QUALITY CHECK AUTOMATION

OBSERVESIGN

https://github.com/Varun-AI-dev/Scale_ai_quality_check/tree/master

TABLE OF CONTENT

PROJECT OVERVIEW

- ObserveSign is focused on advancing the autonomous vehicle industry by improving traffic signal detection and understanding. To interpret and respond to traffic signals more accurately.
- ObserveSign has partnered with Scale AI to label a massive dataset of 250,000 images containing traffic signals.
- To train ML models we need accurate labeling for high-quality model training.
- Human-labeled data can be prone to errors, which may affect model performance if not corrected.
- To ensure data quality, Scale has enlisted assistance to develop a quality check script that will automatically flag any potential human-made errors in annotations.

PROJECT OBJECTIVES

01 **Automate Quality Checks:**

The project’s objective is to create a script that automates the quality control process by identifying any potential human-made errors in annotations and flagging them for correction. This automation will save time, reduce human errors, and increase the reliability of the dataset.

02 **Scalability**

While this project currently focuses on a small sample (8 tasks), the objective is to make the script flexible and robust enough to handle the full dataset of 250,000 images in production.

03 **Usability**

High-quality annotated data is the foundation of effective machine learning models. By ensuring that each annotation is accurate, the project contributes to training autonomous vehicle models that can correctly interpret traffic signals.

TRAFFIC SIGN SPEC

traffic_control_sign

Controls traffic flow; includes stop signs, yield signs, merge signs, traffic lights, etc.

information_sign:

Provides information like pedestrian crossings, speed bumps, directions, curves, etc.

construction_sign:

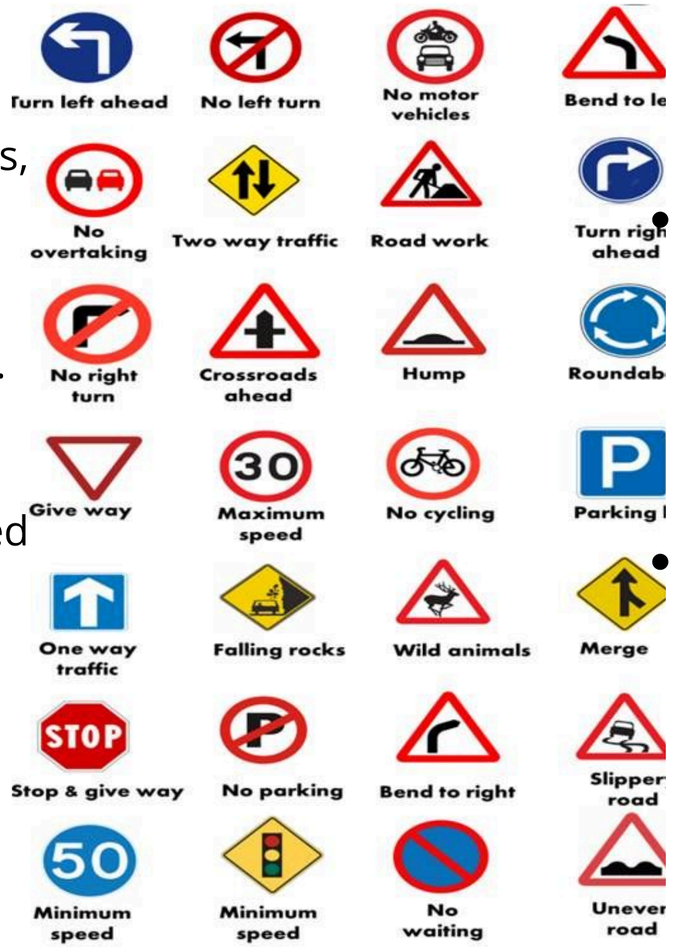
ndicates a construction zone; typically orange.

policy_sign:

Indicates laws or regulations, such as speed limits, parking restrictions, or no U-turn signs.

non_visible_face

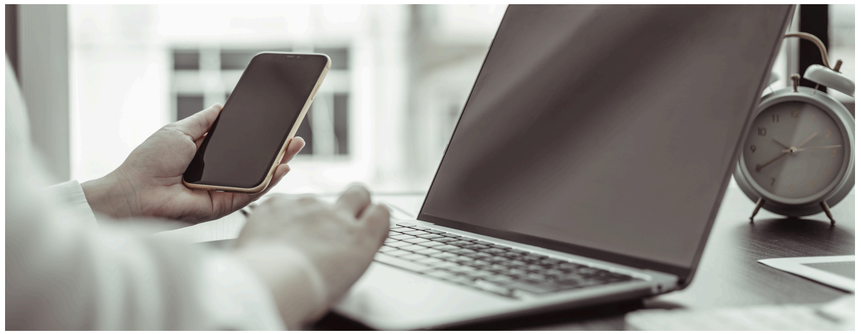
Used when the sign is perpendicular to the camera or only the back of the sign is visible.



Annotation Rules

- **Occlusion**: Measures how much of the sign is obstructed by other objects.
  - Values: 0%, 25%, 50%, 75%, 100% (0% is fully visible, 100% is fully hidden).
- **Truncation**: Measures how much of the sign is cut off by the edge of the image.
  - Values: 0%, 25%, 50%, 75%, 100% (0% is fully visible, 100% is entirely out of view).
- **Background Color**: Identifies the main color behind the sign text.
  - Values: white, yellow, red, orange, green, blue, other, not_applicable (for "non_visible_face" labels).

SCALE API



01 Scale API

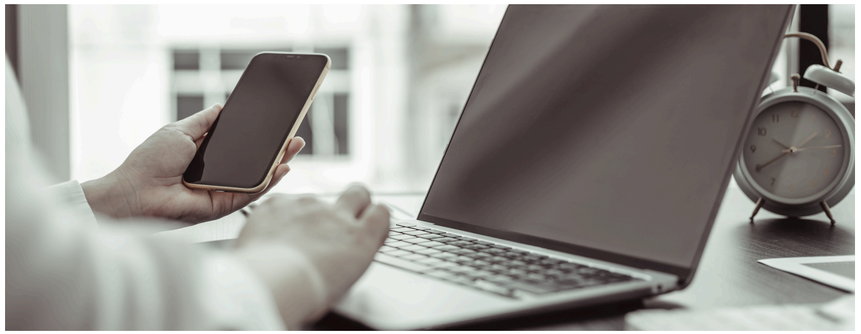
The Scale API is designed around the principles of REST. It uses resource-oriented URLs for predictable interactions, processes form-encoded request bodies, delivers JSON-encoded responses, and employs standard HTTP response codes, authentication protocols, and verbs.

02 API Access

Authentication Examples — Python

```python
import requests
from requests.auth import HTTPBasicAuth

url = "https://api.scale.com/v1/tasks"

headers = {"Accept": "application/json"}

auth = HTTPBasicAuth('{{ApiKey}}', '') # No password

response = requests.request("GET", url, headers=headers, auth=auth)

print(response.text)
```

THE SCRIPT



01 Load Data

```python
import json
import csv  # Required for writing data to CSV files
import requests  # For downloading images from URLs
import os  # For checking if files exist on disk
from PIL import Image, ImageDraw  # For opening, modifying, and saving images

# Load tasks from a JSON file
def load_tasks(file_path):
    # Open the specified file and load its contents as JSON data
    with open(file_path, 'r') as file:
        return json.load(file)  # Returns the JSON data as a Python dictionary
```

02 Quality Check Criteria

Audit results to flag frequently rejected tasks, bounding box area deviation to catch incorrectly sized annotations, label accuracy to ensure correct sign categorization, and occlusion level to exclude highly obstructed signs. These checks improve annotation reliability, aiding in accurate model training.

02 Assign Ratings

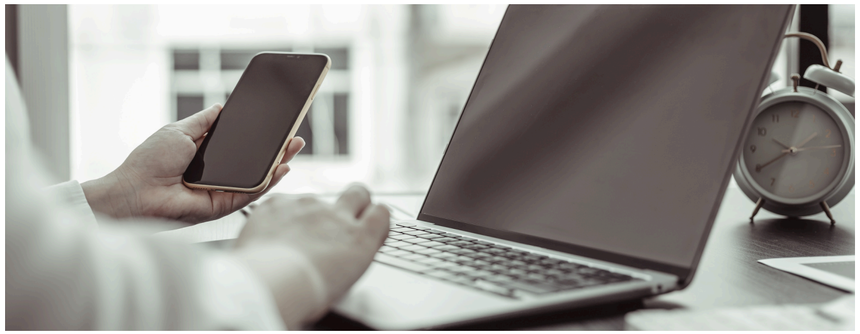
```python
if final_score >= 0.9:
    final_rating = "Gold"
elif final_score >= 0.8:
    final_rating = "Silver"
else:
    final_rating = "Bronze"
```

OUTPUT TO CSV

	A	B	C	D	E	F	G	H
	Task ID	Audit Rating	Area Rating	Label Rating	Occlusion Rating	Final Rating	Annotated Image Path	Issues
	5f127f55fc	1	0.5	1	1	Silver	annotated_5f127f55fdc4150010e37244_annotated.png	Bounding box area deviation issues: 52.38
	5f127f5ab	0	1	1	1	Bronze	annotated_5f127f5ab1cb1300109e4ffc_annotated.png	Audit rejection rate: 100.00%
	5f127f5f3a	0	1	1	1	Bronze	annotated_5f127f5f3a6b100017232099_annotated.png	Audit rejection rate: 100.00%
	5f127f643	0.5	1	1	1	Silver	annotated_5f127f643a6b1000172320a5_annotated.png	Audit rejection rate: 50.00%
	5f127f671	0.5	1	1	1	Silver	annotated_5f127f671ab28b001762c204_annotated.png	Audit rejection rate: 50.00%
	5f127f699	0	1	1	1	Bronze	annotated_5f127f699740b80017f9b170_annotated.png	Audit rejection rate: 100.00%
	5f127f6c3	0.5	1	1	1	Silver	annotated_5f127f6c3a6b1000172320ad_annotated.png	Audit rejection rate: 50.00%
	5f127f6f2(	1	1	1	1	Gold	annotated_5f127f6f26831d0010e985e5_annotated.png	


macy's
macy's
Post
ONE WAY
MONTGOMERY
POST
9
8900
Howard
NO STOPPING
ANY TIME
Yerba Buena/Moscone
South Park
HOWARD
PARKING
ANY
TIME
OVER 3 TONS
EXCEPT ONE
BLOCK FOR
DELIVERIES
STOP

FUTURE IMPROVEMENTS



01 Edge Proximity and Overlapping Boxes

02 Cloud Scalability

03 Color Analysis with OpenCV

04 Use of LLM Model

05 Pre-trained Machine Learning Model

THANK YOU