SYMCOMP 2021
Évora, 25-26 March 2021

DEVELOPMENT OF A MACHINE LEARNING MODEL AND A USER INTERFACE TO DETECT ILLEGAL SWIMMING POOLS

C. Coelho[1*], M. Fernanda P. Costa[1], L.L. Ferrás[1] and A.J. Soares[1]

1: Centro de Matemática
Departamento de Matemática
Universidade do Minho
4710 - 057 Braga, Portugal
e-mail: ceciliaeduarda58@gmail.com, mfc@math.uminho.pt, luislimafr@gmail.com, ajsoares@math.uminho.pt
web: https://cmat.uminho.pt

Keywords: Computer Vision, Deep Learning, Object Detection, ResNet, RetinaNet, Matlab

Abstract. *Portuguese legislation states the compulsory reporting of the addition of amenities, such as swimming pools, to the Portuguese tax authority. The purpose is to update the property tax value, to be charged annually to the owner of each real estate. According to MarketWatch, this decade will bring a global rise to the number of swimming pools due to certain factors such as: cost reduction, increasing health consciousness, and others. The need for inspections to ensure that all new constructions are communicated to the competent authorities is therefore rapidly increasing and new solutions are needed to address this problem. Typically, supervision is done by sending human resources to the field, involving huge time and resource consumption, and preventing the catalogue from updating at a rate close to the speed of construction. Automation is rapidly becoming an absolute requirement to improve task efficiency and affordability. Recently, Deep Learning algorithms have shown incredible performance results when used for object detection tasks. Based on the above, this work presents a study on the various existing object detection algorithms and the implementation of a Deep Learning model capable of recognizing swimming pools from satellite images. To achieve the best results for this specific task, the RetinaNet algorithm was chosen. To provide a smooth user experience with the developed model, a simple graphical user interface was also created.*

1 INTRODUCTION

When looking at an image, the human brain has the ability to instantly locate and recognise different objects on that image (object classification). Nowadays, the extraction of information from a digital image or video is becoming a necessity due to the increasing number of applications that rely on it. For instance, face recognition [1], autonomous driving [2], and pedestrian detection [3]. In the field of Computer Vision (CV), this challenging problem is called object detection [4].

In object detection, the goal is to locate and classify objects in an input image by outputting bounding boxes for each object, and a class label for each box. That is, given an input image X, the output is expected to be in the form (X, Y) where Y is an array containing the class label and the box edge's coordinates.

An image is a visual representation of a real-life object. Each image is made from small boxes called pixels. Each pixel's colour is represented by an RGB (Red, Green, Blue) number from a colour space where each colour's intensity is in the range of 0 to 255 [5]. This colour space is used because it mimics the way the human eye works (the human eye has cone cells located in the retina that are able to identify three colours (red, green and blue) and their combinations [6]).

Computers store RGB images in the form of three matrices (or channels), one for each main colour, where each pixel is converted to a number between 0 and 255 for each channel. When three colour matrices are blended, by adding the values of each pixel, the image is computed.

Traditionally, the CV techniques used for object detection are based on feature descriptors. These have the ability to identify/extract features for each object class, a small list of interesting points that describe an image's content (points, edges or colour variations) and that allow for image classification.

To accomplish this, several CV algorithms are used, namely edge detection, corner detection or threshold segmentation [7]. For example, during training for image classification, all the detected interesting points (the most important features) form a definition's list of the object class. When classifying new images, if a significant number of points belong to the definition list, then the image is classified as belonging to that class (for more information on feature descriptors refer to [8]). If an image contains several objects (classes) to be identified, feature extraction becomes computationally expensive. This problem will be aggravated with the increasing number of features per class. To prevent from this, the user needs to go through a trial and error process to determine which features best describe the different classes [9].

A solution to this problem is given by Deep Learning. Deep Learning is a class of Machine Learning algorithms that uses multiple layers to progressively extract higher level features from the raw input [10]. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as letters or faces. Therefore, Deep Learning is the technique of choice for detecting specific

objects in images. The word *deep* in Deep Learning refers to the number of layers through which the data is transformed (figure 1).

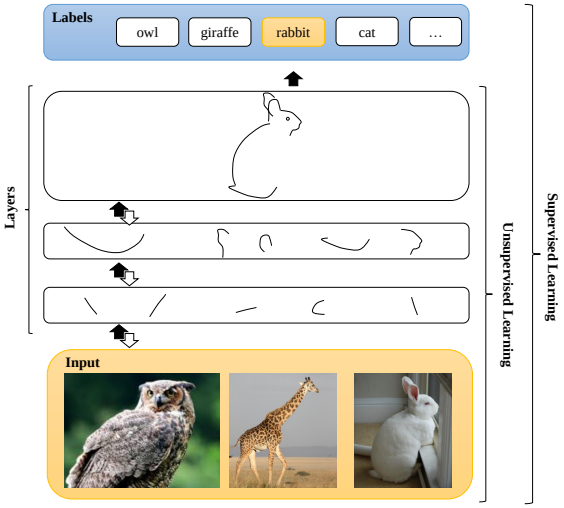


Figure 1: Representing Images on Multiple Layers of Abstraction in Deep Learning.

With the use of Deep Learning methods, the computer receives several images, in which the classes present have been annotated, and the DL method is able to discover the most descriptive features of each class by itself, acting as a human in feature descriptors. This allows for better performance than the traditional methods with the disadvantage of the user not being able to know which features were given higher emphasis (it is used as a black box procedure).

The main objective of this work is the detection of swimming pools by using satellite images, and Deep Learning seems to be the ideal technique to achieve this.

Starting from the work developed in [11], we present the development (training and testing) of a Deep Learning model able to detect swimming pools with high accuracy (Section 2). The model is complemented with a Graphical User Interface (GUI) that makes the process of detecting swimming pools (using satellite images) more user friendly. For instance, to be used by a Government entity to detect this type of illegal constructions (Section 3). The works end with the Conclusions (Section 4).

2 DEEP LEARNING MODELS

Deep Learning is a part of a broader family of Machine Learning methods based on Artificial Neural Networks (ANN or NN) with feature learning (also known as representation learning). ANN models may contain many layers of neurons. These models are trained using an extensive amount of labelled data. This can be described as mathematical models of the human brain with the purpose of processing nonlinear relations between

inputs and outputs. A representation of an ANN can be viewed as a combination of several neurons (perceptrons) with an input and a weight attributed to that input. An activation function that decides to activate or not that neuron, and, an output.

If there are multiple inputs to the network $(x_0, x_1, x_2, ..., x_n)$, each is multiplied by a weight (synapse) $(w_0, w_1, w_2, ..., w_n)$. In order to generate a result, the products are summed and fed to an activation function. This determines whether a neuron is activated or not depending on a certain threshold. Based on the final result, the next layer of neurons may be activated. This leads the NN to learn complex connections between input and output, this process is often designated by training.

An ANN can be built by stacking numerous perceptrons into layers, as seen in figure 2. Thus, the most basic ANN has three components: an input layer that receives the provided data to the NN; a hidden layer that has the task of discovering relationships between the features in the input; and an output layer that produces the result of the given inputs [12].

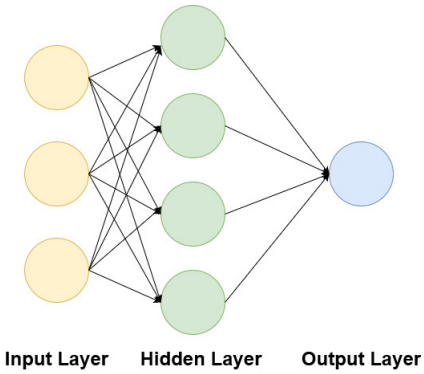


Figure 2: Composition of an Artificial Neural Network with a single-hidden layer.

Deep Learning is introduced when there is more than one hidden layer, meaning that the difference between a single-hidden layer ANN and Deep Learning lies in the depth of the model [10].

There are several Deep learning networks/models (based on object detection algorithms) in the literature: Convolutional Neural Networks, Region-based Convolutional Neural Networks, Fast Region-based Convolutional Neural Networks, Faster Region-based Convolutional Neural Networks, You Only Look Once (YOLO), Single-Shot Detector (SSD) and RetinaNet. These algorithms use different techniques to detect objects, that are based on a trade-off between accuracy and computational cost.

These techniques were tested with images of swimming pools, and RetinaNet was the model that provided the best trade-off between accuracy and computational cost. The main reason was the fact that the satellite images for detection may have different sizes and a huge amount of background. The RetinaNet algorithm is a single-stage detector that

has two new improvements over YOLO and SSD: Focal Loss (that corrects the imbalance between the amount of background and the object to be detected) and Feature Pyramid Network (to overcome the problem of different scales, a possible solution is to give the network several copies of the same image at different scales (resembling a pyramid)).

2.1 Training and testing of RetinaNet

2.1.1 Training

The dataset (a set of images) used for this work can be found on *Kaggle*, an online community where users can find and publish datasets, explore and create Data Science and Machine Learning models, and enter competitions. The chosen dataset has 3748 training and 2703 test satellite images with bounding box annotations and labels for cars (denoted by 1) and swimming pools (denoted by 2) classification. The images are in RGB and have 224×224 pixels. In figure 3 are represented four image examples from the training set.

Figure 3: Four training images of the chosen dataset.

The dataset was adapted for detection of swimming pools, only. A Python script was used to create the new dataset, that has 1993 training images.

The Deep Learning models were trained using 500 iterations and batch size of 1. Since the training dataset, from *Kaggle*, has 1993 images and a batch size of 1 was chosen, the dataset will be divided into 1993 batches, each with one image. This means the model

weights will be updated after each batch. Three different backbone architectures were used in order to analyse which would give better results. Therefore, using the training dataset with 1993 images, three separate models were trained, each with a different architecture, namely: ResNet50, ResNet101 and ResNet152.

2.1.2 Testing

The chosen dataset for testing has 2703 images without labels. A Python script was developed to create labels, resulting in a testing dataset that has 2703 satellite images with a total of 620 swimming pools, distributed across a total of 524 images.

2.2 Results and discussion

To analyse the performance of each RetinaNet model, a confusion matrix was computed (matrix with values TN, TP, FN and FP). This matrix allows a better understanding of the types of errors a model is making. Based on the confusion matrix it is possible to determine the following measures: accuracy, precision, specificity and the Matthews Correlation Coefficient (MCC) of the model. Additionally, two error types, Type I and Type II, can also be computed to deepen the understanding of the model's fouls [13].

- **True Positive (TP):** The expected and the predicted values are positive. In this case, the image has a swimming pool in a certain location and the model accurately identified it. For each swimming pool, correctly detected, is added 1 to the TP value (note that an image with 3 correctly classified objects contributes with 3 to the TP value);

- **True Negative (TN):** The expected and the predicted values are negative. In this case, the image has no swimming pools and the model doesn't classify any part of the image as one. For each image, correctly classified as not having pools, is added 1 to the TN value;

- **False Positive (FP):** The predicted value is positive but the expected is negative. In this case, the image has no swimming pools but the model identifies a part of the image as being one. For each pool detected but not being one, is added 1 to the FP value;

- **False Negative (FN):** The predicted value is negative but the expected is positive. That is, the image has swimming pools but the model isn't able to detect any. For each image, wrongly classified as not having pools, is added 1 to the FN value.

The results obtained with the three models are summarised in table 1.

It can be seen that every model's accuracy is high and remarkably close to each other, indicating that the classifiers are correct most of the times (approximately 0.95 of accuracy). All RetinaNet models have high precision (above 0.95) indicating that an image

	TP	TN	FP	FN	Accuracy	Precision	Specificity	MCC	Type I	Type II
ResNet50	471	2200	18	131	0.9472	0.9631	0.9919	0.8380	0.0081	0.2176
ResNet101	480	2285	7	133	0.9522	0.9626	0.9969	0.8519	0.0031	0.2170
ResNet152	503	2241	6	111	0.9591	0.9882	0.9973	0.8766	0.0027	0.1808

Table 1: Confusion matrix values and computed evaluation quantities for each of the trained models.

labelled as positive is truly positive, as verified by the small number of false positives reported in the table. The model using a ResNet152 backbone architecture is capable of better recognising the class in study (lowest number of false negatives). This is also corroborated by the highest sensitivity value obtained for this model. The three models have high specificity, being above 0.99 for all three models, which means the models predict negative examples well (images without swimming pools). The MCC is higher for the model built on top of a ResNet152, demonstrating, out of the three models, the effectiveness in classifying both classes (presence or absence of pools in an image). Moreover, the ResNet152 has the smallest Type I and Type II errors suggesting that the predictions are correct most of the times.

A closer look at the false positive and false negative classifications, exposes the type of difficulties faced by each model and the kind of objects that compromise the swimming pools detection:

- Every model detects blue rectangles, that may be canopies, for instance, as swimming pools;
- All models have difficulties in detecting empty pools;
- Both ResNet50 and ResNet101 models identify blue basketball fields as pools;
- The model that uses a ResNet50 as backbone architecture wrongly detects circular blue looking canopies.

From this analysis one may conclude that the model using a ResNet152 backbone architecture outperforms the models using ResNet50 and ResNet101. Furthermore, all three models have similar computational cost.

3 GRAPHICAL USER INTERFACE

A Graphical User Interface was developed in order to make the process of detecting swimming pools (using satellite images) user friendly. For instance, to be used by a Government entity to detect this type of illegal constructions.

This GUI was created using the *Electron* framework. It was built using *HTML*, *CSS* and *JavaScript* in connection to the RetinaNet model, with a backbone architecture built on top of a ResNet152, python script and auxiliary scripts. The desktop application's main window is shown in figure 4.

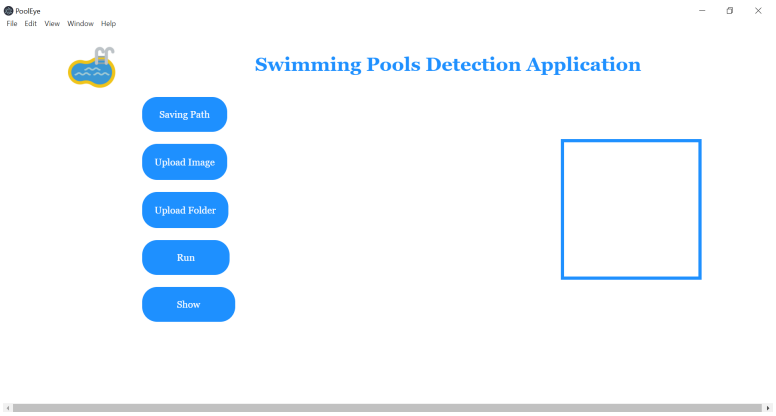


Figure 4: Desktop application main window.

As seen in figure 4, on the right side there is a blue frame where the images in which swimming pools were detected will be shown. On the left side, five buttons with different functionalities are displayed. A saving path, to save the information. One can upload an image, several images or a folder (see figure 5).

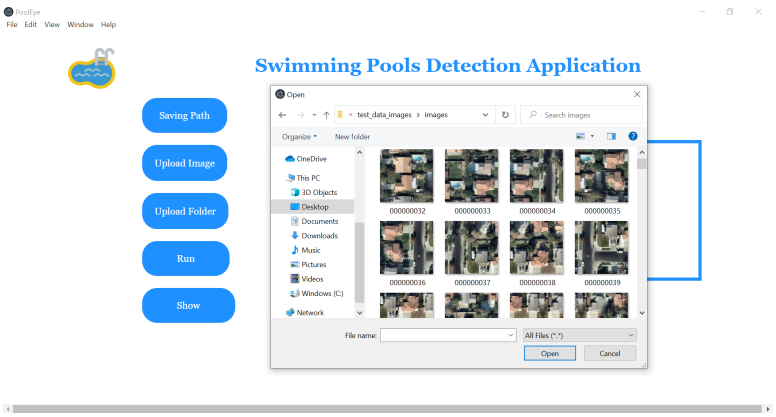
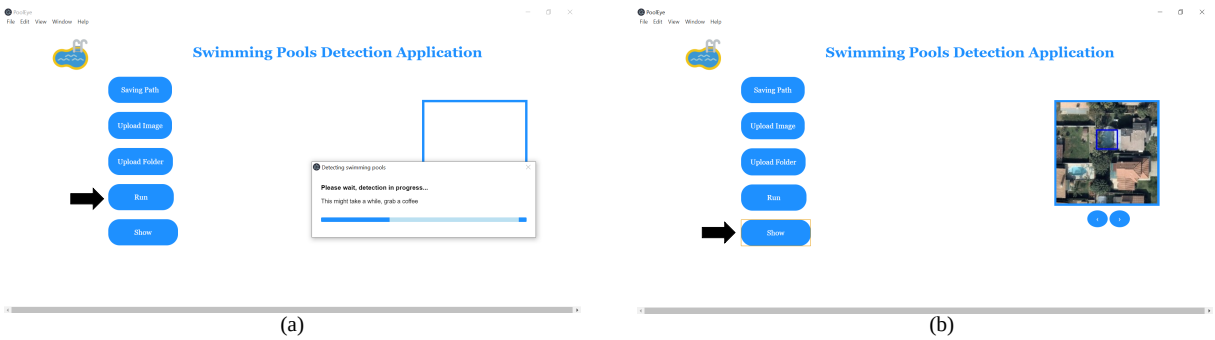


Figure 5: *Upload Image* button action.

After uploading a set of images, the RetinaNet model detects swimming pools by simply clicking the button *Run* (see figures 6 a and b).

When *Show* is clicked, a python script that draws the bounding boxes in the images using the CSV file created by the model is run. Only one box per image is shown. If the user wants to see more images with bounding boxes, it should click the arrows under the frame where the images appear.



(a) (b)

Figure 6: (a)*Run* button action; (b) *Show* button action.

4 CONCLUSIONS

The goal of this work was the detection of swimming pools in satellite images. To achieve this, the RetinaNet algorithm was used. The final considerations and main conclusions are:

- Three RetinaNet models were trained, each built on top of a differing backbone architecture, using a dataset from *Kaggle*. After the training, the models were tested using the test set composed of 2703 images with 620 swimming pools distributed by 524 images;
- The results were analysed using six performance metrics, that are computed based on confusion matrices. From these results, one may conclude that the model using a ResNet152 backbone architecture outperforms the models using ResNet50 and ResNet101 (for detecting swimming pools);
- A Graphical User Interface was developed in order to make the process of detecting illegal swimming pools using satellite images user friendly.

5 ACKNOWLEDGMENTS

The authors acknowledge the funding by FCT - Fundação para a Ciência e a Tecnologia, through projects UIDB/00013/2020 and UIDP/00013/2020, and, the support by CMAT - Centro de Matemática e Aplicações da Universidade do Minho.

REFERENCES

[1] Yang, Z., Nevatia, R. “A multi-scale cascade fully convolutional network face detector”, *Proceedings - International Conference on Pattern Recognition*, pp. 633-638, 2016.

[2] Chen, X., Ma, H., Wan, J., Li, B., Xia, T. "Multi-view 3D Object Detection Network for Autonomous Driving", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, pp. 6526-6534, 2017.

[3] Wojek, C., Schiele, B., Perona, P. "Pedestrian detection: An evaluation of the state of theart", *IEEE transactions on pattern analysis and machine intelligence, Vol.* ***34***, pp. 743-61, 2011.

[4] Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan D. "Object detection with dis-criminatively trained part-based models", *IEEE transactions on pattern analysis and machine intelligence, Vol.* ***32***, pp. 1627-45, 2010.

[5] Singh, H. *Practical Machine Learning and Image Processing For Facial Recognition Using Python*, Apress, Berkeley, CA, 2019.

[6] Ryan, S.J., Wilkinson, C.P., Sadda, S.R., Wiedermann, P. *Retina*, Elsevier, 6th edition, 2017.

[7] Salahat, E., Qasaimeh, M. "Recent advances in features extraction and description algorithms: A comprehensive survey", *Proceedings of the IEEE International Conference on Industrial Technology*, pp. 1059-1063, 2017.

[8] Awad, A.I., Hassaballah, M. *Image Feature Detectors and Descriptors: Foundations and Applications*, Springer International Publishing, 1st edition edition, 2016.

[9] Ghojogh, B., Samad, M.N., Mashhadi, S.A., Kapoor, T., Ali, W., Karray, F., Crowley, M. *Feature selection and feature extraction in pattern analysis: A literature review*, 2019 (Online Available: http://arxiv.org/abs/1905.02845).

[10] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y. *Deep learning*, MIT press, Cambridge, 2016.

[11] Coelho, C. "Machine Learning and Image Processing", Master's thesis, 2020 (to appear in Repositorium, University of Minho, http://repositorium.sdum.uminho.pt)

[12] Feng, J., Lu, S. "Performance Analysis of Various Activation Functions in Artificial Neural Networks" *Journal of Physics: Conference Series, Vol.* ***1237***, 2019.

[13] Powers, D. "From precision, recall and f-measure to roc., informedness, markedness & correlation", *Journal of Machine Learning Technologies*, Vol. **2**, pp. 37-63, 2011.