selection* and *fine-tuning*, the *fine-tuning* approaches seems to provide the best results.

We also observe that HEAD provides results relatively close to or inferior to the *pre-trained* approach. Furthermore, 2STEPS approach always obtains better results than HEAD. This implies that it is necessary to fine-tune the entire networks rather than retraining only its output layer.

5 River Level Estimation Experiments

In this section, our goal is to describe the experiments that we performed to evaluate whether the semantic segmentation networks assessed in the previous section can be used in the context of river level estimation.

5.1 Datasets

Our river level estimation datasets consist of RGB images coming from video streams of Farson Digital river cameras located at 4 different locations in the U.K. [27]. For each location, the camera position and orientation is fixed, which means that the field of view stays the same for all of the images for a location.

Each location is annotated with landmarks for which heights were manually measured during a ground survey. The images composing the datasets were all sampled from the camera video streams with the purpose of observing a specific flood event. On each sampled image, the landmarks were annotated with binary information flooded/unflooded, that could be used to estimate the water levels in the images (see Fig. 1).

From our first location, Diglis Lock, we extracted 141 images and used 7 landmarks. For the second location, Evesham, we extracted 134 images and used 13 landmarks. For the third location, Strensham Lock, we extracted 144 images and used 24 landmarks. For the fourth location, Tewkesbury Marina, we extracted 144 images and used 4 landmarks. In our nomenclature, **Farson** corresponds to

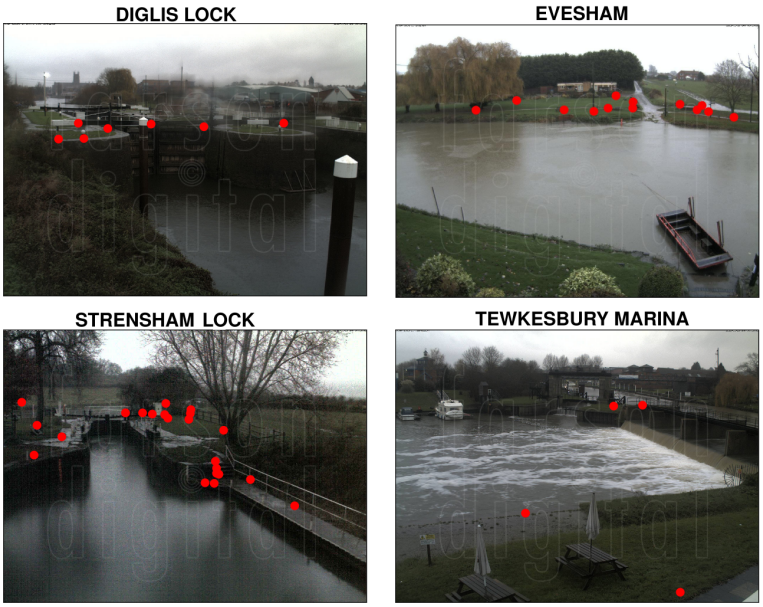

Fig. 3. Images from Farson river camera datasets [27], with their landmarks in red dots.

the union of the images collected from the 4 mentioned locations. Sample images for each of the locations are given in Fig. 3.

5.2 Protocol

Performance criteria. As explained in Section 5.1, only specific landmark (pixel) locations were annotated on those images. This is why we chose to use the balanced accuracy classification score $BAcc$ defined as:

$$BAcc = 100 \times (\frac{1}{2}\frac{TF}{F} + \frac{1}{2}\frac{TU}{U}), \tag{3}$$

where F is the number of actual flooded landmarks, TF the number of correctly classified flooded landmarks, U the number of actual unflooded landmarks and TU the number of correctly classified unflooded landmarks. Given that the extracted time periods of the river camera datasets might create an imbalance between flooded or unflooded landmarks, therefore, we think (3) is a relevant performance criteria to consider.

Experimental design. We reused the networks that were trained in Section 4 to produce binary segmentation masks of the river camera images using the fully trained/fine-tuned networks. A landmark is predicted as flooded if its pixels location is predicted as water, and unflooded otherwise.

A TL approach trying to directly output the water level from the camera images could have been considered. However, this approach requires water-level annotated images for each location as the water levels will vary. Thus, we assess that our landmark classification approach is more relevant.

5.3 Results and analysis.

Table 3. Balanced accuracies (see Section 5.2) of landmark classification using TL on the Farson dataset. Note that Pre-Trained and Sample Selection do not need to be fine-tuned over LAGO or INTCATCH datasets.

			Network trained/fine-tuned on:		
			-	**LAGO**	**INTCATCH**
			BAcc	*BAcc*	*BAcc*
ResNet50-UperNet	*Pre-Trained*		83.4		
Pre-trained on ADE20k	*Fine-Tuning*	HEAD		87.96	78.89
		WHOLE		93.06	88.03
		2STEPS		93.29	88.25
	Sample Selection		90.97		
	Scratch			91.56	80.47
DeepLab	*Pre-Trained*		87.41		
Pre-trained on COCO-stuff	*Fine-Tuning*	HEAD		93.41	91.65
		WHOLE		95.04	94.31
		2STEPS		95.06	93.78
	Sample Selection		91.32		
	Scratch			87.1	85.55

The results are presented in Table 3. The *scratch* approach, which does not use TL, tends to perform worse than the *pre-trained* networks without any kind of fine-tuning. Training the network from scratch on LAGO dataste seems to be the most favorable case. We explain this by the fact that the scratch approach overfits its training dataset, and the LAGO dataset is focusing on river images similar to the Farson dataset.

We can observe that fine-tuning the networks on either LAGO or INTCATCH allows improvement in the landmark classification performance. The WHOLE and 2STEPS approaches that fine-tune the entire networks, obtain the best overall performance. Only retraining the last layer (HEAD) has varying impacts on the performance: while it is always better than retraining the network from scratch, it does not always reach the performance of using the pre-trained network.

The *sample selection* approach provide good performance on both networks. However, when comparing the TL methods, it is always outranked by *fine-tuning* the entire networks (WHOLE and 2STEPS) over LAGO, which is a dataset containing river images. Note that in the context of reusing the semantic segmentation networks for landmark detection over the Farson dataset, the *sample*

selection approach is similar to the WHOLE fine-tuning approaches, the difference being the dataset on which they are fine-tuned.

We can also observe that DeepLab network seems to obtain better results than ResNet50-UperNet overall. From what we have seen on the segmentation results, we believe that the choice of landmark locations played a significant role, and that these results should not be directly correlated to the quality of the segmentation: for example, we observed that while DeepLab seemed to be able to make better distinctions between the edges of the river (where the landmarks are typically located), it was also making more mistakes than ResNet50-UperNet elsewhere in the image (the sky was sometimes considered as water, reflections in the water were not always considered as water). In our case of water segmentation in time series images, several post-segmentation filtering approaches could be considered to improve the landmark detection results: if the N images before and after the current image have segmented landmark X as water/not-water, it is likely that landmark X is also water/not-water in the current image. The information regarding the landmark height could also be used to perform filtering: if N landmarks located at higher locations are segmented as water, it is likely that the lower landmarks should also be segmented as water.

6 Conclusion

In this paper, we have explored the possibilities of using TL in the context of water segmentation, especially for river level detection.

We have shown that TL approaches were able to outperform the current literature in water segmentation on two different datasets. We have also proven that using fine-tuning and/or sample selection could further improve the water segmentation performance. These networks obtained significantly worse performance once retrained from scratch.

We have supplied quantified and encouraging results to demonstrate the utility of our proposed TL approaches in the context of flood modelling, able to predict flood situations with high accuracy.

Future research will focus on an in-depth analysis of our results for practical flood modelling studies, with the aim to provide more advanced statistics helpful to hydrologists, but that are going beyong the scope of this current study [24].

More practically, we would like to consider merging river camera images with LIDAR digital surface model data [10], which can allow to obtain surface elevation of the terrain on a 1m grid. In theory, this could allow our approach to rely on more landmarks for the estimation of water levels, while avoiding the tedious work of performing ground surveys to measure the heights of those landmarks.

Acknowledgements

This work was funded by the UK EPSRC EP/P002331/1. The datasets used in this study are all available as described in references [16, 23, 27].

References

1. Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C., Vairo, C.: Deep learning for decentralized parking lot occupancy detection. Expert Systems with Applications 72, 327–334 (2017)
2. Breiman, L.: Random forests. Machine learning 45(1), 5–32 (2001)
3. Caesar, H., Uijlings, J., Ferrari, V.: Coco-stuff: Thing and stuff classes in context. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1209–1218 (2018)
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence 40(4), 834–848 (2017)
5. Cooper, E.S., Dance, S.L., García-Pintado, J., Nichols, N.K., Smith, P.: Observation operators for assimilation of satellite observations in fluvial inundation forecasting. Hydrology and Earth System Sciences 23, 2541–2559 (2019)
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision 88(2), 303–338 (2010)
8. Fang, Z., Fei, F., Fang, Y., Lee, C., Xiong, N., Shu, L., Chen, S.: Abnormal event detection in crowded scenes based on deep learning. Multimedia Tools and Applications 75(22), 14617–14639 (2016)
9. Filonenko, A., Hernández, D.C., Seo, D., Jo, K.H., et al.: Real-time flood detection for video surveillance. In: IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society. pp. 004082–004085. IEEE (2015)
10. Hirt, C.: Digital Terrain Models, pp. 1–6. Springer International Publishing, Cham (2014)
11. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125–1134 (2017)
12. Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., Bengio, Y.: The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 11–19 (2017)
13. Lin, Y.T., Yang, M.D., Han, J.Y., Su, Y.F., Jang, J.H.: Quantifying flood water levels using image-based volunteered geographic information. Remote Sensing 12(4), 706 (2020)
14. Lo, S.W., Wu, J.H., Lin, F.P., Hsu, C.H.: Visual sensing for urban flood monitoring. Sensors 15(8), 20006–20029 (2015)
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
16. Lopez-Fuentes, L., Rossi, C., Skinnemoen, H.: River segmentation for flood monitoring. In: 2017 IEEE International Conference on Big Data (Big Data). pp. 3746–3749. IEEE (2017)

17. Mason, D.C., Dance, S.L., Vetra-Carvalho, S., Cloke, H.L.: Robust algorithm for detecting floodwater in urban areas using synthetic aperture radar images. Journal of Applied Remote Sensing 12(4), 045011 (2018)
18. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22(10), 1345–1359 (2009)
19. Reyes, A.K., Caicedo, J.C., Camargo, J.E.: Fine-tuning deep convolutional networks for plant recognition. CLEF (Working Notes) 1391, 467–475 (2015)
20. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
21. Sabatelli, M., Kestemont, M., Daelemans, W., Geurts, P.: Deep transfer learning for art classification problems. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 0–0 (2018)
22. Sam, D.B., Surya, S., Babu, R.V.: Switching convolutional neural network for crowd counting. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4031–4039. IEEE (2017)
23. Steccanella, L., Bloisi, D., Blum, J., Farinelli, A.: Deep learning waterline detection for low-cost autonomous boats. In: International Conference on Intelligent Autonomous Systems. pp. 613–625. Springer (2018)
24. Stephens, E., Schumann, G., Bates, P.: Problems with binary pattern measures for flood model evaluation. Hydrological Processes 28(18), 4928–4937 (2014)
25. Tauro, F., Selker, J., Van De Giesen, N., Abrate, T., Uijlenhoet, R., Porfiri, M., Manfreda, S., Caylor, K., Moramarco, T., Benveniste, J., et al.: Measurements and observations in the xxi century (moxxi): innovation and multi-disciplinarity to sense the hydrological cycle. Hydrological sciences journal 63(2), 169–196 (2018)
26. Tian, Y., Luo, P., Wang, X., Tang, X.: Pedestrian detection aided by deep learning semantic tasks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5079–5087 (2015)
27. Vetra-Carvalho, S., Dance, S.L., Mason, D.C., Waller, J.A., Cooper, E.S., Smith, P.J., Tabeart, J.M.: Collection and extraction of water level information from a digital river camera image dataset. Data in Brief 33, 106338 (2020)
28. Moy de Vitry, M., Kramer, S., Wegner, J.D., Leitão, J.P.: Scalable flood level trend monitoring with surveillance cameras using a deep convolutional neural network. Hydrology and Earth System Sciences 23(11), 4621–4634 (2019)
29. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 418–434 (2018)
30. Zhang, C., Li, H., Wang, X., Yang, X.: Cross-scene crowd counting via deep convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 833–841 (2015)
31. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 633–641 (2017)