HESS-2021-20
Vandaele et al. **Deep learning for automated river-level monitoring through river camera images: an approach based on water segmentation and transfer learning**

We thank the reviewer for their useful comments.

Please find our detailed answer here below. The changes to the manuscript are presented in blue font in the answer and the marked up version of the manuscript..

(a) Add some experiments
The necessity of introducing transfer learning into the study has not been fully explored in the paper, which would undermine the scientific significance of your research. It is necessary to set up a control group without using transfer learning method to illustrate the significance of transfer learning. And the comparison between this experiment and the two experiments using transfer learning should be illustrated in table or figure. Through the comparison, the scientific significance of the transfer learning in this task will be further clarified.

We would like to note that the transfer learning aspect of our semantic segmentation approach (usefulness, comparison with state-of-the-art) was studied in a previous work published at a computer vision conference (Vandaele et al., 2020). We have made the following modifications presented in the introduction:

Vandaele et al., 2020 successfully analysed a set of TL approaches for improving the performance of deep water segmentation networks by showing that they could outperform water segmentation networks trained from scratch over the same datasets. This paper builds on the work of Vandaele et al., 2020 and studies the performance of these water segmentation networks trained using TL approaches for the automation of river-level estimation from river-camera images, in the context of flood-related studies. In particular, this work uses water segmentation networks trained using TL approaches in order to carry out novel experiments realised with new river-camera datasets and metadata that consider the use of several methods to extract quantitative water-level observations from the water-segmented river-camera images.

(b) The focus of the introduction
Keeping the description on other types of water level observation and hydrological uses of river cameras in the introduction part is surely no problem, it is just the matter of length. Since the novelty of the work is the modification on the existing deep learning methods rather than to propose a new model to replace the traditional observation methods, it is still recommended that you emphasize more on the existing computer vision methods for water segmentation. From line 58 to line 65, you can add more reference including research on histogram analysis and machine learning methods for water segmentation. The difference and commonality of these methods are also worth introducing. These computer vision methods are the basis of your work, they deserve more description.

We have made following changes to the paragraph from line 58 to 65:

By extracting the location of water-filled pixels from a stream of river camera images (water segmentation), it becomes possible to analyse flood events happening within the field-of-view of a camera. Most attempts that have tried to tackle the problem of automated water detection in the context of floods have been realised through the histogram analysis of the image (Filonenko et al., 2015; Zhou et al., 2020). These algorithms remain sensitive to luminosity and water reflection problems (Filonenko et al., 2015) unless the dynamic aspect of the video feed can be exploited (e.g., 25fps in Mettes et al., 2014) or the camera is set to observe a specific gauge/ruler (Pan et al., 2018), which is not the case for the river cameras used in this work (1 frame per hour). Deep learning approaches have been applied to flood detection using river cameras (Lopez-Fuentes et al., 2017; Moy de Vitry et al., 2019). However, current flood-related studies using river camera images are limited because the observations made on the stream of images must be annotated manually (Vetra-Carvalho et al., 2020b). An accurate, manual annotation of such images is a long and tedious process that compels the analyst to narrow the scope (number of images considered) of the study.

Newly added references:

Zhou, S., Kan, P., Silbernagel, J., & Jin, J. (2020). Application of image segmentation in surface water extraction of freshwater lakes using radar data. *ISPRS International Journal of Geo-Information*, *9*(7), 424. **https://doi.org/10.3390/ijgi9070424**

Mettes, P., Tan, R. T., & Veltkamp, R. (2014, January). On the segmentation and classification of water in videos. In *2014 International Conference on Computer Vision Theory and Applications (VISAPP)* (Vol. 1, pp. 283-292). IEEE. **https://doi.org/10.13140/2.1.2141.2809**

Pan, J., Yin, Y., Xiong, J., Luo, W., Gui, G., & Sari, H. (2018). Deep learning-based unmanned surveillance systems for observing water levels. *IEEE Access*, *6*, 73561-73571. **https://doi.org/10.1109/ACCESS.2018.2883702**

(c) The details about the CNN model
A CNN based model is typically composed of three types of layers, which are convolutional layer, pooling layer and fully-connected layer. However, in Section 2.1.2, only the computational process in convolutional layer is introduced. Are the other two layers used in the model in your work? If so, please supplement the introduction of them, and revise Figure 2 according to the added introduction. This will also help readers to understand the main difference between the two transfer learning strategies in Section 2.2.3 and the scientific significance of the work on comparing different strategies. Additionally, it is also necessary to simply mention the activation function between different layers in CNN model, e.g., ReLu or Softmax.

As explained in L85, our goal with the brief explanation of the CNN was to present the concepts of the CNN to a community that is not oriented towards deep learning so that anyone could read the paper without referring to other work. The aim is not to describe the architectures that we used (Resnet50-Upernet and Deeplab). However, we agree that some details could help the reader's understanding, so we have made the following changes:

Paragraph starting line 97:
As for most image-processing related tasks, recent advances in optimisation, parallel computing and dataset availability have allowed deep learning methods, and specifically deep convolutional neural networks (CNNs) to bring major improvements to the field of

automated semantic segmentation (Guo et al., 2018). CNNs are a type of neural network where input images are processed through convolution layers. As it is shown in Fig. 2, with convolutional neural networks, an image is divided into square sub-regions (tiles) of size F × F that can possibly overlap. The image is processed through a series of convolutional layers. A convolutional layer is composed of filters (matrices) of size F × F × C_i, where C is the number of channels of the input image at layer i . For each filter of the convolutional layer, the filter is applied on each of the tiles of the image by computing the sum of the Hadamard product (element-wise matrix multiplication) - also called a convolution in deep learning - between the tile and the filter (Strang, 2019), which is then processed through an activation function (e.g. ReLU (Nair and Hinton, 2010), sigmoid or identity function). If the products of the convolution operations are organised spatially, the output of a convolutional layer can be seen as another image which itself can be processed by another convolutional layer: *if a convolutional layer is composed of N filters, then the output "image" of this convolutional layers has N channels. CNN architectures vary in number of layers and choice of activation function, but also in terms of additional layers. Typically, SoftMax layers are added at the end of categorization/classification tasks (such as semantic water segmentation) to normalize the last C_l channels into a probability distribution of C_i category/classes. Pooling layers are often used to reduce the dimension of a layer by computing the maximum (max-pooling)/average (average-pooling) of partitions (non-overlapping contiguous regions) of size P x P of the input image.*

During the training of the networks, the weights of the filters (the matrix values) are optimised. The idea is that the filters will converge along the convolutional layers towards weights making the input image more and more meaningful for the task at hand.

Newly added references:

Vinod Nair et Geoffrey E. Hinton, « Rectified linear units improve restricted boltzmann machines », *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010

Paragraph starting line 163:

The semantic segmentation networks that were chosen are addressing semantic segmentation problems with 171 (COCO-stuff) and 150 (ADE20k) labels (see Section 2.2.1) and use a SoftMax layer (see Section 2.1.2) to perform their segmentation, which means that their last layer has as many filter as there are labels. However, the water semantic segmentation problem is a binary segmentation problem, with only two labels: water or not-water. In practice, this means that the last layer of the source semantic segmentation networks and the target semantic segmentation networks will have a different number of filters. In consequence, it is not possible to use the weights of the last layer of the source network to initialise the weights of the last layer of the target network. This is why two fine-tuning strategies were considered in Vandaele et al., 2020:

(d) The setup of the experiments

In Section 3, some key information about model training should be supplemented. Firstly, the setup of hyperparameters is supposed to be added, including learning rate, training epoch, batch size as well as the optimizer for training. Simultaneously, the introduction of learning rate setting would help to understand the meaning of fine-tuning if compared with the learning rate in original training with large dataset. Secondly, the machine learning library used in this study needs to be illustrated,

Pytorch, Tensorflow or Keras? Thirdly, which loss function is used for supervising the training should be illustrated, cross entropy loss, MSE loss or others? Fourthly, could you please give some comparison of two training strategies on convergence speed?

We propose to change the paragraph L179-181 to the following:

As explained in Vandaele et al. 2020, the training used 300 epochs in order to ensure full convergence for all the networks. The initial learning rate value for the fine-tuning was 10 times smaller than its recommended value (0.001) in order to start with less aggressive updates. The other parameters (loss, update schedule, batch size) were chosen as recommended by the authors of the networks (Zhou et al., 2018; Chen et al., 2017). Both authors implemented their network using the Pytorch library.