DETECTION OF POOLS IN SATELLITE IMAGERY USING YOLOv8 AND ROBOFLOW

Elliot Weiner, Ryan Roos

University of Rochester

ABSTRACT

The goal of this project was to develop a model that could identify pools from satellite imagery with a high degree of accuracy. This could be used as both a tool for property information verification and as a computer vision research opportunity. To start, we researched satellite imagery dataset creation and object detection models. We then created a custom dataset using a labeling tool called Roboflow and began training our algorithm with a YOLOv8 pre-trained model. We were able to make a model that was specified for properties in Arizona and New York, and the resulting combined dataset allowed for more regional versatility. Additionally we implemented our own version of transfer learning. In the end, our model achieved a high degree of accuracy in general pool detection.

1. INTRODUCTION

Modern artificial intelligence and machine learning techniques are rapidly improving the abilities of companies to effectively and efficiently complete tasks. The applications seem endless, as models can perform a variety of complex calculations and reduce the amount of time and resources necessary for them. Our project sought to expand on this idea by detecting swimming pools in satellite images.

This project has several applications for both businesses and governments. For instance, insurance companies and local governments which require extra tax or premiums for pool owners would be able to simply check residents' properties and ensure they were adhering to the appropriate guidelines. Additionally, business owners may use this tool to assess interest in pools in a given area, and decide the best market for services like pool cleaning and maintenance.

There is evidence for the potential of such applications going forward. French tax authorities have already begun to implement similar machine learning algorithms to find undeclared pools. In France, both in and above ground pools increase property value, meaning they require permits [1]. Although implemented on just a small scale now, their algorithm has already led to the discovery of over 20,000 undeclared pools and there are plans for country-wide deployment. Our model was never intended to compete with their model, but did serve an important purpose: to be more accessible and manageable. We sought to build ours using free, commercially available applications and tools. Additionally, this project did not only serve as a means for pool detection, but also an exploration into which current computer vision tools exist and their potential to be used in tandem to create more powerful architectures.

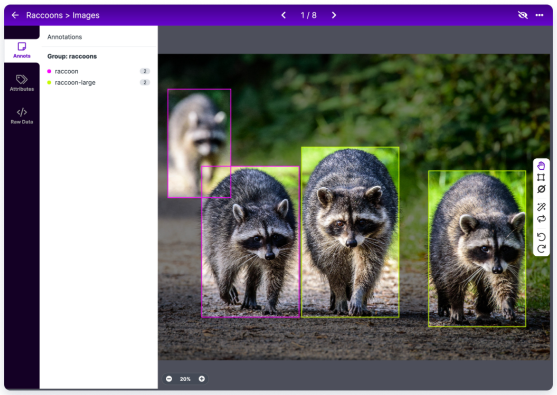
2. METHODS

There were two main areas we had to prioritize in our research: dataset creation and model selection. Initially, we searched for prelabeled training sets on Kaggle and other websites but could not find any. As we were unable to find a dataset, we decided it would be best to build one ourselves. We sourced our initial images on USGS Earth Explorer. This source, which is maintained by the US government, has a variety of files available for download from several different satellites [2]. We settled on images from a High Resolution Ortho-Imagery satellite. The benefit of using this satellite was the availability of high resolution images across several regions.

After pulling GIS files from Earth Explorer, we used QGIS to convert the files into images. These images ended up being too large with dimensions of 6250x6250. Running images this large through a ML model would take far too long to be efficient. It would also make labeling the images difficult. To fix this we clipped the image into 100 smaller images with a 625x625 resolution.

Next, we needed to find a way of labeling our data so that we could train our model. Roboflow was the tool we decided to use for labeling and dataset formatting. It was an invaluable online service that helped with several key tasks [3]. Firstly, it allowed us to upload our images to the website with ease. It also supported collaboration, which allowed us to work on the project at the same time from different computers. In terms of the labeling capabilities, Roboflow had an interface for placing bounding boxes and labels on key objects. This was crucial for efficiently generating a sufficiently large training dataset. It also allowed us to format our labeled dataset for multiple object detection models. From there, we had the option to use the final output dataset in two different ways. One way was to train directly in Roboflow and create accurate models based on an architecture of our choosing. Alternatively, we could have used an API key in a Jupyter Notebook to train a model ourselves. For our purposes, we decided the former would be best. We used the Roboflow training pipeline for our general models and trained another with our own transfer learning implementation. This model could then be tested against our other Roboflow implementations.

Roboflow Example Bounding Boxes



The last methodological decision to make was to determine which object detection model we would use as a base for our project. After extensive research, we settled on using a YOLOv8 pre-trained model. Its parent company, Ultralyrics, maintains extensive support documentation and a large library of functions that offers a streamlined approach to training, transfer learning, and metric recording [4]. The model requires minimal compute resources for inference and is well-suited for detecting smaller objects. This complemented our goal of identifying pools from satellite images. Other major computer vision models we considered, such as SSD (Single-Shot Detection), seemed to better handle large objects [5]. Ultimately, the difference in community and object detection size set YOLO apart from the other available options and led us to its most current version, YOLOv8.

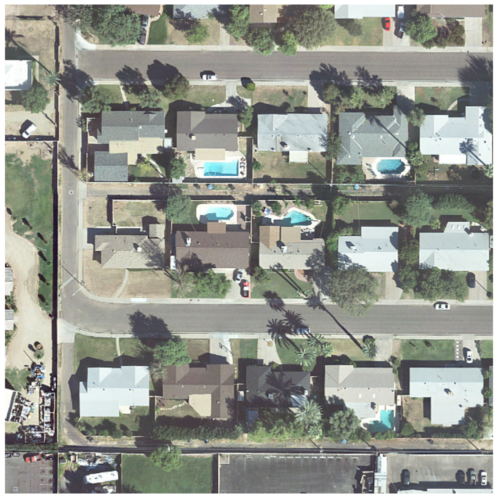
In terms of actual architecture, the model runs according to three distinct steps: gridding, bounding box regression and intersection over union (IoU). The first step divides an image into a grid of much smaller images so that predictions can be made for each box in the grid in the next step. After predictions are made, the boxes are combined based on probabilities until a full object is identified. This model allows for a single pass through the system (coining the model's title, You Only Look Once), which makes it optimal for our design [5].

From here, we sought to combine Roboflow and Yolov8 to create robust and versatile models. We started by creating our datasets which we sourced from satellite imagery of Arizona and New York. Next, we labeled our data and used the Roboflow pipelines to create YOLOv8 models and a combined dataset for both states. To create a training script for transfer learning, we implemented the Roboflow dataset API to pull the Arizona set. Then, we uploaded a pre-trained YOLO model and froze all but the last 5 layers and retrained the model on the new dataset. Finally, we pulled run metrics from the training to see how well our model performed.

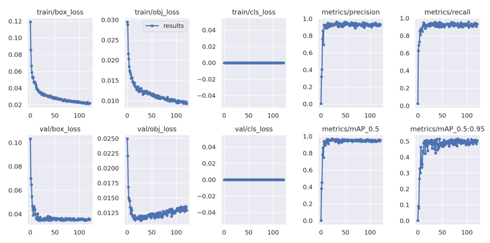
3. EXPERIMENTS

We conducted several experiments while constructing our model. First, we compared training metrics of our model made from the Arizona dataset to those from the model based on the New York dataset. This was a unique experiment primarily because of the geographical differences between both states. AZ tends to have bright, uncovered pools with minimal tree cover. NY tends to have greater tree cover over pools and less contrast between a pool and its surroundings. This makes detecting pools in New York images a much harder task, as the pools tend to blend into the landscape.

Arizona Dataset Example



Arizona Training Metrics

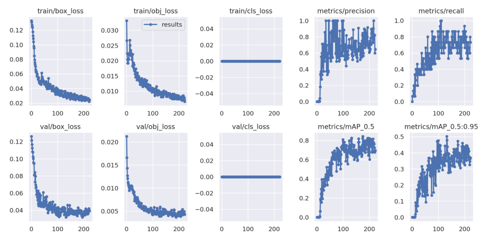


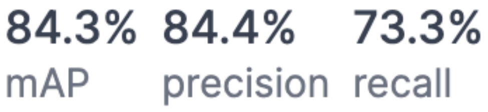
97.1% mAP 93.3% precision 93.2% recall

New York Dataset Example

Jordan Rd
Jordan Rd
Rensselaer Dr
Rensselaer Dr
Rensselaer Dr
Rensselaer Dr

New York Training Metrics

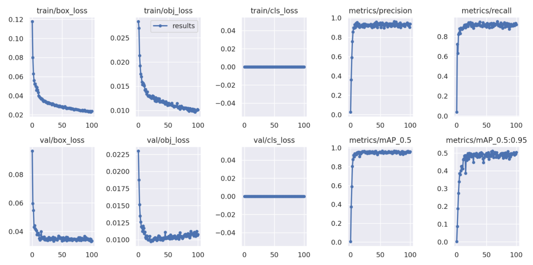


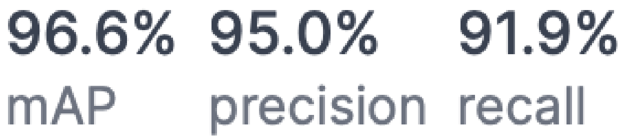


84.3%	84.4%	73.3%
mAP	precision	recall

This difference was shown clearly in the training metrics from our model. While there was only a 9 percent drop in precision from the Arizona dataset, the New York set showed a 20 percent drop in recall. Additionally, the mean Average Precision dropped by 13 percent. After comparing the two models, we decided to see the result of combining the datasets. Doing so produced the following metrics, which show a clear improvement from the New York set and only a slight drop from the Arizona dataset. We interpreted this as an overall improvement in model versatility and accuracy.

Combined United States Dataset Metrics

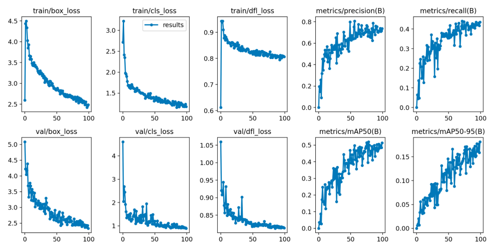




96.6%	95.0%	91.9%
mAP	precision	recall

Our last experiment was to compare our own transfer learning implementation to the model created by Roboflow and its training pipeline. We decided to use the Arizona dataset since it was considerably larger than our New York dataset, making it better for training. The results of the transfer learning model are shown below. Although the model seemed to perform much worse in all major metrics (dropping 20 percent in precision, 50 percent in recall and 46 percent in mAP) it still resulted in relatively good values for each. Results such as those below show a clear understanding of the basics of pool detection.

Arizona Dataset TF Model Metrics



```
Box(P            R          mAP50
0.732        0.433           0.51
```

One experiment that we originally wanted to conduct utilized multispectral imagery. Multispectral imagery can detect through objects, like tree leaves, and can also be accessed for free from US government agencies. The issue we encountered was that the color infrared satellites we found did not have a high enough resolution for pool identification. For this reason we chose to not proceed with the multispectral approach.

Example Multispectral Image

4. CONCLUSIONS

After conducting our experiments, all of our models showed a high level of accuracy when given an unlabelled image. The metrics above corroborate this, as our final results ranged from mid 0.7 to nearly 1.0 for all metrics (1.0 being a perfect result). Additionally, our own transfer learning implementation detected a significant number of pools. Although it wasn't quite as accurate as the pipelined models, it still showed a significant understanding of how to appropriately identify pools in satellite imagery.

The main outcome of our project was a strong understanding of the sheer power of free, commercial computer vision and dataset creation tools available currently. At no cost to us, we were able to do several types of training and use datasets with variability in size and geographic features. Compared to alternatives such as a ground up implementation of a Convolutional Neural Network, these tools made our research easier and faster while producing far more accurate models than we could have produced on our own.

Roboflow provided a simple way to create vast datasets in the order of thousands of images. There was next to no training required as its interface was simple and intuitive for new users. YOLOv8 presented us with free to use models which were pre-trained and easy to manipulate. Its extensive community and thorough documentation made our work more achievable by allowing us to abstract most of the model architecture. In turn, we were able to dedicate more of our focus toward greater tasks and areas of interest, such as the variation of our dataset across regions.

In addition to discovering the value of easily accessible computer vision tools, another valuable result of our work was seeing how powerful models like YOLOv8 perform across geographically diverse locations. When identifying objects in a region such as Arizona, obstructions, like tree cover, are much less concentrated. Therefore, objects are much easier to detect.. However, regions like New York contrast this with more tree and pool cover. Additionally, because of the fluctuating temperature in the northern United States, pool covers were an extra challenge for detection, as reflected in our model metrics. As we suspected, New York sourced datasets had higher variability in correctly-identified pools, which presented challenges for the model which was trained on them. Additionally, it is important to acknowledge other potential shortcomings of our New York model, which are primarily based on the fact that we used a much smaller set of imagery to train (approximately ¼ the size of the Arizona dataset). This may have played a large role in our model performances being so inconsistent with each other. However, we still expect that some inconsistencies would arise due to regional variation. With our combined US dataset model, we were able to create a more versatile object detection algorithm. This result opens the door for broader implementations of our algorithms, as data could be sourced from many different regions to create an all-encompassing general satellite imagery pool detector. Additionally, with different forms of imagery, including multispectral, we think that it is feasible to create an object detection model with greater versatility and accuracy.

5. REFERENCES

[1] EuroNews, "France uses AI to detect more than 20,000 undeclared swimming pools," *euronews*, 30-Aug-2022. [Online]. Available: https://www.euronews.com/my-europe/2022/08/30/france-uses-artificial-intelligence-to-detect-more-than-20000-undeclared-swimming-pools#:~:text=French%20tax%20authorities%20say%20artificial,declared%20and%20are%20correctly%20taxed. [Accessed: 05-May-2023].

[2] UWM, "UWM libraries research and course guides: Finding and using GIS Data: Earth explorer," *Earth Explorer - Finding and Using GIS Data - UWM Libraries Research and Course Guides at University of Wisconsin-Milwaukee*. [Online]. Available: https://guides.library.uwm.edu/c.php?g=567847&p=5338445. [Accessed: 05-May-2023].

[3] "Give your software the power to see objects in images and video," *Roboflow*. [Online]. Available: https://roboflow.com/. [Accessed: 05-May-2023].

[4] "Home," *Home - Ultralytics YOLOv8 Docs*. [Online]. Available: https://docs.ultralytics.com/. [Accessed: 05-May-2023].

[5] Neeraj, "Yolo vs SSD: Which one is a superior algorithm," *Algoscale*, 06-Mar-2023. [Online]. Available: https://algoscale.com/blog/yolo-vs-ssd-which-one-is-a-superior-algorithm/. [Accessed: 05-May-2023].