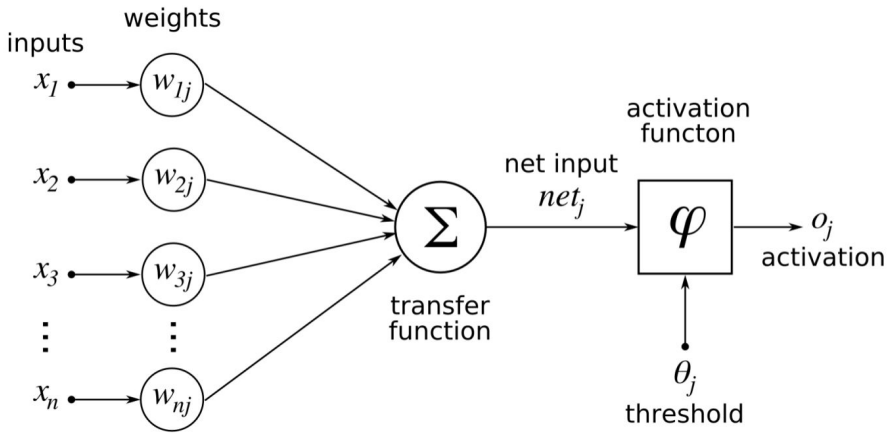
Real-Time Vision-Aided Tracking & Classification

Yue Zhou

Problems Overview

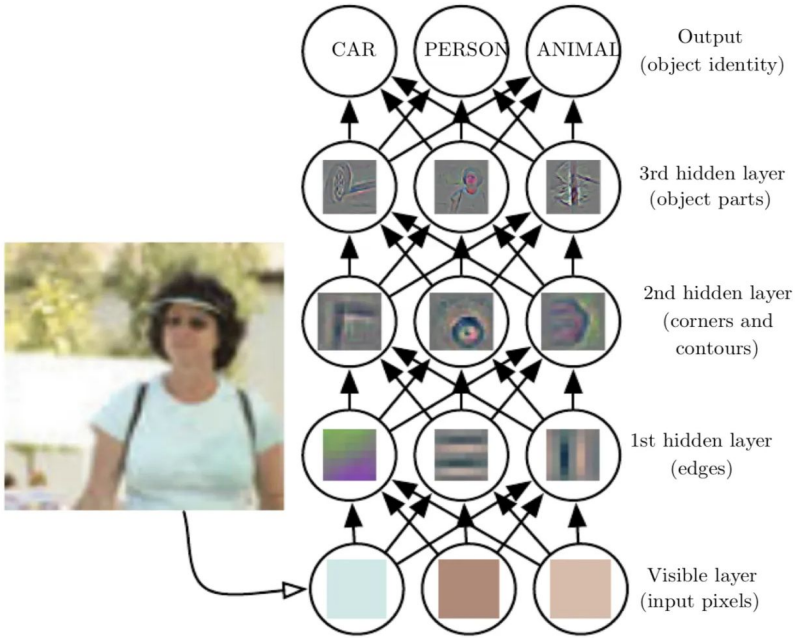
- SAPP system detects drones by using passive RF sensor technology.
- We want to use AI technology to detect, classify, and locate the drone from the real-time image from the PTZ camera, and use the drone information to track the drone and always keep it in the center of the camera.

Background--Convolutional neural network



- CNN: composed of multiple layers of artificial neurons
- Neuron: mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value.
- Convolution: the operation of multiplying inputs by weights and summing them

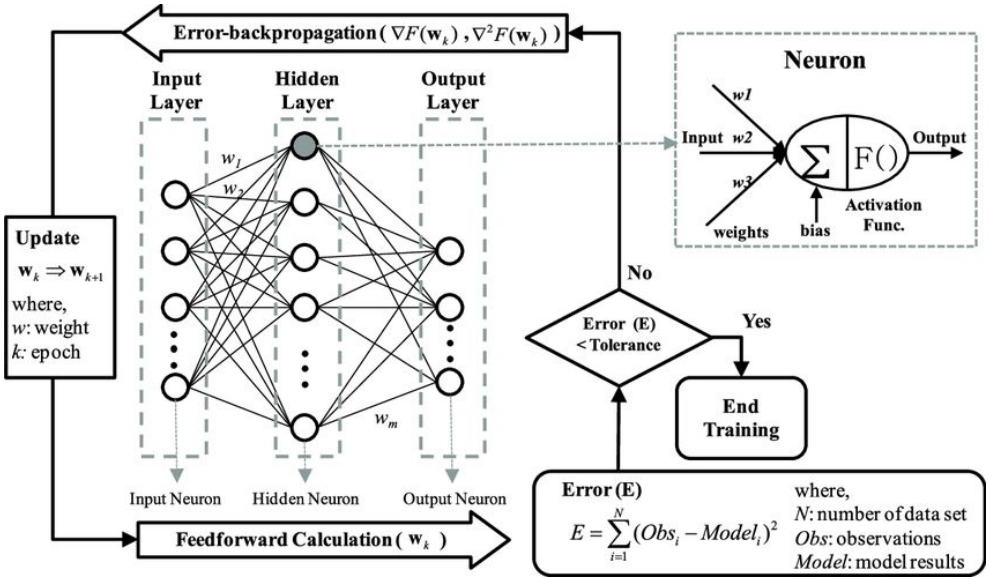
Background--Convolutional neural network



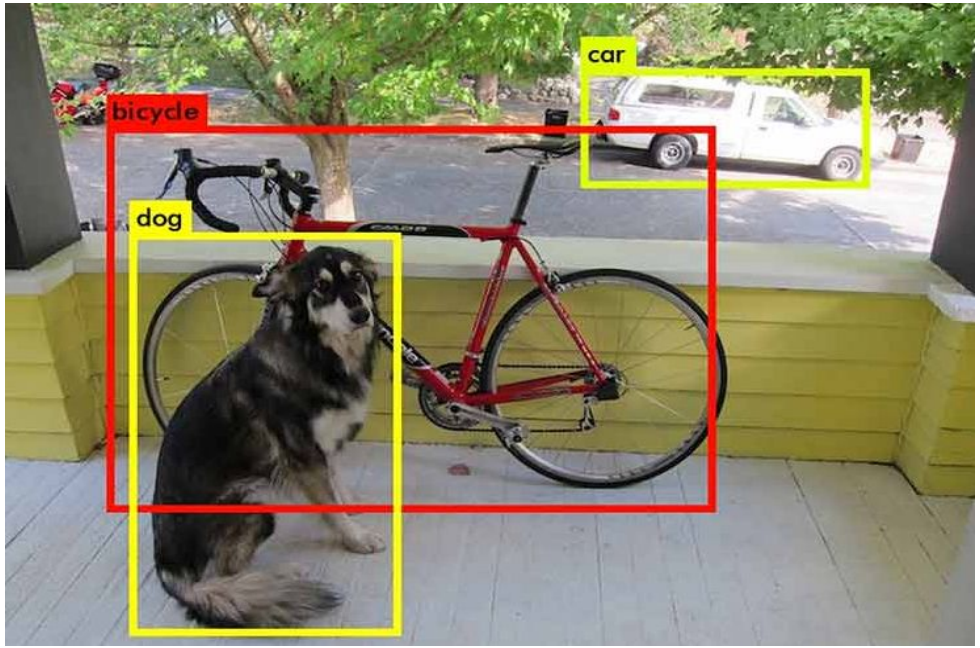
- The first layer: basic features such as horizontal, vertical, and diagonal edges.
- The next layer: extracts more complex features, such as corners and combinations of edges.
- The later layers: higher-level features such as objects, faces, and more.
- The classification layer: outputs a set of confidence scores (values between 0 and 1)

Background--Convolutional neural network

How to adjust the weights of the individual neurons to extract the right features from images? -- Training!



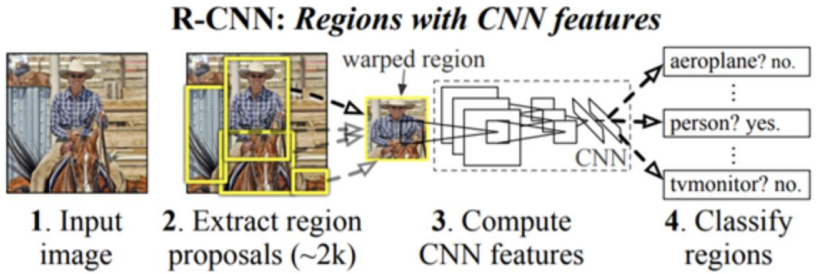
Background--Object Detection

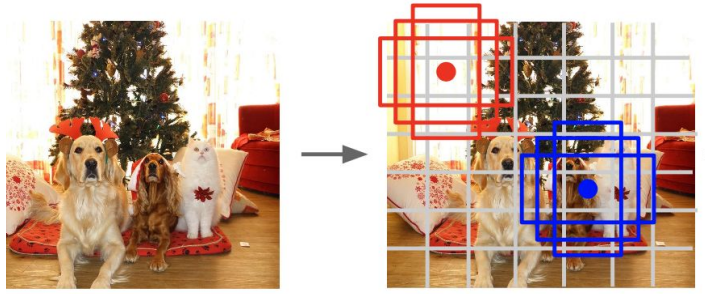


- **Input**:
  Image
- **Output**:
  One or more bounding boxes and a class label for each bounding box

Background--Models

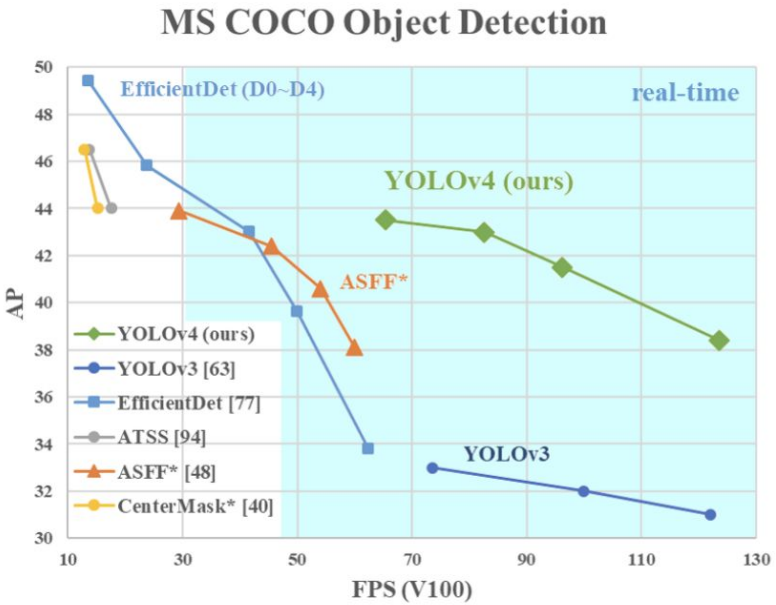
- **R-CNN family**: two-stage, slow, complex pipeline
- **SSD (Single Shot Detector)**: one-stage, combines regional proposals and feature extraction in a single network
- **YOLO (You Only Look Once):** a single neural network that predicts bounding boxes and class probabilities directly from entire images in one evaluation





Background--YOLOv4

- **YOLOv1:** fast but struggles with small objects
- **YOLOv2, YOLOv3:** improve speed and accuracy
- **YOLOv4**: 2020 April, state-of-art for object detection, extremely fast on real-time detection applications

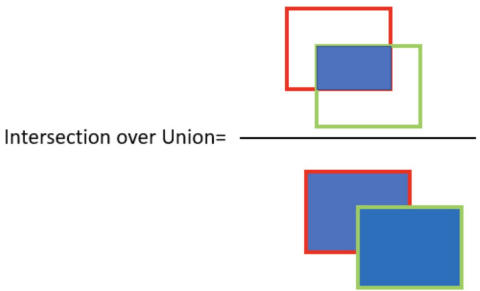


Background--metric

Speed: frames per second (FPS)

Accuracy: mean average precision (mAP)

Background--mAP



$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

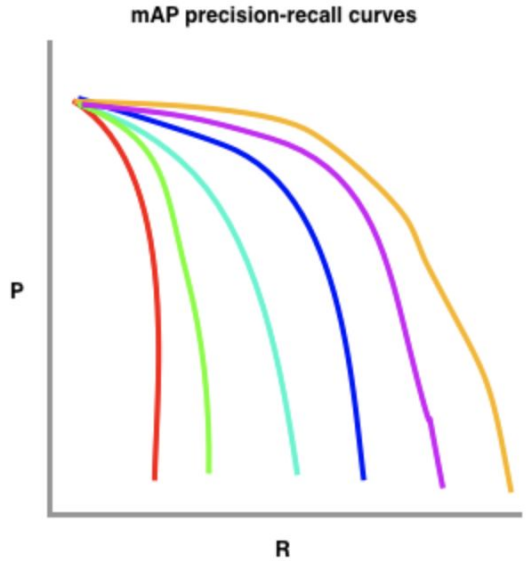
$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

Confidence score: the probability that a box contains an object
IoU: intersection over union

True Positive(TP):
1. Confidence score > threshold 1
2. The predicted class matches the class of a ground truth
3. The predicted bounding box has an IoU greater than a threshold 2 (e.g., 0.5) with the ground-truth.

False Positive(FP): 1 ✅ 2❌ or 3 ❌
False negative (FN): 1 ❌ 2 ✅ 3 ✅
True negative (TN): 1 ❌ 2 ❌ 3 ❌

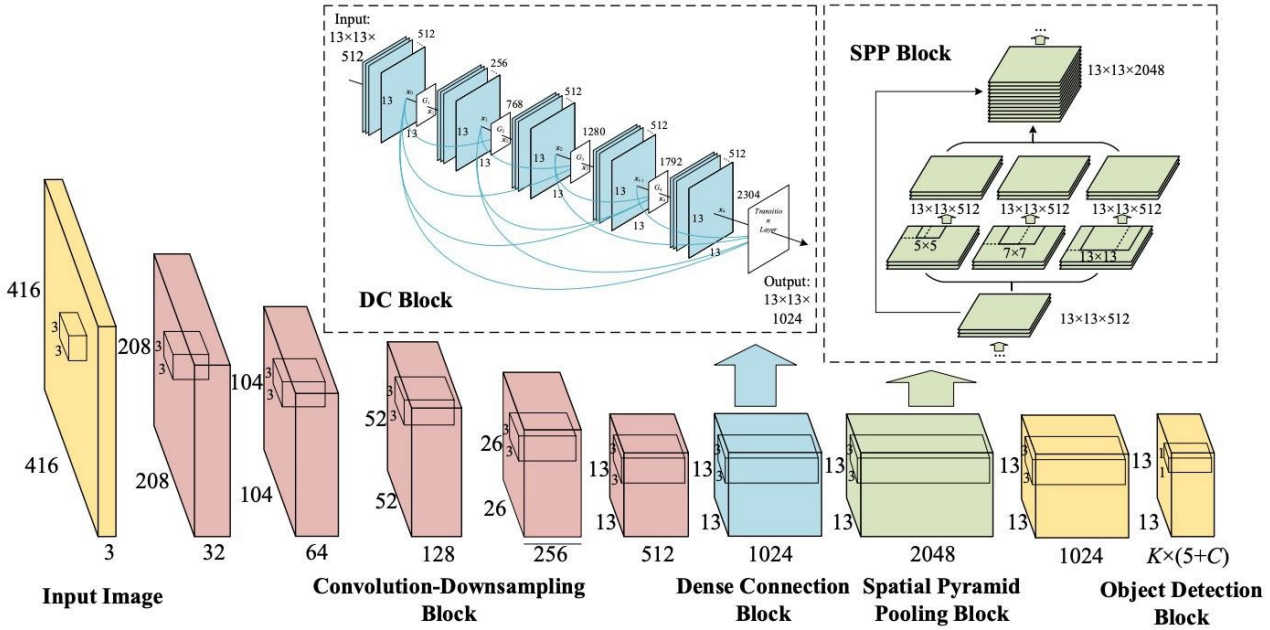
Background--mAP



Average precision (AP):
the precision averaged across all unique recall levels.

Mean Average Precision (mAP):
the mean of AP across all K classes

Background--YOLOv4



Methods

1. Data collection
2. Data preprocessing
3. Model development
4. Deployment, integration with the camera

Data collection--process

1. Fly the drone while control the PTZ camera, keep drone always in the camera manually
2. Record the videos
3. Download the videos from the camera server
4. Cut each video to frames using some online converter (5 frames per second)
5. Select the useful frames, delete blurring images or images without drone
6. Send to taobao

Data collection--strategy

Drone type	Weather condition	Time	Environment	Size	Condition
Mavic	sunny	daytime	Only pure sky	Small (more)	Having payload
Phantom	cloudy	night	With some trees and buildings	Middle	No payload
	overcast		With some people or birds	Large	
	rainy				
	snowing				

Each combination (2*5*2*3*3*2=360) should have at least 500-1000 images.

Data collection--strategy

Drone type	Weather condition	Time	Environment	Size	Condition
Mavic	sunny	daytime	Only pure sky	Small (more)	Having payload ❌
Phantom	cloudy	night ❌	With some trees and buildings	Middle	No payload
	overcast		With some people or birds ❌	Large	
	rainy ❌				
	snowing ⭕				

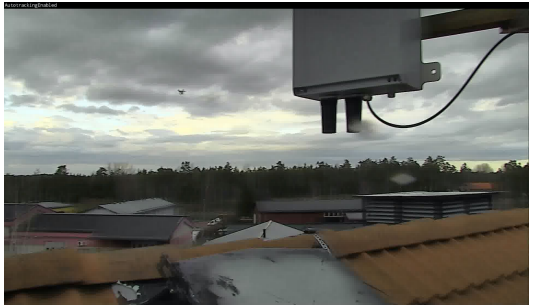
Data distribution now:

- 2500 mavic images, 3900 phantom images.
- Most of the images have small objects (<32*32)

Data collection -- example images

Phantom, small,, pure sky, sunny

Phantom, middle, with house and trees, sunny



Phantom, small, with house and trees, cloudy

Mavic, small, pure sky, overcast

Data preprocessing--Annotation


```
<annotation>
    <folder />
    <filename>p1-frame-001.jpg</filename>
    <path>/p1-frame-001.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>1920</width>
        <height>1080</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>phantom</name>
        <pose>Unspecified</pose>
        <truncated>Unspecified</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>1290</xmin>
            <ymin>407</ymin>
            <xmax>1313</xmax>
            <ymax>427</ymax>
        </bndbox>
    </object>
</annotation>
```

Model Development

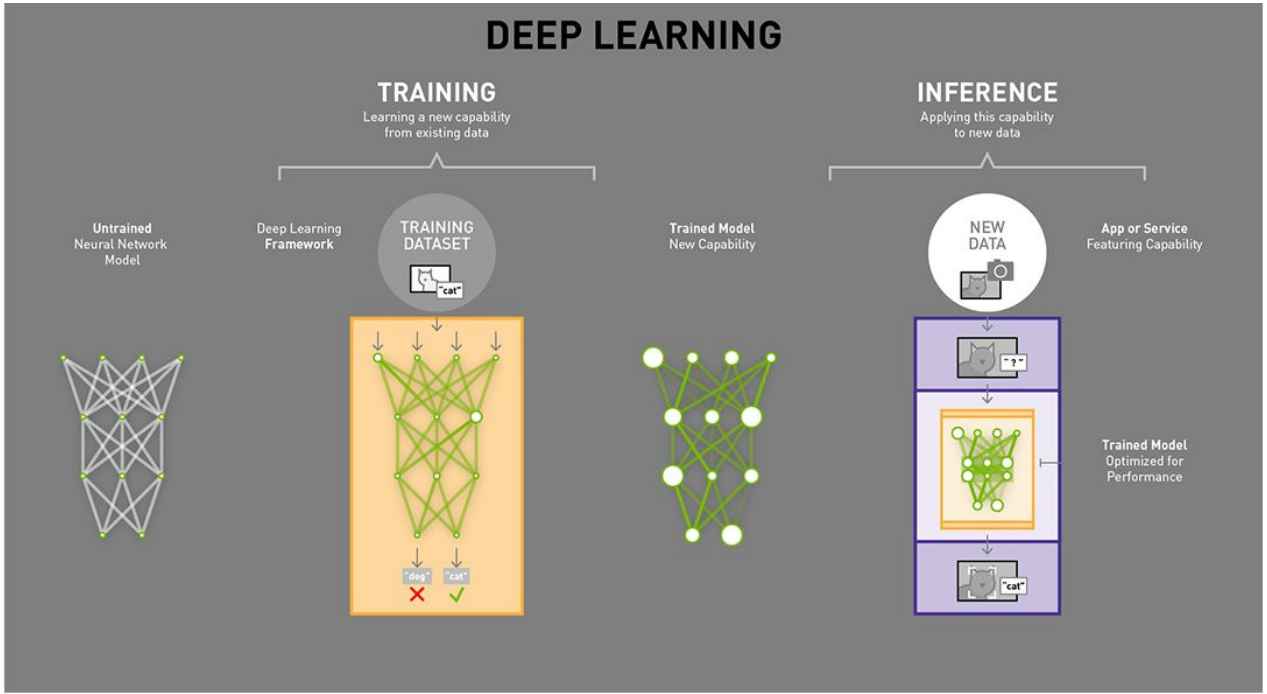
TensorRT, TensorFlow	OpenCV
CUDA driver	
Nvidia Jetson Xavier	

Nvidia Jetson Xavier

Model Development--workflow

1. Configure the environment on **Nvidia Jetson Xavier.**
2. Download the pre-trained YOLOv4, YOLOv3, YOLOv4 Tiny model, use our first 1000 images to train the models.
3. Convert the tensorflow model to tensorRT model.
4. Compare the performance of different models.
5. Choose a suitable model regarding speed and accuracy, we choose YOLOv4.
6. Collect more images to increase the accuracy.

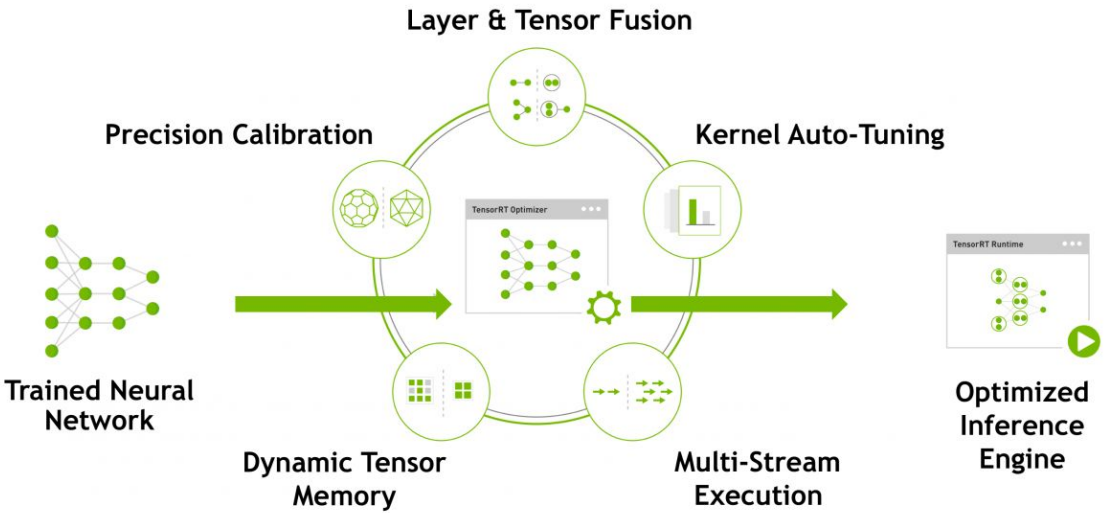
Model development--tensorRT



Deep learning:

Training & Inference

Model Development

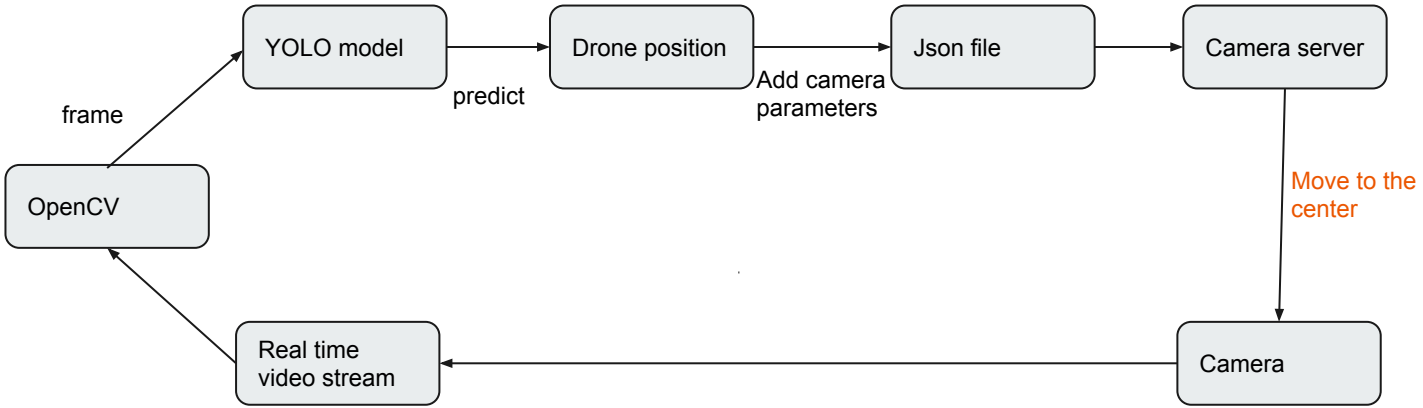


The reason to convert to TensorRT:

can optimize the workflow and increase the speed, which is more suitable for inference.

Speed of prediction	YOLOv4	YOLOv4 Tiny
Tensorflow	3 FPS	12 FPS
TensorRT	12 FPS	40 FPS

Deployment and integration



Future work

- **Finish the camera server part**

  Move the camera so that the drone can always be in the center of the images.

  -- by analyzing the position and camera parameters (zoom in/out, direction, tilt)

- **Train the model with more data according to the scenarios**
- **Add object tracking algorithms**

  Object detection: processes each frame independently

  Object tracking: track a particular object across the entire series of frames (video)

  Deep SORT (Simple Real-time Tracker): Failed because the drone doesn't have constant velocity in the video (the camera moves)