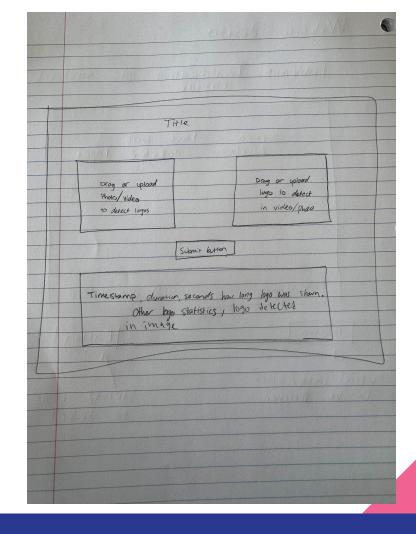
Trademark Identification & Analysis Engine

Team 11 Sprint 2 Lane Keck, Logan Taggart, Caleb Stewart

Website Design



Requirements

Team 11 Requirements

R2 Create Website for Uploading files to backend
R3 Setting up machine learning framework
R4 Train Machine learning model
R5 Allow users to upload live video feed to website

R6 Send Feedback to user on logo recognition

Dataset Analysis

We need to find a good dataset that represents hundreds/thousands of logos, with a bounding box around each logo.

Luckily, the LogoDet-3k dataset exists. This dataset contains 3,000 different logo categories, with over 150,000 images and about 200,000 bounding boxes. This will be plenty of data to train our model.

Sub-Category	Images	Objects	
932	53,350	64,276	
604	31,266	37,601	
432	24,822	30,643	
371	15,513	20,016	
224	9,675	12,139	
213	10,445	12,791	
111	5,685	6,573	
66	3,953	5,041	
47	3,945	5,185	
3,000	158,652	194,261	
	932 604 432 371 224 213 111 66 47	932 53,350 604 31,266 432 24,822 371 15,513 224 9,675 213 10,445 111 5,685 66 3,953 47 3,945	932 53,350 64,276 604 31,266 37,601 432 24,822 30,643 371 15,513 20,016 224 9,675 12,139 213 10,445 12,791 111 5,685 6,573 66 3,953 5,041 47 3,945 5,185

YOLO Model

We decided to use the YOLO algorithm for our object detection. It offers impressive processing speeds, making it suitable for real-time object detection applications. This model is very versatile.

YOLOv8 uses PyTorch to train the models, so our dataset needs to be in this format

Format Description

Below, learn the structure of YOLOv8 PyTorch TXT.

Each image has one txt file with a single line for each bounding box. The format of each row is

class_id center_x center_y width height

Training the YOLO Model

Thanks to the kaggle website, there already exists a format of the dataset that can be used with YOLO and PyTorch.

https://www.kaggle.com/datasets/julianskobic/logodet-3k-yolo/data

All that is left is to train the dataset with the YOLO algorithm... but PyTorch is optimized for NVIDIA graphic cards using CUDA, and we AMD graphic cards.

Instead we can use Google Colabs GPU servers to train our model.

Frontend Design

We have decided to utilize React to build the web application that users will interact with.

This will give us the ability to dynamically update the web page as we send and receive data to and from the backend without needing to reload the page.

Backend Design

For the backend we will be using Python alongside with the Flask framework to allow us to handle the API requests that are made from our frontend to our different routes/endpoints.

By creating the API, we will be able to give our program an organized, easy, and effective way of interacting with our YOLO model, whether it be sending input data or retrieving and calculating the statistics based on the model's data output.

Possible Challenges

- Expensive to train model on dataset
- How to host our frontend website and then the backend
- Handling the POST requests for large video files
- Need to figure out how to compare logos found in video to the logo we want to compare

Moving Forward

- Get YOLO model working for a still image as proof of concept
- Get YOLO model working for a video
- Use the video to generate statistics of logo occurrences
- Create an outline of what our backend API is going to look like
- Create the frontend website and get it to properly integrate with the backend