Exploring semantic segmentation in rowing images

S. E. Berendse
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
s.e.berendse@student.utwente.nl

ABSTRACT

This study is an exploratory work into semantic segmentation of rowing images. Rowing is a highly technical sport, which is very suitable for automated analysis. However, not many systems are available for this yet, with the ones that are available using inertial sensors. Being able to analyse (old) rowing footage could help coaches further improve their crew's technique. This study aims to take a first step towards visual automated analysis of the rowing stroke. In this paper, we retrained a pre-trained Deeplabv3+ model to segment rowers and their boats. The performance of the model was evaluated similarly to Microsoft's COCO challenge, with the primary metric being the mean intersection over union and pitted against the performance of the pre-trained model. The results show an increase in performance of 14.5% in the primary metric when using the retrained model, even though a very limited amount of training was done. These results show that there is potential in using machine learning to create an automated video analysis system for application in rowing.

Keywords

Rowing, semantic segmentation, transfer learning

1. INTRODUCTION

Analysing sports is quickly gaining traction, both in tracking performance with apps such as Strava, as well as in tracking technical aspects of (competitive) sports. Most sports events are well-covered by video, but training sessions are also filmed increasingly often. This leaves a wealth of visual data that can be used to gain insight into the movement of a sport. Being able to analyse old footage to gain a better understanding of what made a great crew so dominant in their heyday might also be beneficial to advance technique for current day athletes. Rowing is a sport that is highly dependent on a combination of technique, endurance, and power. Both power and endurance can be measured fairly easily, for example by making use of an indoor rowing machine such as the Concept2 indoor rower. Technique, however, is more complicated to measure due to the differences in what movement is most efficient for indoor rowing compared to rowing in a boat. For this reason, it is interesting to take an exploratory step in automated image or video processing, with the final goal being analysing movement patterns in rowing, to help support rowing coaches in improving their crew's technique.
The current state-of-the-art in rowing video analysis is without any use of machine learning [14]. The system is fully based on mathematical equations and estimating positions of rower body parts based on the previous video frame. Due to this, it is very limited in its use, as it requires very specific conditions under which the video was shot. This makes such a system very inflexible in its use, hence why it is relevant to propose a system that functions under more natural circumstances.
In this paper, we explore a first step in machine learning for automated video analysis of rowing footage. Our research question for this is as follows: *Can machine learning be used to accurately perform semantic segmentation on rowers and boats in images?* To answer this, we adapt a machine learning system to semantically segment rowers and their boats in images. This is done by retraining a pre-trained state-of-the-art visual detection architecture named Deeplabv3+ [9] using transfer learning. The retrained model used focal loss [12] as the loss function. The dataset which we used in this paper consists of 100 unique images taken from a Stanford dataset [11] and a database of rowing images taken by the Photo Committee of D.R.V. Euros [2]. All images were labelled and then split in training, validation, and test sets, in 75%/5%/20% partitions respectively. To evaluate performance, the same metrics that are used in Microsoft's COCO Stuff Challenge were used. The system was tested with and without post-processing, to determine which version of the system would be compared to the pre-trained model. Minor post-processing by introducing a certainty threshold for the predictions turned out to work better for both models. Results showed a 14.5% improvement in performance over the pre-trained model for the primary evaluation metric when using the retrained model, as well as improvements in all other metrics.
In short, during this research the following was achieved:

- A dataset was gathered and labelled
- A script to pre-process this labelled dataset was implemented
- An existing Deeplabv3+ implementation was adapted to suit the use case of this research
- A Deeplabv3+ model was trained that achieved a 14.5% performance increase over the pre-trained model for the primary evaluation metric.

The rest of this paper is organised as follows: in Section 2, related work and scientific background of this topic are discussed. In Section 3, we outline our research question. In

33^{rd} Twente Student Conference on IT July. 3^{rd}, 2020, Enschede, The Netherlands.

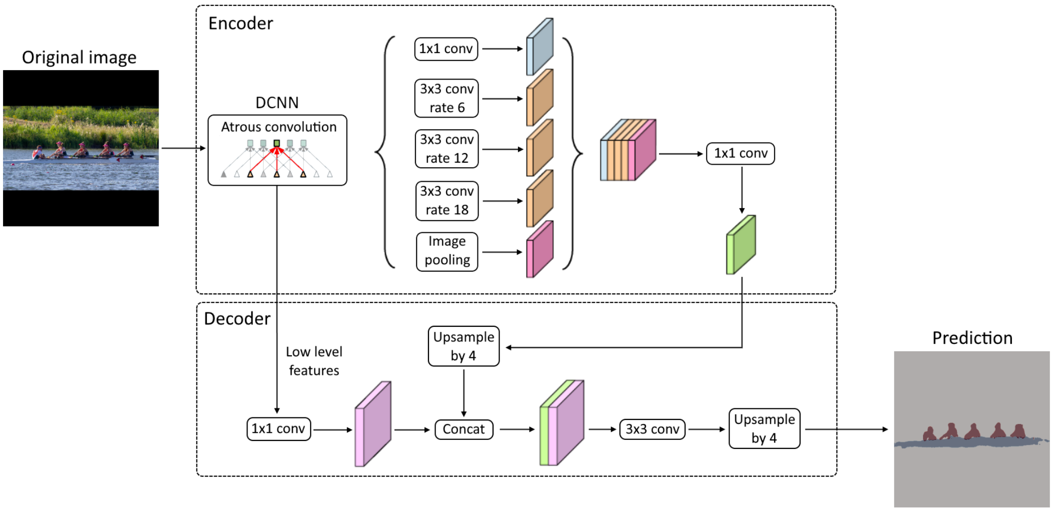

Figure 1. A visualisation of the encoder-decoder architecture used by Deeplabv3+.

Section 4, an in depth explanation of the research methodology is given. In Section 5 we present our results, along with an explanation of our findings based on these results. Furthermore, in Section 6, the limitations and recommendations that follow this research are described. Finally in Section 7, we conclude on our research question.

2. RELATED WORK

2.1 Rowing analysis

As mentioned earlier, sports analysis is becoming widespread nowadays. This field of study is not very old, the first research started in the late 20th century [4]. Biomechanics are highly significant for rowing, as it is a highly technical sport to which many biomechanical concepts apply, with a movement which can be modelled mathematically due to the repetitive, restricted motion patterns [10]. Currently, there exist two types of systems available for analysis of rowing technique. The first uses sensors, the second uses video. Sensor-based systems often use a variety of sensors, such as accelerometers, GPS and force sensors [16]. Systems that utilise sensors, however, will always require extra hardware to be mounted on the boat or oars, or the rowers themselves. They also can only provide data on sessions during which the hardware was mounted.
The second type, video-based systems, are more popular. Filming has become common practice among coaches [15]. The reason is that smartphones nowadays are cheap, yet effective for allowing athletes to review their movements at a later time, from a different perspective. Currently, there is a need for a video processing tool that caters specifically to the needs of rowing coaches.

Aside from the sensor versus video-based systems, there is also a distinction between systems that provide direct feedback and those that allow for post-session evaluation. An example of a direct feedback system would be Sofirow, a system that can give acoustic feedback based on various metrics [13].

The current state-of-the-art in visual motion detection appears to be a method to estimate body positions while rowing, proposed by G. Szűcs and B. Tamás [14]. Their method could extract the position of the head, shoulder, elbow, wrist, hip, knee, and ankle. All of these anchor points are relevant for evaluating rowing technique. The system turned out to be highly accurate, but also strongly dependent on the quality of the video and the circumstances in the video, such as lighting, background, and shadows. The research did not use any machine learning, which could explain why the background subtraction required a rather complex system already.

2.2 Visual object detection

Visual object detection, and more specifically semantic segmentation can be done using a variety of methods. The architecture that was chosen for this research is Deeplabv3+. Deeplabv3+ is a state-of-the-art architecture for semantic segmentation and is the latest version in the Deeplab series of detectors. It is an improvement upon Deeplabv3, which in turn superseded Deeplabv2 and Deeplabv1.

Deeplabv1 was introduced in 2015 to combat the problem existing Deep Convolutional Neural Networks (DCNNs) had in the final layer with localising responses well enough for accurate segmentation [6]. Over a year later, a second iteration was proposed. Deeplabv2 made us of a new technology called Atrous Spatial Pyramid Pooling (ASPP), on top of the Atrous Convolution and Conditional Random Field (CRF) that were carried over from v1 [7].

For Deeplabv3, the entire structure of the system was rethought. The system no longer made use of CRF as a post-processing step, but improved on the ASPP module by using batch normalisation and image-level features. Aside from this, modules that use Atrous Convolution in cascade or parallel to handle multiple-scale segmentation of objects were implemented. The system achieved similar performance to other state-of-the-art models due to these improvements over the previous versions [8].

Finally, Deeplabv3+ is the most recent Deeplab version. Deeplabv3+ makes use of an Encoder-Decoder architecture, in which Deeplabv3 functions as the encoder. Deeplabv3+ extends Deeplabv3 by adding the decoder section of the architecture. The goal of this was to combine the strong features of both spatial pyramid pooling modules and encoder-decoder structures for DCNNs. More specifically, being able to encode multi-scale contextual information like a spatial pyramid pooling system, while also being able to capture sharper object boundaries like an encoder-decoder system. This architecture can be seen below in Figure 1.

The system achieved state-of-the-art performance on the PASCAL VOC2012 semantic segmentation benchmark, outperforming systems like PSPNet and ResNet-38 and the original Deeplabv3 model [9].

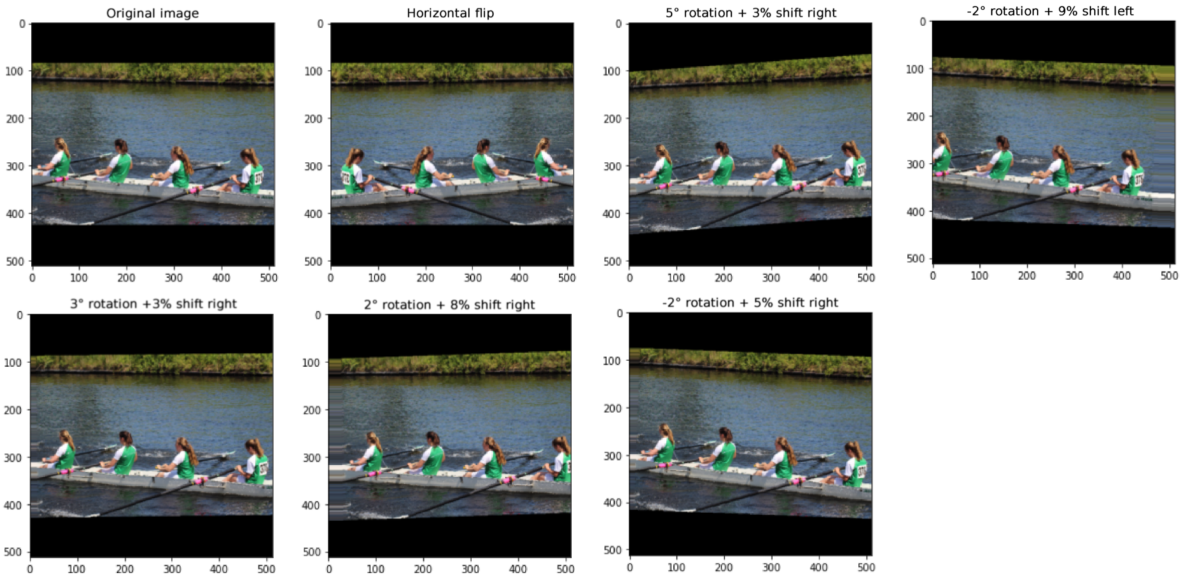
Original image
Horizontal flip
5° rotation + 3% shift right
-2° rotation + 9% shift left
3° rotation +3% shift right
2° rotation + 8% shift right
-2° rotation + 5% shift right
0
100
200
300
400
500

Figure 2. An example of the augmentations performed. The captions above each subplot show the augmentation performed.

3. PROBLEM STATEMENT

As discussed in the previous section, the current state-of-the-art in rowing video processing does not use any form of machine learning and is very restricted in terms of the usable footage. This study aims to determine whether a Deeplabv3+ model trained using transfer learning is well suited to detect the rower and the boat correctly in a variety of circumstances so such a system can be used in a wide range of video footage. The main research question is formulated as follows:
Can machine learning be used to accurately perform semantic segmentation on rowers and boats in images?

4. RESEARCH METHODOLOGY

The research was conducted in four phases. The first phase consisted of labelling the images in the dataset manually, as accurately as possible.
Phase two was writing the script for converting the data generated in the labelling process, to a format that was compatible with the Deeplabv3+ implementation.
In phase three, the Deeplabv3+ script was adapted to work with the rowing dataset and the various loss functions and metrics were implemented.
Finally phase four consisted of training the model using the chosen hyperparameters, applying final post-processing and evaluating the model using the test dataset. This evaluation was done for two variants of the retrained model with post-processing as well as a variant without post-processing. The best performing variant was then compared to a similarly post-processed pre-trained Deeplabv3+ model.

4.1 Resources

The resources necessary for this research were all digital. Keras is the Python library on which the Deeplabv3+ implementation was built. Keras is a high-level neural networks API, using TensorFlow as its back-end. Aside from this, a dataset from Stanford [11] as well as several taken by the Photo Committee of D.R.V Euros [2] were used and labelled manually, to train the system to correctly detect rowers and boats in an image. These photo sets contain numerous images of rowing activities, at various distances from the camera. The lighting and camera angle also vary. Finally, a data pre-processing script by Matterport [3] was used as a base for writing a dataset conversion script and a TensorFlow-based Deeplabv3+ implementation [17] was used as the base script for training the network.

4.2 Data annotation and processing

The total number of unique images in our dataset was 100. This dataset was split in parts of 75%/5%/20% for the training, validation and testing respectively. This is slightly different from the rule of thumb saying the dataset should be split 80%/10%/10%, but the training set could be augmented and the test set being diverse was deemed to be more important than the validation set being diverse.
To build the dataset during phase one, the labelling tool ”COCOAnnotator” was used [5]. This tool was chosen because the export format was in JSON, following the exact polygon coordinate list format that is used for the COCO dataset as well. To make this output compatible with Deeplabv3+, a script was written to load either the training, test or validation data, convert this from JSON polygon coordinate list format to NumPy arrays representing the correct pixel values for both the original image and the masks, augment the data if required and save it in a Deeplabv3+ compatible file. The conversion from JSON to NumPy arrays was done using a pre-existing script from the Mask-RCNN implementation by Matterport [3] that was adapted for this use case. The images were resized to 512x512 pixels, with padding if the aspect ratio was not square, to prevent memory size issues. Masks were saved in 512x512x3 NumPy arrays, making them 3D NumPy arrays with each third dimension containing a 512x512 mask for one of the three classes: background (0), boat (1) or rower (2). To counter the issue of having only 75 training images, the training images were augmented in various ways. For each original image, a horizontally flipped version was generated, as well as five augmentations that were randomly rotated within a range of $-5°$ to $5°$ and/or shifted horizontally by 0% to 10% left or right. For the horizontal shift, the non-existing pixels opposite to the shift direction were added using the nearest-neighbour principle. Examples of this can be seen below in Figure 2.

Table 1. Mathematical formulas for all metrics used.

Mean IoU	Freq. weighted IoU	Mean accuracy	Pixel accuracy
$(\frac{1}{n_{cl}}) * \frac{\sum_i n_{ii}}{(t_i+\sum_j n_{ji}-n_{ii})}$	$(\frac{1}{n_{cl}}) * \frac{\sum_i t_i * n_{ii}}{(\sum_k t_k)(t_i+\sum_j n_{ji}-n_{ii})}$	$(\frac{1}{n_{cl}}) * \frac{\sum_i n_{ii}}{t_i}$	$\frac{\sum_i n_{ii}}{\sum_i t_i}$

This process brought the number of training images up to 525. The validation and test data were not augmented.

4.3 Evaluation metrics

Seeing how semantic segmentation falls under the scope of COCO Stuff, the performance of all techniques were evaluated using the same four metrics used in Microsoft's COCO Stuff challenge: mean intersection over union, frequency weighted intersection over union, mean accuracy, and pixel accuracy [1]. The primary metric is the mean IoU, or mIoU. This metric gives a good idea about the performance of the model, as it takes calculates how well the predicted mask fits the ground truth mask, while only counting true negatives within the ground truth mask area. It is calculated by taking the intersection of the predicted mask and the ground truth mask, and dividing it by the union of those two, as shown in Figure 3. This is done for every class separately, and the mean is taken as the final result. Aside from the mIoU, the frequency weighted IoU, or fwIoU, is also used. This metric is similar to the mIoU, but with a weight assigned to each class based on how many pixels belong to the class within the ground truth mask. Finally, we have the mean accuracy (mAcc) and the pixel accuracy (pAcc). The mAcc is a metric that is calculated by calculating the accuracy for each class and then taking the average of these accuracy values. The main difference with the IoU based metrics, is that this counts all true negatives towards the performance of the model, regardless of whether those true negatives are within the ground truth mask area or not. The pAcc is calculated by simply dividing all correctly predicted pixels, so both true positives and true negatives, by the total number of pixels in the image.

The mathematical formulas for all metrics used can be found in Table 1, where n_{ij} is the number of pixels of class i predicted to belong in class j, n_{cl} is the number of classes being evaluated and $t_i = \sum_j n_{ij}$ is the total number of pixels in class i.

All of these metrics were implemented newly or adapted from existing (partial) implementations, to suit the data format used by the Deeplabv3+ script.

4.4 Architecture

The Deeplabv3+ implementation that was adapted in this research used pre-trained PASCAL VOC2012 weights. To adapt the model to this use case, a softmax activation layer was added to accommodate the focal loss function and all layers except the final 5 layers were frozen and made untrainable. This allowed the model to use all of its pre-learned structures to compensate for the small amount of training data that it had to learn from. The input shape of the tensors was set to 512x512x3, the number of classes to 3 and the backbone used was xception. The output stride of the model was set to 16, because 8 would lead to memory problems due to the size of the feature vector.

4.5 Model training

The optimizer used for training was Keras' Adam, with all parameters left default. The focal loss function used a gamma of 2 and an alpha of 0.25, as suggested to be best in most cases in the original paper [12].

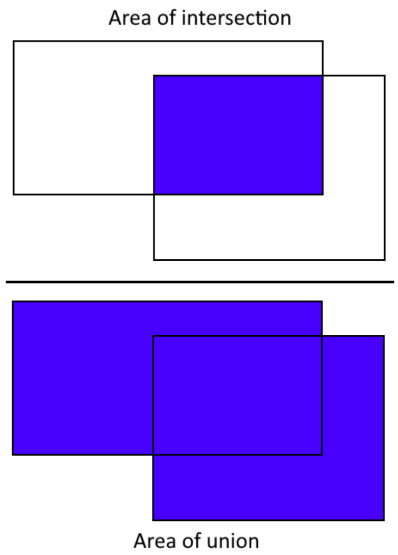

Figure 3. A visualisation of the intersection over union.

The loss used in this research was focal loss. This loss function is well suited to unbalanced datasets and data in which there is a lot of background pixels, as it is specifically designed to combat foreground-background class imbalance in dense detectors [12]. This loss function can be defined using the following formula:

$$L = -\alpha_t(1-p_t)^{\gamma} log(p_t) \tag{1}$$

where γ is a prefixed positive scalar value, α is a weighting factor to balance positive and negative predicted pixels in a range between $[0, 1]$ and

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases} \tag{2}$$

In equation 2, the y specifies the ground-truth class, and p is the probability for the class with label $y = 1$ as estimated by the model. In terms of notation, α_t can be written similarly to p_t.

The model was trained for 50 epochs with a batch size of 5. The model was evaluated with two post-processing options, the first being a certainty threshold of 0.2 for the predictions and the second being a certainty threshold of 0.3. The model without post-processing was also evaluated as a baseline.

The pre-trained Deeplabv3+ model was the original PASCAL VOC2012-trained model. With the exception of the number of classes, all other hyperparameters were the same as the retrained model. This was due to the retrained model containing only 3 classes, whereas PASCAL VOC2012 contains 21 classes, of which the fourth is a class for boats and the fifteenth for persons. For our comparison, we used these classes as our boat and rower classes respectively. This model was post-processed using the same threshold that performed best for the retrained model for the comparison between the two.

5. RESULTS

The results of the research are split in three parts. The first section discusses the validation performance of the new model briefly, to give an insight into the training progress made by the model. The second section contains a comparison between various versions of the retrained model. The third section shows a comparison between a pre-trained Deeplabv3+ model and our retrained variant. These final sections will be more extensive as these results will lead to an answer to our research question.

5.1 Validation performance

The validation loss, depicted in Figure 5, is not as smooth a line as the training loss shown in Figure 4. The same trend is visible however, with the loss initially decreasing significantly. It then started oscillating, as the overall trend still slightly decreased. The validation loss after the final epoch was 0.68. The decreasing trend shows that it was unlikely that overfitting occurred in the model, so it is suitable for testing. The reason why the validation loss had a lot more undulations in the value is most likely due to the limited size of the validation dataset. It is likely that there are images on which the model performed better than others. Variations could be caused by an aggregation of images the model segments with above-average quality in one epoch followed by a below-average segmentation quality batch in the next epoch. This makes sense from the perspective of the training loss as well, as this does not have this problem because the training loss is calculated over a factor 21 more images per epoch.

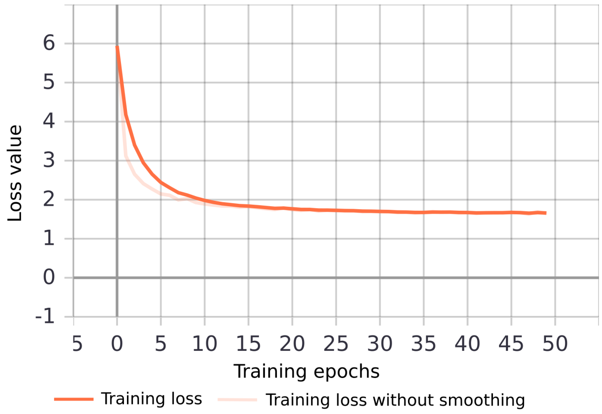

Figure 4. A graph of the training loss over epochs.

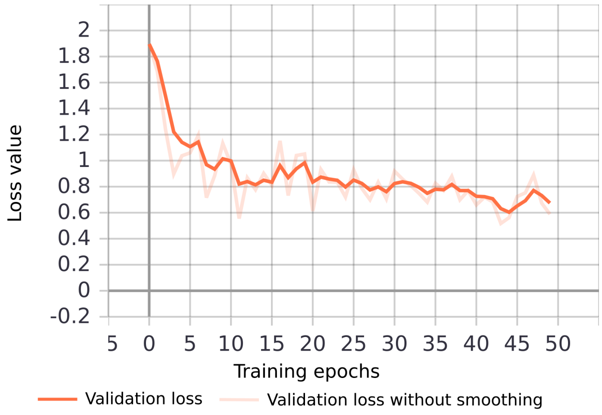

Figure 5. A graph of the validation loss over epochs.

5.2 Test results

The test results for the retrained model were evaluated with and without out certainty thresholds on the predicted values. A bar graph visualising the differences between these versions is shown in Figure 6. The blue bar is the baseline, which is using no certainty threshold. It is clearly visible how the post-processed models leapfrog the base variant in performance in all metrics except the accuracy based metrics. We can also see that while the IoU based metric scores are not incredibly high for any variant, the accuracy based ones are near perfect. This is most likely because accuracy is a metric that is strongly biased towards true negative predictions. As such, due to the small size of most masks in comparison to the entire image, the amount of ground truth negatives is very high, which in turn boosts the accuracy scores significantly. Even if the prediction would be negative for all pixels, it is likely that these scores would approach 85% accuracy. The metrics were included in the research because they are part of the COCO stuff challenge metrics, but the significant bias must certainly be taken into account when looking at the scores in these metrics.

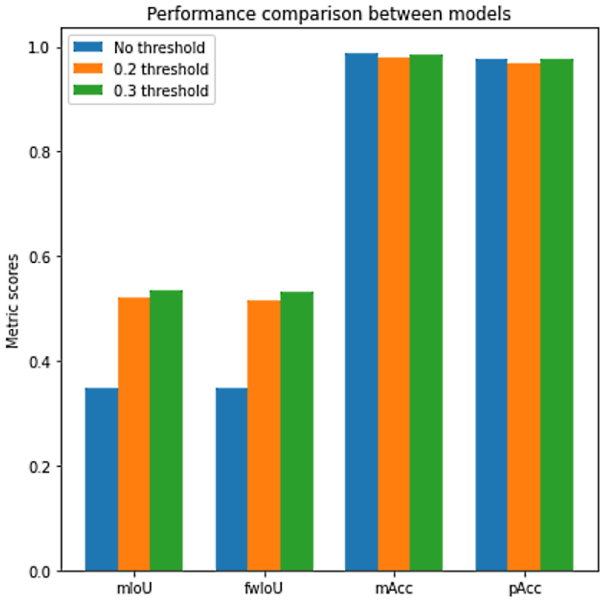

Figure 6. A bar plot of the performance of various retrained model variants.

Table 2. Test results of retrained Deeplabv3+ models with or without post-processing.

Certainty threshold	mIoU	fwIoU	mAcc	pACC
0.0	0.348	0.347	0.987	0.978
0.2	0.521	0.515	0.980	0.968
0.3	0.536	0.531	0.985	0.976

The exact results of the tests can be seen above in Table 2. The large performance increase for IoU based metrics is more clearly defined here. The 0.2 confidence threshold model achieved a 49.7% improvement and the 0.3 threshold achieved a 54.0% improvement over the baseline, for the mean IoU metric. For the frequency weighted IoU, the increase is comparable, but slightly smaller at 48.4% and 53.0% respectively. In the table it is also visible that the accuracy based metrics actually decrease compared to the base model. However, this decrease in performance is 1.0% at the highest, whereas the performance increase in

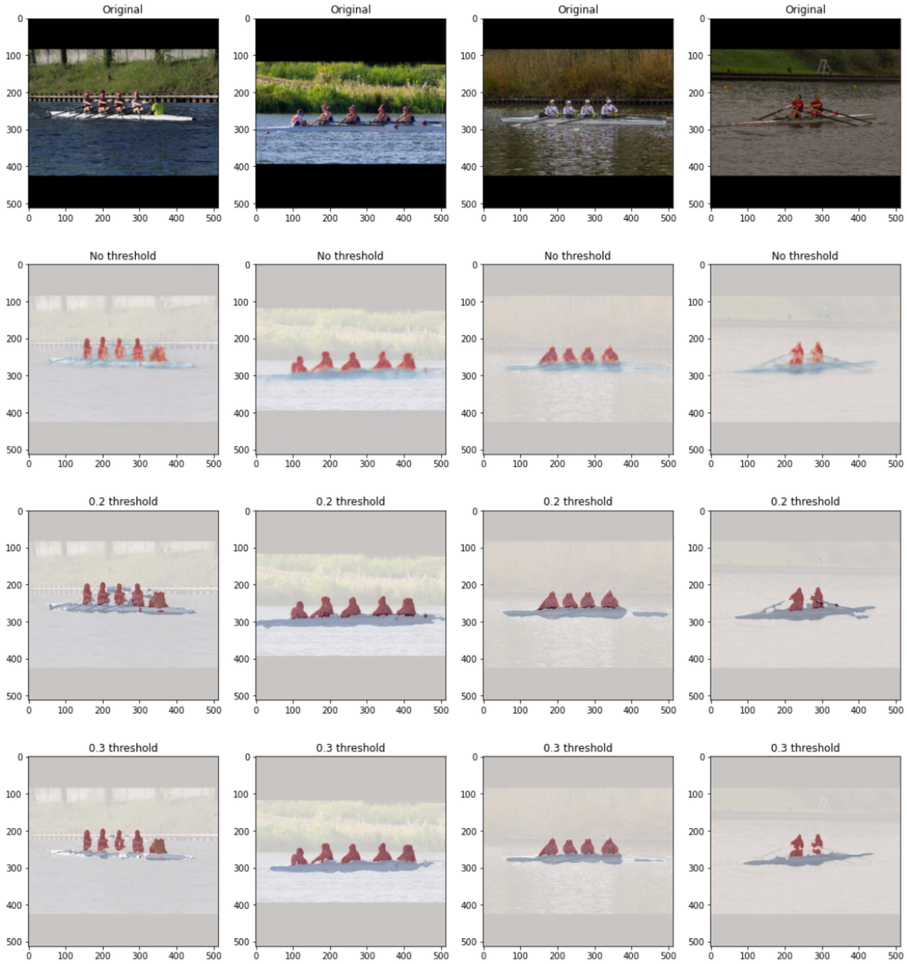

Figure 7. Predictions for the two classes with all three retrained model variants shown separately.

the more relevant metrics is 48.4% at least. Making this trade-off appears merely logical, considering the difference in significance and the higher significance of the IoU based metrics.

Figure 7 shows an random batch of images from the test dataset, along with their segmentation output of each tested variant. As is visible, the reason why the 0.3 threshold variant seems to perform better is due to a smaller amount of false negatives. This is especially visible in the difference between the 0.2 and 0.3 threshold segmentations in the second column. In the 0.2 threshold image we see that the stroke rower (the second person from the left) does not have a separately segmented arm. In the 0.3 threshold image, this is the case. Similarly, for the first column, the 0.2 threshold model classifies much more boat pixels in places where there should not be a boat compared to the 0.3 threshold model.

5.3 Model comparison

Based on the best performing post-processing variant of the retrained model, a similarly post-processed variant of the pre-trained model was evaluated to serve as a baseline and compared. In this case, both models used a certainty threshold of 0.3 as this performed best, following the results from the previous section. Once again, the results have been plotted in a bar chart, visible in Figure 8. We can see a significant increase in performance for the IoU based metrics when compared to the pre-trained model. Similarly to the comparison between retrained model variants, we see very high accuracy based metric scores for both models. As described in the previous section, we should look these values with caution due to the strong bias toward true negatives in accuracy based metrics.

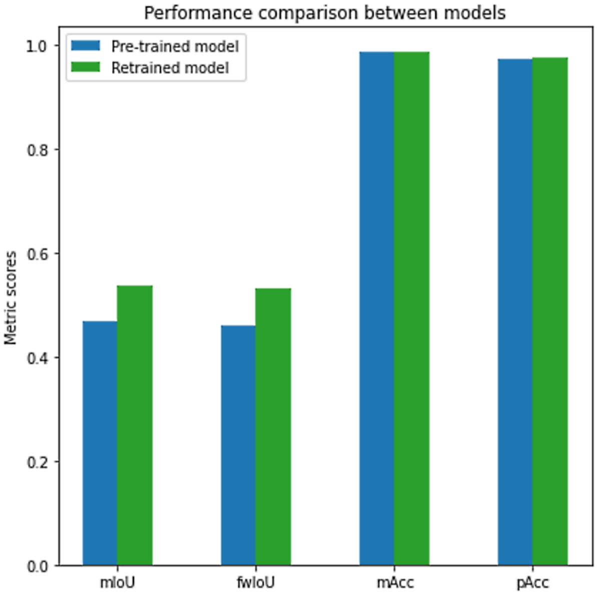

Figure 8. A bar plot comparing the pre-trained and retrained model performance.

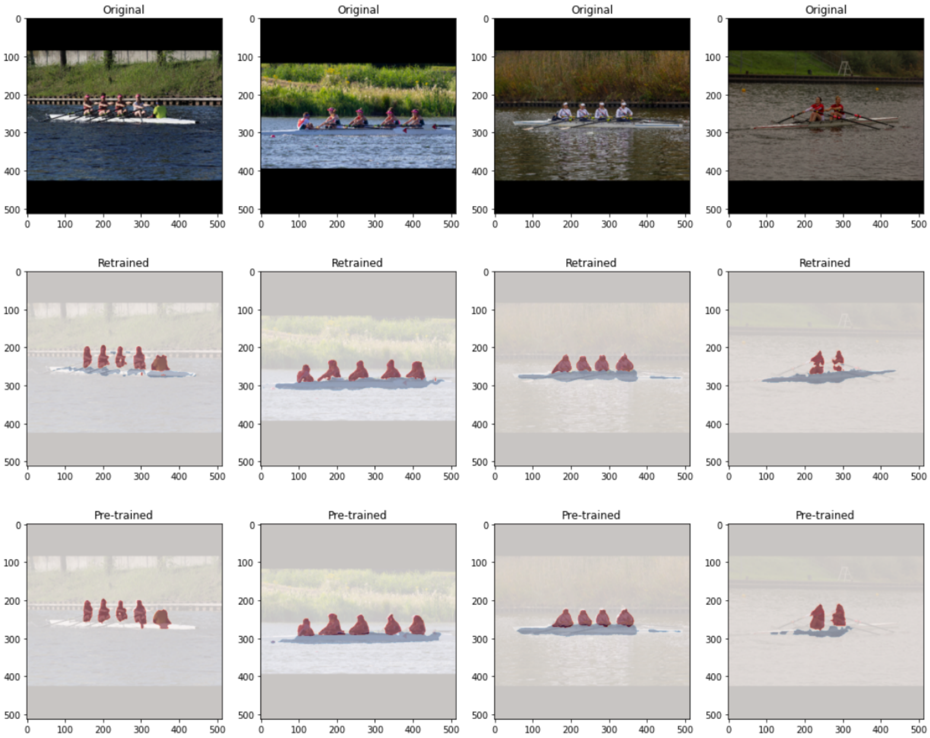

Figure 9. Predictions for the two classes by both tested models shown separately.

Table 3 contains the exact metric scores for both models. The bottom row shows the difference between the scores. From this, we see the increase in performance is around 15% for the IoU based metrics. The difference in accuracy scores is negligible, with a maximum difference of 0.3%.

Interestingly, the pre-trained model's performance does not come close to the performance it achieved on the PASCAL VOC2012 test. In that instance, it reached a mean IoU score 87.8% for the best performing model on that case [9]. This is most probably due to the difference in positions of rowers compared to walking or standing people, which is a more frequent situation in the PASCAL VOC2012 dataset. As for the boats, this might be due to the "atypical" shape of a rowing shell compared to a motorised boat which the pre-trained model has trained on significantly more. Even though it was expected that the pre-trained model would not perform as well as it did for the PASCAL VOC2012 test, it is remarkable that the performance discrepancy is so large.

Table 3. Test results of pre-trained and retrained Deeplabv3+ models.

Model	mIoU	fwIoU	mAcc	pACC
Pre-trained	0.468	0.460	0.985	0.973
Retrained	0.536	0.531	0.985	0.976
Difference	14.5%	15.4%	0.0%	0.3%

Finally, Figure 9 shows a visual comparison between the retrained model's segmentations and the pre-trained model's segmentations. The images are the same four used in 7, to keep things equal. In most images, we see that the pre-trained model struggles heavily with detecting the boat. The leftmost column does not even contain a boat in the pre-trained model's prediction. We also see a difference in accuracy, which is visible in column 2. The retrained model shows a more precise shape with a separate arm for the stroke rower as described in the last section as well, whereas the pre-trained model groups this entire area under the rower instance. It is likely that these false positives were part of the reason for the lower scores of the pre-trained model.

6. DISCUSSION

The dataset size was certainly the biggest limiting factor in this research. The labelling process was a lot more time consuming than expected, which led to a much lower number of labelled training images than generally considered sufficient to train a model on. This was partly countered by making use of transfer learning to leverage the already learned structures in the pre-trained model, but the expectation is that performance could be greatly improved further by expanding the size of the raw dataset by a factor 3 or higher. According to the theory described in Section 5.1, doing so might also result in a smoother validation loss graph, assuming the usual 80%/10%/10% partitioning would be adapted, rather than the current 75%/5%/20% division.

The dataset size problem may also have had an influence on the need for certainty thresholds as post-processing methods. The model still had difficulties filtering out some very low-certainty pixels if no certainty threshold was used. The exact reason for this is unsure, but we can speculate that the small training set led to the model having a low certainty for many pixels. As such, a recommendation to find out whether this is the case would be to repeat the tests with a much larger dataset to train the retrained model on.

Other recommendations for future work are in advancing the model towards a usable system for automatic video analysis. A first step could be training the model to differentiate between bystanders and rowers, possibly using some form of post-processing to relate the position of the candidate rower to the position of a boat. The initial training would require a very large dataset, because the

system would have to be able to distinguish between different "types" of people: rowers and regular persons. As mentioned, post-processing by filtering out predicted rowers that are not in the vicinity of a boat might help, as it is unlikely that a person without a boat nearby is a rower and more likely they are a bystander. An important requirement for this is that the boat segmentation is mostly correct, as random artefacts or poor segmentations would render this system unusable. Furthermore, it would also have to be able to do distinguish between rowing boats from other types of boats. Such post-processing would also introduce many new challenges, such as camera angles making bystanders appear almost equally close to a boat as an actual rower, which would need to be solved one by one.
Another big step towards an automatic analysis system would be segmenting separate body parts, such as the arms, legs and trunk of a rower. These three form the three main body parts from which the rowing stroke follows. If those could be separately segmented properly, at a speed which facilitates (real-time) segmenting of video footage, a body part tracking system would be close, although achieving this would require a very large amount of data. Combining these two would lead to a system that track the body parts of rowers only, which in turn would allow the development of a system that takes this tracking information and tests the movements of the rower against their desired rowing stroke's movements.

7. CONCLUSION

Our results allow us to answer the research question: can machine learning be used to accurately perform semantic segmentation on rowers and boats in images? This study shows that using machine learning for semantic segmentation of rowers and boats is moderately accurate, certainly making it promising. Strong improvement is required, however, to be able to apply such a system to video analysis.

8. ACKNOWLEDGEMENTS

I would like to express my gratitude to Dr. Le Viet Duc, who supervised me during this research and gave me extensive feedback while writing this paper. I would also like to offer my special thanks to L.B. (Daan) van den Oever MSc, who gave me much of his time to explain and help adapt the Deeplabv3+ script.

9. REFERENCES

[1] Coco evaluation metrics, may 2020.

[2] D. r. v. euros website, june 2020.

[3] W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow, 2017.

[4] R. Ballreich and W. Baumann. *Grundlagen der Biomechanik des Sports*. Enke, Stuttgart, 1988.

[5] J. Brooks. COCO Annotator, 2019.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2014.

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2016.

[8] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation, 2017.

[9] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

[10] V. Kleshnev. *The biomechanics of rowing*. The Crowood Press Ltd, Ramsbury, 2016.

[11] L. Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[12] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection, 2017.

[13] N. Schaffert and K. Mattes. Interactive sonification in rowing: Acoustic feedback for on-water training. *IEEE MultiMedia*, 22(1):58–67, 2015.

[14] G. Szűcs and B. Tamás. Body part extraction and pose estimation method in rowing videos. *Journal of Computing and Information Technology*, 26:29–43, 01 2018.

[15] B. D. Wilson. Development in video technology for coaching. *Sports Technology*, 1(1):34–40, 2008.

[16] M. Worsey, H. Espinosa, J. Shepherd, and D. Thiel. A systematic review of performance analysis in rowing using inertial sensors. *Electronics*, 8:1304, 11 2019.

[17] E. Zakirov. Keras implementation of deeplabv3+, 2018.