TRAFFIC SIGN DETECTION & RECOGNITION USING FEW-SHOT LEARNING.

Name: Do Dang Gia Vuong
Nguyen Chinh Dat
Do Xuan Bach
Supervisor: Bui Van Hieu

CONTENTS

01 INTRODUCTION

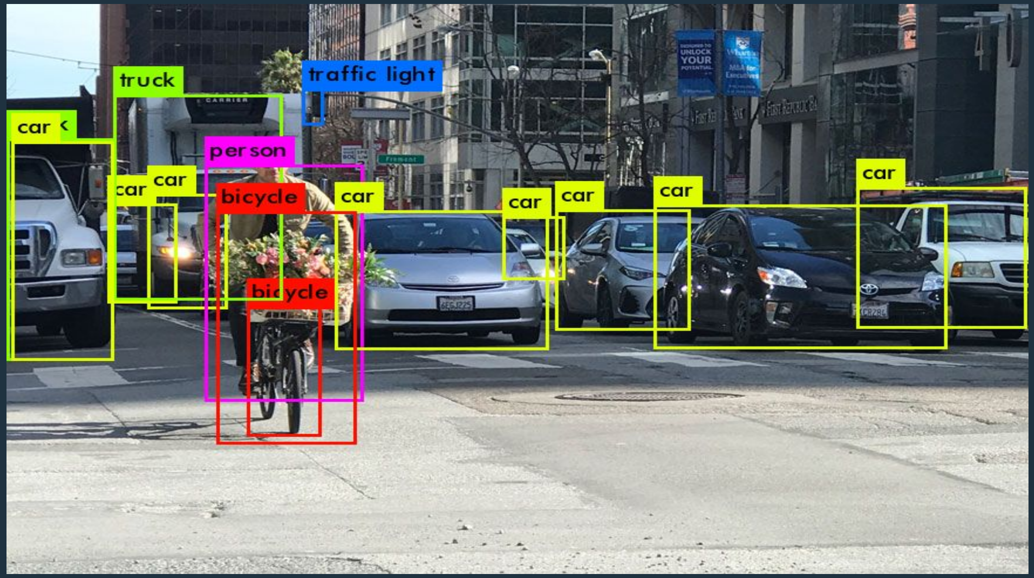
Intelligent traffic sign detection and recognition has become a basic function for intelligent network-linked automobile systems.

THE NEED OF TRAFFIC SIGN DETECTION

- Autonomous driving
- AI systems for vehicles
- Analyze the driving environment
- Make a decision to a situation
- Aware of traffic dangers, restrictions and regulations that a vehicle must follow

SIGNIFICANT OBJECT DETECTION METHOD

- Region-based Convolutional Neural Networks (R-CNN)
- Fast R-CNN
- Faster R-CNN
- Single Shot Detector (SSD)
- Spatial Pyramid Pooling (SPP-net)
- YOLO (You Only Look Once)



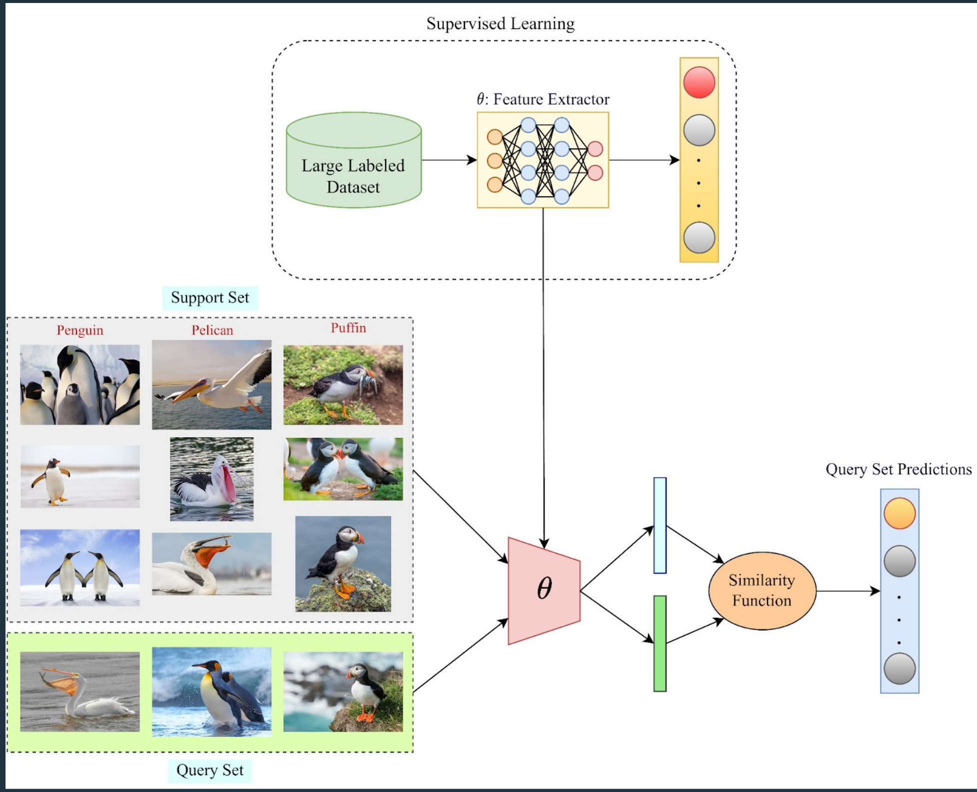
MAIN PROBLEMS

- The complexities of the environment which is affected by buildings, roads, vehicles and plants
- The accuration of understanding the meanings of traffic-signs due to the massive amount of traffic signs
- The processing of applications in real-time
- Required large amount of training data and computational power

One of the drawbacks is there are numerous traffic-signs with different meanings

SOLUTION FOR THE PROBLEMS

- Data augmentation and collecting more data are common solutions for the problem
- Few-shot learning (FSL) is a good model for the tasks, which is the learning method when only a few samples of data are labeled



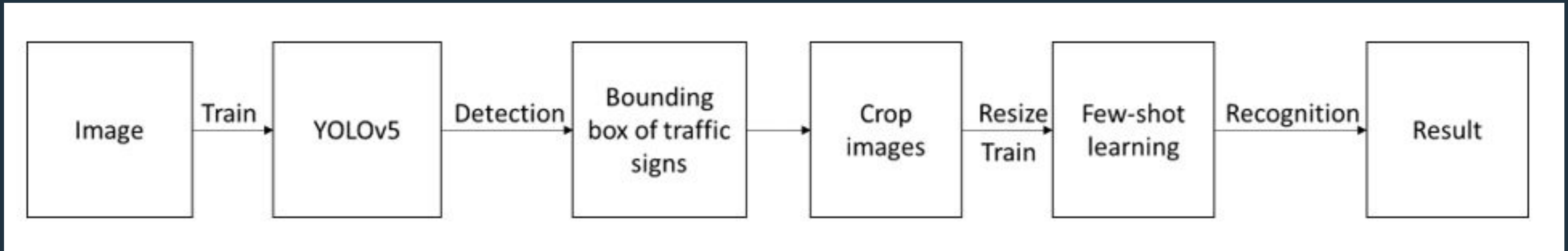
OVERVIEW OF THE METHOD

Traffic sign detection

YOLO v5 is used as the algorithm for indicate traffic-signs positions

Traffic sign recognition

The indicated traffic-signs are classified by few-shot learning



Example result after using few-shot learning classification method

02 RELATED WORKS

To accomplish the job of traffic signs identification, researchers have recently largely chosen a visual scheme based on deep CNNs (convolutional neural networks).

TRAFFIC SIGNS DETECTION

- Supplemented with more realistic and diverse training images generated
- Generative Adversarial Networks (GAN).

- YOLOv5
- Deep neural network to recognize and categorize things in a picture

TRAFFIC SIGNS DETECTION

In order to get the multiscale features in pyramids, the author suggests using a cascaded R-CNN

YOLOv5 method uses a multi-scale detection technique to further enhance traffic sign identification precision

2.2. FEW-SHOT OBJECT CLASSIFICATION

Few-shot learning is a technique that may be used when there is not enough data

Fine-tuning-based multi-scale few-shot detection model

2.2. FEW-SHOT OBJECT CLASSIFICATION

Metric Learning

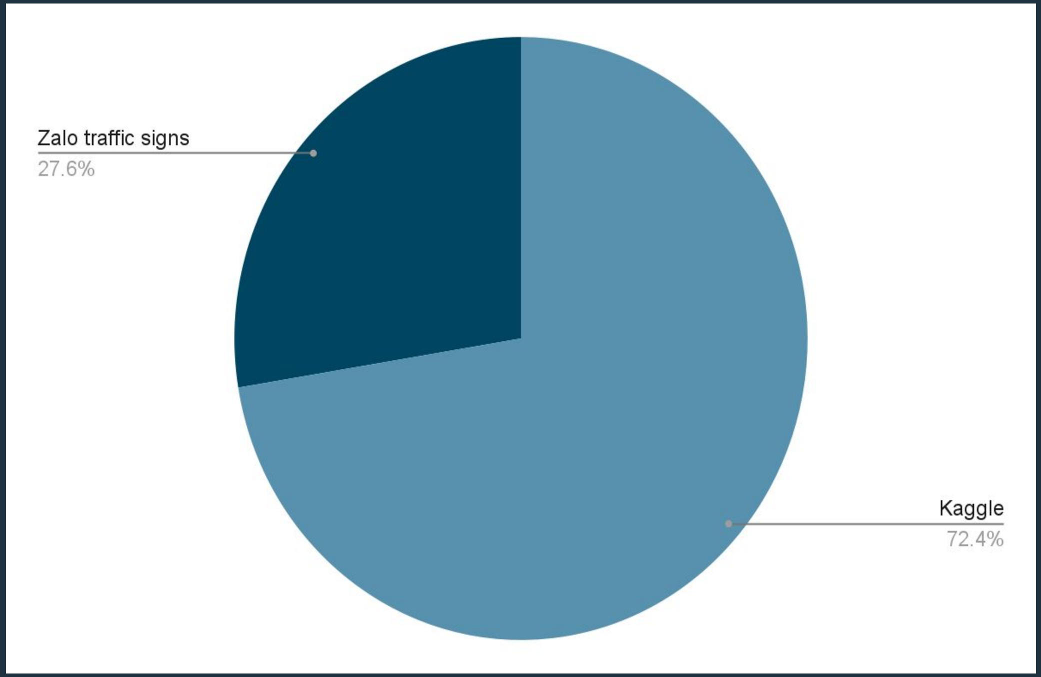
Prototype-based classification

03

DATA PREPARATION

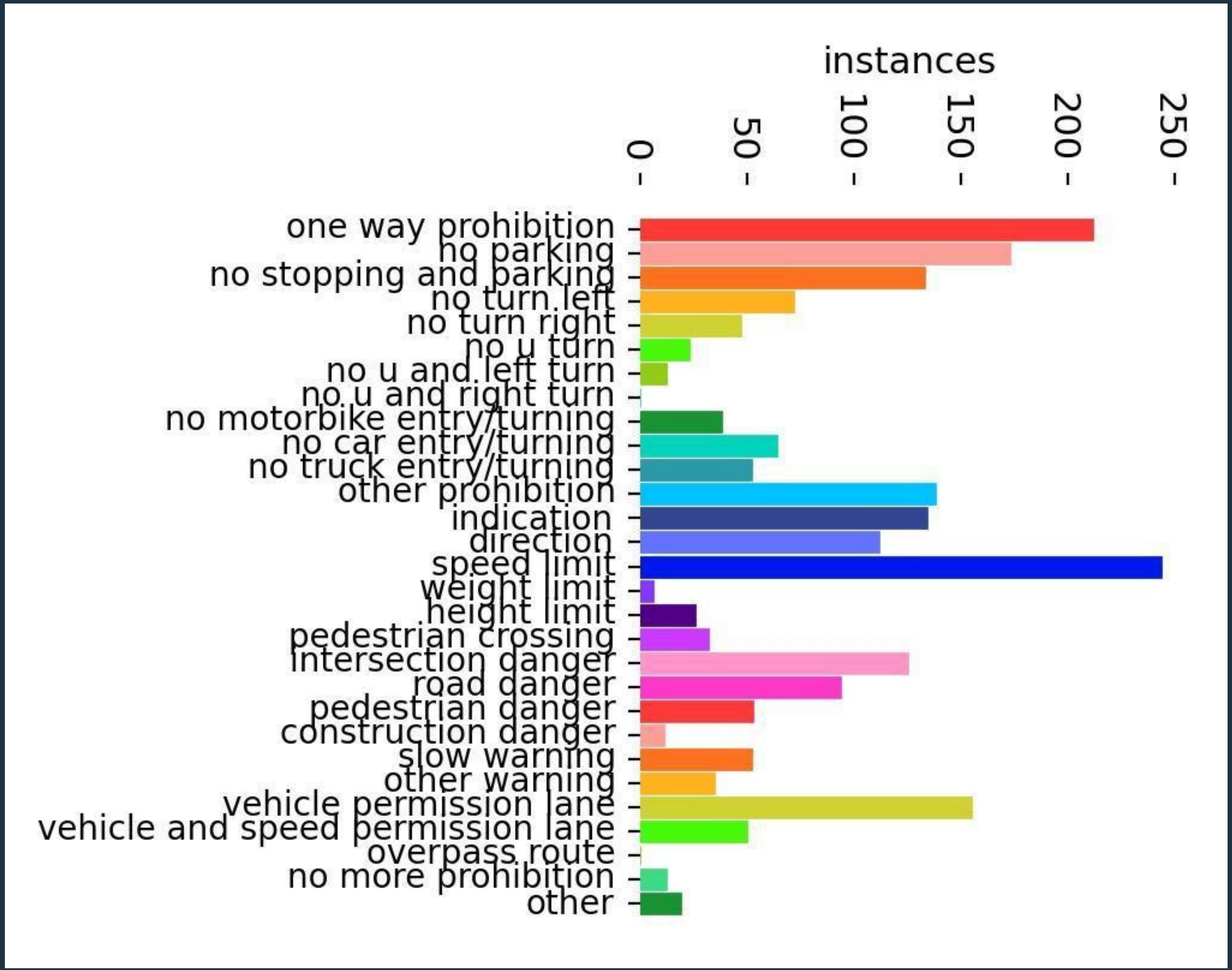
THE DATASET

- Data sources: 1170 Images with resolution of 1622x626 pixels from Kaggle and images from **zalo_traffic_sign**
  - 847 **Viet Nam traffic sign(YOLO format)** images from Kaggle
  - 323 images from **zalo_traffic_sign**



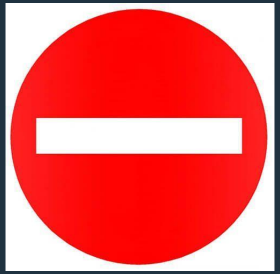

Vietnam Airlines
VN Travel
LOTTERIA
NÚT GIAO
INTERCHANGE
AN PHÚ
2 KM
50
40
NHÀ THUỐC
AnTâm
BIDV

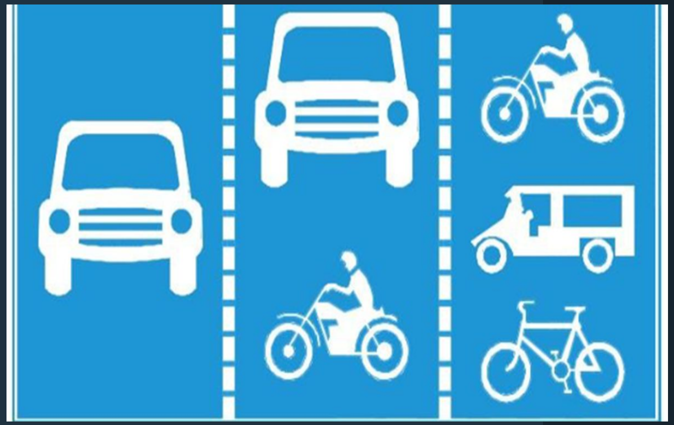
Number of images in every classes in dataset.



GROUPS OF TRAFFIC SIGNS

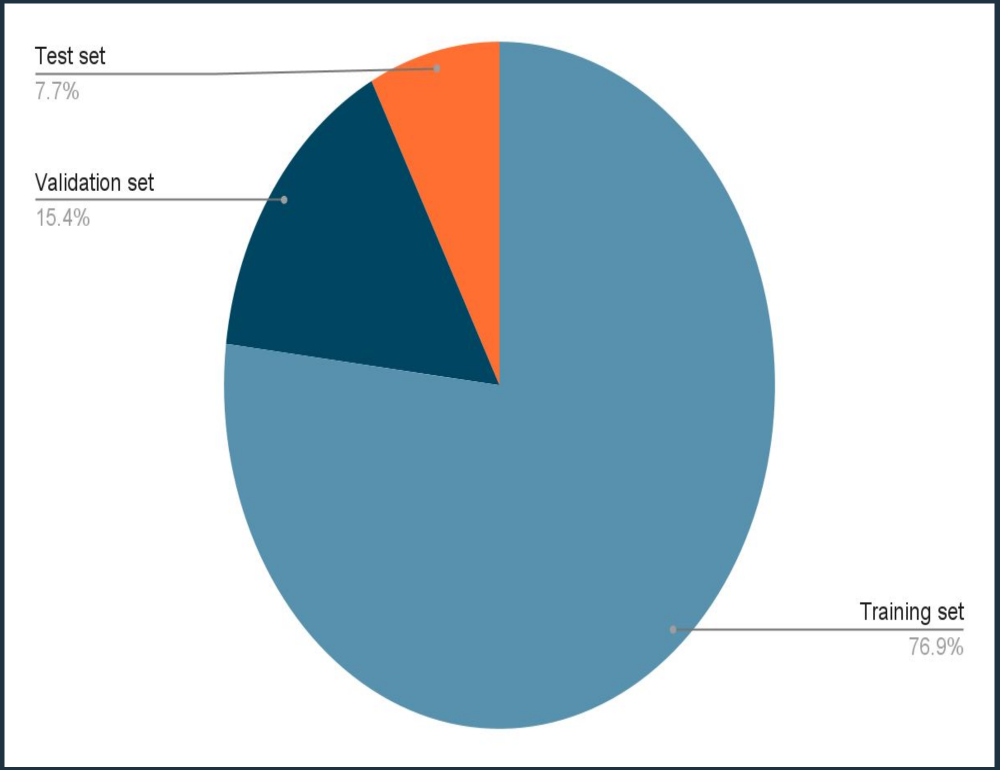
- The ***regulatory signs*** provide drivers with positive instructions, as well as negative instructions.
- The ***warning signs*** alert drivers of potential hazards ahead.
- The ***informative signs*** provide useful information to drivers.





DATA SPLIT

- Data is split into training, validation and test set
  - The training set has 900 images
  - The validation set has 180 images
  - The test set has 90 images



DATA FORMAT

- Images and corresponding text file:
  - Each traffic sign is indicated by a bounding box, consists of values bx, by ,bh, bw.
  - Indicated traffic signs are labeled based on the type of the sign.

04 METHODOLOGY

Object Detection

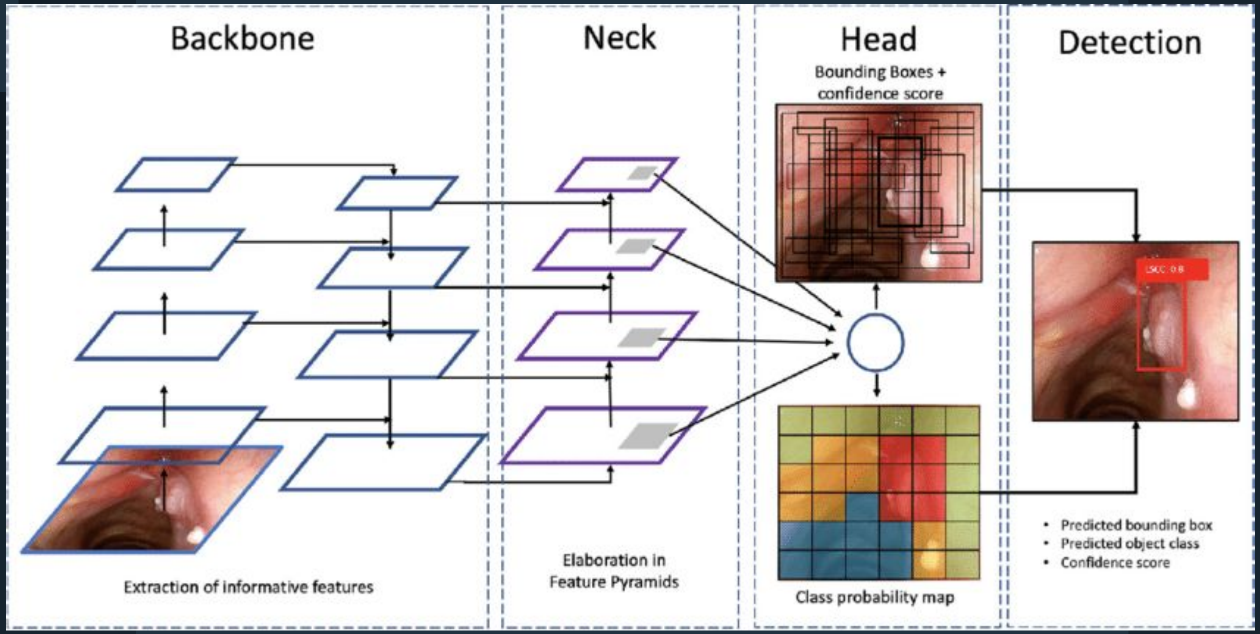
Object Recognition

4.1

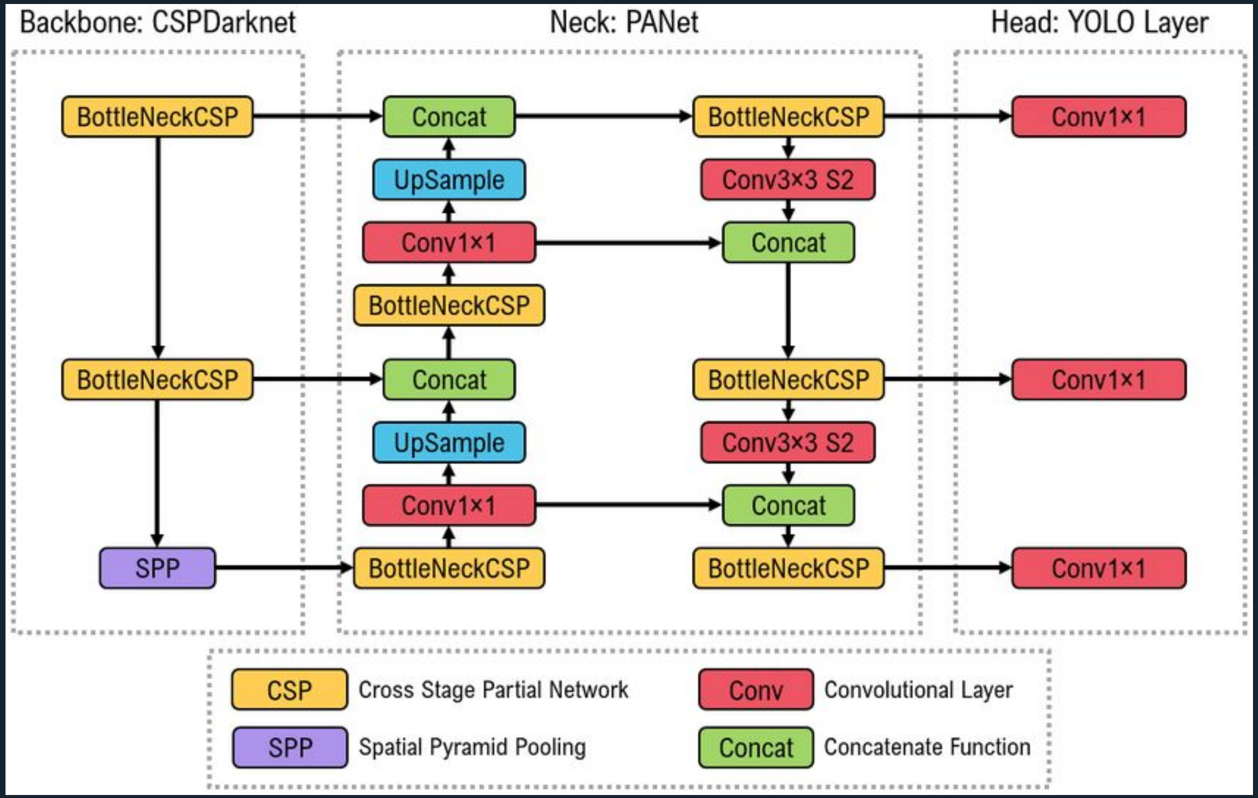
OBJECT DETECTION

“You Only Look Once”, or YOLO. This model can recognize items in each cell of the feature map.

4.1
OBJECT DETECTION



4.1 OBJECT DETECTION



4.1.1

BOUNDING BOX

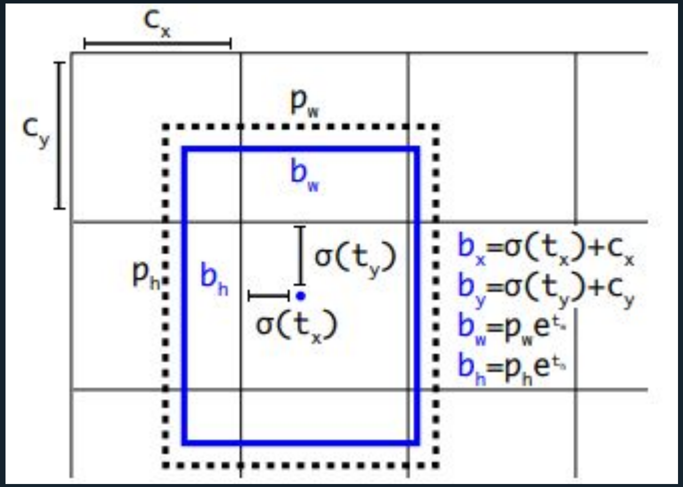
Our approach

$$b_x = \sigma(t_x) + c_x$$

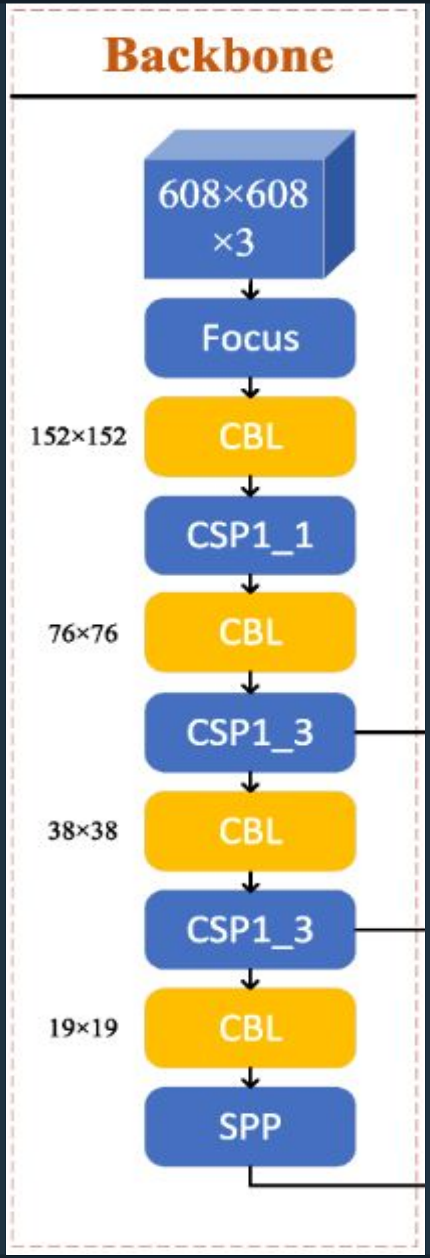
$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w e^{t_w}$$

$$b_h = p_h e^{t_h}$$

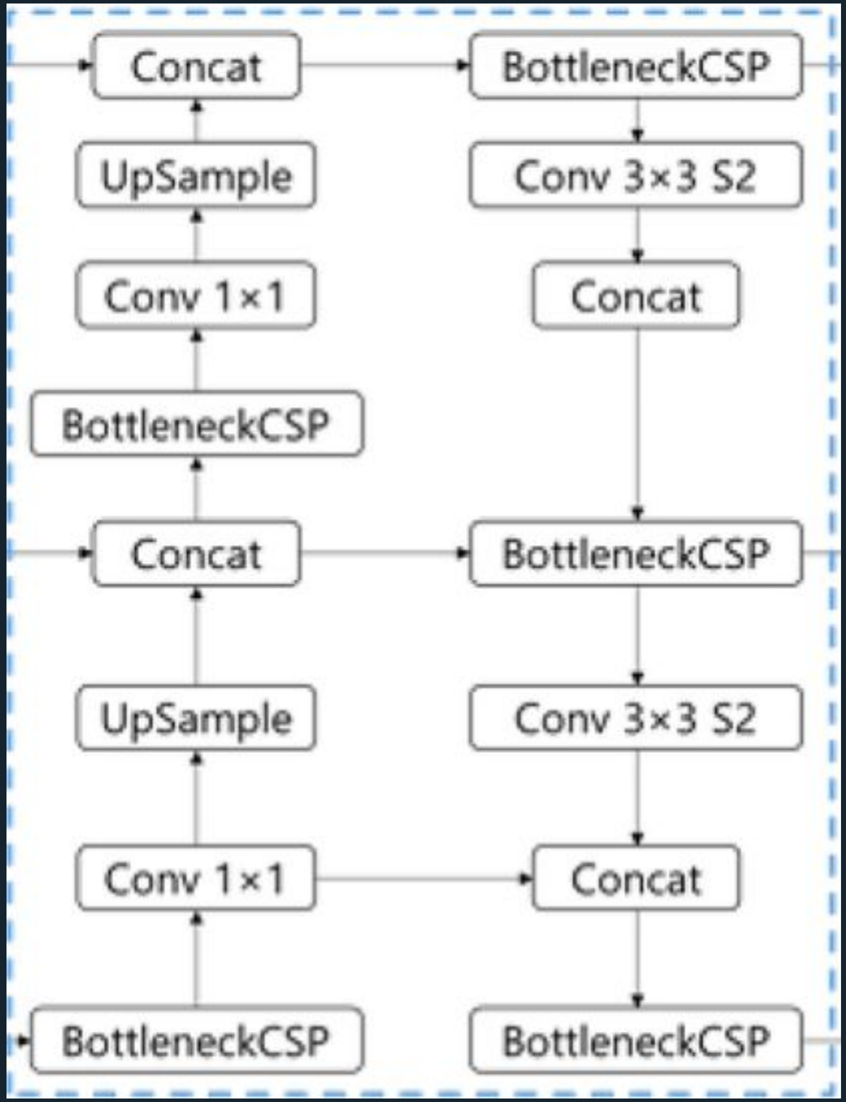


4.1.2 BACKBONENETWORK



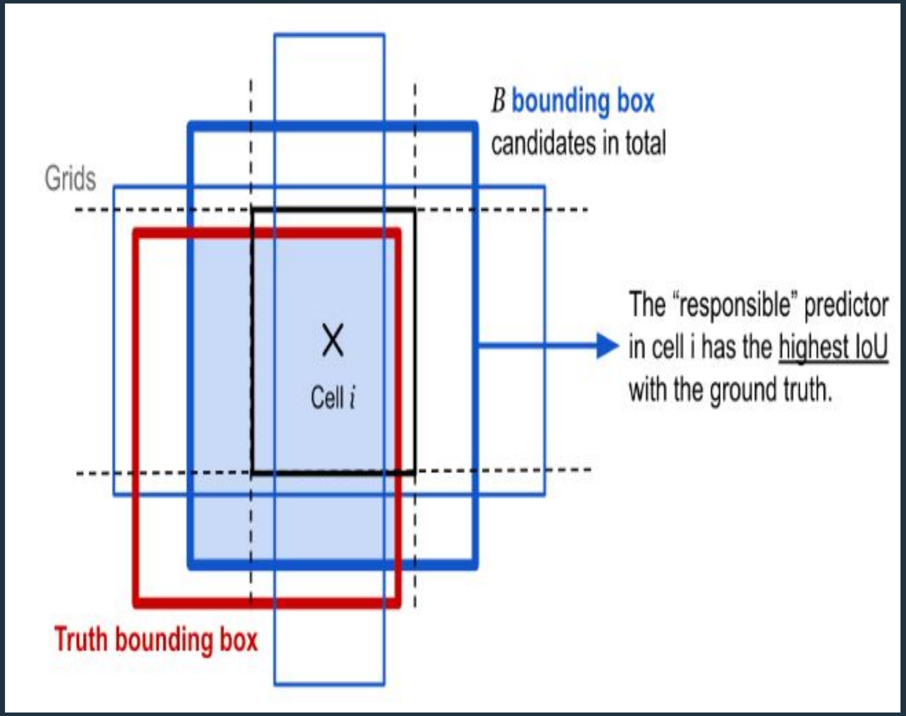
- The architecture of CSP
- Extracts characteristics at several scales
- Input picture is processed by a set of convolutional layers

4.1.3 NECK NETWORK



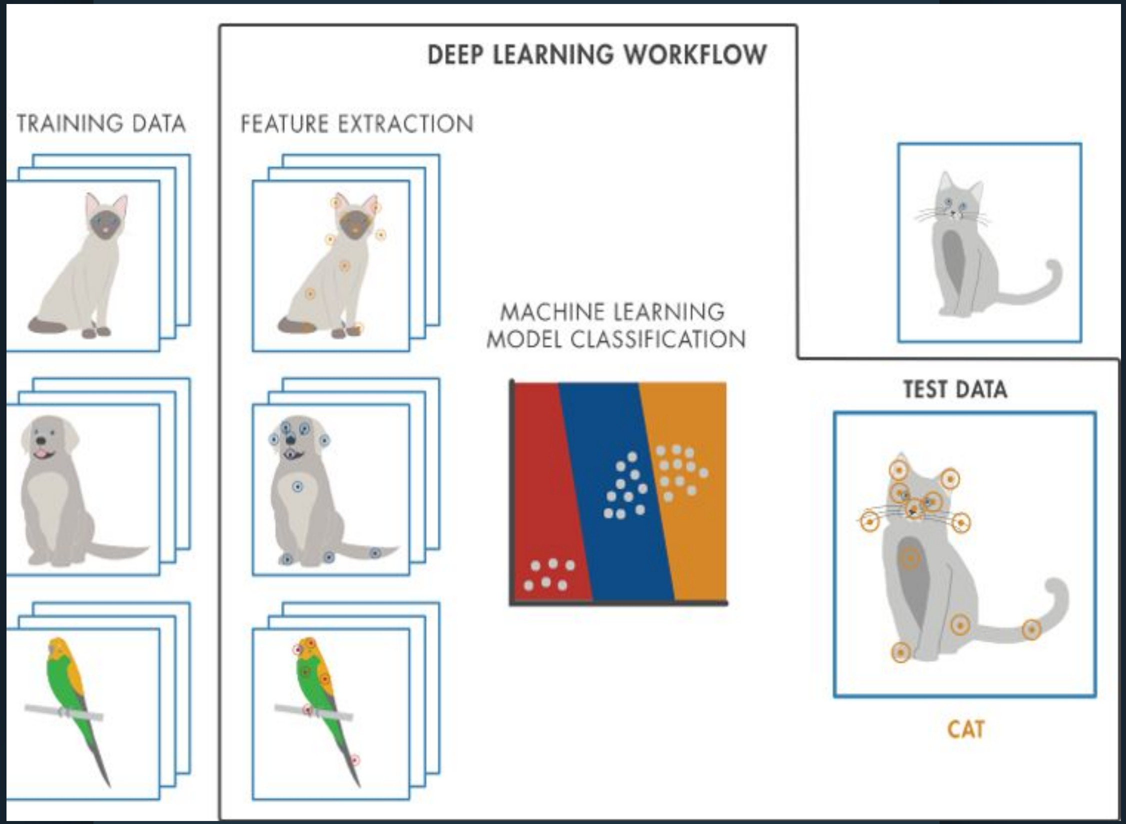
- Combines feature maps generated at various levels of the backbone network
- Convolutional layers, max-pooling layers, and SPP modules

4.1.4
HEAD NETWORK



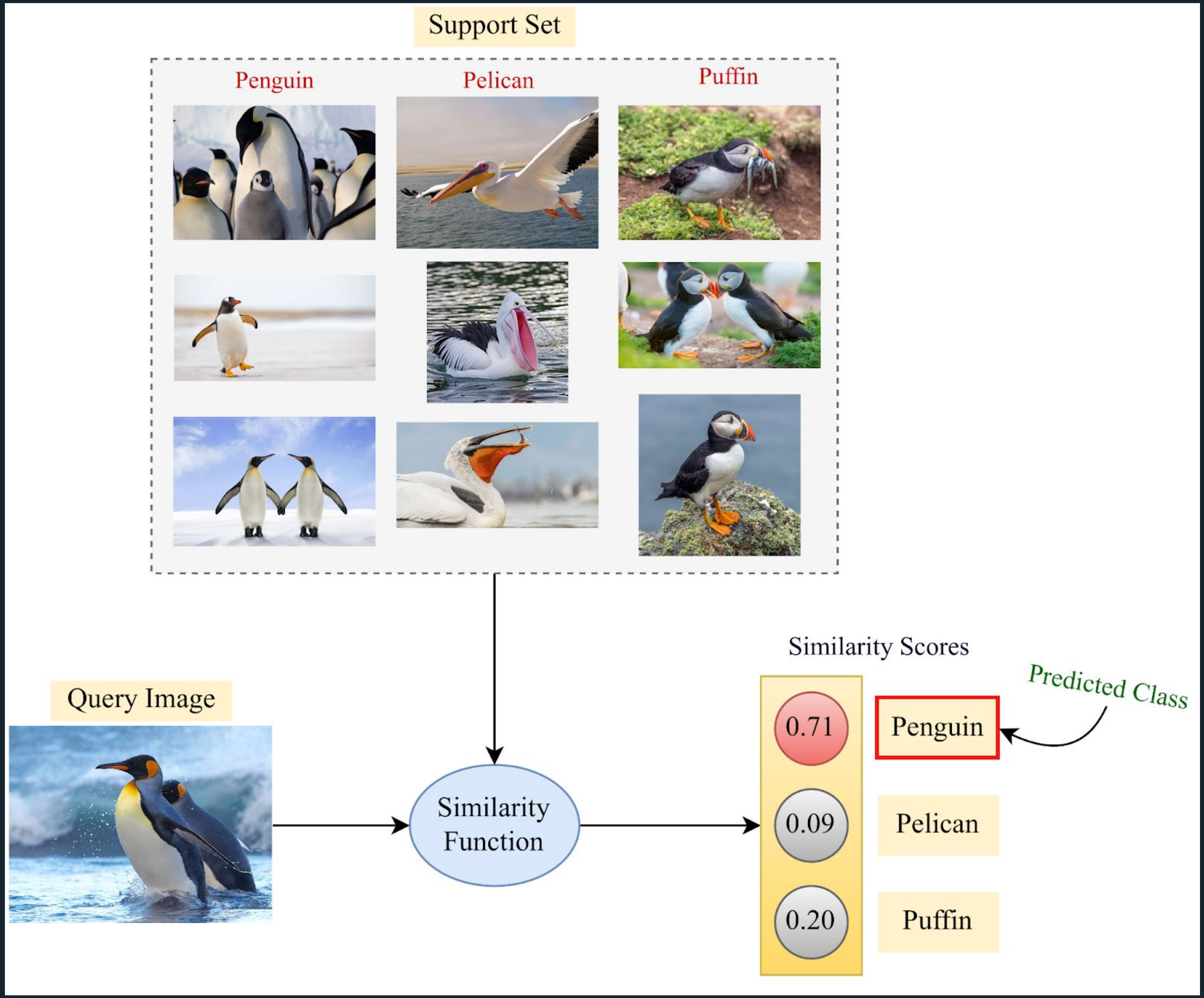
- Using feature maps generated by the backbone and neck networks
- Convolutional layers followed by anchor boxes
- Dynamic anchor assignment

4.3
OBJECT RECOGNITION



4.3.1

FEW-SHOT LEARNING



4.3.1 FEW-SHOT LEARNING

- Few-short learning (FSL) is a machine learning method where a pre-trained model can learn new data using only a few labeled samples per class.

- FSL can classify new objects given a small set of training data, which can save time, computational power and resources.

4.3.1 FEW-SHOT LEARNING

Researchers have classified FSL into four categories:

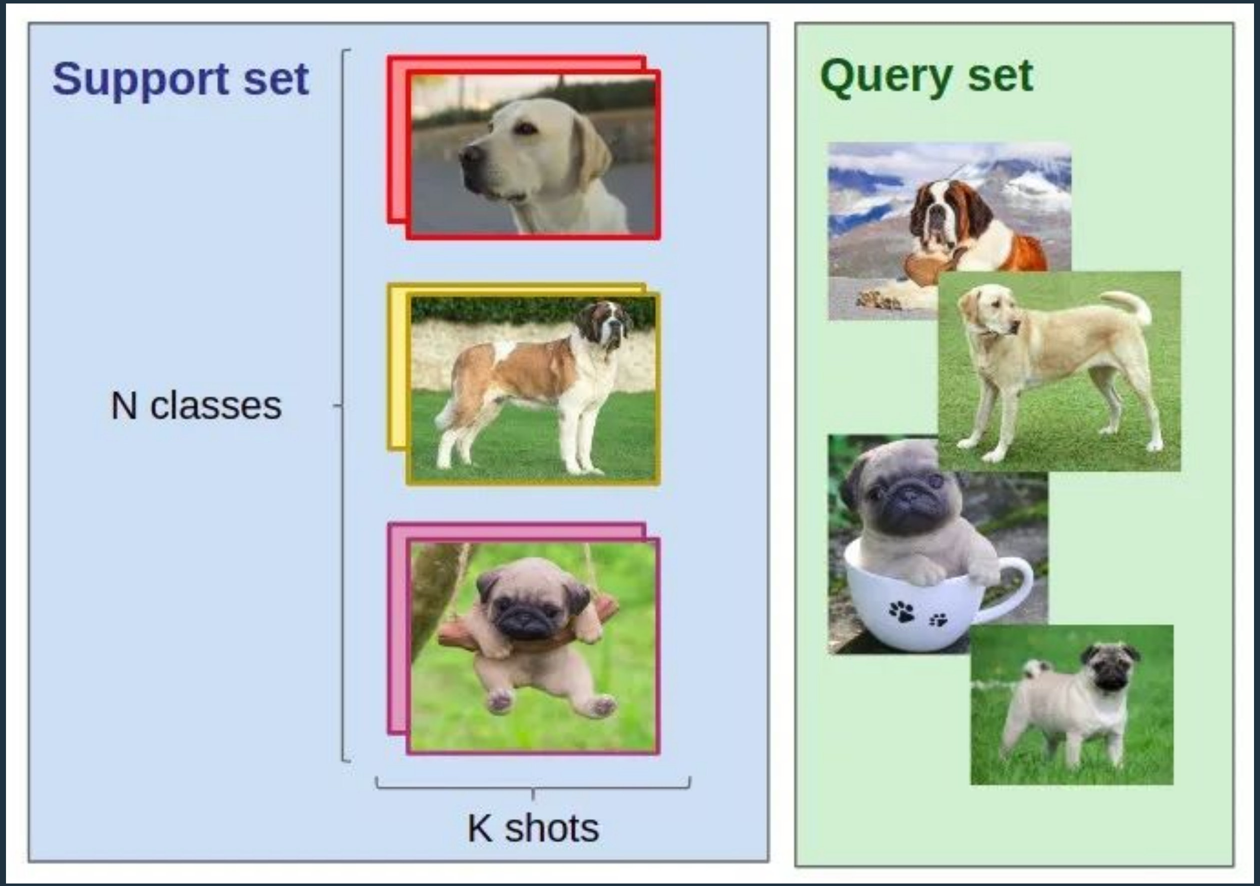
- Zero-Shot Learning (ZSL)
- One-Shot Learning (OSL)
- Few-Shot Learning (FSL)
- N-Shot Learning (NSL)

4.3.1 FEW-SHOT LEARNING

N-way K-Shot classification:

- Support set
  - N is the number of classes
  - K is the number of samples from each class used for training
- Query set
- The objective is to categorize Q images into N classes even with the limited amount of data (N*K) available for the training set

4.3.1 FEW-SHOT LEARNING



4.3.2 FEW-SHOT LEARNING APPROACHES

Data-level

Parameter-level

Metric-level

Gradient-based Meta-learning

4.3.2 FEW-SHOT LEARNING APPROACHES

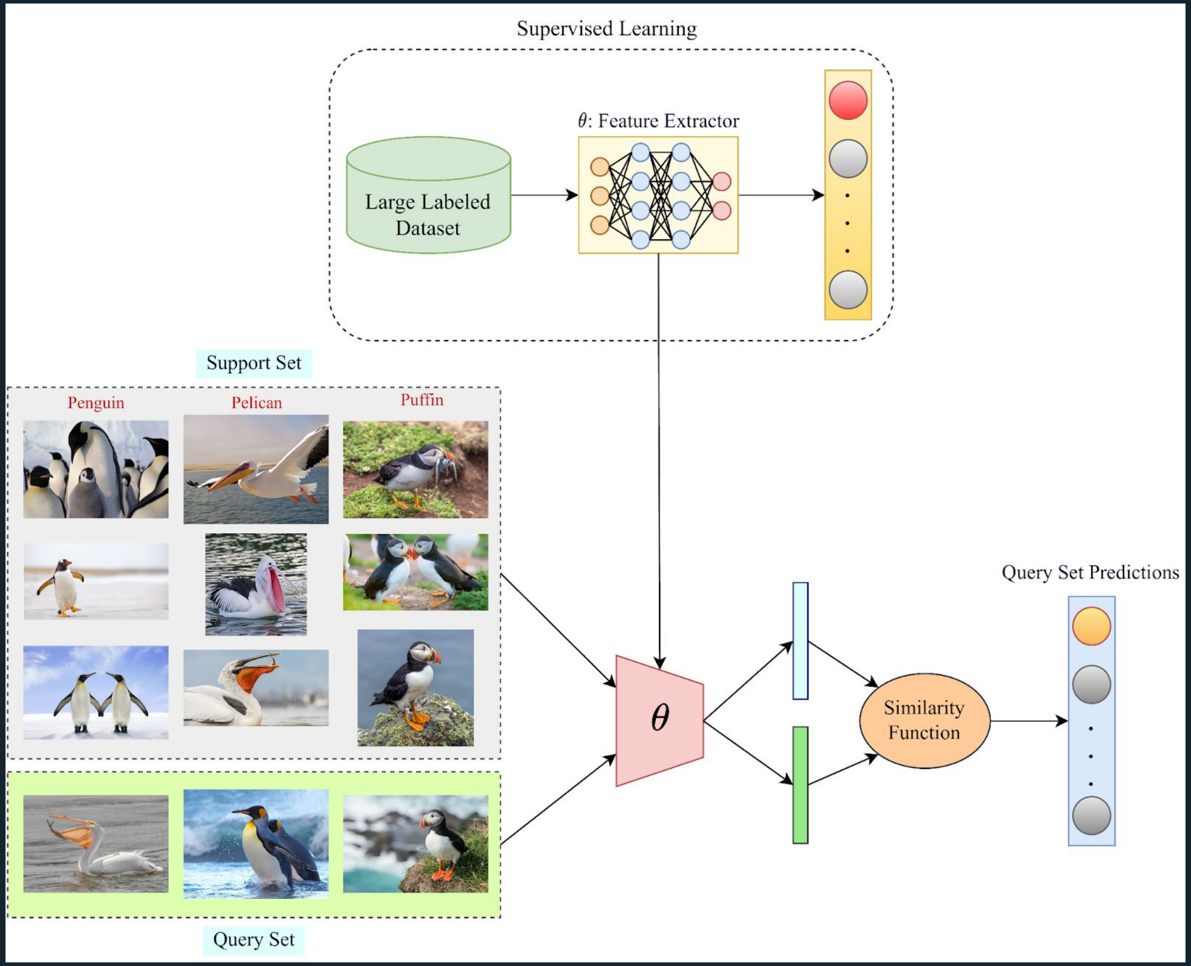
- Data-level:
  - Straightforward concept, enrich the training data for the model:
    - Provide more data for training
    - Data augmentation
    - Synthetic images
    - Use GANs

4.3.2 FEW-SHOT LEARNING APPROACHES

- Parameter-level:
  - To avoid overfitting and underfitting due to small training dataset, we can regulate model's parameters:
    - Parameter space restriction
    - Regularization
    - Loss function fine-tuning
    - Standard optimization

4.3.2 FEW-SHOT LEARNING APPROACHES

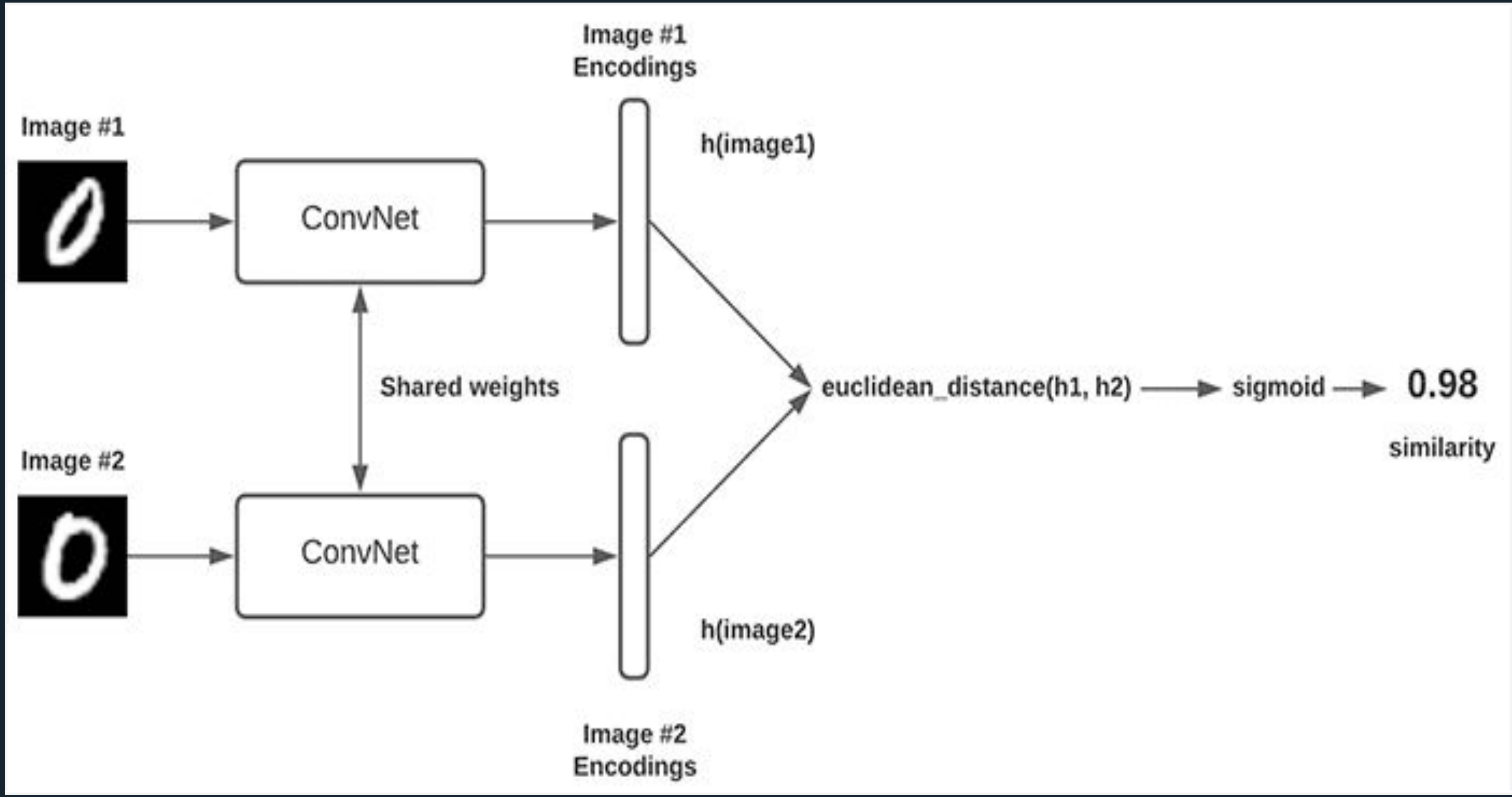
- Metric-level:



4.3.2 FEW-SHOT LEARNING APPROACHES

- Metric-level:
  - The primary idea in Metric-Learning is to find a similarity function that can map the similarities between the classes in the support and query sets.

4.3.3 SIAMESE NETWORK

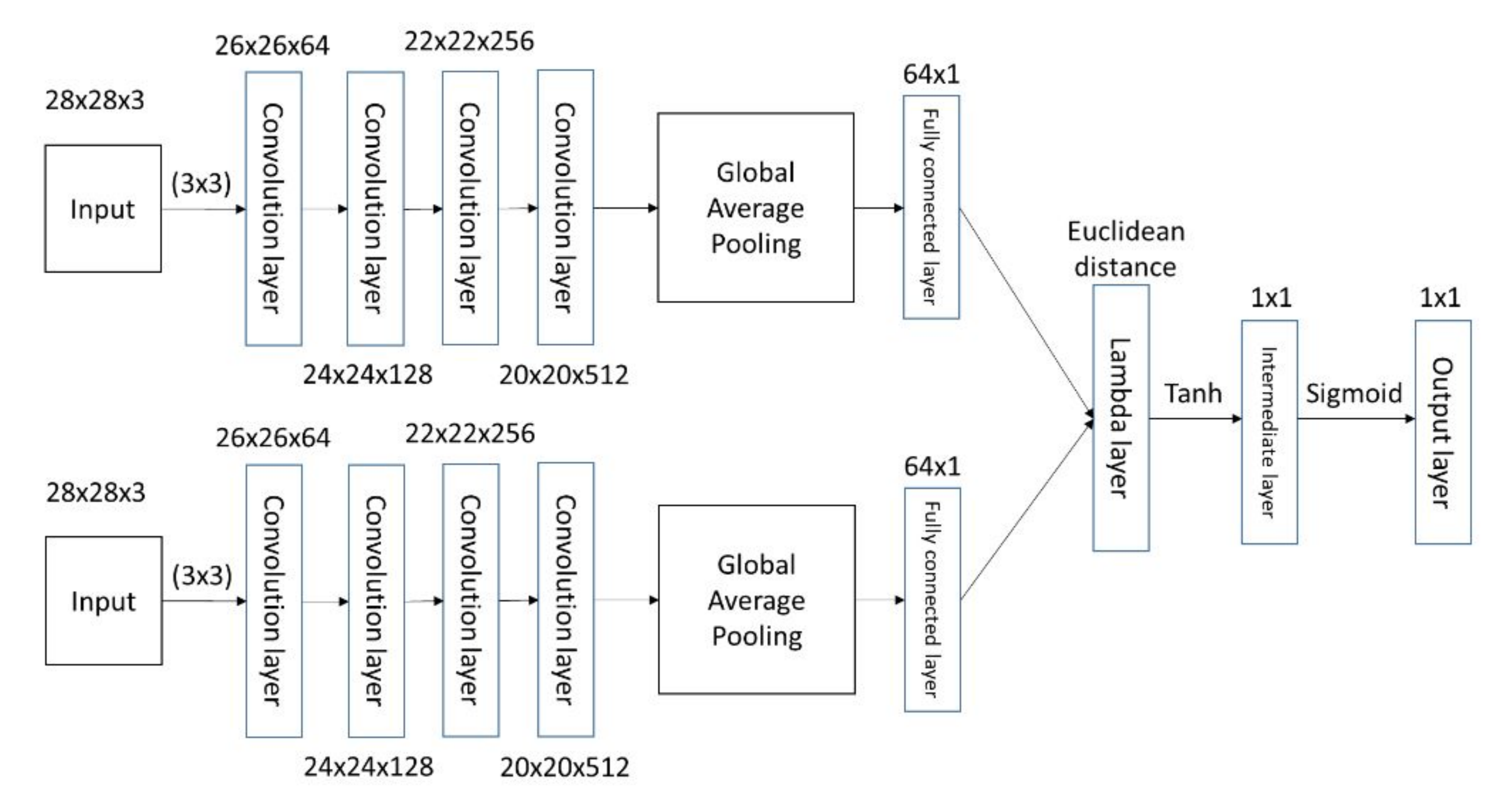


4.3.3 SIAMESE NETWORK

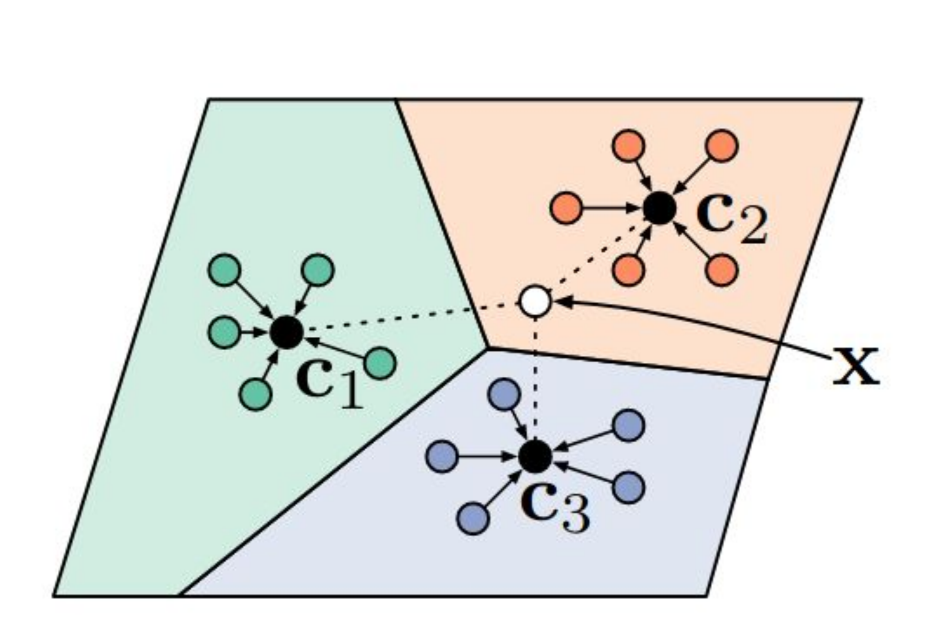
$$d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

- p, q: two points in Euclidean n-space
- q_i, p_i: Euclidean vectors, starting from the origin of the space (initial point)
- n: n-space

4.3.3 SIAMESE NETWORK



4.3.4 NEAREST PROTOTYPE CLASSIFICATION

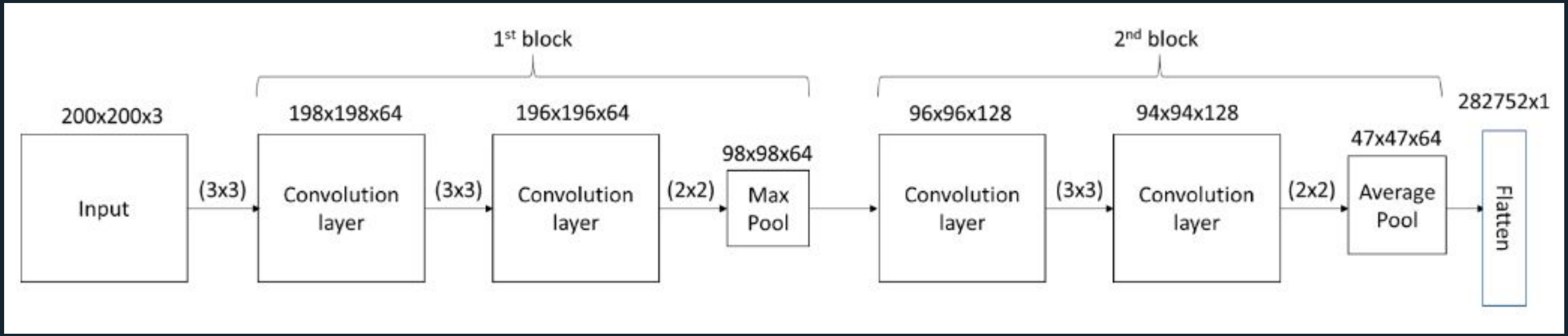


Nearest prototype classification is Metric Learning approach with Cx is the mean of each classes

Nearest prototype classification is to find the minimum distance between centroids and the new data point that needs to be classified.

4.3.4 NEAREST PROTOTYPE CLASSIFICATION

- We built a CNN to extract feature vectors then compute the mean feature vectors of every classes.

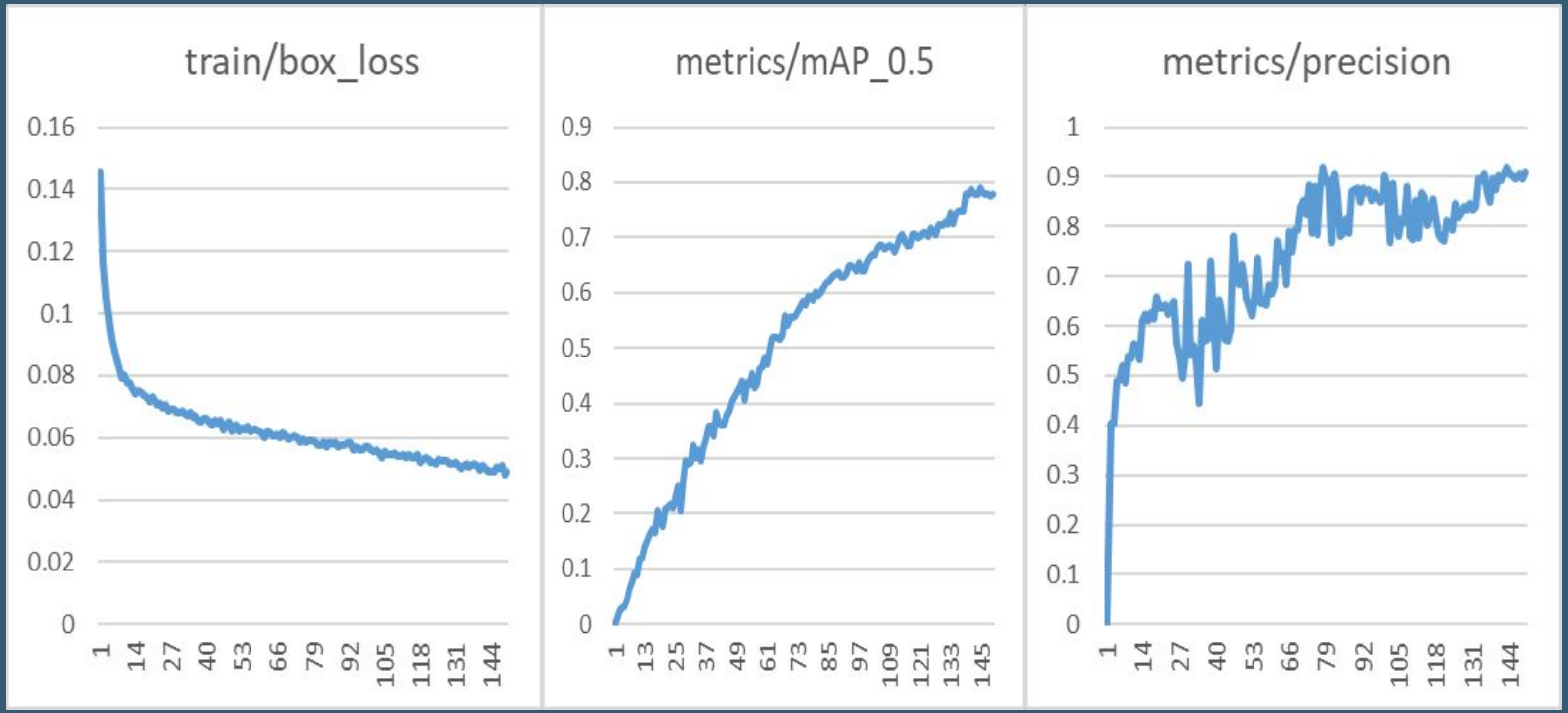


4.3.4 NEAREST PROTOTYPE CLASSIFICATION

- By computing the Euclidean distance between the feature vector of the new image and mean feature vectors of each class,we then find the class with the smallest distance from the new image’s feature vector

4.4

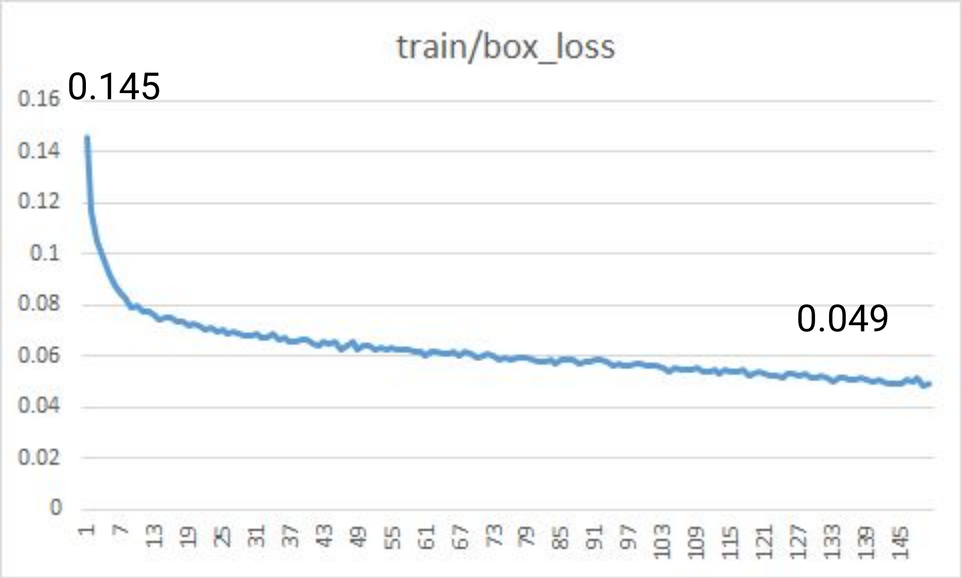
TRAINING AND RESULT

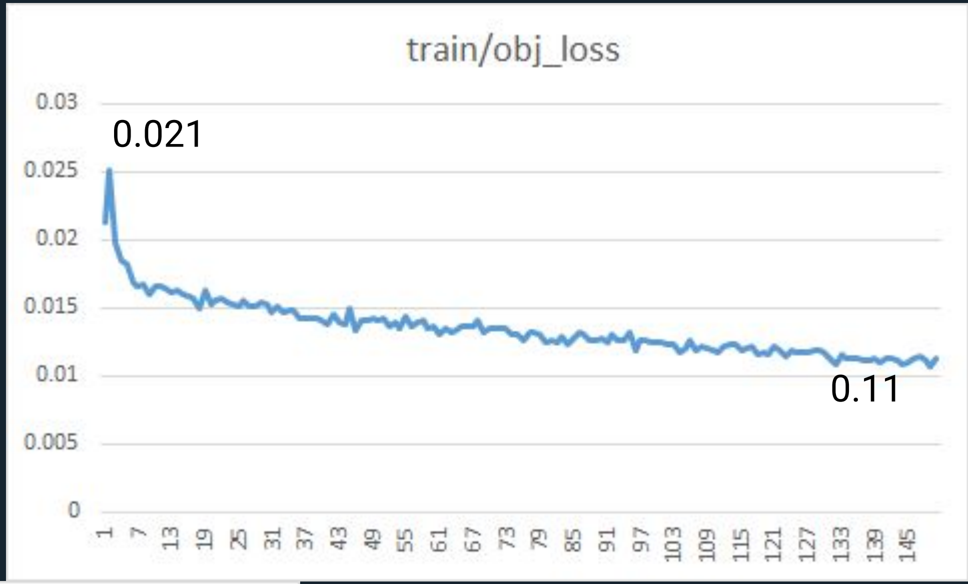


4.4.1 TRAFFIC SIGN DETECTION

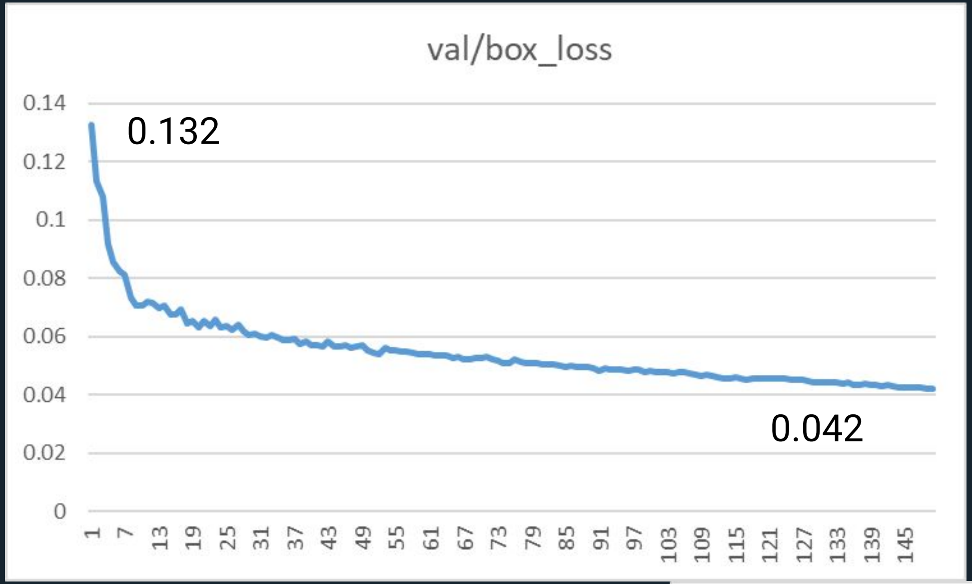
- GPU: AMD Radeon™ Vega 10 Graphics
- We trained YOLOv5 for 2 runs:
  - First run:
    - SGD optimizer, lr = 0.01, weight decay = 0.0005, momentum = 0.937
    - 150 epochs, batch size = 15, runtime = 51.06 mins
    - precision = 0.9, recall = 0.69, mAP 0.5 = 0.778 and mAP 0.5:0.95 = 0.544 after 150 epochs

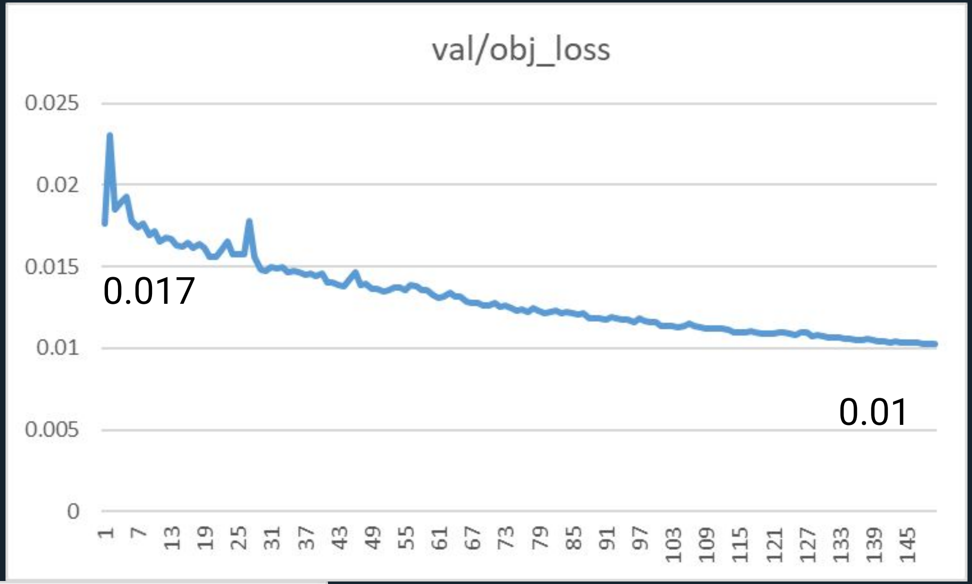
4.4.1 TRAFFIC SIGN DETECTION

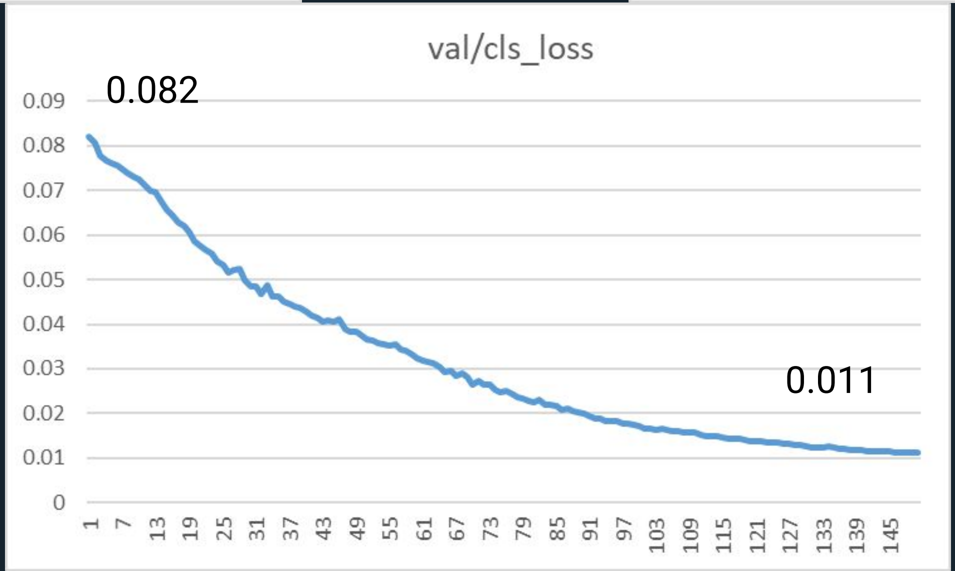




4.4.1 TRAFFIC SIGN DETECTION

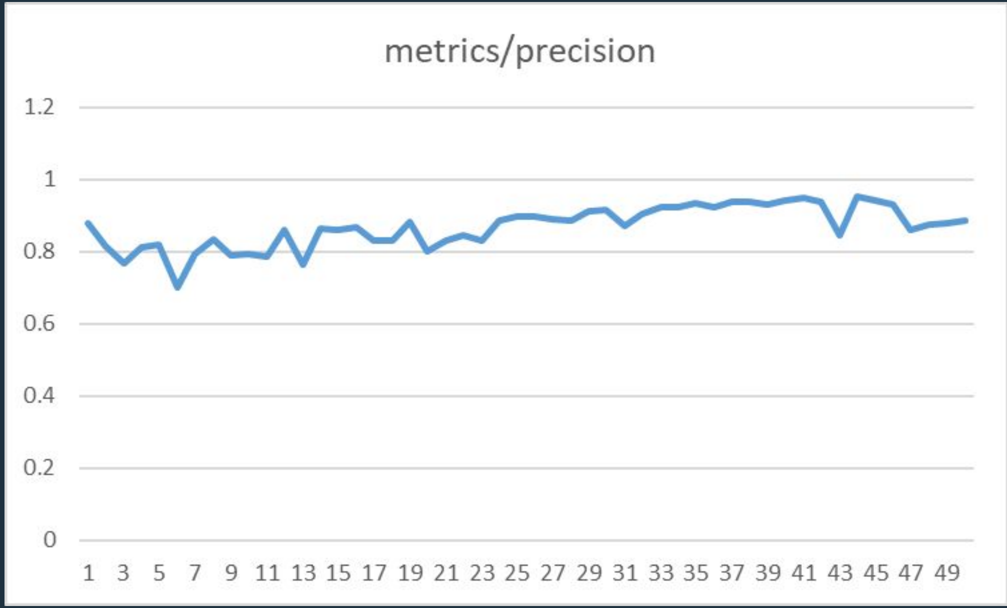


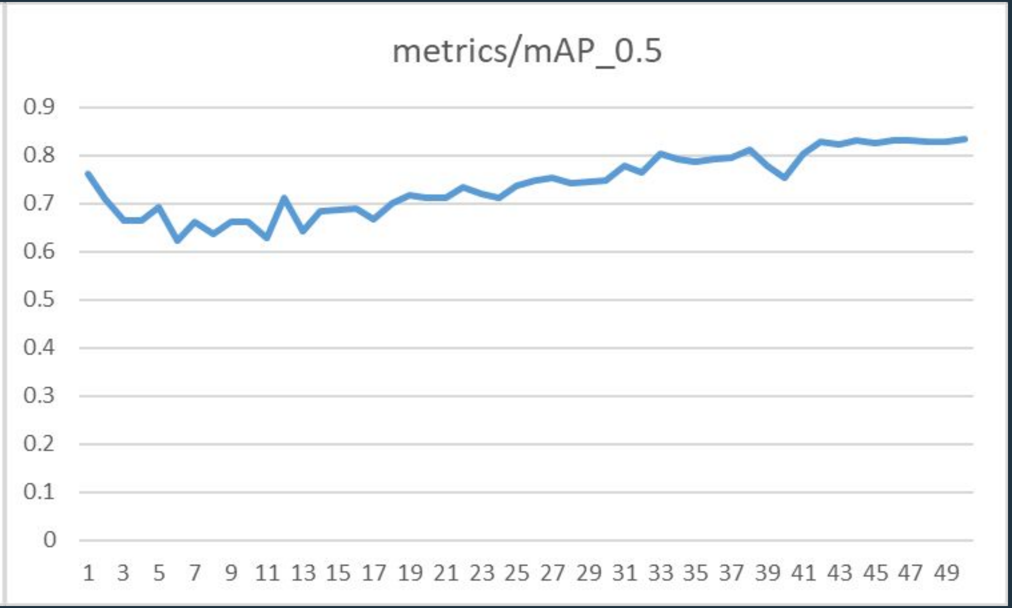




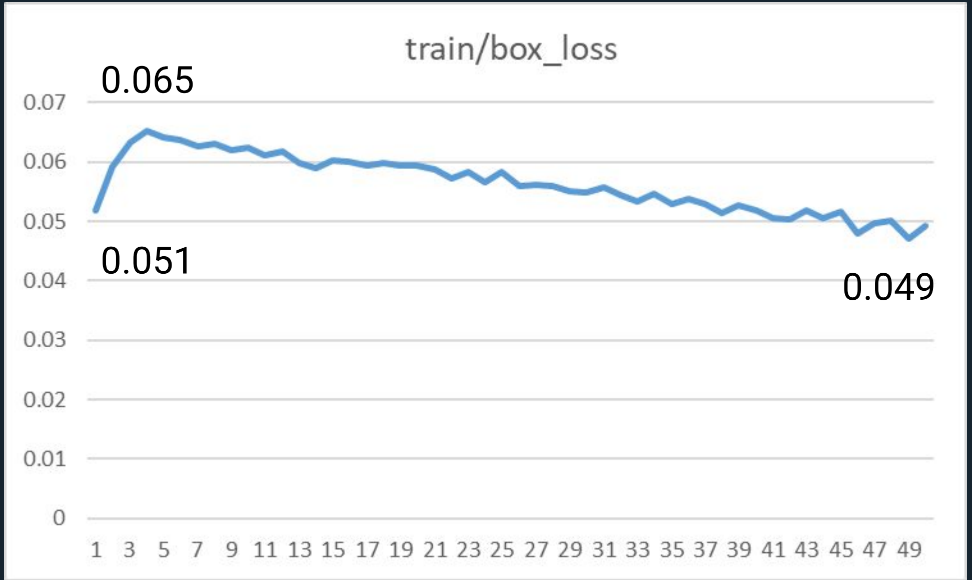
4.4.1 TRAFFIC SIGN DETECTION

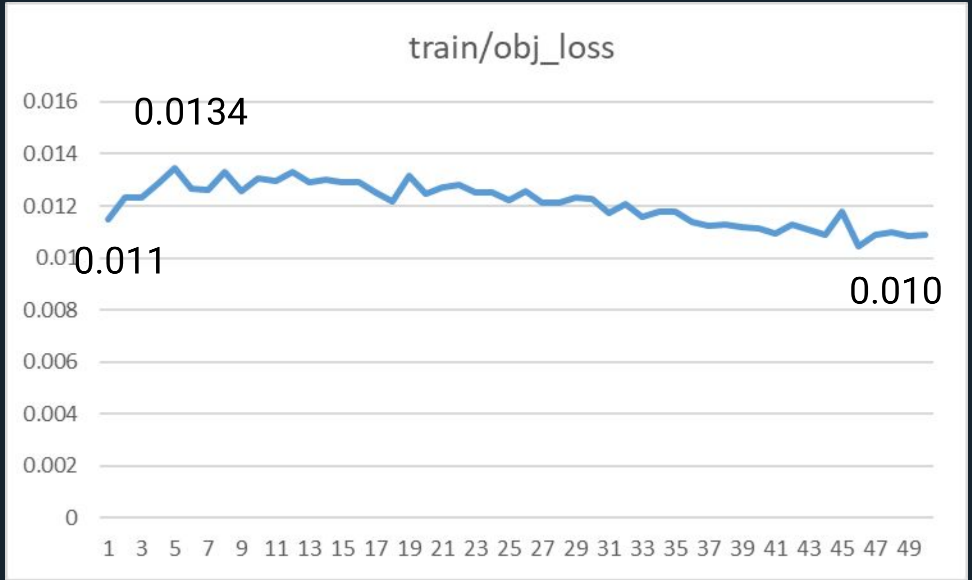
- Second run:
  - 50 epochs, runtime = 12.84 mins
  - precision = 0.887, recall = 0.78, mAP 0.5 = 0.83 and mAP 0.5:0.95 = 0.56 after 50 epochs

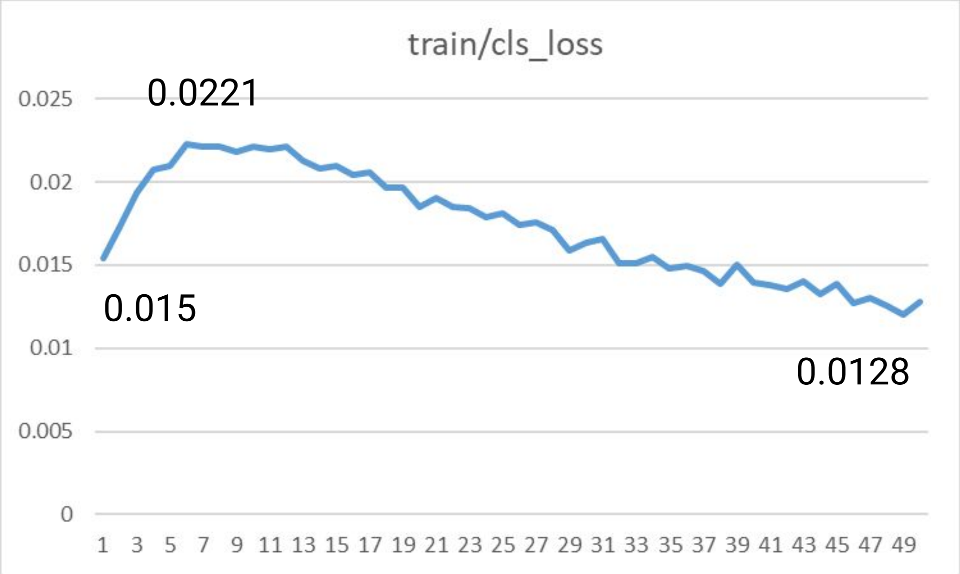




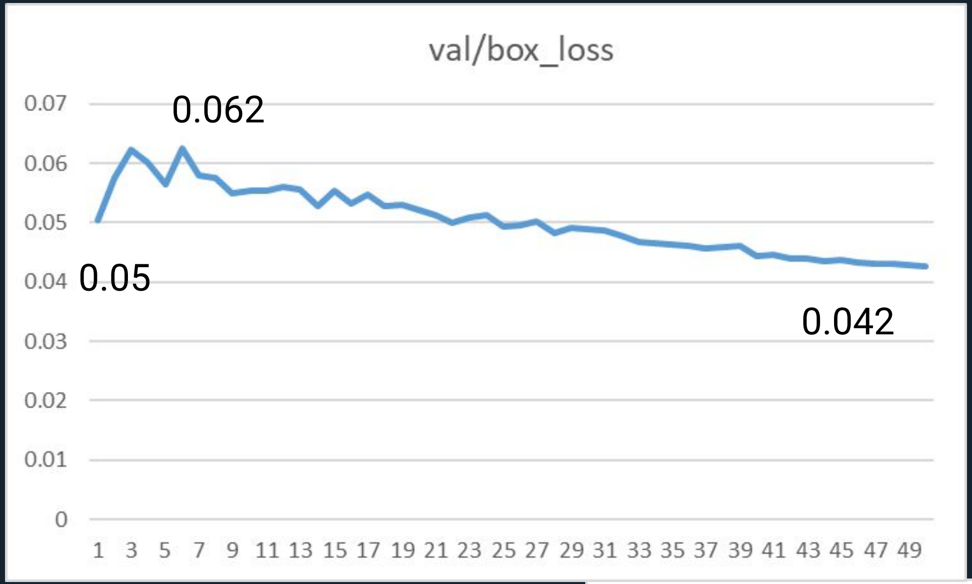
4.4.1 TRAFFIC SIGN DETECTION

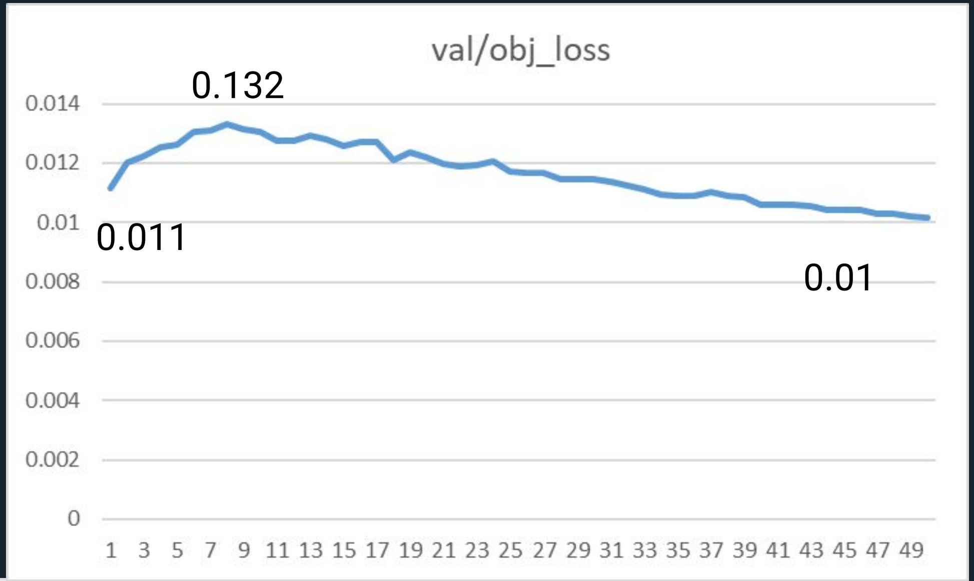


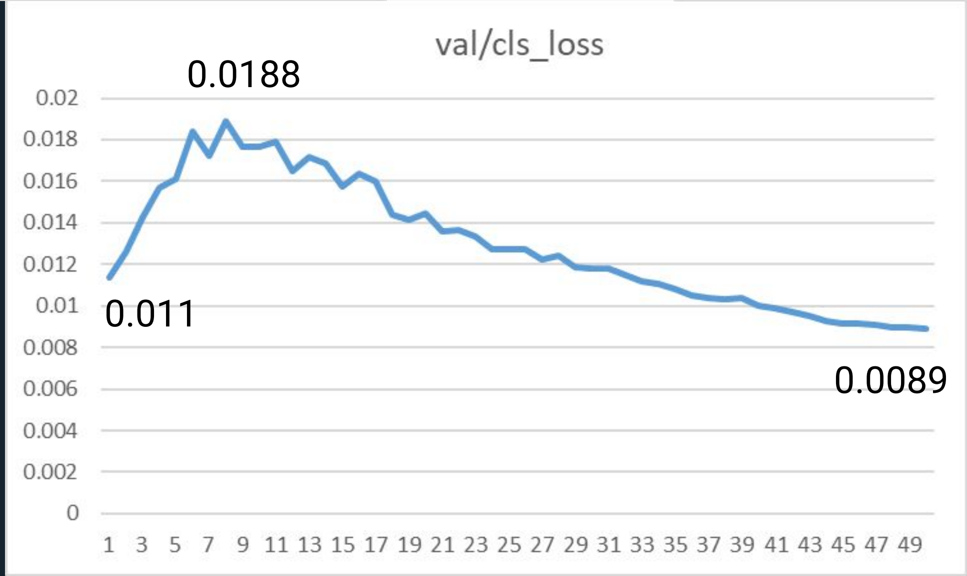




4.4.1 TRAFFIC SIGN DETECTION







4.4.1 TRAFFIC SIGN DETECTION

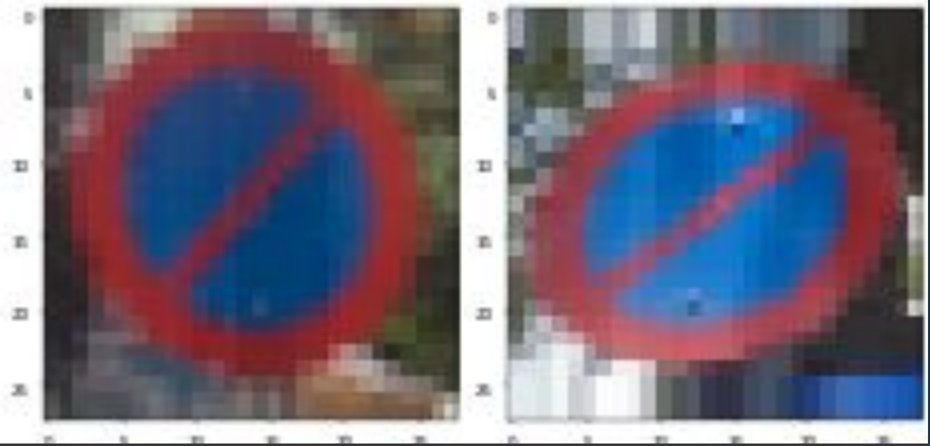
4.4.1 TRAFFIC SIGN DETECTION

- We cropped the detected result of traffic signs to build dataset for training few-shot learning method
- We paired images in support set and query set in all possible pairs


4.4.2 TRAFFIC SIGN RECOGNITION

- GPU: AMD Radeon™ Vega 10 Graphics
- We trained Siamese network
  - Adam optimizer, lr = 0.001, weight decay = 0.0005, momentum = 0.937, binary cross entropy loss
  - 100 epochs, batch size = 32, early stopping after 19 epochs, runtime = 26.65
  - accuracy = 0.9966, loss = 0.01 on the training set; accuracy = 0.997, loss = 0.0101 on the test set.

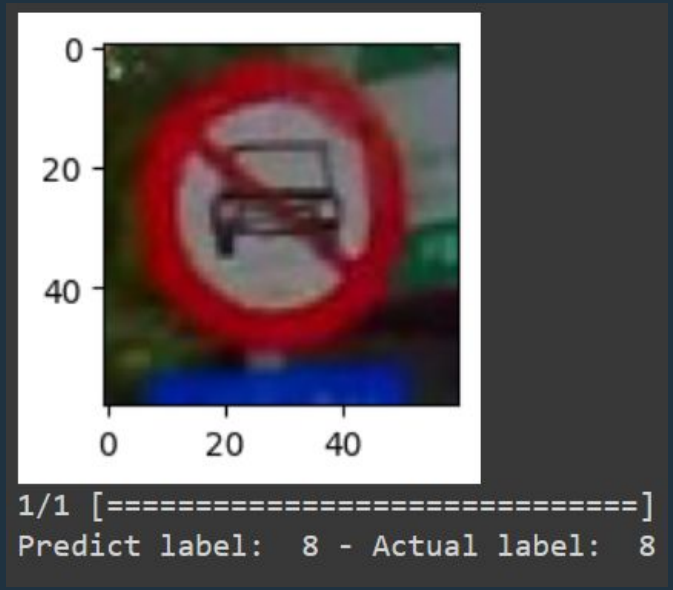
4.4.2 TRAFFIC SIGN RECOGNITION



```
1/1 [==============================] - 0s 21ms/step
Similarity score: 0.9880977
Same class
```

4.4.2 TRAFFIC SIGN RECOGNITION

- With Nearest prototype classification method, we achieve the accuracy on training set of 0.62 but this method performed well on test set with very quick runtime



05 CONCLUSION

In this research, we demonstrate how a computer vision system equipped with machine learning and hyper-learning approaches may enhance the accuracy with which drivers can recognize traffic signs using in-vehicle technologies.

Conclusion

- We had successfully detected traffic signs using YOLOv5 with high accuracy
- Even though YOLOv5 did not perform too well on classification tasks, YOLOv5 perform incredibly good on object localization tasks, which can be utilized for few-shot learning

Conclusion

- We used YOLOv5 results to train Siamese model, achieved accuracy of 0.997 on test set after 12 minutes of training
- We also use Nearest prototype classification(NPC) algorithm as a Metric-learning method and achieve high accuracy on unseen data
- Both Siamese network and NPC show good results with limited training data, short running time and efficient computational power.

CHALLENGES

- Required amount of data for YOLOv5
- Imbalance of data
- Data quality
- To optimize Siamese network performance

FUTURE WORK

- Enhance the models' performance with more traffic signs
- End-to-end few-shot detection
- Refine and extend model to be utilize in real world

THE END!

THANKS FOR WATCHING

<p> Do you have any question? </p>