Vison Based - Pay As You Throw System

Motion Detection for Garbage and Recycle - Based on Faster RCNN

Songyuan Ji, Minglun Gong
MEMORIAL UNIVERSITY
February 2019

2016-17 HARRIS CENTRE - MMSB WASTE MANAGEMENT APPLIED RESEARCH FUND

THE HARRIS CENTRE Memorial University

Motion Detection for Garbage and Recycle Based on Faster RCNN

by

© *Songyuan Ji*

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of *Science*

Department of *Computer Science*

Memorial University of Newfoundland

Feberay, 2019

St. John's Newfoundland

Abstract

This project implemented the motion detection and tracking, the goal of project is detection , tracking and counting the different types garbage. The Faster Region Convlutional Neural Network (Faster RCNN or FRCNN) is used as classifier and detector. Because the count for garbage or recycle is the final result, therefore, the tracker is developed based Kalman filter.

Acknowledgements

I would like to express my sincere thanks to my supervisors, Dr. Minglun Gong and Mr. Kevin Duggan for providing me with the research project, Deep Learning for Motion Detection. My detailed discussions with them and their encouragement of innovative ideas and critical thinking were critical for guiding me to be an independent researcher.

I would like to acknowledge the financial supports from the Computer Vision lab, Department of Computer Science, School of Medicine and School of Graduate Study.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, the deep learning model has gradually replaced the traditional machine vision method and become the mainstream algorithm in the object detection field. How to parse the information that can be understood by the computer from the image is the central problem of machine vision. The deep learning model, due to its powerful representation ability, combined with the accumulation of data and the advancement of computing power, has become a hot research direction of machine vision [6]. To understand a picture, according to the needs of the following tasks, there are three primary levels, as showing in Figure 1.1:

One is classification and it is shown in Figure 1.1(a), which is to structure an image into a specific category of information and describe the picture with a predetermined string or instance ID. This task is the simplest and most basic image understanding task, and it is also the first task of the deep learning model to achieve the breakthrough and achieve large-scale application [6]. Among them, ImageNet is the most authoritative evaluation set, and the annual Large Scale Visual Recognition

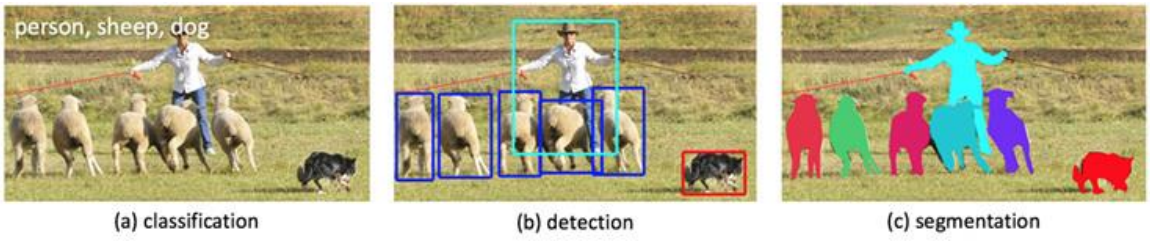

Figure 1.1: The research areas for understand a image

Challenge (ILSVRC) has spawned a large number of excellent deep network structures, providing a basis for other tasks [6]. In the field of application, face and scene recognition can be classified as classification tasks.

The second is detection, it is shown in Figure 1.1(b). The classification task cares about the whole, gives the content description of the whole picture, and the detection focuses on the specific object, and requires the category information and position information of the object to be obtained at the same time. Compared to classification, the test gives an understanding of the foreground and background of the image [6]. We need to separate the object of interest from the background and determine the description (category and location) of the object. Therefore, the output of the detection model is a List, each item of the list uses a data set to give the category and location of the detected object (the coordinate representation of the commonly used rectangle detection box) [6] [9].

The third is segmentation which is shown in Figure 1.1(c). Segmentation includes semantic segmentation and instance segmentation. The former is an extension of the pre-background separation, which requires separation of image parts with different

semantics, while the latter is an extension of the detection task, which requires describing the outline of the object (More detailed than the detection frame). Segmentation is a pixel-level description of an image that gives each pixel category (instance) meaning and is useful for understanding more demanding scenes, such as road and non-road segmentation in driver-less [6].

The area of interest in this project is object detection, the middle level of image understanding. The object detection has many uses, including:

Object Detection Since the mid-2000s, some point-and-shoot cameras have begun to detect facial features for more efficient autofocus. Although it is a narrower type of object detection, the methods used to apply to other types of objects, which we will describe later [6] [9].

Counting A simple but often overlooked use of object detection is counting. The ability to calculate people, cars, flowers, and even microbes is a real-world need for a wide variety of systems that use images. Recently, with the advent of video surveillance devices, the opportunity to use computer vision to turn raw information into structured data is more excellent than ever [6] [9].

Visual search engine Finally, one use case we like is Pinterest's visual search engine. They use object detection as part of the pipeline to index different parts of the image. This way, when searching for a specific wallet, you can find an instance of a portfolio similar to the one you want in a different context. This is much more powerful than just finding similar images, just like Google Image's reverse search engine [6] [9].

Aerial image analysis In the era of cheap drones and (nearly) affordable satellite launches, our world has never had so much data. Companies have used satellite

imagery from companies such as Planet and Descartes Labs to apply object detection to calculate cars, trees, and boats. These application has led to high-quality data, which is impossible (or very expensive) and has now reached a wider audience [6] [9].

The project we finished in this paper: In civil engineering, workers collect garbage every day into the car, place a camera at the entrance of the garbage, and record the daily collection work. And our application is to process these video videos, object, detect and count the garbage in the video. In this project, we will use the deep learning algorithm to classify and identify the garbage bags in the video and then classify and mark each garbage bag. Finally, count the statistics of each type of recognized garbage bags. We will use the Faster RCNN model as the deep learning training models. The operating system is Windows 10, the Google's Tensorflow as a deep learning platform, the Anaconda as the science computation platform, and the development language is Python 3.6, OpenCV, cuDNN and CUDA.

Chapter 2

Background and related work

2.1 Object Detection and tracking

In the field of computer vision, "object detection" mainly solves two problems: where (position), and what (category) multiple objects on the image [6] [9]. Around this issue, people generally divide their development process into three stages:

In present, the classic object detection include three types, they are:

1. Traditional object detection method

    - Object Detection using Haar feature-based cascade classifiers [4].
    - Detection with histogram of oriented gradients (HOG), and support vector machines (SVM).
    - Deformable Parts Model (DPM) and many of the above improvements and optimization algorithms.

2. Candidate window + depth learning classification.

- object detection framework based on R-CNN combined with region proposal and CNN classification (R-CNN, SPP-NET, Fast R-CNN, Faster R-CNN, R-FCN) [4].

3. Regression method based on deep learning.

   - An end-to-end object detection framework (YOLO, SSD) that converts object detection into a regression problem, represented by YOLO [4].
   - The recently emerging uses Recurrent Rolling Convolution (RRC) detection combined with the Recurrent neural network (RNN) algorithm [4];
   - Deformable CNN combined with Deformable part models (DPMs).

The general structure of the object detection process is as follows Figure 2.1:

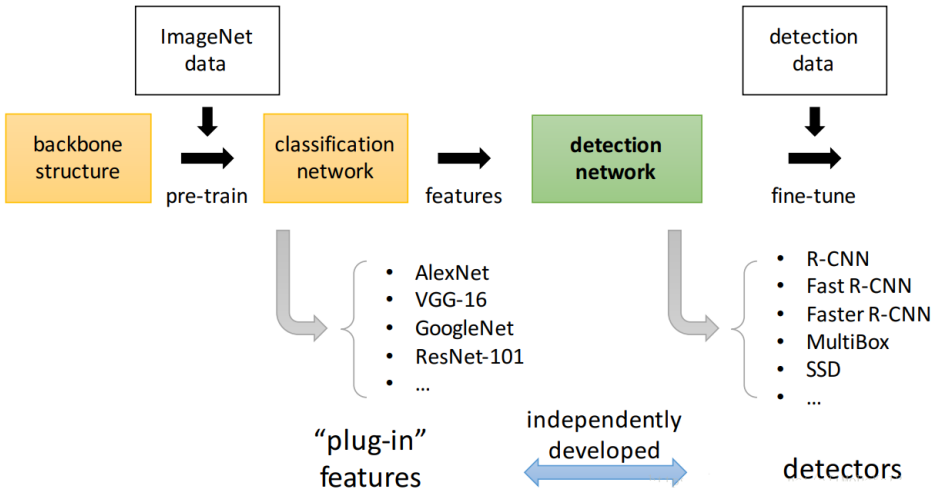

Figure 2.1: Processing of Detection based on deep learning models

The object detection process includes: first, inputting an image to be processed

for pre-training to obtain a feature. Second, the detector detects or tracks the target file (video, picture). Finally, the location information and classification information of the object are obtained and identified in the file. There is a significant problem with traditional object detection: one is that the area selection strategy based on the sliding window is not objected, the time complexity is high, and the window is redundant. Due to the high time complexity, the processing time of object detection is very long, and the practical speed standard cannot be achieved [6]. As the deep learning model is introduced into the object detection, the region proposal provides an excellent solution to the problem of the sliding window. The region proposal uses the texture, edge, color, and other information in the image to pre-discover the possible positions of the object in the image, which can ensure a high recall rate when fewer windows (thousands or even hundreds) are selected [9]. This dramatically reduces the time complexity of subsequent operations and the candidate window obtained is of higher quality than the sliding window. In many deep learning models, FRCNN uses region proposal, which makes FRCNN achieve good results when dealing with object detection. So we use FRCNN to achieve object classification and recognition.

2.2 Dataset for object detection

For a deep learning project, the quantity and quality of the training set are very important. For research on object detection, the classic deep learning libraries include Coco, Pascal VOC, and JPRGImages [2].

This project is to detect, classify and count objects in motion. Moving object detection processes every frame (picture) in the video. The object to be detected

based on this item is a garbage bag but is not included in the existing database. Therefore, we use a special image segmentation tool to process the provided video to generate a new data training set. We will use the PascalVOC dataset in this project.

The specific steps for generating the data set are as follows:

1. Download the samples from website or other public source, obtain the frame image from the video that were supported from customer.

2. Use the special tool software labelImg to process each image manually, determine the object location through the bounding box, and label the category. LabelImg is a graphical image annotation tool. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet [3]. The label data after labeling is as such as showing in Figure 2.2:

   Labeling the recycle bag sample is such as showing in Figure 2.3:

   Because of the real world, the most scene include the mixed object in same picture, labeling the mixed objects is such as showing in Figure 2.4:

3. Select some representative images as training and test sets.

4. Execute the python program to count the training set and test set to produce a .xml index file that can be read by the program.

This dataset objects the scene understanding, which is mainly intercepted from complex daily scenes. The objects in the image are calibrated by accurate detection. The image includes 2 categories of object, 10,000 images, and more than 40,000 labels.

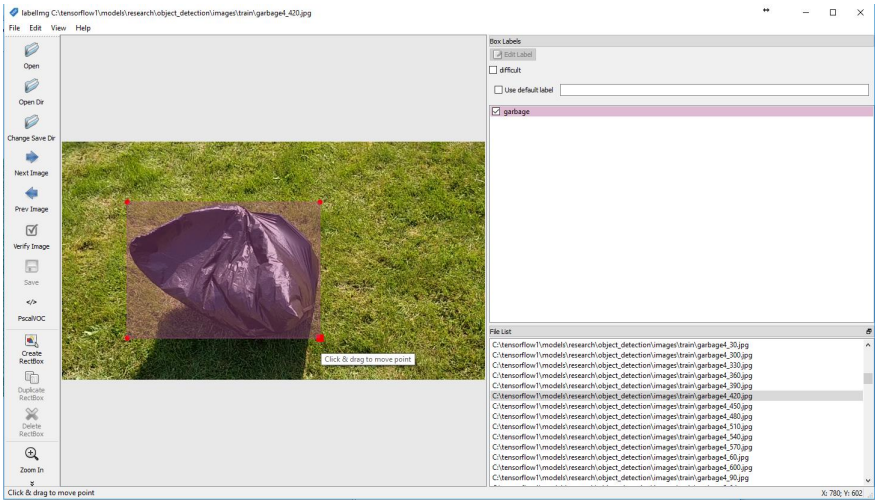

Figure 2.2: Mark for the single object of garbage example: Using LabelImg to locate and label object, and select the data set type that is based on requirement

The dataset mainly solves three problems: object detection, context between objects, and precise positioning in 2 dimensions of the object.

2.3 Faster RCNN structure

Ross B. Girshick proposed a new Faster RCNN in 2016. Structurally, Faster RCNN has integrated feature extraction, proposal extraction, bounding framework regression, and classification into a network. The comprehensive performance has been dramatically improved, especially regarding detection speed [7].

According to the author, as shown in Figure 2.5, Faster RCNN can be divided into four main contents:

1. Conv layers. As a CNN network object detection method, Faster RCNN first

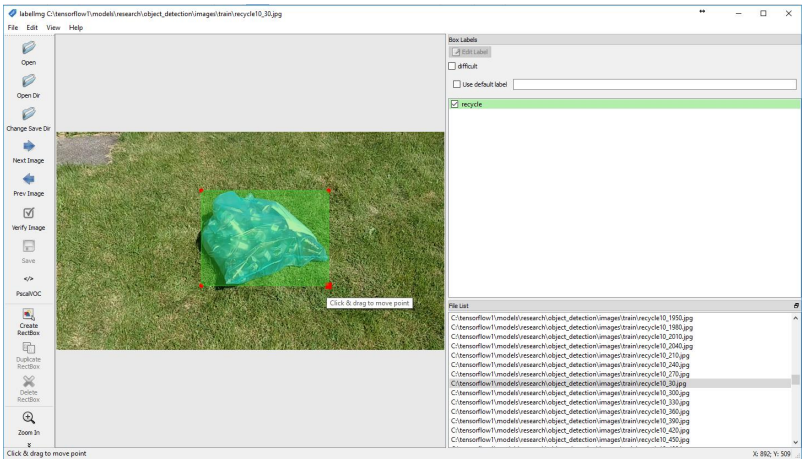

Figure 2.3: Mark for the single object of recycle example: Using LabelImg to locate and label object, and select the data set type that is based on requirement

uses a set of basic conv+relu+pooling layers to extract image feature maps. The feature maps are shared for subsequent RPN layers and fully connected layers [7].

2. Region Proposal Networks. The RPN network is used to generate region proposals. This layer uses softmax to determine that the anchors belong to the foreground or background, and then use the bounding box regression to correct the anchors to obtain accurate proposals [7].

3. Roi Pooling. This layer collects the input feature maps and proposals. After synthesizing the information, it extracts the feature maps and sends them to the subsequent fully connected layer to determine the object category [7].

4. Classification. Use the pseudo feature maps to calculate the category of the proposal, and at the same time bounding the box to obtain the final precise

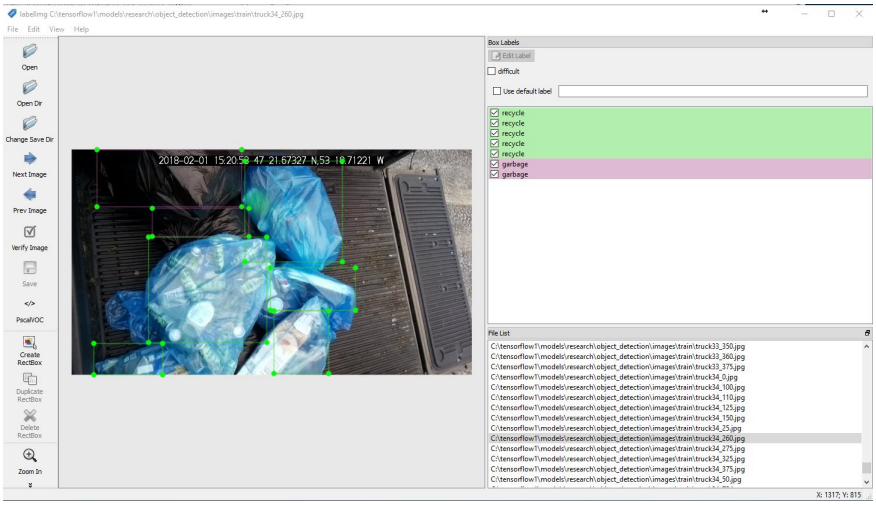

Figure 2.4: Mark for the various object of mixed example: Using LabelImg to locate and label object, and select the data set type that is based on requirement

position of the detection frame [1].

The ultimate goal of this project is to classify and count the collected garbage bags. This project requires the use of deep learning modules that require not only a higher accuracy rate but also a faster processing speed. It is precise because Faster RCNN has two modules, region proposal, and bounding box, which enables Faster RCNN to meet project requirements in handling the speed and accuracy of motion detection [7]. Another important reason that we use Faster RCNN is designing a Region Proposal Network ahead RCNN, this idea make all of program will run based GPU, this structure make the running speed will be high enough for processing video and real time from camera.

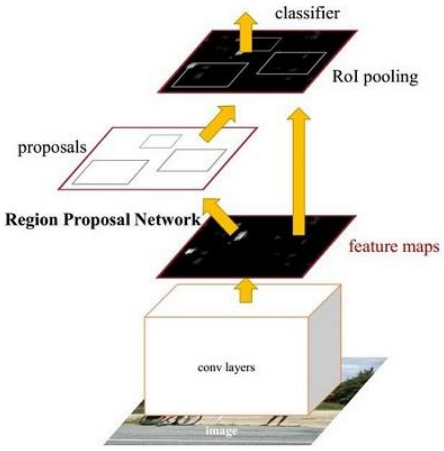

Figure 2.5: Basic processing of Faster RCNN and struture

2.4 Kalman filter

The Kalman filter is named after Rudolph E. Kalman. As early as 1960. When Kalman visited NASA, he found that the filtering method can solve the problem of orbit prediction well [8]. The Kalman filter is different from the frequency filter, which is an iterative filter with time information. Kalman's model equations: measurement equations and state equations [8].

The state equation is:

$$x_k = A_{x_{k-1}} + v_k$$

x_k is the state quantity at time k, and v_k is the noise information at time k. The iterative relationship between the state quantities represented by the state formula. The state quantity at time k is only related to the previous moment. It is irrelevant to any moment before [8].

Measuring equation:

$$z_k = H_{x_k} + w_k$$

z is the measurement information, which can be understood as the information we can obtain, such as the measurement information of the sensor [8].

Kalman filter solution process: After talking about the measurement equation and the state equation, let's talk about the solution process and ideas of Kalman. The filter is divided into two methods, prediction and update [8] [3].

1. Status prediction:

$$x_{\frac{k}{k-1}} = A_{x_{k_1}}$$

$$P_{\frac{k}{k-1}} = AP_{k-1}A^T + Q$$

The state of the previous moment is known to predict the state of the current moment, and no observation is involved. P is the covariance of x, indicates an inaccuracy. Q is the v variance matrix.

2. Time update: During the forecasting process. The predicted value is obtained using the model, but its value is not the exact value of the state value. Because of the accuracy of the modeling and the impact of noise, the inaccuracy of the model is caused, and the value can only update it we observe [3].

(1)

$$x_k = x_{\frac{k}{k-1}} + k(z_k - H_{x_{\frac{k}{k-1}}})$$

(2)

$$k = P_{\frac{k}{k-1}}H^T(HP_{\frac{k}{k-1}}H^T + R)^{-1}$$

(3)

$$P_k = (I - kH)P_{\frac{k}{k-1}}$$

Time update of equation (1): Use the predicted value of the prediction process to predict the measured value: $H_{x_{\frac{k}{k-1}}}$. By measuring the obtained value to subtract the expected measurement, the deviation between the expected value and the actual estimated value can be known. Then use a compensation factor k to compensate for the final estimated state value x_k [3].

Kalman filters have many uses, including control, navigation, computer vision, and time series econometrics [8] [3]. This example explains how to use the Kalman filter for object tracking and is committed to three important features:

- Prediction of the future position of the object; solves the problem that objects appearing during object detection and tracking are occluded and ignored by the system.
- Reduce the noise introduced by inaccurate detection; reduce errors in detecting incorrect objects.
- The process of facilitating the association of multiple objects with their trajectories. Can track multiple objects simultaneously

The tracking process based on Kalman filter is showing in Figure 2.6: During the tracking and detection, the system predicts the motion track follow the rules, and decide whether to label if confirm to the rules [8] [3].

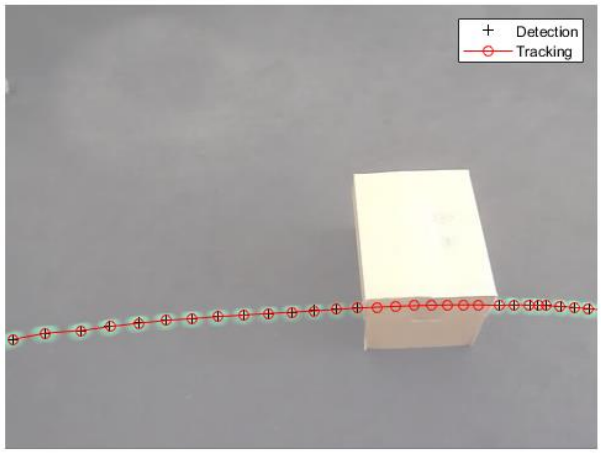

Figure 2.6: Prediction processing example of motion object based on Kalman filter

Chapter 3

Methods

Depend on the goal of this project, we will designed The flow chart for this project is showing in Figure 3.1:

3.1 Implementation for Faster RCNN

We will use FRCNN as classifier and detector, the Faster RCNN model was implemented on Tensorflow that is deep learning platform in Google [1].

3.1.1 FRCNN processing and various layers structure

The network structure based on the Faster RCNN detection is showing in Figure 3.2, which can see the image of the network for an arbitrary size $P \times Q$. The first zoom to a fixed size$M \times N$, then send the $M \times N$ image to the network; Contains 13 conv layers, 13 relu layers and 4 pooling layers in Conv layers; The RPN network first undergoes a 3×3 convolution, and then generates foreground anchors and bounding box regression

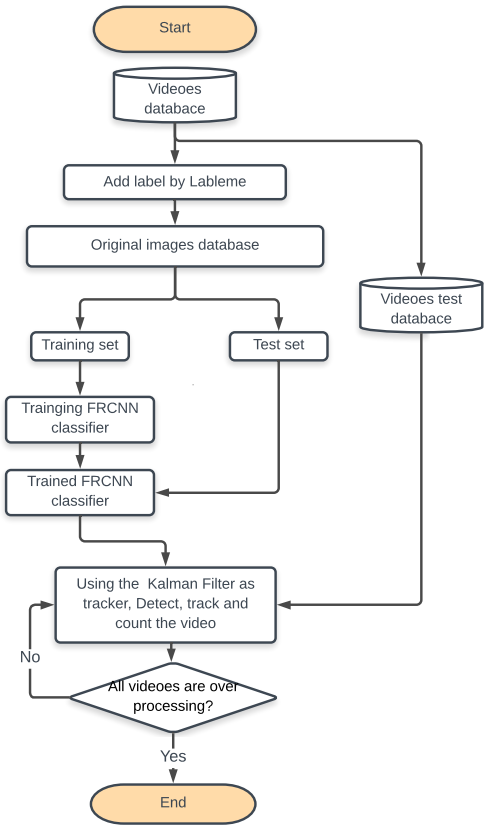

Figure 3.1: Processing of Detection, Tracking and counting for garbage and recycle

offsets, respectively, and then calculates the proposals; The Roi Pooling layer uses the proposal to extract the proposal feature from the feature maps and send it to the subsequent full-connection and softmax network for classification (that is, what the object is classified).

The classic detection method generates a detection frame that is very time-consuming. For example, OpenCV AdaBoost uses a sliding window + image pyramid to generate a detection frame; or RCNN uses the SS (Selective Search) method to generate a detection frame [7] [1]. The Faster RCNN discards the traditional sliding window and SS method and directly uses the RPN to generate the detection frame.

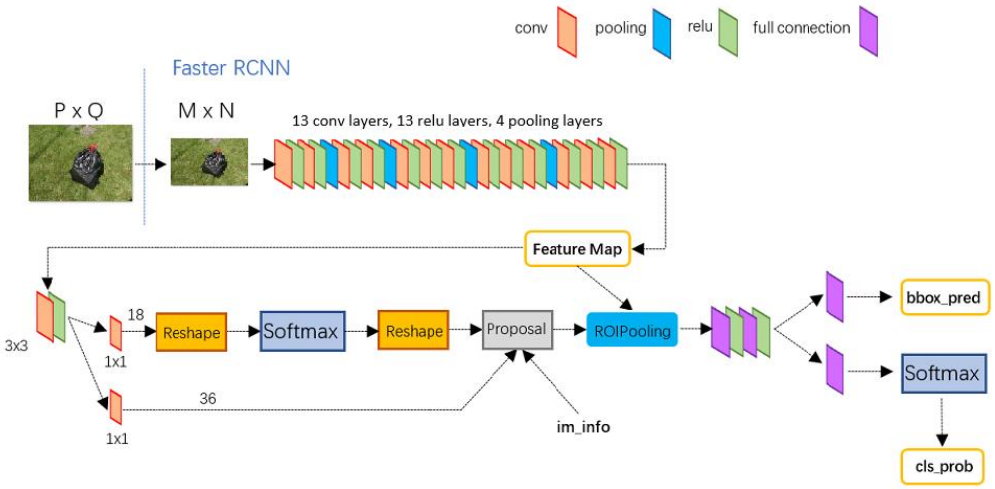

Figure 3.2: The network structure based on the Faster RCNN

This method is also a considerable advantage of the Faster RCNN, which can significantly improve the detection frame generation speed[7]. There are some important part of FRCNN includes:

1. RPN network

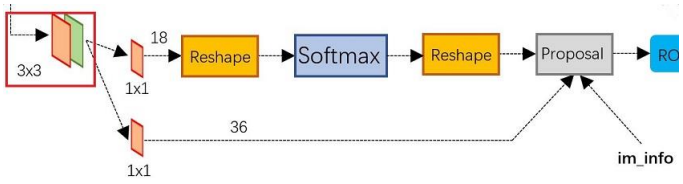

Figure 3.3: Region Proposal network processing and structure

such as Figure 3.3 above shows the specific structure of the RPN network. It can be seen that the RPN network is divided into two lines. The above one gets the foreground and background through the softmax classification anchors (the detection target is the foreground), The following one is used to calculate

the bounding of the anchoring of the anchors to obtain the accurate proposal. The final Proposal layer is responsible for synthesizing foreground anchors and bounding box regression offset acquisitions. Also, remove proposals that are too small and out of bounds. The entire network to the Proposal Layer here, complete the equivalent of the target positioning function[7].

2. Anchors When it comes to the RPN network, the Anchor has to be mentioned. The anchors is a set of rectangles generated by RPN/generate-anchors.py. The multi-scale method commonly used in detection is introduced through anchors[7]. As shown in Figure 3.4:

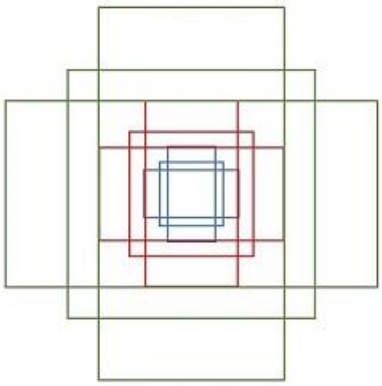

Figure 3.4: The Anchors: the proposal location box of object

The above anchors' size is set based on the detected image. In this project we will use the frame image generated from the original video, the size is 1280 × 720. As shown in Figure 3.5: The feature maps obtained by the Conv layers calculation are traversed, and the nine anchors are equipped for each point as the initial detection frame. In doing so, the detection frame is very inaccurate.

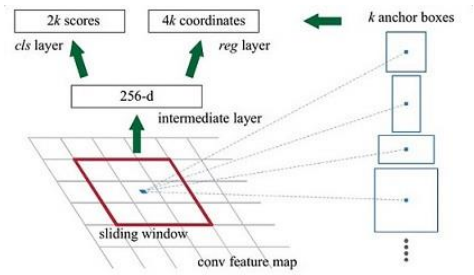

Figure 3.5: The Anchors works structure

Don't worry, and there are two bounding box regressions to correct the position of the detection frame[7].

Explain the numbers in the above picture.

- Because the independent data set will be used in this project, this data set is small. And the small samples are easy to be over-fitting, the Visualizing and Understanding Convolutional Networks(ZF-net) model will be preferred. The last conv5 layer in Conv Layers is num-output= 256, which corresponds to generate 256 feature maps, so the equivalent of a feature map is 256-d.
- After conv5, do rpn-conv 3×3 convolution and num-output = 256, which is equivalent to each point merging the surrounding 3×3 spatial information, while 256-d is unchanged (as shown in the red box in Figures 3.4 and 3.5)
- Suppose there are k anchors at each point in the conv5 feature map (default k=9), and each anchor is divided into foreground and background, so each point is converted from 256-d feature to $cls = 2k$ scores; Anchor has [x, y, w, h] corresponding to 4 offsets, so reg=4k coordinates

- Also, all anchors take too much training, and the training program will select 256 suitable anchors for training.

3. Bounding Box Regression

Introduce the mathematical model and principle of bounding box regression. As shown in Figure 3.6: , the green frame is the Ground Truth (GT) of the

Figure 3.6: Calibrite the bounding box location of object

garbage bag, and the red is the extracted foreground anchors. Even if the red frame is recognized as a garbage bag by the classifier, the picture is equivalent because the red frame is not positioned. The garbage bag was not detected correctly. So we want to use a method to fine-tune the red box so that the foreground anchors and GT are closer[7].

For windows, the four-dimensional vector (x, y, w, h) is generally used to represent the center point coordinates and width and height of the window. such as showing in Figure 3.7: , the red box A represents the original Foreground Anchors, the green box G represents the target GT, and our goal is to find a relationship that causes the input original anchor A to be mapped to get a regression

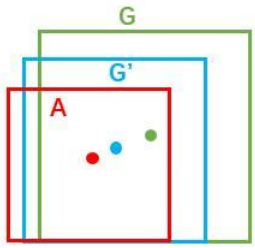

Figure 3.7: Campare the real bounding box with GT

closer to the real window G. Window G', ie given $A = (A_x, A_y, A_w, A_h)$, look for a map f such that $f(A_x, A_y, A_w, A_h) = (G'_x, G'_y, G'_w, G'_h)$, where $(G'_x, G'_y, G'_w, G'_h) \approx (G_x, G_y, G_w, G_h)$ [7] [1].

4. RoI pooling

   The RoI Pooling layer is responsible for collecting the proposals and computing the feature feature maps for subsequent network [7]. The Rol pooling layer has 2 inputs:

   Original feature maps

   Proposal boxes of RPN output (different sizes)

5. Classification

   The Classification part uses the acquired feature feature maps to calculate each proposal specific to that category (recycle, garbage) and output the cls-prob probability vector through the full connect layer and softmax. At the same time, the bounding box regression is used again to obtain the position offset of each proposal. Bbox-pred, used to return a more accurate object detection frame [7] [1]. The network structure of the Classification is shown in Figure 3.8:

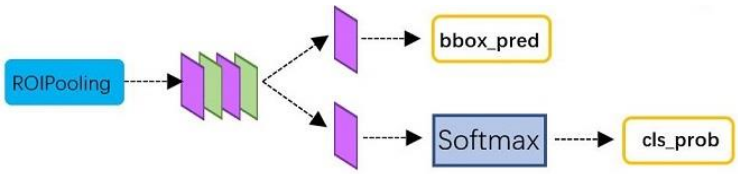

Figure 3.8: Classifying processing for Faster RCNN

3.1.2 The FRCNN model on Tensorflow

Tensorflow is the most popular and efficient deep learning platform available today. Based on the needs of this project, we construct the Faster RCNN model based on the the Tensorflow platform [1]. The actual structure of the FRCNN model for this project is shown in Appendix A.1

3.2 Tracking and counting

For the counting problem of the object detection, according to the data type processed by the Faster RCNN model, the data type is a video file, but actually processes the picture of each frame. For the target (garbage bag) recognition in each frame of the picture, a classification identifier is added to each target, and then the position information of the target is represented by a frame of a different color according to different categories [3]. If you identify and count each identified object, there are some special cases:

1. When an object has been identified and has been counted, but due to camera movement or object movement, the system will re-identify and identify the object, and the count will increase.

2. When an object is covered by another object, the counter will consider that the object is removed and the number will decrease.

3. When an object passes through the camera quickly, it is recorded, but due to the angle of the camera, after the object is recorded and moved to the visible range of the camera, the counter will also think that the object is removed, the number will be It is reduced.

To solve the above problem, we have devised a method to track the target using a Kalman filter. The implementation of the counting is such as showing in Figure 3.9:

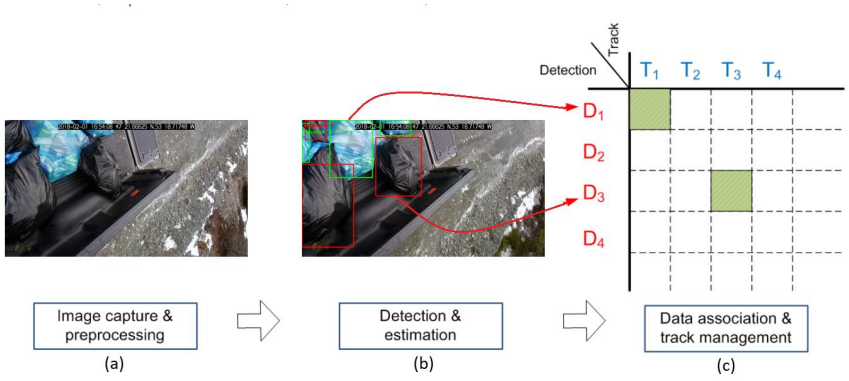

Figure 3.9: Tracking and counting based on Kalman filter

The module takes from current list of trackers and new detections, output matched detections, unmatched trackers, unmatched detections. If there are multiple detections, we need to match (assign) each of them to a tracker. We use intersection over union (IOU) of a tracker bounding box and detection bounding box as a metric [3]. We solve the maximizing the sum of IOU assignment problem using the Hungarian algorithm (also known as Munkres algorithm) [3]. The machine learning package

scikit-learn has a build in utility function that implements Hungarian algorithm.

3.3 Software installation and using methods

Software installation and use steps:

1. This project is developed under the windows 10 platform, the development environment is anaconda + tensorflow, the development languages are python, cuda and opencv.

2. First, install the scientific computing platform anaconda, install the deep learning virtual environment tensorflow based on this platform, and various application function libraries.

3. Download the label classification software labelImg, use this software to classify the garbage bags in each frame of pictures.

4. Activate the deep learning environment tensorflow, run the object detection and counting program. Specific implementation and installation commands are in Appendix A.2.

Chapter 4

Results and analysis

4.1 Detection results and FRCNN performance

After we complete the training, we get the important parameters of training which is total loss, it is showing in Figure 4.1:

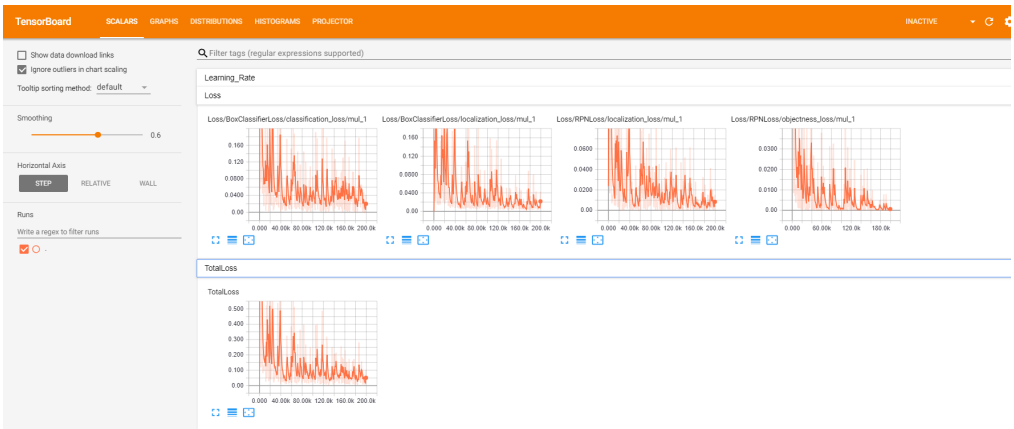

Figure 4.1: Rate performance charts of loss and total-loss for the best result

From these five graphs, we can obtain: During the whole training processing, overall loss value tends to decrease gradually. The loss value has large fluctuations in vibration before 60,000 steps. After 120,000 steps, the loss value will be gentle change, this means training processing will achieve the best results.

4.2 Accuracy analysis and Faster RCNN performance test

4.2.1 Accuracy analysis

The comparison of accuracy based on varies parameters are in Table 4.1 following:

Table 4.1: The comparison of results

sample space	Epoch	Batch size	Training time	Learning Rate	Accuracy
10,000 (1:4)	1000	10	15h05m55s	$0.000001 \sim 0.0001$	0.923
		50	13h36m45s		0.956
		100	17h16m33s		0.942
10,000 (1:4)	2000	10	29h56m05s	$0.000002 \sim 0.0002$	0.935
		50	26h51m55s		0.968
		100	34h05m05s		0.941
10,000 (1:4)	3000	10	42h14m55s	$0.000003 \sim 0.0003$	0.925
		50	38h46m37s		0.947
		100	50h27m56s		0.945

From the Table 4.1 above, we can obtain: For running time, the training processing speed with batch size equal to 50 is the highest. Depend on learning rate, the best accuracy was obtained in the area with from 0.000002 to 0.0002, and epoch is 2000.

4.2.2 Faster RCNN performance

The mean Average Precision (mAP) is the important verification parameter for Object Detection [5]. It is the metric to measure the accuracy of object detectors like Faster RCNN, SSD, etc. It is the average of the maximum precisions at different recall values [5]. It sounds complicated but actually pretty simple as we illustrate it with an example. For obtaining the mAP, we will get three important parameters that include precision, recall and IoU first [5].

Precision measures how accurate is your predictions. i.e. the percentage of your positive predictions are correct. Recall measures how good you find all the positives. For example, we can find 80% of the possible positive cases in our top K predictions [5].

Here are their mathematical definitions:

$$TP = True\ positive$$

$$TF = True\ negative$$

$$FP = False\ positive$$

$$FN = False\ negative$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

The Intersection over union (IoU) measures how much overlap between 2 regions, This measures how good is our prediction in the object detector with the ground truth (the real object boundary) [5]. The IoU is shown in Figure 4.2:

Figure 4.2: The Intersection over union process

We will describe how to demonstrate the calculation of the average precision (AP). In our dataset, we have a total amount for garbage and recycle bags in the whole dataset. We collect all the predictions the model made for bags and rank it according to the predicted confidence level (from the highest confidence to the lowest). AP (average precision) is computed as the average of maximum precision at all recall levels (it is divided by the amount to find the average):

$$AP = \frac{1}{n} \sum_{r \in \{P_r(i)\}}^{i=1} AP_r$$

$$= \frac{1}{n} \sum_{r \in \{P_r(i)\}}^{i=1} p_{interp}(r)$$

where

$$P_{interp}(r) = \tilde{r} \geq r^{max} \, \mathrm{p}\,(\tilde{r})$$

mAP is just the average over all classes. In many datasets, it is often called AP instead [5].

PASCAL VOC is a popular dataset for object detection. For the PASCAL VOC challenge, a prediction is positive if $IoU > 0.5$ [2] [5]. However, if multiple detections of the same object are detected, it counts the first one as a positive while the rest as negatives. The mAP in PASCAL VOC is the same as AP we discussed [5]. The final mAP of this project is shown in Table 4.2 following:

Table 4.2: The comparison of results

Model	Method	Proposals	Data	mAP(%)
VGG	Faster RCNN	2000	PascalVOC 2007	69.2
ZF-net	Faster RCNN	2000	PascalVOC 2007	75

From the Table 4.2, we could obtained that the ZF-net is better than VGG for the independent small dataset, the mAP value is bigger than 0.5 based on ZF-net so that the deep learning model is effective in this project [7].

4.3 Count results analysis

The final goal of this project is counting for the object detection, the detection, tracking and counting processing is showing in Figure 4.3, Figure 4.4:

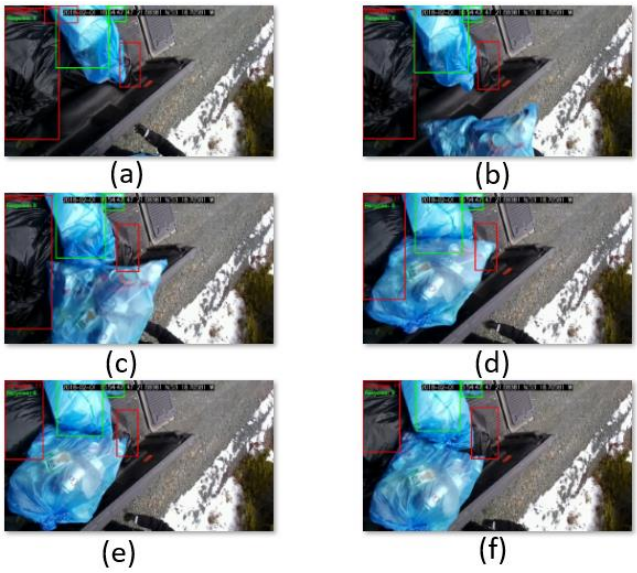

Figure 4.3: The example of Detection and counting for one item from appeared to detection and counting (1)

From all crop images of result videos clip, we can obtain: First, when the new moving object appear in the video, the detector executed detection processing delay some frames because the system will decide whether this moving object is new one [8]. Depend on Kalman filter processing and compute the distance of moving, the label box will be drawn and marked [3]. And then, the counter will plus one. In the Figure 4.5, the detector recognized this moving object is the new one, the count result plus one. For different scenarios in real world, the detection results are shown in Figure 4.6:

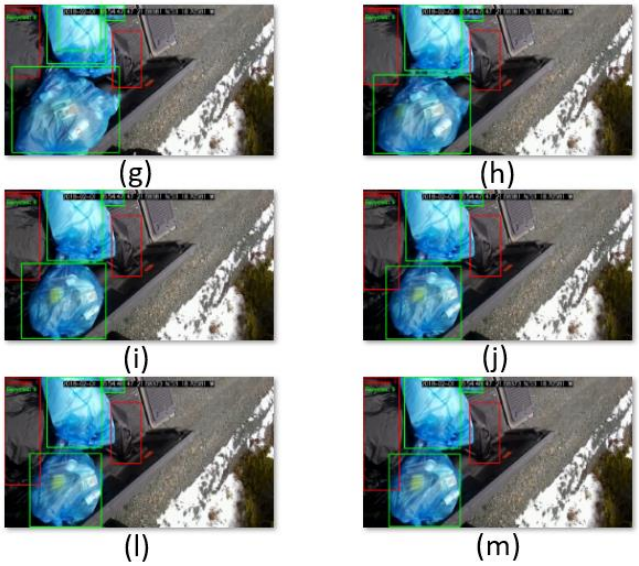

Figure 4.4: The example of Detection and counting for one item from appeared to detection and counting (2)

Figure 4.5: The example of Detection and counting for one item from appeared to detection and counting (3)

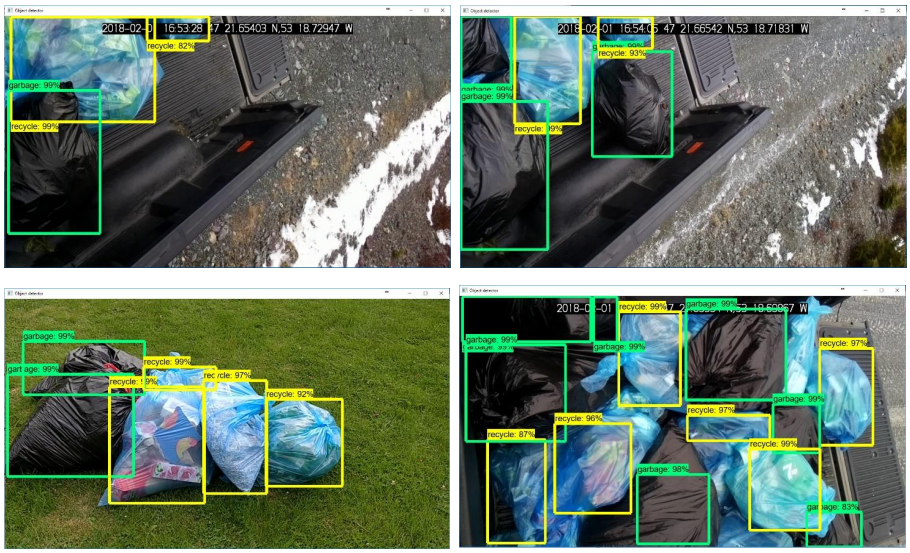

Figure 4.6: The detection results of verious scenario

Chapter 5

Discussion and conclusions

5.1 Conclusions

This project implements the object detection, tracking and counting function for garbage collection. In this project, we used the Faster RCNN model for classification. Because this model handles video files, the processing speed is fast enough for real-time video, making the target count more accurate [7].

During implement this project and training the Faster RCNN, we found: For the independent small data set, the ZF-net will obtain better accuracy [2] [6]. Modified the parameters, for example, modified the batch size, epoch, the training model running time and accuracy will change [4]. The learning rate will effect the final accuracy obviously [9]. For the object detection, the data set and label model is very important because the anchors will calibrate the object location after compare the ground truth during training [9] [7].

5.2 Future extensions

In this project, the object being tested is a garbage bag. The shape of the garbage bag has no fixed form. In order to make the recognition correct rate higher, when the object is marked, we will use the polygon to classify and position for the special object [6]. Also, we will use another deep learning model to reprocess this project.

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] 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*, 88(2):303–338, 2010.

[3] X. Hu, M. Bao, X.-P. Zhang, S. Wen, X. Li, and Y.-H. Hu. Quantized kalman filter tracking in directional sensor networks. *IEEE Transactions on Mobile Computing*, 17(4):871–883, 2018.

[4] L. Leal-Taixé, A. Milan, K. Schindler, D. Cremers, I. Reid, and S. Roth. Tracking the trackers: an analysis of the state of the art in multiple object tracking. *arXiv preprint arXiv:1704.02781*, 2017.

[5] K. Li, Z. Huang, Y.-C. Cheng, and C.-H. Lee. A maximal figure-of-merit learning approach to maximizing mean average precision with deep neural network based classifiers. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4503–4507. IEEE, 2014.

[6] 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*, pages 779–788, 2016.

[7] 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*, pages 91–99, 2015.

[8] K. P. Tsiaras, I. Hoteit, S. Kalaroni, G. Petihakis, and G. Triantafyllou. A hybrid ensemble-oi kalman filter for efficient data assimilation into a 3-d biogeochemical model of the mediterranean. *Ocean Dynamics*, 67(6):673–690, 2017.

[9] R. Van der Hallen, K. Evers, L. de Wit, J. Steyaert, I. Noens, and J. Wagemans. Multiple object tracking reveals object-based grouping interference in children with asd. *Journal of autism and developmental disorders*, 48(4):1341–1349, 2018.

[10] Y.-S. Yun, J. Jung, S. Eun, S.-S. So, and J. Heo. Detection of gui elements on sketch images using object detector based on deep neural networks. In *International Conference on Green and Human Information Technology*, pages 86–90. Springer, 2018.

Appendix A

Appendix title

A.1 Python Code of Discriminator

The FRCNN structure showing based tensor-board:

A.2 Software installation and use steps

See the website: https://github.com/youraustin/GWAS-CNN-/blob/master/The%20Genome%20information%20of%20final%20results.txt

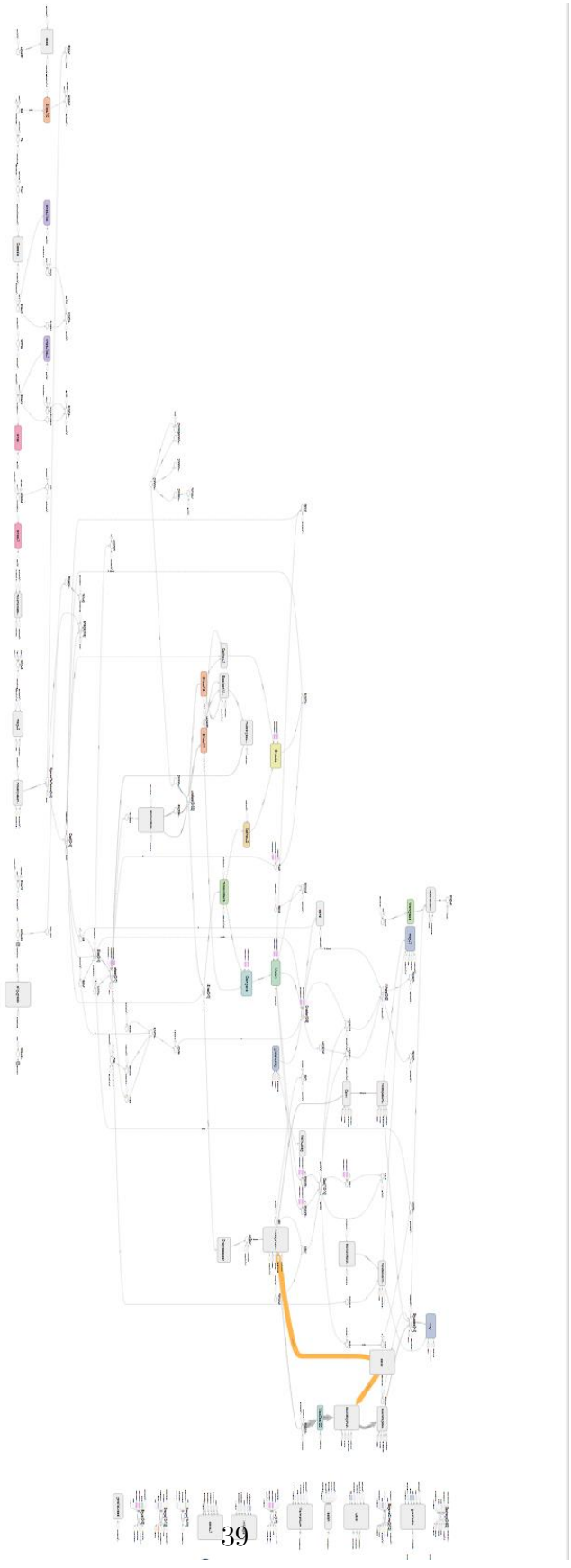

Figure A.1: Processing of Local Receptive Field