SIMILAR CAR FINDER

A Web Application to Find Similar Cars for Rent

Developed as part of the Full-Stack Development course at Mission Ready HQ (2023)

Elena Kroupkin

INTRODUCTION

The Similar Car Finder is a web application designed to assist customers in finding rental cars based on visual similarity. It was developed for Turners, enabling customers to upload an image or select a car they like, and the app recommends similar cars available for rent. The app is especially useful for customers who may not know the exact make or model they are looking for but can visually identify the type of car they want.

The primary focus of this project was on the backend functionality, and the user interface and user experience were not prioritized.

Key Features

- **Upload a Picture of a Car**: Users can upload an image of a car, and the app suggests visually similar cars available for rent.
- **AI-Powered Visual Analysis**: The app leverages Azure Cognitive Services to analyze features such as color, shape, body type, and more, enabling precise car recommendations.
- **AI-Centric Focus**: While the application is functional, the focus of the project was on AI model training and development. As such, the user interface (UI) is minimal, with most attention directed toward refining the model's accuracy and performance.
- **React.js Front-End**: A simple React-based front-end is implemented primarily for image uploads and displaying results, emphasizing functionality over visual design.

Technology Stack

- **Azure Computer Vision API**: Extracts visual features from car images such as color, shape, and body type.
- **Custom Vision AI**: Trains a custom model to classify and tag car images.
- **React.js**: Provides the front-end for image upload and result display.
- **GitHub**: Used for source code management and project collaboration.

Solution Architecture

The Similar Car Finder is built around an image-based recommendation system that integrates cloud-based AI services with a locally stored image database and a React.js front-end. The cloud services handle the visual analysis, while the local image database provides efficient car recommendations.

Step-by-Step Process

- **Image Upload**: Users upload an image of the car they like, either via drag-and-drop or by providing a URL.
- **Image Analysis**: The uploaded image is sent to the Azure Computer Vision API, which extracts features such as color, shape, and body type.
- **Local Database Comparison**: The extracted features are compared with a database of pre-tagged car images stored locally on the developer's computer. This local storage was used for efficient lookups and recommendations based on the user's input.
- **Recommendation**: The app recommends similar cars by matching visual features from the local database.

Image Classification Process

Feature Extraction
The app uses Azure Computer Vision API to extract key visual features from the uploaded car images:

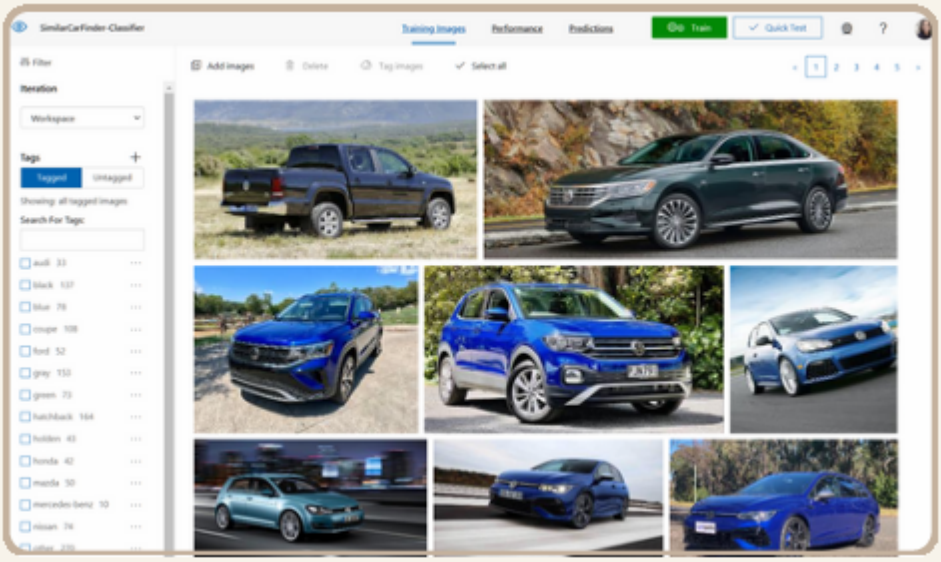
- Color
- Shape
- Body Type (e.g., Sedan, SUV, Sports Car, Pickup Truck)
- Number of Doors

Classification and Categories
Using the Custom Vision AI, the system classifies cars based on:

- Make: e.g., Honda, Toyota, Ford
- Model: e.g., Civic, Corolla, Mustang
- Body Types: Sedan, SUV, etc.

Training the Custom Model

- **Tags**: 25 different tags were used to label car features.
- **Training Images**: 1031 images were used in the training process across 9 iterations.
- **Challenges**: Some tags were undertrained due to a lack of sufficient data.



Front-End Features (React.js)

1. **Image Upload Component**: Users can upload images of cars they like using two methods:
   * **Drag & Drop**: Easily drag and drop an image into the upload area.
   * **URL Input**: Users can also provide a URL of a car image to analyze.

2. **Result Display**: The app shows cars available for rent that are visually similar to the uploaded image.

3. **Categorization**: Results are categorized based on key features such as color, make, and body shape.

API Usage

POST Request: Sends image data to the Azure Computer Vision API.

Response: Retrieves the car's features (e.g., make, color, body type).

Example Request URL:
https://similarcarfinder.cognitiveservices.azure.com/vision/v3.2/analyze?visualFeatures=Color,Tags

API Usage

I used API Console to quickly try the API without writing code

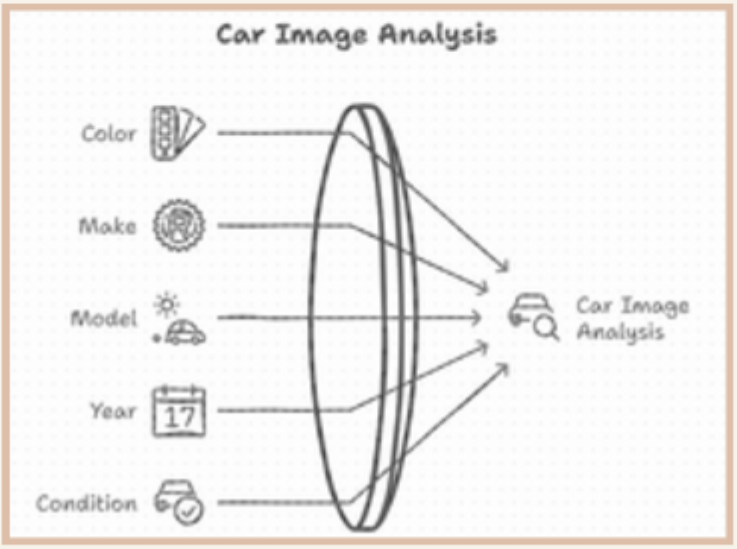
Model Training and Prediction

Data Collection: Used Custom Vision AI to gather and tag car images.

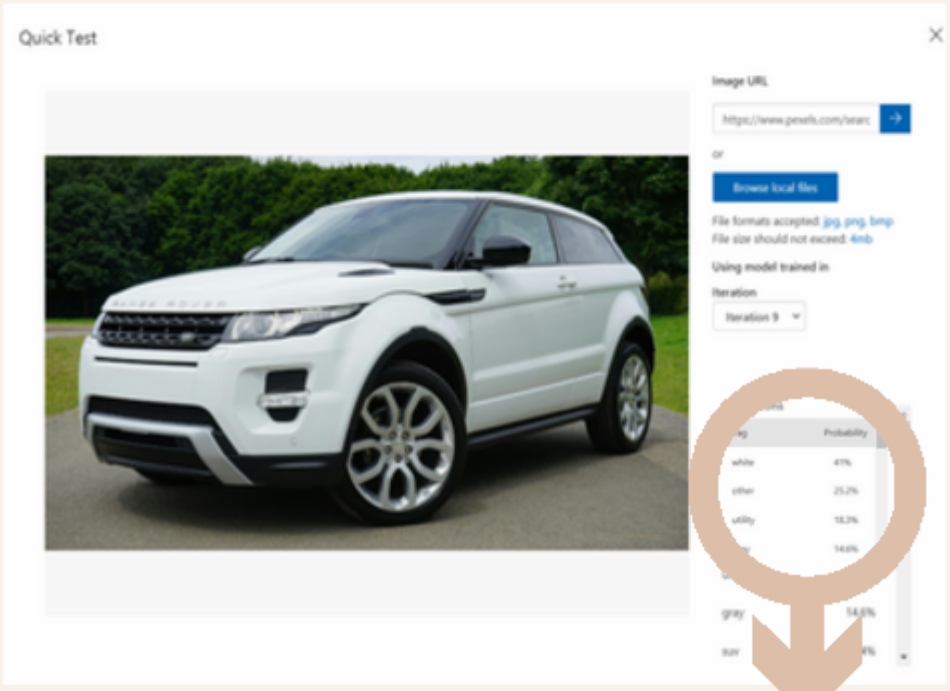
Model Training: Trained the model using labeled car images to identify features like color, make, and body type.

Prediction: Once an image is uploaded, the model predicts the car's make and type based on the extracted features.

Result Display: The app displays similar cars based on the top matches in each category.



Prediction


Quick Test
Image URL
or
Browse local files
Using model trained in
Iteration
Iteration 9


Predictions
Tag
Probability
white 41%
other 25.2%
utility 18.3%
gray 14.6%
suv 12.4%
black 11.9%
volkswagen 9.2%
holden 8.7%
ford 8.6%
green 8.2%
toyota 7.8%
wagon 6.4%

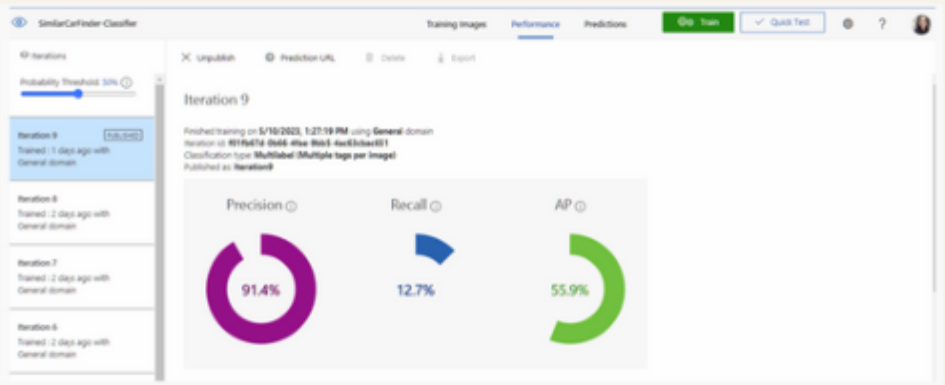

nissan 5.3%
yellow 5.1%
sedan 5%
mazda 4.8%
hatchback 4.6%
coupe 4.6%
blue 4.5%
blue 4.5%
van 4.4%
peugeot 3.8%
mercedes-benz 3.5%
red 3.5%
honda 2.9%
audi 2.7%

Challenges Faced

Incomplete Training: Some categories (tags) were undertrained due to limited training data, resulting in inconsistent recommendations.

Data Management: Handling a large volume of images and tags for training was challenging.

Cost Management: Azure cloud resources were optimized to minimize unnecessary costs.



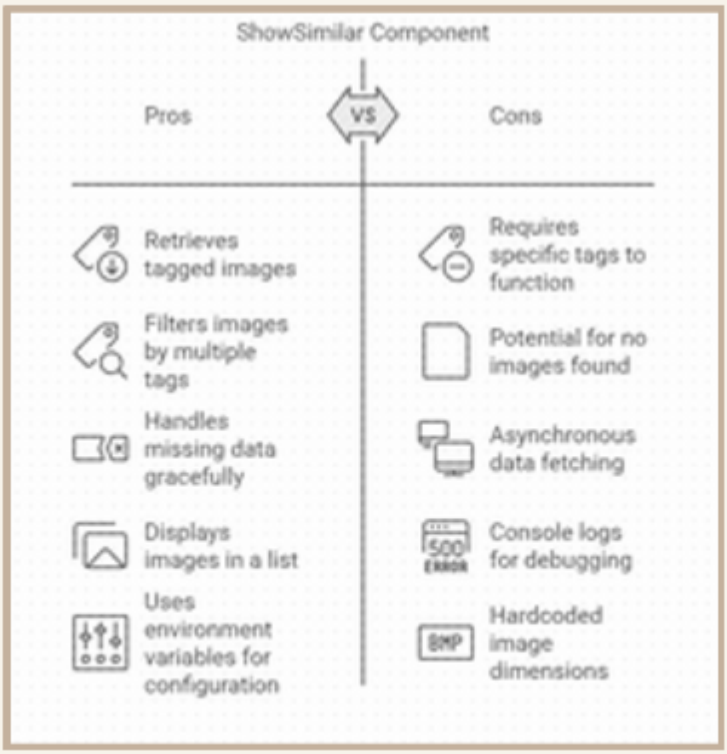
Final Results

Prototype: A functional prototype that successfully finds and recommends similar cars based on visual inputs.

Next Steps: Improve the model's accuracy by training with a larger, more diverse dataset and refining feature extraction for better predictions.

Conclusion

The Similar Car Finder demonstrates how AI-powered image recognition can simplify car rental decisions by providing users with visually similar car options. The project offers valuable hands-on experience in using Azure Cognitive Services, image processing, and model training.



Future Applications

The system can be expanded to recommend not just cars but other vehicles or objects based on visual similarity using similar AI techniques.

GitHub Repository

The complete source code and related documentation can be found at the following repository:

https://github.com/pocpat/similar-cars/