A Platform for Swimming Pool Detection and Legal Verification Using a Multi-Agent System and Remote Image Sensing

Héctor Sánchez San Blas, Antía Carmona Balea, André Sales Mendes, Luís Augusto Silva*, Gabriel Villarrubia González

Expert Systems and Applications Lab—ESALAB, Faculty of Science, University of Salamanca - Plaza de los Caídos s/n, 37008 Salamanca (Spain)

Received 30 September 2021 | Accepted 21 October 2022 | Published 11 January 2023

Abstract

Spain is the second country in Europe with the most swimming pools. However, the legal literature estimates that 20% of swimming pools are not declared or irregular.The administration has a corps of people who manually analyze satellite or drone images to detect illegal or irregular structures. This method is costly in terms of effort and time, and it is also a method based on the subjectivity of the person carrying it out. This proposal aims to design a platform that allows the automatic detection of irregular pools. Using geographic information tools (GIS) based on orthophotography, combined with advanced machine learning techniques for object detection, allows this work. Furthermore, using a multi-agent architecture allows the system to be modular, with the possibility of the different parts of the system working together, balancing the workload. The proposed system has been validated by testing it in different towns in Spain. The system has shown promising results in performing this task, with an F1-Score of 97.1%.

Keywords

Deep Learning, GIS Detection, Illegal Pools Detection, Pool Aerial Recognition.

DOI: 10.9781/ijimai.2023.01.002

I. Introduction

Cartography deals with the conception, production, dissemination, and study of maps that have undergone a good evolution in the last decade. Until a few years ago, the processes of cartographic revision and, mainly, those aimed at calculating the fiscal area have been carried out manually. Specifically, these processes required significant investments in airplanes or helicopters, making the processes more expensive. Because of this, municipalities are unable to perform cartography surveys frequently. One of the most current research fields is to investigate technological capabilities for local authorities to perform detailed surveys of the territory of municipalities at a reasonable cost.

One of the most important milestones that cartography allows is refining fiscal data or verifying geographic information that forms the basis for local taxes. One aspect taken into account in an audit process by the municipalities is the size of the plots and the construction of private swimming pools. Spain, for tax purposes, needs to take into account a private swimming pool on a plot of land. However, this process is not easy, as there are many aspects to consider with the mapping, such as location, time of day when we take the image, how close or far away the image is, or obstacles. In addition, the result of this process must be evaluated by a person, a costly task that depends on the individual's subjectivity, as he or she is the one who must detect the structures built in an area. Depending on the image's characteristics relative to the cartography, these characteristics can lead to errors in determining the existence of such structures.

Knowing the precise location of swimming pools is crucial for tax collection purposes and ecological reasons. It is vital to know the pools with large volumes of water since, in the event of a fire in a nearby area, firefighting teams can make use of it [1]. Another major issue that has caused concern in recent years is controlling and managing limited and indispensable water resources such as drinking water. In particular, the construction of swimming pools in summer impacts the demand for water the municipal supplier needs to prepare. Therefore, it is understandable that the local government asks for an extra contribution from pool owners in a tax. A third eminent problem is mosquito-borne diseases, which affect many people worldwide, mainly in tropical and sub-tropical countries such as Brazil. Pool water in unoccupied homes may not be adequately filtered, and rainwater accumulated along with decaying leaves may not be removed from the pool, providing an ideal habitat for mosquitoes to live and breed [2].

With the proliferation and evolution of UAV, particularly the use of RPAS, the process of pool detection is much faster and more cost-effective than it was a decade ago [3]. However, the numerical quantification of pools from a large amount of data produced during the photographic session is a time-consuming task performed by manual procedures. The different satellite images obtained have different properties, making it challenging to develop different algorithms to detect pools. These images differ in scale, resolution, sensor type,

* Corresponding author.
E-mail address: luisaugustos@usal.es

orientation, quality, and ambient illumination conditions. In addition to these diÿculties, buildings may have complicated structures and could be hidden by other buildings or trees. Both structural and deterministic clues must be taken into account when constructing the solution. Up-to-date and accurate data are essential for municipalities. Therefore orthophotography makes it slightly easier to recognize outdoor pools. Among the existing possibilities to solve this problem is the use of satellite imagery in combination with machine learning techniques [4]

The main problem in these processes is to relate the images collected by satellites or drones with a pool detection system and the corresponding verification of the same within the databases of local systems. The solution for the detection problem followed the advances in the literature of machine learning algorithms, precisely Deep Learning was used, which is a class of Machine Learning algorithms. This type of algorithm uses multiple layers to progressively extract features from the input images [5].

This article presents a novel platform that allows the automatic detection of swimming pools by acquiring images from different sources. For this purpose, the system can apply different algorithms to determine the proliferation of illegal swimming pools in a territory by accessing local control databases. In order to make the system scalable, robust, and able to merge the information coming from several neural networks, this case study uses a multi-agent architecture. A multi-agent system makes it possible to build a dynamically reconfigurable platform. It also allows the resources and capabilities of the system to be distributed evenly among the different elements of the system. In this way, problems that often occur in centralized systems, such as bottlenecks or recurring access to critical resources, are eliminated. In addition, the system's eÿciency in retrieving, filtering, and coordinating information is improved.

The paper is organized as follows: Section II focuses on an in-depth review of the state-of-the-art literature on Deep Learning algorithms used for object detection in images; Section III conducts a review of works similar to the proposed one related to pool detection and pool legality verification; Section IV describes the architecture of the proposed system; On the other hand, Section V will explain each of the blocks that make up the system; Section VI, will show the case study carried out with the results obtained; Finally, the conclusions obtained are in Section VII.

II. Background

Deep Learning is a field that focuses on algorithms based on artificial neural networks. Thanks to deep learning, intelligent document processing (IDP) can combine various AI technologies to classify photos automatically and describe the various elements of images. With their multilevel structures, deep learning models are beneficial for extracting detailed information from input images. Convolutional neural networks can also drastically reduce computational time by taking advantage of the GPU for computation, something that many networks do not use. In the field of object identification in images, two methods stand out: Region Proposal algorithms and regression object detection algorithms.

The first method is to find out in advance the possible locations of the target to detect in the figure. This method can ensure that the highest recovery rate is maintained when fewer windows are selected. Suppose an image is an input and, after a series of convolutions and clustering in the backbone, a feature map of size M × N is obtained, corresponding to the original image's division into M × N areas. The center of each area from the original image represents the coordinates of a pixel in this feature map.

Region Proposal Algorithms find whether the k anchor boxes corresponding to each pixel contain a target. The network must learn to classify the anchor boxes as background or foreground. It must calculate regression coeÿcients to modify the foreground anchor box's position, width, and height. Within these classifiers, we find algorithms such as R-CNN [6], Fast R-CNN [7], Faster R-CNN [8] and MASK-CNN [9].

Of the algorithms mentioned above, Mask R-CNN stands out. This algorithm extends Faster R-CNN and works by adding a branch to predict an object mask in parallel with the existing branch for bounding box recognition. The critical element of Mask R-CNN is pixel-to-pixel alignment, which is the main missing piece in Fast/Faster R-CNN. Mask R-CNN adopts the same two-stage procedure with an identical first stage (which is RPN). In parallel to predicting the class and box offset in the second stage, Mask R-CNN also outputs a binary mask for each RoI. This method is in contrast to most current systems, where classification depends on mask predictions. In addition, Mask R-CNN is simple to implement and train thanks to the faster R-CNN framework, which facilitates a wide range of flexible architecture designs. Moreover, the mask branch only adds a small computational overhead, enabling a fast system and rapid experimentation. Fig. 1 shows a visual example of the segmentation performed by the algorithm.

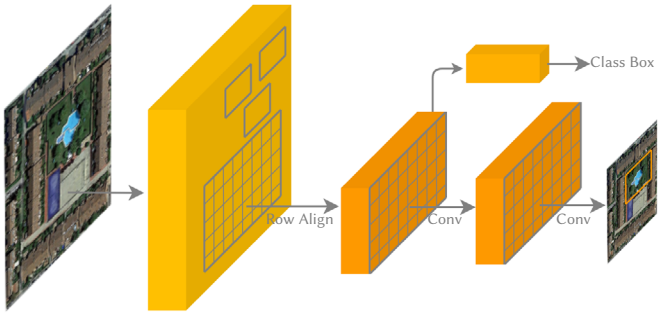


Fig. 1. Mask R-CNN Framework for Instance Segmentation.

Along the same lines as the above, we find Detectron [10], Facebook AI Research's (FAIR) software system that implements state-of-the-art object detection algorithms, including Mask R-CNN. Detectron aims to provide a high-quality, high-performance codebase for object detection research. The design of this algorithm is flexible to support the rapid implementation and evaluation of new research. However, the most successful version is Detectron2 [11], being an enhancement of Detectron. The significant difference between versions is that the latest version is a more modular, flexible, and extensible design, allowing much faster training on GPU-enabled computers. Detectron2 includes high-quality implementations of the most advanced object detection algorithms, such as DensePose, pyramid networks with panoptic features, and numerous variants of the pioneering Mask R-CNN family of models, also developed by FAIR. The creators of the algorithm reproduce the ResNet-50-FPN baselines together with the Scale Jitter algorithm.

A. YoloV4

The above algorithms use detection as a classification problem, that is, first, the algorithm generates object proposals, and then these proposals are sent to the classification/regression regions. However, some methods approach detection as a regression problem based on a similar operation. The YOLO (You Only Look Once) and SSD (Single Shot Detector) algorithms stand out within this field.

The SSD [12] algorithm strikes a good balance between speed and accuracy. SSD runs a convolutional network on the input image only once and computes a feature map. It then runs a small 3×3 convolutional kernel on this feature map to predict bounding boxes

and classification probability. SSD also uses anchor boxes in various aspect ratios, similar to Faster-RCNN, and learns the offset instead of learning the box. To handle scale, SSD predicts bounding boxes after multiple convolutional layers. As each convolutional layer operates at a different scale, it detects objects of various scales. Fig. 2 shows an example of how the SSD algorithm works.

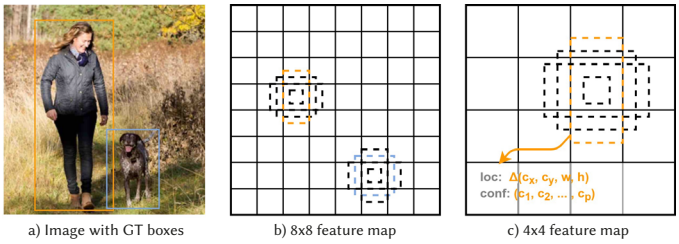


Fig. 2. SSD framework example.

For YOLO [13], detection is a simple regression problem that takes an input image and learns the class probabilities along with the coordinates of the bounding box. YOLO divides each image into an S x S grid, and each grid predicts N bounding boxes and their confidence. The confidence reflects the accuracy of the bounding box and whether the bounding box contains an object, regardless of the class. YOLO also predicts the classification score of each bounding box for each class in training. It can combine both classes to calculate the probability that each class is present in a predicted box. Thus, the algorithm predicts a total of SxSxN bounding boxes. However, most of these boxes have low confidence scores, so if we set a threshold, for example, 30 percent confidence, we can eliminate most of them, as shown in Fig. 3.

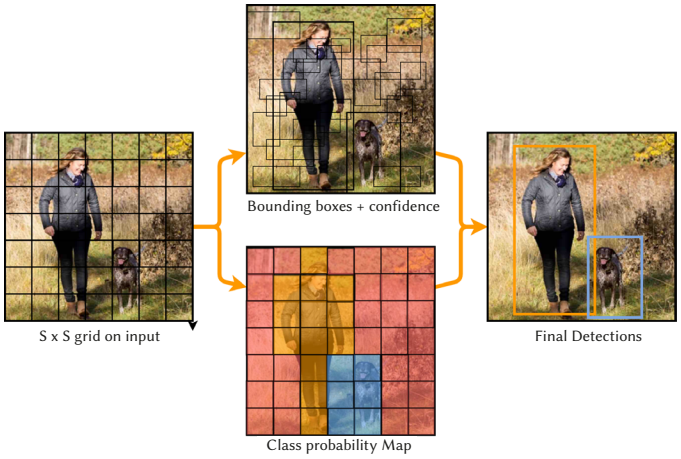


Fig. 3. YOLO workflow example.

YOLO is a algorithm faster than all other detection algorithms, allowing it to run in real-time. Another key difference is that YOLO sees the entire image at once rather than looking only at the proposals of a region generated in previous methods. Thus, this contextual information helps to avoid false positives. However, one of the limitations of YOLO is that it only predicts one class type in a grid, so it has diÿculties with tiny objects. There are several versions of YOLO such as YOLOv2 [14], YOLOv3 [15], and YOLOv4 [16]. There are also variations of this latest version, adapting it to the context of use and improving its results [17], [18].

III. Related Works

The related works listed in this section are recent and present evidence and contributions relevant to the area. However, they are limited and do not have as their primary focus the analyzed points necessary for controlling and managing pool legality verification based on object detection and image classification.

Thus, we start the comparisons with the work proposed by Tien [1] which introduces a Support Vector Machine (SVM) technique to classify small area water bodies, namely swimming pools. The work focus on locating and subsequently using the water source for emergency services in fighting bushfires in urban areas of Australia. First, satellite images were processed, and then the images were segmented.

In Galindo's [19] work, the authors proposed a detection system to locate full pools during drought periods in order to alert local authorities. The work applies color analysis for water and approximate segmentation and active contouring techniques to refine the shape of the pools. This algorithm with satellite imagery and aerial imagery has a 93% of success rate.

Kim *et al.* [20] shows a concern regarding neglected pools in California, as is the case in other countries [2]. These countries seek economic ease in locating pools for ground survey and mosquito control. This research focused on using high spatial resolution (VHR) satellite imagery and image Pansharpening techniques, normalized difference water index, and geographic object-based image analysis. In this way, the authors developed a geographic information system (GIS) database of pool locations. The system demonstrated that VHR imagery could produce a GIS database of pools with high accuracy of 94%.

Rodriguez-Cuenca and Alonso [21] presented in their paper a semi-automatic determination of the location of swimming pools in urban areas from aerial images and LIDAR sensor data. All this without the need for specific training, added indices combined with Dempster-Shafer theory to determine pool locations. The method presented an accuracy rate of 99.86%.

Ferner *et al.* [22] study the detection of homes with swimming pools using convolutional neural networks (CNNs), applied to load heat maps constructed from load profiles. The author presents only a small dataset. Still, the results show that using CNNs, privacy can be broken automatically, without using manual feature generation, which requires a lot of time. The method outperforms the nearest neighbor classifier compared by the authors. Although this work does not use satellite images or aerial images, it is related to the current project as it uses a Convolutional Neural Network (CNN).

Domozi *et al.* [23] also uses a Convolutional Neural Network applied to aerial images by drones. The proposal makes use of an application called Pix4D. The current development allowed to automatically detect the ponds and quantify them employing a neural network on the orthophotos.

Lima *et al.* [24] presented in their article an application integrated into municipal systems, determining the location of swimming pools in urban areas from a GIS system. The application includes the use of deep learning algorithms to detect swimming pools and focus on existing municipal information, with validation in the region of Braga in Portugal. The method has demonstrated an accuracy of 83%, with a significant number of illegal or unknown pools detected.

In this section, we have analyzed different works carried out in the areas involved. To provide a more detailed and clear analysis all the methods explained in this section are summarized in Table I.

IV. Proposed Architecture

The proposed architecture of this section aims to provide a solution to the problem while adapting and introducing new functionality without affecting the other parts of the system. In order to achieve the objective of this research work, the detection and automatic verification of the legality of swimming pools built in private spaces, it

TABLE I. Related Works

Work	Year	Algorithm	Images	Accuracy
Tien *et al.* [1]	2007	Support Vector Machine	Satellite Image	-
Galindo *et al.* [19]	2009	Color and Segmentation Analysis	Satellite Image	93%
Kim *et al.* [20]	2011	Pan-sharpening	Satellite Image	94%
Rodríguez-Cuenca [21]	2014	Support Vector Machine	Aerial Image + LiDAR	99.86%
Ferner *et al.* [22]	2019	CNN / 5-Nearest neighbors	-	68.5% and 71.9%
Domozi *et al.* [23]	2019	R-CNN	Drone Imagery	99%
Passos *et al.* [2]	2020	Faster R-CNN / ResNet-101-C4	Drone Imagery	74%
Lima *et al.* [24]	2021	Faster R-CNN	GeoTIFF Images	83%

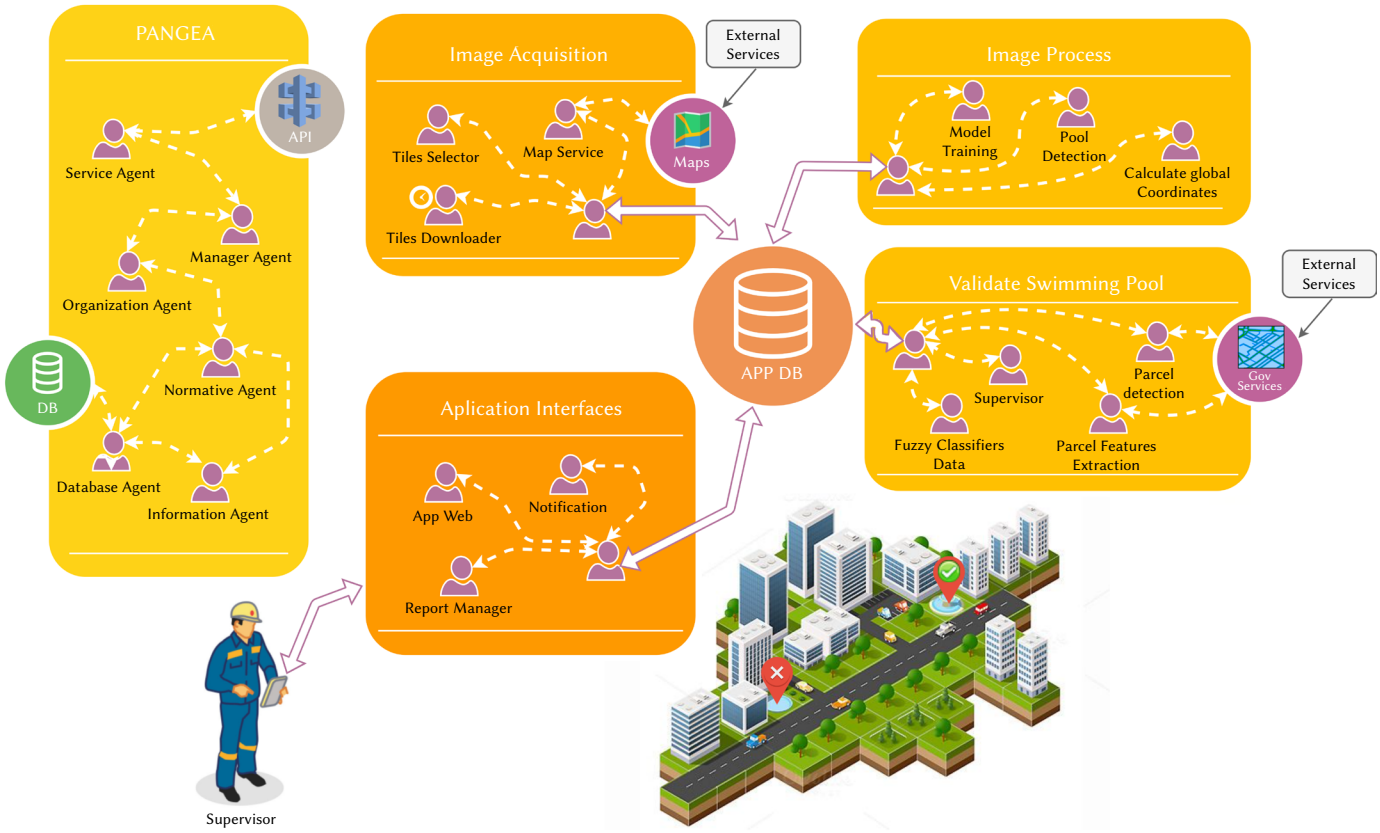


Fig. 4. System Architeture.

is necessary to have an architecture with well-defined characteristics so that the system can operate correctly. For this purpose, an architecture is modeled based on virtual agents, where each of the system's agents works individually to achieve a common goal. One of the primary needs that have led to the system's design using this architecture has been the need to design a distributed system to perform the different tasks of image extraction, model training, and classification in a scalable way. Another essential feature of this architecture based on virtual agents is that another can replace one agent without affecting the rest of the system.

PANGEA works [25] have used a base of the proposed architecture. PANGEA allows the different agents to adapt to the computational needs of the system dynamically. Another advantage of this architecture is that it allows the system's services to rise on demand. When an agent joins the platform, it must communicate which services are available and can offer to other entities. In Fig. 4, the architecture designed for the case study presented in this article is shown. In addition, the agents organized in virtual organizations that make up the system can be seen.

Compared to other existing systems such as SPADE, Python's Library, JADE, or osBrain, the PANGEA multi-agent platform can create virtual organizations. These virtual organizations allow the creation of visual representations and modeling of any system. In addition, being an open-source system, it will be available for use by other researchers who want to replicate the system in the future.

The following is a list of the organizations, the functionality, and the agents of each one:

Image Acquisition: This virtual organization is responsible for obtaining the satellite images that the other organizations will use to detect the pools. One of the main features of this organization is that it allows the use of different map sources for downloading the tiles. For this purpose, the Tiles Selector agent is in charge of calculating the tiles to be downloaded for the zones pre-selected by the system users. Analyzing system needs was determined to perform periodic downloads and checks to detect new pools and pools installed temporarily in the summer. For this purpose, the Tiles Downloader agent can program the downloads according to the indicated periodicity.

Aplication Interface: This organization is the one that allows the information generated by the system to be understandable by humans. This organization serves as an interface between the system and the system's applications. Applications that have access to this organization can access or generate data in the system. In this case, the

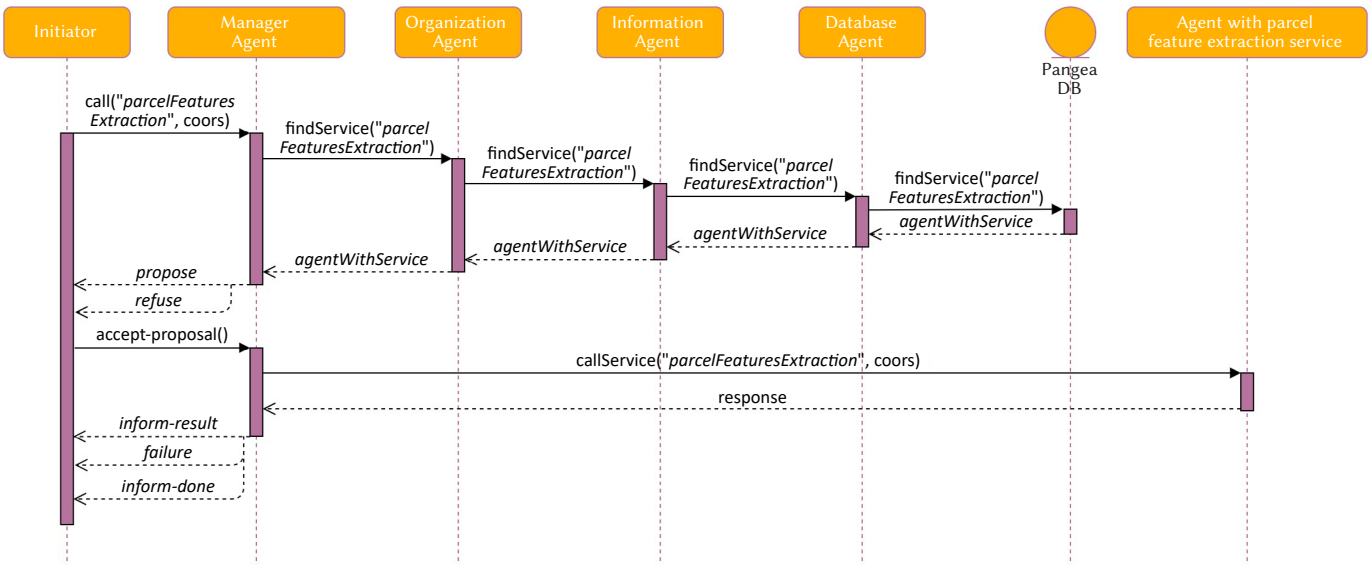


Fig. 5. Sequence Diagram from the System.

human agent with the system can define which zones to inspect and review detection results, alerts, or detection reports.

Image Process: This organization aims to carry out tasks related to image processing and tile pool detection. For this purpose, it has agents with the Model Training agent responsible for the training and retraining the models used with new images labeled by the user. The Pool Detection agent can detect the new tiles pending classification downloaded into the system and proceed to their classification using the pre-trained models. Finally, in this organization, the Calculate Global Coordinates agent's main task is to convert the coordinates of each of the pools detected by the Pool Detection agent to the global system, taking into account the relative coordinates and the zoom level of the tile analyzed.

Validate Swimming Pool: This organization aims to detect which pools are legally registered. To do so, this organization uses a governmental Webservice to obtain the data associated with each of the plots. The information returned by the service is whether any swimming pool is registered. The Parcel Detection agent, using the external services, has the objective of assigning the identifier of the parcel to each of the detections. The parcel uses this identifier Features Extraction agent, which is in charge of obtaining if the pool is registered as a legal form in the parcel. In this way, the system can determine if the detected pool is correctly registered. The Fuzzy Classifiers Data agent is responsible for detecting duplicate pools in adjacent tiles or at a minimal distance.

Pangea Multi-Agent System Organization: In this organization there are the minimum agents necessary for the Pangea system to work. The main milestone of this organization is to carry out the tasks of organizing the virtual organizations and the communication between the agents responsible for each organization. Below the agents that are part of this organization are described:

- *Service Agent:* This agent can expose functionality through web services such as a communication interface between the organization's external agents and those of itself. This interface allows the creation of agents independent of programming language or execution environment.
- *Manager Agent:* Responsible for periodically checking the status of the system, detecting system overloads and possible failures that may occur in the agents of each of the organizations.
- *Organization Agent:* This agent is responsible for verifying the operations of virtual organizations, ensuring security and load balancing. This agent also provides encryption services.
- *Normative Agent:* It is responsible for enforcing compliance with the rules in communications between agents.
- *Database Agent:* This agent is the only agent in the organization that has database access permissions. It is in charge of storing the system status information, analyzing the data persistence and consistency capabilities.
- *Information Agent:* Responsible for managing the services available within the virtual organizations, indicating which services are available for each of the agents. When an agent joins the system, it must indicate which services are available. In this way, when another agent requests a service, it must query this agent to know which entity is in charge of offering it.

For its correct operation and scalability, the system uses different databases; The system uses the database inside the PANGEA organization to store the information of the system agents, the services provided by each one, and the tasks that any agent can carry out. Apart from this, the system has an additional database used to store the specific information of the case study, the areas selected for inspection, detected pools generated tiles.

One of the advantages offered by this architecture is the contact network, where an external agent can search for and execute a service. To do so, the external agent must send a message to the Manager Agent, indicating the required service with the necessary parameters for that service. In collaboration with the other agents of the PANGEA organization, organization Agent, Information Agent, and Database Agent, this agent responds with a list of available agents that can carry out the requested service. Finally, the agent who is to perform the task must accept the proposal to carry out the task. Fig. 5 shows an example of a request for the service of extracting features from a parcel by an external agent.

V. Proposed System

The main challenge of this work is to design a platform capable of automatically detecting illegal pools at the lowest possible cost. This process requires data sources that are updated frequently and at the lowest possible cost. Following the current method for this verification, administrations use images of the exterior of people's residences to check existing constructions and verify that they are in the official register. Therefore, the images fulfill the role of a low-cost and up-to-

date data source thanks to GIS tools and represent a method already used by public administrations. The system adapts to the current way of working without creating any problems.

Considering that the primary source of data obtained is images, reviewing the literature, the systems that obtain the best results in classifying and detecting objects in images are Deep Learning algorithms. The use of these algorithms is crucial in developing this task as they will allow the detection of areas where pools exist from the images. Without such a detection capability, the proposed automatic system would not be possible. This section presents the case study based on the sub-block image classification technique. Fig. 6A shows an example of a tile with zoom 18, and Fig. 6B shows an example with zoom 19.

Fig. 6. Satellite Image for Detection with Zoom 18.

The proposed solution has three main blocks, the block of satellite image generation, the block of detection and classification of pools from the generated images, and, finally, the block of checking their legality with municipal databases. In addition, for interconnecting the parts, the PANGEA multi-agent architecture is used, briefly explained in Section IV.

The following sections describe the development steps. Section V.A deals with the development of the image search system in the maps. Section V.B is where the algorithms and classification methods used are presented, contemplating the second block of the work. Added to that is Section V.B.1, which presents the dataset formed by the images and their annotations, followed by the algorithms in sections V.B.2, V.B.3 and V.B.4. Finally, Section V.C describes the system that allows checking if a pool is legally registered.

A. Images Generation System

It is necessary to have a database covering a large area and containing aerial photos to detect pools. Users can build such a database with private or public tools and use expensive large aircraft systems for less expensive drone-based solutions.

The proposed block for the generation of images is interoperable and obtains data from several providers, such as Bing Maps [26], Google Maps [27], OpenStreetMaps [28], ESRI World Imagery [29], Wikimedia Maps [30], NASA GIB S [31], Carto Light [32], Stamen Toner B & W [33] and the Sentinel [34]. This portfolio of service providers covers a large part of the inhabited areas and are publicly available.

In the present research work, the user draws a zone on the map to inspect the area. In this tool, it is possible to configure, on the one hand, the zoom of the images, and on the other hand, the map data source, as shown in Fig. 7a.

Then, the system generates a grid with small map fragments (tiles) filling the entire drawn area. This process is repeated for each selected zoom as many times as necessary. Fig. 7b illustrates the process of transforming the selected area into the corresponding tiles.

B. Swimming Pool Detection Algorithms

The algorithms used have in common the preprocessing phases of each of the images. The whole process is performed in an identical way and with the same parameters. It is possible to emphasize the process of transformation of the image to grayscale and the subdivision of the image in N blocks of equal size. Subsequently, for each of these blocks, image processing is performed to extract the texture descriptors.

The texture descriptors refer to information about the spatial arrangement of color or intensities in an image. The feature vectors, formed from the texture descriptors, are normalized and used in training the classifiers. The programming language used for the development of the algorithms and experiments was Python 3. In particular, the OpenCV, *skimage* and *mahotas* image processing libraries were used. In addition, the machine learning libraries *scikit-learn, TensorFlow, keras* and *imbalanced-learn* have been used.

1. Datasets

The dataset used for training is a proprietary dataset. The Images Generation System explained in section V.A has been used to build it. To train the model, 999 images of 512x512 pixels with zoom 18 were obtained from Redlands, CA, around Prospect Park because it has a high concentration of swimming pools. Subsequently, the Roboflow web application [35] was used, which allows for easy labeling of the images. The set of images created consists of a single class, Swimming Pools.

The complete dataset have 778 annotated images with pool and 221 images unannotated, that do not contain any pool. In total, there are 2300 labels of the pool class. All images are randomly arranged in the dataset that was separated in two sets (Table II): the training set (80%) with 799 images and 1892 labels, and the validation set (20%) with 200 images and 408 labels.

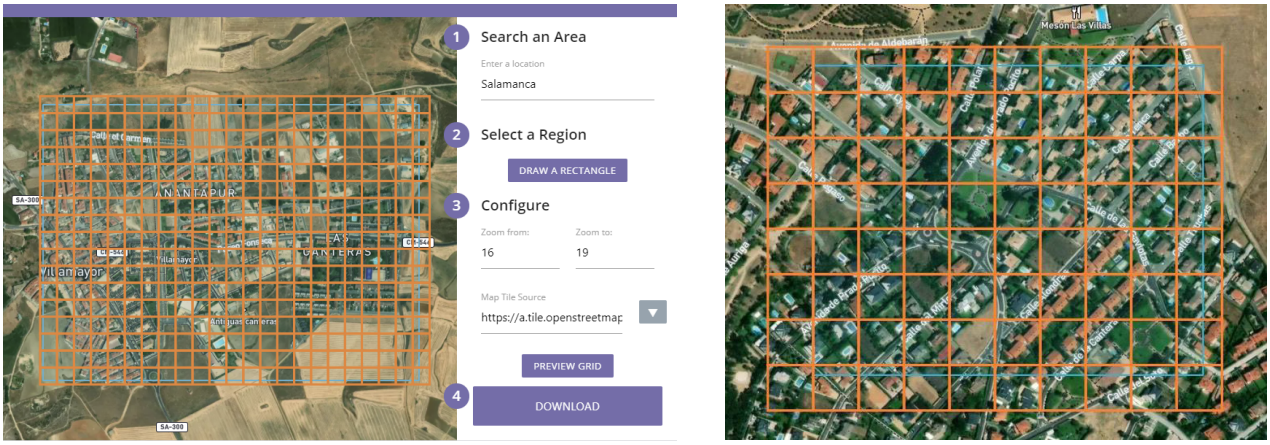
TABLE II. Number of Pools in Each Set of Training Images

Set	Images	Pools
Training	799	1892
Validation	200	408
All	999	2300

The storage format of the bounding box information for the annotations or labeling of this dataset is different for each object detection model. We need three different storage versions of the bounding box data, one for each object detection model used for the comparison. In [36] the three different datasets are published, one for each detection model, for further verification of the results or evaluation of new algorithms. Fig. 8 shows a example of labeling set.

2. Yolov4

For the training of the YoloV4 neural network, we use the Jetson Xavier AGX hardware device. The configuration file used can be found at [37]. It uses the Yolo Darknet annotation format in TXT file format. Note that we set the image dimensions to 512x512 pixels and a maximum of 6000 batches.



(a) Map area selection

(b) Photo generation based on map selection

Fig. 7. Image grid generation tool.

Fig. 8. Example of labeling a training image.

3. MaskRCNN

For training with the MaskRCNN algorithm, a notebook within Google Colab [38] was used. The algorithm makes use of the Pascal VOC annotation format [39] in XML file format. The training parameters set were 150 epochs and minimum detection confidence of 70%. Note that the notebook used saves the trained model at each epoch, which allows us to choose the best model. The trained models that gave the best value in any of the metrics F1, Accuracy, Average Accuracy, True Positive, False Positive, False Negative are selected. Subsequently, the selected model has the best results in the evaluation process from among them.

4. Detectron2

Finally, for the training of the Detectron2 algorithm, a Google Colab workbook [40] has been utilized. It uses the COCO annotation format [41] in JSON file format. It is worth noting that, for this algorithm, we can include the segmentation annotations for each tag. In order to be on equal footing with the previous algorithms, we have not included segmentation annotations except the bounding box annotations.

This method only uses the algorithm Faster-RCNN, whose basic configuration file can be found in the Detectron2 Repository [42]. In addition, we set the learning rate to 0.01 with a maximum of 50,000 iterations and steps in 30,000, 40,000, and 45,000.

C. Legal Registration Check System

For the registration of properties, some organizations aim to guarantee the legal security of the operations carried out in the real estate market. For example, in Spain, the registration of a swimming pool is not compulsory, but it is always advisable, as otherwise, it will not be valid in the eyes of third parties. In addition, the swimming pool affects the payment of real estate tax (IBI) since, like other types of constructions and installations on the property, it adds additional value to the property.

The name and correspondence of the bodies responsible for ensuring the registration of new works vary from country to country. For example, in the USA, the responsibility for these registrations belongs to the municipalities. In Spain, on the other hand, there is a body in charge of this competence called the Dirección General del Catastro (General Directorate of Cadastre). In both cases, these bodies offer a service to check the buildings registered for a given point on the coordinate map used in services such as Google Maps.

Therefore, we propose that the system obtains the coordinates corresponding to these pools after obtaining the pools detected in the images. In this way, the system will check if the property corresponding to the coordinates obtained has the detected pool registered. In this case, the system makes this check through a request to an endpoint of the API offered by the services of the aÿliated organization. Based on the data obtained in response, it will be possible to check the registered constructions, determining whether or not the pool is registered.

VI. Results

The tools used for training the models have methods for calculating some metrics using the validation set. However, we use a new process to evaluate the algorithms without relying on these tools and evaluating some new issues. This method consists of a new set, the evaluation set, which contains representative images of the final system. In addition to comparing the different models trained with the three algorithms, the method compares images with zoom 18 (Z18 set) and images with zoom 19 (Z19 set).

This Section is structured as follows: Section VI.A explains the metrics used. Section VI.B details the evaluation set. Sections VI.C, VI.D and VI.E show the results obtained from each algorithm. Section VI.F shows the comparison between the models trained with the different algorithms is given. Also, a comparison is also made between the Z18 set and the Z19 set. Finally, the VI.G section describes a system for check if the pools are registered legally by the appropriate agency.

A. Metrics

For the quantitative evaluation, we used the following metrics: accuracy, i.e., the relationship between true positives (TPs) and true positives (TPs) along with false positives (FPs) (Equation 1); recall, which is the probability that an image is classified as positive and

Fig. 9. Evaluation image with zoom 18.

Fig. 10. Evaluation image with zoom 19.

the relationship between the TPs and the TPs together with the false negatives (FNs) (Equation 2); and F1, which is combination of the two previous metrics (Equation 3).

We classified the speed measured in frames per second (FPS); the mean average precision (mAP), calculated by the precision and recall curve; and the intersection over union (IoU), which is the overlapping area between the annotated bounding box of the object in the image and the detected bounding box by the model. For the mAP measure, the notation of mAP@X is used, where X indicates the IoU threshold value used to calculate the AP.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3}$$

B. Evaluation Set

For the evaluation of the algorithms, we establish a reduced set of images. We have chosen eight locations in the province of Salamanca (Spain), obtaining a zoom 18 image (Z18 set) and a zoom 19 image (Z19 set) for each location. In total, we have obtained 16 images. Fig. 9 shows a Z18 image of a point and Fig. 10 shows a Z19 image, with their respective labelling.

We consider the following criterion for calculating the metric values for each trained model: It will consider neither a true positive nor a false positive if the model detects a not labeled pool for the evaluation.

These initially unlabelled objects are usually of tiny size in the image. Human experts are also not able to classify whether they are pools or not. This evaluation set contains in total 85 pools. Furthermore, the set splits into two image sets with 50 pools for the Z18 set and 35 pools for the Z19 set (Table III).

TABLE III. Number of Pools in Each Set of Assessment Pictures

Set	Images	Pools
Zoom 18	8	50
Zoom 19	8	35
All	16	85

C. Yolov4

As can be seen in Fig. 11 the model trained with the YoloV4 neural network managed to detect 365 pools correctly and 56 detections as false positives using the validation set. This model has given 89.75% of mAP@0.50, 87% accuracy, 89% recall, and 88% F1-Score. These results come from a confidence threshold of 25%.

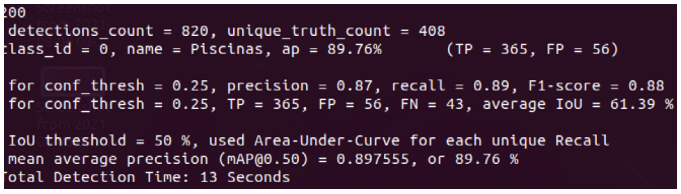


Fig. 11. Validation results of the best model trained with YoloV4.

In Fig. 12 and Fig. 13 we can observe the detection on images of the evaluation set with zoom 18 and zoom 19, respectively. These example of detection use a confidence threshold of 10% as it has yielded the best results between the confidence thresholds 70%, 50%, 25%, and 10%.

Fig. 12. YoloV4 detection with zoom 18.

Fig. 13. YoloV4 detection with zoom 19.

The model trained with the zoomed image 18 has managed to detect 8 pools out of 9, one false positive, and there is one discarded detection as we do not know if it is a pool or not. As for the zoom image 19 it has detected 5 pools out of 5 and there is one discarded detection for the same reason.

Table IV shows the results obtained when using the evaluation set for detection with the model trained on YoloV4 with a confidence threshold of 10 percent. The nomenclatures used in the table are: **TP** = True Positive, **FP** = False Positive, **FN** = False Negative, **GT** = Ground Truth or total of truth pool, **Prec** = Precision, **RC** = Recall and **F1** = F1-Score.

TABLE IV. VALUES OF THE YOLOV4 MODEL METRICS ON THE EVALUATION SET

Set	TP	FP	FN	GT	Prec	RC	F1
Z18	43	2	7	50	95.6%	86.0%	90.5%
Z19	33	0	2	35	100%	94.3%	97.1%
All	76	2	9	85	97.4%	89.4%	93.3%

D. MaskRCNN

Training with MaskRCNN returned all models for each epoch. This algorithm allows us to evaluate each model resulting from each epoch individually. In this case, we have chosen the trained models whose results stand out in some of the metrics returned by the training tool with the validation set, such as mAP@0.5, precision, recall, and F1-Score. For each of these chosen models, we have calculated their metrics with the evaluation set. Among them, we highlight the model resulting from epoch 46 that gave us the best results. This trained model stood out from the other models resulting from this training because of its high accuracy, whose value reached 86.3%. In Fig. 14 we can observe the values of the metrics that the epoch 46 model has had with the validation set.

```
==============] - 90s 691ms/step - loss: 0.5755 - val_loss: 1.2065

ference model (last checkpoint of the train model)

map: 0.7354 TP: 297 FP: 47 FN: 111 total: 408 precision: 0.8633 recall: 0.7279 F1: 0.7898
```

Fig. 14. Validation results of the best model trained with MaskRCNN.

It has correctly detected 297 pools or true positives and incorrectly detected 47 pools or false positives. It has reached a mAP@0.5 of 73.54%, a recall of 72.79%, and an F1-Score of 78.98%.

In Fig. 15 and Fig. 16 we can observe the detections on the images of the evaluation set with zoom 18 and zoom 19, respectively.

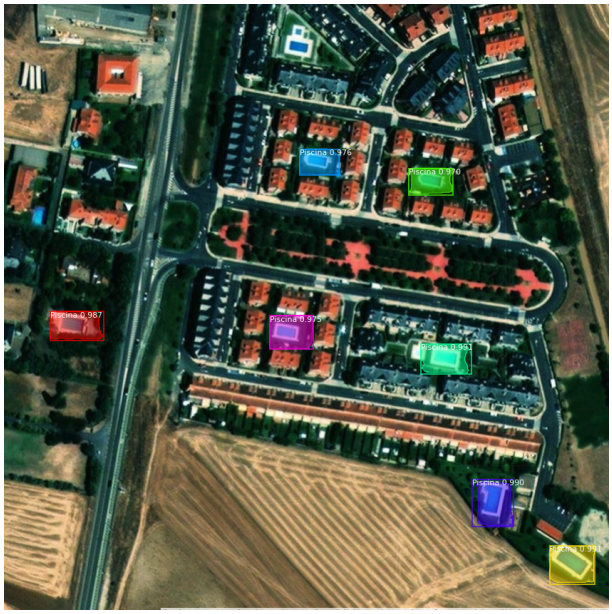


Fig. 15. Example of MaskRCNN detection with zoom 18.

These detections use a confidence threshold of 90% because it yields better results and allows filtering out more false positives. The model has detected 7 pools out of 9 in the zoom 18 image and 4 out of 5 in the zoom 19 image.

Fig. 16. Example of MaskRCNN detection with zoom 19.

Table V shows the results obtained when using the evaluation set for detection with the epoch 46 model trained on MaskRCNNN with a confidence threshold of 90%.

TABLE V. VALUES OF THE METRICS OF THE MASKRCNN MODEL ON THE EVALUATION SET

Set	TP	FP	FN	GT	Prec	RC	F1
Z18	29	0	21	50	100%	58.0%	73.4%
Z19	29	2	6	35	93.5%	82.9%	87.9%
All	58	2	27	85	96.7%	68.2%	80.0%

E. Detectron2

Finally, the model trained with Detectron2 has obtained an AP@0.5 of 87.23% with the validation set, as we can see in Fig. 17. In this case, the tool used for training with Detectron2 also calculates the mAP@0.5:0.95 with a result of 35.06% and the AP@0.75 with 16.43%.

```
|   AP   |  AP50  |  AP75  |  APs   |  APm   |  APl  |
|:------:|:------:|:------:|:------:|:------:|:-----:|
| 35.056 | 87.238 | 16.431 | 20.691 | 36.411 |  nan  |
[09/02 03:44:51 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN.
[09/02 03:44:51 d2.engine.defaults]: Evaluation results for my_dataset_val in csv format:
[09/02 03:44:51 d2.evaluation.testing]: copypaste: Task: bbox
[09/02 03:44:51 d2.evaluation.testing]: copypaste: AP,AP50,AP75,APs,APm,APl
[09/02 03:44:51 d2.evaluation.testing]: copypaste: 35.0561,87.2379,16.4311,20.6906,36.4108,nan
```

Fig. 17. Validation results of the best model trained with Detectron2.

Fig. 18 and Fig. 19 show the detections on the images of the evaluation set with zoom 18 and zoom 19, respectively. These detections use a confidence threshold of 50%. The trained model detects 7 pools out of 9 in the zoom 18 image and 4 out of 5 in the zoom 19 image.

Fig. 18. Example of Detectron2 detection with zoom 18.

Fig. 19. Example of Detectron2 detection with zoom 19.

Table VI shows the results obtained when using the evaluation set for pool detection with the model trained with Detectron2 with a confidence threshold of 50%.

TABLE VI. Values of the Metrics of the Detectron2 Model on the Evaluation Set

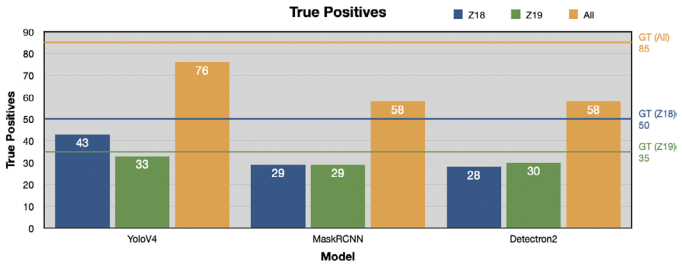
Set	TP	FP	FN	GT	Prec	RC	F1
Z18	28	5	22	50	84.8%	56.0%	67.5%
Z19	30	3	5	35	90.9%	85.7%	88.2%
All	58	8	27	85	87.9%	68.2%	76.8%

F. Comparison

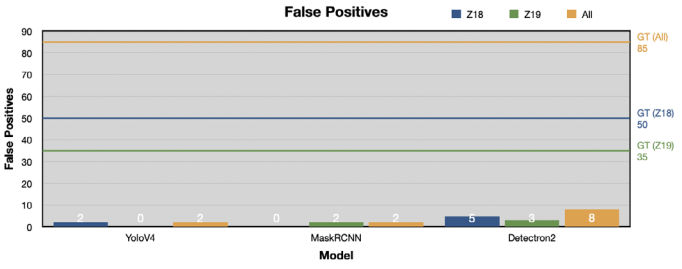
Once we have obtained the metrics of each model trained with the evaluation set, we proceed to compare the algorithms. Fig. 20a, Fig. 20b and Fig. 20c show plots comparing the models with the true positive, false positive and false negative values, respectively.

The model trained with the YoloV4 neural network yielded much better results than the other two algorithms. It managed to detect many more pools correctly, and, in addition, it only detected 2 pools incorrectly. On the other hand, if we use a confidence threshold of 25% or more with the model trained with YoloV4 we achieve that the model does not detect pools incorrectly. With the 25% threshold, we obtain 61 true positives and 0 false positives in the whole set.

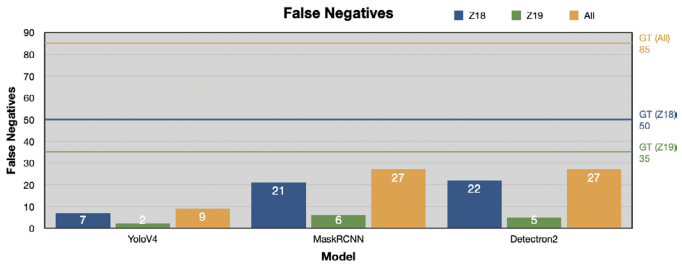
Finally, if we compare between the set of zoom 18 images (Z18) with the set of zoom 19 images (Z19) we can observe that in the three algorithms, there is a higher detection of true positives with the set Z19.



(a) True positives value comparison.



(b) False positive value comparison.



(c) False negative value comparison.

Fig. 20. True positive, false positive and false negative values comparison.

The YoloV4 algorithm has been able to detect 33 pools out of 35 in the Z19 set, while it has detected 43 out of 50 in the Z18 set. This difference is even more marked in the other two algorithms, where the number of true positives is almost identical between set Z18 and set Z19.

If we compare the precision metric of each model (Fig. 21) we obtain that the model trained with YoloV4 is more accurate than the other two algorithms. Furthermore, if we increase its confidence threshold to 25%, we obtain a precision of 100% on all three sets: Z18, Z19, and All. Finally, we can see that there is a lower precision with the Z18 set in all three algorithms compared to the Z19 set.

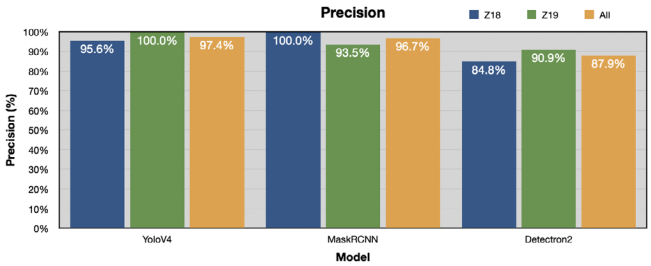


Fig. 21. Precision metrics comparison.

As for the recall metric, the YoloV4 algorithm has yielded better results, reaching 94.3% in the Z19 set. In addition, we see that with the Z19 set, they achieve a higher recall than the Z18 set, with a difference of almost 30% in the case of the Detectron2 algorithm.

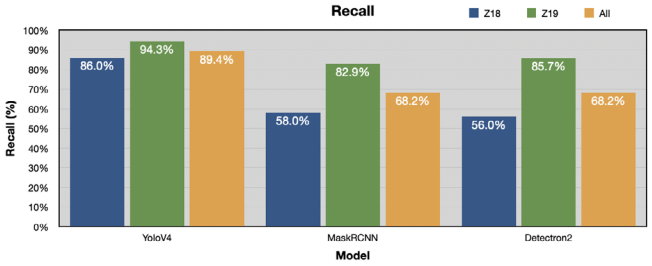


Fig. 22. Recall metric comparison.

Finally, regarding the F1-Score metric, which combines the precision and recall metrics, we can observe in Fig. 23. The YoloV4 algorithm is far superior to the other two algorithms, achieving up to 97.1% with the Z19 set. On the other hand, we see better results with the Z19 set compared to the Z18 set.

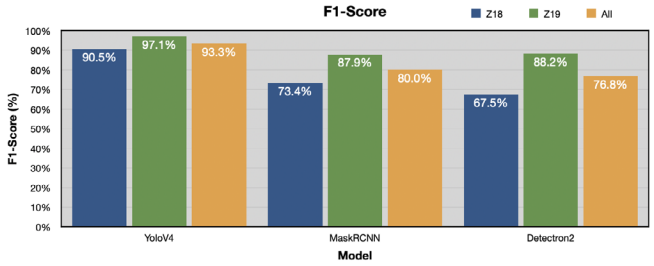


Fig. 23. F1-Score metric comparison.

G. Test Case of the Verification System

Before we can comment on the test case performed for the test system, we must describe it to know the system's situation. In this case, the test case has performed in a town near the city of Salamanca. The name of this town is Villamayor.

The *Dirección General del Catastro* is the body in charge of controlling, verifying and managing the records of buildings in Spain. The system has collected the images and check them, going through the pool detection subsystem. Their coordinates have been obtained from the detected pools to request the service offered by the *Dirección General del Catastro*. This legal system is in charge to check if the pools located in properties were registered. The result obtained from this process is in Fig. 24.

Fig. 24 shows three types of points distinguished by colors:

- **Blue:** This item identifies a pool detected by the system. This item has not yet gone through the system verification process.
- **Green:** This point identifies a pool detected in the system and was verified as registered to the corresponding property through its geographical coordinates.

Fig. 24. Results from the verification system.

- **Red:** This item identifies a swimming pool detected in the system and has no record for the corresponding property with its geographical coordinates.

In the observed village of Villamayor, as observed in Fig. 24, there are actually 27 pools. Of those 27 pools, the system has been able to detect 23 pools and unable to detect 5 pools that actually exist. Of those 23 detected pools, 22 of them are correct, however, one of them is incorrect. Of these pools, 22 are registered and linked to the related property, one is not registered, and one is still pending verification.

VII. Conclusions

This paper demonstrates that it is possible to determine the presence of a pool in an image with an accuracy better than 97% using a multi-agent architecture that allows distributed computing and has allowed the evaluation of different algorithms combined to improve the detection process.

After evaluating the algorithms and comparing them, we highlight the model trained with the YoloV4 neural network, which offers better results in all metrics. We propose to use this algorithm for pool detection using a confidence threshold of 10% in case errors or false positives can be assimilated, allowing a higher detection or recall, or to use a confidence threshold of 25% in case higher accuracy in pool detection is sought. Finally, it has been observed that for the detection of pools, it is better to use images with zoom 19 versus zoom 18. Although for images with zoom 19, it is necessary to process four times more images for the same area than with images with zoom 18, it is very convenient to sacrifice more computational resources to detect pools. Moreover, in this case, study, it is not vital to display the detections in real-time, so spending a few extra seconds in the detection process is not a concern.

Finally, a test case is made to observe a specific population to check the system's operation. In this way, based on the test carried out in the town of Villamayor, an operating system has been verified. This test has been possible to certify the system's effectiveness to determine which pools are registered or not in the corresponding official bodies. In addition, the system has proven to help check the automatic detection of pools and the checking of pool records.

Acknowledgments

Héctor Sánchez San Blas's research is supported by the Spanish Ministry of Universities (FPU Fellowship under Grant FPU20/03014). The research of Luis Augusto Silva has been funded by the call for predoctoral contracts USAL 2021, co-financed by Banco Santander.

References

[1] D. Tien, T. Rudra, A. B. Hope, "Swimming pool identification from digital sensor imagery using SVM," *Proceedings - Digital Image Computing Techniques and Applications: 9th Biennial Conference of the Australian Pattern Recognition Society, DICTA 2007*, pp. 523–527, 2007, doi: 10.1109/DICTA.2007.4426841.

[2] W. Passos, E. Silva, S. Netto, J. Martins, Y. Costa, G. Araujo, A. Lima, "Detecção de Potenciais Focos do Aedes aegypti em Vídeos Aéreos Usando Redes Neurais," pp. 22–25, 2020, doi: 10.14209/sbrt.2020.1570661555.

[3] P. C. Gray, K. C. Bierlich, S. A. Mantell, A. S. Friedlaender, J. A. Goldbogen, D. W. Johnston, "Drones and convolutional neural networks facilitate automated and accurate cetacean species identification and photogrammetry," *Methods in Ecology and Evolution*, vol. 10, no. 9, pp. 1490–1500, 2019, doi: 10.1111/2041-210X.13246.

[4] M. I. Habibie, T. Ahamed, R. Noguchi, S. Matsushita, "Deep Learning Algorithms to determine Drought prone Areas Using Remote Sensing and GIS," pp. 69– 73, 2020, doi: 10.1109/AGERS51788.2020.9452752.

[5] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[6] R. Girshick, J. Donahue, T. Darrell, J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[7] R. Girshick, "Fast R-CNN," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 1440–1448, 2015, doi: 10.1109/ICCV.2015.169.

[8] H. Rampersad, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Total Performance Scorecard*, pp. 159–183, 2020, doi: 10.4324/9780080519340-12.

[9] K. He, G. Gkioxari, P. Dollár, R. Girshick, "Mask r- cnn," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020, doi: 10.1109/TPAMI.2018.2844175.

[10] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, K. He, "Detectron." https://github.com/ facebookresearch/detectron, 2018.

[11] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, "Detectron2." https://github.com/ facebookresearch/detectron2, 2019.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, A. C. Berg, "SSD: Single Shot MultiBox Detector," 12 2015, doi: 10.1007/978-3-319-46448-0_2.

[13] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You only look once: Unified, real-time object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[14] J. Redmon, A. Farhadi, "YOLO9000: Better, faster, stronger," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.

[15] J. Redmon, A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018, doi: http://arxiv.org/abs/1804.02767.

[16] A. Bochkovskiy, C. Wang, H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020, doi: https://arxiv.org/abs/2004.10934.

[17] S.-H. Chen, C.-W. Wang, I.-H. Tai, K.-P. Weng, Y.-Chen, K.-S. Hsieh, "Modified yolov4-densenet algorithm for detection of ventricular septal defects in ultrasound images," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 7, pp. 101-108, doi: 10.9781/ijimai.2021.06.001.

[18] L. A. Silva, H. S. S. Blas, D. P. García, A. S. Mendes, G. V. González, "An architectural multi-agent system for a pavement monitoring system with pothole recognition in uav images," *Sensors*, vol. 20, no. 21, pp. 1–23, 2020, doi: 10.3390/s20216205.

[19] C. Galindo, P. Moreno, J. Gonzalez, V. Arévalo, "Swimming pools localization in colour high- resolution satellite images," vol. 4, pp. IV–510–IV–513, 2009, doi: 10.1109/IGARSS.2009.5417425.

[20] M. Kim, J. B. Holt, R. J. Eisen, K. Padgett, W. K. Reisen, J. B. 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, no. 11, pp. 103– 113, 2011, doi: 10.14358/pers.77.11.1169.

[21] B. Rodríguez-Cuenca, M. C. Alonso, "Semi-automatic detection of swimming pools from aerial high- resolution images and LIDAR data," *Remote Sensing*, vol. 6, no. 4, pp. 2628–2646, 2014, doi: 10.3390/rs6042628.

[22] C. Ferner, G. Eibl, A. Unterweger, S. Burkhart, S. Wegenkittl, "Pool detection from smart metering data with convolutional neural networks," *Energy Informatics*, vol. 2, pp. 1–9, 2019, doi: 10.1186/s42162-019-0097-8.

[23] Z. Domozi, A. Molnar, "Surveying private pools in suburban areas with neural network based on drone photos," *EUROCON 2019 - 18th International Conference on Smart Technologies*, pp. 1–6, 2019, doi: 10.1109/EUROCON.2019.8861770.

[24] B. Lima, L. Ferreira, J. M. Moura, "Helping to detect legal swimming pools with deep learning and data visualization," *Procedia Computer Science*, vol. 181, no. 2019, pp. 1058–1065, 2021, doi: 10.1016/j.procs.2021.01.301.

[25] C. Zato, G. Villarrubia, A. Sanchez, I. Barri, E. Rubión, A. Fernández, C. Sánchez, J. Cabo, T. Álamos, J. Sanz, J. Seco, J. Bajo, J. Corchado Rodríguez, "Pangea - platform for automatic construction of organizations of intelligent agents," vol. 151, 01 2012, doi: 10.1007/978- 3-642-28765-7_27.

[26] J. Schwartz, *et al.*, "Bing maps tile system," 2009. http://msdn.microsoft.com/en-us/library/ bb259689.aspx, (accessed in: 13/09/2021).

[27] "Google maps." https://maps.google.com, (accessed in: 13/09/2021).

[28] "Open street maps." https://www.openstreetmap. org/, (accessed in: 13/09/2021).
[29] "Esri world imagery." https://www.arcgis. com/apps/mapviewer/index.html?layers= 10df2279f9684e4a9f6a7f08febac2a9, (accessed in: 13/09/2021).
[30] "Wikimedia maps." https://maps.wikimedia.org/, (accessed in: 13/09/2021).
[31] "Nasa gibs." https://map1.vis.earthdata.nasa.gov/, (accessed in: 13/09/2021).
[32] "Carto light." https://cartodb-basemaps-c. global.ssl.fastly.net/, (accessed in: 13/09/2021).
[33] "Stamen toner b & w." https://stamen.com/, (accessed in: 13/09/2021).
[34] "Sentinel." https://www.sentinel-hub.com/, (accessed in: 13/09/2021).
[35] "Roboflow." https://roboflow.com, (accessed in: 13/09/2021).
[36] "Swimming Pool Detect." https://github.com/ Hectorssb/SwimmingPoolDetection, (accessed in: 02/11/2022).
[37] "YoloV4 cfg." https://github.com/Hectorssb/ SwimmingPoolDetection/blob/main/Yolov4/cfg/ yolov4-obj.cfg, (accessed in: 02/11/2022).
[38] "Train Mask-RCNN Model on Custom Data." https://colab.research.google.com/drive/1rBuhT8AjP2td20otdUnpuF7CCMmjRb4O, (accessed in: 13/09/2021).
[39] M. Everingham, L. Van Gool, C. K. I. Williams, Winn, A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.
[40] "Detectron2 Beginner's Tutorial." https://colab.research.google.com/drive/1n-_ nulKMxxCF6Jg4WMw20mO8R6rqC078, (accessed in: 13/09/2021).
[41] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, P. Dollár, "Microsoft coco: Common objects in context," 2014. [Online]. Available: https://arxiv.org/abs/1405.0312, doi: 10.48550/ARXIV.1405.0312.
[42] Detectron2, "faster_rcnn_X_101_32x8d_FPN_3x." https://github.com/facebookresearch/ detectron2/blob/main/configs/COCO-Detection/faster_rcnn_X_101_32x8d_FPN_3x.yaml, (accessed in: 13/09/2021).

Héctor Sánchez San Blas

He is a Researcher at the University of Salamanca. He is the beneficiary of an FPU predoctoral contract in the 2020-2021 call. He is a Ph.D. student in computer engineering at the same university and has a degree in Computer Engineering and a Master's degree in Intelligent Systems at the same center. During his master's degree, he was a collaborating researcher at the Expert Systems and Applications Laboratory (ESALab) of this university, collaborating with research projects related to the Internet of Things, Virtual Reality applications, and machine learning. Currently, he is researching the development of IoT and neural networks together with research projects focused on machine vision and Smart Cities.

Antía Carmona Balea

She is a Pd.D. student in Computer Engineering at University of Salamanca (USAL). She has a degree in Chesmistry at the University of Valladolid (UVA), she received the master's degree in Meteorology from UNED and she is currently doing a master in scientific communication. Her doctoral studies deal with IoT and artificial intelligence which she combines with her work as a secondary school teacher.

André Sales Mendes

He received a master's degree in Intelligent Systems and the Ph.D. degree in Computer Engineering from University Of Salamanca (USAL). He is current working on Expert System And Applications Laboratory Research Group, Computer Science Department, University of Salamanca, as an Adjunct Professor. It has a numerous publication in international impact journals indexed in JCR reference ranking. His main research interests focus on the field of artificial intelligence, IoT (internet of things) and robotic. He also collaborates in several research projects within the research group.

Luís Augusto Silva

He received his Master's degree in Applied Computing from the University of Itajaí Valley, Brazil, in 2019. He has a degree in Internet Systems from the Federal Institute of Santa Catarina (IFC), Camboriú, Brazil, ending in February 2017. His research during his master's degree covered the field of Notification Systems, IoT, and Data Privacy. During the master's degree, he was a collaborating researcher at the Laboratory of Embedded and Distributed Systems (LEDS) at UNIVALI, collaborating with research projects related to the Internet of Things. Since August 2020, he has been a Ph.D. student in Computer Engineering at the Universidad de Salamanca - Spain, and a researcher at the Expert Systems and Applications Laboratory (ESALab). His research lines are directly related to Internet of Things, Embedded Drone Systems, and Data Privacy applied to Smart Environments.

Gabriel Villarrubia González

He received the master's degree in intelligent systems from the University of Salamanca, in 2012, the master's degree in Internet Security, in 2014, the master's degree in information systems management, in 2015, and the Ph.D. degree from the Department of Computer Science and Automation, University of Salamanca. He is a Computer Engineer at the Pontifical University of Salamanca in 2011. He is currently a Research Professor with the Department of Informatics. Throughout his training, he has followed a well-defined line of research, focused on the application of multi-agent systems to ambient intelligence environments, with special attention to the definition of intelligent architectures and the fusion of information.