Water Tanks and Swimming Pools Detection in Satellite Images: Exploiting Shallow and Deep-Based Strategies

Eduardo A. M. Fernandes, Pedro F. Wildemberg, Jefersson A. dos Santos
Department of Computer Science
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
Email: {eduardofernandes,pedrowildemberg,jefersson}@dcc.ufmg.br

***Abstract*—This paper aims to study and to evaluate two distinct approaches for detecting water tanks and swimming pools in satellite images, which can be useful to monitor water-related diseases. The first approach, shallow, consists of using a Support Vector Machine in order to classify into *positive* and *negative* a discretized color histogram of a given segment of the original image. The second method employs the Faster R-CNN framework for detecting those objects. We built up swimming pools and water tanks datasets over the city of Belo Horizonte to support our experimental analysis. Our results show that the deep learning method greatly outperforms the shallow strategy, achieving an average precision at 0.5 IoU of over 93% on the swimming pool detection task, and over 73% on the water tank one. All the code and datasets are publicly available.**

***Index Terms*—Remote Sensing, Swimming Pool, Water Tank, Detection, Deep-Learning, SVM**

I. Introduction

Remote sensing is of great importance for vector-borne disease control, since, through unmanned aerial vehicles (UAV) and satellite imagery, it can provide fast, precise, and large-scale surveillance of areas infected by the vector. This information can, then, be used by health officers to locate the agent's breeding sites and determine where it should act to best combat the infection [1].

When it comes to mosquitoes, responsible for transmitting a variety of deadly and expanding diseases across the world with millions of cases registered every year, the main breeding spots are containers and debris holding still water pools, such as tires, plant vases, bottles, water tanks and poorly maintained swimming pools [2] [3]. More specifically, the last two are usually large enough to be spotted in satellite images, making them possible objects of interest when applying remote sensing techniques to detect the breeding sites of such species. On this subject, with the advent of Deep Learning (DL) techniques for image segmentation and object detection, heavily dependent on GPUs usage, Shallow Learning (SL) methods, mostly CPU demanding, are becoming more obsolete every day for said problems. A big advantage of the former method is the automation of the feature extraction step when attempting to classify an image or detect an entity in it, whereas, in the latter, it is the data scientist responsibility to identify and design visual feature extraction strategies to better represent the image for the task [4]. Moreover, in recent years, DL methods have been outperforming SL ones in several computer vision problems in terms of accuracy, leading to a major shift in approaches used for solving such tasks.

With water tanks and swimming pools as targets, this paper aims to compare the performance of SL and DL approaches for object detection. Such analysis is of great importance in this scenario, since such mosquito-borne diseases are a life threatening problem in least developed and developing countries, where the choice for a computational application can be heavily influenced by the available budget for hardware expenses. Accordingly, two new datasets were assembled, containing thousands of annotated swimming pools and water tanks in high resolution satellite images. Our datasets and analysis of two well-established approaches aim to serve as a new starting point for works related to the detection of mosquitoes breeding spots [2], [3], focusing on the trade-off between computational complexity and performance, and all the code has been made publicly available [1].

II. Related Work

Literature contains a variety of papers related to both the topic of detecting swimming pools in satellite images and comparing SL and DL methods in several different tasks. Specifically in the former problem, some papers stand out for the use of shallow methods to try and solve the task and the motivations that led them to do so.

Tien, Rudra & Hope employed a support vector machine (SVM) [5] to detect swimming pools in satellite imagery, by calculating the difference between the blue-red and blue-green combinations of all pixels and feeding this information to the machine. The goal of this work was to locate water bodies in Australia and help in bushfire fights in the country [6].

In [7], McFeeters proposes the Normalized Difference Water Index (NDWI) [7] to delineate open water features in aerial imagery, by making use of the near-infrared and green bands of the given image, and in [8], Kim, Holt, Eisen, Padgett, Reisen,

[1] https://github.com/EduardoFernandes1410/PATREO-Dengue

& Crof integrate the index with the rectangular fit space metric [9], proposing to delineate water features in aerial images in order to identify pools and aiming to assist in the control of the Culex mosquito population, vector of the West Nile Virus, in the United States of America. The method achieved a user's accuracy of 92.8% in pool negative samples and 80.1% in pool positive ones. Moreover, Alonso and Rodríguez-Cuenca described and used the Normalized Difference Swimming Pools Index (NDSPI) to semi-autonomously detect swimming pools, alongside with region adjacency graph (RAG) and principal component analysis (PCA) [10], scoring an overall accuracy of 99.86% according to the authors [11].

When it comes to comparing shallow and deep learning methods in tasks from different areas of study, Pasupa & Sunhem tested the performance of a convolutional neural network (CNN) [4] and a SVM in a classification problem with a small dataset, and found that, without applying data-augmentation techniques, the latter performed better then the former. However, when making use of such techniques to improve the dataset, the CNN method presented results that were competitive to the shallow method [12].

In [13], Koutsoukas, Monaghan, Li & Huan compare the performance of a deep neural network (DNN) [4], Naïve Bayes [14], K-nearest neighbours [15], random forest [16] and SVM methods for the problem of modeling bio-activity data, and conclude that DNNs, with optimal hyper-parameters and low noise levels, outperforms every other method applied.

Finally, Liu, Abd-Elrahman, Morton, & Wilhelm compared CNNs, random forest and SVM in a remote sensing task to map wetlands from a object-based level [17]. In agreement to [12], the authors found that the shallow learning methods performed better than the deep learning ones when the training dataset was small, but once more training samples were used, the latter obtained the superior results.

III. Assembly of the Dataset

Two separate datasets were assembled in this work: BH-Pools and BH-WaterTanks, with annotated swimming pools and water tanks respectively, and can be freely downloaded in this link [2]. Both datasets consist of imagery from several neighbourhoods in the city of Belo Horizonte, Minas Gerais, Brazil. The data was acquired through the Google Earth Pro tool. The images were exported from an eye altitude of 330 meters with a resolution of 3840x2160 (4K), and the image bands are the three visible ones: red, green and blue. For each occurrence of the target objects found on the images, a polygon was drawn in order to generate the segmentation masks of the instance. For the detection problem, a bounding box for each of the annotated objects was calculated through said masks. Fig. 1 and Fig. 2 show a few examples of the images and ground-truths in BH-Pools and BH-WaterTanks, respectively, and Table I summarizes the datasets specifications.

[2]http://www.patreo.dcc.ufmg.br/bh-pools-watertanks-datasets/

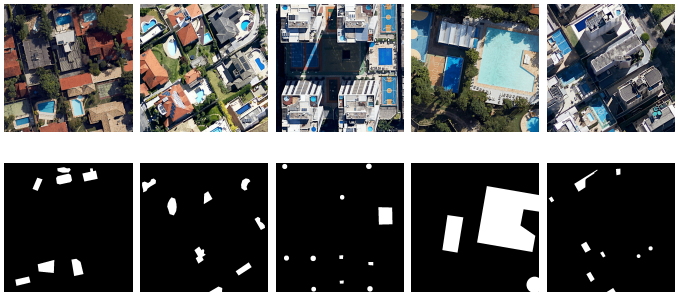

Fig. 1: Example crops from BH-Pools

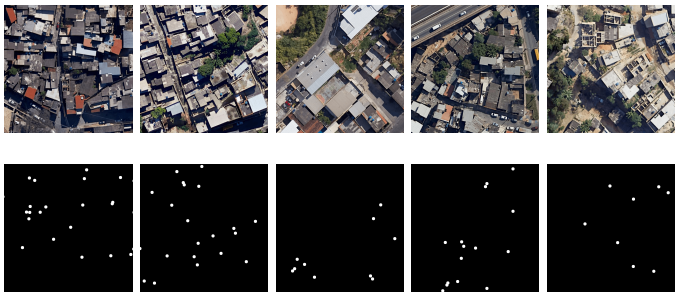

Fig. 2: Example crops from BH-WaterTanks

A. Data Preparation

Each 4K image was cropped into 6 smaller ones of size 1280x1080, without overlap, and then the ones without annotations were removed. Afterwards, 80% of the images from each neighbourhood were used as a training dataset, as the other 20% were used as a test dataset. The same preparation was made for the images of BH-Pools and BH-WaterTanks. These prepared datasets were used for both the SL and DL methods.

B. BH-Pools

The BH-Pools dataset consists of 200 4K images of 8 different neighbourhoods (25 images for each one) and contains 3980 annotated pools. The data preparation step resulted in 655 images designated for training and 160 for testing.

C. BH-WaterTanks

The BH-WaterTanks dataset is made up of 150 4K images of 6 neighbourhoods (25 images for each one) and contains 16216 annotated water tanks. The data preparation step resulted in 608 cropped images designated for training and 148 for testing.

IV. Shallow-Based Approach

A. Methodology

The shallow-learning-based method consists of several different steps, ranging from feature extraction processes to learning and prediction ones, as illustrated in Fig. 3. In contrast to the deep-learning method, the feature extraction strategies had to be selected and designed manually for this approached,

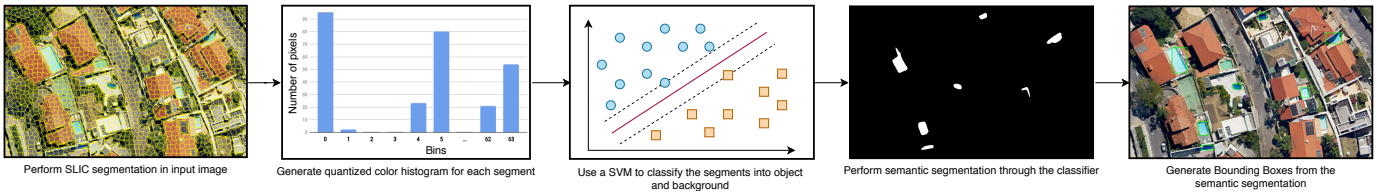

Fig. 3: Diagram illustrating the shallow learning method

TABLE I: BH-Pools and BH-WaterTanks specifications

	BH-Pools	**BH-WaterTanks**
Source	Google Earth Pro	Google Earth Pro
Image resolution	3840x2160	3840x2160
Eye altitude (meters)	330	330
Image bands	RGB	RGB
Number of images	200	150
Number of annotated objects	3980	16216

focusing on what visual features of any given image would better represent the target objects, whereas, in the other, this process is completely automated.

1) SLIC: The first step of the object detection process is to divide the original image into *superpixels* through the SLIC algorithm [18]. Due to the resolution of the images in the two datasets and the average size of the target objects in each of them, the parameters chosen were: 2,000 desirable labels for BH-Pools images and 10,000 ones for BH-WaterTanks. For both datasets, the sigma value was set to 5. The implementation used was the one available in the *scikit-image* toolkit [19].

2) Color Histogram: Moving forward, to each pixel of the image it was attributed an integer number between 0 and 63, accordingly to (1), so that pixels with similar RGB values are allocated in the same group. This way, the number of values describing a given pixel is reduced from three to one, simplifying the classification process down the pipeline.

$$\nu = \frac{\rho}{64} + 4 \times \frac{\gamma}{64} + 16 \times \frac{\beta}{64} \tag{1}$$

where ν = number attributed to pixel
ρ = value of the red channel
γ = value of the green channel
β = value of the blue channel

Afterwards, the histogram of each segment generated by the SLIC algorithm was obtained, based on the new number calculated for every individual pixel, containing 64 bins representative of the color distribution of the given superpixel. This information is stored in a matrix, where the rows represent an individual segment and the columns 0 to 63 contain the number of pixels with that value on the segment.

3) Support-vector Classifier: In the next step, a Linear Support-vector Classifier (SVC) was used to classify the segments of the input image into *positive* or *negative*, using the color histogram obtained in the previous step. The implementation of the SVC used was the one available in the *scikit-learn* [20] toolkit.

Therefore, the batch of images designated for training in each dataset was used for fitting the classifier. It was decided that a given segment is representative of a target object if 50% or more of it is annotated as positive. Moreover, a standardization of the data was performed, in order to center the features around 0 and make them have unit-variance.

Lastly, the trained classifier was applied on the testing batch, in order to perform the semantic segmentation of the target objects in the images. The SVC outputs a confidence score for each given segment, which represents the signed distance of that sample to the hyperplane that separates the two classes. In this case, if said distance is greater than zero, than the segment belongs to the *positive* class.

4) Bounding Boxes: Once the semantic segmentation is completed, another operation is performed in order to obtain the bounding boxes of the detected objects, alongside with their confidence scores. For that, the Multi-dimensional Image Processing packet from the SciPy ecosystem was used to aggregate individual positive segments which are next to each other and that, combined, represent a full object. The bounding box coordinates would then be calculated over such objects. Finally, the confidence score of each bounding box was defined as the average of the confidence scores of the segments that compose it.

B. Hardware and Software Setup

This method was implemented using the Python (version 3.6) programming language. The experiment was performed on a Intel i7-5930X machine with 3.50GHz of clock and 64GB of RAM.

V. Deep-based Approach

A. Framework

The deep learning method for water tanks and swimming pools detection is based on the Faster R-CNN framework [21]. For that, it was used the implementation from the TorchVision package.

Faster R-CNN unifies in one architecture the module consisting of a Region Proposal Network (RPN) and the module responsible for object detection (Fast-RCNN [22]), as illustrated in Fig. 4. In simple terms, a CNN receives the input image and then provides a feature map, which is used by the RPN to indicate to the Fast-RCNN where to look for objects. From this information, a series of Fully Connected layers make

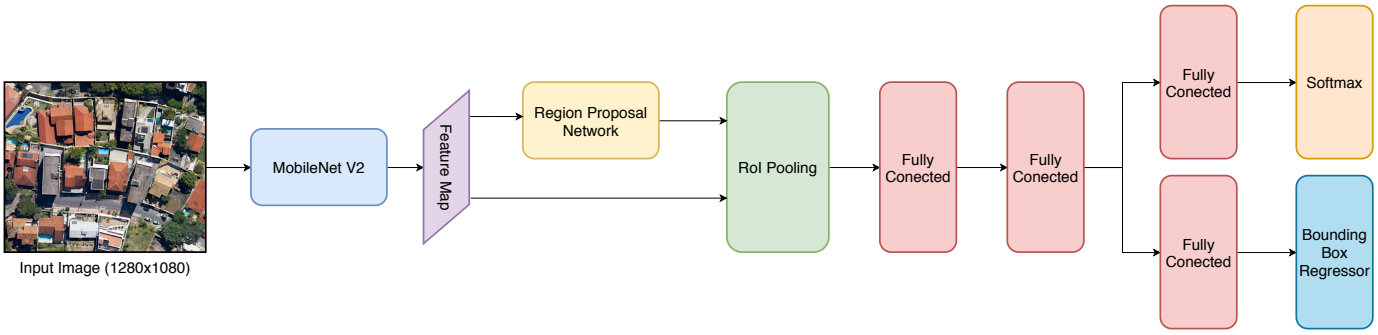

Fig. 4: Simplified diagram illustrating the Faster R-CNN architecture

predictions of the location of bounding boxes in the image and their respective labels.

B. Methodology

The feature extraction network utilized as backbone was the MobileNetV2 [23], pre-trained on the COCO dataset [24]. By using pre-trained weights of a CNN, this DL method performs a fine tuning to improve the training process speed. The MobileNetV2 is a light weight model with improvements suitable for the use of on-device computer vision deep applications, like in mobile devices or any device with low computational power. This choice of backbone helps possible future applications with this deep learning method to not be compromised by a low-budget hardware.

For the training step, it was used the Adam optimizer and a learning rate of 0.0001, chosen empirically. The model was trained for 50 epochs with a batch size of 4 and a random horizontal flip (0.5) transform in the data loader.

Lastly, the testing step was performed on both a GPU-available environment and a CPU-only one. This way, it would be possible to evaluate the possibility of training the network on a high-end device to reduce its training time, but deploying it on inexpensive machines to reduce its cost in regions with budget limitations.

C. Hardware and Software Setup

This method was implemented using Python (version 3.6) with the Pytorch library. The experiment was performed on the Google Colaboratory (Colab) platform, with a configuration composed of an Intel Xeon processor (not specified) with 2.3 GHz of clock, 13 GB RAM and a Tesla T4 GPU.

VI. Results and Discussion

In order to best evaluate the performance of both methods on the proposed datasets, two standard metrics were chosen to be calculated: Average Precision (AP) at an IoU of 0.5, used in the PASCAL VOC challenge, and AP averaged over 10 IoU values, ranging from 0.5 to 0.95 with a step of 0.05, adopted by the COCO challenge [24].

AP takes into account the true predicted positives / total predicted positives (precision) and true predicted positives / total real positives (recall) ratios. An intersection over union (IoU) threshold is defined to determine if a prediction is a true positive or a false positive one. The IoU measures how much the predicted bounding box overlaps with the ground truth bounding box annotation.

The AP with IoU of 0.5 would indicate if the method applied is simply able to correctly detect the target objects in an image, whereas the AP with IoU of 0.5:0.05:95 (the average AP for IoU threshold from 0.5 to 0.95 with a step size of 0.05) would measure how precisely the technique is able to draw the bounding boxes around such objects. Alongside with these metrics, the training and testing times were also measured for each approach, so that it would be possible to compare their usability in real-case scenarios.

All the obtained results of the evaluated methods are presented in Table II. Moreover, Fig. 5 and Fig. 6 show qualitative results for each technique, illustrating the differences between them.

The deep learning method performed significantly better in both datasets compared to the shallow learning one according to the metrics used. On the BH-Pools dataset, the latter was able to detect many of the swimming pools present, but usually only their brightest parts, leading to low IoU values, and also made a lot of False Positive predictions. Conversely, the deep method achieved very high results, scoring an AP at IoU=0.50 equal to 2.27x the one achieved by the other, and an AP at IoU=0.50:0.05:0.95 3.06x the one scored by its opponent, indicating that it not only detected more swimming pools, but also drew more precise bounding boxes around them. Meanwhile, the shallow method obtained extremely poor results on BH-WaterTanks, failing to detect almost every single water tank in it, in contrast to the deep method, which presented satisfactory results. However, the AP at IoU=0.50:0.05:0.95 scored by the deep-learning approach was still lower than half of its score with an IoU=0.50, indicating a greater difficulty in precisely delineating those objects.

Finally, the DL techniques were able to perform the training and testing steps many times faster than the SL ones, including completing the test phase in a matter of minutes, even on CPU-only machines, as opposed to the several hours needed for the others.

VII. Conclusion

In this work, we evaluated and compared two approaches for swimming pool and water tanks detection. The shallow method

TABLE II: Results obtained on the proposed datasets using different object-detection approaches

(a) BH-Pools dataset

Method	AP at IoU=0.50 (%)	AP at IoU=0.50:0.05:0.95 (%)	Training Time (Hours)	Testing Time (Hours) (CPU)	Testing Time (Hours) (GPU)
Shallow learning	40.97	15.96	8.02	1.10	-
Deep learning	**93.13**	**64.79**	**2.70**	**0.08**	**0.01**

(b) BH-WaterTanks dataset

Method	AP at IoU=0.50 (%)	AP at IoU=0.50:0.05:0.95 (%)	Training Time (Hours)	Testing Time (Hours) (CPU)	Testing Time (Hours) (GPU)
Shallow learning	0.13	0.03	19.00	4.20	-
Deep learning	**73.43**	**32.99**	**4.63**	**0.09**	**0.02**

(a) Ground Truth (b) Shallow Method (c) Deep Method

Fig. 5: Example comparing swimming pools detection by the shallow and deep learning methods

(a) Ground Truth (b) Shallow Method (c) Deep Method

Fig. 6: Example comparing water tanks detection by the shallow and deep learning methods

consists of a segmentation using SLIC [18] followed by a classification with SVM [5]. The deep-based approach consists of a Faster-RCNN framework [21] with MobileNetV2 backbone [23]. The methods were trained and then evaluated with our two proposed datasets: BH-Pools and BH-WaterTanks. Both metrics and visual results were compared for a final analysis.

It was clear that the shallow method did not work well for the water tanks detection. Residential water tanks are relatively small objects in satellite images and they could not get a precise segmentation with SLIC, compromising the rest of this approach. On the other hand, the shallow-based approach performed well with the swimming pools detection. Swimming pools are considerably larger and could get a more precise segmentation. However, unusual pools formats, shady areas and blue geometric terrain (e.g. sports courts) were easily misclassified.

The deep learning method worked really well with the water tanks detection, despite their small size. Moreover, it increased the precision of the swimming pool detection. For this reason, we can infer that the deep method used the spatial context more wisely. This method performed better, faster and has been shown of great potential for the task of water tanks and swimming pools detection in high resolution satellite images in practical applications, being a great option even on environments where a GPU is not available to perform the prediction step, given that the network has been trained previously.

Finally, when it comes to the detection of swimming pools, the shallow-learning method can still be a reasonable option if a powerful enough GPU is not available to train the deep-learning network. The method is able to detect many of the swimming pools presented to it in a fraction of the time it would take for a human operator to do so, and does not require a highly computationally complex machine, demonstrating its usefulness in this remote sensing task. Unfortunately, the same cannot be said about residential water tanks detection, due to their smaller size in satellite images.

REFERENCES

[1] S. Kalluri, P. Gilruth, D. Rogers, and M. Szczur, "Surveillance of arthropod vector-borne infectious diseases using remote sensing techniques: a review," *PLOS Pathogens*, 2007.

[2] M. A.Tolle, "Mosquito-borne diseases," *Current Problems in Pediatric and Adolescent Health Care*, vol. 39, pp. 97–140, 2009.

[3] F. Chiaravalloti Neto and H. A. d. S. L. Pereira, "Aedes aegypti na região de são josé do rio preto, estado de são paulo," Master's thesis, Universidade de São Paulo, 1993.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *V. Machine Learning*, pp. 273–297, 1995.

[6] D. Tien, T. Rudra, and A. Hope, "Swimming pool identification from digital sensor imagery using svm," 01 2008, pp. 523–527.

[7] S. K. McFEETERS, "The use of the normalized difference water index (ndwi) in the delineation of open water features," *International Journal of Remote Sensing*, vol. 17, no. 7, pp. 1425–1432, 1996.

[8] M. Kim, J. Holt, R. Eisen, K. Padgett, W. Reisen, and J. Croft, "Detection of swimming pools by geographic object-based image analysis to support west nile virus control efforts," *Photogrammetric Engineering and Remote Sensing*, vol. 77, pp. 1169–1179, 11 2011.

[9] J. A. Saghri and D. A. Cary, "A rectangular-fit classifier for synthetic aperture radar automatic target recognition," in *Applications of Digital Image Processing XXX*, A. G. Tescher, Ed., vol. 6696, International Society for Optics and Photonics. SPIE, 2007, pp. 511 – 521.

[10] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[11] M. Alonso and B. Rodríguez-Cuenca, "Semi-automatic detection of swimming pools from aerial high-resolution images and lidar data," *Remote Sensing*, vol. 6, pp. 2628–2646, 04 2014.

[12] K. Pasupa and W. Sunhem, "A comparison between shallow and deep architecture classifiers on small dataset," *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2016.

[13] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *Journal of Cheminformatics*, vol. 9, no. 1, 2017.

[14] A. McCallum, K. Nigam *et al.*, "A comparison of event models for naive bayes text classification," in *AAAI-98 workshop on learning for text categorization*, vol. 752, no. 1. Citeseer, 1998, pp. 41–48.

[15] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[16] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and qsar modeling," *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.

[17] T. Liu, A. Abd-Elrahman, J. Morton, and V. L. Wilhelm, "Comparing fully convolutional networks, random forest, support vector machine, and patch-based deep convolutional neural networks for object-based wetland mapping using images from small unmanned aircraft system," *GIScience & Remote Sensing*, vol. 55, no. 2, pp. 243–264, 2018.

[18] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," *Technical report, EPFL*, 06 2010.

[19] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, "scikit-image: image processing in Python," *PeerJ*, vol. 2, p. e453, 6 2014.

[20] 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.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.

[22] R. Girshick, "Fast r-cnn," 2015.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2018.

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.