Intelligent Traffic System [1]

Pham Thanh Phong
Founder
ITM Vision
phongpham663@gmail.com

Ha Quang Phuoc
AI Engineer
ITM Vision
hqphuoc129@gmail.com

February 26, 2021

[1]Special thanks for the guidance of Msc.Vo Phi Son

CONTENTS

List of Figures

Chapter 1

OVERVIEW ITS SYSTEM

1.1 ITS

1.2 Hardware for Inference

Chapter 2

OBJECT DETECTION

2.1 Overview

2.1.1 Traditional Methods

- Traditional object detection methods are built on handcrafted features and shallow trainable architectures
- Features used in traditional methods: color feature, HOG feature, edge feature, optical flow features, texture features,...
- The pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction, and classification

2.1.2 Deep Learning Based Method

2.1.2.1 CNNs Overview

- These methods all take advantage of CNN. CNN uses convolutional layers and pooling layers. Convolutional layers filter inputs for useful information. They have parameters that are learned so that filters are adjusted automatically to extract the most useful information for a certain task. Multiple convolutional layers are used that filter images for more and more abstract information after each layer. Pooling layers are used for limited translation and rotation invariance. Pooling also reduces the memory consumption and thus allows for the usage of more convolutional layers.
- Pooling layers provide an approach to down sampling feature maps by summarizing the presence of features in patches of the feature map. Two common pooling methods are average pooling and max pooling that summarize the average presence of a feature and the most activated presence of a feature respectively.
- Convolution operation is often interpreted as a filter, where the kernel filters the feature map for information of a certain kind. The convolution operation is usually known as kernels. By different choices of kernels, different operations of the images can be obtained. Operations typically include edge detection, blurring, sharpening etc. A convolutional layer is primarily a layer that performs convolution operation. Its main task is to map. The result of staging convolutional layers in conjunction with the following layers is that the information of the image is classified like in vision.
- The pooling layer is responsible for reducing the spacial size of the activation maps. Although it reduces the dimensionality of each feature map, it retains the most important information. There are different strategies of the pooling which are max-pooling, average-pooling and probabilistic pooling.
- After several convolutional and max pooling layers, the high-level reasoning in the neural network is done via Fully Connected Layers (FCLs). A FCL takes all neurons in the previous layer and connects it to every single neuron it has. FCLs are not spatially located anymore, that means they can be visualized as one-dimensional.The sum of output probabilities from the Fully Connected Layer is 1. This is ensured by using the Softmax as the activation function in the output layer of the Fully Connected Layer.

- A pooling layer is a new layer added after the convolutional layer. Specifically, after a nonlinearity (e.g. ReLU) has been applied to the feature maps output by a convolutional layer; for example the layers in a model may look as follows:

  1. Input image
  2. Convolutional layer
  3. Nonlinearity
  4. Pooling layer

2.1.2.2 Categorize

- Thanks to Deep Neural Network, a more significant gain is obtained with the introduction of regions with convolutional neural network (CNN) features (R-CNN). DNNs, or the most representative CNNs, act in a quite different way from traditional approaches. They have deeper architectures with the capacity to learn more complex features than the shallow ones. Also, the expressivity and robust training algorithms allow to learn informative object representations without the need to design features manually.
- Since the proposal of R-CNN, a great deal of improved models have been suggested, including fast R-CNN that jointly optimizes classification and bounding box regression tasks, faster R-CNN that takes an additional subnetwork to generate region proposals, and you only look once (YOLO) that accomplishes object detection via a fixed-grid regression.

2.1.2.3 Region Proposal Based Method

- Generate region proposals at first and then classify each proposal into different object categories. Frameworks may be included such as: R-CNN, Faster R-CNN, region-based fully convolutional network R-FCN, feature pyramid networks (FPN), and Mask R-CNN.
- Region proposal-based frameworks are composed of several correlated stages, including region proposal generation, feature extraction with CNN, classification, and bounding box regression, which are usually trained separately. Even in the recent end-to-end module Faster R-CNN, an alternative training is still required to obtain shared convolution parameters between RPN and detection network. As a result, the time spent in handling different components becomes the bottleneck in the real-time application.

2.1.2.4 Regression / Classification Based Method

- Regression/classification based framework solves object detection problem by regarding it as regression or classification problem. Some frameworks may be accounted for examples are: MultiBox, AttentionNet, G-CNN, YOLO, Single Shot MultiBox Detector (SSD), YOLOv2, YOLOv3.
- One-step frameworks based on global regression/classification, mapping straightly from image pixels to bounding box coordinates and class probabilities, can reduce time expense.

2.1.3 One Stage Algorithm

2.1.4 Two Stage Algorithm

2.2 Comparison One Stage and Two Stage

2.3 Comparison YOLO SSD Faster-RCNN

2.3.1 YOLO

2.3.1.1 Network Architecture

The whole system can be divided into two major components: Feature Extractor and Detector; both are multi-scale. When a new image comes in, it goes through the feature extractor first so that we can

obtain feature embeddings at three (or more) different scales. Then, these features are feed into three (or more) branches of the detector to get bounding boxes and class information.

Figure 2.1: Major components

2.3.1.2 Feature Extraction

The feature extractor YOLO V3 uses is called Darknet-53. Darknet-53 contains 53 layers and borrows the ideas of skip connections to help the activations to propagate through deeper layers without gradient diminishing from ResNet. But the Darknet-53 claims to be more efficient than ResNet101 or ResNet152.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 2.2: Feature Extraction of YOLOv3

Inside the block, there's just a bottleneck structure (1x1 followed by 3x3) plus a skip connection. If the goal is to do multi-class classification as ImageNet does, an average pooling and a 1000 ways fully connected layers plus softmax activation will be added. However in the case of object detection, the classification head won't be included, instead, the "detection" head will be added to this feature extractor. Features from last three residual blocks are used in the later detection.

2.3.2 SSD

2.3.3 Faster-RCNN

2.4 Metrics and Evaluations

Chapter 3

OVERVIEW MULTIPLE OBJECT TRACKING

Chapter 4

INFERENCE

4.1 Architecture Optimization

4.2 Algorithm Optimization

4.3 TensorRT Framework

4.4 Post-Processing

4.4.1 Speed Estimation

4.4.2 Anomaly Detection

Chapter 5

MODEL TRAINING

5.1 Dataset

5.2 Loss Function

5.3 Hyperparameters

Chapter 6

SOFTWARE DESIGN

6.1 System Overview

6.2 Hardware Design

6.3 Software Design

6.3.1 Dependencies

6.3.2 User Interface

6.3.3 Administration