Multi-Class Object Detection for Architectural Photography

Michael Renehan

Table of Contents

Problem

Architecture studios generate a lot of imagery and image research that is valuable, but not always easy to access or search.

Goal

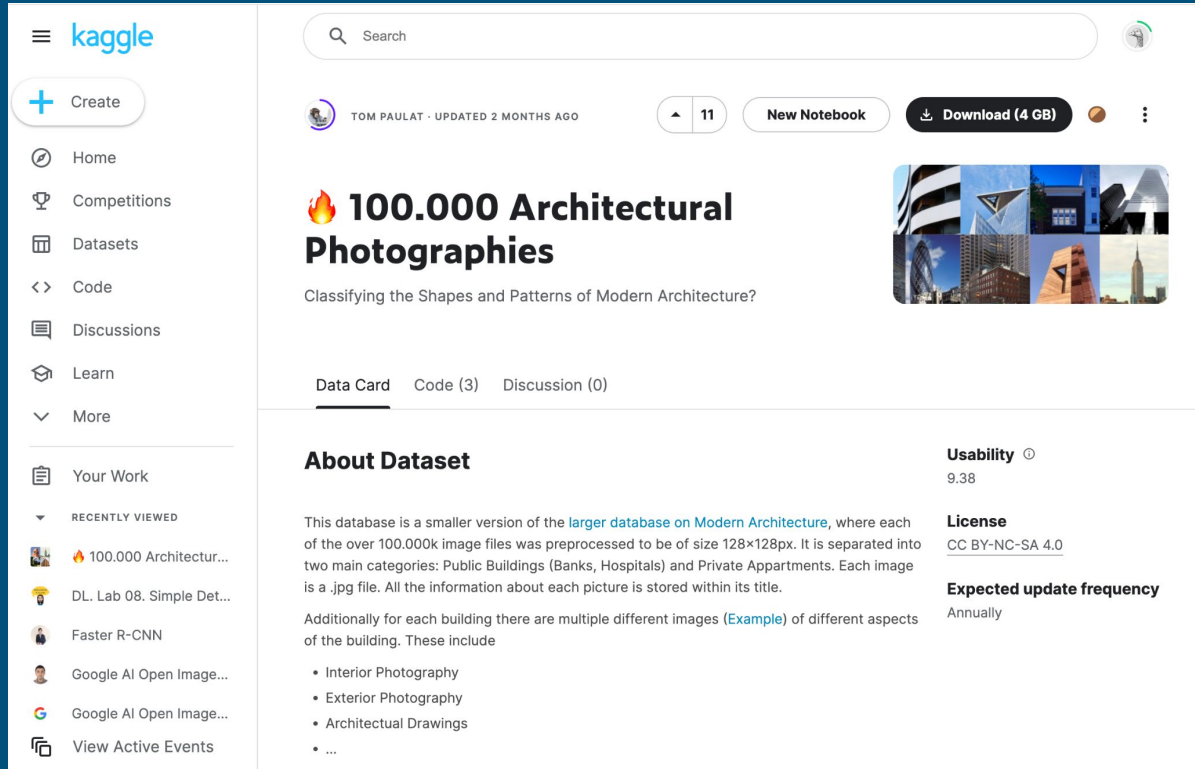
The goal of this project is to train an object-detection model on many classes of interest to architectural designers using photography of modern design, for the purposes of design research and knowledge management in the studio.

Data

Kaggle: Modern Architecture Dataset

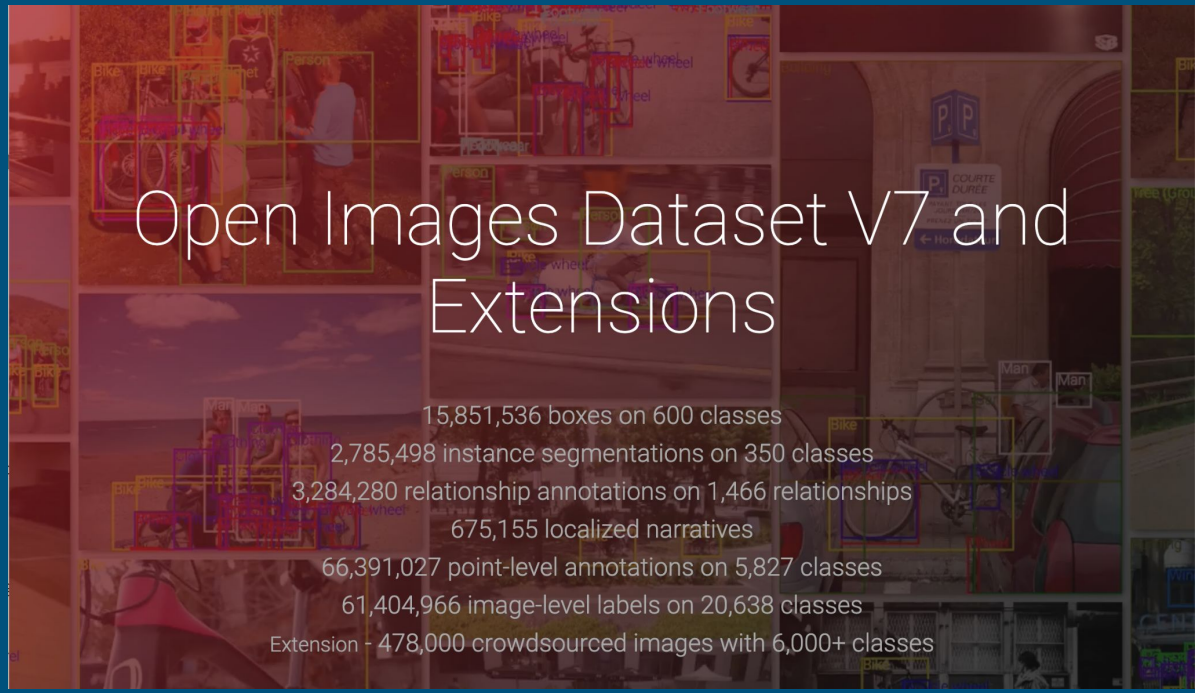
100,000 Architectural Photographs, with filenames that contain image-level labels.

https://www.kaggle.com/datasets/tompaulat/modern-architecture-100k-small-images



Google Open Images

Robust dataset with millions of images along with annotations.



Identify Classes of Interest with CountVectorizer

Using the tags in the filenames of the 100K image set, I identified 20 well-represented tags that were of interest.

TERM	COUNT	TERM	COUNT
Window	39353	**Interior**	16041
Chair	16076	**Exterior**	8617
Table	13407	**Lighting**	3156
Stairs	5649	**Bathroom**	3067
Countertop	5585	**Shelving**	2335
Door	5347	**Patio**	2330
Bed	2331	**Brick**	4237
Sink	3972	**Concrete**	1754
Couch	3143	**Living Room**	1959
Bathtub	1119	**Bedroom**	4008

Classes by Supercategory

These classes also provided an interesting and challenging cross-section of different kinds of detection, not limited strictly to “objects.”

Places

- Bedroom
- Living Room
- Bathroom
- Patio

Materials

- Brick
- Concrete
- Wood

Objects/Features

- Window
- Chair
- Table
- Stairs
- Bed
- Sink

Image-Level Attributes

- Interior
- Exterior

Challenges of Contemporary Design

Much of contemporary architecture is unconventional, making detection more challenging.

The scale and complexity of architectural photography also presents challenges.

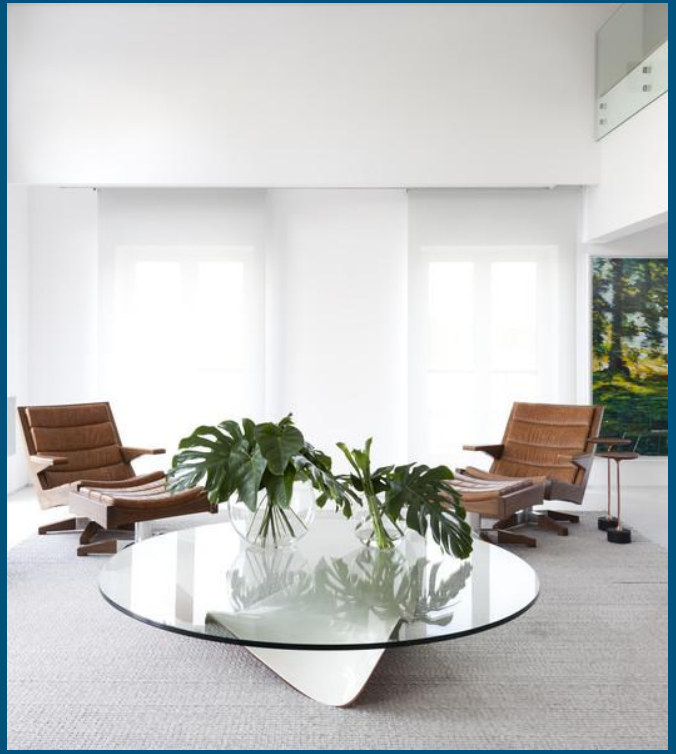
(IBA Timber Prototype House ICD University of Stuttgart. Door, Window, Facade)

Exploratory Modeling: Image Level Labeling

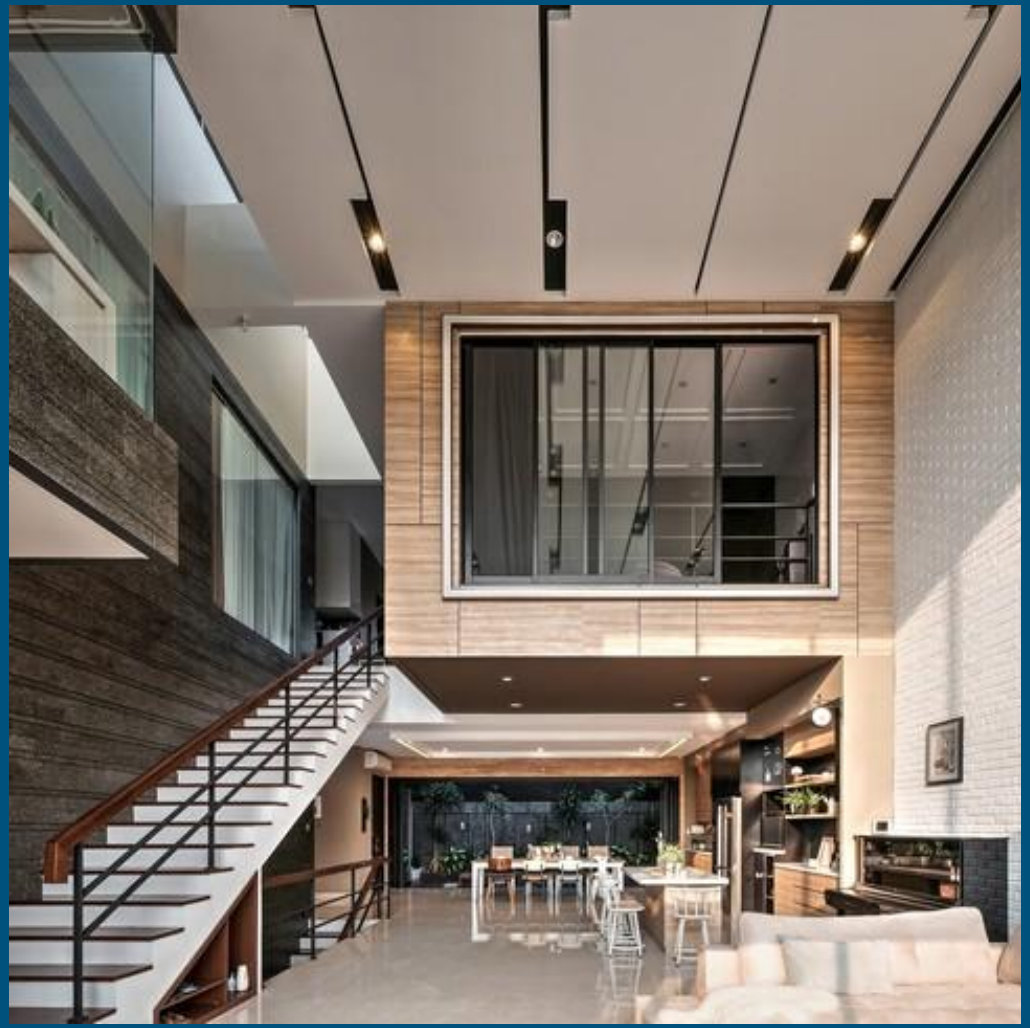
To begin, I built a simple CNN to test whether or not image level labels in the training data were sufficient for binary classification of things like chairs, tables, windows, etc.

They were not.

For object detection within scenes, I would need bounding-box training data.



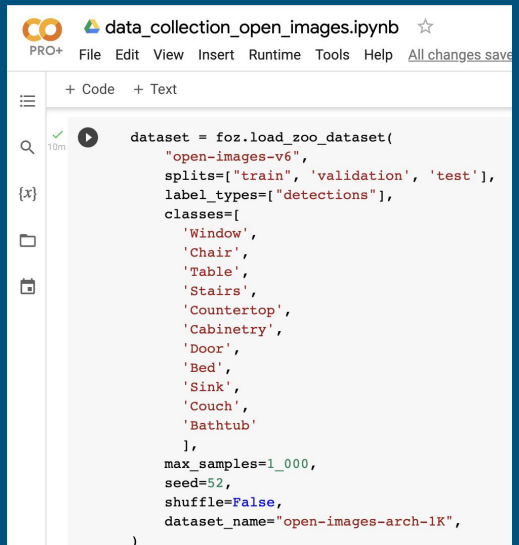
Bounding Box Approach

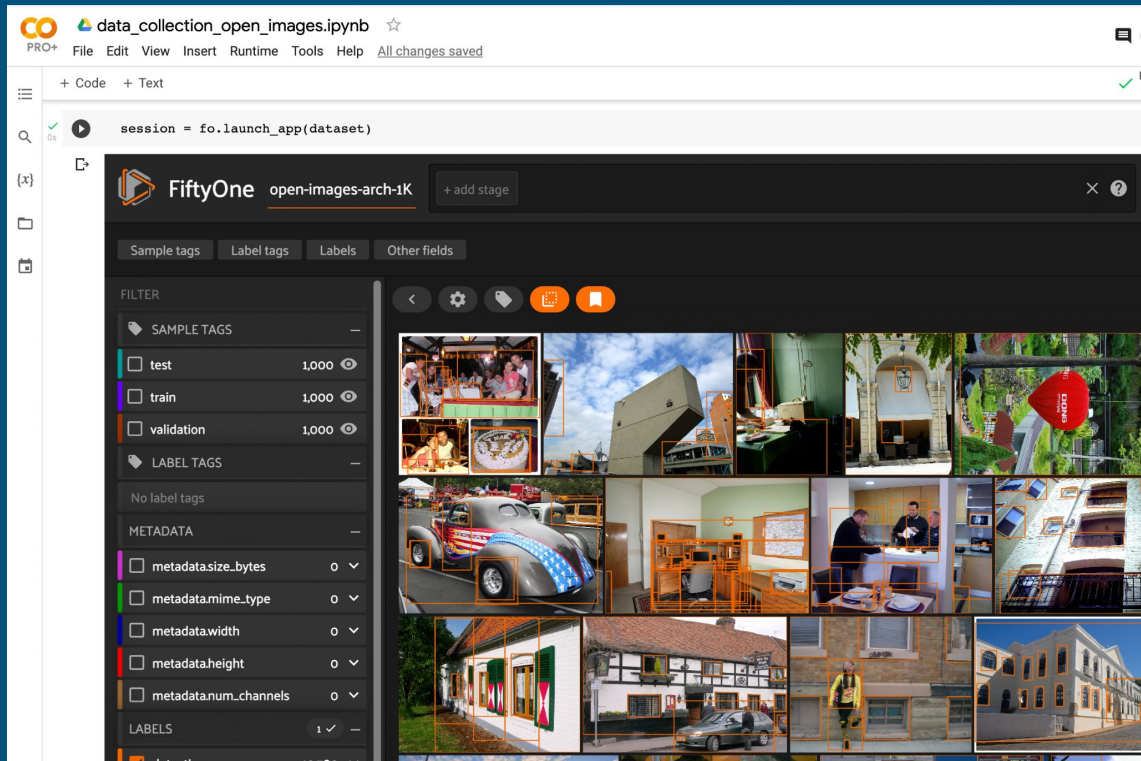


FiftyOne

https://voxel51.com/docs/fiftyone/index.html

Open source tool for building datasets with integration to Google Open Images

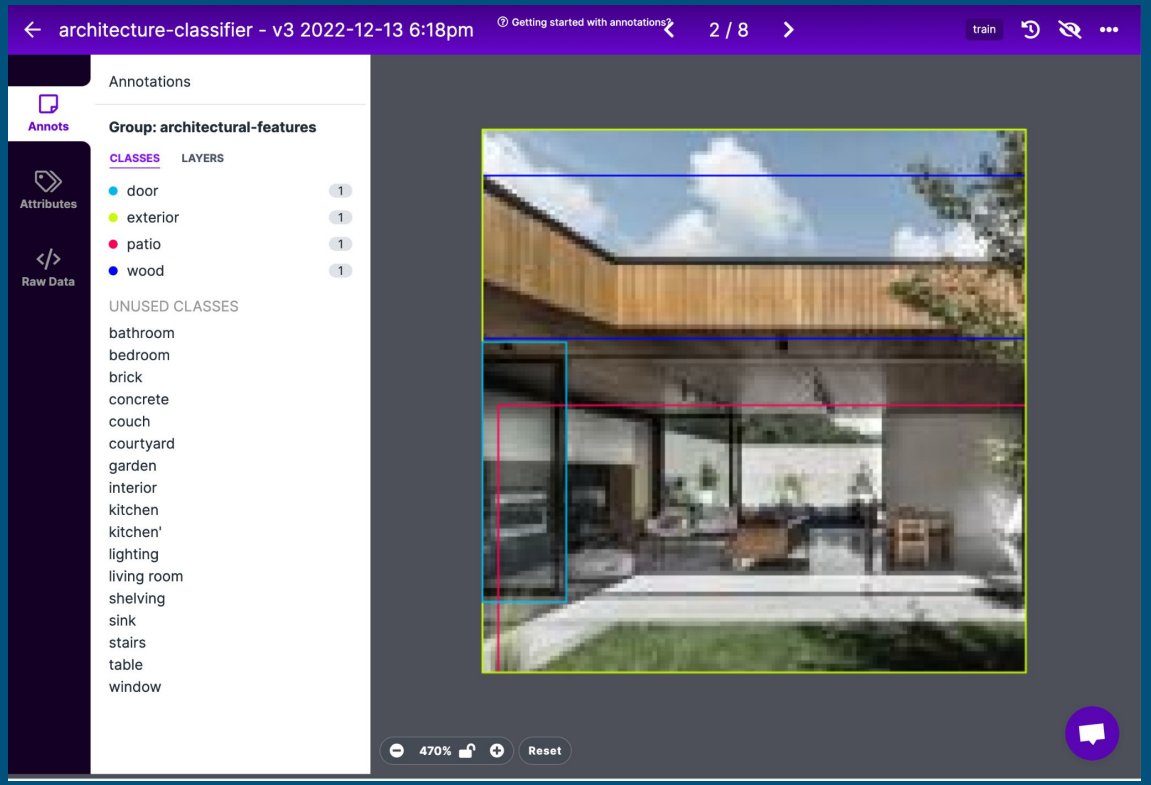




Roboflow

https://app.roboflow.com/

Suite of tools for building image datasets.



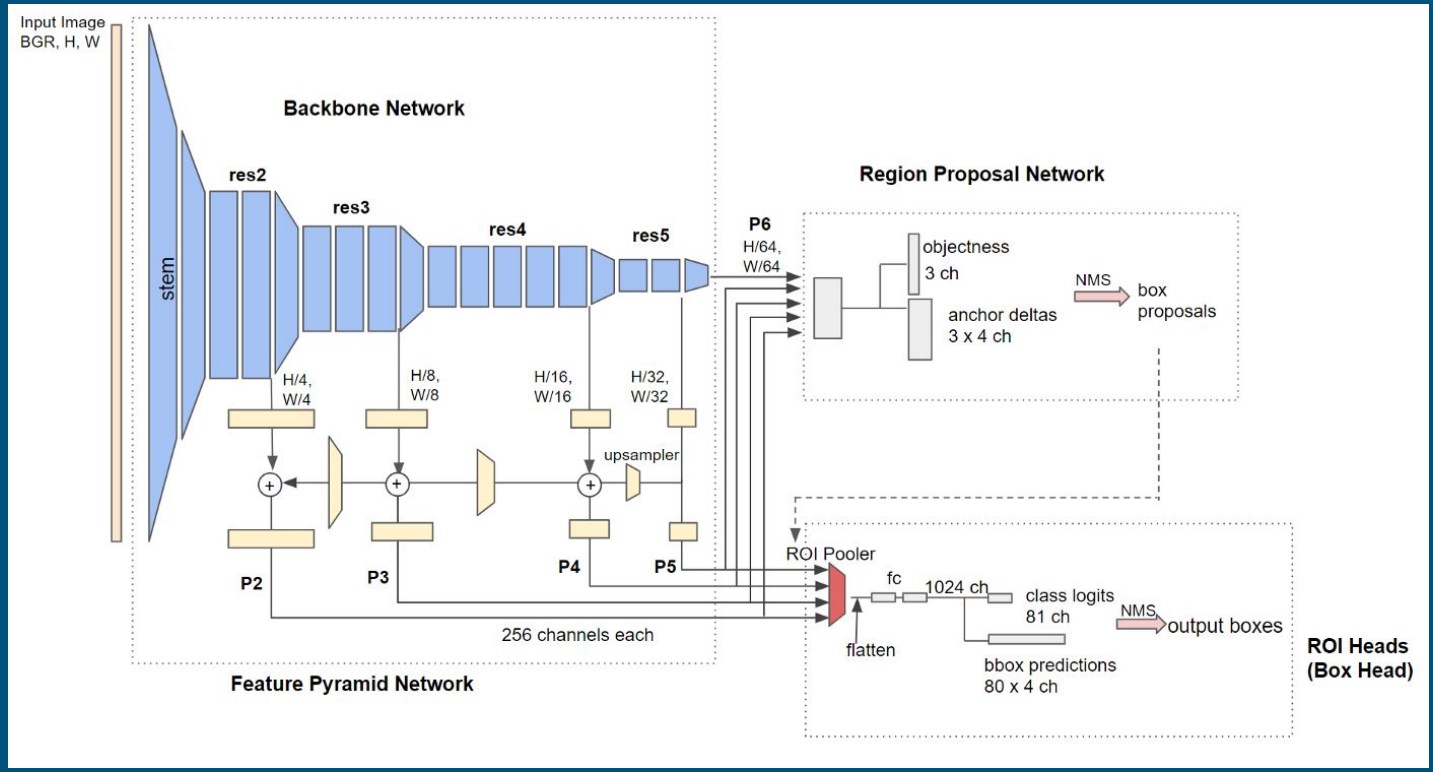
Detectron2

“Detectron2 is Facebook AI Research’s next generation library that provides state-of-the-art detection and segmentation algorithms.”

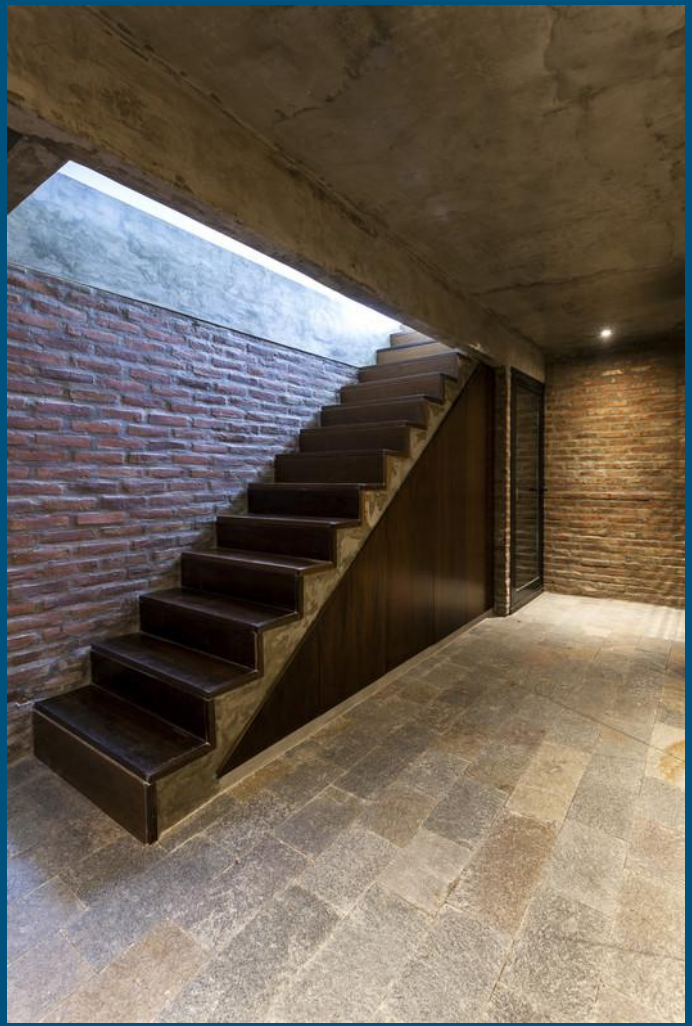
✅ COCO JSON	Segmentation	Complex Dependencies
Pascal VOC XML	Classification	✅ Colab Friendly
CreateML JSON	✅ Object Detection	Deprecated Dependencies

Detectron2 Architecture

From “Digging in to Detectron2” by Hiroto Honda



Metrics



AP by Class 3K Iterations on R101-FPN

Class	AP	Class	AP	Class	AP
Bathtub	9.199	**Bed**	46.962	**Cabinetry**	6.043
Chair	16.666	**Couch**	16.962	**Countertop**	6.754
Door	10.672	**Sink**	14.036	**Stairs**	6.754
Table	25.528	**Window**	14.957	**Brick**	0
Concrete	0	**Exterior**	0	**Interior**	0
Lighting	0	**Living Rm**	0	**Kitchen**	0
Patio	0	**Shelving**	0	**Wood**	0
Bathroom	0				

AP by Class 3K Iterations on X101-FPN

Class	AP	Class	AP	Class	AP
Bathtub	6.525 🔻	**Bed**	41.782 🔻	**Cabinetry**	6.891 ⬆️
Chair	16.003	**Couch**	14.312 🔻	**Countertop**	6.932
Door	9.948	**Sink**	19.959 🔻	**Stairs**	8.814 ⬆️
Table	26.952 ⬆️	**Window**	14.517	**Brick**	0
Concrete	0	**Exterior**	13.579 ⬆️⬆️	**Interior**	0
Lighting	0	**Living Rm**	0	**Kitchen**	0
Patio	3.387	**Shelving**	0	**Wood**	0
Bathroom	0				

AP by Class 6K Iterations on X101-FPN

Class	AP	Class	AP	Class	AP
Bathtub	5.718 🔻	**Bed**	46.427	**Cabinetry**	8.247 ⬆️
Chair	17.184 ⬆️	**Couch**	14.129 🔻	**Countertop**	7.442
Door	12.393 ⬆️	**Sink**	15.977 ⬆️	**Stairs**	15.156 ⬆️⬆️
Table	30.327 ⬆️	**Window**	14.341	**Brick**	0
Concrete	0	**Exterior**	0	**Interior**	0
Lighting	0	**Living Rm**	0	**Kitchen**	0
Patio	0	**Shelving**	0	**Wood**	0
Bathroom	0				

AP by Class 18K Iterations on X101-FPN

Class	AP	Class	AP	Class	AP
Bathtub	16.331 ⬆️⬆️	**Bed**	47.330	**Cabinetry**	9.973 🔻
Chair	18.185 ⬆️	**Couch**	20.511 ⬆️⬆️	**Countertop**	6.879
Door	14.067 ⬆️⬆️	**Sink**	23.853 ⬆️⬆️	**Stairs**	14.559 ⬆️⬆️
Table	32.178 ⬆️⬆️	**Window**	15.824 ⬆️	**Brick**	0
Concrete	0	**Exterior**	62.29 ⬆️⬆️⬆️	**Interior**	0
Lighting	0	**Living Rm**	0	**Kitchen**	0
Patio	12.22 ⬆️⬆️⬆️	**Shelving**	0	**Wood**	0


Window 74%
Window 85%
Window 84%
Window 88%
Window 79%
Window 86%
Window 86%
Window 93%
Window 87%
Window 86%
Window 86%
Window 86%
BK Bruton Knowles Property Consultants
Door 40%
Door 52%
ADVERTISE YOUR COMPANY HERE
07725 822 530
Johnsons THE CLEANERS

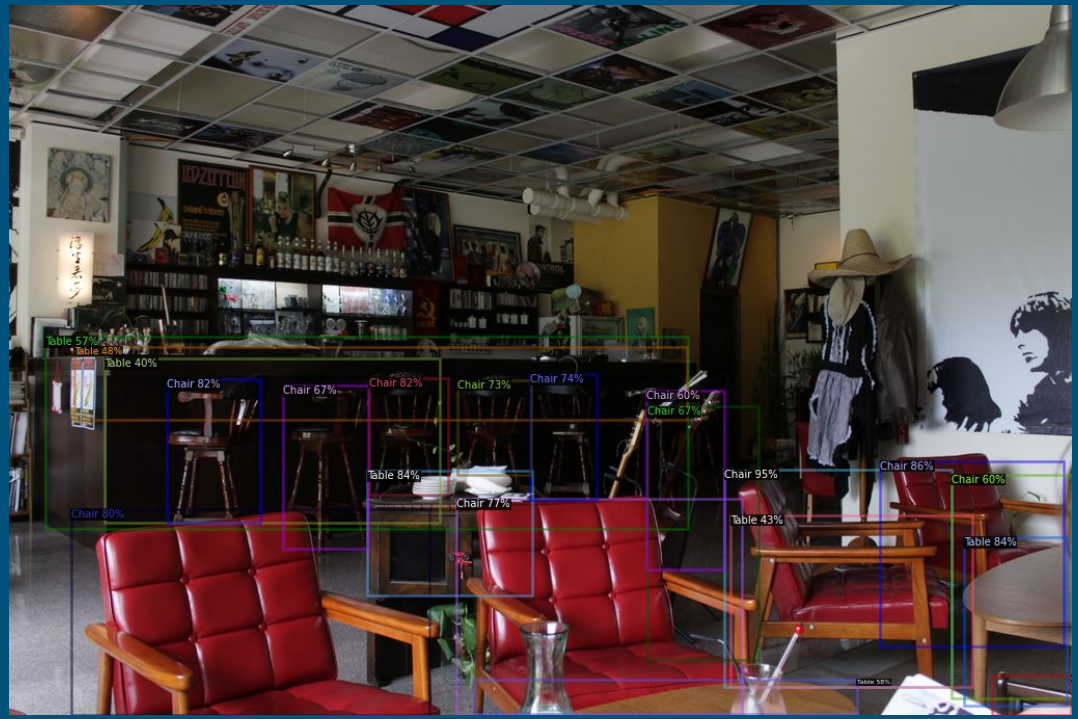
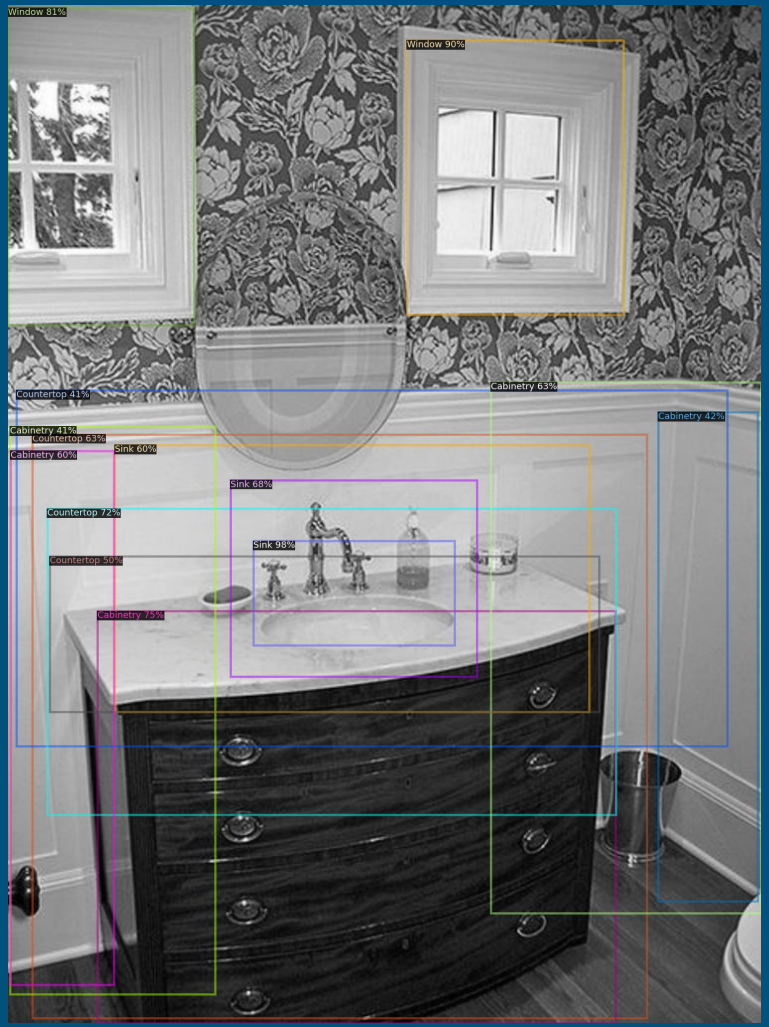
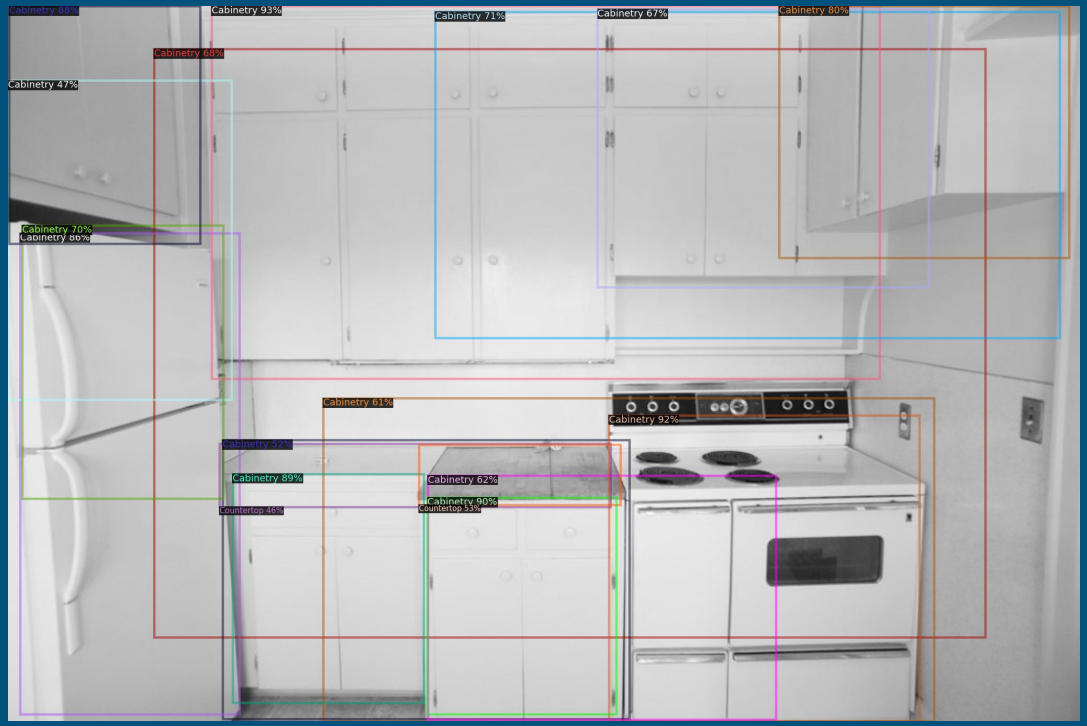


Table 57%
Table 48%
Table 40%
Chair 82%
Chair 67%
Chair 82%
Chair 73%
Chair 74%
Chair 60%
Chair 67%
Table 84%
Chair 77%
Chair 80%
Chair 95%
Chair 86%
Chair 60%
Table 43%
Table 84%

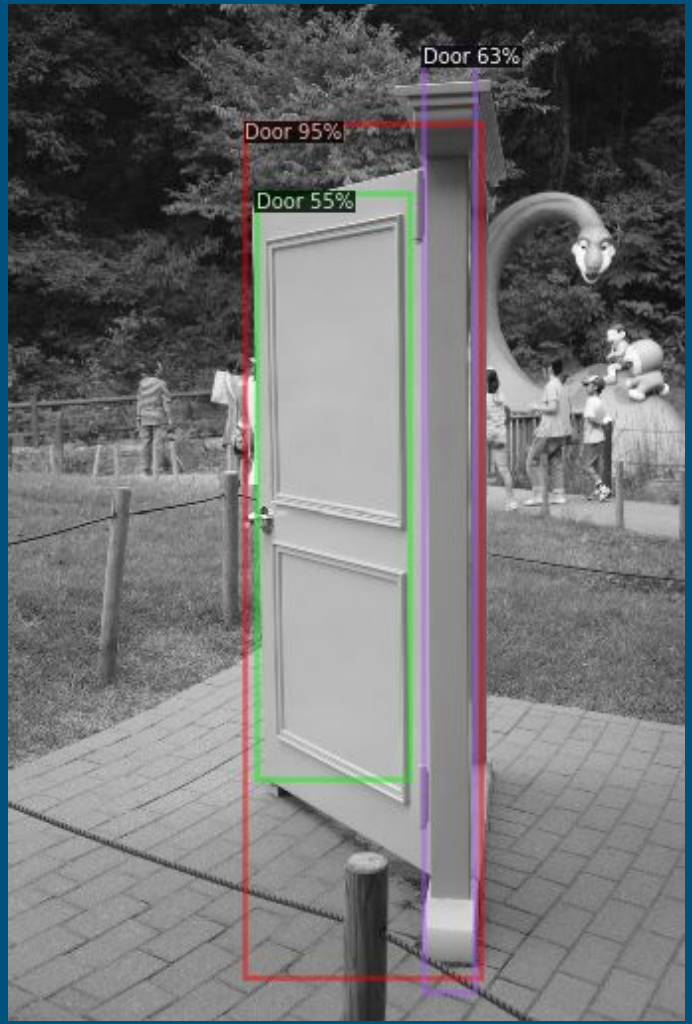

Window 81%
Window 90%
Countertop 41%
Cabinetry 63%
Cabinetry 42%
Cabinetry 41%
Countertop 63%
Sink 60%
Cabinetry 60%
Sink 68%
Countertop 72%
Sink 98%
Countertop 50%
Cabinetry 75%

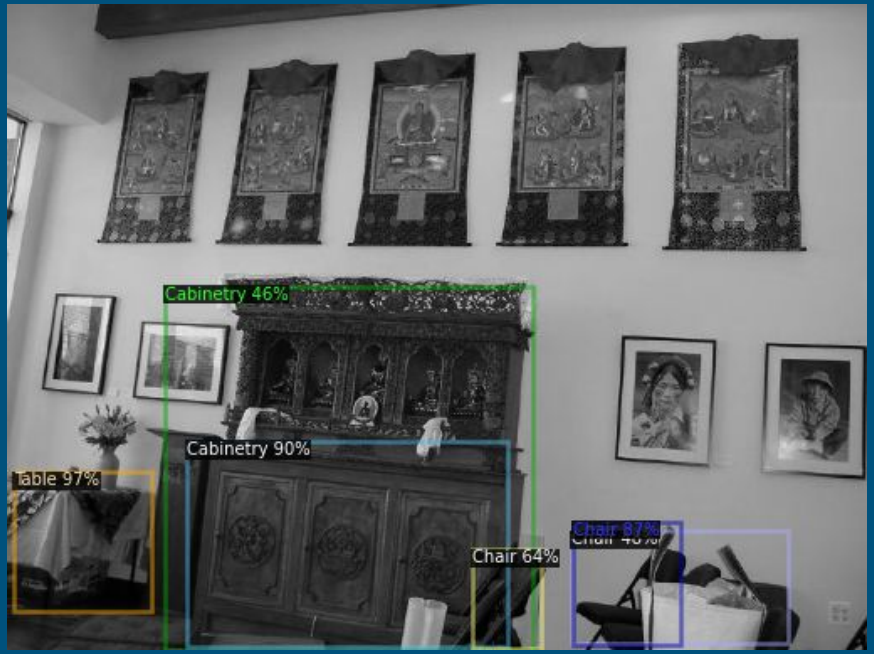

Cabinetry 88%
Cabinetry 93%
Cabinetry 71%
Cabinetry 67%
Cabinetry 80%
Cabinetry 68%
Cabinetry 47%
Cabinetry 70%
Cabinetry 86%
Cabinetry 61%
Cabinetry 92%
Cabinetry 52%
Cabinetry 89%
Cabinetry 62%
Countertop 46%
Cabinetry 90%
Countertop 53%

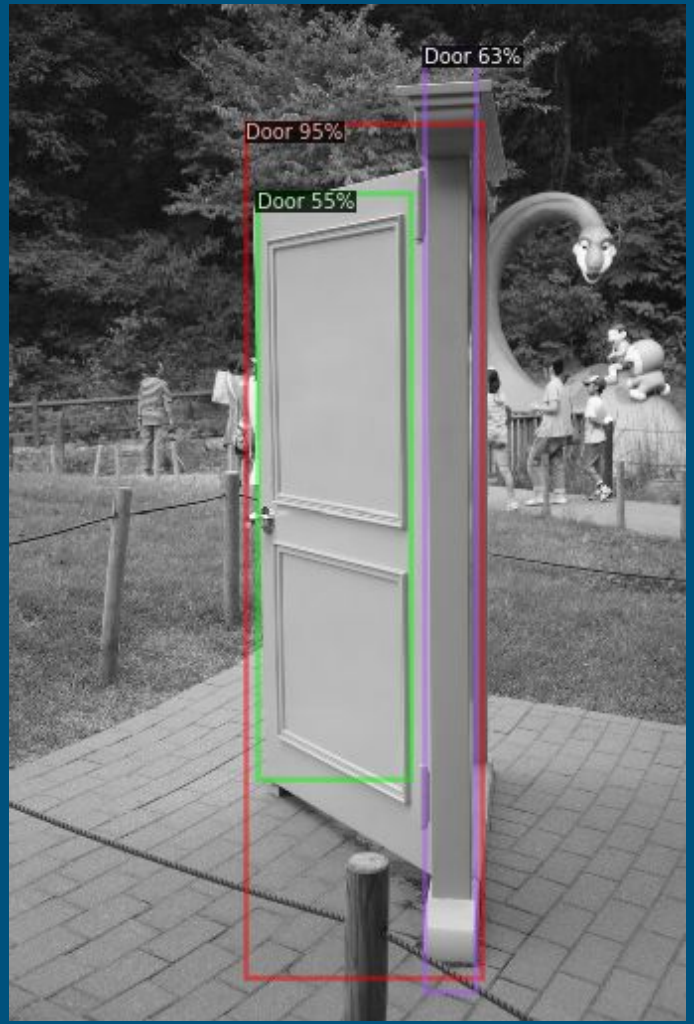

Window 64%
Window 63%
Window 53%
Table 44%
Table 99%


Window 54%
Door 53%
Window 62%
Door 50%
Door 72%
Door 44%
Window 66%
Door 55%
Door 41%

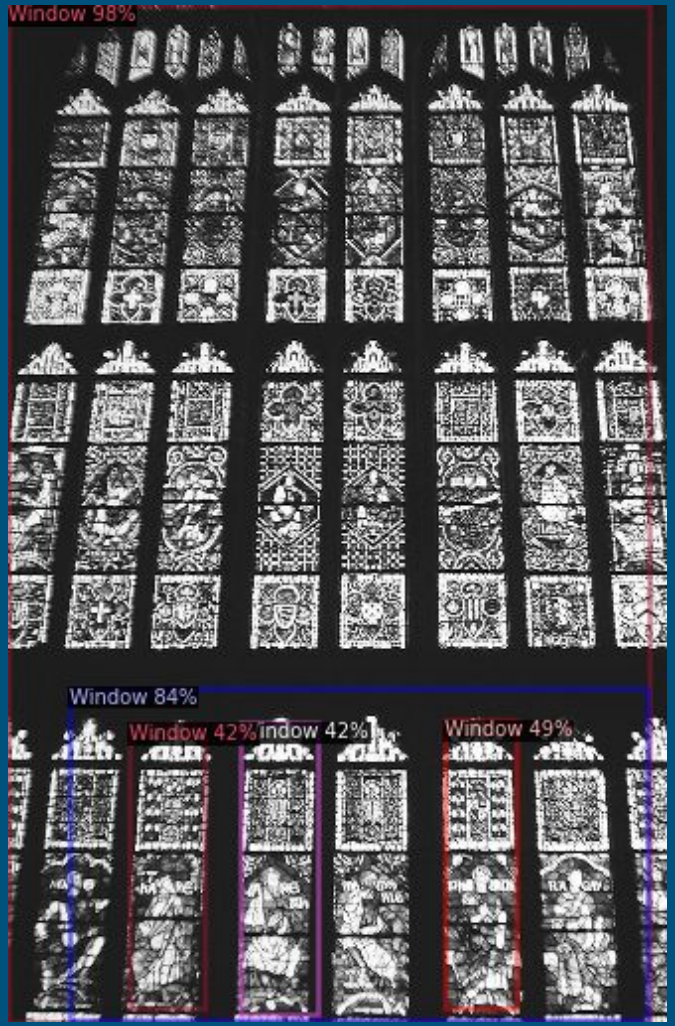

exterior 71%
patio 49%


Door 63%
Door 95%
Door 55%


Cabinetry 46%
Cabinetry 90%
Chair 64%
Chair 87%


Door 63%
Door 95%
Door 55%


Door 98%


Window 98%
Window 84%
Window 42%
Window 42%
Window 49%


Window 43%
Window 56%
Window 54%
Window 63%
Window 49%
Window 68%
Window 76%
Window 47%
Window 63%
Window 44%
Window 81%
Window 52%
Window 52%
Window 66%
Window 42%
Window 67%
Window 68%
Window 84%
Window 76%
Window 46%
Window 76%
Window 87%
Window 46%
Window 59%
Window 59%
Window 45%
Window 67%
Window 71%
Window 49%
2005. 05. 30


Window 66%
Window 95%
Window 93%
Window 96%
Window 91%
Window 57%
Stairs 69%
Stairs 40%
Door 88%


Window 88%
Window 78%
Door 77%
Window 54%
Stairs 87%

Conclusion

Conclusions

For the problem of classifying architectural photography:

- Developing a domain-specific model is possible with enough quality data
- Some abstract classifications, like materials or types of rooms, may need to be thought of differently than objects, such as including relationships between objects in the definition of a room.
- Similarly, a more complex set of classes such as "Minimalist Living Room" or "Gothic Window" (with parent categories) trained on many specific examples in each subclass may perform much better.

Conclusions (cont’d)

For object-detection generally:

- As computer vision approaches become more sophisticated, they also seem to be more opinionated, potentially creating a barrier to entry.
- Creating the right end-to-end workflow for data and evaluation can be more important than the specific modeling technique.

Thanks!

Resources and References

- Detectron2
- Digging into Detectron 2 by Hiroto Honda
- Architectural Photography
- Roboflow and Fiftyone

category	#instances	category	#instances	category	#instances
Bathtub	72	Bed	419	Cabinetry	975
Chair	6629	Couch	257	Countertop	416
Door	1161	Sink	228	Stairs	442
Table	3876	Window	21025		
total	35500				

FiftyOne

```
# Views of train, val, and test from the dataset
train_dir = "/content/drive/MyDrive/ga_data/20221213_7k/train"
test_dir = "/content/drive/MyDrive/ga_data/20221213_7k/test"
val_dir = "/content/drive/MyDrive/ga_data/20221213_7k/validation"

#labeled bounding boxes as json
label_field =  "detections"

dataset_type = fo.types.COCODetectionDataset

# Export the datasets
train_view.export(
    export_dir=train_dir,
    dataset_type=dataset_type,
    label_field=label_field
)
test_view.export(
    export_dir=test_dir,
    dataset_type=dataset_type,
    label_field=label_field
)
val_view.export(
    export_dir=val_dir,
    dataset_type=dataset_type,
    label_field=label_field
)
```