Comparison of YOLOv4-tiny and YOLOv7-tiny for instream wood detection

Marc O'Callaghan – University of Lausanne, Faculty of Geosciences and Environment

Abstract

The presence of instream wood has strong implications for river ecology, morphology, and hazard assessment. Detecting it manually is a tedious, if not impossible task, however, due to the large amount of data to process. This work presents an attempt to automate the process with two different fully convolutional neural networks YOLOv4-tiny and YOLOv7-tiny.

1. Introduction

The impact of instream wood on river ecology and geomorphogenesis has been under increasing study in recent years (e.g. Wilcox & Wohl, 2006; Ruiz-Villanueva *et al.*, 2018; Friedrich *et al.*, 2022). Logs transported by the stream may mechanically exacerbate channel widening and damage infrastructures such as bridges and levees (Ruiz-Villanueva *et al.*, 2018; Wohl *et al.*, 2016). Wood accumulations, on the other hand, may obstruct the stream and thus increase habitat diversity in the river ecosystem (Wohl *et al.*, 2019). However, this may also cause overbank flooding or increase downstream scour (Friedrich *et al.*, 2022; Wohl *et al.*, 2016). Fig. 1 summarises the effects of instream wood at river segment scale.

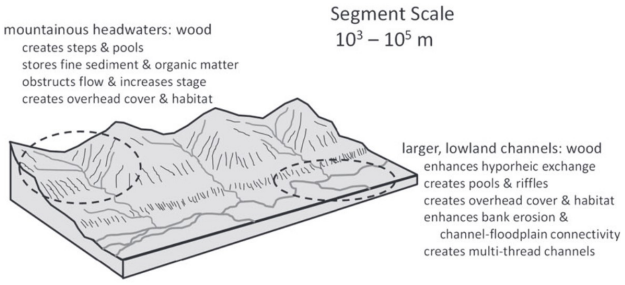

Figure 1. Summary of the effects of wood of a river ecosystem by Wohl *et al.* (2016:318).

It follows that monitoring wood transport in rivers is essential to assess both ecosystem health and hazardousness. Owing to the large quantity of data to process, and to the difficulty to access certain study sites, tracking instream wood manually is impractical, if at all possible. Aarnink *et al.* (2022) have proposed a machine learning approach to automate this process.

The aim of this work is to implement two fully convolutional neural networks (FCN) for this purpose, as they are recognised to be well suited to object detection (Géron, 2019). A comparative approach aims to assess the performance of both models as a starting point of future developments of an algorithm best suited to instream wood detection.

The *You Only Look Once* (YOLO) algorithm series (launched by Redmon *et al.*, 2016) was chosen in view of its speed of execution. We used YOLOv4-tiny (Bochkovskiy *et al.*, 2020) due to the availability of extensive documentation on its implementation (e.g. TheAIGuy's YouTube tutorial (2020-06-29) and assorted GitHub repository and Alexey Bochkovskiy's GitHub repository (Bochkovskiy *et al.*, 2020). The second FCN we chose was YOLOv7-tiny (Wang *et al.*, 2022), which can be implemented through Bochkovskiy's `darknet` executable as well, but was originally made available on Wong Kin Yu's GitHub repository. The *tiny* versions were used for reasons of computational efficiency, as they are considerably shallower and therefore faster to train than the full-scale YOLOv4 and YOLOv7. Wang *et al.* (2022) indicate that YOLOv7 performs significantly better than previous YOLO installments on real-time detection in the MS COCO dataset. The present work studies whether this also applies to the respective *tiny* versions, and whether one of them can be used reliably for instream wood detection.

The implementation focused on detecting one class `singlewood`, which describes single logs of large wood (i.e. wider than 10 cm and longer than 1 m, as defined by Wohl *et al.*, 2010). Its aim is to return a bounding box around the objects of interest in each image.

2. Data and methods

2.1. Images

For this work, we used a subset of images provided by Janbert Aarnink (University of Lausanne, Institute of Earth Surface Dynamics), which originally comprised 5'429 labelled images. These were taken from videos shot with smartphones above or beside a stream into which logs had been cast manually, as no instream wood was spontaneously present at the time of filming. The set of 2'000 still pictures

Figure 2. Example of a horizontally flipped image used in the augmented training dataset. Note the log at the centre left.

taken from these videos was augmented by random flipping, rotating, and colour alteration (cf. Fig. 2).
In order to reduce computation time in the scope of this project, we restricted the set to 45 labelled images from the augmented dataset. This was further split into a training and test set with `sklearn.model-selection.train_test_split` using `test_size = 0.2` and `random_state = 12`. A test size of 20% of the total set was chosen in alignment with Géron (2019). This yielded a training set containing 36 elements and a test set containing 9.

2.2. Algorithms

YOLOv4 (Bochkovskiy *et al.*, 2020) and YOLOv7 (Wang *et al.*, 2022) both consist of a sequence of backbone, neck and head components (Fig. 3). The exact description of these components' setup lies beyond the scope of this work, but it must be mentioned that they are both one-stage detectors using dense prediction in the head.

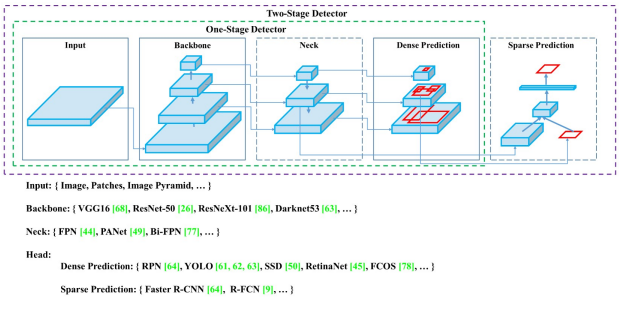

Figure 3. Architecture of YOLOv4 (Bochkovskiy *et al.*, 2020). Although the individual components differ, YOLOv7 follows the same global architecture (Wang *et al.*, 2022).

The CFG files provided by Bochkovskiy consist of 162 layers for YOLOv4 while YOLOv7 consists of 143. The *tiny* versions consist of 38 and 99 layers, respectively (cf. Bochkovskiy's yolov4-tiny-custom.cfg and yolov7-tiny.cfg files on GitHub [retrieved 2022-12-14]).

Batch size	64
Subdivisions	16
Max iterations	6000
Steps low	4800
Steps high	5400
Network size	416x416
Classes	1
Filters	18

Table 1. Hyperparameters used for both algorithms (only divergences from the default are shown). The `filters` were only changed in the convolutional layers directly preceding the YOLO layers.

As they are reduced versions of the full-scale YOLOv4 and YOLOv7 and not standalone algorithms, there is no documentation focusing on their comparative performance nor on their recommended setup. For this reason, we chose to use the hyperparameters recommended by Bochkovskiy in the readme of his GitHub repository (Tab. 1 [retrieved 2022-12-14]). It is indicated that the `steps` should correspond to the maximum number of iterations $\pm$ 10 % and the filters of a convolutional layer directly preceding a YOLO layer should be set to 5 $\times$ classes + 3, i.e. 18 in the case at hand.

3. Results and discussion

Fornari's (2022) findings in a previous project with the same scope, but using full-scale YOLOv4, reached a mean average precision (mAP; Hendry & Chen, 2019) of 80.88% at IoU $>$ 0.5. They used a dataset consisting of 12 training images and 4 test images, because a larger dataset proved unreasonably long to process.
In our case, YOLOv4-tiny achieved a best performance of 78 % mAP, which later decreased and plateaued at 67 %. Processing was terminated at 3505 iterations as no significant improvement had occured over the last approx. 1000 iterations (Fig. 4). YOLOv7-tiny, on the other hand, performed considerably better on the training set, achieving up to 91% mAP (Fig. 4).
Training and testing on the 45-image set took approx. 3 h in both cases, a considerable improvement from Fornari's (2022) attempt. After training, we attempted to run both algorithms on four other images from separate datasets, also provided by Aarnink. Here, it became clear that the performance, though not quantified, was much less satisfactory (Fig. **??**). Although both algorithms performed poorly on these new inputs, it appears that YOLOv7-tiny was slightly more adaptable than YOLOv4-tiny (it successfully detected logs that YOLOv4-tiny missed in all but one of the four images). On the fourth image, neither algorithm detected instream wood at all.
The training and testing set were both selected from the same dataset and were temporally contiguous, which presumably

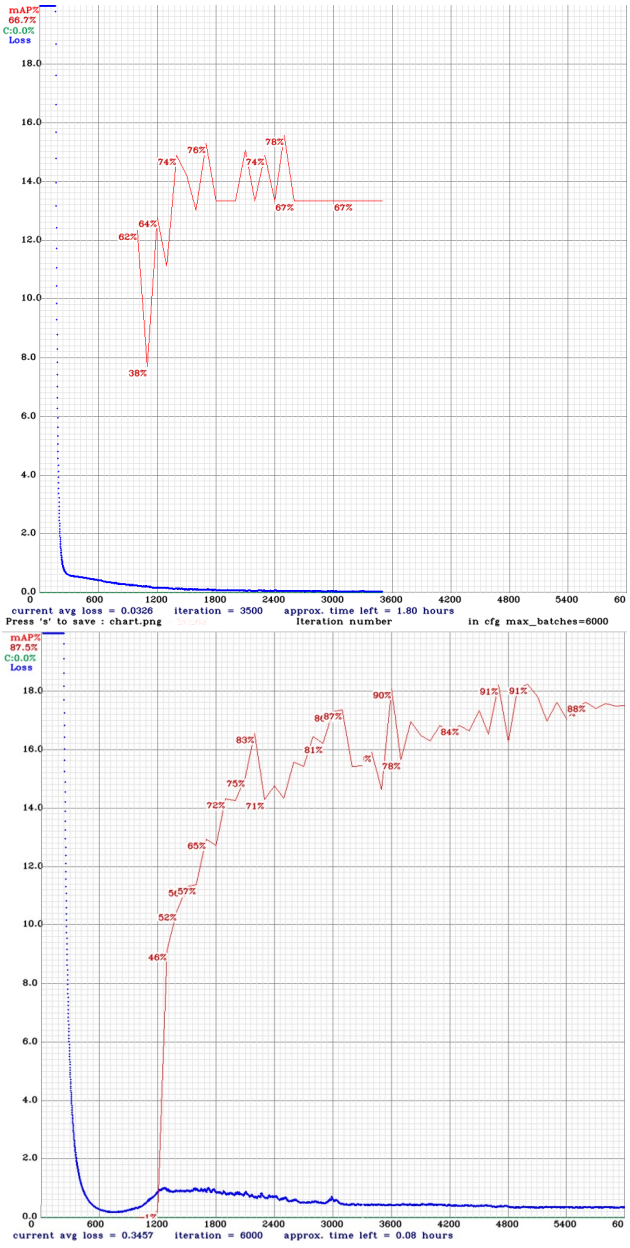

Figure 4. mAP and test loss of YOLOv4-tiny (*top*) and YOLOv7-tiny (*bottom*).

Figure 5. One of the manual test images, as processed by YOLOv7-tiny. YOLOv4-tiny detected no logs.

resulted in both models overfitting and only recognising those logs which were visible in that sequence. Although YOLOv7-tiny proved slightly more transferable, it identified logs in the manual test with considerably less confidence (46% in image 4).

A more careful selection of training data among the images provided by Aarnink, would presumably have resulted in more transferable learning. Owing to the limited available time, as well as the limited resources offered by Google Colab, however, it proved impossible to do so.

The implementations proposed by Bochkovskiy in his GitHub repository are unfortunately not comprehensively documented. As such, it is not sure to what extent human errors may have been introduced in our use of these algorithms due to misunderstandings on our part. The fact that they are designed for use on Google Colab is also problematic due to the limited resources this platform offers. A portable version of their code, which can be implemented on local IDEs, would be advantageous. They do, however, offer a relatively straightforward implementation of complex algorithms which researchers who are not data scientists would be incapable of designing.

4. Conclusion

We have shown that the *tiny* versions of YOLOv4 and YOLOv7 can be trained in considerably less time than their full-scale counterparts. This can be of interest when rapid implementation of these algorithms is needed. YOLOv7-tiny, in particular, reached high accuracies in the training and test set and was able to recognise wood in other datasets as well, despite certainly overfitting the training data. We have also shown that ready-made implementations, as found on GitHub, can be used in a realtively easy way to imple-

ment complex algorithms for research.
A more thorough evaluation of the performance of both algorithms should be performed by selecting a more diverse training set. Within the limited time frame and resources available, however, our results seem to point that YOLOv7-tiny has greater potential for instream wood detection than YOLOv4-tiny.

5. Code and data availability

The final version of our code and data is available at https://github.com/ocallaghanm/2022_ML_EES/blob/main/OCallaghanM_project.ipynb. Indications to access the data used in this work are included in the same file.

Bibliography

Aarnink, J., Ruiz-Villanueva, V., & Vuaridel, M. (2022). Machine learning and RFID-based large wood tracking in rivers. https://doi.org/10.5194/egusphere-egu22-3974

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection (arXiv:2004.10934v1). arXiv preprint.

Fornari, A. (2022). *Instream Large Wood detection trough YOLOV4*. Machine Learning for Earth and Environmental Sciences course, SS 2022. University of Lausanne, FGSE.

Friedrich, H., Ravazzolo, D., Ruiz-Villanueva, V., Schalko, I., Spreitzer, G., Tunnicliffe, J., & Weitbrecht, V. (2022). Physical modelling of large wood (LW) processes relevant for river management: Perspectives from New Zealand and Switzerland. *Earth Surface Processes and Landforms*, 47(1), 32–57. https://doi.org/10.1002/esp.5181

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow (2nd ed.)*. O'Reilly.

Hendry & Chen, R.-C. (2019). Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning. *Image and Vision Computing*, 87, 47–56. https://doi.org/10.1016/j.imavis.2019.04.007

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016, May 9). You Only Look Once: Unified, Real-Time Object Detection. IEEE Conference on Computer Vision and Pattern Recognition. http://arxiv.org/abs/1506.02640

Ruiz-Villanueva, V., Badoux, A., Rickenmann, D., Böckli, M., Schläfli, S., Steeb, N., Stoffel, M., & Rickli, C. (2018). Impacts of a large flood along a mountain river basin: The importance of channel widening and estimating the large wood budget in the upper Emme River (Switzerland). *Earth Surface Dynamics*, 6(4), 1115–1137. https://doi.org/10.5194/esurf-6-1115-2018

Ruiz-Villanueva, V., Mazzorana, B., Bladé, E., Bürkli, L., Iribarren-Anacona, P., Mao, L., Nakamura, F., Ravazzolo, D., Rickenmann, D., Sanz-Ramos, M., Stoffel, M., & Wohl, E. (2019). Characterization of wood-laden flows in rivers. *Earth Surface Processes and Landforms,* 44(9), 1694–1709. https://doi.org/10.1002/esp.4603

Wilcox, A. C., & Wohl, E. E. (2006). Flow resistance dynamics in step-pool stream channels: 1. Large woody debris and controls on total resistance. *Water Resources Research*, 42(5). https://doi.org/10.1029/2005WR004277

Wohl, E., Bledsoe, B. P., Fausch, K. D., Kramer, N., Bestgen, K. R., & Gooseff, M. N. (2016). Management of Large Wood in Streams: An Overview and Proposed Framework for Hazard Evaluation. *JAWRA Journal of the American Water Resources Association*, 52(2), 315–335. https://doi.org/10.1111/1752-1688.12388

Wohl, E., Kramer, N., Ruiz-Villanueva, V., Scott, D. N., Comiti, F., Gurnell, A. M., Piegay, H., Lininger, K. B., Jaeger, K. L., Walters, D. M., & Fausch, K. D. (2019). The Natural Wood Regime in Rivers. *BioScience*, 69(4), 259–273. https://doi.org/10.1093/biosci/biz013