CAPSTONE SUMMER-FALL 2021 TEAM B

Team Members: Ashley Maddix, Chiharu Akiyama, Ritvik Chitram, Sanad Thapa, Tim Hall, Zhe Han

Sightline Applications

Provides onboard video processors for advanced camera systems and applications.

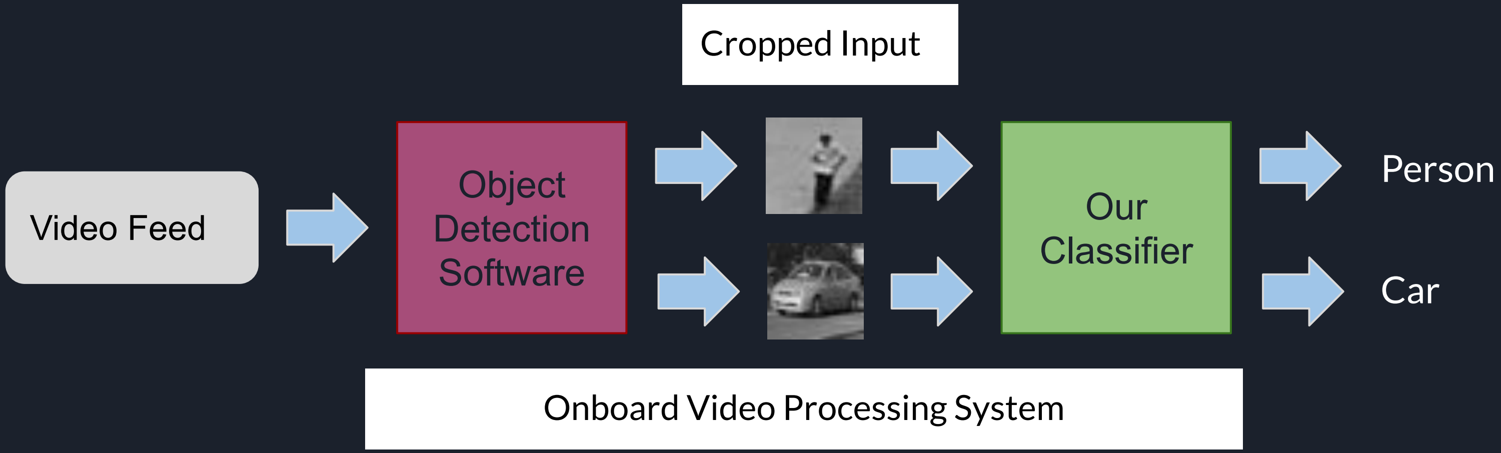
Sponsor   Leila Kaneda

Roles

Team Lead:	Tim
Primary Testers:	Ritvik Sanad Zhe
Secondary Testers:	Ashley Chiharu
Dataset Validators:	Ashley Chiharu
Documenters:	Everyone
Scripters:	Zhe Tim

what does it do?

The base image classifier that we will be tuning for SightLine Applications must be able to analyze drone/camera footage and successfully classify vehicles, humans, background (stationary elements of the scene) as well as others (dynamic/moving elements of the scene, e.g. boats, planes, pets, etc.)



Cropped Input

Video Feed

Object Detection Software

Our Classifier

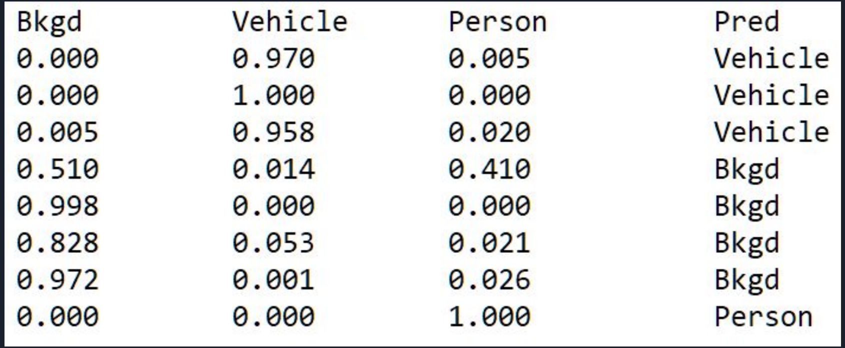
Person

Car

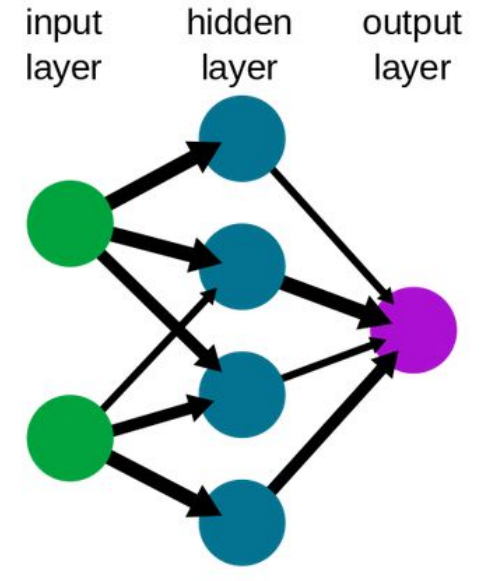
Onboard Video Processing System

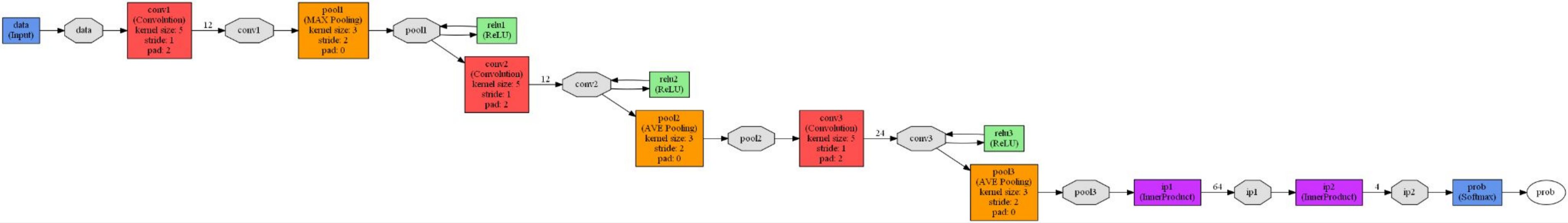
The Classifier

- Convolutional Neural Network (CNN)
- 12 Layers



```
Bkgd       Vehicle    Person       Pred
0.000      0.970      0.005        Vehicle
0.000      1.000      0.000        Vehicle
0.005      0.958      0.020        Vehicle
0.510      0.014      0.410        Bkgd
0.998      0.000      0.000        Bkgd
0.828      0.053      0.021        Bkgd
0.972      0.001      0.026        Bkgd
0.000      0.000      1.000        Person
```



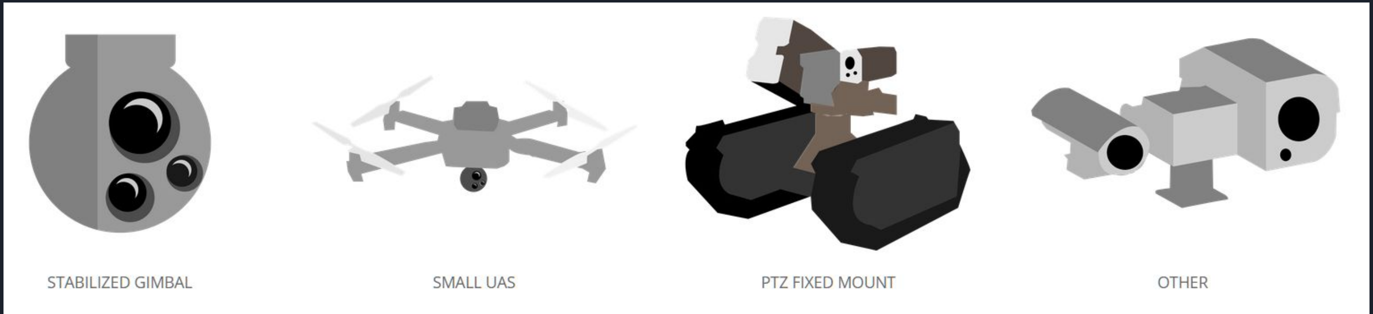

data (Input)
data
conv1 (Convolution) kernel size: 5 stride: 1 pad: 2
12
conv1
pool1 (MAX Pooling) kernel size: 3 stride: 2 pad: 0
pool1
relu1 (ReLU)
conv2 (Convolution) kernel size: 5 stride: 1 pad: 2
12
conv2
relu2 (ReLU)
pool2 (AVE Pooling) kernel size: 3 stride: 2 pad: 0
pool2
conv3 (Convolution) kernel size: 5 stride: 1 pad: 2
24
conv3
relu3 (ReLU)
pool3 (AVE Pooling) kernel size: 3 stride: 2 pad: 0
pool3
ip1 (InnerProduct)
64
ip1
ip2 (InnerProduct)
4
ip2
prob (Softmax)
prob

Why is it needed? Who will use it?

Sightline customers use their classifiers for unique reasons. Typically used for

- Intelligence/surveillance/Reconnaissance
- Security
- Inspections

This particular classifier system is ment to work onboard advanced drones/cameras.

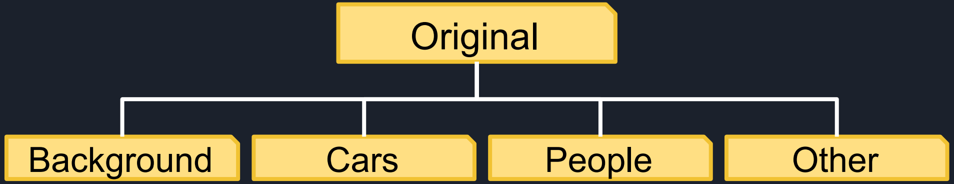


Deliverables

- Classifier settings file
- Our Dataset
- Documentation
  - Testing results and our observations
  - Sources

Dataset

- .png format
- 32x32 pixels
- Black and white



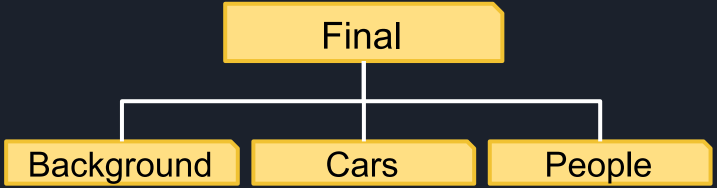
Original

Background

Cars

People

Other



Final

Background

Cars

People

Constraints

- Built using Sightlines proprietary classifier
- Identify vehicles and people from ground and birds eye view w/ > 95% accuracy
- Need to build our own dataset
    - Use images from ground and birds eye view
    - Images have to be free for commercial use
    - Minimum of 1000 images
    - cropped/resized/converted to black and white

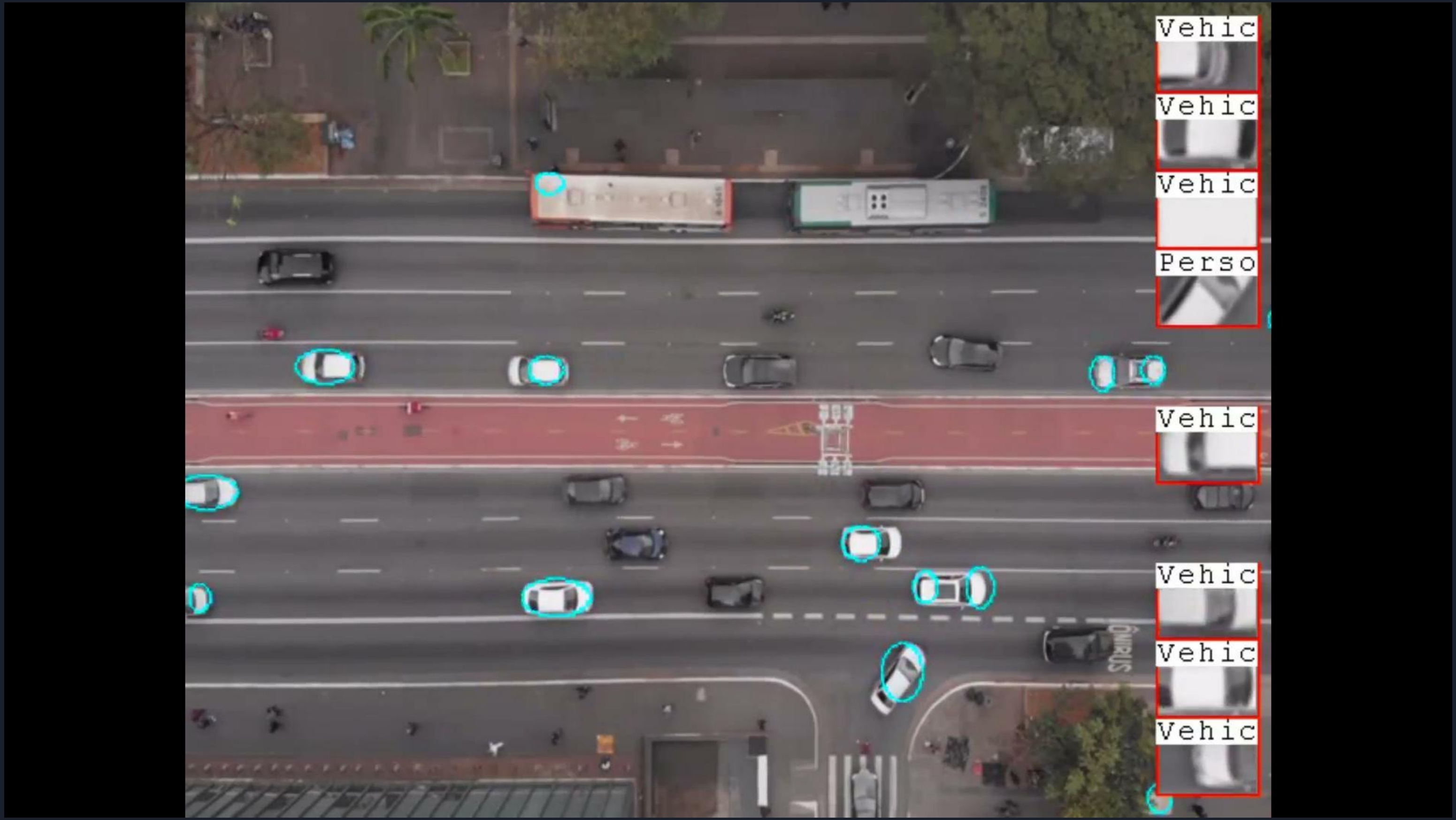
Features

- Runs in a Low voltage and low processing power environment
- Runs in python 3.5
- Database of 1000+ un-augmented images
- Documentation of our tests, results and observations

Features

- Runs in a Low voltage and low processing power environment
- Runs in python 3.5
- Database of 4,800+ un-augmented images
  - (9,000+ w/augmentation)
- Documentation of our tests, results and observations
  - 10 different testing situations
  - Results for 500+ tests (1200+ ran overall)
- Scripts to
  - Preprocess Raw Images
    - Convert images to B&W, resize and name files to our convention
  - Dataset Augmentation
  - Testing Automation
  - Model Analysis

Demo



Process

- Project was managed in a scrum environment
- Weekly sponsor check ins

1. Learn the library/Preparing for development
   a. Set up testing environment
   b. Sightline Tools
   c. Tools for building the Dataset
2. Create a dataset
   a. Finding image datasets free for commercial use
   b. Creating our own image datasets
   c. Cropping and resizing images
   d. Manual review
3. Train a model
   a. Start experimenting with a setting and take the best results
   b. Experiment with another setting and take the best of those
   c. Repeat to refine
4. Create Documentation/ package deliverables

Planned Schedule

Week									
1	2	3	4	5	6	7	8	9	10
Build Dataset			Model Training			Buffer		Documentation & Refinement	

Actual Schedule

Week									
1	2	3	4	5	6	7	8	9	10
Build Dataset			Model Training			Buffer		Documentation & Refinement	

Week									
1	2	3	4	5	6	7	8	9	10
Build Dataset			Model Training		Refactor Dataset	Model Training	RD	Documentation & Refinement	

Changes to the original plan

- Ditching the Split team plan for a more brute force approach
- Increasing dataset size
- Adding the background and other class
- Sticking with the 3 class model

The Bad

- Dataset refactors during testing
- Didn’t get to test our model in the wild till later than we planned
- Model was getting good accuracy with our data but not good enough in the wild
- Inconsistent results

Lessons learned

- Bigger is better for the dataset
- Optimizing our process so it exploited hardware capabilities
- Scripts are nice

Thank You!