To appear in MTS/IEEE OCEANS Conference Kobe, 2018. DOI: not yet available

Looking for Seagrass: Deep Learning for Visual Coverage Estimation

Gereon Reus*, Thomas Möller*, Jonas Jäger*‡, Stewart T. Schultz†, Claudia Kruschel†,
Julian Hasenauer*, Viviane Wolff* and Klaus Fricke-Neuderth*
*Department of Electrical Engineering and Information Technology, Fulda University of Applied Sciences, Germany
Email: gereon.reus@et.hs-fulda.de
†Department of Ecology, Agronomy and Aquaculture and CIMMAR, University of Zadar, Croatia
Email: ckrusche@unizd.hr
‡Computer Vision Group, Friedrich Schiller University Jena, Germany
www.inf-cv.uni-jena.de

***Abstract*—Underwater videography enables marine researchers to collect enormous amounts of seagrass image data. This collection is fast and cheap but the manual analysis of such data is slow and expensive. Therefore, we propose a machine-learning approach for the automatic seagrass coverage estimation of the sea bottom. Our contribution is the investigation of CNN features to describe patches and superpixels of seagrass. CNN features are the activations of a specific layer in a deep convolutional neural network. We also provide the first public available dataset of seagrass images that can be used as a benchmark for automatic seagrass segmentation. Our best method achieves an accuracy of 94.5% for seagrass segmentation on the provided dataset. Our code and dataset is available on GitHub: https://enviewfulda.github.io/LookingForSeagrass/**

1. Introduction

Seagrass meadows have a major impact on the coastal environment. They provide carbon within both the grazing and detrital food webs, improve water quality, stabilize the shoreline, and offer living surface and habitat for juvenile and adult fish and invertebrates. Therefore most countries with ocean territory have mandated seagrass monitoring programs as a component of overall management of the ocean environment. Since the quantification of seagrass meadows in situ by human divers is labor intensive, we use underwater images to quantify seagrass coverage of the sea bottom at the Adriatic sea in Croatia. In that way we can collect a huge number of seagrass images with the corresponding GPS location. All images are taken by an autonomous underwater vehicle (AUV). AUVs are a time- and cost efficient way for seagrass monitoring as shown in [1], but our results also apply to other means of obtaining images, such as with ROVs, towed cameras, or divers. In a post processing step all images need to be analyzed by a human to estimate the fraction of seagrass in the image. This process is slow and error-prone. In order to overcome these problems, automatic methods for visual seagrass coverage estimation are proposed in literature.

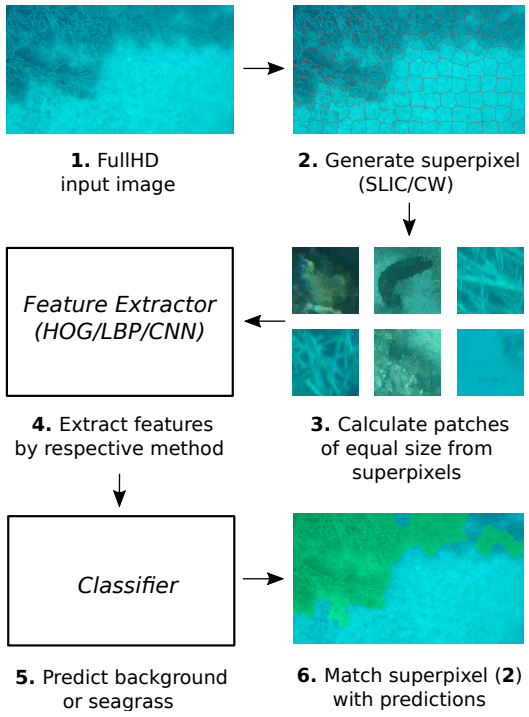

Figure 1. Prediction procedure using superpixels: From an input image superpixels are extracted. From these superpixels we calculate rectangular patches of equal size and extract their features. Using the features, a classifier predicts each patch. Afterwards the predictions are matched with the previously calculated superpixel in order to obtain a pixel-accurate prediction.

2. Related Work

Massot-Campos et al. [2] divide the image in small patches and classify each patch as *seagrass* or *background*. They test different classifiers and describe the image patches with texture based features. The papers [3], [4] and [5] follow also a patch classification approach to segment the image as proposed in [2]. Bonin-Font et al. [4] use also a pixel refinement method as post processing step to get better than with pure patch based classification. In contrast

to that we try a superpixel based [6] classification approach. Gonzalez-Cid et al. [5] utilize a convolutional neural network (CNN) trained on seagrass images and compare it with a support vector machine (SVM) trained with texture features. Their CNN also performs the classification process using a classification layer. As opposed to this work we are using a specific layer of a CNN to extract features. With these features we train a seagrass classifier.

3. Dataset

Recording The dataset consists of 12682 images recorded with an AUV along the coast of the island of Murter, Croatia by a team of marine biologist. The diving device is equipped with a pressure and sonar sensor to measure the distance to the water surface as well as the distance to the bottom of the sea. An integrated control system ensures that the device slide over the sea bottom at a distance of 2m (tolerance $\pm$0.5m) During the image capturing the AUV is moving with approximately 1m/s along the sea bottom. Figure 2 illustrates a layout of the diving operation. A waterproof GoPro Hero 2 Action Cam was used for the image capturing. The frame rate was set to 1 frame per second, which leads to approximately one image per covered meter. All images have a resolution of 1920 x 1080 pixels.

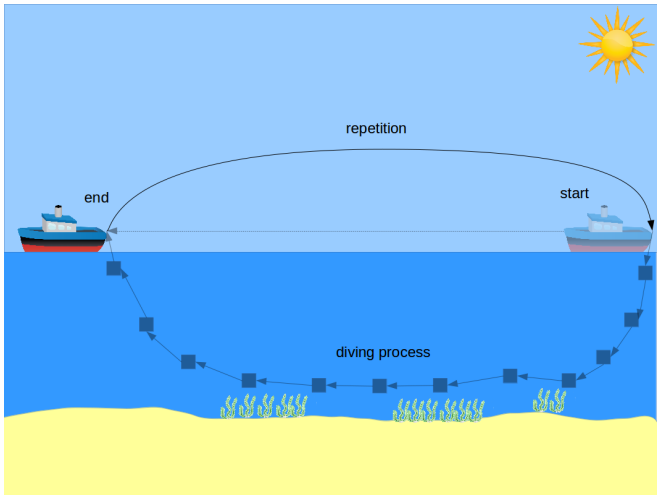

Figure 2. Schematic illustration of the image capturing process using an AUV

Annotation Each image has been annotated with polygons indicating if a pixel belongs to the class *seagrass* or *background*. These polygons have been transformed to black/white pixelmaps where black means *seagrass* and white signifies *background*. The annotation work was done by two learned human annotators, trained by an expert in marine biology. One annotator was able to annotate approximately 400 images per day. In whole we needed two weeks to annotate all seagrass images.

The annotated dataset consists of 6037 annotated images that have been taken in different distances to the ground, since the AUV is diving up and down between different transects. The annotators stated, that it was hard for them to annotate images when they became blurry. This has already been the case for distances of more than 4 meters. If the distance is more than 6 meters, images become out of focus, and for this reason, we did not annotate them. The images show a combination of seagrass, algae, sediment background and occasional fish and invertebrates. Table 1 shows the distribution of the images over the different depths in relation to the sea bottom. Our public available dataset can be downloaded at:
https://enviewfulda.github.io/LookingForSeagrass/

TABLE 1. IMAGES: DISTRIBUTION BY DEPTH IN RELATION TO THE SEA BOTTOM

depth d	# images
$0 >= d < 1$m	89
$1 >= d < 2$m	2522
$2 >= d < 3$m	2273
$3 >= d < 4$m	471
$4 >= d < 5$m	332
$5 >= d < 6$m	349
$6 >= d < 31$m	6645 (not annotated)

4. Method

We propose a *superpixel* classification approach to tackle the problem of underwater seagrass coverage estimation in images.

Figure 1 demonstrates the main idea of our method: (**1.**) Given an input image, (**2.**) we use a superpixel method to extract segments that adjust smoothly to the boundaries between seagrass and background. As superpixel methods we test *Simple Linear Iterative Clustering* (SLIC) [6] and *Compact Watershed* (CW) [7] algorithm. (**3.**) Each superpixel is transformed into a rectangular patch. (**4.**) The transformed patches are feed into a feature extractor. (**5.**) A logistic regression classifier classifies each patch either into the class *seagrass* or *background*. (**6.**) The predicted patches are matched with the previous determined superpixels in order to obtain a contour-accurate prediction.

4.1. Superpixel

Superpixel algorithms search for homogeneous regions in the image, while such a region is called a Superpixel. Our main idea was to have smoothly adapted boundaries between seagrass and background regions. As superpixel methods we test *Simple Linear Iterative Clustering* (**SLIC**) [6] and *Compact Watershed* (**CW**) [7] algorithm. We choose these algorithms in order to obtain regular and similar sized superpixels. However, it cannot be guaranteed that each superpixel contains the same number of pixels. Since the classifier needs to be fed with equal sized feature vectors, we convert each superpixel into a rectangular patch of the same size. To get back to a pixel accurate prediction, the

classified rectangular patches are mapped to the corresponding superpixels.

4.2. Feature Extraction

In our experiments we test three different types of features to describe the image patches: histograms of oriented gradients (HOG) [8], local binary patterns (LBP) [9] and convolutional neural networks (CNN) [10].

Histogram of Oriented Gradients (HOG) The histogram of oriented gradients algorithm by [8] offers a good representation of texture features. Especially for seagrass images the texture representation is important, since the long seagrass leafs are a good marker to distinguish it from algae, snails and the sea bottom. The HOG method of Dalal et al. is based on the assumption that the local appearance or shape of an object is quite well represented by the distribution of the local intensities or edge directions which can be characterized without having any knowledge about the positions of the corresponding gradients or edges. We use the standard implementation of the HOG algorithm which is part of the *OpenCV* library [11].

Local Binary Patterns (LBP) As well as HOG, the use of local binary patterns [9] is a versatile method for extraction of texture features in images. Particularly the simple calculation and the invariance to monotonous brightness changes are typical characteristics of this method. We use the standard implementation of the LBP algorithm which is part of the *scikit-image* library [12].

Convolutional Neural Networks (CNN) As third representation to describe seagrass images we use CNN features. These features can be obtained from a pretrained convolutional neural networks when feeding an image into the network. For feature extraction the neural activations of a specific layer in a convolutional neural network are utilized. As described by Donahue et al. [10] a network can also be used for feature extraction if it was trained for another task, since the network learns a general feature representations from the dataset it was trained on. In our case we utilize a network that was trained on ImageNet [13]. ImageNet is a dataset consisting of 1000 different categories like dog breeds, balloon or taxi. As network architecture we use InceptionNetV3 [14]. The features are extracted from Layer *pool_3*, which is known to be rich of semantics.

4.3. Training

The training procedure is visualized in Figure 3: (**1.**) From our training dataset described in Section 3 we use the polygon annotated images (pixelmaps) and (**2.**) split the original image into rectangular patches. Each patch belongs either to class *seagrass* or *background*. (**3.**) Using these patches we extract features with the respective extraction method (Section 4.2). (**4.**) As last step we train a logistic regression classifier using these features.

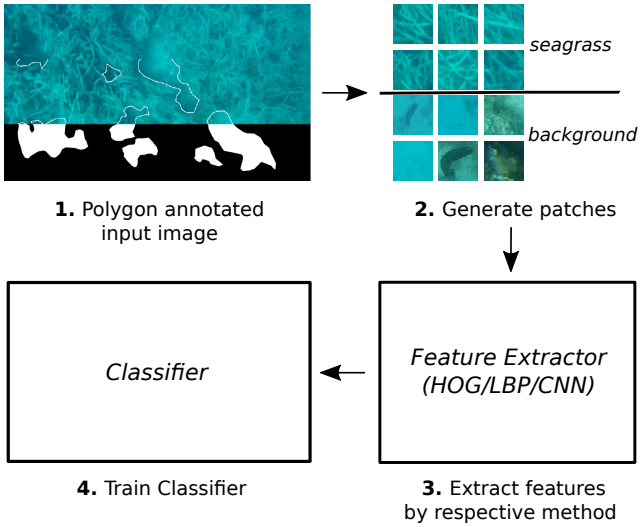

Figure 3. Training procedure: Features are extracted from certain patches in order to train a logistic regression classifier.

4.4. Rectangular Patches

As a second method we propose a patch classification approach as described by Massot-Campos et al. [2] and subsequent papers [3], [4], [5]. During the first step of this approach the input image is divided into *rectangular patches*. Afterwards features for each patch are extracted. In our project we use the presented features from Section 4.2. Then each patch is classified as *seagrass* or *background* by a machine-learning classifier. As presented in Section 4 we use a logistic regression classifier. The result image contains patchwise predictions about the *seagrass* and *background* classes. In the following we will refer to this approach as **RP**, since rectangular patches are generated. Our contribution here is the application of CNN features to describe these image patches.

5. Experiments

5.1. Implementation Details

The implementation of our experiments is based on Python using the *TensorFlow* (CNN), *OpenCV* (HOG), *scikit-image* (LBP) and *scikit-learn* [15] (logistic regression classifier) libraries. We used the following hardware setup for our experiments: Intel Xeon CPU E5-2620 v3 2.4Ghz, 32GB of RAM and a Geforce TitanX 12GB. Note that the GPU was only used for CNN feature extraction. Other feature extraction methods as well as the training and prediction of the classifier were computed by the CPU. The code for our experiments is available on GitHub: **https://enviewfulda.github.io/LookingForSeagrass/**

5.2. Evaluation Protocol

Dataset We utilize 70% of the polygon annotated images for training, 20% for testing and 10% for validation. The

train- , test- and validation sets are equally distributed among the different depth classes of the dataset.

Metrics Shelhamer et al. [16] presented four different metrics adapted to semantic segmentation. They use the pixel accuracy and intersection over union (IU) metrics to evaluate their semantic segmentation algorithms. Everigham et al. [17] are convinced that pixel accuracy is not sufficiently meaningful since this evaluation will end up in a perfect score in the case of assigning the same label to all pixels. However, we do disclose the results of this evaluation method for comparative purposes. With that in mind we decided to utilize the following metrics of Shelhamer et al. [16].
Mean intersection over union (mean IU):

$$\text{mean IU} = \frac{1}{n_{cl}} \sum_i \frac{n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

Frequency weighted intersection over union (f.w. IU):

$$\text{f.w. IU} = \frac{1}{\sum_k t_k} \sum_i \frac{t_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

Pixel accuracy (pixel acc.):

$$\text{pixel acc.} = \frac{\sum_i n_{ii}}{\sum_i t_i}$$

Mean accuracy (mean acc.):

$$\text{mean acc.} = \frac{1}{n_{cl}} \sum_i \frac{n_{ii}}{t_i}$$

With n_{cl} as the number of different classes, n_{ij} as the number of pixels of class i predicted to belong to class j. So n_{ii} is the number of correct predicted pixels. And $t_i = \sum_j n_{ij}$ as the total/ true number of pixels belonging to class i.

5.3. Results

Experiment I: Feature comparison The aim of the first experiment is to find out which features are most suitable. The three different feature methods (HOG, LBP, CNN) from Section 4.2 are tested using superpixel patches generated with SLIC, CW (Section 4.1) or rectangular patches RP (Section 4.4). For this experiment all annotated images of the dataset have been utilized.

Table 2 shows the results of Experiment I where different combinations of patch generation and features are tested for a fixed patchsize. It can be observed that CNN features work best for all patch generation methods and LBP features are better than HOG features. For patch generation regular patches and SLIC superpixels do an equally well job for a patchsize of 240×240, while compact watershed (CW) performs worst. This is astonishing since SLIC is able to generate much smoother boundaries between seagrass and background than RP from a human point of view. Please see Figure 4 for a qualitative comparison of the different patch generation methods.

TABLE 2. RESULTS FOR EXPERIMENT I: PATCHSIZE 240 X 240 PIXEL

method	mean IU	f.w. IU	pixel acc.	mean acc.
HOG-RP	62.76	67.75	72.86	73.22
CNN-RP	**80.96**	89.23	93.21	**85.72**
LBP-RP	73.05	80.42	85.01	80.69
HOG-SLIC	63.26	69.46	74.32	72.33
CNN-SLIC	80.55	**89.30**	**93.40**	84.93
LBP-SLIC	72.46	80.43	84.95	79.20
HOG-CW	62.58	68.77	73.59	71.48
CNN-CW	75.27	84.71	89.28	79.92
LBP-CW	68.63	76.91	81.56	74.80

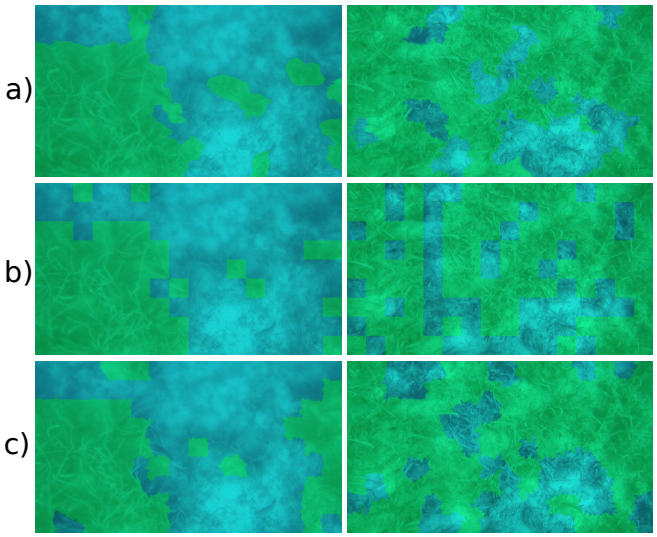

Figure 4. Examples for good and bad predictions for different patch generation methods. On the left side you can see good examples and on the right bad predictions. In the first row a) SLIC is used for patch generation. The second row b) shows rectangular patches and row c) presents patches generated with compact watershed (CW). For all predictions CNN features have been used. This Figure is best viewed in color. The light green regions indicate seagrass predictions.

Experiment II: The distance to the sea bottom The goal of this experiment was see to how well the proposed methods adjust to images that have been taken from different distances to the sea bottom; images in a larger distance to the ground are less focused than near ones.

For this experiment we took the best methods of Experiment I and performed them with a fixed patch size of 240×240. Table 3 presents the results for SLIC and RP patch generation in combination with CNN features. We compare the segmentation performance when using all annotated images of the dataset in a distance from 0 to 6 meters to the sea bottom and in a distance range from 0 to 2 meters. As expected, the performance for closer images is better and CNN-SLIC and CNN-RP do a similar good job. But it can also be observed that the performance is only

around one percentage point better. This seems not to be significant since the error of the human annotators should be higher than one percent for this dataset. Thus, our method adjusts well to images taken from different distances to the sea bottom on our dataset.

TABLE 3. EXPERIMENT II: INFLUENCE OF DIFFERENT DISTANCES TO THE GROUND. PATCHSIZE 240 X 240 PIXEL

method	mean IU	f.w. IU	pixel acc.	mean acc.
CNN-RP (0-6m)	80.96	89.23	93.21	85.72
CNN-SLIC (0-6m)	80.55	89.30	93.40	84.93
CNN-RP (0-2m)	**81.88**	**90.83**	**94.35**	**86.25**
CNN-SLIC (0-2m)	80.93	90.42	94.14	85.24

Experiment III: Patchsize comparison This third experiment investigates the influence of different patchsizes on the segmentation accuracy and processing speed. Since we found out in Experiment I that SLIC superpixel and rectangular patches (RP) work much better than compact watershed (CW) as patch generation method, we only use SLIC and RP for this experiment. We also saw in Experiment I that CNN features perform best to describe seagrass images, so we just use CNN features for Experiment III. Since we found in Experiment II that the distance to the ground has no significant influence on the accuracy, we only use images that have been taken in a distance between 0 and 2 meters from the bottom for this experiment.

In Table 4 we see CNN-RP and CNN-SLIC performance for patchsizes of 240×240, 180×180 and 120×120. CNN-RP performs always better than CNN-SLIC among all patchsizes. The segmentation performance seems to get a bit better for smaller patchsizes, but processing time per image doubles with each tested smaller patchsize.

In conclusion, the chosen patchsize has a huge impact on processing time but a small on the segmentation accuracy.

TABLE 4. RESULTS FOR EXPERIMENT III. COMPARISON OF DIFFERENT PATCH SIZES AND PROCESSING TIME PER IMAGE.

method	mean IU	f.w. IU	pixel acc.	mean acc.	test time per image
CNN-RP-240	81.88	90.83	94.35	86.25	**1 s**
CNN-SLIC-240	80.93	90.42	94.14	85.24	4 s
CNN-RP-180	82.70	**91.14**	**94.50**	87.20	2 s
CNN-SLIC-180	82.42	91.08	94.53	86.66	5 s
CNN-RP-120	**83.25**	90.93	94.16	**88.48**	4 s
CNN-SLIC-120	82.22	90.54	93.95	86.92	9 s

6. Conclusion

We tackle the problem of automatic seagrass segmentation in underwater images to estimate the sea bottom coverage. This coverage can be used as a measure to monitor the change of seagrass meadows over time. Our machine-learning approach utilizes CNN features extracted from image patches to classify each patch as *seagrass* or *background*.

We also tested other features in our experiments (Section 5.3) and found that CNN features clearly outperform HOG and LPB features. In order to improve boundaries between seagrass and background regions we tested superpixel for image patch generation. But the surprising result of our experiments was that superpixel work just as well as rectangular patches, even though the boundaries of superpixel patches look much better adjusted to seagrass regions (see Figure 4) from a human point of view. Our best method (see Table 4) achieves a mean intersection over union of 83.25% and a frequency weighted IU of 91.14%.

We further introduce a new dataset of 12684 seagrass images (Section 3) that is publicly available:
https://enviewfulda.github.io/LookingForSeagrass/

In the future we plan to fine-tune the deep neural network to seagrass images which should further improve our method. Furthermore, other networks can be tested for their performance. A further step will be to test a semantic segmentation approach using fully convolutional networks as described for example in [16]. Since our best method already provides useful results, we plan to implement an easy-to-use web-based interface for marine biologists. In terms of the dataset it would be interesting to determine the human error in order to have an upper bound for the accuracy a machine could achieve on maximum.

References

[1] A. Vasilijevic, N. Miskovic, Z. Vukic, and F. Mandic, "Monitoring of seagrass by lightweight auv: A posidonia oceanica case study surrounding murter island of croatia," in *22nd Mediterranean Conference on Control and Automation*, June 2014, pp. 758–763.

[2] M. Massot-Campos, G. Oliver-Codina, L. Ruano-Amengual, and M. Mir-Juli, "Texture analysis of seabed images: Quantifying the presence of posidonia oceanica at palma bay," in *2013 MTS/IEEE OCEANS - Bergen*, June 2013, pp. 1–6.

[3] A. Burguera, F. Bonin-Font, J. L. Lisani, A. B. Petro, and G. Oliver, "Towards automatic visual sea grass detection in underwater areas of ecological interest," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sept 2016, pp. 1–4.

[4] F. Bonin-Font, A. Burguera, and J. L. Lisani, "Visual discrimination and large area mapping of posidonia oceanica using a lightweight auv," *IEEE Access*, vol. PP, no. 99, pp. 1–1, 2017.

[5] Y. Gonzalez-Cid, A. Burguera, F. Bonin-Font, and A. Matamoros, "Machine learning and deep learning strategies to identify posidonia meadows in underwater images," in *OCEANS 2017 - Aberdeen*, June 2017, pp. 1–5.

[6] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov 2012.

[7] P. Neubert and P. Protzel, "Compact watershed and preemptive slic: On improving trade-offs of superpixel segmentation algorithms," in *2014 22nd International Conference on Pattern Recognition*, Aug 2014, pp. 996–1001.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 886–893 vol. 1.

[9] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915–928, June 2007.

[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *CoRR*, vol. abs/1310.1531, 2013.

[11] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[12] S. van der Walt, J. L. Schnberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. a. Yu, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, Jun. 2014. [Online]. Available: https://doi.org/10.7717/peerj.453

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, April 2017.

[17] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results," http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.