VIDEO SURVEILLANCE USING DEEP LEARNING WITH FEW DATA SAMPLES

BY

LIAN YEE FU

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology
(Kampar Campus)

JAN 2020

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: **VIDEO SURVEILLANCE USING DEEP LEARNING WITH FEW DATA SAMPLES**

Academic Session: **JAN 2020**

I **LIAN YEE FU**

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:
32, SUNGAI DURIAN,
TANJONG TUALANG, 31800,
PERAK

Ng Hui Fuang

Supervisor's name

Date: **21/04/2020**

Date: 21 / 4 / 2020

VIDEO SURVEILLANCE USING DEEP LEARNING WITH FEW DATA SAMPLES

BY

LIAN YEE FU

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology
(Kampar Campus)

JAN 2020

DECLARATION OF ORIGINALITY

I declare that this report entitled "**VIDEO SURVEILLANCE USING DEEP LEARNING WITH FEW DATA SAMPLES**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : **LIAN YEE FU**

Date : **21/04/2020**

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr. Ng Hui Fuang who has given me this bright opportunity to engage in deep learning based computer vision project. It is my first step to establish a career in deep learning based computer vision field. A million thanks to you.

Finally, I must say thanks to my parents and my family for their love, support and continuous encouragement throughout the course.

ABSTRACT

Data quantity is the essential element in determining the performance of an object detector as well as the performance of a video surveillance system. However, the availability of annotated image dataset for certain target domains is often limited. In a video surveillance system, the detector will usually have to detect new objects with limited annotated datasets due to the rapid changes of detecting requirements. As a reliable and flexible video surveillance system, the system should be able to perform object detection with limited annotated images provided yet with relatively good performance. Therefore, previous work on object detection framework using deep learning with few data samples is studied, implemented and improved. The improvements are performed to the existing framework to result in higher object detection performances with few data samples. In this project, a feature extractor *(YOLOv2)* will be implemented with a re-weighting module which is used to re-weight the feature extracted from feature extractor in order to detect *N* classes objects *(including new classes)*. By having this architecture, the model is able to reuse the prior knowledge on general object features *(edges and corners)* from feature extractor and combine with the class-specific feature from the re-weighting module. Since the amount of data is the key determinant on the performance of the object detector, therefore improvement is done by **increasing the amount of novel annotated images with different data augmentations** before model fine tuning to detect new objects. Nevertheless, the **re-weighting module is modified so that more information is captured** to re-weight the features from the feature extractor in order to detect new class objects. The experiment results showed that data augmentation and modified re-weighting module achieved **higher mean average precision *(mAP)*** because fine tuning data is increased and more re-weighting information is able to be captured.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BD	*Background Depression*
CNN	*Convolutional Neural Network*
FCN	*Fully Convolutional Network*
FPN	*Feature Pyramid Network*
FRCNN	*Faster Regional Convolutional Neural Network*
FSD	*Few Shot Detector*
GAN	*Generative Adversarial Network*
IOU	*Intersect of Union*
LSTD	*Low Shot Transfer Detector*
mAP	*Mean Average Precision*
NMS	*Non-Maximum Suppression*
RCNN	*Regional Convolutional Neural Network*
ROI	*Region of Interest*
RPN	*Region Proposal Network*
SSD	*Single Shot Detector*
TK	*Transfer Knowledge*
YOLO	*You Only Look Once*

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

Undeniably, deep learning-based models have achieved considerable human or beyond human level performance in solving computer vision problems. However, the deep learning-based models still require a **large amount of training data** in order to result in a good performance model. To support this statement, Brendan and Le (2018) concluded that larger datasets will result in better performance *(precision and recall)* of class imbalance problems in their research. Moreover, Pedro Domingos (n.d.) also claimed that if a dumb algorithm is fed with a massive amount of data, it can outperform a modest data fed clever algorithm. Hence, the effect of training data on the performance of deep learning-based models is significant. In video surveillance, new object detection and changes of illumination level introduce some challenges in the video surveillance monitoring. The challenges are discussed below:

1) **New object detection**: In general, video surveillance cameras monitor a certain target domain object at a fixed location for long periods of time. However, if the viewing angle or position of the camera is changed or the target domain objects are changed, it is a challenge for the surveillance system to collect enough training data ***(especially the dataset for the new object is rare)*** in a short period of time in order to learn to detect the new object categories with good performance.

2) **Illumination change**: In video surveillance, surveillance cameras are operated all day long under different weather conditions *(cloudy, thunderstorm, sunny)*. When the light in the scene changes gradually *(for example, from morning to evening, the light dims gradually, raining)* or suddenly *(for example, when the light in a room has been turned on)*, how the model can robustly detect that the target object is from the same category rather from a new category simply because illumination level changes.

1.2 BACKGROUND AND MOTIVATION

A reliable and scalable video surveillance should be able to detect new objects as quickly as possible with few datasets in order to save modelling and training time. Currently, the most popular approach to train a model to detect new object categories with few datasets is through transfer learning. However, direct **transfer learning is not suitable for object detection** and it is also a difficult task compared to transfer learning in object classification. This is because **over-fitting problems may occur especially when the datasets are limited** as the object detector is required to learn the optimal parameters for both localization and classification (Chen et al. 2018). Due to this reason, reinitializing and retraining a few last layers of the model in conventional transfer learning technique on object classification cannot be simply applied on object detection.

Few shot learning with few dataset in object detection is a worthwhile research area as it provides a direction on how to adapt the object detector in a short period of time with few data. By having this research, the challenges of video surveillance mentioned in the previous section can be solved easily with small amounts of new object categories data. On the other hand, the result of this project is not only limited within video surveillance but it is also applicable in other fields as well especially the fields such as medical and criminal face detection where the data is usually limited during building deep learning models.

1.3 PROJECT OBJECTIVES

The main objectives of this project are to implement and enhance the existing object detection architecture with few dataset for video surveillance by altering the re-weighting module and applying few data augmentation techniques. The objectives to be achieved are as below:

a) Higher mean average precision *(mAP)* on new class object detection through applying **data augmentations** before fine tuning.
b) Higher mean average precision *(mAP)* on new class object detection through **modifying the re-weighting module**.
c) Higher mean average precision *(mAP)* on new class object detection through applying **data augmentation and modification of the re-weighting module**.

1.4 PROPOSED APPROACH / STUDY

In this project, few shot object detection via feature re-weighting (Kang et al. 2018) is referred to as the architecture for **few shot meta learning**. The architecture is composed of 2 modules, feature extractor module and re-weighting module. This model provides a promising few shot learning solution for object detection by adding a re-weighting module to re-weight the features extracted from feature extractor to detect the *N* classes objects *(base and new objects)*. However, in order to further enhance the ability and accuracy of the model on meta learning, **data augmentation** operations are performed on the data collected before model fine tuning to detect new class objects. The data augmentation applied is categorized into basic and combined data augmentation. The basic data augmentation techniques are **random rotation, random translation, random scaling, random shearing, random horizontal flip and random colour jitter**. Colour jitter augmentation is applied to reduce the illumination changes problem mentioned in the challenges of video surveillance monitoring as this augmentation technique enhances the detection generalizability of the same class in different illumination levels. Combined augmentation is the sequential data augmentation from the basic data augmentation techniques mentioned above to further diversify the limited annotated dataset. Other than data augmentation, the **re-weighting module is altered** to retain more class specific information in order to enhance the re-weighting strength of the model for detecting new objects.

1.5 PROJECT ACHIEVEMENTS

The performance of the referred object detector (Kang et al. 2018) is improved after the adding of data augmentation pipeline and modification of re-weighting module. The data augmentation improved the *mAP* with the average of ***1.68 mAP*** whereas the modified re-weighting module improved the *mAP* with the average of ***1.24 mAP***. However, the combination of both data augmentation and modified re-weighting module improved the performance of the detector by around ***2.22 mAP*** which is higher than the improvements by data augmentation or modified re-weighting module alone.

1.6 IMPACT, SIGNIFICANCE AND CONTRIBUTION

In this project, a deep learning approach for video surveillance by using a **small amount of training data** is studied and enhanced. The proposed approach is able to fine tune the models with the small amount of annotated data collected in new environments or scenes and **perform object detection with higher accuracy**. With the completion of this project, an enhanced deep learning-based approach video surveillance using small amounts of dataset will be delivered. Moreover, the proposed approach is able to help the system in adapting and detecting for new objects despite the challenges encountered in video surveillance such as the changes of illumination level and changes of target domain objects. In such a way, the performance of video surveillance can be maintained even with the limited dataset provided.

The results of the project can be applied to all video surveillance in the area of remote video monitoring, facility protection, monitor operations, loss prevention, vandalism deterrence, public safety, production line and et cetera. Moreover, the results of this project is not only limited within video surveillance but also can be applied to the fields where the annotated dataset is limited for object detection such as face detection, face recognition, manufacturing industry, medical imaging and so on.

1.7 REPORT ORGANIZATION

Table 1.7.1: Report organization.

Chapter 1: Introduction	➢ Problem Statement ➢ Background and Motivation ➢ Project Objectives ➢ Proposed Approach / Study ➢ Project Achievements ➢ Impact, Significance and Contribution ➢ Report Organization

Chapter 2: Literature Review	➢ Deep Learning Video Surveillance ➢ Transfer Learning ➢ Meta Learning ➢ Object Detection Models ✧ Two Stage Object Detection ✧ One Stage Object Detection ➢ Low Shot Object Detection Transfer Learning ➢ Proposed Study Comparison
Chapter 3: System Design	➢ Design Specifications ✧ Few Shot Learning ✧ Data Augmentation ✧ Re-weighting Module Modification ➢ Design Overview
Chapter 4: Result Analysis and Tools	➢ Performance Analysis ➢ Detection Visualization (New Classes) ➢ Research Tools
Chapter 5: Conclusion	➢ Discussions ➢ Implementation Issues and Challenges ➢ Novelties and Contributions ➢ Future Works

CHAPTER 2 LITERATURE REVIEW

2.1 DEEP LEARNING VIDEO SURVEILLANCE

Video surveillance is an action of monitoring or observing a scene to discover abnormal behaviour that indicates the occurrence of improper activities (Lawrence 2017). The backbone of video surveillance is an object detector which performs both object localization and object classification on the input video stream. The frames of the input video stream are extracted and object detection is performed on the frames to identify the location and class of the object in the image. Therefore the efficiency and performance of video surveillance greatly depends on the object detector architecture in the backbone. If the object detector is not responding fast enough to the input frames, delay problems will occur during video surveillance and the system is not able to capture and alarm abnormal activities in real time.

Conventional video surveillance monitored by human operators is impractical because humans have limited ability and human resources are expensive (N. Sulman et al 2008). Due to the high demand for better security and the desire to replace human operators, the growth of intelligent video surveillance systems is faster than ever before. Intelligent video surveillance is a combination of computer vision and deep learning that aims to produce a more reliable and highly efficient surveillance system. The expanding and growing of intelligent video surveillance field leads the field to become one of the important research areas in deep learning because the integration of deep learning and video surveillance unlocked the vision of the computer to enable it to detect, recognize and monitor the objects as good as human or even beyond human level. The convention flow of intelligence video surveillance systems is shown in *Figure 2.1.1.*

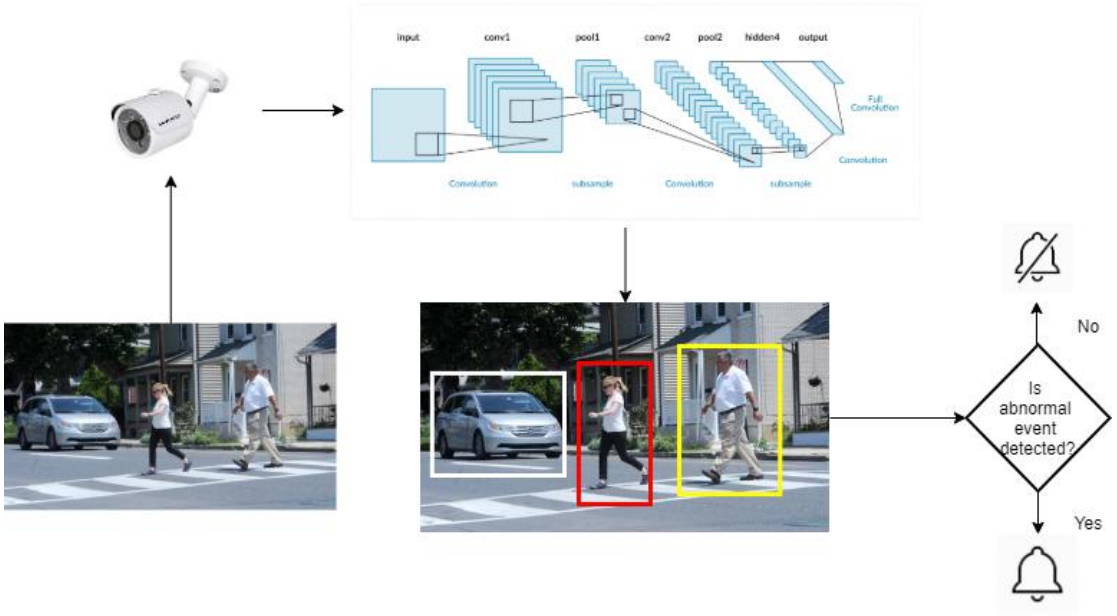


Figure 2.1.1: Normal flow of intelligence video surveillance systems.

As shown in *Figure 2.1.1*, the input video frame from the video surveillance system is fed into the convolutional neural network object detector. The features of the input image such as texture, edge, shape, motion of are extracted for the later layers to identify the existence of objects in the image. Based on the existence of objects or the activities, the system will raise alarm to inform the detection of abnormal events.

Deep learning based indoor video surveillance by using Mask RCNN (Liu et al 2019) which was pre-trained on the MS COCO dataset is an example of video surveillance using a deep learning approach. Mask RCNN as its name suggests, it performs instance segmentation along with class and bounding box predictions for the objects in the image. This purpose of applying instance segmentation in this video surveillance system is to further understand the semantic information of video content and raise alarm for abnormal events.

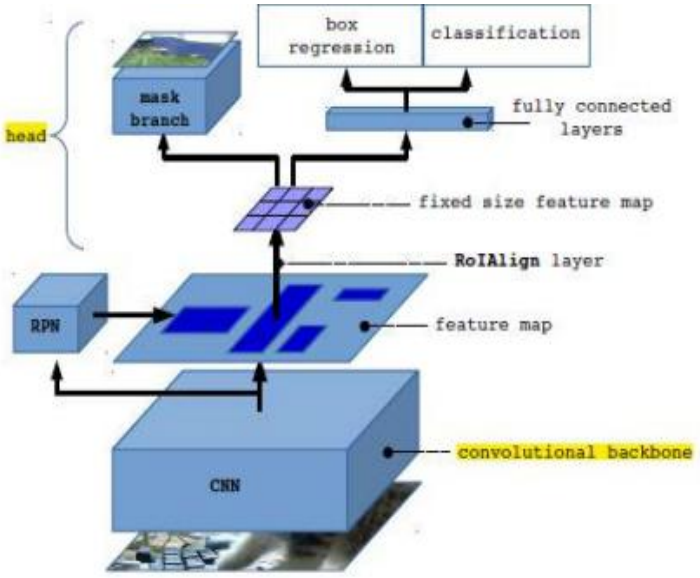


Figure 2.1.2: Mask RCNN network architecture. (Liu et al. 2019)

Basically, Mask RCNN is categorized into 2 parts, an upper layer network and a lower layer backbone architecture. Lower layer neural network is the ResNet or FPN convolutional neural network and it is mainly used to extract the features of the image, whereas the upper layer network is a Faster RCNN with Fully Convolutional Network *(FCN)* used to predict classification and border regression parallel with performing mask prediction in mask branch .

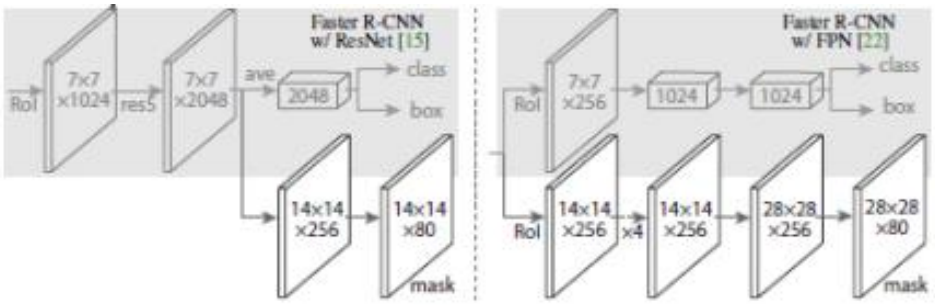


Figure 2.1.3: ResNet as backbone network (left), FPN as backbone network (right). (Liu et al. 2019)

Figure 2.1.3 above shows the Mask RCNN upper layer network. The grey section shows the Faster RCNN combined with ResNet or FPN network whereas the lower section shows the added FCN for the prediction mask. Feature Pyramid Network *(FPN)* combines several image information such image features and

semantic information. As a result, the ROI features extracted are different scales and rich with semantic information for mask prediction.

2.2 TRANSFER LEARNING

On the other hand, transfer learning is a technique of transferring knowledge from learned source domain tasks and reuse it to enhance the learning in target domain tasks like how human learners reapply knowledge from previous experiences to the new tasks encountered (Lisa.T & Jude.S, n.d.). In fact, people solely train CNN from scratch because of insufficient data size. In practical terms, people will prefer to fine tune a pre-trained CNN on a massive image dataset such as ImageNet *(consists 1000 classes)* by using transfer learning to detect new target domain objects. Due to the fact that CNN features are more general in the few previous convolutional layers and more task-specific in the later layers, if the new dataset is limited compared to the original dataset, only last few layers of CNN are required to re-initialize and re-train with the new dataset in order to detect the new objects. Previous CNN layers can be preserved as they help the model in detecting general features *(low to intermediate level)* such as edges and corners which are very useful in detecting new objects.

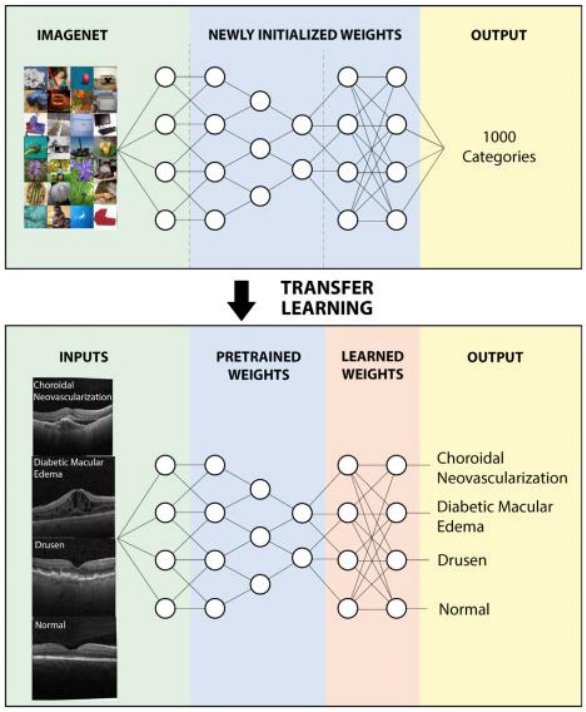


Figure 2.2.1: General transfer learning in object classification (Beenish.Z, Ramesh.I & Bob.R, 2018).

In domains such as medical image processing and remote sensing where training samples are difficult to obtain, many studies have been carried out to apply transfer learning using only a small amount of training data. Transfer learning uses prior knowledge learned in a known domain and applies it in the target domain, so that the related problem can be solved using a smaller number of training samples obtained in the target domain. Several studies have investigated the issues of how to transfer the prior knowledge learned in a source domain, which prior knowledge should be discarded and which should be transferred. For example, 2018 conference paper by Li et al. shows the success example of transfer learning to card object domain on YOLO network architecture. As a result, the approach takes full advantage of real-time detection features of YOLO network and enhancement of the generalization from transfer learning.

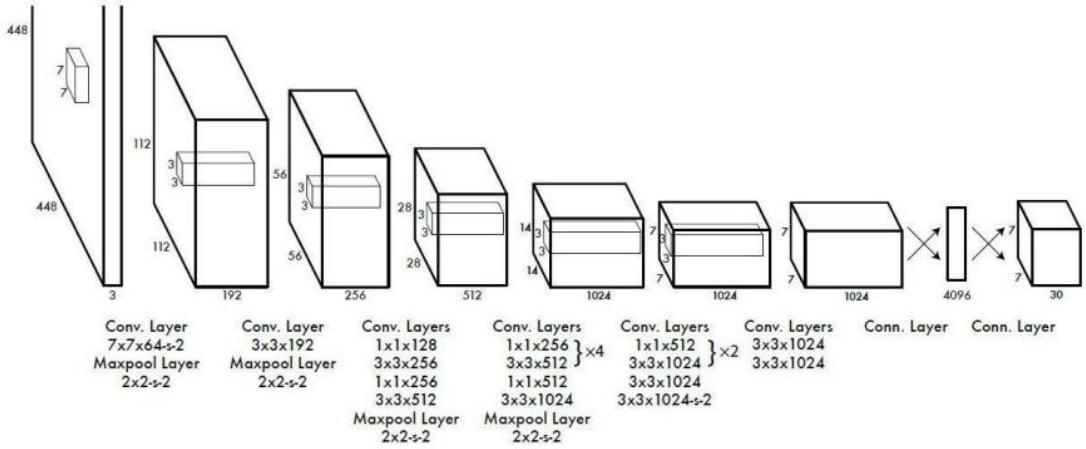


Figure 2.2.2: YOLO network architecture. (Li et al. 2018)

YOLO is a one-stage object detection method. First the image is divided into 7x7 grids and next several anchor boxes of different sizes are set for each grid to detect object and predicts the category that the object belongs to. The final output contains the regression parameters of the position of the anchor boxes in each grid. The advantage of such one-stage method is that it is very fast compared to two-stage object detection approaches. *Figure 2.2.3* below shows the overall training process of transfer learning on YOLO by Li et al. 2018.

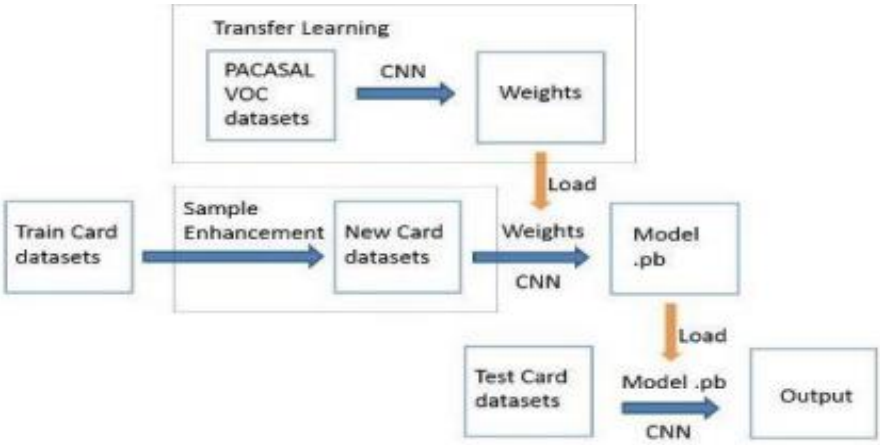


Figure 2.2.3: Overall training process of transfer learning on YOLO. (Li et al. 2018)

First, the network is trained on PASCAL VOC dataset, a large dataset with rich image samples, and next the trained network is used as basic weight initialization for training the network on new dataset. Since the object categories are different between PASCAL VOC dataset and the new dataset, in the paper there are 6 types of cards, the weight of the classification layer needs to be randomly initialized before fine-tuning it. Because the small dataset contains only 300 samples of poker cards, the dataset is augmented before being entered into the network for fine-tuning. Three conclusions can be drawn from the experimental results of this approach. First, while performing transfer learning under the same object detection framework, it is better to select lighter network architectures for small sample size. Complex networks such as the two-stage networks, although their accuracy is usually higher than the one-stage object detection models, they do not show outstanding detection accuracy when the dataset is small. Second, when the number of training samples is less than 50, the performance of the network model will decline rapidly. Finally, the current experiment is to detect different types of poker images under a simple background. It is worth exploring that in a more complex background, whether one can still perform transfer learning using small sample size without sacrificing its accuracy.

One approach proposed by Wang et al. (2018) is also an example of transfer learning on the existing network. The method combines SSD and transfer learning to detect ships in complex environments, such as in the ocean or near islands. Previous ship detection focused on detecting vessels in the centre of the ocean where the background was relatively simple. Because of the difficulty of obtaining images,

about 200 small images of inshore vessels were used as training samples in this article. SSDs were selected because they can improve detection accuracy more than YOLO while maintaining a certain speed.

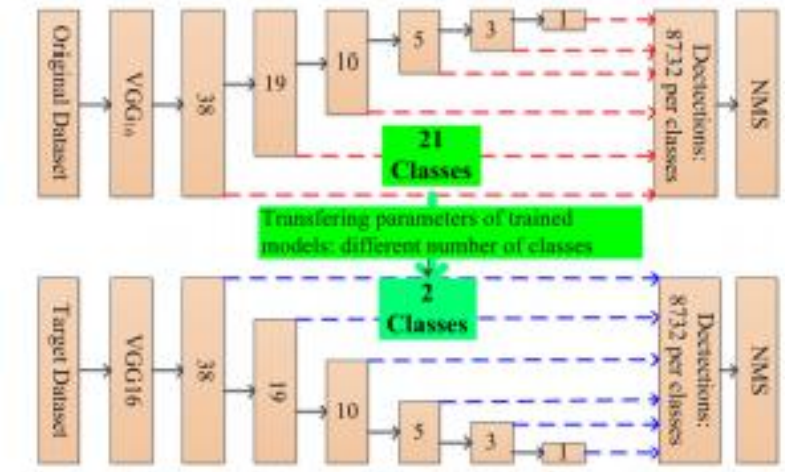


Figure 2.2.4: Transfer learning based on SSD Architecture. (Wang et al. 2018)

Figure 2.2.4 shows the process of transfer learning based on SSD architecture. First the SDD network is trained using the original large-scale dataset. Since the target domain contains only 2 types of vessel and the background, the parameters of the classification layer are randomly initialized and then fine-tuned with a small number of new samples. The proposed approach has 2 main issues: First, the target domain contains only 1 ship category, will the framework still work when the number of categories increases? And second, data augmentation is important for training with small samples, avoiding over-fitting due to the small amount of training samples. Moreover, traditional transfer learning is not suitable for urgent tasks and dynamic environments because it requires hundreds of iterations training on target domain (Kun.F et al. 2019).

2.3 META LEARNING

Meta-SSD is a meta learning approach for few shot object detection fast adaptation (Kun.F et al. 2019). Meta-SSD is proposed to overcome performance, long training processes and over-fitting problems due to limited dataset size. The proposed architecture consists of an object detector and a meta-learner. With the existence of meta-learner, the object detector is able to recognize from limited examples in a single

updating step. Therefore, Meta-SSD is able to adapt quickly with learned general knowledge across different tasks.

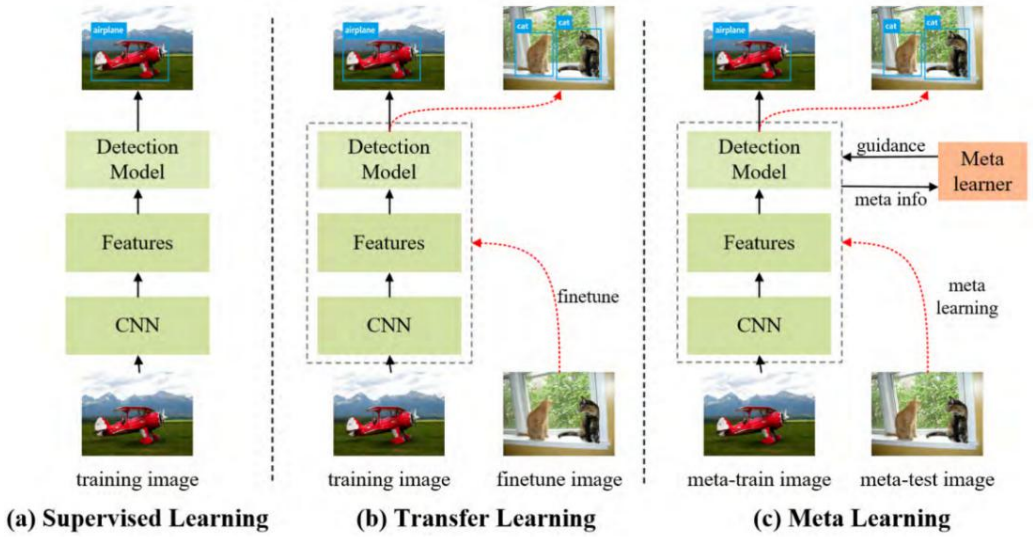


Figure 2.3.1: Comparisons between supervised learning(a), transfer learning(b) and meta learning(c) (Kun.F et al. 2019).

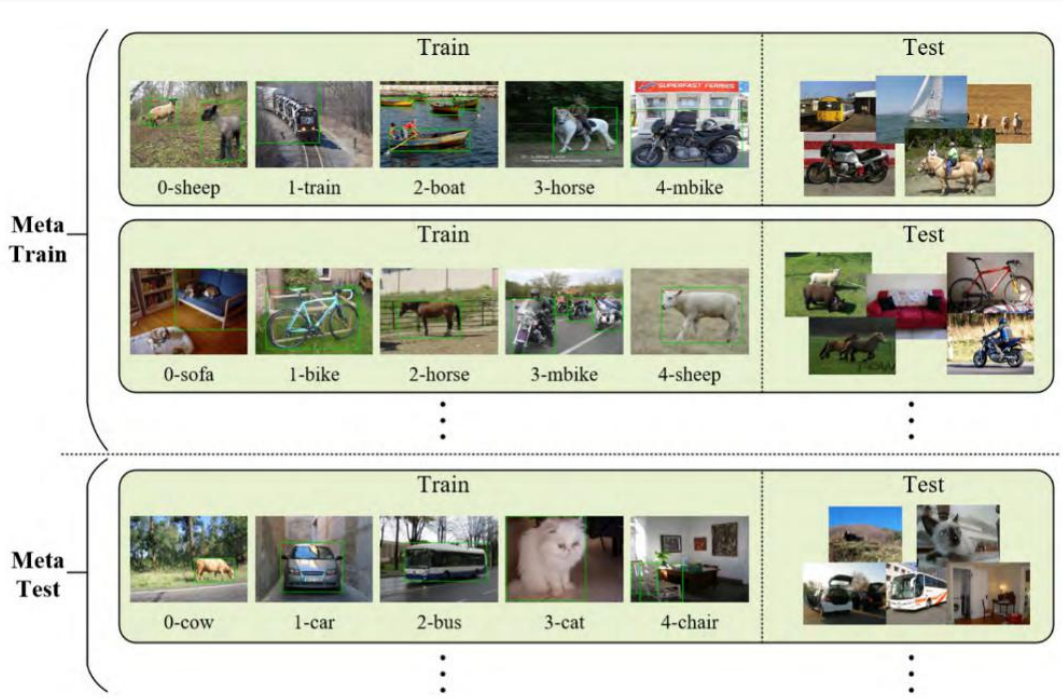


Figure 2.3.2: Example of dataset up (5 way 1 shot) in Meta-SSD (Kun.F et al. 2019).

Based on the example of dataset up in *Figure 2.3.2* the dataset (PASCAL VOC) is categorised into meta-train and meta-test. Meta-test contains the objects of new

classes but meta-train contains the objects of seen classes. In other words, meta-test is a new task for the model to adapt. There are further divisions of training subset and testing subset data in each task (corresponds to each row in *Figure 2.3.2*) where the training images and testing images are from the same classes. For meta-test's training set and meta-train's training and testing subset, 1 image is randomly selected per class, whereas 15 images per class are randomly selected for the testing set of meta-test as an evaluation benchmark. The training subset is used to train the object detector by updating the parameters with meta-learner's guidance whereas the testing subset is used to fine tune the meta-learner. By having the training design this way, the model can learn how to detect new objects with limited data in meta-test under the guidance of the meta learner. In short, the whole training process is to train the meta-learner on learning how to detect unseen objects with limited data provided.

2.4 OBJECT DETECTION MODELS

Object detection is an algorithm to classify objects in the image and predict the locations of the classified objects (Hulstaert 2018). In order to classify and detect an object(s) in an image, 2 sets of output are predicted; class probability and bounding box of the object(s). There are few existing models that are able to perform object classification and detection tasks with satisfactory accuracy and speed. The examples of the popular existing models are **Faster RCNN** (Ren et al. 2015), **Single Shot MultiBox Detector** (Liu et al. 2016), **YOLO** (Joseph et al. 2015) and **Low-Shot Transfer Detector for Object Detection** (Chen et al. 2018) to facilitate transfer learning in object detection.

2.4.1 TWO STAGE OBJECT DETECTION

RCNN performs object detection based on the region proposals generated by selective search (Ross et al. 2013). Selective search will generate around 2000 region proposals per image for object detection. However, the region proposal computation in RCNN is computationally expensive and it is impractical to be implemented as a real time object detection because of the slow response time. Moreover, no learning process is undergoing in selective search because it is a static algorithm and this will lead to bad region proposals. The overview of RCNN is simplified in *Figure 2.4.1* below.

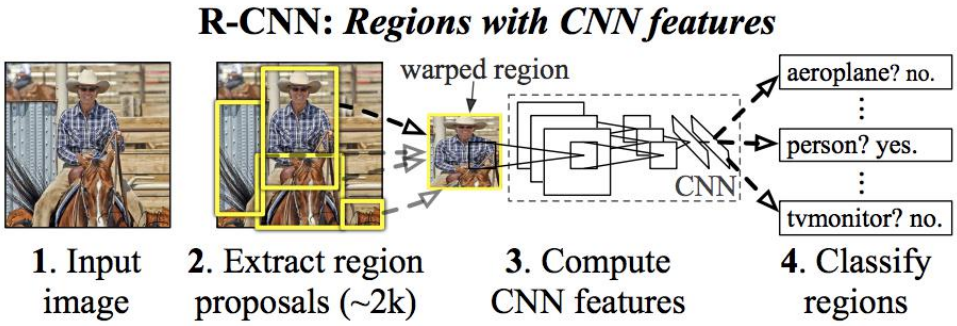


Figure 2.4.1: Overview of RCNN with selective search. (Ross et al. 2016)

The improved version of RCNN is Fast RCNN proposed by Ross et al. (2015). Instead of generating 2000 region proposals before feeding into CNN, Fast RCNN takes an approach to first generate convolutional feature maps on the input image and then apply selective search to generate region proposals from the feature maps. The region proposals are wrapped into fixed size in order to pass them into the fully connected layers for object detection. *Figure 2.4.2* below shows the overview of Fast RCNN.

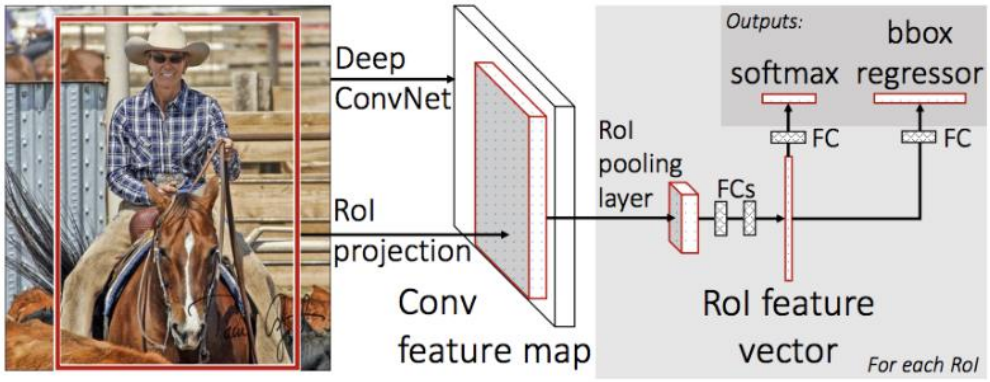


Figure 2.4.2: Overview of Fast RCNN. (Ross et al. 2015)

As a move to further improve the speed of Fast RCNN, Faster RCNN is proposed to resolve the computationally expensive region proposal problem in both RCNN and Fast RCNN. Faster RCNN introduces a region proposal network *(RPN)* on top of the existing Fast RCNN to share the image convolutional features with the detection network to achieve a low-cost region proposal. RPN is proposed to replace computational expensive selective search to generate region proposals in both Fast

RCNN and RCNN. RPN is a fully convolutional neural network that predicts the region proposal for object detection and classification.

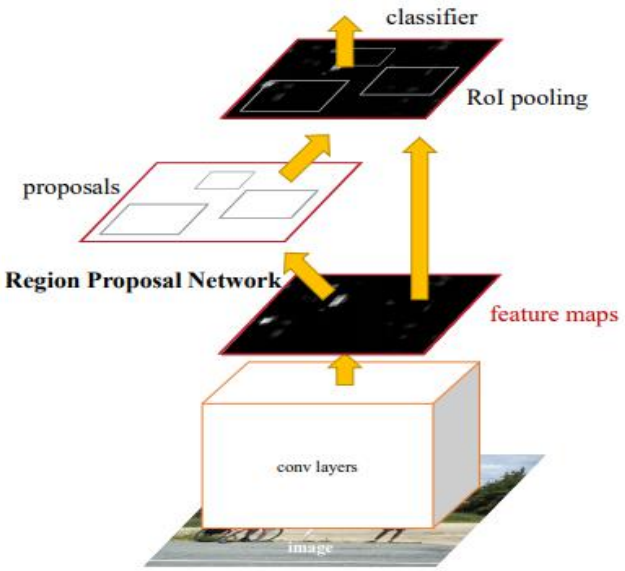


Figure 2.4.3: Faster R-CNN Architecture. (Ren et al. 2015)

Faster RCNN consists of 2 two modules including deep fully convolutional network, region proposal network that proposes regions of interest and Fast RCNN detector that uses the results from RPN to detect and classify the object. The networks are unified and combined as a single network for object detection as shown in *Figure 2.4.3* above.

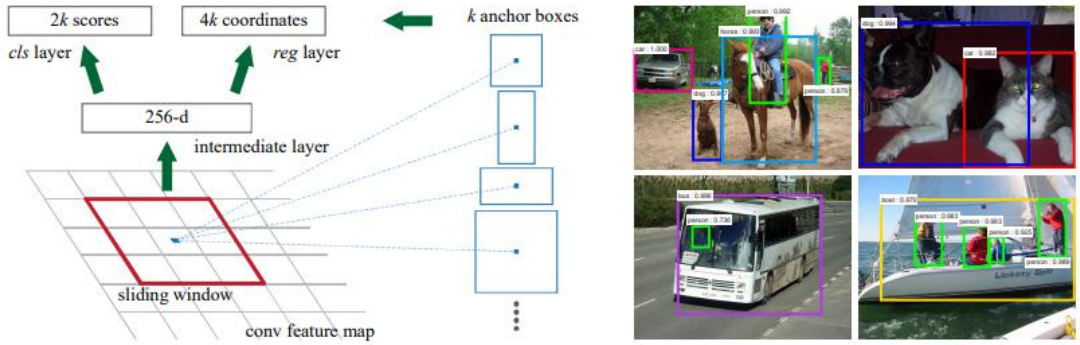


Figure 2.4.4: Region Proposal Network, RPN (left). Demonstration of results of implementing RPN (right). (Ren et al. 2015)

RPN proposes multiple regions simultaneously by sliding a window over the input image. As shown on *Figure 2.4.4 (left)* above, the sliding window will propose *k*

possible predefined anchor boxes at each grid cell. Then the *reg* layer will compute *4k* coordinates which indicate the location of bounding boxes and the *cls* layer will output *2k* scores to predict the probability of objectness for each proposal. Since *k* proposals will be parametrized according to *k* anchor boxes, therefore this allows translation invariant properties in detecting image features because if the object in the image is translated, the proposal region and the function that is used to compute the proposal region should be translated accordingly.

2.4.2 ONE STAGE OBJECT DETECTION

In order to even speed up the object and detecting process, models are proposed to perform region proposal, classification and bounding box regression at the same time. The popular one stage object detectors are YOLO and SSD. YOLO is an approach to achieve real-time object detection with quick response time and high accuracy. YOLO's approach is to treat object detection as a regression problem to predict the bounding boxes and probabilities of each class. In this approach, bounding boxes and class probabilities will be predicted in one stage. The performance of the detection is optimized because the classification and bounding box prediction are performed at the same time.

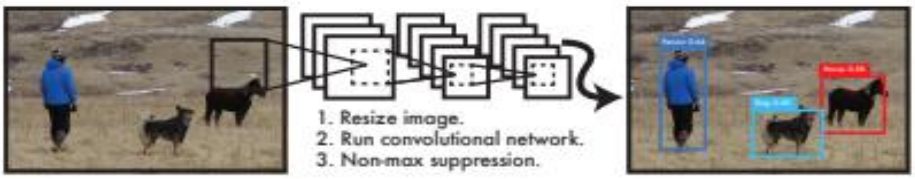


Figure 2.4.5: YOLO Detection System. (Joseph et al. 2015)

Figure 2.4.5 shows the YOLO detection system. Multiple bounding boxes and class probabilities for the boxes are done in single evaluation unlike the architecture in Fast RCNN or Faster RCNN which consist of a separate region proposal model. This approach has few benefits compared with traditional approaches to object detection such as improvement of performance, implicitly learning the contextual information of the classes and highly generalizable on object detection.

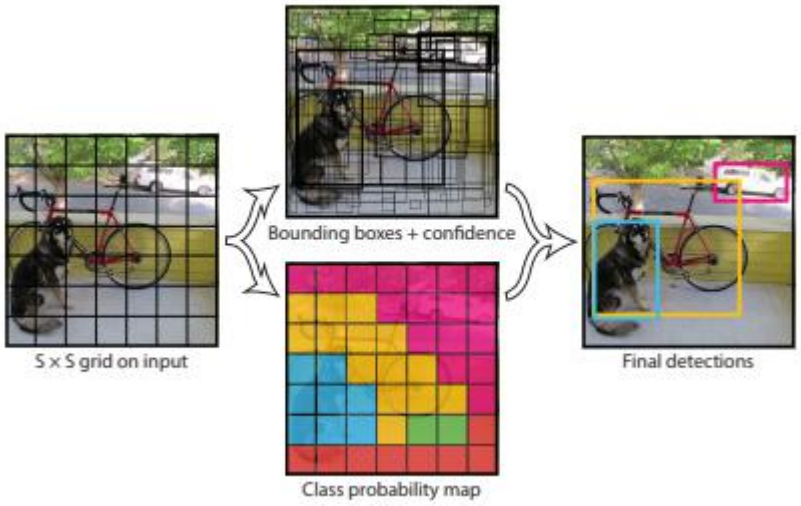


Figure 2.4.6: YOLO Detection Model. (Joseph et al. 2015)

YOLO model divides the image into an S x S grid cell as shown in *Figure 2.4.6.* The grids will be responsible for detecting the object if the centre of the object falls on that particular grid cell. Bounding boxes B and confidence scores *(class probabilities)* for the boxes will be calculated by each grid cell to determine whether the grid cell consists of object or not. If no object is found on the grid cell, the confidence scores will be 0, otherwise the confidence scores will be equal *Pr(Object) x IOU(truth pred)*. Other than confidence score, each grid cell will also calculate a conditional class probabilities, $Pr(Class_i \mid Object)$ if the grid cell contains an object to determine the class of object in that grid cell. Moreover, a set of class probabilities will be calculated on each grid cell, regardless of the number boxes B. Then, class-specific confidence scores for each box will be calculated at the test time by multiplying *Pr(Object)* and $Pr(Class_i \mid Object)$ and which will eventually produce the class probability map as shown in *Figure 2.4.6* above. Finally, non max suppression is applied on the predicted bounding boxes to remove the overlapped redundant bounding boxes.

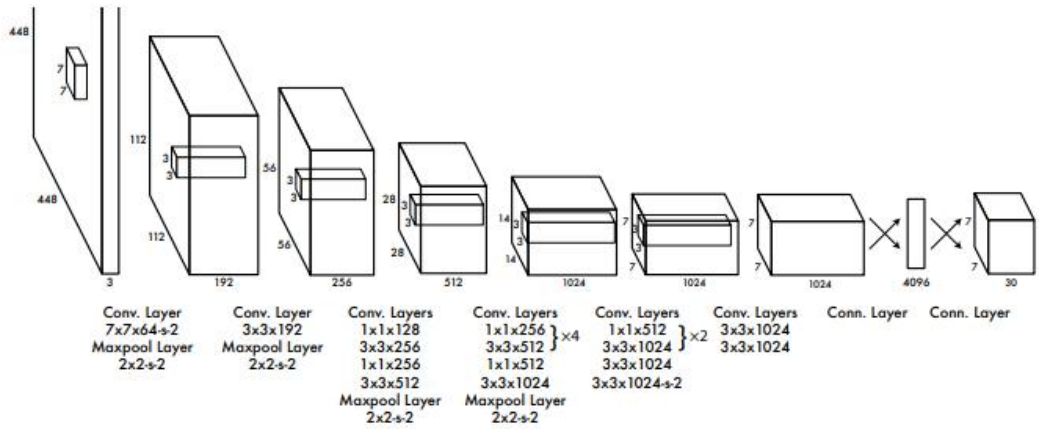


Figure 2.4.7: YOLO Architecture. (Joseph et al. 2015)

YOLO architecture consists of 24 convolutional layers and 2 fully connected layers. Beside that, alternating l x l convolutional layers are used to reduce the feature space from preceding layers in order to reduce the computational power spent on training and reduce over-fitting problem.

However, YOLO has several drawbacks due to the network and architecture design. First, strong spatial constraints are imposed on the bounding box predictions as each grid cell is restricted to predict 2 boxes and can only predict 1 class. The model struggles on detecting small objects among groups such as flocks of birds. Second, the model will have a difficult time when detecting objects in new or different configurations or aspect ratios because the model is restricted to predict bounding boxes from given data. Lastly, the loss function applied treat errors the same in both small and large bounding boxes. This will greatly affect the IOU of small bounding boxes.

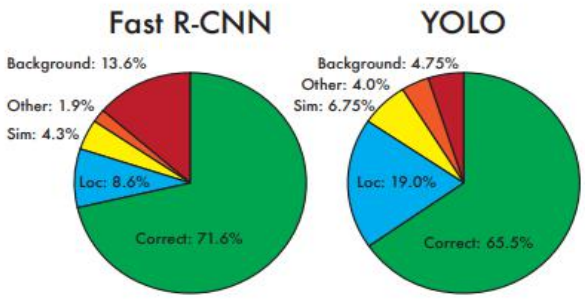


Figure 2.4.8: Error analysis (localization and background) between Fast RCNN and YOLO. (Joseph et al. 2015)

The classification of the section in the pie chart in *Figure 2.4.8* above is based on the rules below:

- Correct: correct class and IOU > .5
- Localization errors: correct class and .1 < IOU < .5
- Similar objects: class is similar and IOU > .1
- Objects: class is wrong and IOU > .1
- Background or unlabelled objects: IOU < .1 for any object

The comparison result shows that YOLO will have difficulty to localize objects correctly compared to Fast RCNN but YOLO performs better in differentiating the background from the objects.

Single Shot Multibox Detector *(SSD)* is another one stage detector approach to completely remove proposal generation. SSD is able to perform object detection and recognition faster than other two stage approaches such as Fast RCNN because of the removal of computational expensive proposal generation. In SSD, different resolution feature maps are combined to handle different size objects. During the prediction and classification, scores are computed for the presence of object category and adjustments are made on the box to match the object shape better. However, the main drawback is deterioration of accuracy. SSD discrete output of bounding boxes into a few default boxes over different scales and ratios per feature map location as shown in *Figure 2.4.9.*

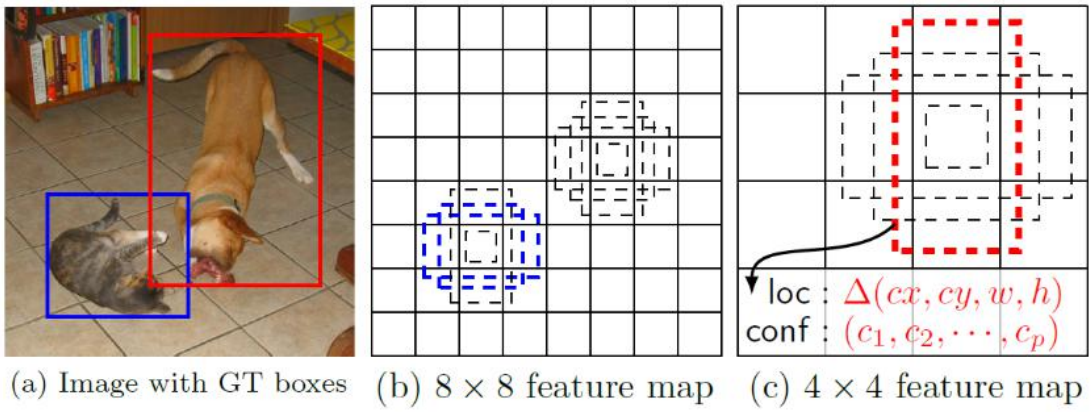


Figure 2.4.9: Different feature maps (b&c) predicted by SSD.

SSD is an approach on a feed-forward convolutional network that results on a fixed-size of bounding boxes and scores for object class instances. SSD consists of a

few extra feature layers at the end of the base network which are used to predict the offsets of different aspect ratios, scales and the corresponding confidence scores.

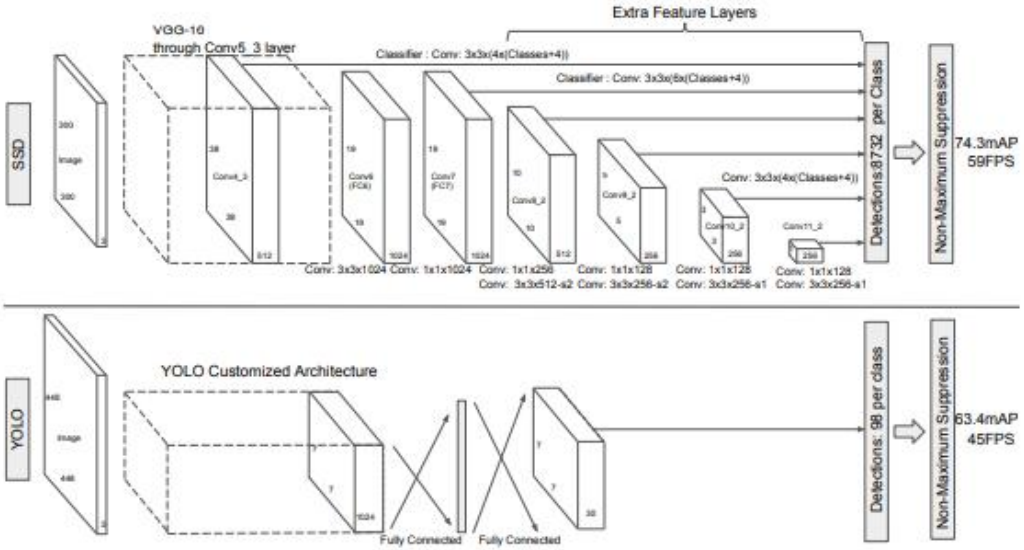


Figure 2.4.10: Comparisons between 2 one shot detection models, SSD and YOLO. (Liu et al. 2016)

2.5 LOW SHOT OBJECT DETECTION TRANSFER LEARNING

Most of the methods reviewed above require large datasets to achieve expected accuracy. However, Low-Shot Transfer Detector *(LSTD)*, the deep framework proposed by Chen et al. (2018) is able to build the object detection model by using transfer learning. This framework is proposed to address the challenges encountered in transfer learning in low-shot detection.

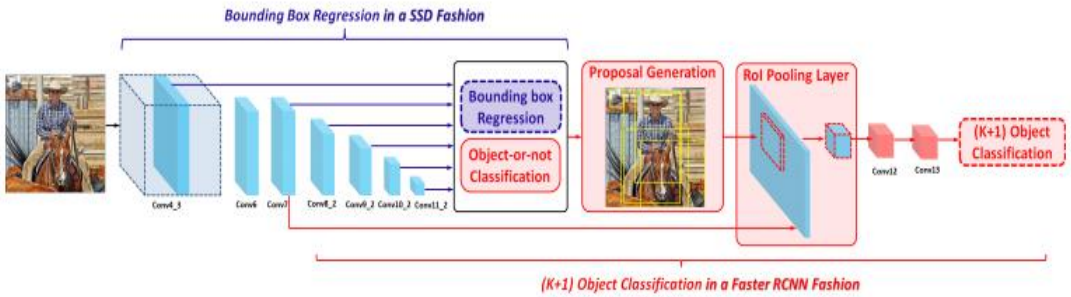


Figure 2.5.1: Basic deep architecture of low-shot transfer detector (LSTD). (Chen et al. 2018)

In LSTD framework, Faster RCNN is selected as the two-stage object detection method. In the original Faster RCNN, RPN layer is first used to propose candidate anchor boxes which might contain objects in the input image, and then feeds the candidate anchor boxes into Faster RCNN for object classification and position regression. In this method, the regression of anchor boxes is separated from object classification. As a result, the transferability of the model will be reduced when the regression parameters are randomly initialized without using parameters that are trained from the source domain. On the other hand, unlike Faster RCNN which uses only the last layer of feature map, in SSD anchor boxes are set on multiple layers of feature maps. Multilayer feature maps can better deal with objects with different scales. Especially in the case when the number of samples is small, it is necessary for the network to have the ability to learn to detect objects at different scales. Therefore, the regression task in object detection will be implemented using SSD. In addition, as mentioned above, SSD sets the anchor boxes on each layer of the feature map, and directly classifies multiple object categories on each of the anchor boxes. However, since the classification of new object categories requires the re-initialization of the classification layer for retraining, this might not be feasible when only a small sample is available. A two-stage approach, such as Faster RCNN, which first determines whether an anchor box contains foreground objects, and then performs classification only if foreground objects have been detected. Such coarse-to-fine method does not degrade the transferability of the model. As a result, the classification in object detection will be implemented using Faster RCNN.

The aforementioned detector considered issues encountered by transfer learning when sample size is small and combined the respective advantages of the one-stage and two-stage object detection methods. The overall network architecture of LSTD and its process flow are shown in *Figure 2.5.1*. The input image is first fed into the SSD for detection where anchor boxes are set on the different layers of the feature map. The SSD detector outputs the regression parameters of the anchor boxes. SSD detector also outputs the probability of whether an anchor box contains foreground objects. Next, non-maximum suppression *(NMS)* is performed on the outputs and the filtered anchor boxes are fed into the Faster RCNN detector for object classification. Here only perform object classification and do not perform position regression as in the original method. In addition to the basic network architecture, 2 other

architectures were designed to assist transfer learning during positioning and classification. The overall architecture is shown in *Figure 2.5.2*.

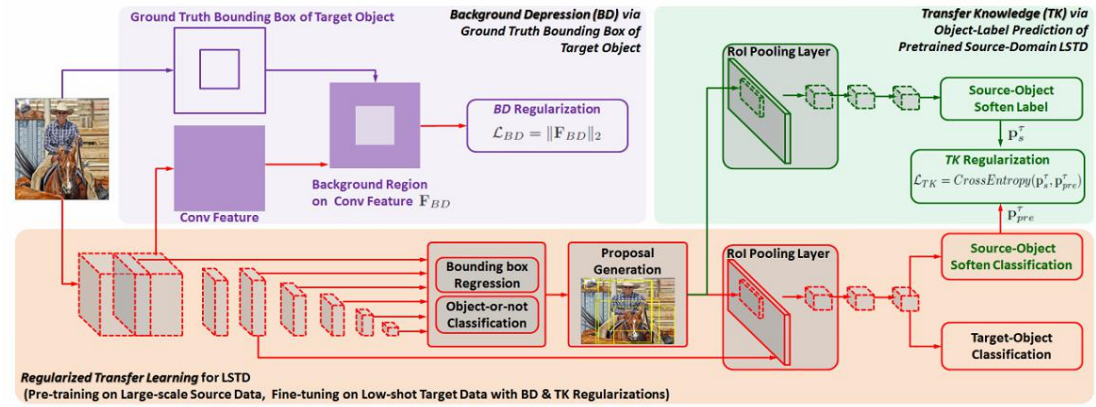


Figure 2.5.2 Overall architecture of LSTD with regularization. (Chen et al. 2018)

The region highlighted in purple in *Figure 2.5.2* is Background Depression *(BD)*. Complex background in the image can affect the training of object localization. Thus, the feature map of the LSTD detector is extracted during training, and the corresponding ground truth bounding boxes of target objects are used to provide a mask for the feature map. The masked feature map is normalized using L2 normalization and later added into the loss calculation during network training. Background depression forces the network to be more focused at the regions of the target and suppresses the response of the background in the feature map. Second, the region highlighted in green in *Figure 2.5.2* is Transfer Knowledge *(TK)*. The purpose of TK is to enhance the knowledge transfer from the pre-trained source domain. More specifically, object-label prediction of source network is used as source domain knowledge to regulate the training of target network for few-shot detection. Visualization of Background Depression regularization and Transfer Knowledge regularization are shown in both *Figure 2.5.3* and *Figure 2.5.4* respectively.

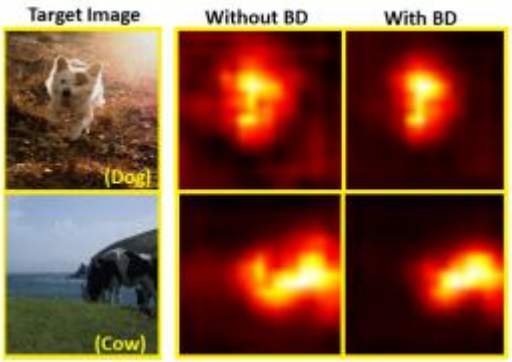


Figure 2.5.3: Background-Depression (BD) regularization. BD can reduce background noise on the feature heat-map to allow LSTD focus on target objects. (Chen et al. 2018)

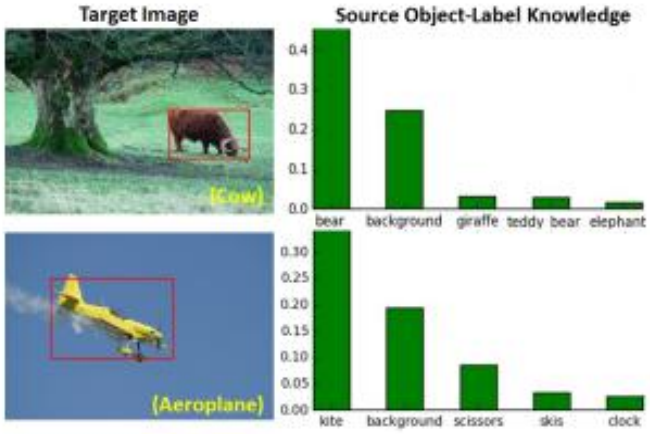


Figure 2.5.4: Transfer-Knowledge regularization (TK). TK can provide important source domain knowledge for target object proposals. (Chen et al. 2018)

2.6 PROPOSED STUDY COMPARISON

Previous works on object detector transfer learning by Li et al. (2018) on fine tuning YOLO to detect 6 types of poker cards and Wang et al. (2018) on detecting ships in complex environments by fine tuning SSD showed the success of applying transfer learning in the field of object detection. However, the transfer learning methods 6 types of poker cards and ships detection are relatively simpler than detecting the real world objects. Furthermore, these methods still require a lot annotated images for transfer learning and it may not be feasible for transfer learning to the target domain where the availability of dataset is limited. Therefore the

workability of the transfer learning on video surveillance is questionable. Low shot transfer detector framework by Chen et al. (2018) combines both Faster RCNN and SSD with background-depression and transfer-knowledge regularizations have succeeded in reducing the effort of transfer learning on object detection. The drawback of the low shot transfer detector is that once the detector is fine tuned from source domain to target domain, the performance of detecting on source domain objects is not promising.

In contrast, this project aims to implement and enhance the few shot object detection via feature re-weighting (Kang et al. 2018) which is able to perform with relatively good results even when the new class object data is 1 *(one-shot learning)*. In other words, this few shot object detection is able to perform transfer learning tasks like the previous proposed methods even though the availability of the dataset is sacred. In this way, the time spent on collecting data and model training can be drastically reduced yet a well performed object detector can be modelled to detect both base and new objects.

CHAPTER 3 SYSTEM DESIGN

3.1 DESIGN SPECIFICATIONS

Current practice in deep learning research for dealing with data scarcity issues is to first apply data augmentation to generate thousands of training images, and then use transfer learning to migrate the knowledge learning from the source domain to the target domain. The original object detection architecture used in the source domain is usually used directly without modification, only the classification layer is fine-tuned using the augmented samples. Such approaches still require many samples compared to few-shot learning. According to Li et al. (2018), when the number of training samples is less than 50, the detection performance will drop dramatically.

The aim of this study is to develop and enhance a **deep learning approach for video surveillance using small amounts of training samples**. The study will be focusing on adding data augmentation pipeline and modifying existing object detection architecture to take into account the difficulties that are encountered due to small training size. In particular, the proposed method consists of the following steps:

i. **Few-shot learning**

ii. **Data augmentation**

iii. **Re-weighting module modification**

The proposed approach is based on 3 complementing techniques, including **few-shot learning, data augmentation and re-weighting module modification**. The following subsections will cover these 3 techniques in detail.

3.1.1 FEW-SHOT LEARNING

Few-shot learning is an approach for transfer learning, where the number of training samples per category is usually 1, or a few. Compared with previous methods which required tens of thousands of images or several hundred through image synthesis or data augmentation, few-shot learning takes the challenge to learn only from a single sample image or from a few images from each category. The idea comes

from the ability of humans to learn based on previous knowledge, and through comparing the differences between a new image and the previous one where humans can learn to adjust their previous knowledge. For example, to learn to classify 4 different types of face images and be able to identify which category of a new face image belongs to among the 4, one can simply calculate the feature distances between the new face image and the 4 types of faces. A smaller distance indicates that 2 faces are similar, and thus the probability of belonging to the same class. At present, most few-shot learning studies are focusing on image classification. It is more challenging for object detection tasks since object detection requires object localization in addition to image classification. Therefore, more efforts are needed on designing the network architecture for object detection.

Meta-learning-based framework by Kang et al. (2018) is referred to get insight on the details of implementation of few-shot learning. The proposed method utilizes the low-level features learned in a network to form the mid-level features, and which in turn can be used to form the high-level feature representation. For the new category where only a small number of images are available for learning new features, the basic features learned from a rich set of other object categories, and then the small number of new images can be used to adjust the intermediate features so that they can form high level features conformed to the new category. The meta-learning-based framework is shown in *Figure 3.1.1*.

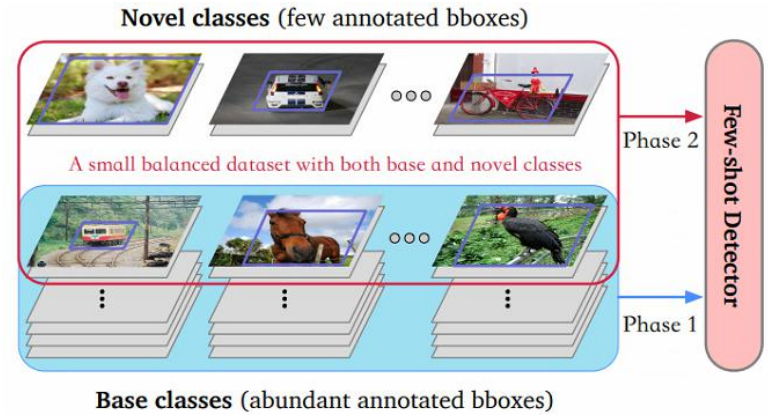


Figure 3.1.1: Meta learning based framework. (Kang et al. 2018)

First, a rich set of data is used to train the basic network, and next the network will be trained together with the new object class that contains only a small number of

samples to adjust the weight of the original intermediate features. The detailed architecture of the meta-model is shown in *Figure 3.1.2*.

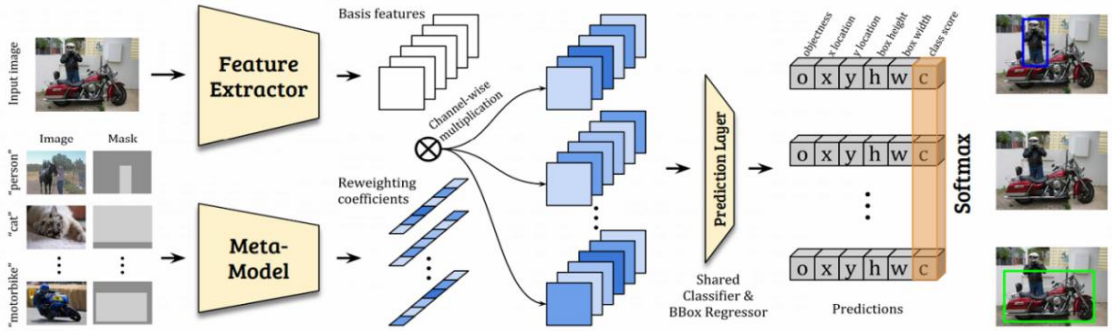


Figure 3.1.2: Architecture of the meta-model. (Kang et al. 2018)

This model architecture consists of 2 parts, as shown in the *Figure 3.1.2* above. The first part is the basis feature extractor network based on the YOLOv2 one-stage object detection method. A large set of images are used to train the YOLOv2 to extract basis to intermediate feature representations. The second part is the meta-model, which is a lightweight network that takes *N* classes' images as input, representing the *N* object classes. The network outputs *N* feature re-weighting coefficients; each is responsible for adjusting the basis features from the feature extractor network to detect the objects from the corresponding class. As shown in the *Figure 3.1.2* above, the first set of re-weighting coefficients for "person" class represents the coefficients needed for re-weighting the basis features so that they can describe high level characteristics of the person class. Therefore, the *N* feature re-weighting coefficients will be used to perform channel-wise multiplication with the basic features extracted from the feature extractor, which can be done efficiently using 1×1 convolution. The weighted features represent high level features for each of the object classes and they are fed into the prediction layer and output the *N* final prediction vectors. The final vectors include the object/no-object score, the 4 regression parameters for the anchor box, and the object class score of the anchor box. Note that the meta-model is only used to train the *N* re-weighting coefficient; it is not used in the testing phase.

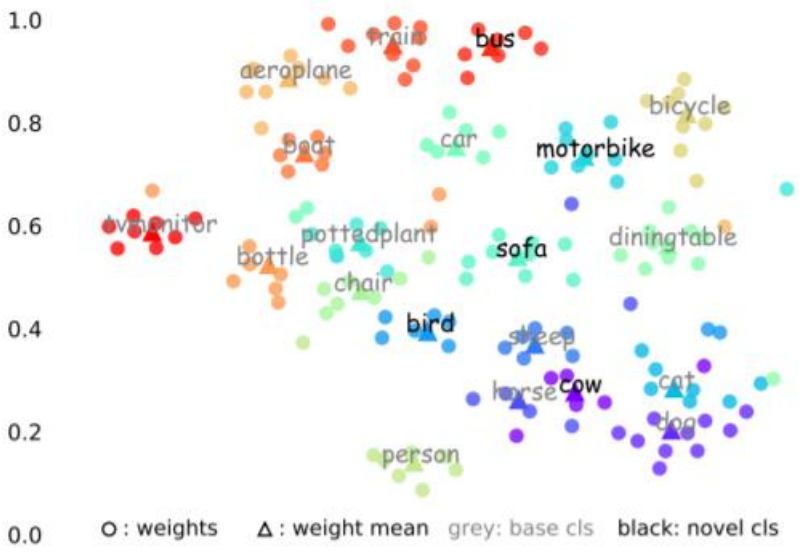


Figure 3.1.3: t-SNE visualization of re-weighting coefficients (Kang et al. 2018)

Based on the visualization of re-weighting coefficients in *Figure 3.1.3*, similar objects tend to cluster together. For example, the 4 legged animal classes *(sleep, cow, horse, cat and dog)* are close to each other at the bottom right of the visualization. Therefore, the re-weighting module is learning some useful information to re-weight the features from feature extractor.

The experimental results of the aforementioned method have concluded that object detection using a small number of samples is a feasible research direction. However, it is also clear that the performance of detection improves as the number of training samples increase. For that reason, data augmentation methods and image synthesis methods are referred to improve the performance of few-shot learning. The goal is to use the same few-shot learning architecture but to expand the number of training samples to 100 each class using data augmentation. Data augmentation is discussed in the next section.

3.1.2 DATA AUGMENTATION

Deep learning neural networks are heavy rely on big data to prevent the network from over-fitting. Data augmentation is a technique of representing a more comprehensive set of possible data points to reduce the error distance between training and validation or even testing set (Shorten & Khoshgoftaar 2019). By applying data augmentation, quality and size of datasets can be enhanced for training.

For instance, Hojjat et al. (2017) proposed a way to address the lack of training data or the uneven training data issue in image classification. Their method not only can increase the amount of training data but also increase data diversity, and has obtained good results using AlexNet and GoogleNet. The method is based on radial transformation, which generates new training images with different radial variations from the original images. The method converts Cartesian coordinates of an image to Polar coordinates at a specified centre points and produces a new image (see *Figure 3.1.4*). Multiple new images can be generated using different centre points. As shown in *Figure 3.1.5*, the image on the right is the result of radial transform of the image on the left at the centre of the purple circle.

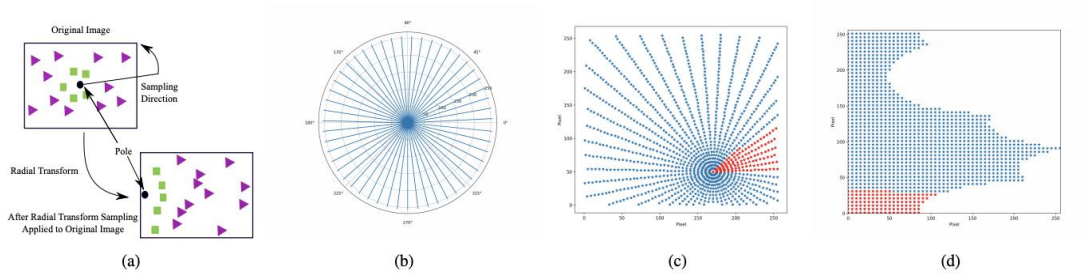


Figure 3.1.4: Radial transformation process. (Hojjat et al. 2017)

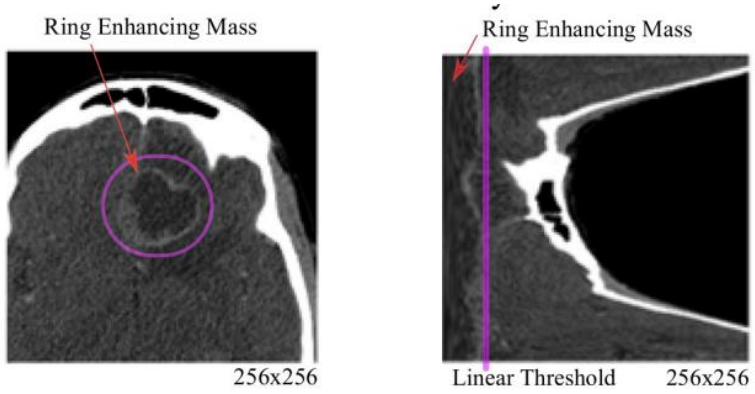


Figure 3.1.5: Radial transform example. (Hojjat et al. 2017)

As for object detection, previous work on data augmentation for object detection (Zoph, B et al. 2019) is referred to provide some insights for this project. Their method increases the mAP on COCO dataset by more than 2.3 mAP whereas improves the mAP on PASCAL VOC by more than 2.7 mAP with the best augmentation policy. The data augmentation operations performed are include colour operations *(equalize, brightness, contrast),* geometric operations *(rotate, shearing,*

translation) and bounding box operations *(flipping)* which only distort the content within the bounding box. The augmentation policy is defined as an unordered set of *K* sub-policies. During training, only 1 of the *K* sub-policies will be selected randomly and applied to the image. Each sub-policy consists of *N* image transformations. In this paper, *K* is defined as 5 whereas *N* is defined as 2, which means there are 5 sub-policies and each policy there are 2 image transformation operations as shown in *Figure 3.1.6.* Each transformation operation is defined by 2 values which are the probability of the operation and the magnitude of the operation.

Based on the results from previous works with data augmentation, the performance of the models are improved after applying data augmentation. Therefore, different data augmentation operations *(shearing, flipping, translation, rotation, scaling, colour jitter)* will be performed separately and sequentially in this project to enhance the amount of data so that the performance of few shot detector can be improved.

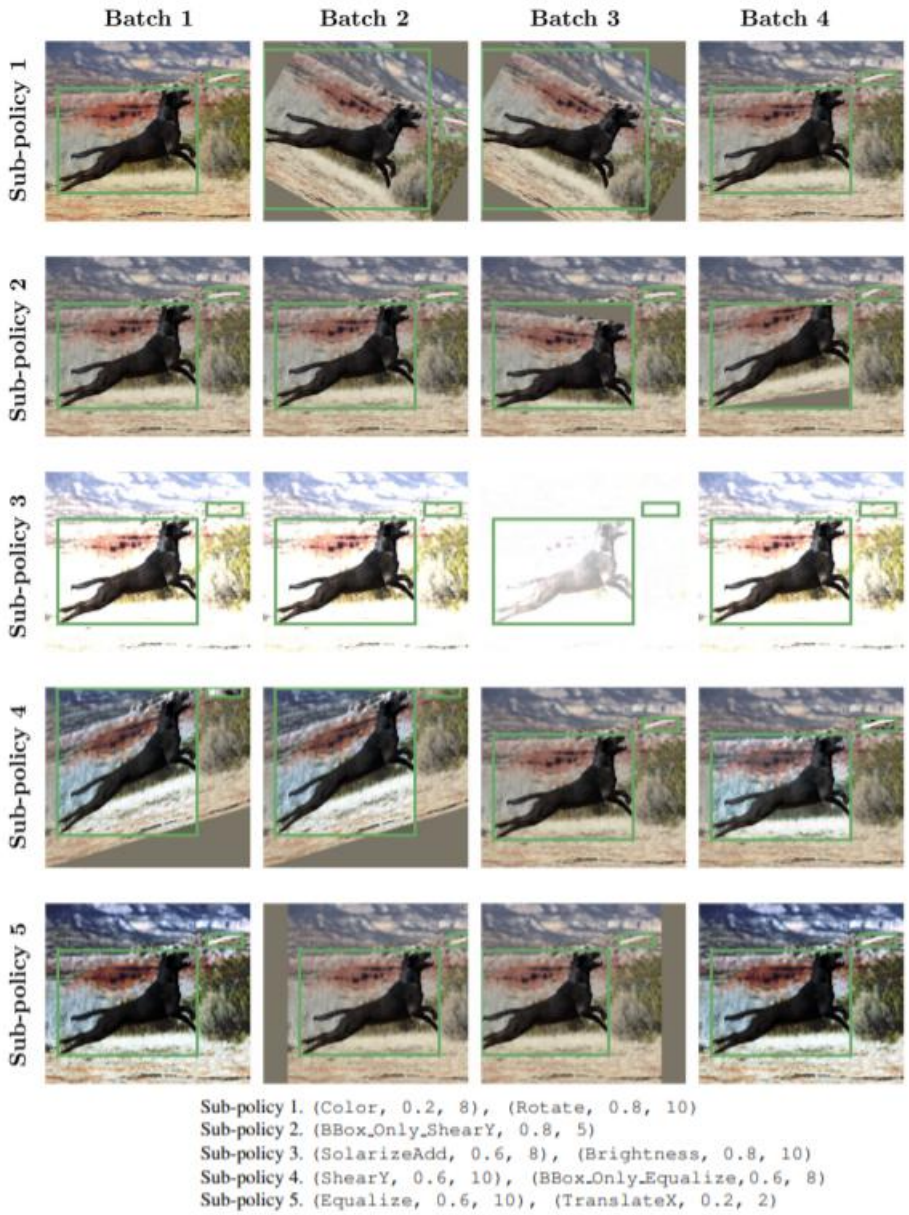


Figure 3.1.6: Augmentation sub policies. Each operation is defined by probability and magnitude of the operation.(Zoph, B et al. 2019)

3.1.3 RE-WEIGHTING MODULE MODIFICATION

The shape of the re-weighting coefficients from the few shot detector framework is defined as *N x C x H x W* where *N* is number of re-weighting classes, *C* is the number of channels, *H* and *W* are the height and width of the re-weighting module. The original shape of *H* and *W* of the re-weighting coefficients is *1 x 1* which is generated by applying global maximum operation on the final receptive field of the re-weighting module. Re-weighting coefficients will be used to re-weighting the features from feature extractor by applying channel-wise multiplication as shown in

Figure 3.1.7. The authors (Kang et al. 2018) claimed that the global features of the new image object can be captured by designing the module in this way.

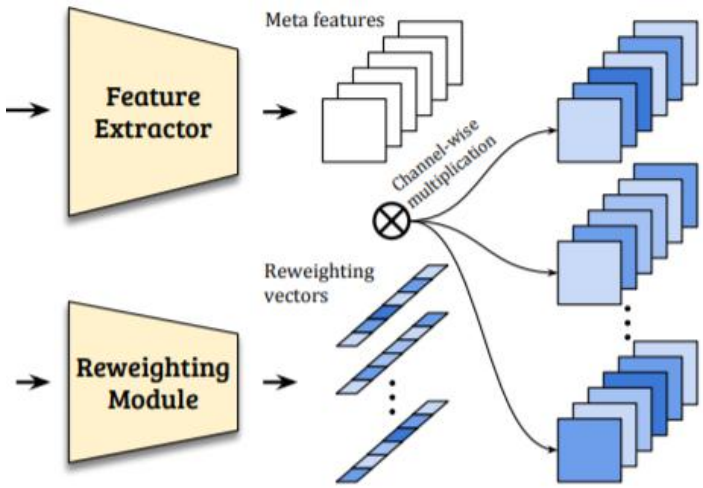


Figure 3.1.7: Re-weighting process.

However having a global maximum layer at the final layer may lead to the loss of class-specific information since the receptive field size is shrank to only the size of *1 x 1*. Shrinking the receptive field of the re-weighting module into *1 x 1* may deteriorate the performance of the detector especially in few shot learning where the fine tuning of the new object dataset is limited. Therefore instead of having *1 x 1* re-weighting coefficients, the re-weighting module is redesigned to generate re-weighting coefficients which have the same size as the meta features from the feature extractor as shown in *Figure 3.1.8* below. The re-weighting vectors will be re-weighting the meta features by using pixel-wise multiplication instead of channel-wise multiplication. By modifying the re-weighting module in this way, it is believed that more class specific information can be retained for re-weighting the features from feature extractor to detect new class objects.

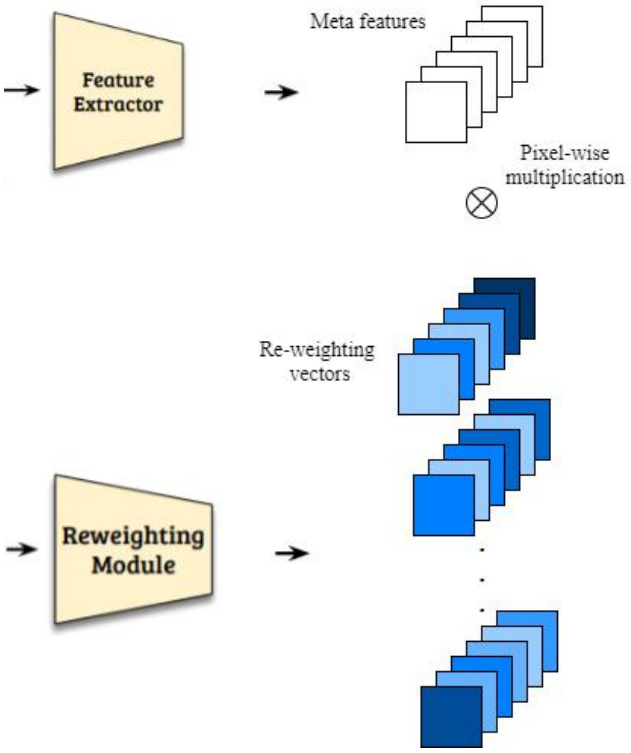


Figure 3.1.8: Redesigned re-weighting vectors.

3.2 DESIGN OVERVIEW

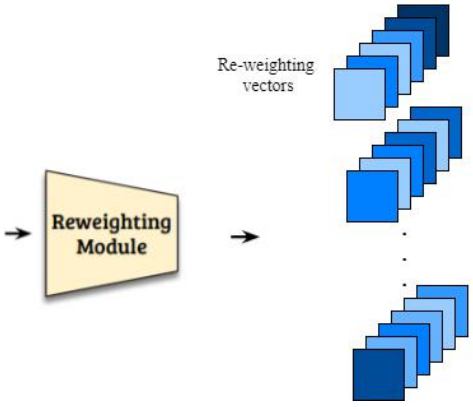
At first, a large amount of base classes *(15 classes)* of PASCAL VOC dataset will be used to train the underlying network *(YOLOv2 as the base detector with a meta-model as feature extractor)* to detect base class objects. The neural network architecture design of the base detector is shown in *Figure 3.2.1* below.

layer	filters	size	input		output
0 conv	32	3 x 3 / 1	416 x 416 x 3	->	416 x 416 x 32
1 max		2 x 2 / 2	416 x 416 x 32	->	208 x 208 x 32
2 conv	64	3 x 3 / 1	208 x 208 x 32	->	208 x 208 x 64
3 max		2 x 2 / 2	208 x 208 x 64	->	104 x 104 x 64
4 conv	128	3 x 3 / 1	104 x 104 x 64	->	104 x 104 x 128
5 conv	64	1 x 1 / 1	104 x 104 x 128	->	104 x 104 x 64
6 conv	128	3 x 3 / 1	104 x 104 x 64	->	104 x 104 x 128
7 max		2 x 2 / 2	104 x 104 x 128	->	52 x 52 x 128
8 conv	256	3 x 3 / 1	52 x 52 x 128	->	52 x 52 x 256
9 conv	128	1 x 1 / 1	52 x 52 x 256	->	52 x 52 x 128
10 conv	256	3 x 3 / 1	52 x 52 x 128	->	52 x 52 x 256
11 max		2 x 2 / 2	52 x 52 x 256	->	26 x 26 x 256
12 conv	512	3 x 3 / 1	26 x 26 x 256	->	26 x 26 x 512
13 conv	256	1 x 1 / 1	26 x 26 x 512	->	26 x 26 x 256
14 conv	512	3 x 3 / 1	26 x 26 x 256	->	26 x 26 x 512
15 conv	256	1 x 1 / 1	26 x 26 x 512	->	26 x 26 x 256
16 conv	512	3 x 3 / 1	26 x 26 x 256	->	26 x 26 x 512
17 max		2 x 2 / 2	26 x 26 x 512	->	13 x 13 x 512
18 conv	1024	3 x 3 / 1	13 x 13 x 512	->	13 x 13 x1024
19 conv	512	1 x 1 / 1	13 x 13 x1024	->	13 x 13 x 512
20 conv	1024	3 x 3 / 1	13 x 13 x 512	->	13 x 13 x1024
21 conv	512	1 x 1 / 1	13 x 13 x1024	->	13 x 13 x 512
22 conv	1024	3 x 3 / 1	13 x 13 x 512	->	13 x 13 x1024
23 conv	1024	3 x 3 / 1	13 x 13 x1024	->	13 x 13 x1024
24 conv	1024	3 x 3 / 1	13 x 13 x1024	->	13 x 13 x1024
25 route	16				
26 conv	64	1 x 1 / 1	26 x 26 x 512	->	26 x 26 x 64
27 reorg		/ 2	26 x 26 x 64	->	13 x 13 x 256
28 route	27 24				
29 conv	1024	3 x 3 / 1	13 x 13 x1280	->	13 x 13 x1024
30 dconv	1024	1 x 1 / 1	13 x 13 x1024	->	13 x 13 x1024
31 conv	30	1 x 1 / 1	13 x 13 x1024	->	13 x 13 x 30
32 detection					

Figure 3.2.1: Base detector (YOLOv2) architecture design.

Based on *Figure 3.2.1* above, *layer 30* of the base feature detector is a dynamic convolutional layer. This layer will be re-weighted by the re-weighting coefficients from features generated by the re-weighting module.

In the case of video surveillance, when the scene is shifted to a new scene or the detection requirement is changed to detect new objects, a small number of annotated samples from the new object classes are required to fine tune the network, and this is done with the *N* class re-weighting coefficients generated from the modified re-weighting module. The architecture design of the modified re-weighting module is shown in *Figure 3.2.2* below.




```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   4   ->   416 x 416 x  32
    1 max          2 x 2 / 2   416 x 416 x  32   ->   208 x 208 x  32
    2 conv     64  3 x 3 / 1   208 x 208 x  32   ->   208 x 208 x  64
    3 max          2 x 2 / 2   208 x 208 x  64   ->   104 x 104 x  64
    4 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128
    5 max          2 x 2 / 2   104 x 104 x 128   ->    52 x  52 x 128
    6 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256
    7 max          2 x 2 / 2    52 x  52 x 256   ->    26 x  26 x 256
    8 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
    9 max          2 x 2 / 2    26 x  26 x 512   ->    13 x  13 x 512
   10 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
```

Figure 3.2.2: Architecture design of modified re-weighting module.

From *Figure 3.2.2* above, the input channel for *layer 0* is 4 channels instead of 3 channels because input channels consist 3 channels for RBG *(red, blue, green)* colour channels and extra 1 channel for masked object channel *(grey-scale)* for better fine tuning result for objects in the image. The dimension output of the last convolutional layer *(layer 10)* is matched with the input dimension of the dynamic convolutional layer of base detector shown in *Figure 3.2.1*. This is because the output vectors

(13x13x1024) from *Figure 3.2.2* will be used to re-weight the input vectors of *layer30* of the feature extractor in *Figure 3.2.1.*

As discussed in the data augmentation section, before the limited new object images are fed to fine tune the object detector, varying annotated image augmentation techniques will be applied on each new annotated image to produce higher quantity and higher diversity dataset in order to increase the performance of the object detector. The basic data augmentations applied include random translation, random rotation, random scaling, random shearing, random horizontal flip and colour jitter. Other than basic data augmentation, combined augmentation which combines few basic augmentations mentioned on an image is also applied.

Table 3.2.1: Data augmentation settings.

Operations	**Settings**
Horizontal Flip	*Probability = 0.5*
Scaling	*Scale factor = 1+ random.uniform((-0.2, 0.2))*
Rotation	*Rotate angle = random.uniform((-20, 20))*
Shearing	*Shear factor = random.uniform((-0.2, 0.2))*
Translation	*Translate factor x, y = random.uniform((-0.2, 0.2))*
Colour Jitter	*Brightness, Contrast, Sharpness, Colour = np.random.rand()*
Combined	*Horizontal Flip + Scaling + Translation + Colour Jitter*

As shown in *Table 3.2.1* above, there are total 7 data augmentation operations will be applied on each image before fine tuning. The operations include 6 basic operations *(horizontal flip, scaling. rotation, shearing, translation and colour jitter)* and 1 combined operations where *horizontal flip, scaling and translation and colour jitter* operation will be applied sequentially on the image. Hence, for each image there are 7 new images will be augmented through the operations discussed above. The augmented samples are shown in the following pages.

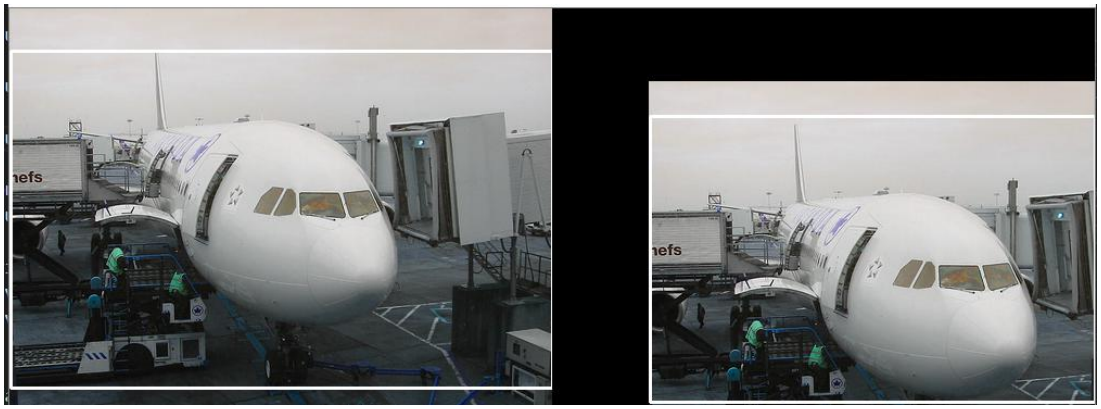


Figure 3.2.3: Random translation augmentation
(left:original image, right:augmented image).

Figure 3.2.4: Random rotation augmentation
(left:original image, right:augmented image).

Figure 3.2.5: Random scaling augmentation
(left:original image, right:augmented image).

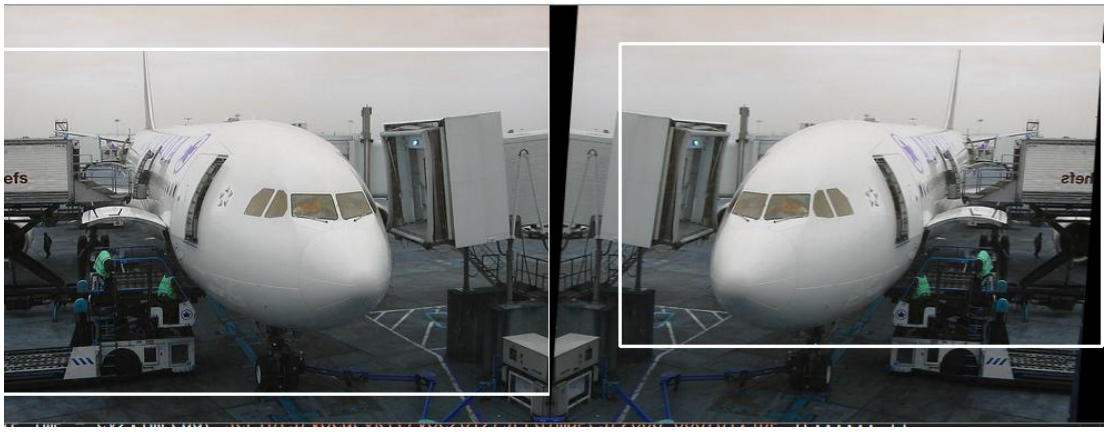


Figure 3.2.6: Random shearing augmentation
(left:original image, right:augmented image).

Figure 3.2.7: Random horizontal flip augmentation
(left:original image, right:augmented image).

Figure 3.2.8: Random colour jitter augmentation
(left:original image, right:augmented image).

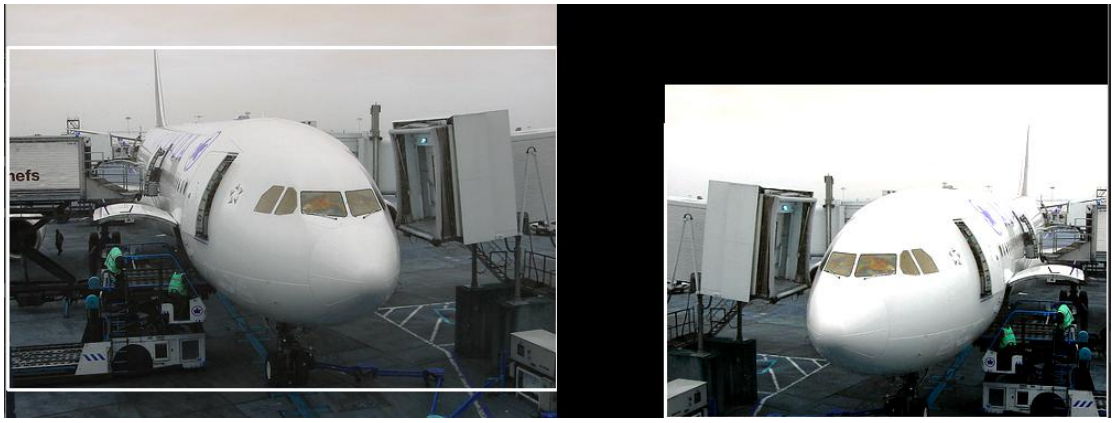


Figure 3.2.9: Combined (flipping, scaling, translation, colour jitter) augmentation (left:original image, right:augmented image)

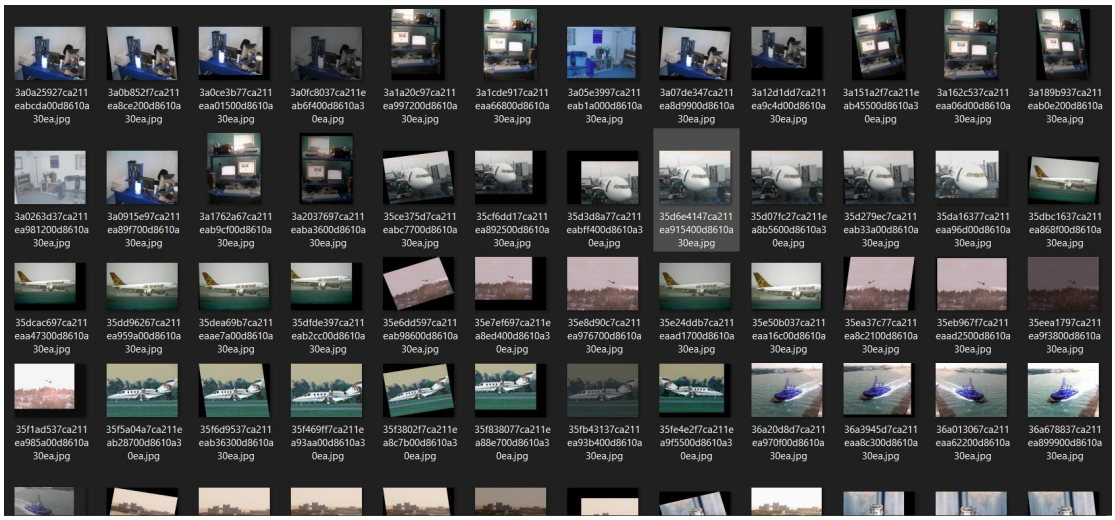


Figure 3.2.10: Augmented image sets for 2 shot learning.

After fine tuning, the object detector is able to detect new class objects with the combination of features from the feature extractor *(edges and corners)* and re-weighting coefficients from the re-weighting module. The prior knowledge in the source domain from the feature extractor is able to be integrated with the re-weighting module to detect new target domain objects.

PASCAL VOC Dataset 2007 and 2012 are chosen as the dataset for this project as it contains an adequate amount of object classes for detection and performance analysis. PASCAL VOC dataset contains 20 different object classes as shown in the *Table 3.2.2* below. Training and validation sets from VOC 2007 and 2012 will be combined and used for training whereas the testing set VOC 2007 will be used for testing and evaluation.

Table 3.2.2: PASCAL VOC Dataset classes.

Classes of PASCAL VOC Dataset	*Aeroplane, Bicycle, Bird, Boat, Bottle, Bus, Car, Cat, Chair, Cow, Dining Table, Dog, Horse, Motorbike, Person, Potted Plant, Sheep, Sofa, Train, TV Monitor*

The whole training process is divided into 2 phases, which are base training and meta training *(fine tuning)*. Before the beginning of the training process, 20 classes of PASCAL VOC dataset are split into 2 classes, 1 set for base classes and 1 set for new classes. The splitting is done according to the table shown in *Table 3.2.3* below.

Table 3.2.3: Splitting of PASCAL VOC classes into base classes and new classes.

Base classes	*Aeroplane, Bicycle, Boat, Bottle, Car, Cat, Chair, Dining Table, Dog, Horse, Person, Potted Plant, Sheep, Train, TV Monitor*
New classes	*Bird, Bus, Cow, Motorbike, Sofa*

Pre-trained weight on ImageNet will be used as the weight initialization for the detector. Firstly, abundant data from base classes will be provided for base training so that the network is able to learn the general features for objects whereas new classes will be used for meta training to fine tune the whole object detector framework to learn the class-specific features from new classes' objects with limited dataset provided. Then the extracted features from both base feature extractor and re-weighting coefficients are combined so that the model is able to detect the new classes.

Fine tuning of the re-weighting module is done by few shot learning. New class objects for fine tuning the meta model extractor are preprocessed into *k* shot where *k* indicates the number of images will be provided for a particular class for fine tuning. For example, 3 shots will be indicating 3 images from each new class will be input for the re-weighting module fine tuning. In this project, 1, 2, 3, 5 and 10 shot learnings for fine tuning are conducted with data augmentation to enhance the performance of the object detector under situations of data scarcity.

The project is divided into 3 experiment sets. 1st experiment is to investigate the **effect of the data augmentation** on the performance of few shot detector on new

class objects, whereas 2nd experiment is to investigate the **effect of modified re-weighting module** on the performance of few shot detector on new class objects and 3rd experiment is to investigate the effect of **combination of both data augmentation and the modified re-weighting module** on the performance of few shot detector on new class objects.

CHAPTER 4 RESULT ANALYSIS AND TOOLS

4.1 PERFORMANCE ANALYSIS

Note: *The data supplied for the baselines below is either 1, 2, 3, 5 or 10 image(s) per class.*

Baselines:

YOLO-joint	*: Straightforward train on both base and new classes together.*
YOLO-ft	*: Without re-weighting module, 2 phases training without fully converges.*
YOLO-ft-full	*: Without re-weighting module, 2 phases training until fully converges.*
LSTD(YOLO)	*: LSTD framework (YOLOv2 as backbone network) with transfer learning without fully converges.*
LSTD(YOLO)-full	*: LSTD framework (YOLOv2 as backbone network) transfer learning until fully converges.*

Table 4.1.1: Few-shot detection performance (mAP) on PASCAL VOC dataset. Comparisons between baselines and modified versions.

Method / # Shots	**1**	**2**	**3**	**5**	**10**
YOLO-joint	0.0	0.0	1.8	1.8	1.8
YOLO-ft	3.2	6.5	6.4	7.5	12.3
YOLO-ft-full	6.6	10.7	12.5	24.8	38.6
LSTD(YOLO)	6.9	9.2	7.4	12.2	11.6
LSTD(YOLO)-full	8.2	11.0	12.4	29.1	38.5
Few shot detector via feature re-weighting (original)	14.8	15.5	26.7	33.9	47.2

After data augmentation	**15.7**	**17.2**	**28.3**	**36.2**	**49.1**
Modified re-weighting module	**15.5**	**16.9**	**27.8**	**35.4**	**48.7**
Data augmentation + modified re-weighting module	**16.0**	**17.6**	**28.9**	**36.9**	**49.8**

Table 4.1.2: Improvement of the performance compared to original method. (Calculate by averaging the improvements of 1, 2, 3, 5, 10 shot mAP respectively)

Methods	***Improvements***
Data augmentation	*1.68 mAP*
Modified re-weighting module	*1.24 mAP*
Data augmentation + modified re-weighting module	*2.22 mAP*

Based on the *mAP* shown in *Table 4.1.1* and *Table 4.1.2*, the performance of the object detector is improved when compared to the original detector *(few shot detector via feature re-weighting)* after the adding of data augmentation pipeline and modification of re-weighting module. It is worth noting that the effect of the modified re-weighting module on the improvement is lower than the effect of data augmentation. The data augmentation improved the *mAP* with an average of ***1.68 mAP*** whereas the modified re-weighting module improved the *mAP* with an average of ***1.24 mAP***. However, the combination of both data augmentation and modified re-weighting module improved the performance of the detector by around ***2.22 mAP*** which is higher than the improvements by data augmentation or modified re-weighting module alone. The next table shows the detailed average precision on each new class objects in different shot learnings *(1, 2, 3, 5, 10 shot learning)*.

Table 4.1.3: Detection performance (AP) for base categories on PASCAL VOC dataset.

# Shots		New classes					mean
		bird	**bus**	**cow**	**mbike**	**sofa**	
1	YOLO-joint	0.0	0.0	0.0	0.0	0.0	0.0
	YOLO-ft	6.8	0.0	9.1	0.0	0.0	3.2
	YOLO-ft-full	11.4	17.6	3.8	0.0	0.0	6.6
	LSTD(YOLO)	12.0	17.8	4.6	0.0	0.1	6.9
	LSTD(YOLO)-full	13.4	21.4	6.3	0.0	0.0	8.2
	Few shot detector via feature re-weighting (original)	13.5	10.6	31.5	13.8	4.3	14.8
	After data augmentation	**13.9**	**12.1**	**33.1**	**14.3**	**5.1**	**15.7**
	Modified re-weighting module	**14.0**	**11.8**	**32.9**	**14.2**	**4.8**	**15.5**
	Data augmentation + modified re-weighting module	**14.5**	**12.3**	**33.4**	**14.7**	**5.3**	**16.0**
2	YOLO-joint	0.0	0.0	0.0	0.0	0.0	0.0
	YOLO-ft	11.5	5.8	7.6	0.1	7.5	6.5
	YOLO-ft-full	16.6	9.7	12.4	0.1	14.5	10.7
	LSTD(YOLO)	12.3	10.1	14.6	0.1	8.9	9.2
	LSTD(YOLO)-full	17.3	12.5	8.6	0.2	16.5	11.0
	Few shot detector via feature re-weighting (original)	21.2	12.0	16.8	17.9	9.6	15.5
	After data augmentation	**22.8**	**13.9**	**18.2**	**19.1**	**12.1**	**17.2**
	Modified re-weighting module	**22.6**	**13.5**	**17.9**	**18.8**	**11.9**	**16.9**

	Data augmentation + modified re-weighting module	**23.1**	**14.3**	**18.5**	**19.7**	**12.5**	**17.6**
3	YOLO-joint	0.0	0.0	0.0	0.0	9.1	1.8
	YOLO-ft	10.9	5.5	15.3	0.2	0.1	6.4
	YOLO-ft-full	21.0	22.0	19.1	0.5	0.0	12.5
	LSTD(YOLO)	12.3	7.1	17.7	0.1	0.0	7.5
	LSTD(YOLO)-full	23.1	22.6	15.9	0.4	0.0	12.4
	Few shot detector via feature re-weighting (original)	26.1	19.1	40.7	20.4	27.1	26.7
	After data augmentation	**27.9**	**20.3**	**42.8**	**21.8**	**28.5**	**28.3**
	Modified re-weighting module	**27.3**	**20.0**	**41.9**	**21.5**	**28.1**	**27.8**
	Data augmentation + modified re-weighting module	**28.1**	**21.0**	**43.2**	**22.8**	**29.3**	**28.9**
5	YOLO-joint	0.0	0.0	0.0	0.0	9.1	1.8
	YOLO-ft	11.6	7.1	10.7	2.1	6.0	7.5
	YOLO-ft-full	20.2	20.0	22.4	36.4	24.8	24.8
	LSTD(YOLO)	12.9	8.1	13.6	16.1	10.2	12.2
	LSTD(YOLO)-full	24.1	30.2	24.0	40.0	25.6	29.1
	Few shot detector via feature re-weighting (original)	31.5	21.1	39.8	40.0	37.0	33.9
	After data augmentation	**33.0**	**24.3**	**42.1**	**42.2**	**39.3**	**36.2**
	Modified re-weighting module	**32.4**	**22.5**	**41.6**	**41.9**	**38.3**	**35.4**

	Data augmentation + modified re-weighting module	**33.3**	**25.2**	**43.0**	**43.2**	**39.7**	**36.9**
10	YOLO-joint	0.0	0.0	0.0	0.0	9.1	1.8
	YOLO-ft	11.4	28.4	8.9	4.8	7.8	12.2
	YOLO-ft-full	22.3	53.9	32.9	40.8	43.2	38.6
	LSTD(YOLO)	11.3	32.2	5.6	1.3	7.7	11.6
	LSTD(YOLO)-full	22.8	52.5	31.3	45.6	40.3	38.5
	Few shot detector via feature re-weighting (original)	30.0	62.7	43.2	60.6	40.6	47.2
	After data augmentation	**33.8**	**63.3**	**44.5**	**62.0**	**41.9**	**49.1**
	Modified re-weighting module	**33.2**	**62.9**	**44.2**	**61.5**	**41.7**	**48.7**
	Data augmentation + modified re-weighting module	**34.0**	**64.4**	**45.4**	**62.6**	**42.3**	**49.8**

4.2 DETECTION VISUALIZATION (NEW CLASSES)

The following pages will show the new class object detection before and after model fine tuning. The fine-tuned model for detecting the new class object images is the model with **data augmentation pipeline added** and **modified re-weighting module**.

Figure 4.2.1: Bus image detection (above: before fine tuning, below: after fine tuning). No bus is detected before fine tuning.

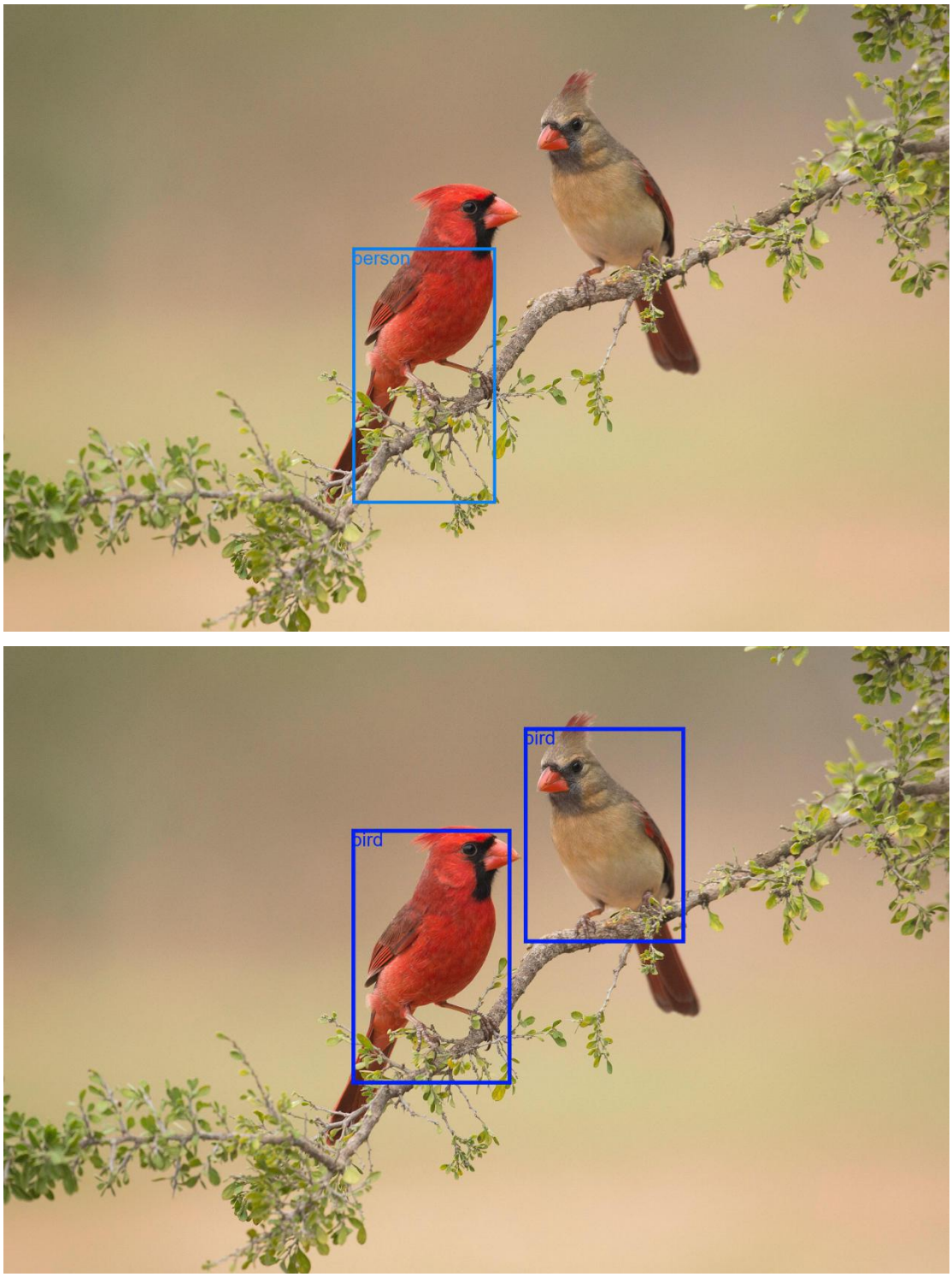


Figure 4.2.2: Bird image detection (above: before fine tuning, below: after fine tuning). Before fine tuning the model classify the bird as person.

Figure 4.2.3: Cow image detection (above: before fine tuning, below: after fine tuning). No cow is detected before fine tuning.

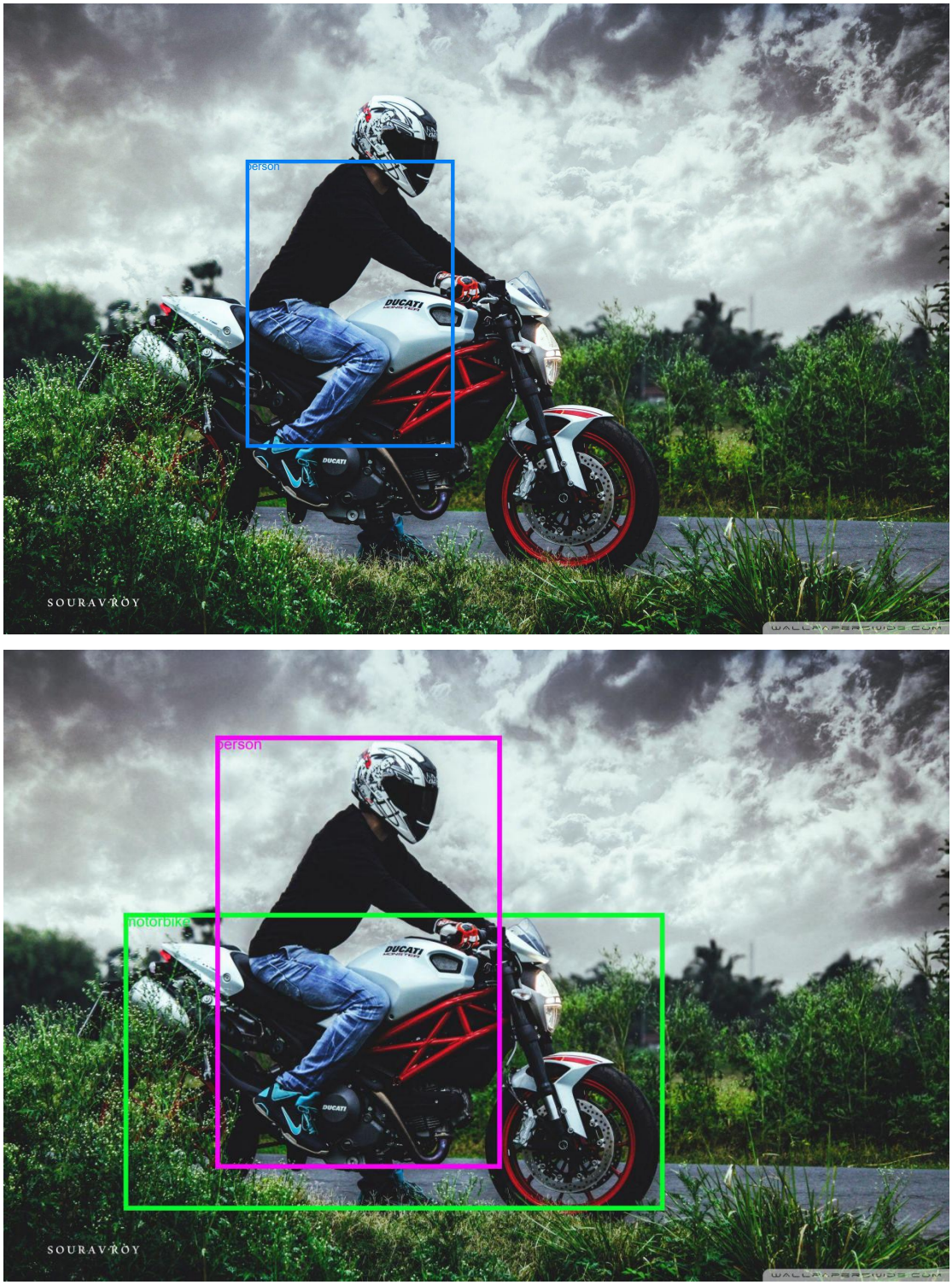


Figure 4.2.4: Motorbike image detection (above: before fine tuning, below: after fine tuning). Before fine tuning only person class is detected.

Figure 4.2.5: Sofa image detection (above: before fine tuning, below: after fine tuning). Before fine tuning, part of sofa is detected as chair because chair has the most similar characteristic with sofa.

4.3 RESEARCH TOOLS

Table 4.3.1: Research tools list.

Hardware	2 x GeForce RTX 2080 Ti Graphics Card, AMD Ryzen Threadripper 2950X 16-Core
Software	Python 3.6.9, Pytorch 1.0.1, Visual Studio Code
Datasets	PASCAL VOC 2007 & 2012

CHAPTER 5 CONCLUSION

5.1 DISCUSSIONS

In summary, this project improved the performance of the existing few-shot detector via feature re-weighting through **data augmentation and re-weighting module modification** for video surveillance. With this, objectives of the project *(results higher performance through data augmentation and re-weighting module modification)* are achieved. The experiment results show that the performance of the few shot detector is able to be improved with data augmentation ***(improved 1.68 mAP)***, modified re-weighting module ***(improved 1.24 mAP)*** and the combination of data augmentation and modified re-weighting module ***(improved 2.22 mAP)*** on detecting new class objects *(bird, bus, cow, motorbike, sofa)* from PASCAL VOC Dataset. The reasons behind the improvements are due to the **increases of data diversity** through augmentation operations and the **class specific information loss of the modified re-weighting module is minimized**. Compared to previous few shot detector framework where the re-weighting coefficients are represented in *1 x 1* via global maximum layer, modification is made on the size of re-weighting vectors so that the size of re-weighting coefficients is same as the size of receptive field of feature vectors from feature extractor in order to prevent massive information loss. Once the information loss is minimized, more class-specific features can be re-weighted with the features extracted from feature extractor in order to detect new class objects with higher performance.

5.2 IMPLEMENTATION ISSUES AND CHALLENGES

Throughout the implementation of this project, several issues and challenges are encountered. Firstly, the **computing resources** required by the project is not supported by my own laptop and the computing resources provided by FICT is limited. As the result, debugging is performed on my own laptop and model training is performed on the machine provided by FICT as the computing resources for debugging is not as large as model training. Furthermore, accessing of FICT machine

is based on a first come first serve basis, therefore some time is wasted on waiting for the availability of the machine.

Moreover, **version incompatibility** of version updating of Pytorch and Python is also the issues encountered. The referenced code was written in the previous Pytorch and Python version. Due to version incompatibilities, the referenced codes are not able to be compiled successfully. Moreover, even though the codes are compiled successfully, the output may be different from the one shown by the owner. As a result, extra efforts and time have to be spent on fixing the version incompatibility issues.

5.3 NOVELTIES AND CONTRIBUTIONS

The main contribution of this project is delivering an improved few shot detector via feature re-weighting through the adding of a data augmentation pipeline and modified re-weighting module for solving some challenges *(illumination level changes, limited data for detecting new targets)* faced in video surveillance. However, the contribution of the project is not restricted within the video surveillance field, but it is also applicable in other fields where the training data is difficult to obtain such as medical imaging and manufacturing industry.

5.4 FUTURE WORKS

In future works, different improvement approaches will be attempted to further enhance the performance of few shot detector. For example, the improvements will be attempted on designing a more compact and meaningful loss function specifically for object detection few shot learning. Nevertheless, the data augmentation techniques applied in this project are relatively simple and the diversity of the augmented data is limited. Hence, data augmentation with Generative Adversarial Network *(GAN)* will be attempted to create a high amount of novel and expressive samples for each class to fine tune the model. Other than that, improvement will be sought through by modifying the whole architecture for better new class features learning in fine tuning.

BIBLIOGRAPHY

Alex, K., Ilya, S. & Geoffrey, E.H. 2012, *ImageNet Classification with Deep Convolutional Neural Networks* [online]. Available from <https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf> [Accessed 3 August 2019].

Amira, B.M. & Ezzeddine, Z. 2018, "Abnormal behavior recognition for intelligent video surveillance systems: A review", *Expert Systems With Applications*, vol 91, pp. 480-491. Available from <https://www.sciencedirect.com/science/article/pii/S0957417417306334> [Accessed 3 August 2019].

Brendan J. & Le H.S. 2018, *Precision-Recall versus Accuracy and the Role of Large Dataset* [online]. Available from <https://cpb-us-w2.wpmucdn.com/sites.wustl.edu/dist/c/1120/files/2017/11/aaai-2018-9-xkpq6j.pdf> [Accessed 3 August 2019].

Cs231n.github.io. (2019). CS231n Convolutional Neural Networks for Visual Recognition. [online] Available from: <http://cs231n.github.io/transfer-learning/> [Accessed 14 Nov. 2019].

Chen H., Wang Y.L., Wang G.Y. & Qiao Y. 2018, *LSTD: A Low-Shot Transfer Detector for Object Detection* [online]. Available from: <https://arxiv.org/pdf/1803.01529.pdf>. [Accessed 4 August 2019].

He K.M., Georgia G., Piotr D. & Ross G. 2017, *Mask R-CNN* [online]. Available from: <https://arxiv.org/pdf/1703.06870.pdf>. [Accessed 4 August 2019].

Hojjat S., Shahrokh V., Timothy D. & Joseph B. 2017, "Image augmentation using radial transform for training deep neural networks", *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Alberta, Canada, pp. 1-5, 2018. Available from: <https://arxiv.org/abs/1708.04347>. [Accessed 4 August 2019].

Hulstaert, L. (2019). *A Beginner's Guide to Object Detection.* [online] DataCamp Community. Available from: <https://www.datacamp.com/community/tutorials/object-detection-guide>[Accessed 12 Aug. 2019].

Joseph R., Santosh D., Ross G. & Ali F. 2015, *You Only Look Once: Unified, Real-Time Object Detection*. Available from: <https://arxiv.org/abs/1506.02640>. [Accessed 4 August 2019].

Kang B., Liu Z., Wang X., Fisher Y., Feng J. & Trevor D., "Few-shot object detection via feature reweighting", no., pp. 1-12, 2018. Available from: <https://arxiv.org/abs/1812.01866>[Accessed 12 Aug. 2019].

Kun F., Zhang T.F., Zhang Y., Yan M.L., Chang Z.H., Zhang Z.Y. & Sun X. 2019, "Meta-SSD: Towards Fast Adaptation for Few-Shot Object Detection With Meta-Learning", *IEEE Access Volume 7,* 12 June 2019. Available from: <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8735792>.[Accessed 14 November 2019].

Lawrence J. F. 2017, *Effective Physical Security* [online], 5th edn. Butterworth-Heinemann, Oxford. Available from: <https://www.sciencedirect.com/book/9780124158924/effective-physical-security>[Accessed 12 Aug. 2019].

Li G.Q., Song Z.Y. & Qiang F. 2018, A New Method of Image Detection for Small Datasets under the Framework of YOLO Network, *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC 2018),* IEEE, Chongqing, China, 12-14 Oct 2018. Available from: <https://ieeexplore.ieee.org/document/8577214>. [Accessed 4 August 2019].

Lisa T. & Jude S., n.d. "Transfer Learning". Available from: <http://ftp.cs.wisc.edu/machine-learning/shavlik-group/torrey.handbook09.pdf>. [Accessed 11 November 2019].

Liu W., Dragomir A., Dumitru E., Christian S., Scott R., Chen Y.F. & Alexander C.B. 2016, *SSD: Single Shot MultiBox Detector* [online]. Available from: <https://arxiv.org/pdf/1512.02325.pdf>. [Accessed 4 August 2019].

Liu Y.X., Yang Y., Aijun S., Peng J.G. & Liu H.W. 2019, Intelligent monitoring of indoor surveillance video based on deep learning, *2019 21st International Conference on Advanced Communication Technology (ICACT),* IEEE, PyeongChang Kwangwoon_Do, Korea (South), 17-20 Feb 2019. Available from: <https://ieeexplore.ieee.org/document/8701964/>. [Accessed 4 August 2019].

Noah S., Thomas S., Dmitry G. & Rangachar K. 2008, How effective is human video surveillance performance? *2008, 19th International Conference on Pattern Recognition*, Tampa, FL, USA, 8-11 Dec 2018. Available from: <https://ieeexplore.ieee.org/document/4761655>. [Accessed 4 August 2019].

Pedro D., n.d., "A Few Useful Things to Know about Machine Learning". Available from: <http://www.astro.caltech.edu/~george/ay122/cacm12.pdf>. [Accessed 4 August 2019].

Ren S.Q., He K.M., Ross G. & Sun J. 2015, *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks* [online]. Available from: <https://arxiv.org/abs/1506.01497>. [Accessed 4 August 2019].

Ross G. 2015, *Fast R-CNN* [online]. Available from: <https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf>. [Accessed 4 August 2019].

Ross G., Jeff D., Trevor D. & Jitendra M. 2015, *Region-based Convolutional Networks for Accurate Object Detection and Segmentation* [online]. Available from: <http://islab.ulsan.ac.kr/files/announcement/513/rcnn_pami.pdf> [Accessed 4 August 2019].

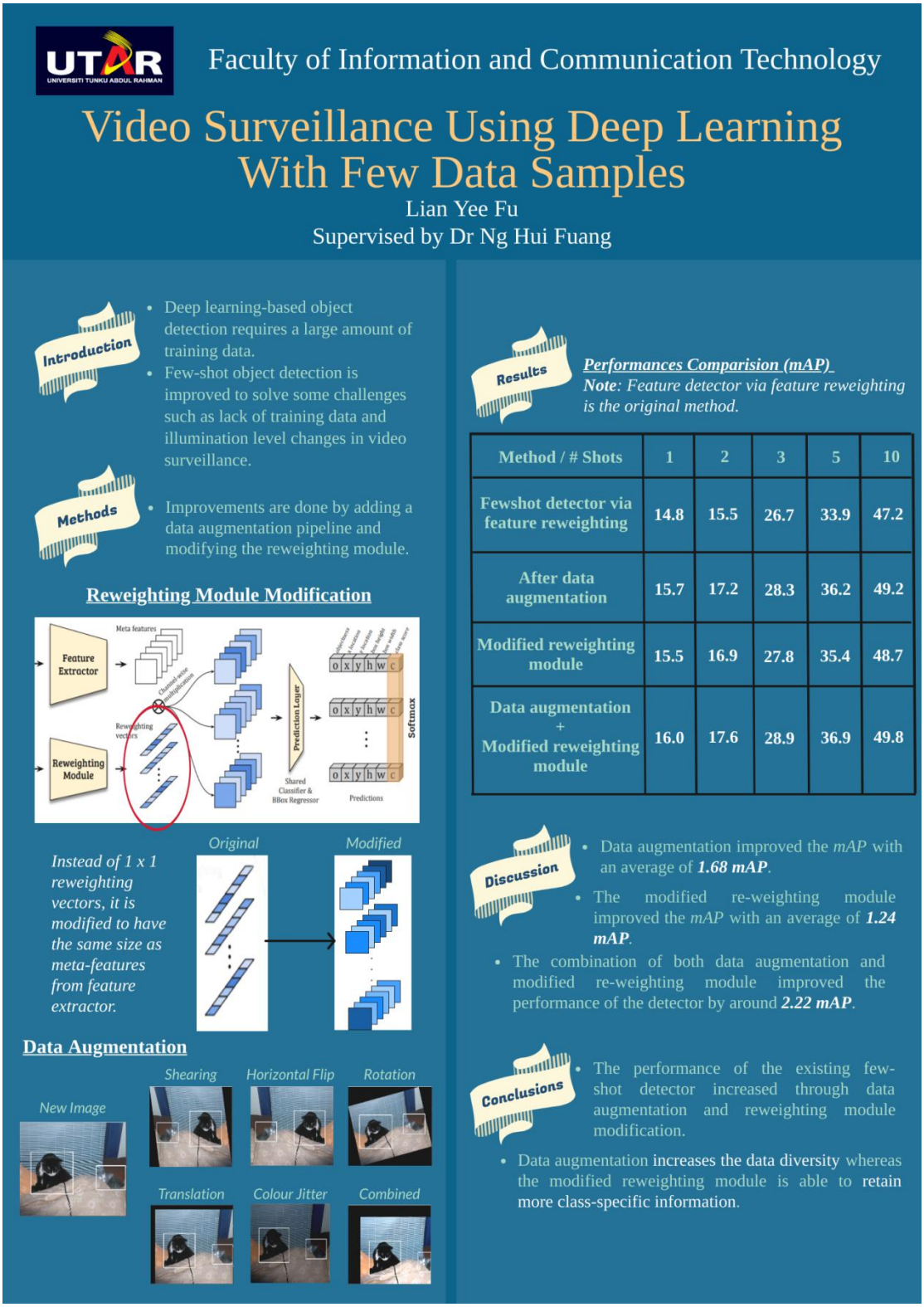
Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). Available from: <https://doi.org/10.1186/s40537-019-0197-0> [Accessed 4 January 2019].

Software.intel.com. (2019). Use Transfer Learning For Efficient Deep Learning Training On Intel® Xeon® Processors. [online] Available from: <https://software.intel.com/en-us/articles/use-transfer-learning-for-efficient-deep-learning-training-on-intel-xeon-processors> [Accessed 14 Nov. 2019].

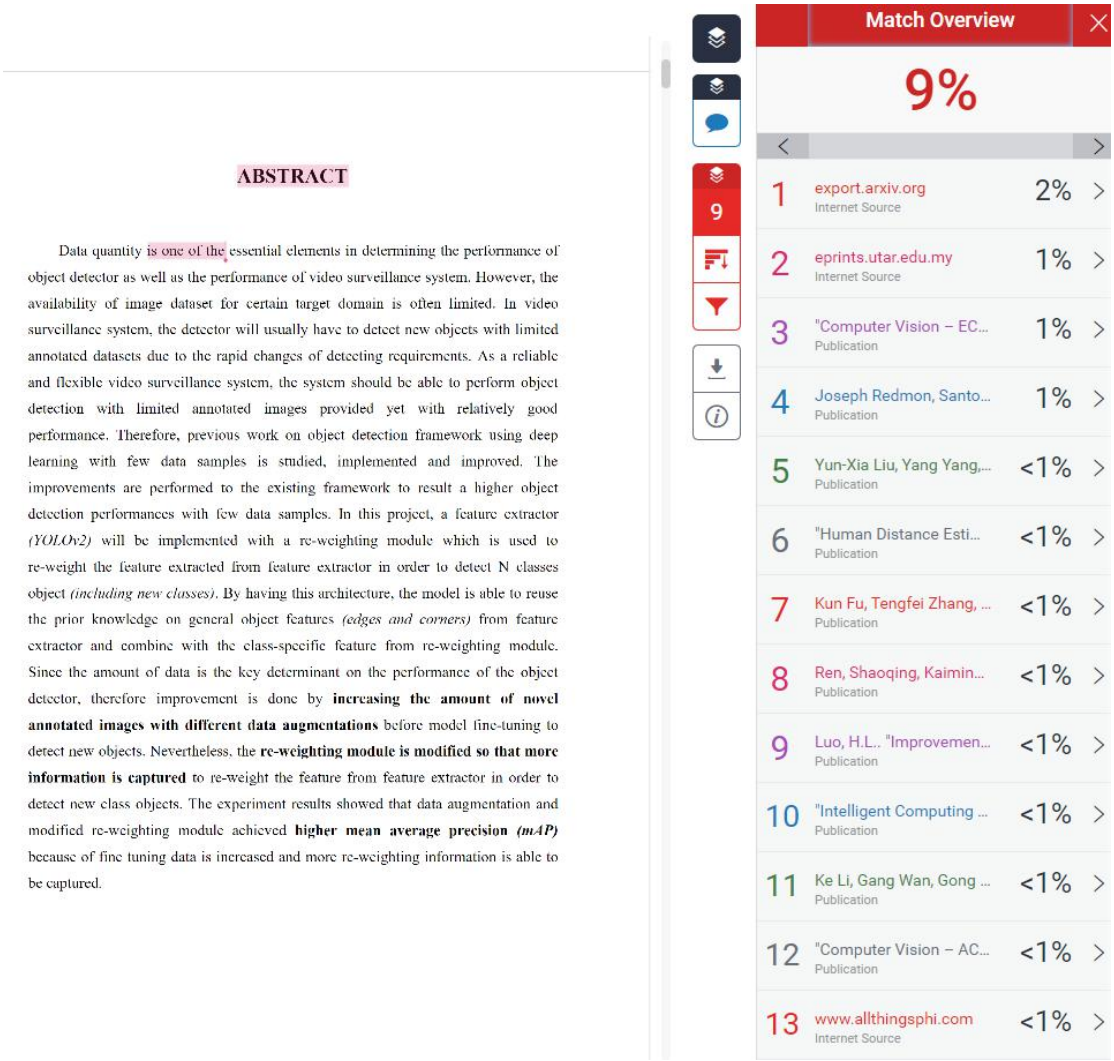
Wang Y., Wang C. & Zhang H. 2018, "Combining a single shot multibox detector with transfer learning for ship detection using sentinel-1 SAR images," *SAR in Big Data Era: Models, Methods and Applications (BIGSARDATA),* IEEE, Beijing, China, 13-14 Nov 2017. Available from: <https://ieeexplore.ieee.org/document/8124924>. [Accessed 4 August 2019].

Zoph, B., Cubuk E., Ghiasi Golnaz, Lin T.Y., Jonathon S., Le Q. (2019). *Learning Data Augmentation Strategies for Object Detection*. Available from: <https://arxiv.org/pdf/1906.11172.pdf>. [Accessed 4 August 2019].

POSTER


UTAR
UNIVERSITI TUNKU ABDUL RAHMAN
Faculty of Information and Communication Technology
Video Surveillance Using Deep Learning With Few Data Samples
Lian Yee Fu
Supervised by Dr Ng Hui Fuang
Introduction
Deep learning-based object detection requires a large amount of training data.
Few-shot object detection is improved to solve some challenges such as lack of training data and illumination level changes in video surveillance.
Methods
Improvements are done by adding a data augmentation pipeline and modifying the reweighting module.
Reweighting Module Modification
Feature Extractor
Meta features
Reweighting vectors
Reweighting Module
Prediction Logits
Softmax
Shared Classifier & BBox Regressor
Predictions
Instead of 1 x 1 reweighting vectors, it is modified to have the same size as meta-features from feature extractor.
Original
Modified
Data Augmentation
New Image
Shearing
Horizontal Flip
Rotation
Translation
Colour Jitter
Combined
Results
Performances Comparision (mAP)
Note: Feature detector via feature reweighting is the original method.
Method / # Shots | 1 | 2 | 3 | 5 | 10
Fewshot detector via feature reweighting | 14.8 | 15.5 | 26.7 | 33.9 | 47.2
After data augmentation | 15.7 | 17.2 | 28.3 | 36.2 | 49.2
Modified reweighting module | 15.5 | 16.9 | 27.8 | 35.4 | 48.7
Data augmentation + Modified reweighting module | 16.0 | 17.6 | 28.9 | 36.9 | 49.8
Discussion
Data augmentation improved the mAP with an average of 1.68 mAP.
The modified re-weighting module improved the mAP with an average of 1.24 mAP.
The combination of both data augmentation and modified re-weighting module improved the performance of the detector by around 2.22 mAP.
Conclusions
The performance of the existing few-shot detector increased through data augmentation and reweighting module modification.
Data augmentation increases the data diversity whereas the modified reweighting module is able to retain more class-specific information.

PLAGIARISM CEHCKING



ABSTRACT

Data quantity is one of the essential elements in determining the performance of object detector as well as the performance of video surveillance system. However, the availability of image dataset for certain target domain is often limited. In video surveillance system, the detector will usually have to detect new objects with limited annotated datasets due to the rapid changes of detecting requirements. As a reliable and flexible video surveillance system, the system should be able to perform object detection with limited annotated images provided yet with relatively good performance. Therefore, previous work on object detection framework using deep learning with few data samples is studied, implemented and improved. The improvements are performed to the existing framework to result a higher object detection performances with few data samples. In this project, a feature extractor *(YOLOv2)* will be implemented with a re-weighting module which is used to re-weight the feature extracted from feature extractor in order to detect N classes object *(including new classes)*. By having this architecture, the model is able to reuse the prior knowledge on general object features *(edges and corners)* from feature extractor and combine with the class-specific feature from re-weighting module. Since the amount of data is the key determinant on the performance of the object detector, therefore improvement is done by **increasing the amount of novel annotated images with different data augmentations** before model fine-tuning to detect new objects. Nevertheless, the **re-weighting module is modified so that more information is captured** to re-weight the feature from feature extractor in order to detect new class objects. The experiment results showed that data augmentation and modified re-weighting module achieved **higher mean average precision *(mAP)*** because of fine tuning data is increased and more re-weighting information is able to be captured.

Match Overview

9%

1	export.arxiv.org Internet Source	2%
2	eprints.utar.edu.my Internet Source	1%
3	"Computer Vision – EC... Publication	1%
4	Joseph Redmon, Santo... Publication	1%
5	Yun-Xia Liu, Yang Yang,... Publication	<1%
6	"Human Distance Esti... Publication	<1%
7	Kun Fu, Tengfei Zhang, ... Publication	<1%
8	Ren, Shaoqing, Kaimin... Publication	<1%
9	Luo, H.L.. "Improvemen... Publication	<1%
10	"Intelligent Computing ... Publication	<1%
11	Ke Li, Gang Wan, Gong ... Publication	<1%
12	"Computer Vision – AC... Publication	<1%
13	www.allthingsphi.com Internet Source	<1%

FYP2 Turnitin

ORIGINALITY REPORT

9%	4%	6%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	export.arxiv.org Internet Source	2%
2	eprints.utar.edu.my Internet Source	1%
3	"Computer Vision – ECCV 2016", Springer Science and Business Media LLC, 2016 Publication	1%
4	Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016 Publication	1%
5	Yun-Xia Liu, Yang Yang, Aijun Shi, Peng Jigang, Liu Haowei. "Intelligent monitoring of indoor surveillance video based on deep learning", 2019 21st International Conference on Advanced Communication Technology (ICACT), 2019 Publication	<1%

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Lian Yee Fu
ID Number(s)	16ACB05136
Programme / Course	BCS (Hons) Computer Science
Title of Final Year Project	Video Surveillance Using Deep Learning With Few Data Samples

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 9 % **Similarity by source** Internet Sources: 4 % Publications: 6 % Student Papers: 0 %	
Number of individual sources listed of more than 3% similarity: ____	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Ng Hui Fuang

Date: 21 / 4 / 2020

Signature of Co-Supervisor

Name: _____

Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	1605136
Student Name	Lian Yee Fu
Supervisor Name	Dr. Ng Hui Fuang

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. (Signature of Student) Date: 21/04/2020	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. (Signature of Supervisor) Date: 21 / 4 / 2020