

# **Household Waste Segregation Using Intelligent Vision System**

**Teh Junjie**


**A project report submitted in partial fulfillment of the  
requirements for the award of Bachelor of Engineering  
(Honours) Mechatronics Engineering**

**Lee Kong Chian Faculty of Engineering and Science  
Universiti Tunku Abdul Rahman**

**January 2020**

**DECLARATION**

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature :  \_\_\_\_\_

Name : Teh Junjie \_\_\_\_\_

ID No. : 1607063 \_\_\_\_\_

Date : 1/10/2020 \_\_\_\_\_

**APPROVAL FOR SUBMISSION**

I certify that this project report entitled “**Household Waste Segregation Using Intelligent Vision System**” was prepared by **Teh Junjie** has met the required standard for submission in partial fulfillment of the requirements for the award of Bachelor of Engineering (Honours) Mechatronics Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature :   
\_\_\_\_\_

Supervisor : Mr. Chai Tong Yuen  
\_\_\_\_\_

Date : 1/10/2020  
\_\_\_\_\_

Signature : \_\_\_\_\_

Co-Supervisor : \_\_\_\_\_

Date : \_\_\_\_\_

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© Year, Name of candidate. All right reserved.

## ABSTRACT

Waste segregation is a process to separate the wastes accordingly into their specific categories. Each waste goes into its category at the point of dumping or collection. Waste segregation is an important step to improve the effectiveness of waste management. Normally, this process is done manually by human hand picking in Malaysia. The waste generated in Malaysia is increasing gradually every year, the environment will be polluted if the waste is not managed properly and it will also endanger human's health. Thus, an intelligent vision system is proposed to improve the efficiency of waste segregation. According to literature review, CNN appeared to be a promising way to develop an intelligent vision system for waste segregation. However, the CNN models take a long time to train and predict. In this project, a study on the types of household waste generated in Malaysia is conducted. The identified wastes are being divided into 6 different classes for this experiment which are glass, metal, cardboard, plastic, paper, and other wastes. Next, several famous CNN architectures such as VGG-19 and Inception V3 are studied and experimented in this project to benchmark with the state-of-the-arts. Besides that, we have proposed a novel method which is the hybrid CNN-ELM model. The hybrid model aimed to improve the efficiency of the system in real-time application. The training and predicted time of the hybrid CNN-ELM model is 720 times faster than the conventional CNN architectures. All the models built in this project are tested by two different publicly available databases which are the Trash-Net dataset and the 0528qsw dataset. The test accuracy of VGG-19 is the best among the others which scored above 90% in both datasets. However, the InceptionV3+ELM model can achieve an accuracy of 90% in the 0528qsw dataset. The proposed hybrid CNN-ELM model has higher computational efficiency compared to the conventional deep learning methods as the time taken for the model to compute is only 5.4s whereas the VGG-19 model takes 2954s.

## TABLE OF CONTENTS

|  |                                  |             |
|--|----------------------------------|-------------|
| <b>DECLARATION</b>                     |                                  | <b>ii</b>   |
| <b>APPROVAL FOR SUBMISSION</b>         |                                  | <b>iii</b>  |
| <b>ABSTRACT</b>                        |                                  | <b>v</b>    |
| <b>TABLE OF CONTENTS</b>               |                                  | <b>vi</b>   |
| <b>LIST OF TABLES</b>                  |                                  | <b>viii</b> |
| <b>LIST OF FIGURES</b>                 |                                  | <b>ix</b>   |
| <b>LIST OF SYMBOLS / ABBREVIATIONS</b> |                                  | <b>xi</b>   |
| <b>LIST OF APPENDICES</b>              |                                  | <b>xii</b>  |
| <br><b>CHAPTER</b>                     |                                  |             |
| <b>1</b>                               | <b>INTRODUCTION</b>              | <b>1</b>    |
| 1.1                                    | Background                       | 1           |
| 1.2                                    | Problem Statement                | 1           |
| 1.3                                    | Motivation                       | 3           |
| 1.4                                    | Aim and Objectives               | 3           |
| <b>2</b>                               | <b>LITERATURE REVIEW</b>         | <b>4</b>    |
| 2.1                                    | Household Waste                  | 4           |
| 2.2                                    | Related Work                     | 5           |
| <b>3</b>                               | <b>METHODOLOGY AND WORK PLAN</b> | <b>8</b>    |
| 3.1                                    | Introduction                     | 8           |
| 3.2                                    | Data Pre-processing              | 10          |
| 3.3                                    | Overview of CNN                  | 11          |
| 3.3.1                                  | Convolutional Layer              | 12          |
| 3.3.2                                  | Pooling Layers                   | 12          |
| 3.3.3                                  | Stride and Padding               | 13          |
| 3.3.4                                  | ReLU (Rectified Linear Units)    | 15          |
| 3.4                                    | Classifier                       | 15          |
| 3.4.1                                  | Fully Connected Layer            | 16          |
| 3.4.2                                  | Softmax Function                 | 16          |

|          |       |                                |           |
|----------|-------|--------------------------------|-----------|
|          | 3.4.3 | Extreme Learning Machine (ELM) | 17        |
|          | 3.5   | VGG                            | 20        |
|          | 3.6   | Inception V3                   | 21        |
|          | 3.7   | Transfer Learning Approach     | 22        |
|          | 3.8   | Evaluation Metrics             | 24        |
|          | 3.9   | Datasets                       | 25        |
|          | 3.9.1 | TrashNet dataset               | 25        |
|          | 3.9.2 | 0528qsw dataset                | 26        |
| <b>4</b> |       | <b>Results and Discussion</b>  | <b>28</b> |
|          | 4.1   | Model Evaluation               | 28        |
|          | 4.1.1 | Results on TrashNet Dataset    | 28        |
|          | 4.1.2 | Results on 0528qsw Dataset     | 30        |
|          | 4.2   | Confusion Matrix               | 31        |
|          | 4.3   | Precision, Recall and F1 score | 32        |
|          | 4.4   | Discussion                     | 33        |
| <b>5</b> |       | <b>Conclusion</b>              | <b>36</b> |
|          | 5.1   | Conclusion and Future work     | 36        |
|          | 5.2   | Future Recommendation          | 37        |
|          |       | <b>REFERENCES</b>              | <b>38</b> |
|          |       | <b>APPENDICES</b>              | <b>42</b> |

**LIST OF TABLES**

|   |    |
|---|----|
| Table 2.1: Summary of Literature Review                                 | 7  |
| Table 3.1: Gantt Chart  | 8  |
| Table 3.2: Architecture of Proposed VGG-19 Model                        | 20 |
| Table 3.3: Process of Applying ELM                                      | 21 |
| Table 3.4: Pseudocode of implementing CNN-ELM                           | 24 |
| Table 3.5: Sample of image from TrashNet dataset                        | 25 |
| Table 3.6: Number of images in each class of TrashNet dataset           | 26 |
| Table 3.6: Sample of image from 0528qsw dataset                         | 26 |
| Table 3.7: Types of Household Waste in 0528qsw dataset                  | 27 |
| Table 4.1 Classification Result on Trash-Net dataset                    | 28 |
| Table 4.2 Classification Result on 0528qsw dataset                      | 30 |
| Table 4.3 Evaluation Matrix of top 4 trained model in Trash-Net dataset | 33 |
| Table 4.4 Evaluation Matrix of top 4 trained model in 0528qsw dataset   | 33 |
| Table 4.5: Benchmark Result with the State of Arts                      | 34 |



## LIST OF FIGURES

|   |    |
|---|----|
| Figure 3.1: Detail flow chart of the process to develop an intelligent vision system                                  | 10 |
| Figure 3.2: Example of Image Augmentation (Image Augmentation   Pytorch Image Augmentation 2019)                      | 11 |
| Figure 3.3: Overview of CNN architecture (Saha 12/16/2018)  | 11 |
| Figure 3.4: Maxpool with a 2x2 filter and a stride of 2 (Deshpande 6/26/2019)   | 12 |
| Figure 3.5: Stride =1 (Deshpande 6/26/2019)   | 13 |
| Figure 3.6: Stride =2 (Deshpande 6/26/2019)   | 13 |
| Figure 3.7: Zero padding of 2 added to the input volume of 32x32x3(Deshpande 6/26/2019)                               | 14 |
| Figure 3.8: Graph of ReLU activation function (Dr. Sebastian Raschka 2020)  | 15 |
| Figure 3.9: (Fully Connected Layers in Convolutional Neural Networks: The Complete Guide - MissingLink.ai 3/22/2020). | 16 |
| Figure 3.10: Softmax Regression Layer (Rizwan 5/25/2018)  | 17 |
| Figure 3.11: Structure of extreme learning machine (Yang et al. 2019)   | 17 |
| Figure 3.12: InceptionV3 architecture (Raj 5/30/2018)   | 22 |
| Figure 3.13: Strategies applied for transfer learning approach (Marcelino 10/23/2018)                                 | 22 |
| Figure 3.14: CNN-ELM Hybrid Model (Andreas Kölsch, Muhammad Zeshan Afzal Markus Ebbecke Marcus Liwicki 2017)          | 23 |
| Figure 4.1: Training Loss and Accuracy of VGG-19 pre-trained model for 100 epochs (Trash-Net Dataset)                 | 29 |
| Figure 4.2: Training Loss and Accuracy of VGG-19 pre-trained model for 100 epochs (0528qsw Dataset)                   | 31 |
| Figure 4.3: Confusion matrix of VGG-19 model trained on TrashNet dataset  | 31 |

- Figure 4.4: Confusion matrix of InceptionV3 +ELM model trained on TrashNet dataset 32
- Figure 4.5: Representative of the waste item which have low classification accuracy 35

**LIST OF SYMBOLS / ABBREVIATIONS**

|      |   |
|------|---|
| CNN  | Convolutional Neural Network                |
| ELM  | Extreme Learning Machine                    |
| FN   | False Negative                              |
| FP   | False Positive                              |
| GPU  | Graphic Processor Unit                      |
| KNN  | K-Nearest Neighbours                        |
| PET  | Polyethylene Terephthalate                  |
| ReLU | Rectified Linear Unit                       |
| SVM  | Support Vector Machine                      |
| SW   | Solid Waste Management and Public Cleansing |
| TN   | True Negative                               |
| TP   | True Positive                               |

**LIST OF APPENDICES**

|   |    |
|---|----|
| APPENDIX A: Python Code for Data Pre-processing                         | 35 |
| APPENDIX B: Python Code to Load and Train the VGG-19 model              | 36 |
| APPENDIX C: Python Code to Load and Train the Inception-V3 model        | 37 |
| APPENDIX D: Python Code to Create and Train the Hybrid CNN-ELM<br>model | 38 |

## CHAPTER 1

### INTRODUCTION

#### 1.1 Background

An intelligent vision system is one of the most rapidly growing technologies and it is being widely used in a variety of industrial situations to monitor and control the manufacturing process. A lot of industries have already successfully employed the vision system for a long while. A vision system is used as a feature extraction system to extract features for further work. An Intelligent vision system is known as the hybrid system which has a combination of vision and artificial intelligence. The features extracted through the vision system are further processed using artificial intelligence. Artificial Intelligence is used to do the calculation and processing of the features from the image captured by the camera for piece part recognition, orientation, tracking and so forth. The extracted features are then trained by artificial intelligence which can be used later in objects classification.

Household waste is also known as the domestic waste or residual waste. Some of these wastes will endanger human health. Hazardous waste is something that will cause harm to the surroundings if they are not handled carefully and dispose of properly. Meanwhile non-hazardous wastes are referring to the wastes that can be composted or recycled such as food waste, paper, PET bottles and so forth. Household waste is now a major issue all around the world as the number of household waste produced in Malaysia.

#### 1.2 Problem Statement

Malaysia is overflowing with waste nowadays. The amount of waste generated last year is regularly about 1.17kg each day. This figure is doubled in the year of 2005, whereas the waste generation is about 0.8kg each day by an individual according to the Solid Waste Management and Public Cleansing Corporation (SW Corp). The growing trend of the waste will affect the sustainability of the environment and cause pollution. This might endanger the health of residents in Malaysia.

According to the Natural Resource and Environmental Board- NREB, the most common waste treatment method used in Malaysia is the landfilling method whereby more than 70% of waste is disposed using this method. However, it is not efficient to use landfilling methods for waste decomposition because there will be the needs of a lot of land space as the waste takes a long time to completely decompose in the land space. For example, it takes about 450 years for plastic to decompose. Moreover, they are hard and expensive to maintain and operate as the waste inside the landfill area will release toxic, methane (CH<sub>4</sub>) and CO<sub>2</sub> to the air if they are not being processed properly.

Next, incineration is the second favourite choice of Malaysia used to decompose waste. This method is always being criticized because during the incineration process, greenhouse gases will emit and cause the air pollution problem. Besides that, the construction and operation costs of the incineration tank is also expensive. Moreover, there will be extra cost charges when the incineration method is used in Malaysia because Malaysia is located near the equator and experiences a tropical climate, thus the humidity will cause the waste to have more moisture content and extra energy is needed to burn the moisture waste.

The alternative method that is better in handling waste is recycling. 3R is referring to reduce, reuse and recycle. By practicing 3R, it will be able to help in reducing the waste generated in Malaysia as recycling is all about reprocessing the waste material into a new product, thus the waste can be reused. There are a lot of benefits obtained from recycling process such as reducing environmental damage, saving energy, resource conservation and so forth. However, the recycling rate in Malaysia is still very low compared to other developed countries such as Japan and Germany. The recycling rate in Japan and Germany is above 50%, meanwhile Malaysia is only at a mere 28%.

Most of the people in Malaysia do not practice recycling because it is cheaper to dump than recycle. Waste segregation is important to sort out the recyclable materials among the residual wastes instead of just throwing them away as residue. Waste segregation is a prerequisite for any feasible recycling activity to proceed successfully and economically. If those recyclable wastes are not properly separated and mixed with organic waste, it will be hard to process and it might not be suitable to recycle.

### **1.3 Motivation**

The awareness of caring for the environment in Malaysia is still low. There are not many people practicing waste segregation in Malaysia. This might be due to the common thinking of handling waste as something which is below an acceptable level of social dignity. Government has actively held up several campaigns and recycling programs. The studies showed low participation rates from residents via those campaigns and programs carried out. Besides that, the government has put in a lot of efforts to encourage residents in Malaysia to participate in recycling by setting up more recycling bins in residential areas. However, misuse of recycle bins is found. According to the study, 40-60% of the contents found in the recycle bin are non-recyclable items. The mixed waste will reduce the efficiency of the recycling process because the recyclable waste will be contaminated and hard to process. The studies showed that it is not a wise decision to allocate the resources to train and educate the public about waste segregation through campaigns. Moreover, waste segregation is a repetitive and tedious work, thus not many people love to do that.

Hence, our motivation is to create an intelligent vision system that is capable of segregating the household waste automatically in order to improve the efficiency of recycling. Throughout the improvement of recycling rate, it will benefit the economy and the environment.

### **1.4 Aim and Objectives**

In this project, we aim to propose an intelligent vision system which can automatize the waste segregation process. The system is aimed to increase the efficiency of recycling and reduce the pollution in Malaysia. As the system is planned to be used in real-time application, thus the accuracy and the processing time of the system are important. We will devise a learning-based algorithm to develop a system which can achieve high classification accuracy and have a high computational efficiency.

The objectives of this project are shown below:

- i) To study and identify the types of household waste in Malaysia
- ii) Devise a learning-based algorithm to segregate types of waste automatically.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Household Waste

Japan is known as one of the developed countries which are taking serious on waste management and able to turn the wastes into resources effectively or dispose them appropriately. The wastes in Japan are divided into a few groups which are combustible waste, non-combustible waste, bottles and cans, old clothes and used paper, and oversized garbage. In Japan, waste such as glasses and metals are counted non-recyclable waste.

According to the study, the household area contributed most to the waste generated in Malaysia. There are about 20 different categories of waste generated in the household area such as waste from the food, the empty plastic bottle, metal cans, and so forth. All of these wastes can be grouped into organic and non-organic waste (Moh and Abd Manaf 2014). The composition of waste of eight housing areas in Balakong City, Malaysia was shown in (Mohd Armi Abu Samah et al. 2013). Organic waste (73.3%) is the main household waste generated followed by paper (10.7%), plastic (8.7%), glass (2.67%), metal (1.2%), and others (3.43%). Meanwhile, the organic waste generated in some worthwhile areas in Malaysia such as Kuala Lumpur is having a percentage of around 48.32%. This is followed by the paper waste, metal, and others. From the study above we can tell that the household waste generated in Malaysia can mainly divide into 6 classes which are paper, glasses, metals, organic residual waste, wood, and others.

The Solid Waste and Public Cleansing Management Act has been enforced by the government since 2007 to focus on waste segregation. The wastes are divided into two types which are recyclable and non-recyclable waste. The non-recyclable waste is also known as the residual waste which refers to the contaminated and non-recyclable waste such as disposable diapers and food waste. Meanwhile, the waste that is recyclable consists of paper, plastic, glass/ceramic, metals, and others such as electronic waste, rubber, bulky items, and garden/farm waste. (SEPARATION-AT-SOURCE 5/22/2018)



## 2.2 Related Work

Nowadays, due to the uncontrolled disposal of household waste, garbage has become a major problem worldwide. Thus an efficient and effective waste management system is needed to reduce the negative influence on health and the environment. (Md Shafiqul Islam et al. 2012). Next, an automated recognition system using a deep learning algorithm is proposed to replace humans in waste segregation manually because it is time-consuming and less efficient. Moreover, doing this manually may cause health hazards.

There are a lot of automated waste sorting techniques that have been applied. CNN is nowadays widely used in computer vision problems. In (Chu et al. 2018), a high-resolution camera and a bridge sensor are deployed to collect data to feed in the system named multilayer hybrid deep-learning system(MHS) for waste classification. The system aimed to classify the waste into 2 classes which are recyclable and others. AlexNet is used in the system to extract the features of the image. The extracted features were fed to a multilayer perceptron for consolidation and classification purposes. This system is trained manually on labelled items containing 50 different waste items commonly found on 4 main classes which are paper, plastic, metal, glass, and others. This system which relies on the input of sensors and images can achieve an accuracy above 90%. The reason for the low classification accuracy in the models is caused by the wastes that are lacking distinctive image features such as those wastes that have cylinder shapes are always wrongly classified as bottles.

Besides AlexNet, there are a lot of different CNN architectures that have been developed to solve image classification problems since 2012 (Krizhevsky et al. 2017). In (C. Bircanoğlu et al. 2018), several types of popular deep convolutional neural network architectures and optimization technologies have experimented. The architectures tested are Resnet50, MobileNet, Inception ResNetV2, DenseNet121, DenseNet169, DenseNet2201, Xception, and lastly a RecycleNet. Two different optimization approaches are chosen in this experiment which is Adam and Adadelta. All the CNN based methods can score more than 75% when trained from scratch on the Trashnet Datasets. Some models like InceptionResNetV2 can even score 90%. Besides that transfer learning approach is applied by fine-tuning the weight parameters. There is some architecture like DenseNet 121 that can even achieve a result of 95% test

accuracy after the fine-tuned. The disadvantages of these networks are some of them slightly slower in prediction time. The proposed classification model called "RecycleNet" is used to classify waste into six different classes which are paper, glass, plastic, metal, cardboard, and other trash. The architecture chosen to implement in Recycle Net is DenseNet121 and the connection patterns of the skip connection inside the dense blocks are modified to reduce the estimation time. The test accuracy of RecycleNet model was 81% when trained and tested on the Trashnet dataset.

There is another paper studied about the performance of a few popular CNN-based systems to classify waste such as VGG-16, ResNet-50, MobileNet V2, and DenseNet-121. These models are tested by 10-fold-cross-validation which is known for error and model selection. These models were appraised based on classification accuracy and classification confusion matrix. The models are used to classify the waste items into 20 categories under 4 different types. ResNet is outperformed among others in waste item classification with accuracy of 91.3% but these models do not do well in waste type classification as the model is pre-trained to classify waste according to appearance. (C. Srinilta and S. Kanharattanachai 2019)

In this experiment, the automated waste sorting system proposed is based on Pre-trained VGG-16, AlexNet, and also the traditional machine learning method such as Support Vector Machine (SVM), K-Nearest Neighbor and Random Forest(RF) is employed for classification and regression purpose. The system aims to classify four different trash categories which are glasses, metals, papers, and plastics. VGG-16 methods have shown a high accuracy of 93% in this experiment. The experiment showed that a deep learning approach is better than traditional techniques in waste segregation. (Costa et al. 2018)

A smart waste material allocation system is developed in this paper by using CNN as the feature extractor and the fully connected layer is substituted by Support Vector Machine (SVM). SVM is used to classify the features passed from the CNN. The (ResNet-50) Convolutional Neural Network pre-trained model is used in this experiment. The model aimed to classify the waste into a few groups such as glass, metal, paper, and plastic. The expected model was trained and tested using the dataset developed by the Trashnet dataset. Data augmentation is applied to the dataset during the preprocessing stage due to the

small size of the dataset. The train/test ratio of the dataset is set at 8:2 It can accomplish an accuracy of 87% on the dataset and stopped increasing after 12 epochs. (Adedeji and Wang 2019). Table 2.1 shows the summary of the relative information of the model used and the parameter set in the related studies.

Table 2.1: Summary of Literature Review

| Title  | Waste to be classified  | Dataset applied  | CNN architecture used  | Train/test ratio   | Training Epoch  | Accuracy  |
|--|---|--|--|--|---|---|
| RecycleNet: Intelligent Waste Sorting Using Deep Neural Networks                         | Glass<br>Paper<br>Plastic<br>Metal<br>Trash   | TrashNet dataset (Yang and Thung)  | 1.ResNet50, Adam<br>2.MobileNet, Adam<br>3. Inception ResNetV2, Adadelta<br>4. DenseNet121, Adam<br>5. DenseNet169, Adadelta<br>6.DenseNet201,Adam<br>7.Xception,Adam<br>8. DenseNet121 (fine tuned with adam for initialization, stochastic gradient descent)<br>9. InceptionResNetV2(fine tuned with adam for initialization, stochastic gradient descent)<br>10.RecycleNet,Adam | Total: 2527<br>Train: 1768<br>Validation: 328<br>Test: 431 | 1. 200<br>2. 500<br>3. 200<br>4. 200<br>5. 100<br>6. 200<br>7. 100<br>8. 210<br>9. 210<br>10. 200 | 1)75%<br>2)76%<br>3)86%<br>4)80%<br>5)82%<br>6)82%<br>7)85%<br>8)95%<br>9)87%<br>10)81% |
| Multilayer Hybrid Deep Learning Method for Waste Classification and Recycling            | Paper<br>Plastic<br>Metal<br>Glass  | 0528qsw dataset  | AlexNet +Multilayer Perceptron   | Train: 5000<br>Test: 150                                   | N/A   | >90%  |
| Municipal Solid Waste Segregation with CNN   | 20 waste item classes<br>4 waste-type classes:<br>-General<br>-Recyclable<br>-Hazardous<br>-compostable | 1)Food-101 dataset<br>2)Cola bottle identification dataset<br>3)Home object dataset<br>4)Flickr Material database<br>5)Glassense-Vision dataset<br>6)Glasses and bottles<br>7)Waste images scraped through Google search | VGG-16<br>ResNet-50<br>MobileNet<br>DenseNet-121   | Total: 3200<br>Train/test ratio: 70: 30                    | 10fold cross validations: 30 epochs   | Waste item: >80%<br>Waste type: 87.37-94.86%  |
| Artificial Intelligence in Automated Sorting Trash Recycling                             | Glass<br>Paper<br>Metal<br>Plastic  | [Yand and Thung] dataset   | CNN:<br>VGG-16, AlexNet<br>Traditional Machine Learning methods:<br>Support Vector Machine,<br>KNN and Random Forest   | N/A  | N/A   | Average Correlation:<br>VGG-16:93%<br>AlexNet: 91%<br>KNN: 88%<br>SVM: 80%<br>RF: 85%   |
| Intelligent Waste Classification System using Deep Learning Convolutional Neural Network | Glass<br>Metal<br>Paper<br>Plastic<br>Others  | [Yand and Thung] dataset   | Feature extractor: ResNet50<br>Classifier: Support Vector Machine  | 1989 images (8:2)  | 12 epoch  | 87%   |

Based on table 2.1, CNN is proven to be a more promising and powerful tool in image classification. Thus, the overview of CNN architecture and the implementation of CNN will be discussed in the following section.



Table 3.1 showed the planning of the work for this project. The total duration of the project will take about 28 weeks. First, we will take around 6 weeks to identify the problem about waste segregation and study about related methods applied to develop an intelligent vision system for waste segregation. Secondly, we will start to do sourcing to collect relevant databases for the research and also generate some preliminary results according to the methods applied on the state-of-arts. The preliminary results will be discussed in the FYP report 1. Next, we are going to explore different CNN architecture which can be applied to improve the classification accuracy of the system. In our project, we have proposed 2 different CNN architecture which are the VGG-19 and Inception V3. Furthermore, we will also study and create a hybrid CNN-ELM model which we have proposed to improve the computational efficiency of the system. The whole training and testing process for the three proposed models will take around 6 weeks which are from week 15 to week 20. After the models are trained and tested, we will start evaluating the models and write the report. Besides report writing, we will also need to make preparation for the FYP poster and planned to make a draft for journal submission.

Subsequently, the details of the flow to develop an intelligent vision system will be illustrated in figure 3.1. The details of each part in the flow chart will be further discussed in the following sections.

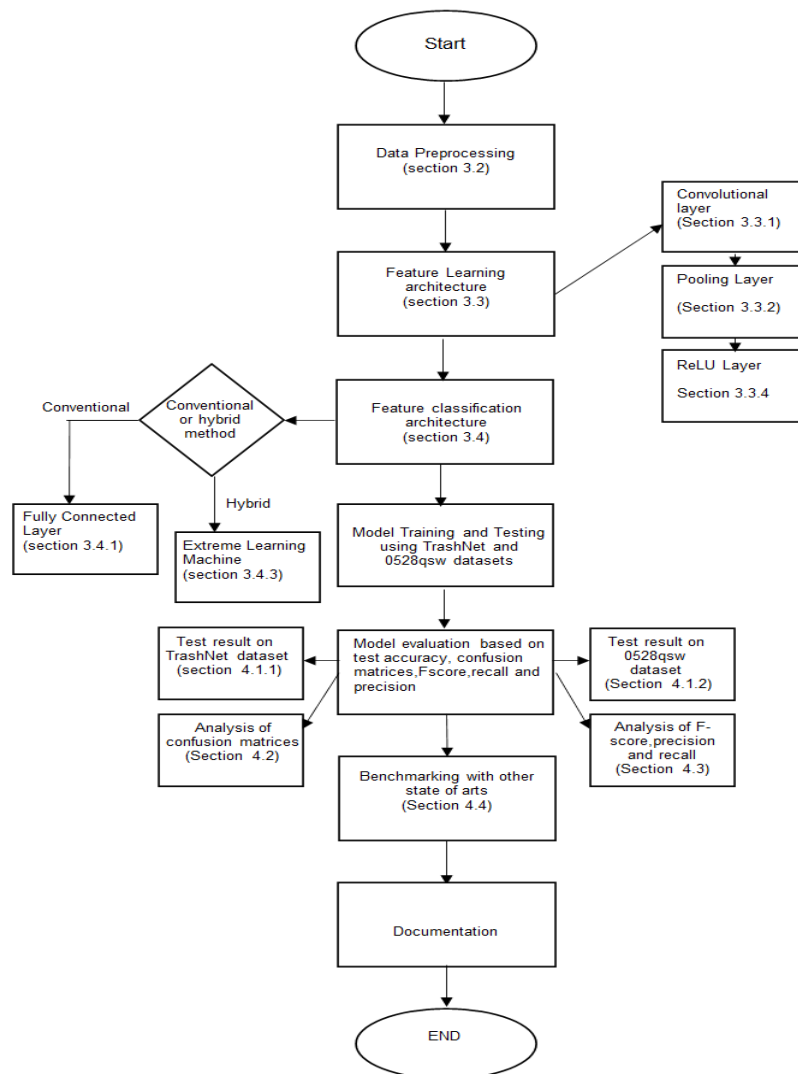


Figure 3.1: Detail flow chart to develop an intelligent vision system

### 3.2 Data Pre-processing

Data is important to acquire deep learning models with high accuracy. The performance of the model will increase when the number of data increases generally. However, it is hard for us to acquire massive amounts of data, thus we can make use of the image augmentation techniques. New images can be generated from our existing database to train our deep learning model by applying image augmentation. Thus, we do not have to collect them manually. In this experiment, several data augmentation techniques are applied in the training dataset such as, image shifts, image flips, image rotations, and image zoom as shown in figure 3.2.



Figure 3.2: Example of Image Augmentation (Image Augmentation | Pytorch Image Augmentation 2019)

### 3.3 Overview of CNN

The input of CNN is different from neural networks as the input is a multi-channel image. CNN mainly consists of 2 parts which are the convolutional base that is responsible for feature learning and classifier for classification purpose as shown in figure 3.3. There are a total of 3 different important layers applied in CNN which are the convolutional layers, pooling layers and fully connected layers. In the convolutional base, the convolutional layer is the main layer and it is often followed by a pooling layer. Important features will be extracted from the input images for feature learning by applying convolutional and pooling mechanism. Next, the extracted features will be flattened and feed into the fully connected layer for classification process.

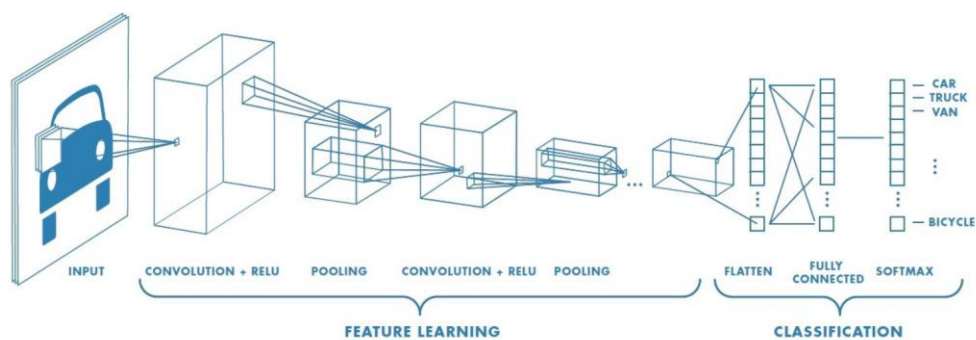


Figure 3.3: Overview of CNN architecture (Saha 12/16/2018)

### 3.3.1 Convolutional Layer

The convolutional layer is used to extract the features from the input images. The multiplication between the array of input data and the designed 2-dimensional arrays of weights named ‘kernel’ is performed at the convolutional layer. Normally, the size of the kernel is smaller than the input image.

In CNN, the lower convolutional layer which is closer to the input consists of mostly general features meanwhile the higher layer of the convolutional base contains the specialized features. Generally, the first few convolutional layers are used to extract some low-level features such as the colours, edges, and so forth. High level features will be extracted when the number of layers increases. There must be a transition somewhere from general to specific in the network (Yosinski et al.).

### 3.3.2 Pooling Layers

The pooling layer is mostly chosen to apply after some ReLU layer. Pooling layers have no weights and parameters, just a few hyperparameters can be set here such as the stride, padding, and the filter size. Max-Pooling is the famous one among other pooling layers. The working principle of max-pooling is taking the maximum number of each sub-region as the output when the filter convolves around the input volume. Figure 3.4 shows the example of applying a 2x2 max-pooling kernel on the input volume of 4x4 with a stride of 2. By using max-pooling, the features detected in the quadrants can preserve.

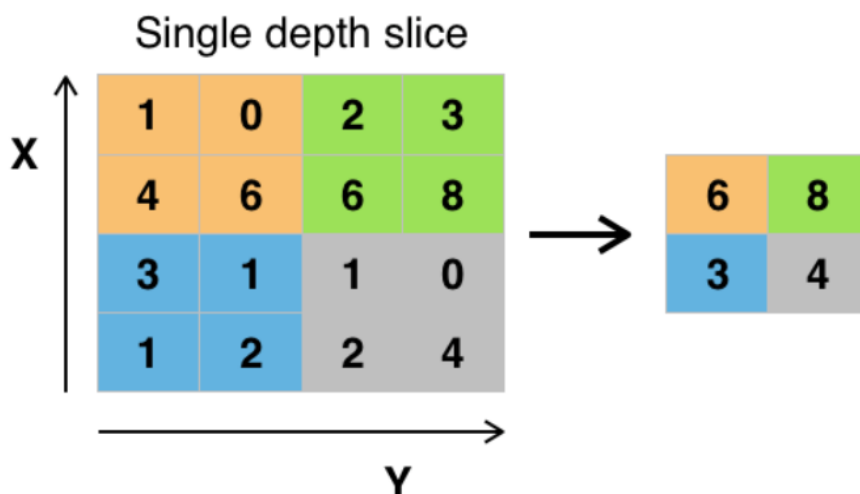


Figure 3.4: Maxpool with a 2x2 filter and a stride of 2 (Deshpande 6/26/2019)



### 3.3.3 Stride and Padding

There are several parameters that can be set when a filter is applied to the input image. For example, the size of the filter, stride, and padding. Stride determines how the input volume is going to convolve. When we set the stride at 1, the filter will move along the input volume as shown in figure 3.5. Whereas a 7x7 input volume will end up becoming a 5x5 output volume. When stride is set at 2, we can see from figure 3.6 that the kernel is shifting by 2 units now and this will cause the shrink of output volume (Deshpande 6/26/2019).

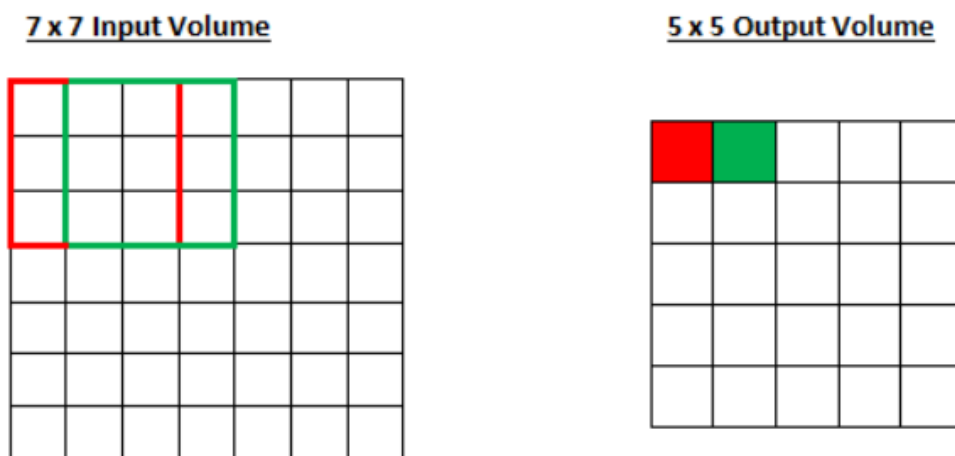


Figure 3.5: Stride =1 (Deshpande 6/26/2019)

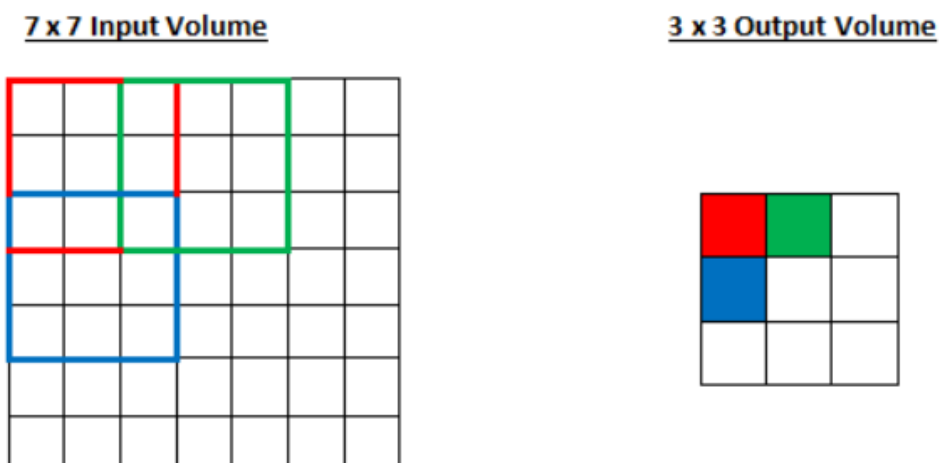


Figure 3.6: Stride =2 (Deshpande 6/26/2019)

Normally, the number of strides will increase when there is a need to reduce the overlapping between the receptive fields and a smaller spatial dimension is required.

Padding is used to preserve the size of the input volume. For example, when a  $5 \times 5 \times 3$  filter is applied to the input volume of  $32 \times 32 \times 3$ , the output of this convolution will be  $28 \times 28 \times 3$ . The spatial dimension is decreased. As the size of the applied filter increases, the spatial dimension of the output volume will decrease. To preserve the original input volume so that those low-level features can be extracted, padding can be applied to the layer. For example, when zero-padding of size 2 is applied onto the input volume, it will pad the input volume with zeros around the border. The original input image of  $32 \times 32 \times 3$  will become  $36 \times 36 \times 3$  as shown in figure 3.7. The size of the output convolve features will be the same as the input image volume although we applied a  $5 \times 5 \times 3$  convolution filter with the stride of 1 because of padding. (Deshpande 6/26/2019).

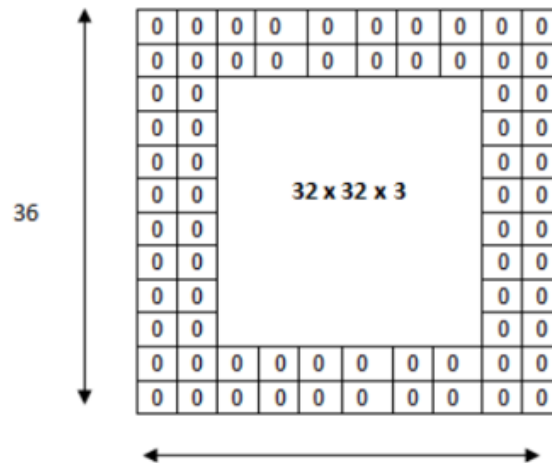


Figure 3.7: Zero padding of 2 added to the input volume of  $32 \times 32 \times 3$  (Deshpande 6/26/2019)

The formula used to calculate the output volume size is shown above:

$$output = \frac{n+2p-f}{s} + 1 \quad (1)$$

Where  $n$  = size of input,  $p$  = padding,  $s$  = stride,  $f$  = filter size

### 3.3.4 ReLU (Rectified Linear Units)

Rectified Linear unit is a type of activation function used to provide non-linearity to the system after each convolutional layer. Normally, the equation of  $y = \max(0, x)$  is used to describe the ReLU activation function mathematically. The plot of the function is illustrated in figure 3.8 which showed all the negative values are zero and the positive values are increasing linearly. The ReLU activation function is chosen instead of other nonlinear functions like tanh and sigmoid because it has better training speed and performance. Besides that, rectified linear units keep the information of related intensities as information travels through multiple layers of feature detectors unlike binary units (Vinod Nair and Geoffrey E. Hinton).

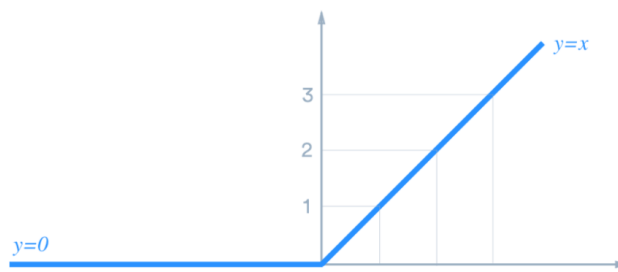


Figure 3.8: Graph of ReLU activation function (Dr. Sebastian Raschka 2020)

## 3.4 Classifier

The classifier is the second part in the CNN before getting the output as shown in figure 3.3. The output features from the convolutional and pooling layer will be flattened and fed into a regular Neural Network for classification problems. Fully connected layer is normally used to learn the non-linear combination of high-level features because it is one of the cheapest ways to do so. However, further study on Extreme Learning Machine will also be done in the following section as it has proved to outperform other classifiers such as SVM and KNN in (Zhang and Zhang 2015).

### 3.4.1 Fully Connected Layer

The output from the convolutional /pooling mechanism will be flattened into a single vector of values and fed to the fully connected layer as the input. The fully connected layer will process the input and use them to classify images according to labels. The inputs are multiplied by weights and pass through an activation function which typically ReLU is applied, then pass to the output layer. In the output layer, each neuron represents a classification label (Fully Connected Layers in Convolutional Neural Networks: The Complete Guide - MissingLink.ai 3/22/2020). Figure 3.9 demonstrated how the input is fed into the fully connected layers and trained throughout the fully connected layer by the backpropagation process to get the most accurate weights.

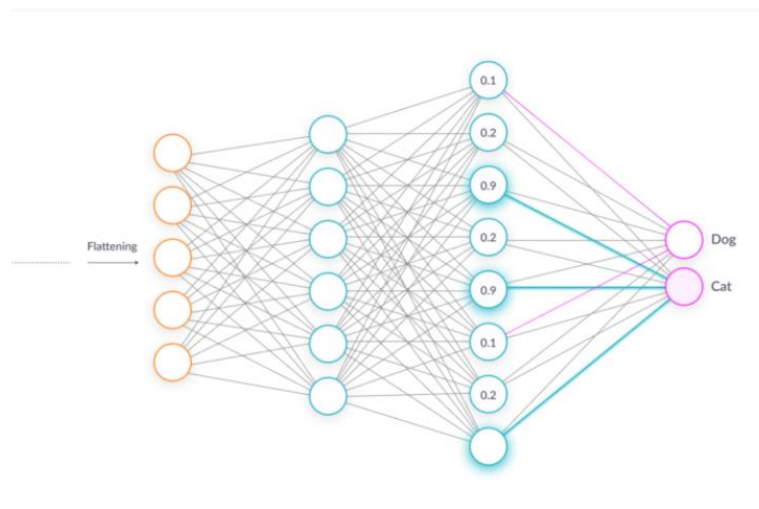


Figure 3.9: (Fully Connected Layers in Convolutional Neural Networks: The Complete Guide - MissingLink.ai 3/22/2020).

### 3.4.2 Softmax Function

Softmax regression applies logistic regression while trying to make predictions and recognition of multiple classes. The numerical output of the last linear layer of the multi-class classification neural network will be converted into probabilities by applying softmax function. The sum of the probabilities assigned to each class is equal to 1. For example, to compute the last layer in the neural network which is assigned as layer  $L$  in the figure 3.10,  $z^{[L]} = \omega^{[L]}a^{[L-1]} + b^{[L]}$  is used. After computing  $z$ , softmax activation function is needed to apply here to turn the logits of the layer  $L$  into probabilities by taking

the exponents of each output then normalizing each number by the sum of those exponents. The formula used to calculate the probabilities is shown below

$$\sigma(z)_j = \frac{e^z}{\sum_{k=1}^K e^z} \text{ for } j = 1, \dots, K \quad (2)$$

Where  $\sigma$  = probability,  $z$  = numeric output,  $K$  = number of classes

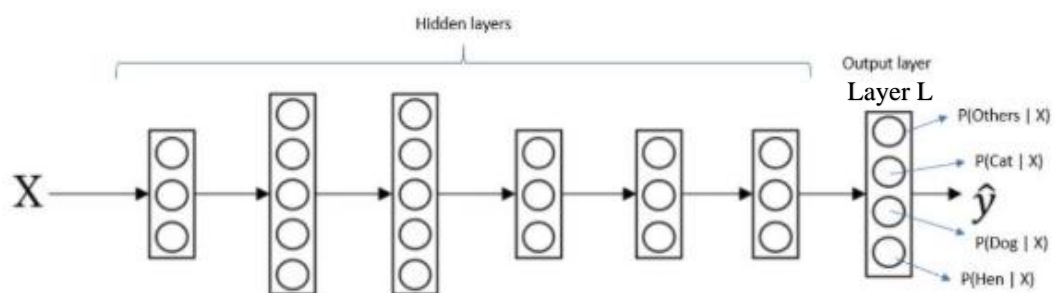


Figure 3.10: Softmax Regression Layer (Rizwan 5/25/2018)

### 3.4.3 Extreme Learning Machine (ELM)

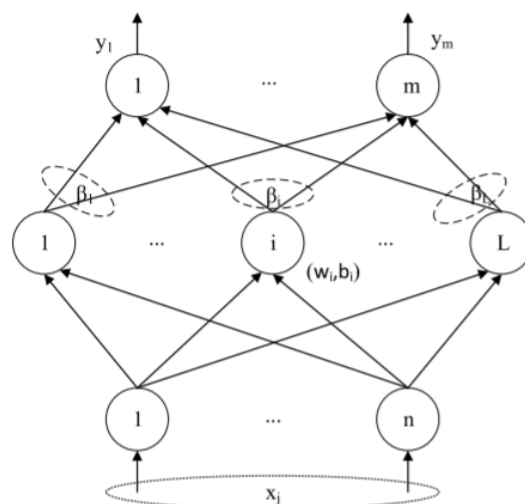


Figure 3.11: Structure of extreme learning machine (Yang et al. 2019)

The extreme learning machine is a modern learning algorithm designed to randomly choose hidden nodes and analytically determine the output weights of a single hidden layer feedforward neural networks (SLFNs). The predicted

errors and the norm of the output weight in both classification and regression problems are aimed to be minimized using ELM. (Huang et al. 2006). The principle of ELM on classification problems will be briefly introduced below.

Given a dataset consisting of input samples  $X = [x_1 + x_2, \dots, x_n] \in R^n$  and target samples  $T = [t_1, t_2, \dots, t_m] \in R^m$ , and a SLFN with  $\tilde{N}$  hidden nodes. The mathematical function for the model with activation function  $g(x)$  is:

$$\sum_{i=1}^{\tilde{N}} \beta_i g(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(w_i \cdot x_j + b_i) = o_j \quad (3)$$

$$J = 1, \dots, N,$$

Where  $w_i$  is the weight vector connecting the hidden node and input while the weight factor connects to the hidden node and the output node is  $\beta_i$ . To achieve zero error which means that

$$\sum_{j=1}^{\tilde{N}} \|o_j - t_j\| = 0 \quad (4)$$

$$\sum_{i=1}^{\tilde{N}} \beta_i g(w_i \cdot x_j + b_i) = t_j \quad (5)$$

The equation above can be computed as

$$H\beta = T, \quad (6)$$

Where H is representing the output matrix of the hidden layer.

$$H = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \dots & g(w_L \cdot x_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(w_1 \cdot x_N + b_1) & \dots & g(w_L \cdot x_N + b_L) \end{bmatrix} \quad (7)$$

$$\beta = \begin{bmatrix} \beta_{11} & \dots & \beta_{1m} \\ \vdots & \ddots & \vdots \\ \beta_{L1} & \dots & \beta_{Lm} \end{bmatrix} \quad (8)$$

$$T = \begin{bmatrix} t_{11} & \dots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{N1} & \dots & t_{Nm} \end{bmatrix} \quad (9)$$

Different from the traditional way, the initial weight and hidden layer bias of the neural networks are assigned randomly and remain unchanged. Thus,

an equation is derived to find the least-squares solution  $\hat{\beta}$  of the linear system  $H\beta = T$  to train the SLFN.

$$\begin{aligned} & \|H(w_1, \dots, w_{\tilde{N}}, b_1, \dots, b_{\tilde{N}})\hat{\beta}\| \\ &= \min_{\beta} \|H(w_1, \dots, w_{\tilde{N}}, b_1, \dots, b_{\tilde{N}})\beta - T\| \end{aligned} \quad (10)$$

The matrix  $H$  is a square matrix and invertible if the number of hidden nodes  $\tilde{N}$  is equivalent to the number of specific training samples  $N$ . In that case, the error can compute approximately according to the equation above.

However, the number of hidden nodes is much lesser than the number of distinct training samples in most cases, thus it is hard to get an invertible square matrix. Here, the Moore-Penrose generalized inverse of matrix is introduced to solve the non-square matrix.

$$\hat{\beta} = H^+T \quad (11)$$

Where  $H^+$  is the Moore-Penrose generalized inverse of matrix  $H$ .

If  $N$  is larger than  $\tilde{N}$ , the gradient equation is over-determined, the properties of the Moore-Penrose generalized inverse matrix can achieve the minimum training error in this case. By applying the least square solution, it can reduce the error between  $H\beta = T$ , thus the training error can be minimized.

In another case, if the  $\tilde{N}$  is  $> N$ , then it will be an under-determined problem whereas the special solution in using Moore Penrose inverse is able for us to achieve the smallest norm of weight. The output of ELM can be expressed as

$$f(x) = h(x)\beta = h(x)H^+T \quad (12)$$

where  $h(x)$  is the activation function for the output.

Table 3.2: Process of applying ELM

| Algorithm of ELM  |
|---|
| Input: Given a training set $X = \{(x_i, t_i) \mid x_i \in R^n, t_i \in R^m, i=1, \dots, \check{N}\}$       |
| 1) Assign the input weight $w_i$ and bias $b_i$ randomly whereas $i=1, \dots, \check{N}$                    |
| 2) Compute the hidden layer output matrix $H$   |
| 3) Compute the output weight $\beta$ by using the formula $\beta = H^+ T$ , where $T = [t_1, \dots, t_N]^T$ |

### 3.5 VGG

A paper titled "Very Deep Convolutional Networks For Large-Scale Image Recognition" has introduced the pre-trained convolutional neural network name VGG. Approximately 1.2 million images from the ImageNet Dataset are used to train this model by (Simonyan and Zisserman 2014). In this CNN, the filter size of the convolutional layer is set at 3x3 with a stride 1 and the same padding. Throughout the whole architecture, the filter size of a convolutional layer is set at 3x3 with the stride of 1 and the same padding meanwhile, the filter size of the pooling layer is set at 2x2 with the stride of 2. The output from the convolution and pooling layer will feed into the 2 fully connected layers followed by softmax for output. Different from VGG16, VGG-19 has 19 layers which consists of 16 convolutional layers, 5 max-pooling layers, and 3 dense layers. There are a total of 24 layers but only 19 of them are weight layers. The architecture and settings of the proposed VGG-19 pre-trained model are shown in table 3.3.



Table 3.3: Proposed VGG19 pre-trained model

|         | Type of Layer     | No. of Filter | Output Shape | Kernel size | No. of Stride | No. of Padding | Layer Trainable |
|---------|-------------------|---------------|--------------|-------------|---------------|----------------|-----------------|
|         | Image Input layer |               | 224x224x3    |             |               |                |                 |
| Group 1 | Conv-1            | 64            | 224x224x64   | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-1            |               | 224x224x64   |             |               |                |                 |
|         | Conv-2            | 64            | 224x224x64   | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-2            |               | 224x224x64   |             |               |                |                 |
| Group 2 | Pool-1            | 1             | 112x112x64   | 2x2         | 2x2           | 0x0            |                 |
|         | Conv-3            | 128           | 112x112x128  | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-3            |               | 112x112x128  |             |               |                |                 |
|         | Conv-4            | 128           | 112x112x128  | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-4            |               | 112x112x128  |             |               |                |                 |
|         | Pool-2            | 1             | 56x56x128    | 2x2         | 2x2           | 0x0            |                 |
| Group 3 | Conv-5            | 256           | 56x56x256    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-5            |               | 56x56x256    |             |               |                |                 |
|         | Conv-6            | 256           | 56x56x256    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-6            |               | 56x56x256    |             |               |                |                 |
|         | Conv-7            | 256           | 56x56x256    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-7            |               | 56x56x256    |             |               |                |                 |
|         | Conv-8            | 256           | 56x56x256    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-8            |               | 56x56x256    |             |               |                |                 |
| Group 4 | Pool-3            | 1             | 28x28x256    | 2x2         | 2x2           | 0x0            |                 |
|         | Conv-9            | 512           | 28x28x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-9            |               | 28x28x512    |             |               |                |                 |
|         | Conv-10           | 512           | 28x28x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-10           |               | 28x28x512    |             |               |                |                 |
|         | Conv-11           | 512           | 28x28x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-11           |               | 28x28x512    |             |               |                |                 |
|         | Conv-12           | 512           | 28x28x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-12           |               | 28x28x512    |             |               |                |                 |
|         | Pool-4            | 1             | 14x14x512    | 2x2         | 2x2           | 0x0            | FALSE           |
| Group 5 | Conv-13           | 512           | 14x14x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-13           |               | 14x14x512    |             |               |                |                 |
|         | Conv-14           | 512           | 14x14x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-14           |               | 14x14x512    |             |               |                |                 |
|         | Conv-15           | 512           | 14x14x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-15           |               | 14x14x512    |             |               |                |                 |
|         | Conv-16           | 512           | 14x14x512    | 3x3         | 1x1           | 1x1            | FALSE           |
|         | ReLu-16           |               | 14x14x512    |             |               |                |                 |
|         | Pool-5            | 1             | 7x7x512      | 2x2         | 2x2           | 0x0            |                 |
|         | Flatten Layer     |               | 25088x1      |             |               |                | FALSE           |
|         | Dense Layer-6     |               | 512x1        |             |               |                |                 |
|         | Dropout-6         |               | 512x1        |             |               |                | TRUE            |
|         | ReLu-6            |               | 512x1        |             |               |                |                 |
|         | Dense Layer-7     |               | 64x1         |             |               |                |                 |
|         | Dropout-7         |               | 64x1         |             |               |                | TRUE            |
|         | RelLu-7           |               | 64x1         |             |               |                |                 |
|         | Dense Layer-8     |               | 6x1          |             |               |                |                 |
|         | Softmax Layer-8   |               | 6x1          |             |               |                | TRUE            |

### 3.6 Inception V3

Inception V3 was the first runner up model for image classification in ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2015. It is a 42-layer deep learning network which has adequate complexity as a VGG net. The InceptionV3 module is acting as a "multi-level feature extractor" which computes 1x1, 3x3, and 5x5 convolutions layer within the same module of the

network. Although it has 42 layers deep but the computational cost is just 2.5 higher than GoogLeNet. Besides that, the efficiency of Inception V3 also proves to be higher than VGG Net in (Simonyan and Zisserman 2014). Figure 3.12 shows the architecture of the Inception V3 deep learning network.

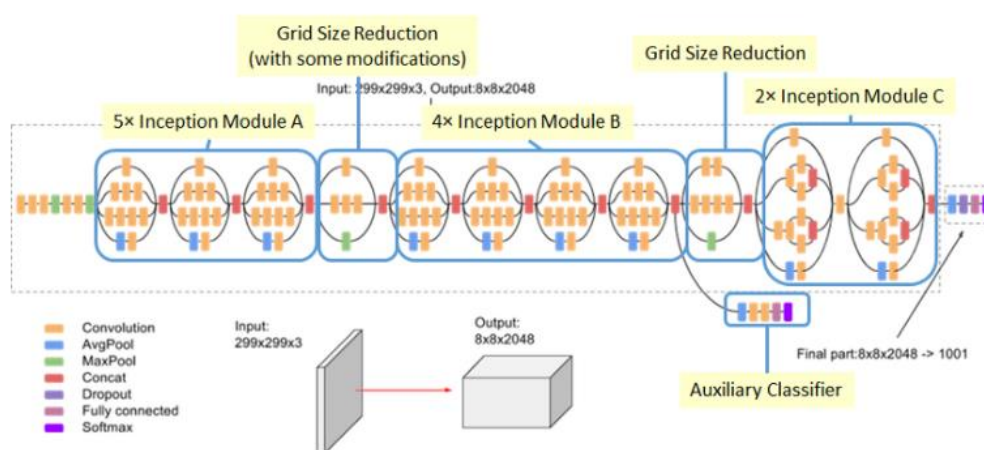


Figure 3.12: InceptionV3 architecture (Raj 5/30/2018)

### 3.7 Transfer Learning Approach

As mentioned in section 3.3, a typical CNN mainly consists of 2 parts. The convolutional pooling mechanism breaks up the image into features and analyses them while the fully connected layer which acts as the classifier, takes the output from the convolutional base and predicts the best label to describe the image.

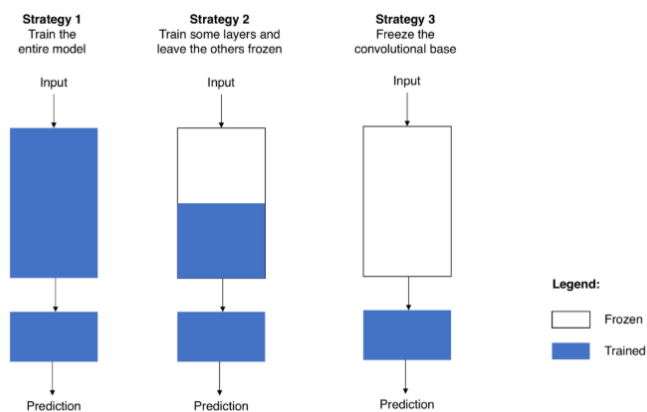


Figure 3.13: Strategies applied for transfer learning approach (Marcelino 10/23/2018)

There are several strategies to reuse the pre-trained model as shown in figure 3.13. For example, the first strategy trained the entire model from scratch which was applied in (C. Bircanoğlu et al. 2018). Strategy 2 trained some layers and left the others frozen. Meanwhile, the third strategy froze the convolutional base and just trained the classifier part. As the dataset used in this experiment only consists of 2527 images which are very less. Thus, strategy 3 is used in this experiment which is freezing the convolutional base and using it as the image feature extractor. The output features extracted will be fed to the classifier that we trained. The details of the process to apply the strategy 3 of transfer learning approach can refer to the appendix.

Next, we have proposed a novel method which substitutes the Extreme Learning Machine (ELM) to be the classifier of the system because ELM is proved to outperform other classifiers such as SVM and KNN in (Zhang and Zhang 2015).

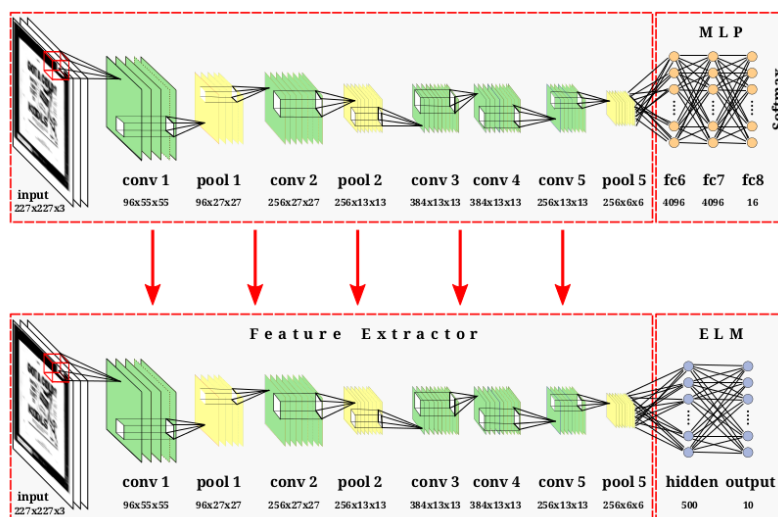


Figure 3.14: CNN-ELM Hybrid Model (Andreas Kölsch, Muhammad Zeshan Afzal Markus Ebbecke Marcus Liwicki 2017)

The architecture of CNN-ELM hybrid model is shown in figure 3.14 whereas the extreme learning machine has substituted the conventional multilayer perceptron layer as the classifier. The pseudocode used to implement the hybrid CNN-ELM model is discussed in table 3.4.

Table 3.4: Pseudocode of implementing CNN-ELM

---

```

Process of Implementing the CNN-ELM model


---


#Construct a pretrained CNN model
vgg = VGG19(include_top = False, weights = 'imagenet', input_shape=(224,224,3))
#Construct CNN middle layer output
output = vgg.layers[-1].output
output = keras.layers.Flatten()(output)
vgg_model=Model(vgg.input,output)
#Freeze the convolutional layer
vgg_model.trainable = False
for layer in vgg_model.layers:
    layer.trainable = False
#Construct an ELM classification model
elm_model = hpelm.elm.ELM(cnn_train_result.shape[1], 5)
elm_model.add_neurons(500, func='sigm')
elm_model.train(cnn_train_result, target_train_oh, 'c') #train the model

```

---

### 3.8 Evaluation Metrics

There are several metrics to evaluate the performance of the trained model. Accuracy is known as the total percentage of the images that are correctly classified. However, accuracy is not the only metric used to measure the trained model. Precision, Recall, and F1 score are useful to measure the model which trained with an unbalanced dataset to reduce the bias forecasting. The correctness of the classification by the trained model is defined as precision, while recall represents the effectiveness of the trained classification system. To seek the balance between Precision and Recall, F1 score is used.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (13)$$

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

$$Recall = \frac{TP}{TP+FN} \quad (15)$$

$$F1score = 2x \frac{(Precision*Recall)}{Precision+Recall} \quad (16)$$

Where

TP = True Positive, FP = False Positive, TN = True Negative, FN = False Negative

### 3.9 Datasets

Two different datasets are collected for this experiment, namely Trash-Net and 0528qsw from (Chu et al. 2018).

#### 3.9.1 TrashNet dataset

The dataset used in this experiment is the TrashNet (Gary and Thung) dataset created by (ThungYang-ClassificationOfTrashForRecyclabilityStatus-re). This dataset is publicly available and well accepted by the research community. The size of the data set is small which consists of 2528 images. The images in the dataset are classified into 6 categories which are cardboard, metal, glass, plastic, metal cans, and other trash. The images are resized to 512x284 and most of the images in the dataset have a white background.

Table 3.5: Sample of image from TrashNet dataset













|   |   |   |  |
|---|---|---|--|
| Cardboard   |   | Glass   |  |
|  |  |  |  |
| Metal Can   |   | Paper   |  |
|  |  |  |  |
| Plastic   |   | Others  |  |
|  |  |  |  |

Table 3.6: Number of images in each class for TrashNet dataset

| No | Classes   | Number of images |
|----|-----------|------------------|
| 1  | Cardboard | 403              |
| 2  | Glass     | 501              |
| 3  | Metal     | 410              |
| 4  | Paper     | 594              |
| 5  | Plastic   | 482              |
| 6  | Trash     | 137              |

### 3.9.2 0528qsw dataset

This is a dataset created by Xiao gang Xiong in (Chu et al. 2018) which consists of 2320 self-collected images made from 21 different types of household waste. The items are placed under a white background and captured by a camera (model 0V9712). The size of the images is 640 x 480. The household waste is divided into 5 categories which are paper plastic, metal, glass, and other waste.

Table 3.7: Sample of image from 0528qsw dataset





| Plastic bottle  | Can   | Newspaper  | Shampoo bottle  |
|---|---|--|---|
|  |  |  |  |

Table 3.8: Types of household waste in 0528qsw dataset

| Class          | Group                    | Item            | Quantity  |
|----------------|--------------------------|-----------------|-----------|
| Recyclable     | Paper                    | books           | 5         |
|                |                          | boxes           | 4         |
|                |                          | papercup        | 5         |
|                | Plastic                  | General bottles | 6         |
|                |                          | Shampoo bottles | 4         |
|                |                          | Pen             | 1         |
|                |                          | Watch           | 1         |
|                | Metal                    | Can             | 7         |
|                |                          | Key             | 1         |
|                |                          | Scissor         | 1         |
|                |                          | Beer cap        | 1         |
|                | Glass                    | Bottle          | 4         |
| <b>Sum</b>     | <b>4</b>                 | <b>12</b>       | <b>40</b> |
| Non-recyclable | Fruit/Vegetable/Plant    | Apple           | 1         |
|                |                          | Banana          | 1         |
|                |                          | Carrot          | 1         |
|                |                          | Cabbage         | 1         |
|                |                          | Carrot          | 1         |
|                |                          | Rose            | 1         |
|                | Kitchen waste and others | Others          | 1         |
|                |                          | Egg             | 1         |
|                |                          | Lunch box       | 1         |
|                |                          | Trash bag       | 1         |
|                | bowl                     | 1               |           |
| <b>Sum</b>     | <b>3</b>                 | <b>10</b>       | <b>10</b> |

## CHAPTER 4

### Results and Discussion

#### 4.1 Model Evaluation

In this section, we have evaluated the performance of the proposed models such as VGG-19, InceptionV3, CNN + SVM models and also the hybrid CNN-ELM models on two datasets including the TrashNet dataset and 0528qsw dataset. Furthermore, the performance of the proposed methods will be compared with the state-of-art methods stated in table 2.1 such as VGG-16 (Costa et al. 2018), ResNet + SVM (Adedeji and Wang 2019), and DenseNet121(C. Bircanoğlu et al. 2018) in the following section. Python 3.7 and Keras framework were used to implement the experimental methods. All models in the experiment are trained on the kernel at Kaggle with the Nvidia K80 GPU provided by Kaggle.

##### 4.1.1 Results on TrashNet Dataset

Table 4.1: Classification Results on TrashNet dataset

| Method             | Accuracy (%) | Time(s)       |
|--------------------|--------------|---------------|
| VGG16              | 87           | 3490.4        |
| VGG16 + ELM        | 64           | 3.3           |
| VGG16 + SVM        | 76           | 705.3         |
| <b>VGG19</b>       | <b>91</b>    | <b>3956.7</b> |
| VGG19 + ELM        | 66           | 4.1           |
| VGG19 + SVM        | 78           | 726.1         |
| DenseNet121        | 80           | 3981.1        |
| DenseNet121 + ELM  | 66           | 5.4           |
| DenseNet121 + SVM  | 85           | 750.3         |
| InceptionV3        | 83           | 3695.1        |
| InceptionV3 +ELM   | 63           | 5.132         |
| Inception V3 + SVM | 80           | 1488.6        |

In this experiment, the input image size is resized into 224x224x3 and rescaled by 255 for all methods. The results of the three state-of-art methods including VGG16, DenseNet121 and ResNet + SVM were regenerated based on the similar configuration described in their works. The regenerated test result for



DenseNet121 is similar as shown in table 2.1, meanwhile the test result for VGG-16 have a 6% of difference with the state of art. However, the methods are proved to be reliable to use as a benchmarking purpose with just a slight difference in the result. For the proposed methods, the pre-trained weight from the 'ImageNet' dataset is used in the convolutional base of CNN. Next, the settings of the classifier have 3 dense layers with the output size of 512 x 1, 64 x 1, and 6x1 respectively. Each dense layer is followed with a dropout layer in the architecture to prevent the model from overfitting. The example of the CNN architecture is shown in table 3.3. For the CNN-ELM hybrid model, the number of hidden layer neurons of the ELM is set at 500,  $\alpha = 0.5$ , and sigmoid activation is used in the architecture. From table 4.1, we can see the VGG-19 model achieves the best classification accuracy which is 91% on the TrashNet dataset with the computational time around 3956.7s. Meanwhile, we can see that all the CNN-ELM hybrid models perform poorly on the TrashNet dataset. The InceptionV3 + ELM model obtained the lowest classification accuracy among other models. As a result, the potential of ELM is failed to be exploited in this experiment.

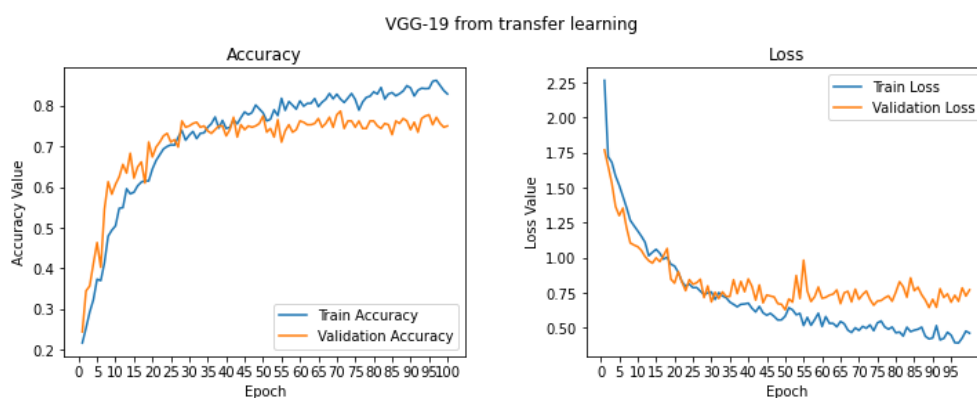


Figure 4.1: Training Loss and Accuracy of VGG-19 pre-trained model for 100 epochs (TrashNet Dataset)

The loss and accuracy of the training and validation sets over 100 epochs on TrashNet dataset are shown in figure 4.1. Based on figure 4.1, we can tell that the VGG-19 model is well trained because there is no overfit or underfit of the model based on the training loss and accuracy graph. Besides that, figure 4.1

indicated that the model is expected to reach the highest accuracy after 30 epochs. This shows that the VGG-19 model can achieve stable generalization on the TrashNet dataset quickly.

#### 4.1.2 Results on 0528qsw Dataset

Table 4.2: Classification result on 0528qsw dataset

| Method             | Accuracy (%) | Time(s) |
|--------------------|--------------|---------|
| VGG16              | 92           | 2998    |
| VGG16 + ELM        | 85           | 3.1     |
| VGG16 + SVM        | 94           | 158.3   |
| VGG19              | 93           | 2985    |
| VGG19 + ELM        | 85           | 3.82    |
| VGG19 + SVM        | 94           | 157.6   |
| DenseNet121        | 84           | 3015    |
| DenseNet121 + ELM  | 87           | 5.9     |
| DenseNet121 + SVM  | 94           | 313.4   |
| InceptionV3        | 91           | 2940    |
| InceptionV3 +ELM   | 90           | 5.4     |
| Inception V3 + SVM | 94           | 403.6   |

The setting of all the models trained in this section is the same as the one trained in the TrashNet dataset. From table 4.2, we can see that all the CNN-SVM models achieved the highest classification accuracy which is 94%. VGG-19 is having the second-highest accuracy of 93% followed by the VGG16 model which is 92%. In addition, the overall performance of the hybrid CNN-ELM models is good in this experiment as all the hybrid CNN-ELM models can achieve an average accuracy of 85% and the InceptionV3+ELM model can even achieve an accuracy of 90% with the computational time of 5.4s. Compared to the CNN+SVM models which achieve 94% test accuracy, the InceptionV3 + ELM model is more computational effective because the shortest computational time taken for the CNN+SVM models is from the VGG16 +SVM model which is 30 times slower than the hybrid CNN-ELM model and there is just 4 % difference in terms of accuracy between the two models.

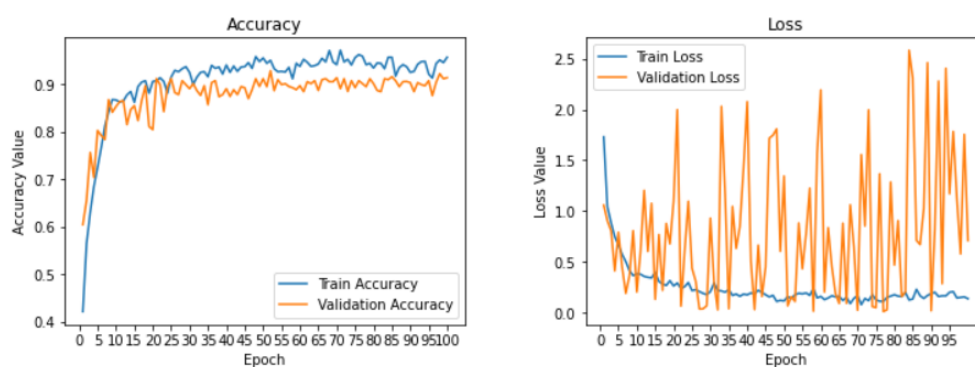


Figure 4.2: Training Loss and Accuracy of VGG-19 pre-trained model for 100 epochs (0528qsw dataset)

## 4.2 Confusion Matrix

We need to review the confusion matrix before we dive into the precision and recall matrix. The confusion matrix can provide us more insight. Besides the performance of the predictive model, we can know which classes are predicted correctly, which class is falsely predicted and what is the error made through the confusion matrix.

|              |           | Predicted class based on VGG-19 |       |       |       |         |        |      |
|--------------|-----------|---------------------------------|-------|-------|-------|---------|--------|------|
|              |           | cardboard                       | glass | metal | paper | plastic | others |      |
| Actual Class | cardboard | 72                              | 0     | 0     | 6     | 1       | 0      | 91.1 |
|              | glass     | 0                               | 79    | 8     | 1     | 10      | 1      | 79.8 |
|              | metal     | 0                               | 1     | 74    | 2     | 1       | 1      | 93.7 |
|              | paper     | 5                               | 0     | 2     | 114   | 1       | 0      | 93.4 |
|              | plastic   | 0                               | 7     | 3     | 5     | 88      | 4      | 82.2 |
|              | others    | 0                               | 1     | 1     | 7     | 2       | 9      | 45   |
|              |           | 93.5                            | 89.8  | 84.1  | 84.4  | 85.4    | 60     |      |

Figure 4.3: Confusion matrix of VGG-19 model trained on TrashNet dataset

|              |         | Predicted class based on InceptionV3 +ELM |       |        |       |         |      |
|--------------|---------|---|-------|--------|-------|---------|------|
|              |         | glass                                     | metal | others | paper | plastic |      |
| Actual Class | glass   | 12  | 2     | 0      | 0     | 0       | 85.7 |
|              | metal   | 0   | 70    | 0      | 0     | 0       | 100  |
|              | others  | 0   | 4     | 147    | 0     | 2       | 96.1 |
|              | papers  | 0   | 6     | 4      | 54    | 19      | 65.1 |
|              | plastic | 0   | 3     | 0      | 0     | 45      | 93.8 |
|              |         | 100                                       | 82.4  | 97.4   | 100   | 68      |      |

Figure 4.4: Confusion Matrix of InceptionV3 +ELM model trained on 0528qsw dataset

Figure 4.3 and 4.4 show two confusion matrices that are plotted based on two models trained on two different datasets respectively. Figure 4.3 showed the VGG-19 model which is trained by the TrashNet dataset. There are a total of six classes in the TrashNet dataset which are [cardboard, glass, metal, paper, plastic, and trash]. Meanwhile, the one shown in figure 4.4 is the InceptionV3 +ELM model which is trained by a 0528qsw dataset which consists of 5 classes [glass, metal, others, paper, and plastic]. Throughout the confusion matrices, we can tell that both the models are performing well in classifying the waste with just some minor misclassification. We can notice that the 'others' waste is the most difficult class to classify in the TrashNet dataset while 'plastic' has the highest misclassification rate in the 0528qsw dataset.

### 4.3 Precision, Recall and F1 score

In this section, we will look further into the precision, recall, f1 score, and also the confusion matrix plot of the models to compare the classifier methods in a more detailed way.

Table 4.3: Evaluation matrix of top 4 trained model in TrashNet dataset

| Evaluation Matrix | Model |       |             |              |
|-------------------|-------|-------|-------------|--------------|
|                   | VGG16 | VGG19 | DenseNet121 | Inception V3 |
| F1 score (%)      | 0.828 | 0.903 | 0.523       | 0.836        |
| Precision (%)     | 0.877 | 0.911 | 0.512       | 0.843        |
| Recall (%)        | 0.805 | 0.897 | 0.518       | 0.836        |

Table 4.4: Evaluation matrix of Top 4 trained model in 0528qsw dataset

| Evaluation Matrix | Model |       |              |                   |
|-------------------|-------|-------|--------------|-------------------|
|                   | VGG16 | VGG19 | Inception V3 | Inception V3 +ELM |
| F1 score (%)      | 0.898 | 0.901 | 0.901        | 0.898             |
| Precision (%)     | 0.895 | 0.893 | 0.905        | 0.898             |
| Recall (%)        | 0.911 | 0.922 | 0.913        | 0.895             |

Table 4.3 and 4.4 showed the top 4 accuracy models which were trained by using the TrashNet dataset and 0528qsw dataset respectively. Based on table 4.3, we can see that the VGG19 model which achieves the best classification accuracy in the Trash-Net dataset has also achieved the highest precision, recall, and f1 score among other models. Meanwhile, the precision, recall, and f1 score of the models in table 4.4 are nearly the same which had the values at an average of 0.9.

The results from table 4.3 and 4.4 show that the models we trained are having high precision and recall. This indicated that the performance of the trained model is good in the imbalanced classification problems.

#### 4.4 Discussion

In this section, we will do some benchmarking comparisons between the proposed framework and the state-of-the art methods on two different datasets which are the Trash-Net and 0528qsw dataset.

Table 4.5: Benchmark result with the state of arts

| Method  | Dataset  | Accuracy | Processing time (s) |
|---|----------|----------|---------------------|
| VGG-16 (C. Srinilta and S. Kanharattanachai 2019) | TrashNet | 93%      | -                   |
| ResNet+SVM (Adedeji and Wang 2019)                | TrashNet | 87%      | -                   |
| Inception ResNetV2 (C. Bircanoğlu et al. 2018)    | TrashNet | 80%      | -                   |
| DenseNet121 (C. Bircanoğlu et al. 2018)           | TrashNet | 83%      | -                   |
| AlexNet+MLP (Chu et al. 2018)                     | 0528qsw  | 90%      | -                   |
| Proposed DenseNet121+SVM                          | TrashNet | 85%      | 750.3               |
|   | 0528qsw  | 84%      | 313.4               |
| Proposed DenseNet121+ELM                          | TrashNet | 66%      | 5.4                 |
|   | 0528qsw  | 87%      | 5.9                 |
| Proposed VGG 19                                   | TrashNet | 91%      | 3956.7              |
|   | 0528qsw  | 93%      | 2985                |
| Proposed VGG19+ELM                                | TrashNet | 66%      | 4.1                 |
|   | 0528qsw  | 85%      | 3.82                |
| Proposed VGG19+SVM                                | TrashNet | 78%      | 726.1               |
|   | 0528qsw  | 94%      | 157.6               |
| Proposed Inception V3                             | TrashNet | 83%      | 3695.1              |
|   | 0528qsw  | 91%      | 2940                |
| Proposed Inception V3+ELM                         | TrashNet | 63%      | 5.132               |
|   | 0528qsw  | 90%      | 5.4                 |
| Proposed Inception V3+SVM                         | TrashNet | 80%      | 1488.6              |
|   | 0528qsw  | 94%      | 403.6               |

The author in (C. Bircanoğlu et al. 2018) stated that InceptionV2 can achieve a great performance in classifying the waste in the Trash-Net dataset into 6 different categories. The test accuracy of the model is 80% by training from scratch as shown in Table 2.1. The model is trained by using Adam as an optimizer for 100 epochs and data augmentation is done on the input images. In our experiment, we have proposed a method named InceptionV3 with the same settings and trained in TrashNet dataset. The architecture of the InceptionV3 is like InceptionV2. However, the training algorithms are different. In InceptionV3, the RMSprop, label smoothing regularizer, and an auxiliary head with batch norm is added to improve the training of the model. The testing accuracy we get for InceptionV3 in table 4.5 proved that the model is better than InceptionV2 by having an accuracy of 83%.

Next, a CNN model is built by using the transfer learning approach. The VGG-16 pre-trained model is used as a feature extractor meanwhile fully connected layers and the Softmax layer are used to classify the object. Based on Table 2.1 we can see that the author in (Costa et al. 2018) can achieve a high accuracy of 93% in classifying the waste into 6 categories by using the pre-trained VGG-16 model. However, we are only able to regenerate an accuracy of 87% in our experiment by using VGG-16 as a pre-trained model with the same

settings. The VGG-19 model proposed by us is outperforming VGG-16 by using a transfer learning approach with a test accuracy of 91% as shown in Table 4.5.

Last but not least, we have also done a comparison with the model trained in the 0528qsw dataset. A Multilayer Hybrid Deep Learning Method which is trained by using 0528qsw dataset can achieve a testing accuracy  $> 90\%$  as stated in (Chu et al. 2018). From Table 4.5, we can see that VGG-19, Inception V3, and InceptionV3+ELM can score  $>90\%$  in the 0528qsw dataset. In one particular performance metric for the waste sorting problem is the real-time implementation potential, the proposed hybrid CNN-ELM model can score a test accuracy of 90% with the training time of 5.4 seconds which is very effective for real-time process.

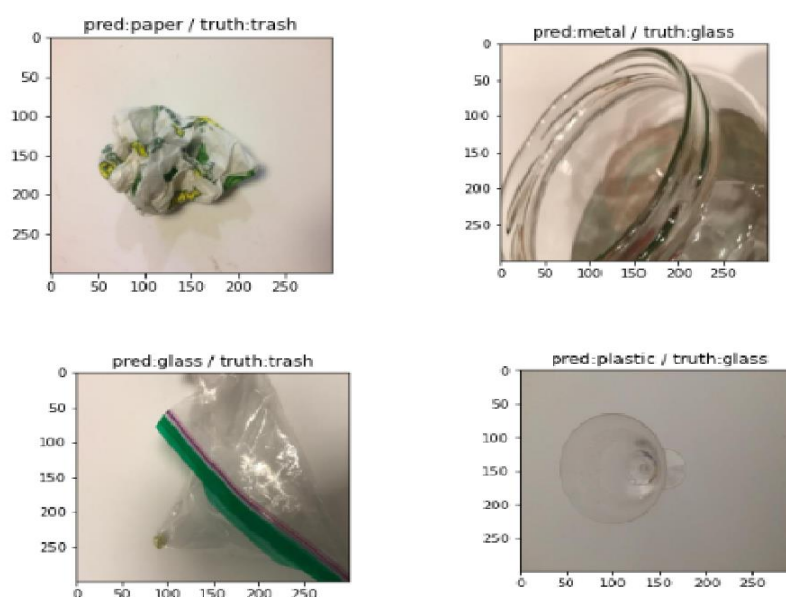


Figure 4.5: Representative of the waste item which have low classification accuracy

The example of wastes that are misclassified by our trained model is shown in figure 4.5. The trained model is unable to perform well on the waste that are lacking distinctive image features. According to the study, most of the wastes that are wrongly classified lack strong image features such as the waste in the ‘trash’ and ‘others’ class. Moreover, the waste in the cylinder shape is also easily caused ambiguity and being misclassified as a bottle.

## CHAPTER 5

### Conclusion

#### 5.1 Conclusion and Future work

In short, we have successfully identified the type of household waste in Malaysia and we categorised them into 6 different classes in our project which are the cardboard, glass, metal, paper, plastic, and other wastes.

Besides that, we have proposed and experimented two different learning algorithms to develop an intelligent vision system for waste classification which are the VGG-19 and Inception V3. In our experiment, we have also benchmarked the performance of the proposed model with other state-of-arts which are shown in table 2.1. The VGG-19 model can achieve the accuracy of 91% and 93% in the Trash-Net and 0528qsw dataset which is the highest among other methods.

Moreover, we have also proposed a novel method which is the hybrid CNN-ELM method to improve the computational efficiency of the system because the conventional CNN methods took a long time to train and predict which are not efficient for real-time usage. The performance of the hybrid CNN-ELM method is better in the 0528qsw dataset compared to the TrashNet dataset. The proposed InceptionV3 +ELM hybrid model can achieve an accuracy of 90% in 0528qsw dataset with an efficient computational time which is 720 times faster compared to the VGG-19 model.

This study demonstrates the proposed CNN models and hybrid CNN-ELM model are improving the efficiency and effectiveness of waste classification. The proposed method is both economically and environmentally beneficial to the global as the volume of waste is increasing gradually and the urgent requirements for environmentally friendly waste processing.



## 5.2 Future Recommendation

Although the results showed VGG-19 is an efficient approach to this problem compared to the hybrid CNN-ELM method, but the conventional CNN approach still tends to be more computationally expensive compared to the hybrid CNN-ELM method.

Due to time limitation, we failed to collect a large dataset which contains more varieties and classes of household waste in Malaysia. Moreover, we did not have enough time to explore more on the ELM architecture to improve the accuracy and efficiency of the hybrid CNN-ELM model. In future work, we will recommend collecting a larger size of dataset which consists of more varieties of waste because a deep learning model will perform better with a larger size of data. Besides that, we might also explore different ELM algorithms such as Multi-Layer Extreme Learning Machine (MLELM), Sparsity Extreme Learning Machine (SPELM), Incremental Extreme Learning Machine and so forth which can help to improve both the robustness and invariance of the ELM algorithm.

## REFERENCES

- Adedeji, Olugboja; Wang, Zenghui (2019): Intelligent Waste Classification System Using Deep Learning Convolutional Neural Network. In *Procedia Manufacturing* 35, pp. 607–612. DOI: 10.1016/j.promfg.2019.05.086 .
- Andreas Kölsch, Muhammad Zeshan Afzal Markus Ebbecke Marcus Liwicki (2017): Real-Time Document Image Classification using Deep CNN and Extreme Learning Machines. Available online at <http://blog.mindgarage.de/project/document%20classification/deep%20learning%20architectures/2017-11-elm>, updated on 10/31/2018, checked on 4/25/2020.
- C. Bircanoğlu; M. Atay; F. Beşer; Ö. Genç; M. A. Kızrak (2018): RecycleNet: Intelligent Waste Sorting Using Deep Neural Networks. In : 2018 Innovations in Intelligent Systems and Applications (INISTA). 2018 Innovations in Intelligent Systems and Applications (INISTA), pp. 1–7.
- C. Srinilta; S. Kanharattanachai (2019): Municipal Solid Waste Segregation with CNN. In : 2019 5th International Conference on Engineering, Applied Sciences and Technology (ICEAST). 2019 5th International Conference on Engineering, Applied Sciences and Technology (ICEAST), pp. 1–4.
- Chu, Yinghao; Huang, Chen; Xie, Xiaodan; Tan, Bohai; Kamal, Shyam; Xiong, Xiaogang (2018): Multilayer Hybrid Deep-Learning Method for Waste Classification and Recycling. In *Computational intelligence and neuroscience* 2018, p. 5060857. DOI: 10.1155/2018/5060857
- Costa, Bernardo; Bernardes, Aiko; Pereira, Julia; Zampa, Vitoria; Pereira, Vitoria; Matos, Guilherme et al. (2018): Artificial Intelligence in Automated Sorting in Trash Recycling.
- Deshpande, Adit (6/26/2019): A Beginner's Guide To Understanding Convolutional Neural Networks Part 2. Available online at <https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/>, updated on 6/26/2019, checked on 3/22/2020.

Dr. Sebastian Raschka (2020): Why is the ReLU function not differentiable at  $x=0$ ? Available online at <https://sebastianraschka.com/faq/docs/relu-derivative.html>, updated on 8/8/2020, checked on 8/22/2020.

Fully Connected Layers in Convolutional Neural Networks: The Complete Guide - MissingLink.ai (3/22/2020). Available online at <https://missinglink.ai/guides/convolutional-neural-networks/fully-connected-layers-convolutional-neural-networks-complete-guide/>, updated on 3/22/2020, checked on 3/22/2020.

He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (3/16/2016): Identity Mappings in Deep Residual Networks. Available online at <https://arxiv.org/pdf/1603.05027.pdf>.

He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015): Deep Residual Learning for Image Recognition. Available online at <https://arxiv.org/pdf/1512.03385.pdf>.

Huang, Gao; Liu, Zhuang; van der Maaten, Laurens; Weinberger, Kilian Q. (8/25/2016): Densely Connected Convolutional Networks. Available online at <https://arxiv.org/pdf/1608.06993.pdf>.

Huang, Guang-Bin; Zhu, Qin-Yu; Siew, Chee-Kheong (2006): Extreme learning machine: Theory and applications. In *Neurocomputing* 70 (1-3), pp. 489–501. DOI: 10.1016/j.neucom.2005.12.126 .

Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. (2017): ImageNet Classification with Deep Convolutional Neural Networks. In *Commun. ACM* 60 (6), pp. 84–90. DOI: 10.1145/3065386

Marcelino, Pedro (10/23/2018): Transfer learning from pre-trained models. In *Towards Data Science*, 10/23/2018. Available online at <https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751>, checked on 3/31/2020.

Md Shafiqul Islam; Maher Arebey; M. A. Hannan; H. Basri (2012): Overview for solid waste bin monitoring and collection system. In *undefined*. Available online at <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6236399>.

- Moh, Yiing Chiee; Abd Manaf, Latifah (2014): Overview of household solid waste recycling policy status and challenges in Malaysia. In *Resources, Conservation and Recycling* 82, pp. 50–61. DOI: 10.1016/j.resconrec.2013.11.004 .
- Mohd Armi Abu Samah; Lazawardi Ab. Manaf; Amimul Ahsan; Wannor Azmin Sulaiman; P. Agamuthu; Jeffrey Lawrence D'Silva (2013): Household Solid Waste Composition in Balakong City, Malaysia: Trend and Management. Available online at <https://pdfs.semanticscholar.org/7be7/cff4a8a58ec385939f27f7f66bc823e569f9.pdf>.
- Raj, Bharath (5/30/2018): A Simple Guide to the Versions of the Inception Network. In *Towards Data Science*, 5/30/2018. Available online at <https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202>, checked on 7/26/2020.
- Rizwan, Muhammad (5/25/2018): Softmax Regression. In *Muhammad Rizwan*, 5/25/2018. Available online at <https://engmrk.com/softmax-regression/>, checked on 3/23/2020.
- Saha, Sumit (12/16/2018): A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. In *Towards Data Science*, 12/16/2018. Available online at <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>, checked on 4/20/2020.
- SEPARATION-AT-SOURCE (5/22/2018). Available online at <https://www.kpkt.gov.my/separationatsource/en/>, updated on 5/22/2018, checked on 3/27/2020.
- Simonyan, Karen; Zisserman, Andrew (2014): Very Deep Convolutional Networks for Large-Scale Image Recognition. Available online at <https://arxiv.org/pdf/1409.1556.pdf%20http://arxiv.org/abs/1409.1556.pdf>.
- ThungYang-ClassificationOfTrashForRecyclabilityStatus-re. Available online at <http://cs229.stanford.edu/proj2016/report/ThungYang-ClassificationOfTrashForRecyclabilityStatus-report.pdf>.

Yang, Xiqi; Zhang, Qingfeng; Li, Zhan (2019): Contour Detection in Cassini ISS images based on Hierarchical Extreme Learning Machine and Dense Conditional Random Field. Available online at <https://arxiv.org/pdf/1908.08279>.

Yosinski, Jason; Clune, Jeff; Bengio, Yoshua; Lipson, Hod: How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems 27*. Available online at <https://arxiv.org/pdf/1411.1792.pdf>.

Zhang, Lei; Zhang, David (2015): SVM and ELM: Who Wins? Object Recognition with Deep Convolutional Features from ImageNet. Available online at <https://arxiv.org/pdf/1506.02509>.

## APPENDICES

### APPENDIX A: Python Code for Data-Preprocessing

```
#define the path of the input image
training_base_path = '../input/garythung/Dataset/Training'
validation_base_path='../input/garythung/Dataset/Validation'
testing_base_path='../input/garythung/Dataset/Test'
categories=['cardboard', 'glass', 'metal', 'paper', 'plastic', 'trash'
]
# apply image augmentation on the input image
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.1,
    zoom_range=0.1,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    vertical_flip=True,
)

test_datagen = ImageDataGenerator(
    rescale=1./255,
)

train_generator = train_datagen.flow_from_directory(
    training_base_path,
    target_size=(224, 224),
    batch_size=16,
    class_mode='categorical',

    seed=0
)

validation_generator = test_datagen.flow_from_directory(
    validation_base_path,
    target_size=(224, 224),
    batch_size=16,
    class_mode='categorical',

    seed=0
)

labels = (train_generator.class_indices)
print(labels.items())
labels = dict((v,k) for k,v in labels.items())

print(labels)
```

## APPENDIX B: Python Code to Load and Train the VGG-19 model

```

# load VGG-19 pre-trained model which excluded the classifier

vgg = VGG19(include_top = False, weights = 'imagenet', input_shape=(
224,224,3))
print(vgg.summary())
#vgg.load_weights('./input/keras-pretrained-models/vgg16/vgg16_w
eights_tf_dim_ordering_tf_kernels_notop.h5')
output = vgg.layers[-1].output
output = keras.layers.Flatten()(output)
print('Output shape of flatten layer:', output.shape)
vgg_model=Model(vgg.input,output)
vgg_model.trainable = False
for layer in vgg_model.layers:
    if layer.name == 'block5_conv1':
        break
    layer.trainable = False
## add 3 dense layer as the classifier

input_shape = vgg_model.output_shape[1]
model = Sequential()
model.add(vgg_model)
model.add(Dense(512, activation='relu', input_dim =input_shape))
model.add(Dropout(0.3))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(6, activation='softmax'))
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(loss='categorical_crossentropy', optimizer=opt, met
rics=['acc'])
model.summary()
start_time = time()
history = model.fit_generator(train_generator, epochs=100,
                             validation_data=validation_generato
r, validation_steps=30,
                             verbose=1)

```

## APPENDIX C: Python Code to Load and Train the Inception-V3 model

```

#Load pre-trained Inception-V3 pre-trained model without classifier

inception= InceptionV3(include_top = False,weights = 'imagenet',in
put_shape=(224,224,3))
print(inception.summary())
#vgg.load_weights('./input/keras-pretrained-models/vgg16/vgg16_w
eights_tf_dim_ordering_tf_kernels_notop.h5')
output = inception.layers[-1].output
output = keras.layers.Flatten()(output)
print('Output shape of flatten layer:',output.shape)
inception_model=Model(inception.input,output)
inception_model.trainable = False
for layer in inception_model.layers:
    layer.trainable = False
# add in 3 dense layer as classifier

input_shape = inception_model.output_shape[1]
model = Sequential()
model.add(inception_model)
model.add(Dense(512, activation='relu',input_dim=input_shape))
model.add(Dropout(0.3))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(6, activation='softmax'))
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['acc'])
model.summary()
start_time = time()
history = model.fit_generator(train_generator, epochs=100,
                             validation_data=validation_generato
r, validation_steps=30,
                             verbose=1)

```



## APPENDIX D: Python Code to Create and Train the Hybrid CNN-ELM model

```

# load the pre-trained CNN model without classifier

vgg = VGG19(include_top = False, weights = 'imagenet', input_shape=(
224, 224, 3))
print(vgg.summary())
#vgg.load_weights('../input/keras-pretrained-models/vgg16/vgg16_w
ights_tf_dim_ordering_tf_kernels_notop.h5')
output = vgg.layers[-1].output
output = keras.layers.Flatten()(output)
print('Output shape of flatten layer:', output.shape)
vgg_model=Model(vgg.input,output)
vgg_model.trainable = False
for layer in vgg_model.layers:
    if layer.name == 'block5_conv1':
        break
    layer.trainable = False

# extract features from the input images as the input to train in ELM

cnn_train_result=vgg_model.predict(x_train)
elm_model = hpelm.elm.ELM(cnn_train_result.shape[1], 6)
elm_model.add_neurons(500, func='sigm')
elm_model.train(cnn_train_result, target_train_oh, 'c')

```