Learning to Detect Pedestrians by Watching Videos

Andrew Y. Chen
The University of Texas Rio Grande Valley

LEARNING TO DETECT PEDESTRIANS BY WATCHING VIDEOS

A Thesis

by

ANDREW Y. CHEN

Submitted to the Graduate College of
The University of Texas Rio Grande Valley
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2019

Major Subject: Computer Science

LEARNING TO DETECT PEDESTRIANS BY WATCHING VIDEOS

A Thesis
by
ANDREW Y. CHEN

COMMITTEE MEMBERS

Dr. Hongkai Yu
Chair of Committee

Dr. Jingru Zhang
Committee Member

Dr. Bin Fu
Committee Member

Dr. Emmett Tomai
Committee Member

December 2019

ABSTRACT

Chen, Andrew Y., Learning to Detect Pedestrians by Watching Videos . Master of Science (MS), December, 2019, 35 pp, 2 tables, 21 figures, 44 references.

The field of deep learning has experienced a resurgence in the recent years, particularly resulting with the advent of Convolutional Neural Networks. Supervised learning is currently the most common and practical machine learning method. The struggle with employing supervised learning to approach problems is that it requires a large number of training data. Sufficient training data is correlated with performance for deep learning models. The issue is that preparing the training data can be a tedious and labor intensive task, especially on a large scale. The purpose of this thesis is to determine how efficient a machine can learn when trained on automatically annotated data. The goal is to determine whether or not automatically generated training data is viable for machine learning. In this thesis, we focus on the research problem of pedestrian detection from images. This proposed method is able to automatically label data via background subtraction based pseudo ground truth and train regular and deep classifiers for pedestrian detection. The methods are evaluated on a custom dataset and the experimental results show the effectiveness and accuracy of the proposed method.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Hongkai Yu, for the guidance and mentorship over the course of this thesis. I would also like to Dr. Jingru Zhang, Dr. Emmett Tomai, and Dr. Bin Fu for serving in this committee. I want to also thank my family for supporting me during my studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Computer vision and machine learning have always been hard problems in computer science. Is it possible for computers to learn to see like humans can? The brain gives humans the capacity to recognize text and read or differentiate between a monkey and another human. While these tasks are fathomable to humans, it has been a difficult problem to solve using computers. However, in the recent years, there has been major progress in the field of computer vision with the rediscovery of deep learning methods. Groundbreaking work with deep Convolutional Neural Networks has shown that, with the current potential available from computer hardware, computers have matched or exceeded human performance in some domains of visual recognition tasks. As humans, we are able to subconsciously process images at any given point all day. How is it possible for us to see, label, classify, and recognize patterns?

With the development of neural networks and deep neural networks, the computers have been able to better see, label, classify, and recognize patterns recently. We have developed architecture of neural networks based on our understanding on the human brain. We've also developed methods to model how humans process visual images. This research dates back to the 1950s and 1960s where D.H Hubel and T.N Wiesal were able to study and model the brain of mammals [17]. They were able to demonstrate that the neuron's in the visual cortex of a cat and monkey responded directly to the perceived environment. Hubel and Wiesal discovered that there were two types of visual neuron cells, simple cells and complex cells [17]. In vision, there is the notion of a receptive field. This field highlights the region of retina which activates the neuron. The first Convolutional Neural Network was pioneered by Le Cun, Bengio, Bottou, and Haffner for their work with LeNet [20]. Their network shed some light on CNNs, a special kind of multi-layer neural networks that, like most neural networks, are trained with some form of back-propagation. CNNs are designed to, with minimal preprocessing, take an image input and recognize visual patterns. The group produced LeNet which was able to take in images of hand-written numbers and classify each digit. LeNet architecture is fundamental in this field because it provided insight about images and their features. Then, more powerful Convolutional Neural Networks are proposed for more accurate performance, such as AlexNet [19], VGG Net [36], ResNet [14], Fully Convolutional Networks [24], DenseNet [16], and so on. Although these deep neural networks achieved high performance in many vision tasks, they all need a large number of manually annotated ground truth, which are expensive and sometimes hard to obtain in some real-world applications.

Sufficient training data is correlated with performance for deep learning models. The issue is that preparing the training data can be a tedious and labor intensive task, especially on a large scale. The purpose of this thesis is to determine how efficient a machine can learn when trained on automatically annotated data. The goal is to determine whether or not automatically generated training data is viable for machine learning. In this thesis, we focus on the research problem of pedestrian detection from images. The proposed method in this thesis is able to automatically label data via background subtraction based pseudo ground truth and train traditional machine learning and deep learning models for pedestrian detection. The methods are evaluated on a custom dataset collected by ourselves and the experimental results show the effectiveness and accuracy of the proposed method.

1.1 Computer Vision and Deep Learning

The basic mechanism behind a neural network is taking an input through hidden layers. These layers are columns of neurons which are fully connected to every neuron in the preceding layer. After the input goes through all the layers, it will reach a last fully connected layer where the output will be predicted. The final classifier layer contains classification scores for all possible predictions. To summarize, neural networks receive a vector in parameters as input, take the input through a series of hidden layers of neurons that are fully connected to all neurons in previous layers, and then through a final fully connected layer containing scores for each classification.

Convolutional neural networks are similar to regular neural networks like the one discussed above. The layers contain neurons that have weights and can learn. The neurons in each layer receive inputs from the preceding layer, performs the dot product, followed by an optional non-linearity. The main architectural differences of a CNN vs a regular neural network comes from the fact that CNNs are explicitly designed to take an image as the input. This allows for the design of the network to be geared towards certain desirable properties which can reduce the amount of parameters needed and allow for have an efficient feed forward function.

Convolutional neural network architectures are geared towards handling the nature of an image's input. A CNNs layers have a 3-dimensional structure to consider: width, height, and depth. Rather than having all the neurons connect to the neurons in the preceding layer, only a small region is connected. By the time the input reaches the final layer, it will have been transformed into a single vector of scores for the possible classification or prediction. Each layer between the input and the final output layer will be a series of hidden layers that have these basic components: convolutional layer, rectified linear unit (ReLU) layer, or a pooling layer. The two main tasks that are done by the layers of a CNN are: using hidden layers to extract features and using the final layer as a classifier.

1.2 Challenges for Computer Vision

Computer vision deals with image processing techniques to extract information from digital images and uses machine learning methods to build models on the information to understand and recognize certain features or patterns. Most of machine learning methods depend on the supervised learning method that requires a lot of ground truth. Supervised learning requires guidance, hence "supervision". There exists prior knowledge about the data and what they've been labeled. The machine eventually optimizes a function that maps an input data to its appropriate label. Supervised learning tasks generally deal with classification or regression.

The purpose of this thesis is to explore methods to help the tedious manual process of obtaining the ground truth. The hardest problem about training machines is not the learning algorithms or the methods, but in data preparation and labeling. When considering the ground truth data, it would be better to have more data to help the machine better understand the data and produce a more effective mapping function. What would it take and how much data is necessary? Sufficient training data correlates with good performance. Obtaining a large scale of ground truth data is extremely tedious and labor intensive. Some annotation tasks are more difficult to achieve than others. Taking pedestrian detection as an example, it usually takes several seconds to minutes to annotate each pedestrain in one image. There are some annotation tools for the ground truth annotation [31], however they are still time-consuming.

1.3 Pedestrian Detection

As we have witnessed with the recent breakthroughs in computer vision research, it is possible for computers to be trained to beat human performance in certain domains of image recognition tasks. Computers are tools that can make our every day lives easier. If our tools could see, we can live in a society where we can build our cars to drive safer and smarter on the road. One phenomenon of large interest is autonomous driving. Delivering reliable autonomous vehicles to your everyday consumer is a mammoth of a challenge. Creating a fully autonomous vehicle requires systems that have a multitude of components, sensors, and data. One important system for autonomous vehicles is detecting pedestrians. Pedestrian detection is a sub-problem for object detection.

We want to design a framework that can train a pedestrian detector by watching videos. A key feature of this framework is that it utilizes automatically generated training data. Image recognition tasks, especially those that are using supervised learning algorithms, require labeled or annotated data. As mentioned above, the task of annotating data on a large scale can be labor intensive. This thesis will evaluate whether or not it is viable to use automatically annotated data to train a pedestrian detector.

The proposed framework generates pseudo-labeled data by converting static videos into images, computing the background, and performing background subtraction to each frame to get the foreground mask. This foreground mask are then used to crop out the foreground objects, which in this case are pedestrians. These pedestrian crops are then treated as the positive samples. The feature extraction method being used, is the histogram of oriented gradients (HOG). Using HOG for feature extraction will reduce the input and feature map size as the patch goes through the classifier. Negative samples for training will be randomly cropped from the target testing dataset. This allows the classifier to adapt to its target domain. The positive samples are obtained from a public dataset [4] and our collected custom dataset. This framework is inspired by [21] and the train a machine for human segmentation by watching videos. Training and testing are based on the object proposals in the images. Non maximum-suppression (NMS) is used to eliminate redundant overlapping bounding boxes.

We try to solve pedestrian detection in two ways: several machine learning methods based on SVM and a deep learning method based on YOLOv3.

CHAPTER II

RESEARCH BACKGROUND AND RELATED WORK

Computer vision involves recognition tasks such as: classification, localization, segmentation, saliency, pattern recognition, etc. We want to create a framework that can help us automatically label human crops for training a pedestrian detector. When considering this task, it is important to consider two questions. Is there a way to automatically annotate images? Can we successfully train a machine for pedestrian detection with these automatically annotated images? These automatically annotated images are called pseudo ground truth in this research.

2.1 Pedestrian Detection

Pedestrian detection has wide applications, such as autonomous driving [35, 2, 26, 3, 25, 39, 40, 18], surveillance [41, 1, 28], and action recognition [42, 4]. For example, in autonomous driving, there have been efforts to produce datasets to aid in creating systems that can visually understand complex urban street scenes, such as CityScapes [6]. A more narrow subset of interest in understanding street scenes is being able to detect pedestrians. Being able to distinguish between pedestrians and other objects on the road is an essential element of smart vision. This has become so apparent that, in efforts to achieve better performance, Zhang et al. created a dataset to help train pedestrian systems to be able to generalize over multiple benchmarks [40].

2.2 Background Subtraction

Detecting moving objects is a vital part of computer vision systems. A common approach for finding moving objects is background subtraction [29, 37]. A simple way to detect moving objects in a static video sequence is through background subtraction [37]. There are several methods that are able to segment the objects in the foreground from the background. The background in this framework is computed by calculating the average of all the frames in the video sequence. The foreground mask for each frame can be obtained by subtracting the computed background.

The biggest limitation of this approach is that finding the background is more difficult in a sequence where the background changes in each frame. To consider this limitation, the positive samples of human crops were taken from video sequences that were static, where the camera was still and the background did not change frequently.

2.3 Pseudo Ground Truth

The motivation behind this thesis is to find a way to eliminate the need for human annotation. Creating datasets such as ImageNet [8], CityScape [6], or CityPerson [40] have been shown to be labor intensive tasks. We hope this framework can show that there are tools that can automate our annotation process. We also understand that algorithms should be given the proper training in not only quality, but quantity. We hope that automating data labeling can also help us give the algorithms the proper training that they need.

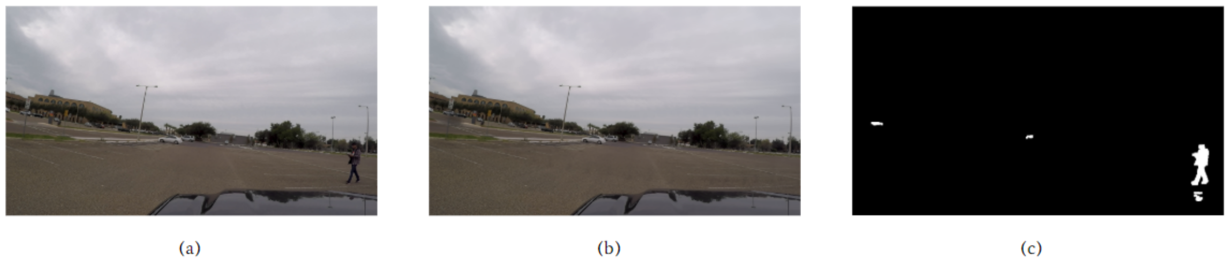

Figure 2.1: Example of background subtraction on our custom dataset: (a) a frame in one sequence in custom dataset, (b) the corresponding background for that sequence, (c) the resulting foreground mask from the background subtraction.

Background subtraction has been shown to be effective for detecting moving objects through a static video in Figure 2.1. We realized that, with a static camera, we can use this as our tool to locate moving pedestrians across a scene. The custom dataset that we collected on our campus has several sequences where the camera does not move. Those sequences were used for background computation and subtraction for annotation. To keep the task simple, we used sequences where there is only one pedestrian present.

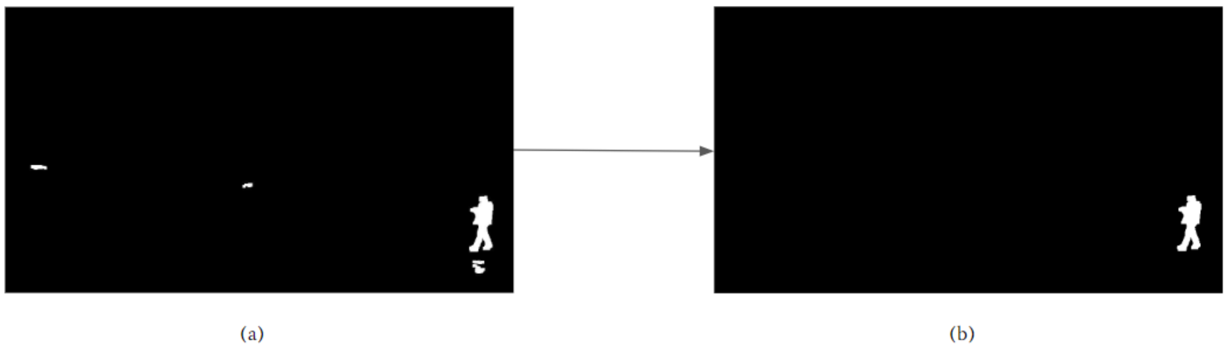
Figure 2.2: Example of cleaning foreground masks: (a) the initial foreground mask, (b) results of the final clean foreground mask.

A few foreground masks from the sequence require cleaning because of noise. To clean the foreground mask we use a few image processing techniques such as image dilation to fill in empty pockets and missing heads, legs, or arms. After image dilation, we find all pixel groups by finding connected components and keeping only the largest pixel group, our object of interest (the pedestrian) as shown in Figure 2.2 and Figure 2.3.

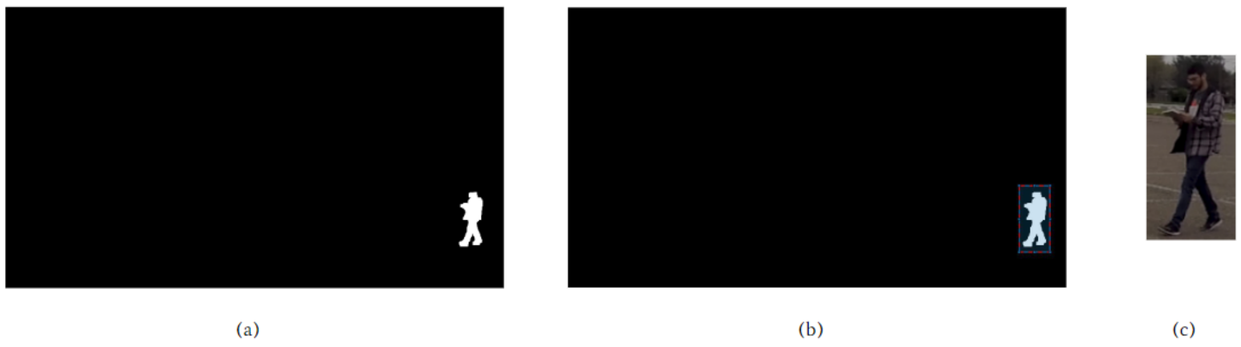
Figure 2.3: Example of pseudo ground truth bounding boxes: (a) a foreground mask for a frame in a sequence, (b) the bounding box coordinates from that frame, (c) the resulting crop from those bounding box coordinates.

This is the background subtraction based data annotation flow that our framework follows to generate the pseudo ground truth. Because the sequences contain only one pedestrian, it is a simple method to locate bounding box coordinates for object detection. The same will be applied to the public dataset [4] as shown in Figure 2.4. The negative samples for training are generated by selecting random sized boxes in the target testing set as shown in Figure 2.5. These positive and negative samples are used to create the training datasets for our machine learning and deep learning methods.

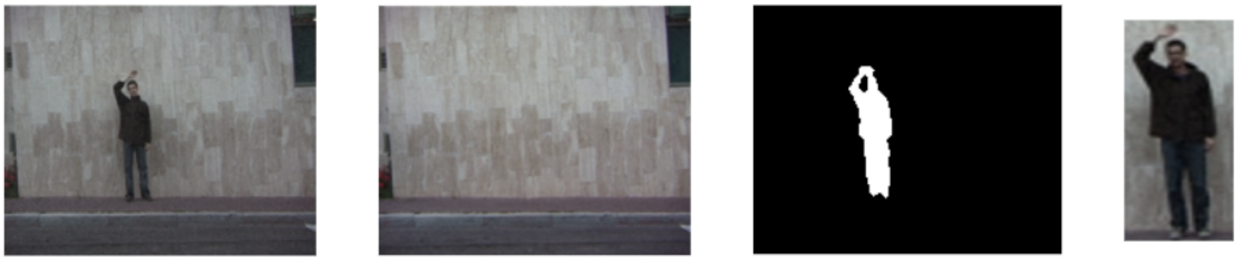

Figure 2.4: The same background subtraction based data annotation is applied to the public dataset [4].

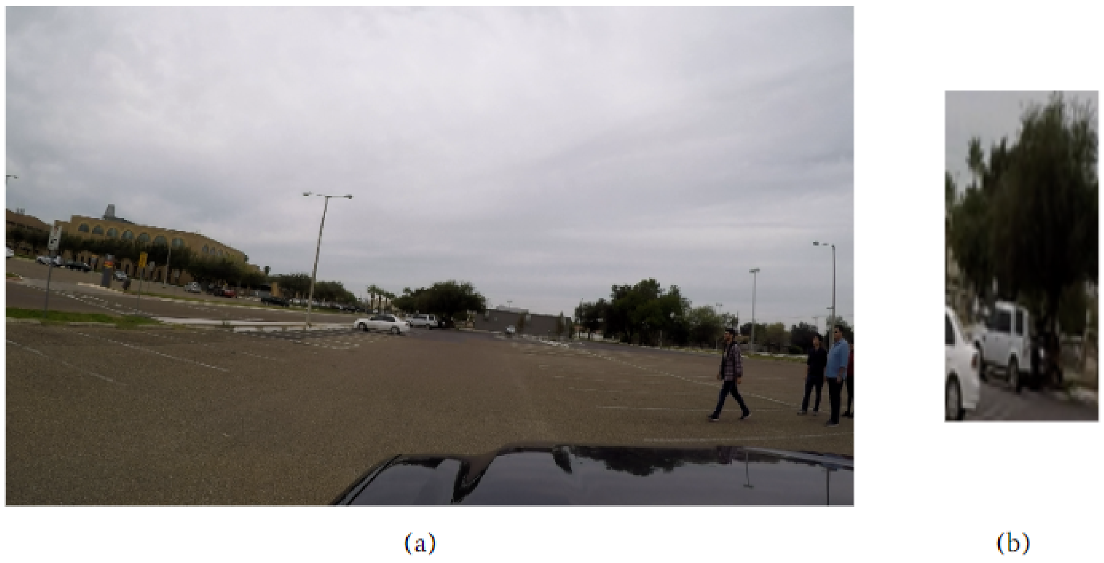

Figure 2.5: Randomly cropped negative samples: (a) original image, (b) randomly cropped and resized negative patch.

CHAPTER III

MACHINE LEARNING BASED PEDESTRIAN DETECTION

3.1 SVM Classifier

A common task that involves supervised learning is classification. One fundamental supervised learning method for classification are Support Vector Machine (SVM). This group of classifiers formally separate classes by a hyperplane. The function, given labeled training data, will determine which class an input belongs to. In the most simplest case of two dimensional space, a line will categorize the data into their respective class as shown in Figure 3.1.

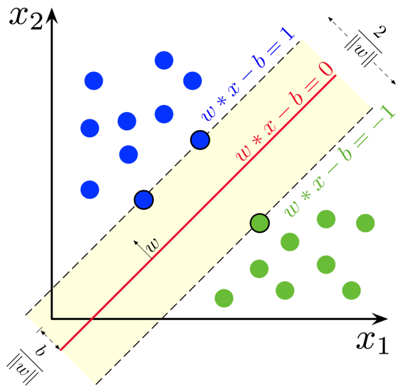

Figure 3.1: Example of Support Vector Machine (image taken from [38]).

Our machine learning based framework uses the SVM as the classifier. It is a simple yet effective method for classification.

3.2 HOG as Feature Extraction

An image can be structured as a 2D matrix. An image that is 500×500 in size has 250,000 elements. This can require extensive computation to process. To consider this expense, histogram of oriented gradients (HOG) is employed. In addition, HOG is considered as an effective feature representation for the object shape and structure.

In [37], they show that gradients, orientation binning and local contrast normalization are all important for human based feature representation. They were able to experimentally show that HOG descriptors outperform most existing feature extraction methods for human detection. Their work produced nearly perfect results on some public pedestrian benchmarks [3].

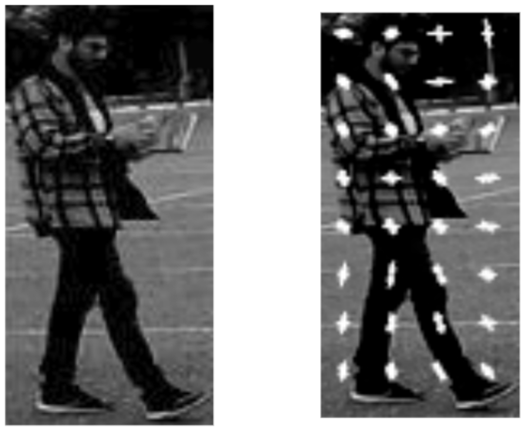

Figure 3.2: HOG features on one custom data crop with 15x15 masks.

Based on the success of HOG features in human detection, our machine learning framework utilizes HOG for the feature extraction phase of processing the image. Every sample is resized to 128×64, which originally produces a feature map of size 8192. Using HOG with 15×15 masks reduces the feature map to the size of 756.

3.3 Proposal Generation

A common component for object detection systems is to generate proposals. It is not reasonable to search every possible bounding box in an image. To resolve this issue, object based proposal generation is used to focus on the object regions. For our framework, we also generate proposals for further pedestrian localization.

3.3.1 Edgebox Proposals

Proposals of objects has been shown to be an effective approach to deal with the computational complexity of object detection. It can be exhaustive to run a detector through each every portion of the image. The method Edgebox [44] is able to generate bounding-box proposals for objects in an image. Edgebox proposals have been proven successful in many computer vision tasks, for example, tracking randomly moving objects [43].

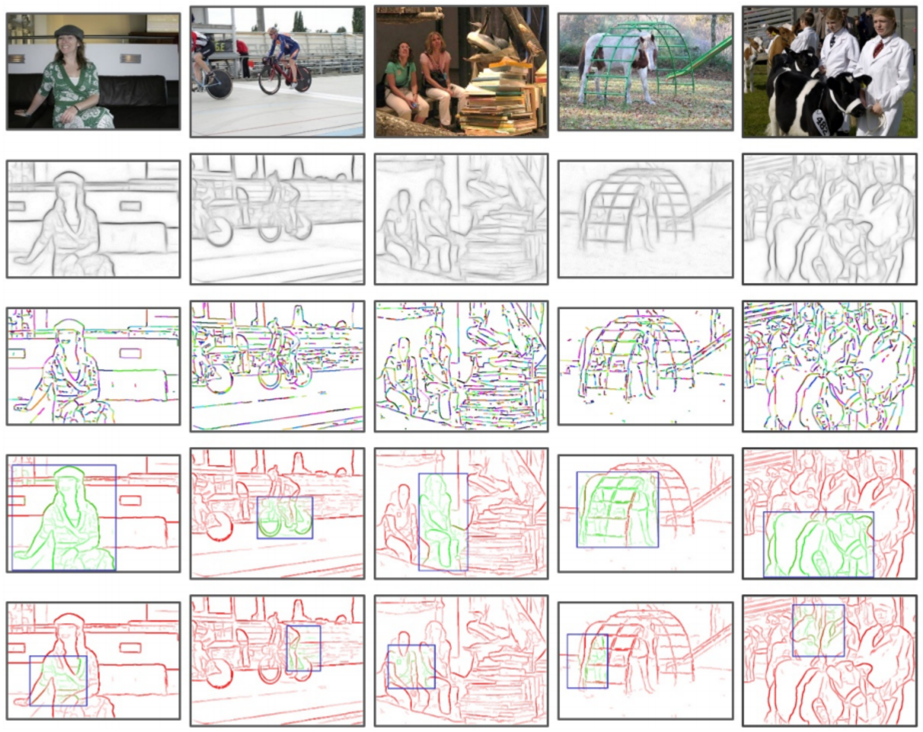

Figure 3.3: Edgebox proposal examples for objects (image taken from [44]).

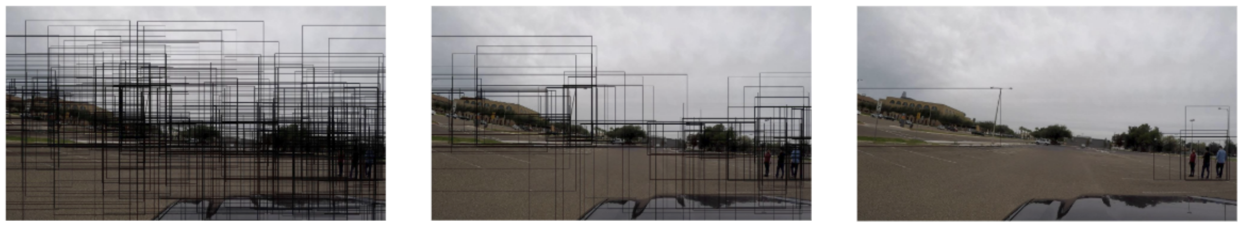

Figure 3.4: Example of Edgebox proposals. From left to right: 1000 proposals, 200 proposals, and 10 proposals.

Edges and contours can provide key insights about an image. Contours that are completely contained within a bounding box might be indicative of a complete object inside, as shown in Figure 3.3. Our framework will test utilizing Edgebox proposals to find complete objects in an image. When processing an image for human detection, up to 1000 object proposals are generated. Depending on the domain, bounding-box proposals that are too big or too small will be deleted as shown in Figure 3.4. After filtering, each bounding box area is cropped and sent to the classifier to determine if the object is a pedestrian.

3.3.2 YOLOv3 Proposals

One of the important components in many object detection systems is the proposal generation. To demonstrate the effects of the proposal generation, we test methods with a strong proposal method by using a pre-trained YOLOv3 human detector [31] as the proposals. What we expect to see is that the classification results from YOLOv3 proposals should have much better performance than using Edgebox as the proposal method.

3.4 Non-Maximum Suppression

The number of Edgebox proposals generated is large scale that there will be numerous boxes that have a high percentage of intersection. This is also a common issue that object detection tasks encounter [27, 15]. If multiple detections are highly overlapped, some of them with high overlaps and lower confidences should be removed. This procedure is named as Non-Maximum Suppression (NMS), as shown in Figure 3.5.

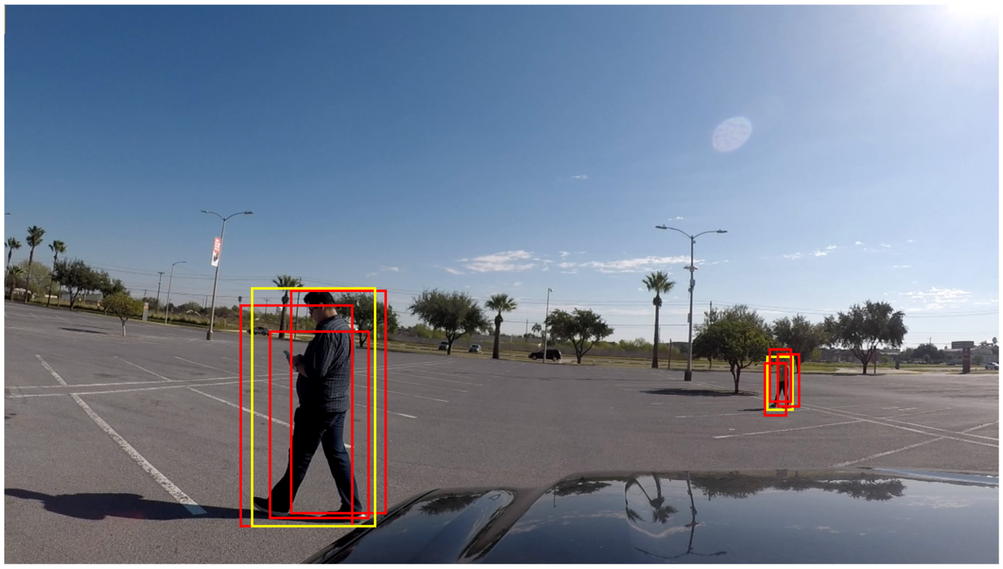

Figure 3.5: Example of NMS. Given two pedestrians shown in this figure, multiple proposals might be detected, but they are highly overlapped, so NMS removed the detections with high overlaps and lower confidences. In this case after NMS, only the yellow box would be kept as the detection result.

Edgebox proposal is not perfect, one bounding-box proposal might cover multiple objects, and multiple bounding-box proposals might cover one object. NMS can be used to ignore bounding boxes that overlap each other and reduce detection errors.

3.5 Training

The framework accomplishes two things: 1) automatically labeling and generation of training data and 2) training pedestrian detectors. One of our methods utilizes SVM for classification. We follow a similar structure to previous research using SVM for pedestrian classification [7, 3]. These authors have witnessed good performance using HOG as feature extraction and SVM as a classifier. The training pipeline is highlighted in Figure 3.6.

We use our background subtraction based method for generating positive samples and our random crop method for generating negative samples. These samples are resized to 128×64 resolution patches and HOG is used for feature extraction. Using 15×15 masks, this reduces the input image from a 8,192 dimension vector to 756 dimensions and shows more powerful feature representation.

3.6 Methods

The most common flow for object detection is two stages: proposal generation and classification. We utilize two tools for our proposal generation: Edgeboxes and pre-trained YOLOv3 detection results. Histogram of Oriented Gradients (HOG) is used for feature extraction. Support Vector Machine (SVM) is used for classification. Non-maximum suppression (NMS) is used for merging overlapping detection results. The overall testing pipeline for our SVM based methods is highlighted in Figure 3.7.

In this thesis, we study 4 machine learning methods for pedestrian detection. The methods are highlighted in Table 3.1. Initially, we hoped that the public Weizmann dataset [4] would be effective for learning, which provides a good range of human poses for actions such as walking, running, bending, waving, jumping, etc. They also provide different humans such as male and female. We believed that this would provide good training. As an alternative, we wanted to evaluate

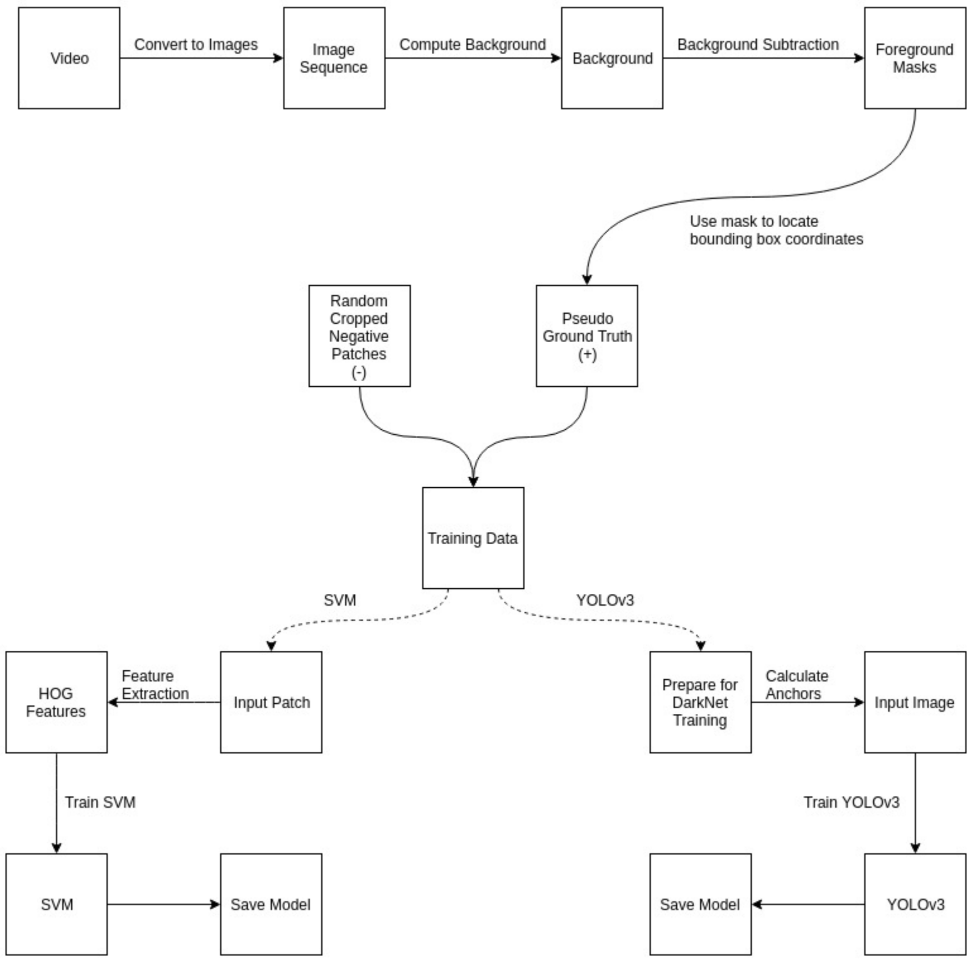

Figure 3.6: Training pipeline for pedestrian detection. It includes 1) pseudo ground truth generation and 2) training machine learning model SVM or training deep learning model YOLOv3.

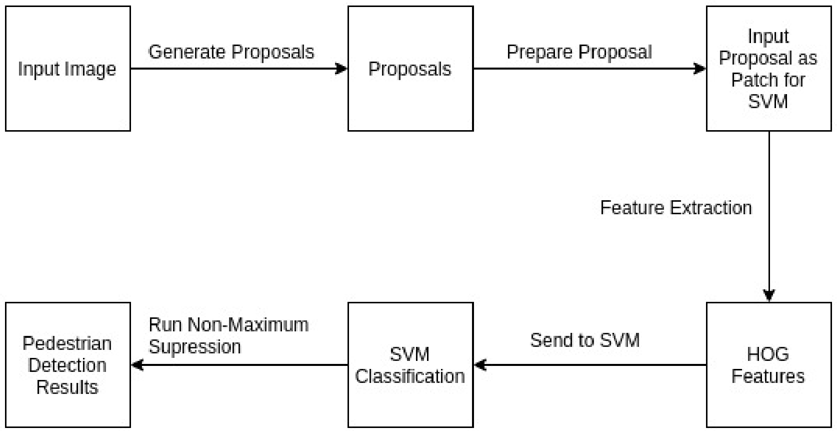

Figure 3.7: Testing pipeline for the machine learning based pedestrian detection methods.

Table 3.1: Machine Learning Based Pedestrian Detection.

Classifier	**Training Data**	**Proposal**
SVM	Pseudo ground truth of Weizmann dataset	Edgebox
SVM	Pseudo ground truth of Weizmann dataset	Pre-trained YOLOv3
SVM	Pseudo ground truth of Custom training dataset	Edgebox
SVM	Pseudo ground truth of Custom training dataset	Pre-trained YOLOv3

the performance for training on data that is closer to the domain of the testing set. To accomplish this, we train methods on our Custom dataset based pseudo ground truth as well.

A vital component in object detection systems is proposal generation. Edgebox has been proven to be effective in recent research, but we hope to see that a better proposal method can yield better results. The alternative method for generating proposals was the detection results from a pre-trained YOLOv3 human detector.

CHAPTER IV

DEEP LEARNING BASED PEDESTRIAN DETECTION

Deep learning has shown great success in many computer vision domains. Some systems developed with deep learning methods have surpassed human performance in many areas of image recognition tasks. Convolutional neural networks are able to process visual data at a fast rate and extract features effectively. We hope to demonstrate the capabilities of deep learning by training a state-of-the-art pedestrian detector with automatically generated pseudo ground truth by background subtraction.

4.1 YOLO: You Only Look Once Object Detector

You Only Look Once (YOLO) object detector was proposed in [31]. The one important limitation of existing deep learning based object detectors is the slow rate, which is unreliable for a real-time object detection system. The main motivation behind YOLO was to create a object detector that can be used for real-time systems [31]. They were able to create an object detector that not only achieved high accuracy, but could also process 45 frames per second.

Region proposal classification networks, such as Fast RCNN, work by running detection on various proposed regions [13, 12, 34]. This can be inefficient because this requires multiple predictions to be run on a single image. If there are 2000 region proposals, that image will require 2000 prediction results. YOLO treats the object detection as a single regression problem from pixels to bounding box locations and class confidences [31]. YOLO architecture predicts multiple bounding boxes and the associated class probabilities simultaneously in a single deep neural network, as shown in Figure 4.1. YOLO is fast because solving object detection as a regression problem eliminates the need of the proposal generation. Because YOLO takes the entire image as input,

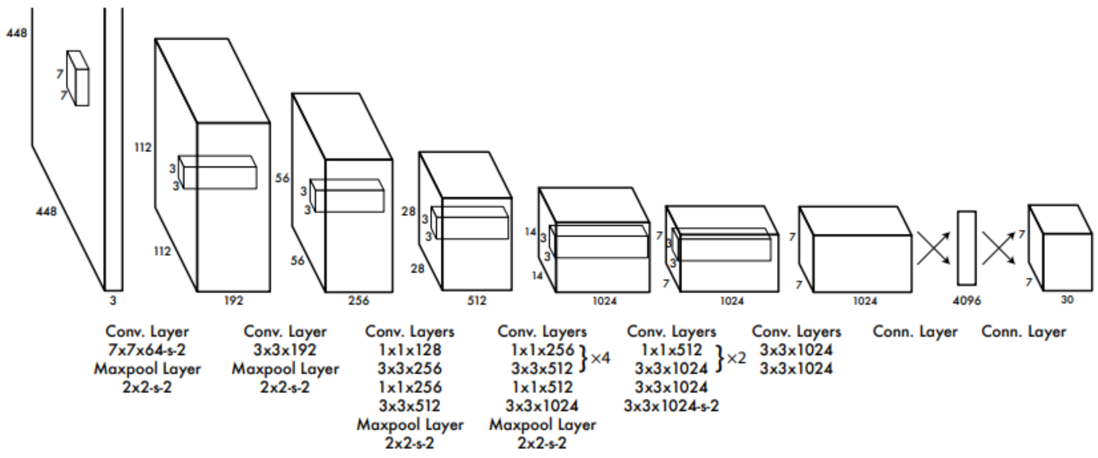

Figure 4.1: YOLO architecture (image taken from [31]).

it is able to encode contextual information when considering predictions. YOLO is able to learn generalized representation of objects and obtains good performance on some large-scale benchmarks [31].

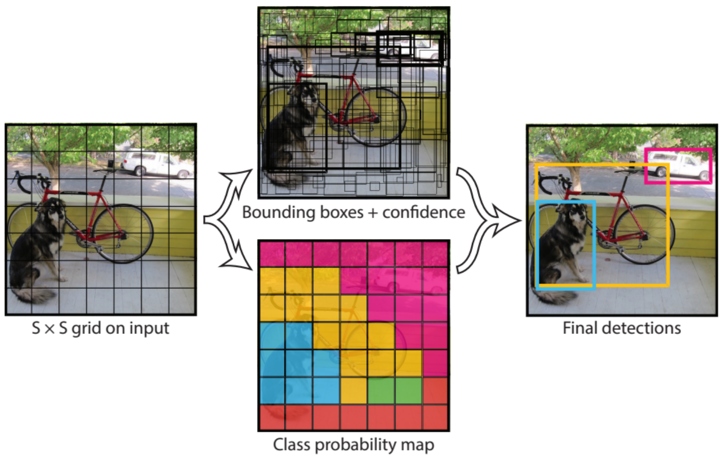

Figure 4.2: YOLO principle (image taken from [31]).

The secret to treating object detection as a regression problem is using anchor boxes and dividing the image into an $S \times S$ grid, as shown in Figure 4.2. Each cell in this grid will have B anchor boxes. Similar to the structure of FCN, the input is an $N \times N$ image and the output is $S \times S \times (B \times 5 + C)$, where B is the number of anchor boxes and C is the number of classes. By splitting the image into a grid, each cell will be responsible for using anchor boxes to locate and classify the objects.

4.2 YOLOv2

However, YOLO has higher localization errors and lower recall compared to other deep learning methods, like Fast R-CNN. YOLOv2 [32] was then made to be better and faster by boosting recall and localization without losing classification accuracy. YOLOv2 accomplished this by adding batch normalization on all convolutional layers in YOLO. In addition, they implemented additional fine tuning with higher resolution images. The authors removed the fully connected layers from YOLO so that they could use anchor boxes to predict bounding boxes [32]. This results in more than 1000 anchor boxes generated compared to the 98 anchor boxes in the original YOLO.

Another realization that YOLOv2 had was that a faster feature extractor correlates with faster prediction results. A common network used for feature extraction in object detection systems is VGG-16 [32]. YOLOv2 built their own feature extractor, DarkNet-19, following a similar structure to the VGG networks that performed better while requiring less FLOPs [32].

4.3 YOLOv3

YOLOv2's performance is continually improved in the YOLOv3, which used a more powerful feature extractor DarkNet-53 for better accuracy [33]. Another significant contribution of YOLOv3 was including prediction across scales. YOLOv3 uses a concept similar to feature pyramid networks [22] to predict boxes at varying scales [33]. The new YOLOv3 has higher accuracy with real-time detection speed.

4.4 Training and Testing for Deep Learning based Pedestrian Detection

The training for our deep learning based method follows the training pipeline shown in Figure 3.6. For the deep learning method, we use YOLOv3 as the tool for pedestrian detection. The pseudo ground truth method gives us two options: bounding box coordinates or patches. One difference in the YOLOv3 framework is that it takes full size images as inputs. Rather than using bounding box coordinates to make patches for training, YOLOv3 takes the entire image and the anchor-box coordinates. When preparing the training set for YOLOv3, we convert the pseudo ground truth bounding box coordinates to anchor-box coordinates.

The testing for our deep learning based method follows Figure 4.2, which predicts the bounding-box locations and class confidences. Based on the predicted bounding-box locations and class confidences, NMS is then used to eliminate the detections with high overlaps and low confidences.

The training and testing parameters and settings will be discussed in the experiment chapter.

CHAPTER V

EXPERIMENT

The experiments for this thesis were carried out on a computer equipped with a Ryzen 5 1600 CPU, 16GB of Memory, and an NVIDIA GeForce GTX 980 card with 4GB GPU Memory. In the following sections, we will introduce the programming environment, data, setting, evaluation and results.

5.1 Programming Environment

The computer used was running on the Ubuntu 18.04 operating system. Our pseudo ground truth framework was built using the image processing toolbox in MATLAB. The Edgebox, non-maximum suppression, and SVM code were based on MATLAB.

Our deep learning method was based on the YOLOv3 object detector. YOLOv3 was made available only through DarkNet, an independent neural network framework built by Joseph Redmon and his team using C and CUDA. We pulled the Darknet framework from their GitHub Repo [30]. We follow their instructions to run the Darknet.

5.2 Data

We used two datasets for training in this experiment. The first dataset is the public Weizmann dataset [11]. It is a dataset of video sequences of human actions. We believed that it provided good variability in human poses. It also considered both male and female human bodies. The only limitation of this dataset is that the image resolution is low. From the Weizmann dataset, we obtained 90 color videos recorded at 25 frames per second with 180×180 resolution. To test our framework on a more clear dataset, we collected a new dataset on the University of Texas Rio Grande Valley campus. These were recorded from GoPro cameras mounted on a vehicle. We picked sequences from 4 color videos (100 frames per video leading to 400 images in total) recorded at 30 frames per second at 1280×720 resolution. The 400 images are used for background subtraction to get the pseudo ground truth, which are further used to train SVMs and YOLOv3. To make this problem simple, only one pedestrian shows up in each training image.

We used one new collected dataset (500 random images in total) for testing in this experiment. This testing set was also collected in the University of Texas Rio Grande Valley campus, recorded from GoPro cameras mounted on a vehicle. Each image has a 1280×720 resolution. During testing, each image might have multiple pedestrians. We call the collected dataset by ourselves as Custom training dataset (4 videos, i.e., 400 images) and Custom testing dataset (500 images) in this thesis.

On the public Weizmann dataset, there were 7,130 human crops (pseudo ground truth) produced by background subtraction as positive samples. Then, 7,130 negative samples were taken from randomly selected crops of the Custom testing dataset. The Weizmann training set yielded a total of 14,260 samples of labeled data. Our custom training dataset generated 400 human crops (pseudo ground truth) via background subtraction. We then use histogram scaling and vertical flipping to augment the data to 5,600 human crops (pseudo ground truth). To maintain a balanced ratio, 5,600 negative samples were taken from randomly selected crops of the custom testing dataset. As a result, our Custom training set had a total of 11,200 samples of labeled data. The features from each sample crop is extracted via the histogram of oriented gradients mentioned above. Using 15×15 masks for the HOG, each crop produces a 756-dimension vector to train the SVMs. The SVMs trained are based on the linear kernel. The same training set will be used as the validation set for determining performance from training.

The YOLOv3 model was trained on the Custom training dataset with the associated pseudo ground truth by background subtraction.

5.3 Setting

Our machine learning (SVM) methods use HOG for feature extraction. We used 15×15 masks in HOG computation to reduce the input size for both training and testing. We use the linear-kernel SVM for both training and testing. When using Edgebox proposals for testing, the

SVM pipeline considered only the top 200 scored proposals. The NMS overlap threshold was set to 0.4.

The deep learning based method was achieved by fine-tuning the YOLOv3 object detector that was pre-trained on the COCO dataset [23], provided by [31]. We used the 400 images from our custom training set to fine-tune the pre-trained YOLOv3. During training, the batch size is set as 16 and the training epoch was set as 160. The learning rate was set to 0.00001.

5.4 Evaluation

One important consideration when making object detectors is evaluation. How do we evaluate the performance? The object detection research community has already come up with a standard for evaluation this task. The benchmark PASCAL VOC [10] has created test servers and evaluation metrics for object detection since 2005. Other datasets have been created for different domains of image recognition tasks, each with their own evaluation method such as COCO [21], but for the context of pedestrian detection, we only care about object detection evaluation.

For the purpose of our pedestrian detection, we adopted the PASCAL VOC 2012 evaluation metrics with the GitHub library [5]. This library is what we use to obtain our precision and recall values as well as the associated Precision-Recall Curve.

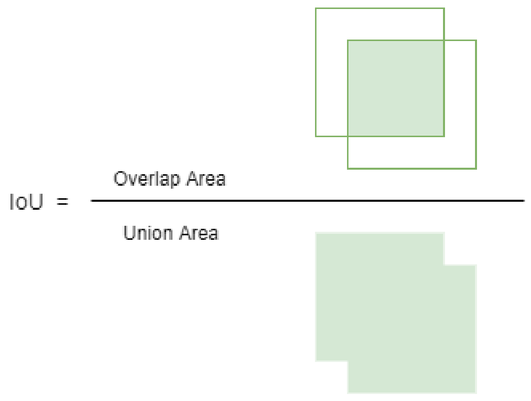

Figure 5.1: Intersection-over-Union to evaluate the object detection.

In object detection, we primarily care about 2 metrics: precision and recall. These metrics are determined based on *true* or *false* positives and negatives. There are two kinds of bounding boxes, the right answer and the detection result. Based on the overlap of these two boxes, we obtain the Intersection-over-Union as shown in Figure 5.1. If the threshold is over 50%, then that detection result is considered a true positive, otherwise a false positive.

Precision tells us *what percentage of detections were correct*? The formal definition of Precision is shown in Equation 5.1. Recall lets us know *what percentage of the objects were detected*? The formal definition of Recall is shown in Equation 5.2. An example of calculating both precision and recall for one image is shown in Figure 5.2.

$$Precision = \frac{tp}{tp + fp} \tag{5.1}$$

$$Recall = \frac{tp}{tp + fn} \tag{5.2}$$

With each predicted detection result, the precision and recall values are updated. Each detection result should also have a corresponding confidence percentage. To obtain the Precision-Recall curve, we *rank* our detection results in descending order by confidence values. As the library goes through the list of ranked detections, the precision and recall curves are changed and plotted, which leads to the Precision-Recall curve for the object detection performance.

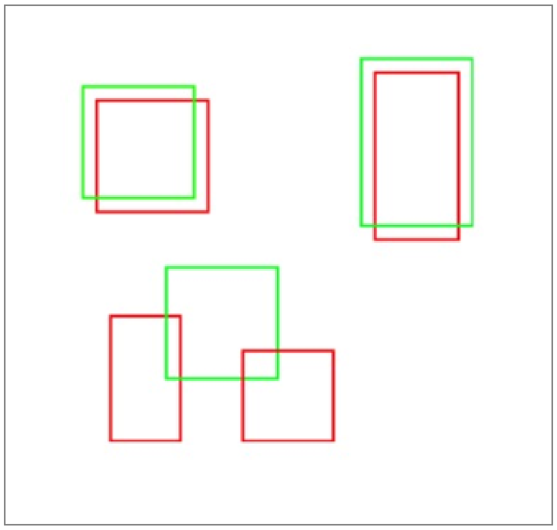

Figure 5.2: Example of evaluation using IoU > 50%. In this example, red boxes are the detections and green boxes are the ground truth answers (Precision: 2/4, Recall: 2/3 for this case).

The overall performance of the object detection is defined as Average Precision. This can be simply defined as the area underneath the Precision-Recall curve. If we end with 100% precision and recall, then our average precision would be 100%. If we were considering several classes of objects, we would compare our Mean Average Precision (mAP) with other methods. The mAP is the average of all the average precision values for each class. For our pedestrian detection methods, we only have 1 class, the pedestrian, so our mAP would be the same as our Average Precision.

5.5 Results

We evaluate the viability of the pseudo ground truth by training several methods. We provide the Precision-Recall curves and Average Precision values for each machine learning and deep learning method.

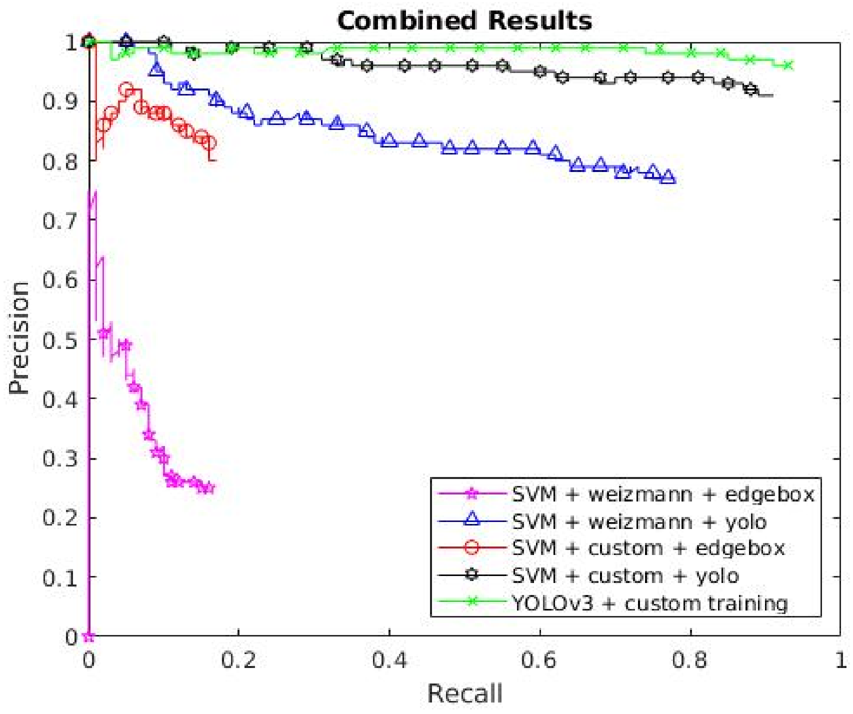

Figure 5.3: Precision-Recall Curves for all the methods on the Custom testing set.

Based on the results in Figure 5.3, the best performing method is the deep learning method with YOLOv3. Our SVM based method using the custom training dataset as training and a pre-trained YOLOv3 proposal performed second best. Our SVM based method using the Weizmann dataset as training and EdgeBox proposal performed the worst. The performance of our SVM methods were weak when using Edgebox as the proposals. With a better proposal like a pre-trained YOLOv3 or training on the custom training dataset, it can get better performance.

Table 5.1: Performance of predestrian detection on the custom testing set.

Method	**Average Precision**
SVM+Weizmann+Edgebox	6.34%
SVM+Weizmann+YOLOv3	66.99%
SVM+Custom+Edgebox	14.73%
SVM+Custom+YOLOv3	*87.85%*
YOLOv3+Custom	**91.53**%

From Table 5.1, we can see that the deep learning method YOLOv3 outperforms all machine learning methods. For our SVM based methods, we see that training on our custom training dataset yields better results due to the smaller domain gap with the custom testing dataset. Furthermore, using a more powerful proposal like a pre-trained YOLOv3 than the Edgebox proposal will generate better average precision.

5.6 Visualizations

The visualization example for pedestrian detection by different methods are displayed in Figures 5.4, 5.5, 5.6 and 5.7. They demonstrate the detection result for each method. The SVM method trained on the Weizmann based pseudo ground truth obtained low-accuracy detection. The errors were worse when using the Edgebox proposals. Performance was improved when the SVM methods were trained using the custom training set based pseudo ground truth. There was a larger improvement when the methods used stronger object proposals, such as the pre-trained YOLOv3 human detector. We also see that the deep learning based method, YOLOv3 trained on the custom training data, outperforms all other methods.

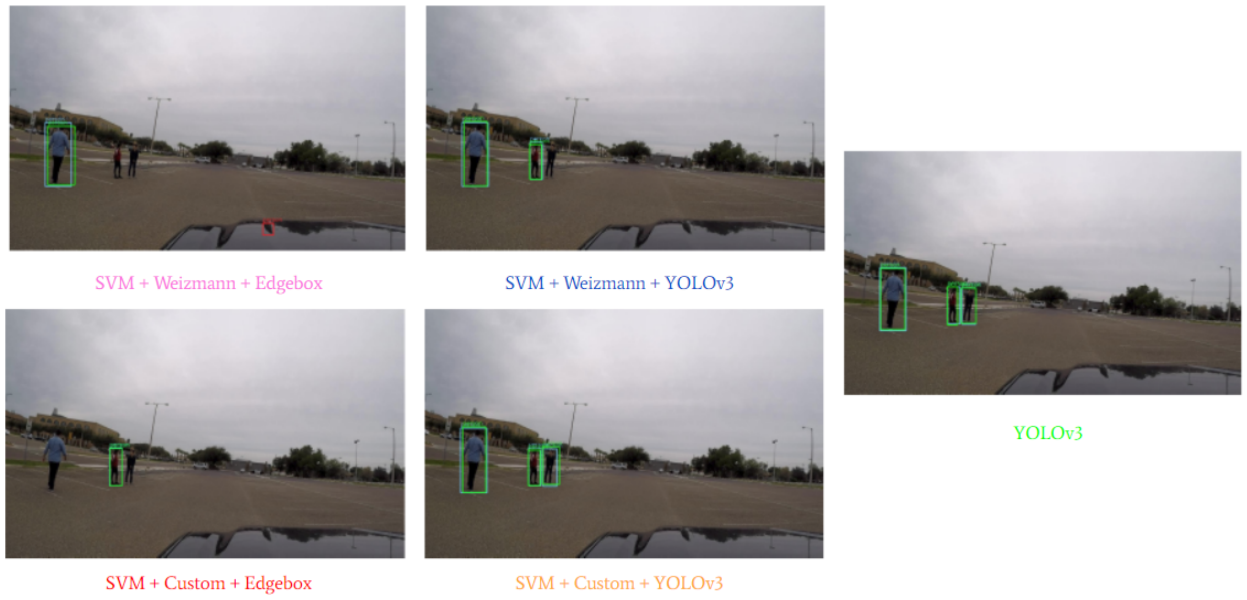

Figure 5.4: Visualization example 1 for pedestrian detection. In the visualization, blue boxes indicate the ground truth answers and green boxes are true positive detection results and red boxes are false positives.

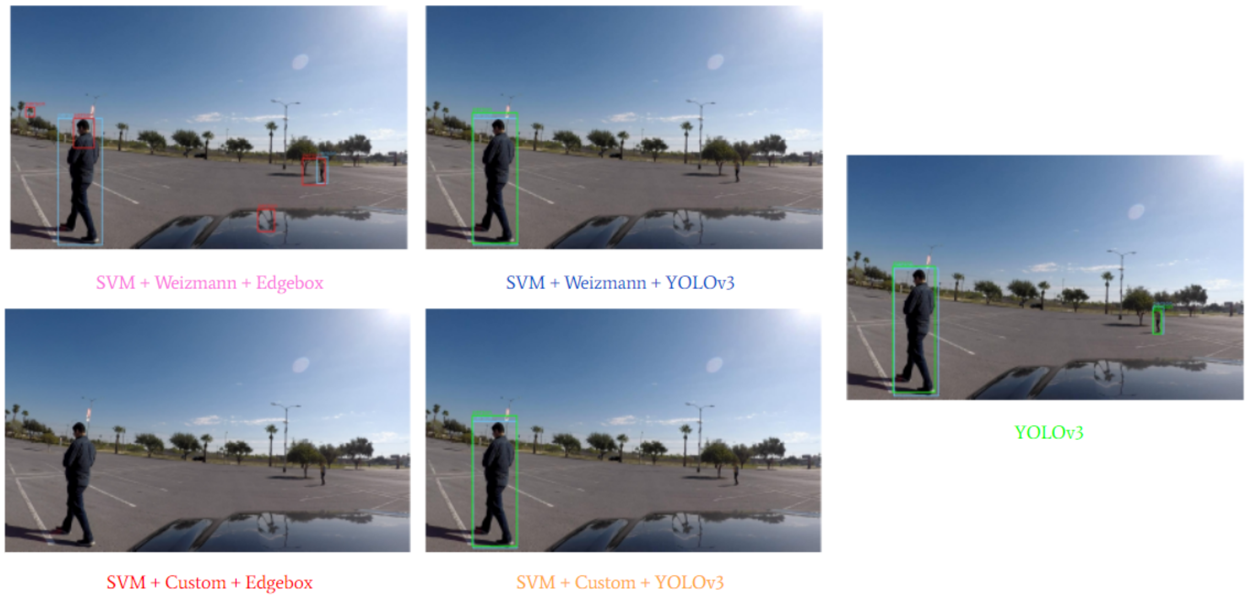

Figure 5.5: Visualization example 2 for pedestrian detection.

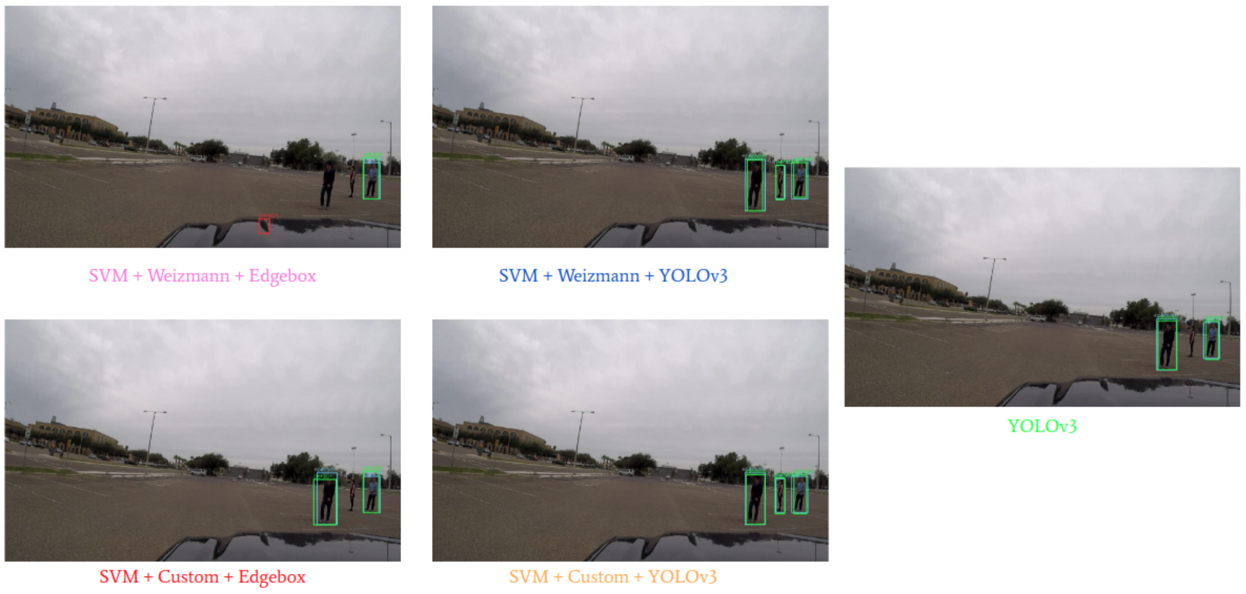

Figure 5.6: Visualization example 3 for pedestrian detection.

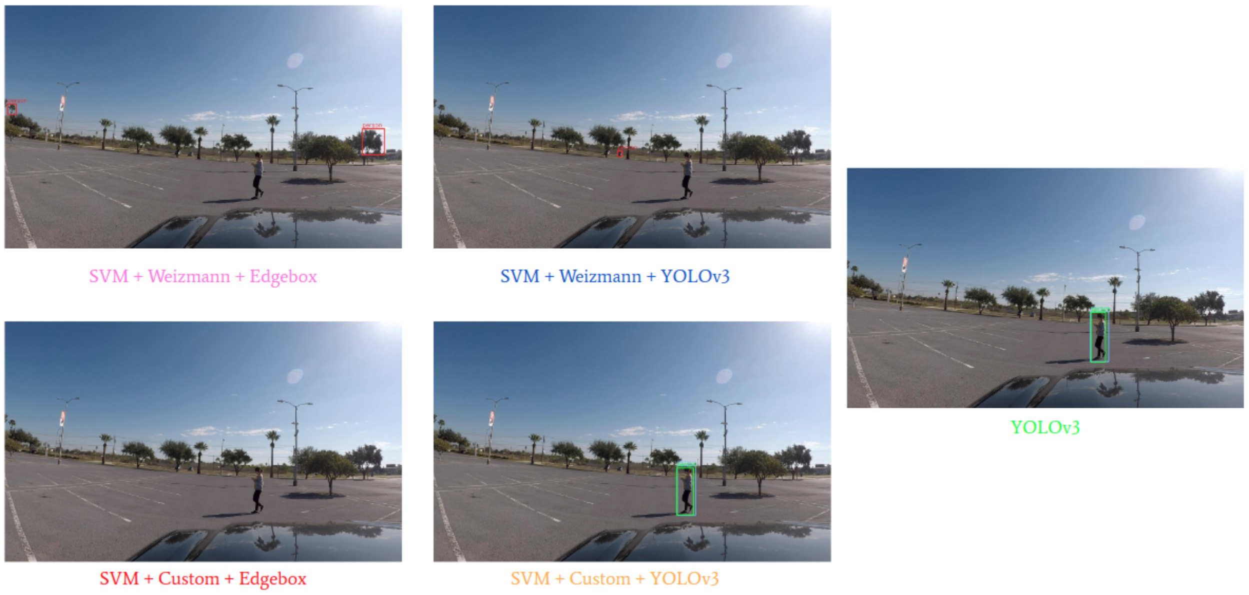

Figure 5.7: Visualization example 4 for pedestrian detection.

CHAPTER VI

SUMMARY

6.1 Conclusions

In this thesis, we show that it is possible to learn an effective and accurate pedestrian detector by watching the pedestrian-moving videos. We studied a method to automatically annotate images for training from the image sequence, i.e., background subtraction. We demonstrated the viability of this method of labeling by training different pedestrian detectors. We choose to test the pseudo ground truth with both traditional machine learning methods and a recent deep learning method.

Our findings are 4-fold. 1) It is possible to automatically annotate images using background subtraction. By watching pedestrian videos with the background subtraction method, we can obtain the pseudo ground truth to train a reliable pedestrian detector. 2) Better proposal method yields better detection results. 3) Deep learning methods are much stronger than traditional machine learning methods, because deep learning can extract more advanced deep neural network features than the HOG features and the deep learning method's classifier might be more discriminative than the SVMs. 4) Training on the custom training dataset yields better results than training on the Weizmann dataset due to lesser feature distribution discrepancy between the training and testing datasets.

One limitation of the proposed framework is that the background subtraction method only performs well on video sequences recorded on a static camera. If the video is taken by a moving camera or the background is frequently changed or if there were many different classes of moving objects in the image, the background subtraction method might not obtain high-quality pseudo ground truth. In these complex cases, some robust foreground extraction algorithms are required.

6.2 Future Work

Most of the efforts of the thesis was focused on creating the pseudo ground truth framework. Initially, we wanted to evaluate to the pseudo ground truth on only SVMs. We later decided to try deep learning methods like YOLOv3.

The custom training dataset currently contains 4 videos (400 images). The YOLOv3 model was trained on these 400 images. We hope to use more videos to generate more pseudo ground truth for training. Although the deep learning method performed the best, we anticipated higher performance. In the future, we will use a large number of videos to generate labels and train a more robust pedestrian detector.

Pedestrian detection is a hot topic that has been always studied in computer vision. There are benchmark datasets for evaluating pedestrian detectors, such as the CalTech Pedestrian Dataset [9]. We hope to see how our pseudo ground truth trained methods compares with other state-of-the-art methods. We also hope to test the trained pedestrian detector on a general dataset, like the PASCAL VOC [10].

BIBLIOGRAPHY

[1] P. Banerjee and S. Sengupta, “Human motion detection and tracking for video surveillance,” in *Proceedings of the National Conference of Tracking and Video Surveillance Activity Analysis*, 2008, pp. 88–92.

[2] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, “Pedestrian detection at 100 frames per second,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2903–2910.

[3] M. Bertozzi, A. Broggi, M. Del Rose, M. Felisa, A. Rakotomamonjy, and F. Suard, “A pedestrian detector using histograms of oriented gradients and a support vector machine classifier,” in *IEEE Intelligent Transportation Systems Conference*. IEEE, 2007, pp. 143–148.

[4] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, “Actions as space-time shapes,” in *IEEE International Conference on Computer Vision*, vol. 2. IEEE, 2005, pp. 1395–1402.

[5] Cartucho, *mAP (mean Average Precision)*, Accessed October 13, 2019. [Online]. Available: https://github.com/Cartucho/mAP

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

[7] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2005, pp. 886–893 vol. 1.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255.

[9] P. Dollar, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the art,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2011.

[10] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[11] D. M. Gavrila, “Pedestrian detection from a moving vehicle,” in *European Conference on Computer Vision*. Springer, 2000, pp. 37–49.

[12] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[15] J. Hosang, R. Benenson, and B. Schiele, “A convnet for non-maximum suppression,” in *German Conference on Pattern Recognition*. Springer, 2016, pp. 192–204.

[16] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[17] D. H. Hubel and T. N. Wiesel, *Brain and visual perception: the story of a 25-year collaboration*. Oxford University Press, 2004.

[18] S. Hwang, J. Park, N. Kim, Y. Choi, and I. So Kweon, “Multispectral pedestrian detection: Benchmark dataset and baseline,” in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2015.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[20] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] X. Liang, Y. Wei, L. Lin, Y. Chen, X. Shen, J. Yang, and S. Yan, “Learning to segment human by watching youtube,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1462–1468, 2016.

[22] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[24] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[25] P. Luo, Y. Tian, X. Wang, and X. Tang, "Switchable deep network for pedestrian detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014.

[26] S. Munder and D. M. Gavrila, "An experimental study on pedestrian classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1863–1868, 2006.

[27] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *International Conference on Pattern Recognition*, vol. 3. IEEE, 2006, pp. 850–855.

[28] W. Niu, J. Long, D. Han, and Y.-F. Wang, "Human activity detection and recognition for video surveillance," in *2004 IEEE International Conference on Multimedia and Expo*, vol. 1. IEEE, 2004, pp. 719–722.

[29] M. Piccardi, "Background subtraction techniques: a review," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 4. IEEE, 2004, pp. 3099–3104.

[30] Pjreddie, "Darknet," Accessed October 2, 2019. [Online]. Available: https://github.com/pjreddie/darknet

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[32] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.

[33] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[34] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[35] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. Lecun, "Pedestrian detection with unsupervised multi-stage feature learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2013.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[37] A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," *Computer Vision and Image Understanding*, vol. 122, pp. 4–21, 2014.

[38] Wikipedia, "Support-vector machine — Wikipedia, the free encyclopedia," https://en.wikipedia.org/wiki/Support-vector_machine, 2019, [Accessed 03-October-2019].

[39] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele, "Towards reaching human performance in pedestrian detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 973–986, 2017.

[40] S. Zhang, R. Benenson, and B. Schiele, "Citypersons: A diverse dataset for pedestrian detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[41] R. Zhao, W. Ouyang, and X. Wang, "Unsupervised salience learning for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3586–3593.

[42] Y. Zhou, H. Yu, and S. Wang, "Feature sampling strategies for action recognition," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3968–3972.

[43] G. Zhu, F. Porikli, and H. Li, "Tracking randomly moving objects on edge box proposals," *arXiv preprint arXiv:1507.08085*, 2015.

[44] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European Conference on Computer Vision*. Springer, 2014, pp. 391–405.

BIOGRAPHICAL SKETCH

Andrew Yiru Chen completed his undergraduate degree in a Bachelor's of Science in Chemistry with a minor in Computer Science at the University of Texas at Austin in 2017. He has then moved on to the University of Texas Rio Grande Valley where he obtained his Master's of Science in Computer Science in December of 2019. In addition to holding positions as a graduate assistant each semester, he was given the opportunity to intern at ExxonMobil as a software engineer over the Summer of 2019. After the graduate program, he will be returning to industry to work full-time. His email is aychen.c@gmail.com.