C2TSR: Concurrent Canada-based Traffic Signpost Recognition System

Department Of Computer Science

CS 842: Introduction to Data Science : Project

Aug 19, 2020

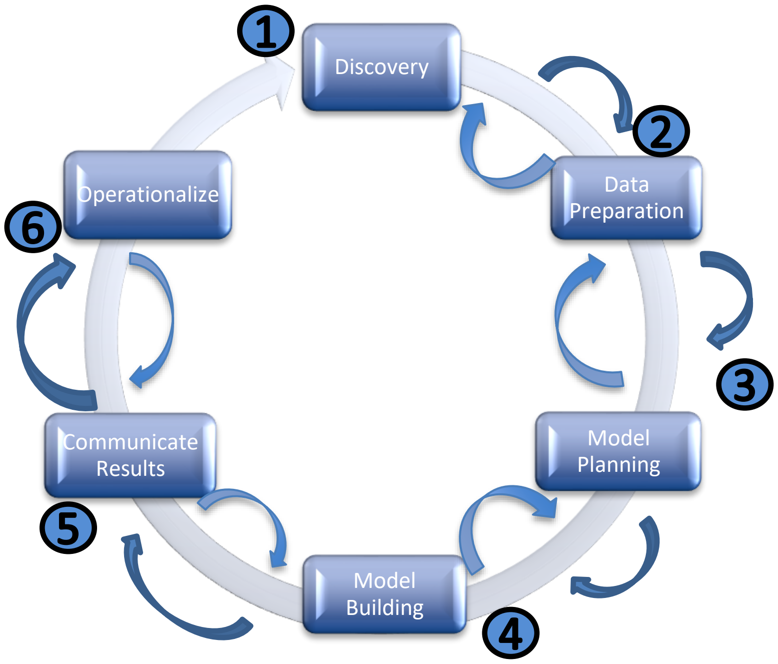
Ms. Suby Singh (ssz389@uregina.ca)

University of Regina

Introduction

- A deep learning model
- Detect and classify the traffic signposts (signals and signs)
- Based on Canada traffic signs
- Motivational factors of introducing this model:
  - ➢ A prelude of fully automated driving or semi-automotive driving systems (e.g. ADAS)
  - ➢ Assist the driver
  - ➢ Help in regulating the traffic
  - ➢ Enhance the comfort and safety of the driver
  - ➢ Assist visually impaired individual
  - ➢ Less research available for Canadian traffic signs
  - ➢ Achieve zero accident

C2TSR Lifecycle



University of Regina

1. Discovery

•Road accidents

•Human errors: detection and recognition errors

•Analysed dataset formulation (image dataset)

•Figured out –

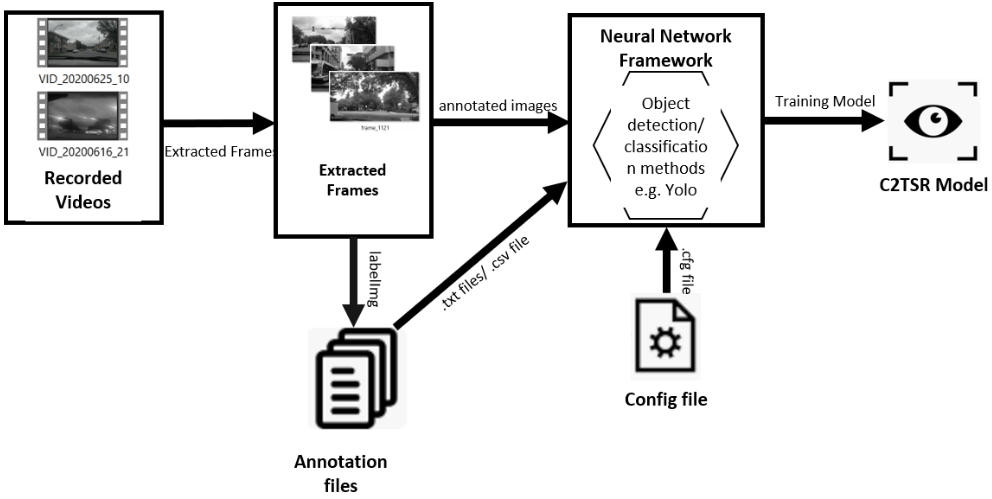
- a deep learning model/solution for this issue
- platform to train and test it : Google Colab
- installing python and dependent module on local machine

1. Discovery (Cont.)

- Solution: C2TSR Model
  - Deep learning method (YOLO - state-of-the-Art)
  - Dataset creation – frames from the recorded videos and annotation files
  - Around 57 different classes
  - Simulate this model in the real-time scenario

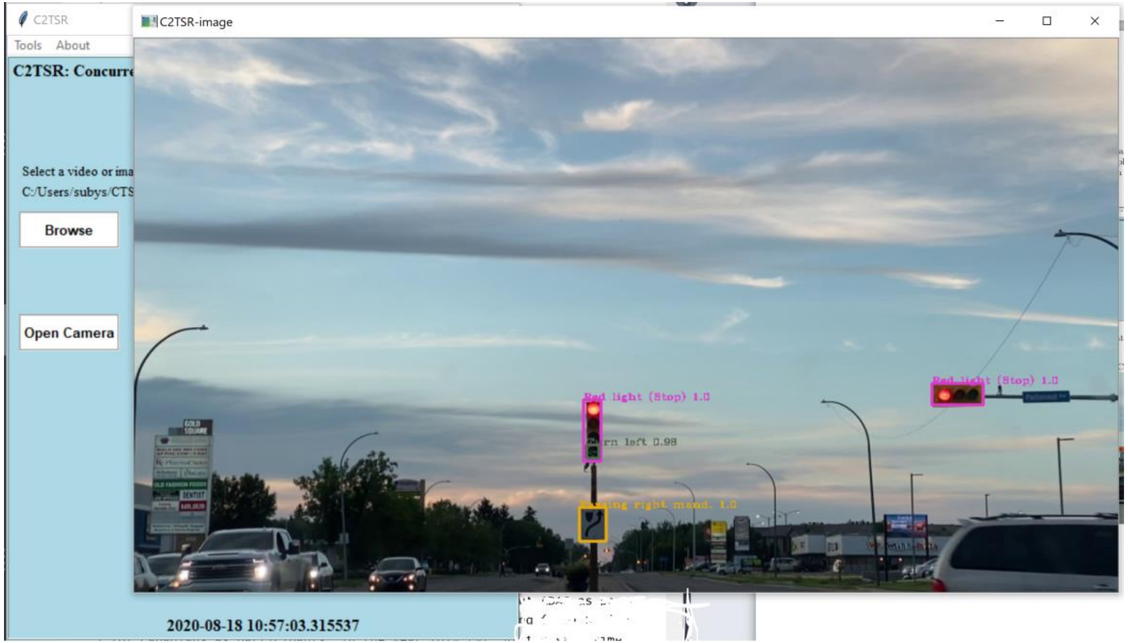
1. Discovery (Cont.)

C2TSR Architecture



1. Discovery (Cont.)

Expected Outcome



1. Discovery (Cont.)

Tools

- Annotation : LabelImg
- Framework : darknet
- Modules : OpenCV-python, Numpy, Panda, Matplotlib, Os, Re, Glob, lxml, Tkinter, datetime, filetype, mimetype, random, etc
- Platform : Jupyter lab, Google Colab and Overleaf
- Storage : Google Drive

2. Data Preparation

•Created new image dataset based on Canadian traffic signs

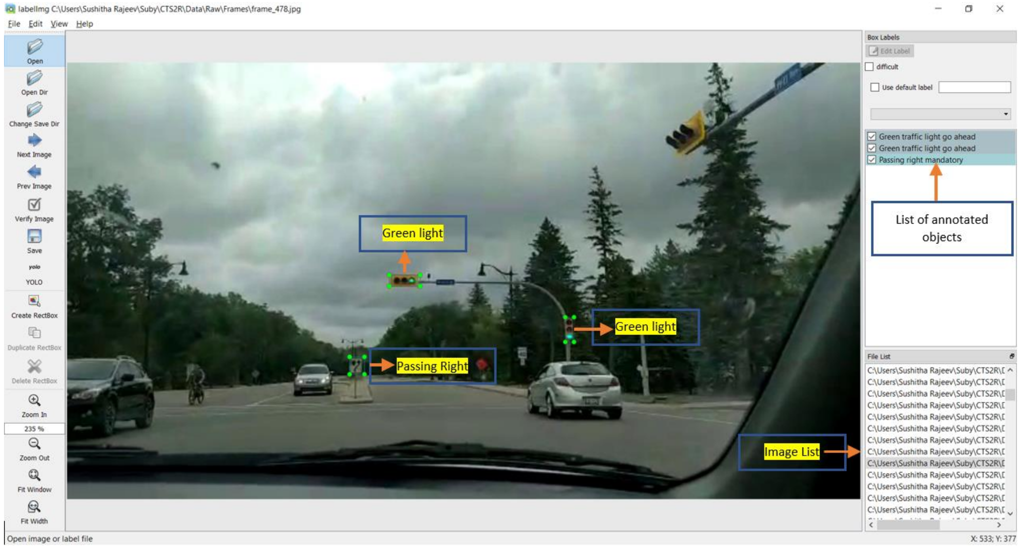
- Record Videos
- Extract frames (1 frame per second) and remove unwanted frames
- Create annotation files corresponding to each frame using labeling tool such as labelimg
- Data augmentation for image dataset such as dilation, erosion, blurring, increasing or decreasing contrast and brightness

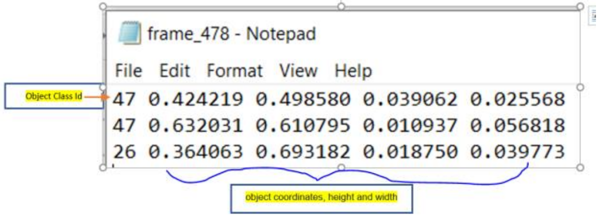
•Storage and access: Store dataset on Google Drive and access using Google Colab notebook

•Statistics: 3.5 GB (13,664 images + 13,664 annotation files)

2. Data Preparation (Cont.)

Command: Python lableimg.py ..\raw\frames ..\miscFiles\classes.txt

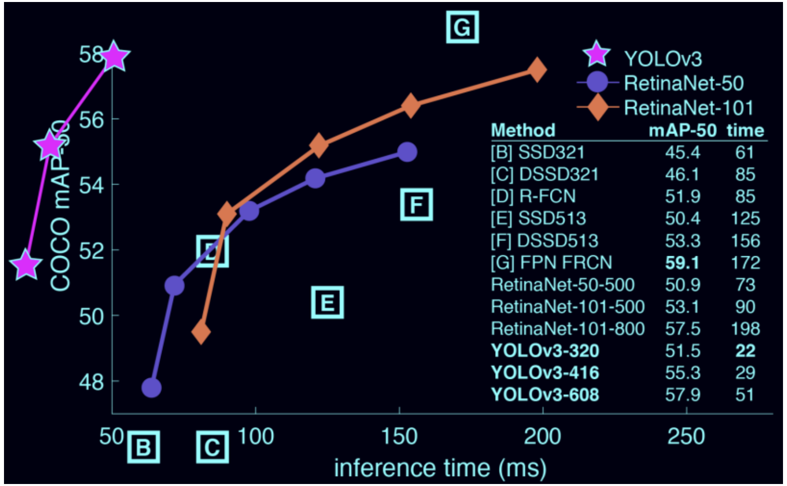




University of Regina

3. Model Planning

- R-CNN, SSD, Fast R-CNN
- Why YOLO algorithm?
- Source - https://pjreddie.com/darknet/yolo/



4. Model Building

- Darknet source: https://github.com/AlexeyAB/darknet

```
classes = 57
train = data/train.txt
valid = data/test.txt
names = data/obj.names
backup = /mydrive/trainC2TSR/latest/
```

Data file

Configuration file

```
[net]
# Testing
batch=64
subdivisions=16
# Training
# batch=64
# subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 114000
policy=steps
steps=91200,102000
scales=.1,.1
```

Command to train the model

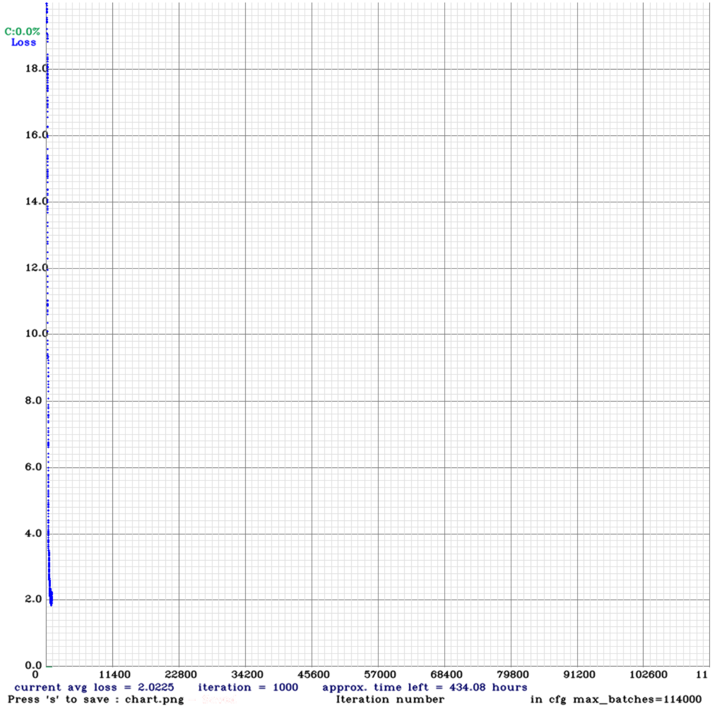
```
[ ] # train your custom detector
    !./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show
```

•Link to training script : https://drive.google.com/file/d/1Nf-_hcVh6aUj7d3KvAqpsNsrMtaF3CVi/view?usp=sharing

5. Communicate Results

At iteration 1000

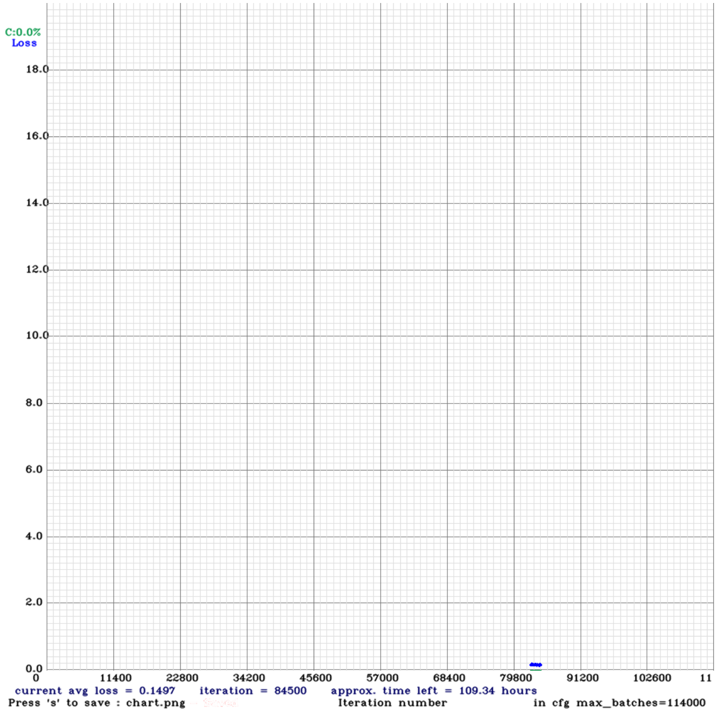
- Avg loss – 2.0225
- Time left for training ~430 hours



5. Communicate Results (cont.)

At iteration 80,000

- Avg loss – 0.14
- Time left for training ~109 hours

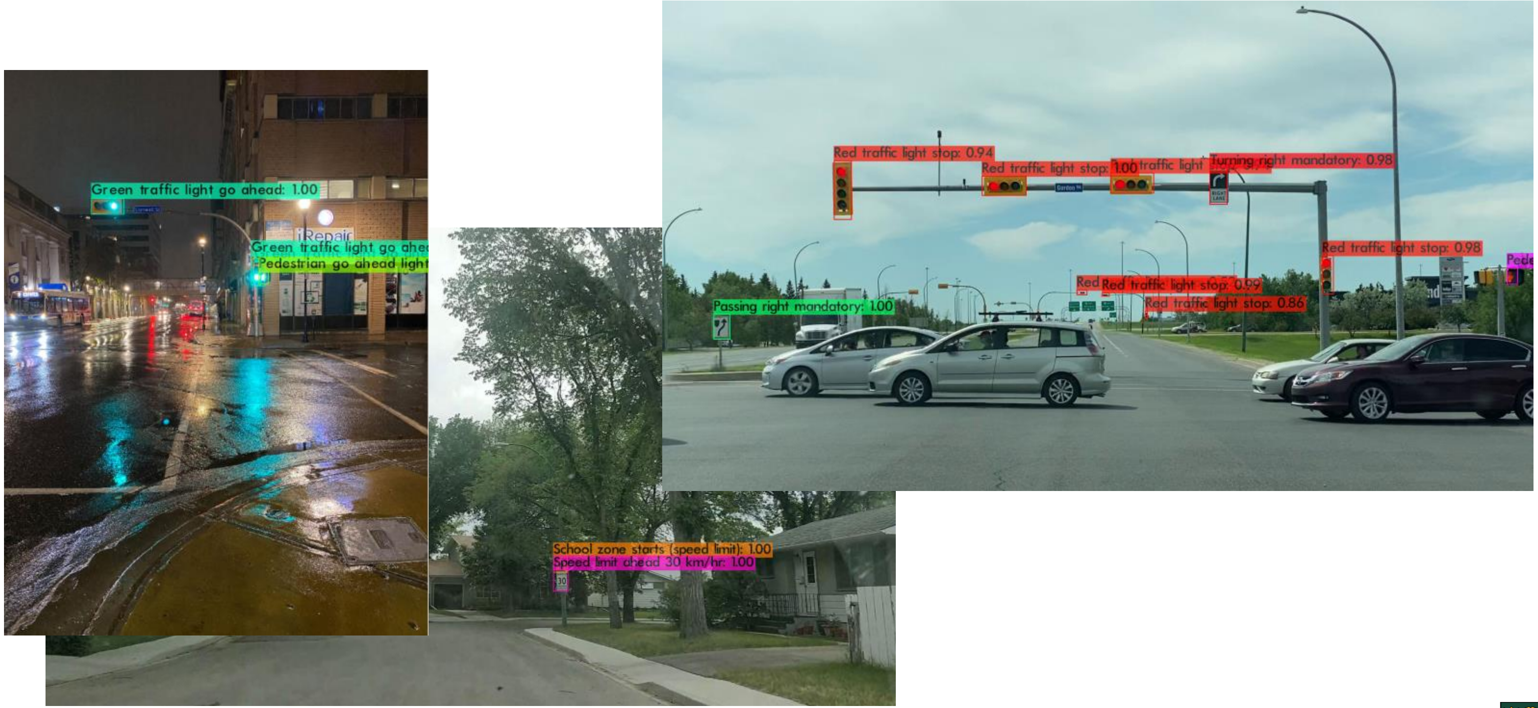


5. Communicate Results (cont.)

```python
#loading model and configuration files
net = cv2.dnn.readNet("../Model/yolov3_custom_last.weights", "../miscFiles/yolov3_custom.cfg")
#get layers
layer_names = net.getLayerNames()
#get output layer
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = numpy.random.uniform(0, 255, size=(len(classes), 3))
#input video
cap = cv2.VideoCapture(file_path)
#set font of the text
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
starting_time = time.time()
frame_id = 0
while True:
    #read frames
    _, frame = cap.read()
    frame = cv2.resize(frame, None, fx=0.4, fy=0.4)
    frame_id += 1
    height, width, channels = frame.shape
    # Detecting objects
    blob = cv2.dnn.blobFromImage(frame, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
    net.setInput(blob)
```

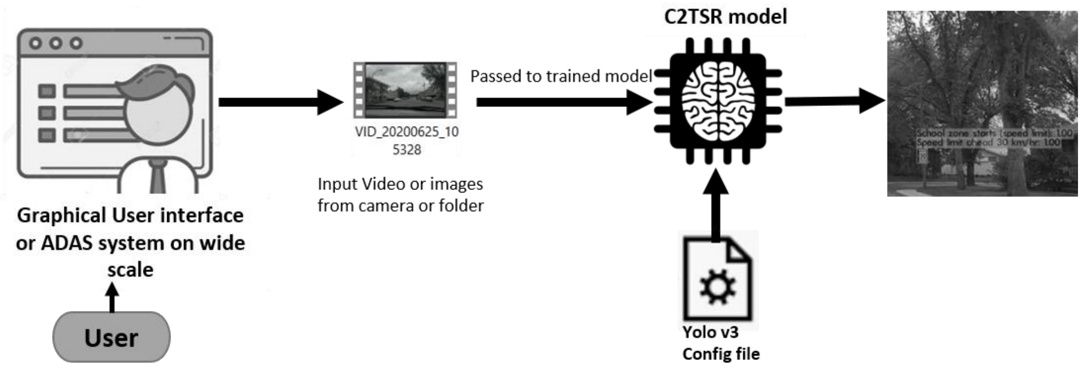
https://www.youtube.com/watch?v=I7xEClga5yI&feature=youtu.be -Output link

Actual results



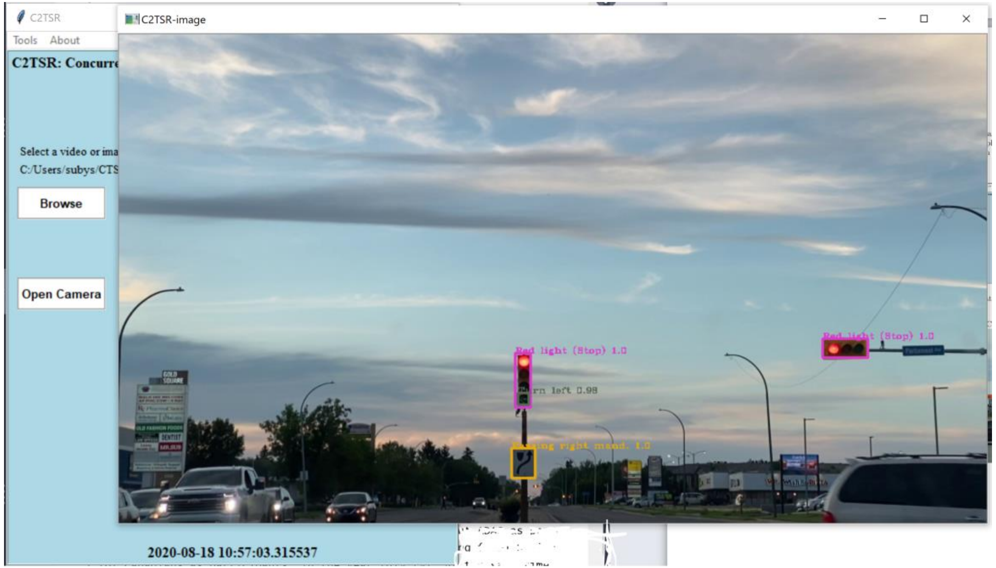
6. Operationalize

- ADAS system
- Warning system for drivers
- Smart Eye Glasses which have camera lenses



6. Operationalize

https://youtu.be/jlay6j7ZRp4 - Deployment video



Potential Application of the Model:

- ADAS technology
- advanced smart eyeglasses having camera lenses

Business value of the Model:

- market value for Autonomous vehicles is expected to reach to $77Billion by 2035
- growth in the numbers of workers i.e. nearabout 248,000 workers in 2021

Thank you

and if you have any questions, please drop an email to ssz389@uregina.ca.