Vegetables and Fruit Classification using Deep Convolutional Neural Network

THESIS

By

Michael Kristianus	2001560635
Amadeus Suryo Winoto	2001571670

Computer Science Global Class Program

Computer Science Study Program

School of Computer Science

Universitas Bina Nusantara

Jakarta

2020

Vegetables and Fruit Classification using Deep Convolutional Neural Network

THESIS

Proposed as one of the requirements

for obtaining Bachelor Degree at

Computer Science Study Program

Bachelor Degree

By

Michael Kristianus	2001560635
Amadeus Suryo Winoto	2001571670

Computer Science Global Class Program

Computer Science Study Program

School of Computer Science

Universitas Bina Nusantara

Jakarta

2020

Vegetables and Fruit Classification using Deep Convolutional Neural Network

Thesis

Prepared by:

Michael Kristianus	**Amadeus Suryo Winoto**
2001560635	**2001571670**

Approved by:

Supervisor and Head of Study Program

Irene Anindaputri Iswanto, S.Kom., M.Sc.Eng.
D5874

Fredy Purnomo, S.Kom., M.Kom.
Head of Computer Science Study Program

Universitas Bina Nusantara

Jakarta

2020

HALAMAN PERNYATAAN DOSEN PENGUJI

STATEMENT

Hereby I,

Name	: Michael Kristianus
NIM	: 2001560635
Thesis Title	: Vegetables and Fruit Classification using Deep

Convolutional Neural Network

Give to Universitas Bina Nusantara my **non-exclusive** rights to store, duplicate, and distribute my Thesis, as a whole or as part or just the summary, in printed or electronic format.

State that I will preserve my **exclusive** right, to use the whole or some parts of my Thesis, for future work development, such as article, book, software, or information system.

Jakarta, April 25, 2020

Michael Kristianus
2001560635

STATEMENT

Hereby I,

Name	: Amadeus Suryo Winoto
NIM	: 2001571670
Thesis Title	: Vegetables and Fruit Classification using Deep Convolutional Neural Network

Give to Universitas Bina Nusantara my **non-exclusive** rights to store, duplicate, and distribute my Thesis, as a whole or as part or just the summary, in printed or electronic format.

State that I will preserve my **exclusive** right, to use the whole or some parts of my Thesis, for future work development, such as article, book, software, or information system.

Jakarta, April 25, 2020

Amadeus Suryo Winoto
2001571670

UNIVERSITAS BINA NUSANTARA

Computer Science Global Class
Computer Science Study Program
School of Computer Science
Computer Science Bachelor Thesis
Odd Semester 2019/2020

Vegetables and Fruit Classification using Deep Convolutional Neural Network

Michael Kristianus 2001560635
Amadeus Suryo Winoto 2001571670

ABSTRACT

Artificial intelligence is one of the most developed fields in Computer Science where a lot of research has been done to make the computer smarter to perform human-like task. Computer can also recognize an object by using Convolutional Neural Network (CNN). While improving CNN model can be done by simply increasing the depth of its architecture, some research proves that as the CNN architecture go deeper, the accuracy will get worse. ResNet, with their residual layer, successfully lift the limitation, but ResNet by itself is too heavy for a mobile or portable computation device. So, this thesis proposes a new model which could reach the accuracy of ResNet while having faster prediction time. Then our model which has been trained on our own fruits and vegetables dataset will be compared to the other model. The result shows that our CNN model is as accurate as ResNet50, ResNet110, and DenseNet while being faster than those models.

Keywords : CNN, Deep Learning, Image Recognition, Artificial Intelligence

ABSTRAK

Kecerdasan buatan adalah salah satu bidang yang paling dikembangkan dalam Ilmu Komputer dimana banyak riset sudah dilakukan untuk membuat komputer lebih pintar dan dapat melakukan pekerjaan seperti manusia. Komputer juga dapat mengenali object menggunakan Convolutional Neural Network (CNN). *Model* CNN *dapat ditingkatkan dengan membuat model tersebut lebih dalam, namun beberapa riset membuktikan seiring dengan dalamnya sebuah arsitektur* CNN *akurasinya akan menjadi lebih buruk.* ResNet *membuktikan bahwa arsitektur dapat dibuat lebih dalam dengan menggunakan* residual layer, *namun* ResNet *sendiri terlalu berat untuk alat pengitung seluler dan portabel. Maka dari itu, skripsi ini mengusulkan sebuah model baru yang dapat mengenali objek sebaik* ResNet *dengan lebih cepat. Lalu model yang telah diusulkan akan dibandingkan dengan* ResNet *dan* DenseNet *yang telah dilatih menggunakan kumpulan data buah dan sayuran. Hasil dari skripsi ini membuktikan bahwa* CNN *model yang telah diusulkan memiliki akurasi sebaik* ResNet50, ResNet110, *dan* DenseNet *dengan waktu prediksi yang lebih cepat.*

Kata kunci : CNN, Deep Learning, Image Recognition, Artificial Intelligence

PREFACE

Firstly, we would like to thank the Almighty God, for giving us the time and ability to complete this thesis smoothly as our tribute for our future and many others who have helped us while writing this thesis. We also would like to thank everyone who played a major part on our life as well as in the time that we write this thesis including:

1. Prof. Dr. Ir. Harjanto Prabowo, M.M. as the Rector of Bina Nusantara University who allows us to learn as a BINUS University Student and applies BINUSIAN SPIRIT in our daily life,
2. Fredy Purnomo, S.Kom., M.Kom. as the Dean of School of Computer Science Bina Nusantara University who gives us the chance to learn by BINUS University Curriculum which have an A accreditation and good quality,
3. Dr. Derwin Suhartono, S.Kom., M.T.I. as the Head of Computer Science Study Program Bina Nusantara University who also gives us the chance to study as an exchange student for a semester,
4. Irene Anindaputri Iswanto, S.Kom., M.Sc.Eng. as our Thesis Supervisor who gives us many suggestion regarding our research and being so patient with us while doing her best for us,
5. Chintaka Premachandra, B.Sc., M.Sc., Ph.D. as our Laboratory Supervisor who supervise us while teaching us and giving us chances to learn from his past experience,
6. Our Parents, who supports us in our decision and giving us the chances on every chance that we could have,
7. Our friends who also helps and supports us since we begin our journey in Bina Nusantara University and also while we are abroad,
8. To many other people that we couldn’t include in here, we are thankful for everyone that we have met.

Lastly, we hope this thesis can be a starting checkpoint for everyone who wishes to start their research on Artificial Intelligence or unto the Image Recognition field.

Japan, 27 January 2019

Michael Kristianus

Amadeus Suryo Winoto

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ATTACHMENTS

CHAPTER 1
INTRODUCTION

1.1. Background

Currently, technologies have an important role in human daily life. Technology has aided human in various field which has helped us in doing human's job. Technology also played some important role on making the daily work easier, like help us in making agenda, reading texts, translating or summarizing them, cooking some foods, and many others. Therefore, in regard of minimizing the mistakes, the technology is getting smarter and smarter whilst learning from the correct examples and correcting the mistakes, in which they use Artificial Intelligence and the Machine Learning techniques. Mohammed, Khan, & Bashier (2017) and Smola & Vishwanathan (2008) stated that machine learning techniques are Supervised, Unsupervised, Semi-supervised and Reinforcement learning.

Based on Russell & Norvig (1995), Artificial Intelligence divided into various sub-field, such as: Neural Networks, Evolutionary Computation, Vision, Robotics, Expert Systems, Speech Processing, Natural Language Processing, Planning, and Machine Learning. Artificial Intelligence also learns like how a human learns by using machine learning. The machine learning techniques itself is said to be copying some periods or stage of human's learning life, as a child will learn using the Supervised Learning, then as a teenager will learn using the Semi-supervised Learning, and as an adult will learn using the Unsupervised Learning. Because of it, humans have been trying to teach the Artificial Intelligence how to learn, and according to the studies, the best way of studying is by imitating other people, in which, watching others, then copying what they are doing, or by visual and practice.

As a human, one of the primary necessities is food. But as this thesis is being written, mostly people still often to make the same mistake by forgetting to buy or buying something. On restocking vast amount of supplies and ingredients on periodical restocking agenda this could be a problem over time. Considering that every human life aspect already has connected to the electronic and the internet means that everyone has a smartphone nowadays. Thus the most important part of Automated Online System or widely known as Smart Fridge will be created as the result.

Previously, Smart Fridge system as in Miniaoui, Atalla, & Hashim (2019) used RFID and Barcode to classify or recognize the ingredients. Another idea as in Buzzelli, Belotti, & Schett (2018) that proposed an idea to use Convolutional Neural Network to recognize the ingredients then shows the data into the tablet that has been planted in the Fridge’s door. Both researches concludes that Automated Online Inventory System itself can be consisted of object classification, which could be the image recognition or RFID, the database, and the UI. This thesis focuses on the image recognition as it is the most important part on the Automated Online Inventory System. If the image recognition part is good enough, the user’s role to supervise the system can be minimized. And an UI system to test the Image Recognition will be created.

As the idea in Buzzelli, et al (2018) and Hou, Feng, and Wang (2017) which implemented the Convolutional Neural Network state-of-art architectures consisted of AlexNet, GoogLeNet, VGG, ResNet, NASNet to the SmartFridge. The Convolutional Neural Network itself also has other state-of-arts which is ResNet and DenseNet. As from Bianco, Cadene, Celona, and Napoletano (2018), ResNet and DenseNet have more than 90% accuracy, but, ResNet and DenseNet have some drawback in the prediction time. Because of the time limitation, a new model will be proposed. And to prove that the proposed model can compete with the state-of-art architecture, they will be evaluated using the accuracy and time along with the complexity.

1.2. Problem Statement

A fast Fruit and Vegetables Classification using Convolutional Neural Network model that can predict the object accurately will be created as the result of this thesis. In short, this research is expected to answer these questions:

1. How to implement Convolutional Neural Network to recognize fruit and vegetables?
2. Using the proposed model, how to achieve high accuracy in the fruit and vegetables recognition?
3. How to create a fast and robust Convolutional Neural Network model for recognizing fruit and vegetables?
4. How to design and create application that can help user to test the accuracy of the recognition system?

1.3. Hypotheses

The hypotheses will be divided into 4: the null hypothesis (h0), and the alternate hypotheses (h1-h3).

H0: Convolutional Neural Network model can get more than 80% accuracy with prediction time less than 42ms per image,

H1: Convolutional Neural Network model get less than 80% accuracy with prediction time less than 42ms per image,

H2: Convolutional Neural Network model can get more than 80% accuracy with prediction time more than 42ms per image,

H3: Convolutional Neural Network model can get less than 80% accuracy with prediction time more than 42ms per image.

1.4. Scope and Limitation of the Study

The scope that will be taken is classifying 11 types of fruit and vegetables which consisted of Apple, Avocado, Banana, Carrot, Garlic, Grape, Kiwi, Mango, Orange, Pineapple, and Tomato. And there is also some limitation to the study regarding the system:

1. The system viewer will be written using Python using tkinter library,
2. The classification or identifying part will be written in python,
3. The classification developed is used to detect one item in an image,
4. The system needed some library to be installed in Python, such as:
    a. Keras,
    b. Tensorflow,
    c. Pandas,
    d. Pillow,
    e. Numpy.
5. The classification can classify 11 class,
6. The system might not work under certain conditions, such as:
    a. When trying to detect more than one variance at one image,
    b. When the item is not included in the 11 class,
    c. When the input image is not square or 1:1.

1.5. Objective and Significance

1. This research's objectives are:
   a) Design a robust Image Recognition model to recognize fruits and vegetables,
   b) Design a Convolutional Neural Network which can reach high accuracy,
   c) Design a Convolutional Neural Network with fast prediction and fast computational time,
   d) Design an UI to test and analyze the result.
2. This thesis hoped to contribute in easing the human life. The findings of the study could benefit to:
   a) The future implementation

      For the future implementation, this result of this thesis could be combined with the Segmentation Head to make the model possible to detect multiple variance or objects in an image. The result of this thesis also hoped to be trained for other classification system by transfer-learning or solving other classification problem. It also hoped to help in the future development for a fast and accurate Convolutional Neural Network to be used in Automated Online Inventory System and other mobile or portable devices.
   b) The research in the same field of study

      It could be combined with the Optical Character Recognition (OCR) to make a system that could work autonomously by scanning the receipt and labeling the images or object captured by the system to make a bigger dataset to train. I suppose it could also be made as a system that could give a report on financial health periodically.

1.6. Research Methodology

This research uses some methods including data collection, building the system and evaluation. The data will be analyzed then evaluated from the scores of every part. The methodology will be:

1. Analysis and Planning

   This step analyses the needed requirement for the system including the database, the Deep Convolutional Neural Network (DCNN) model, and User Interface. Aside from analyzing the requirement, the software development along with the needed software development cycle to finish the needed software within 4 until 5 months of development and research will be planned.

2. Data Collection

   After deciding the model and the requirement, the data to be trained in CNN model will be collected. Thus in this step some images are collected as the dataset to be trained with the CNN model so that it could recognize the object as many as the dataset class has.

3. Backend Building

   In this phase, the backend using the python language will be build. This phase contain image recognition and will be a CNN Model which will be built and trained manually later on.

4. Building UI for model viewer

   In this phase, the UI to show the result of the image classification will be made by using tkinter library within python.

5. Final Testing and Thesis Writing

   In this phase the whole system will be tested to see whether the system already working correctly or not. After the testing is done, the thesis writing about the project will be begun.

1.7. Writing Structure

This thesis is written using this structure to explain about this research and what of each chapter will consist of.

Chapter 1 Introduction

This chapter consists of background, problem statement, hypotheses, scope and limitation of the study, objective and significance, research methodology, and writing structure.

Chapter 2 Literature Review

This chapter is explaining some of the previous similar research, the general concept or theories that support this research based on journals or papers.

Chapter 3 Methodology

This chapter is deeply shows the steps to build this system, and how to answer or prove the hypothesis. There will also be the system design, and ideas.

Chapter 4 Analysis and Result

In this chapter, all of the result from the system will be discussed, including the reason which will be justifies by the data in form of tables, graphic, and images.

Chapter 5 Conclusion and Suggestion

The conclusion in regards of the hypotheses are going to be written in here, and suggestion to the next or future works that still on the same work or topic.

CHAPTER 2
LITERATURE REVIEW

2.1 Current Implementation and Development

Currently, many big companies already tried to produce a smart fridge (Luo, Jin, & Li, 2009). Current smart fridge is almost the same with the normal fridge, but it has a tablet or display that already planted in the fridge's door. The smart fridge also could detect what inside it using an RFID tag or Barcode which firstly scanned before the item got in (Miniaoui et al., 2019). Beside the idea of the RDIF tag and Barcode Scanner, there are also some ideas on applying Convolutional Neural Network (CNN) inside the Smart Fridge. Those implementations of fruits and vegetables recognition is using hierarchical labeling and appending the convolutional layers (Buzzelli et al., 2018). The implementation of NASNet and hierarchical labeling works better than the other state-of-art CNNs which is ResNet, VGG, GoogLeNet, and AlexNet along with the HybridNet (Hou et al., 2017).

2.2 Artificial Intelligence

Artificial Intelligence (AI) is part of Computer Science area which develop machine to have human like intelligence so the machine could make decision, solving problem, learning, and to make a future prediction. Today's Artificial Intelligence already has so many subfields with so many activities. Example of Artificial Intelligence Practice can be divided into following subfields: Robotic or Autonomous Vehicles which is driverless robotic vehicle also known as self-driving car, Autonomous planning and scheduling which is used by NASA Remote Agent Program to control scheduling of operation for a spacecraft, Game playing which is design of AI to make computer be able play the game without human interaction, Speech Recognition, Image Recognition, and many other fields (Russell & Norvig, 1995). Some branch fields of Artificial intelligence used in this thesis and can be defined as follow:

2.2.1 Machine Learning

Based on Mohammed et al. (2017), Machine Learning is a sub category of Artificial Inteligence which enables computer to learn by itself to perform their job skillfully. But in reality, the computer actually still needed human on learning the best way and to extract and get the rules and conditions. Mohammed et al.

(2017), Smola & Vishwanathan (2008), and Saravanan & Sujatha (2018) also stated that the general Machine Learning Methods or Techniques is divided into 4 types: Supervised; Unsupervised; Semi-supervised; and Reinforcement Learning, in which types are differ one another.

2.2.1.1. Supervised Learning

Supervised learning technique utilizes previous information and knowledge with help of label or template to make decision and forecast events. Supervised learning has input variable and output variable. It called supervised learning because in the learning process training data set is being observed and taught the correct output.

2.2.1.2. Unsupervised Learning

Unsupervised learning is a technique without using label and template and training data is not classified. This technique requires recognizing pattern in data set and clustering the training data. Unsupervised learning has input variable but do not have output variable.

2.2.1.3. Semi-supervised Learning

Semi-supervised learning is a combination technique between supervised and unsupervised learning where input variable is greater quantity than output variable. This learning technique use both labeled and unlabeled data for training process. This technique can attain higher accuracy.

2.2.1.4. Reinforcement Learning

Reinforced learning technique is based on Markov decision process. This learning technique should obtain maximum output by trial and error based on information gathered from environment. This technique interacts with environment by take action and locates error or rewards. By trial, error search and delayed reward it could enable system to identify the ideal behavior to increase accuracy when decision making.

2.2.2 Computer Vision

Computer vision is a system processing image acquired from the sensor like a camera, which is similar with human vision system where the brain process image derived from the eyes (Nixon & Aguado, 2012). Computer vision aim to make computer or machine gain the same system as human visual system by gaining some understanding level from image or video captured. Human can perceive three-dimensional structure with ease and can tell shape and translucency through subtle pattern of light and shading. Facial expression makes the computer to be able to differentiate people and can even guess their emotion. Szeliski (2010) stated that many studies try to understand how the visual system work, but a complete solution remains elusive. Until today, computer vision already used in a wide variety of real-world application, such as:

a) Optical Character Recognition (OCR): OCR is a computer vision field that specifically recognizes text in an image. For examples OCR is implemented in reading handwritten text, automatic number plate recognition (ANPR), and e-ticketing.
b) Retail: Object recognition for automated checkout lanes.
c) Match moves: merging Computer Generated Imagery (CGI) with real live footage to track feature point to estimate camera motion.
d) Biometrics: For authentication and forensic application. Implemented in Fingerprint recognition, face recognition, and voice recognition.
e) Stitching: turning overlapping photo into single panorama.
f) Exposure bracketing: merging multiple exposures taken under challenging lightning condition into single exposure image.
g) 3D modeling: converting one or more image into a 3D model of the image.

2.2.3 Artificial Neural Network

Other popular field of AI today is Artificial Neural Network (ANN). On the contrary from most people think, Artificial Neural Network rather aged. According to Russell & Norvig (2010) and Hagan, Demuth, & Beale (2014) the origin of the neural network began in the 1940s with the work of Warren McCulloch and Walter Pitts, but interest in neural network falter during late

1960s because of the lack of innovation and resource, but then research in neural networks increased adequately in mid 1980s. The use of statistical mechanics to define operation of recurrent network class to be used as an associative memory and several different groups reinvented of back propagation algorithm for training multilayer perceptron networks first invented in 1969 by Bryson and Ho.

ANN inspired by biological neural network of human brain. There are two key similarities which are building blocks of both networks are simple computational devices and connection between neurons determines the function of the network (Hagan et al., 2014). Model and depth of ANN varies depending on approach and practical purposes. One of ANN subfield for image recognition that used in this thesis is Convolutional Neural Network (CNN).

I. Convolutional Neural Network

Convolutional Neural Network (CNN) is subfields of ANN used for various purpose such as image and video processing, image recognition, speech recognition, natural language processing, etc. CNN is different from conventional ANN as the input of the image directly goes to the network and its layers are chosen so they spatially match the input data, so it avoids feature extraction and data reconstruction process in conventional algorithm (Hou, Wu, Sun, Yang, & Li, 2016). CNN architecture consists of input layers, multiple hidden layer, and output layers. According to Sharma, Singh, Sudhakaran, and Verma (2019), some of the hidden layers are classified as:

- Convolutional Layers: Convolutional layers learn weight and biases within input fields which produces representations to use the information of local spatial structures. These layers have a small receptive field which extends through entire input depth. Convolutional operation will enhance the original signal features and reduce noise.
- Pooling Layers: Pooling layers is a form of linear down-sampling and provides spatial invariance. These layers are using the principle of correlation to reduce amount of data processing while preserving useful information. Conventional pooling methods used are max pooling and average pooling.
- Fully Connected Layers: These layers are regular dense ANNs and used before the output layer.

- Output Layers: Output layers for classification problems commonly use the Softmax classifier or function as below.

Currently, there are some newer layers that used to improve the result and accuracy which are:

- Batch Normalization Layer: Ioffe & Szegedy (2018) stated that batch normalization layer is a layer that normalizes the input and as a regularizer, which could be lessen the usage of dropout layer. Higher learning rates and lowering the number of epoch needed to reach the peak could also be used (Ioffe & Szegedy, 2018).
- Dropout Layers: Dropout layer actually removes or dropping layer (not training a layer) by a certain point to regularize the hidden layer as to remove noises with less computation (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014).

Based on Nwankpa, Ijomah, Gachagan, & Marshall (2018), convolutional architecture also have some activation functions to increase the accuracy while performs various computation inside the neural network, some of them are Sigmoid, TanH, Softmax, Softsign, Rectified Linear Unit (ReLU), Softplus, Exponential Linear Unit (ELU), Maxout, Swish, and ELiSH function. The commonly used are ReLU activation function in the hidden layer, while Softmax function are used in the output layer, thus ReLU and Softmax function as it is widely used (Nwankpa et al., 2018) is explained below.

- ReLU Function: The ReLU function is a linear function that utilizes eq. 2.1. This function make sure that there is no negative function as it only takes the parameter *x* if *x* is positive. Or it will set *x* to 0 if *x* is negative. While ReLU avoid the negative value, Leaky ReLU (LReLU) as written by Maas, Hannun, & Ng (2013) utilizes eq. 2.2, where *a* uses a very small constant value as small as 0.01, which still gives a very small negative value.

$$f(x) = \max(0, x) \tag{2.1}$$

$$f(x) = \begin{cases} x, & x > 0 \\ ax, & x \leq 0 \end{cases} \tag{2.2}$$

- Softmax Function: The Softmax function is actually counting the probability of each class with the total sum of the probabilities as one. Equation 2.3 shows how to get the probability by counting the exponents of each output divided by the sum of all result.

$$f(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{j} e^{x_k}} \tag{2.3}$$

Some Convolutional Neural Network architecture model will be explained below:

i.) Residual Network

Residual Network or ResNet is a neural network architecture that implements Residual Learning. Deeper neural networks known to be difficult to train, although stacked number of layers can improve levels of feature. Increasing depth of a network might get accuracy to decrease where such problem is not caused by overfitting (He, Zhang, Ren, & Sun, 2015). Figure 2.1 below will represent that deeper layer will have more error.

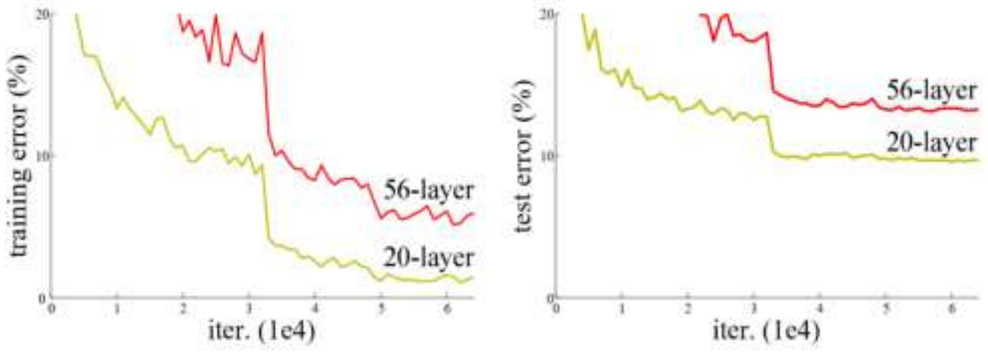


Figure 2.1 Training Error (Left) and Test Error (Right) on CIFAR-10. Retrieved from He, Zhang, Ren, & Sun (2015)

To settle those problems, He, Zhang, Ren and Sun (2015) propose a method called deep residual learning framework. Previously, depth of DCNN are capped from 16 to 19 layers, but residual learning can help in achieving deeper network while still producing high performance for image classification and image segmentation (Li & He, 2018). He, Zhang, Ren, and Sun (2015) stated that residual mapping is easier to optimize rather than normal, unreferenced mapping because if an identity mapping is optimal, it would be easier to push the residual to zero rather than fitting an identity mapping to a stack of nonlinear layers. This Residual Learning block can be seen in Figure 2.2 below.

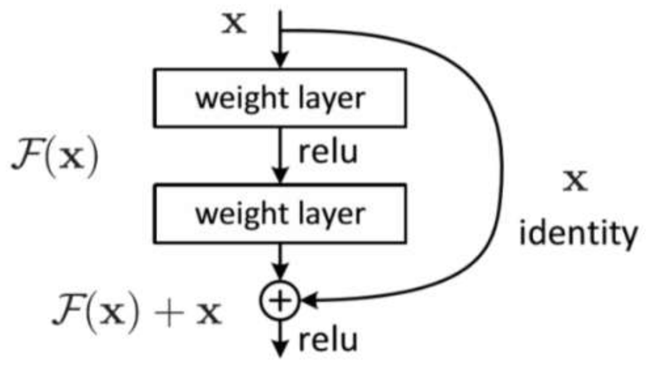

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2.2 Residual Learning: a Building Block. Retrieved from He, Zhang, Ren, & Sun (2015)

ii.) You Only Look Once (YOLO)

YOLO is a neural network architecture that focuses on detection speed for real-time image recognition. Redmon, Divvala, Girshick and Farhadi (2016) state that YOLO is more advantageous than older model in few aspects such as: speed, lower background error, and endurance. Those aspects are supported by the fact that YOLO only use a single network as a pipeline which make it very fast, YOLO also can see image as a whole when making a prediction which could make it understand contextual information to reduce background error, and YOLO is highly generalizable resulting in lower chance of breaking when handling unexpected input, nevertheless, the accuracy of YOLO still falls behind and struggling to detect small object. YOLO architecture can be seen in Figure 2.3.

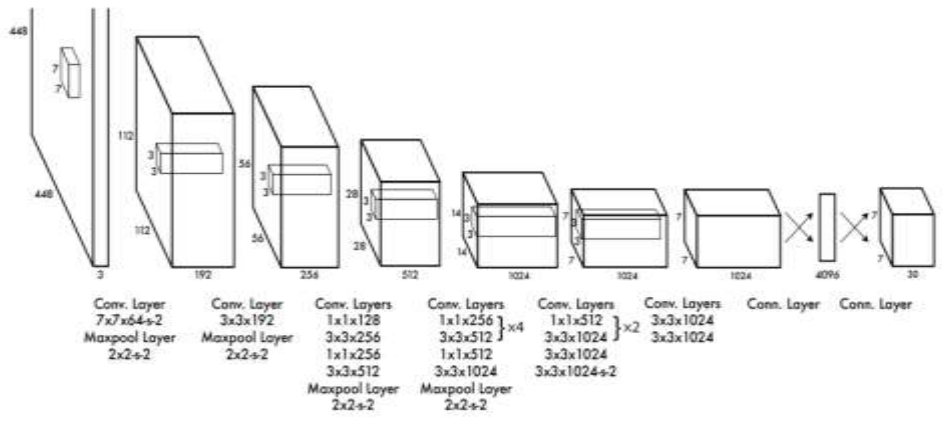
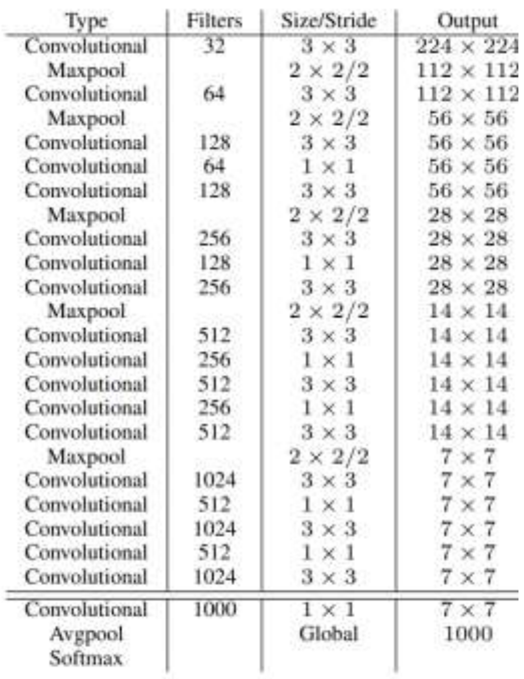


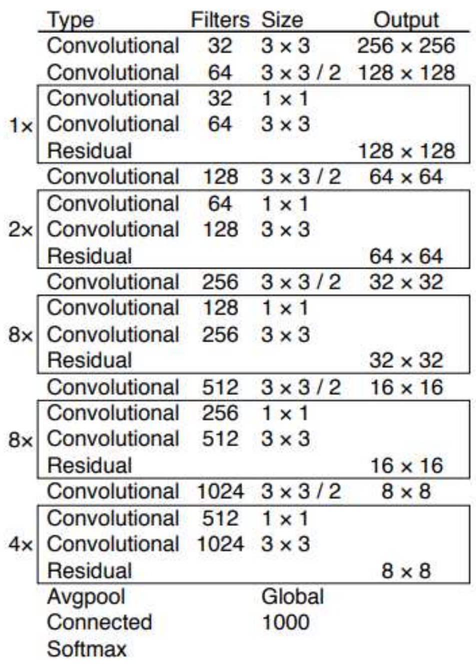
Figure 2.3 YOLO Architecture. Retrieved from: Redmon, Divvala, Girshick, & Farhadi (2016)

In his work YOLO implement Darknet framework as its backbone. In the first version they add four convolutional layers and two fully connected layers, then in last layer make class probabilities prediction and find bounding box coordinates (Redmon, Divvala, Girshick and Farhadi, 2016). In their second version, Redmon and Farhadi (2017) proposed a new model which is similar to VGG models and they use global average pooling to make prediction, compress the feature representation, batch normalization to stabilize training, speed up convergence, and regularize model. The final model has 19 convolutional layers and 5 maxpooling layers. This newly proposed Darknet model named after the number of its convolutional layers which is Darknet-19 and the detail can be seen in Figure 2.4.



Type	Filters	Size/Stride	Output
Convolutional	32	3 × 3	224 × 224
Maxpool		2 × 2/2	112 × 112
Convolutional	64	3 × 3	112 × 112
Maxpool		2 × 2/2	56 × 56
Convolutional	128	3 × 3	56 × 56
Convolutional	64	1 × 1	56 × 56
Convolutional	128	3 × 3	56 × 56
Maxpool		2 × 2/2	28 × 28
Convolutional	256	3 × 3	28 × 28
Convolutional	128	1 × 1	28 × 28
Convolutional	256	3 × 3	28 × 28
Maxpool		2 × 2/2	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Maxpool		2 × 2/2	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	1000	1 × 1	7 × 7
Avgpool		Global	1000
Softmax			

Figure 2.4 DarkNet-19. Retrieved from Redmon & Farhadi (2017)



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

Figure 2.5 DarkNet-53. Retrieved from Redmon & Farhadi (2018)

In the third version, Redmon and Farhadi (2018) improve their network by adding Residual learning and become remarkably deeper. It has 53 convolutional layers and like the previous network naming manner, they call it Darknet-53. Detail architecture of Darknet-53 can be seen in Figure 2.5. Redmon and Farhadi (2018) also stated that Darknet-53 performance is similar to ResNet model but with fewer floating-point operation which is make it more efficient and faster.

iii.) InceptionNet or GoogLeNet

Szegedy et al. (2015) state that the most convenient way to improve neural network performance is by enlarging its size. By size, it could mean neural network depth which is number of levels or width which is number of units in each level. Bigger network size indicates a larger number of parameters which could also imply the network will prone to overfitting. The original Inception module is shown in Figure 2.6

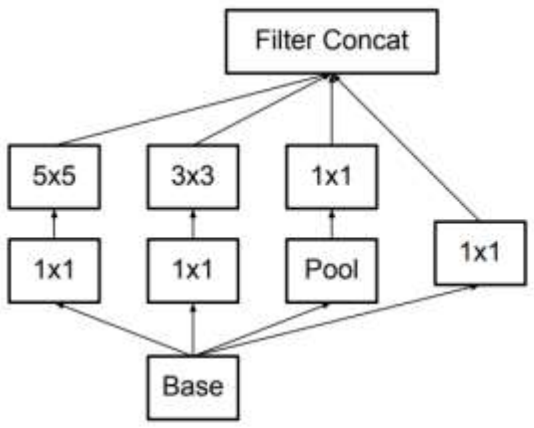


Figure 2.6 Original Inception Module. Retrieved from Szegedy, Vanhoucke, Ioffe, Shlens, and Wojna (2016)

In the third version Szegedy, Vanhoucke, Ioffe, Shlens, and Wojna (2016) propose an idea to improve the network by using factorizing Convolutions. Factorization can lessen computational cost by reduce the number of parameters while maintain the network efficiency. Factorizing convolutions can be performed by replacing a convolution layer with a bigger kernel to multiple convolution layers with smaller kernel. The example of factorization convolutions can be seen in Figure 2.7 and for spatial factorization into Asymmetric Convolutions in Figure 2.8.

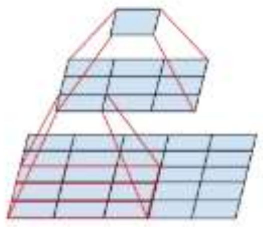


Figure 2.7 Factorizing Convolutions Replacing 5x5 Convolutions. Retrieved from Szegedy et al. (2016)

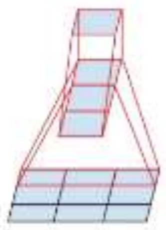


Figure 2.8 Replacing 3x3 Convolutions into Asymmetric Convolutions. Retrieved from Szegedy et al. (2016)

In Figure 2.7, 5x5 convolutions will have small fully connected layers as each output, and by replacing the fully connected by two-layer convolutional architecture which is first, a 3x3 convolution and then fully connected layer on top of the first layers. By sliding this into input activation grid will replace the 5x5 convolution with two layers of 3x3 convolution as seen in Figure 2.9.

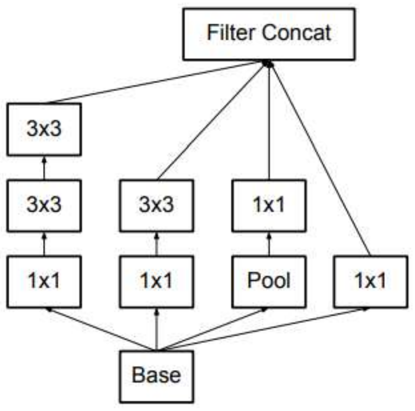


Figure 2.9 Factorized Inception Modules. Retrieved from Szegedy et al. (2016)

Szegedy et al. (2016) state that this theory could be extended even further and still could be used even the size of kernel is big as this can replace any $n \times n$ convolution. By replace any $n \times n$ convolution using asymmetric convolution as shown in Figure 2.8 to be a $1 \times n$ convolution followed by a $n \times 1$ convolution it will save computational cost significantly. Factorization will give very good result on medium grid-size when m in $m \times m$ feature maps in range 12 and 20.

iv.) Densely Connected Convolutional Networks

Densely Connected Convolutional Network or more known as DenseNet architecture utilize short path characteristic that has been implemented in many network topologies. Huang, Liu, Maaten, and Weinberger (2017) propose a simple connectivity pattern with short path characteristic to establish maximum information flow in the network by connect each layer directly. This means, DenseNet preserve feed-forward nature by append all preceding inputs to the result of current layer. DenseNet and ResNet might look similar but DenseNet always concatenate the feature before passed into a layer rather than adding them like ResNet. Example of DenseNet with 5 dense blocks can be seen in Figure 2.10.

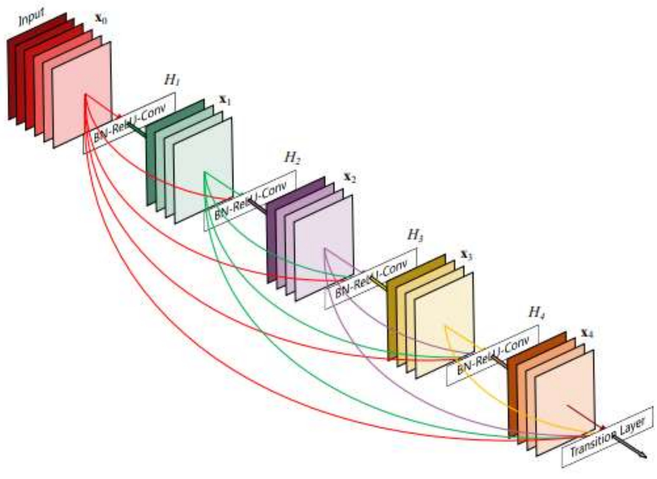


Figure 2.10 5-layer Dense Block. Retrieved from Huang, Liu, Maaten, Weinberger (2017)

2.3 Smart Fridge

Currently research on smart home is blooming. In some developed countries, they offer home with built in smart technologies. Significant development in this area will be seen and in the future will make a major impact on daily life. Developing smart applicant is critical step in creating smart home. And one of most important places in the house is its kitchen. Inside the kitchen there will be many house appliances that can be upgrade with intelligence system with aim to provide a better service (Luo, Xia, Gao, Jin, & Athauda, 2008).

Most of food waste is come from refrigerator because of forgotten stock or mismanagement of fridge inventory as buying too much unneeded product which leading to spoiled product and go waste (Miniaoui et al., 2019). This research is conducted to prevent such thing to happen by making a smart fridge with manageable online inventory. Most smart fridge researches are utilizing the RFID technology. RFID or Radio Frequency Identification belongs to the Auto – ID family and its purpose is to identify object. Examples of Auto – ID are smart card, OCR, biometric procedure, magnetic card, barcodes, and RFID. RFID usually using tag or label which contains a unique serial identification number to ensure the identity of the associated object. This thesis aimed to make a smart fridge that can self-identifying and cluster each object inside it (Hachani, Barouni, Said, Amamou, 2016).

2.4 Unified Modeling Language Diagram

Based on Satzinger, Jackson, and Burd (2014) Unified Modeling Language (UML) diagram is a graphical way to make software documentation. UML diagram are consisted of many diagrams such as class diagram, object diagram, component diagram, composite structure diagram, use case diagram, sequence diagram, communication diagram, state diagram, activity diagram, deployment diagram, package diagram, timing diagram, interaction overview diagram (Satzinger et al., 2014). But as for this thesis, only use case diagram and sequence diagram that are used. The definition of use case diagram and sequence diagram as written by Satzinger et al., (2014) are:

2.4.1. Use Case Diagram

Use case is depicted with solid line ellipse with the name inside of it. Use case diagram itself consisted of use case and actor, where usually use case is connected to an actor. Use case diagram are used to describe a sequence of actions that performed within a system which shows an output to particular actor. Use case diagram also shows the static view of a system along with the relationship between an actor and use case. Use cases notation can be seen on Figure 2.1

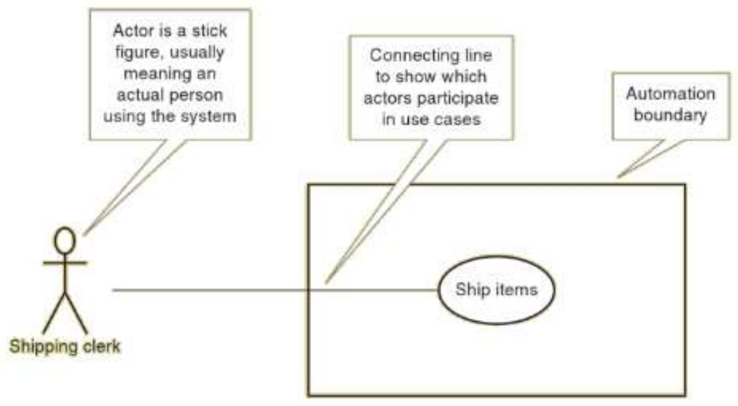

Actor is a stick figure, usually meaning an actual person using the system
Connecting line to show which actors participate in use cases
Automation boundary
Ship items
Shipping clerk

Figure 2.11 Use case notation (Satzinger et al., 2014)

Example of use case diagram is also shown in following Figure 2.12, where the customer as an actor, and 4 use cases which are fill shopping cart, search for item, view accessory combinations, and view product comments and ratings. There is association between customer and fill shopping cart which indicate participation of actor in the use case. Fill shopping cart has include arrow to search for item, view product comments and ratings, and view accessory combination use case which indicate the behavior of those 3 use case is included in fill shopping cart.

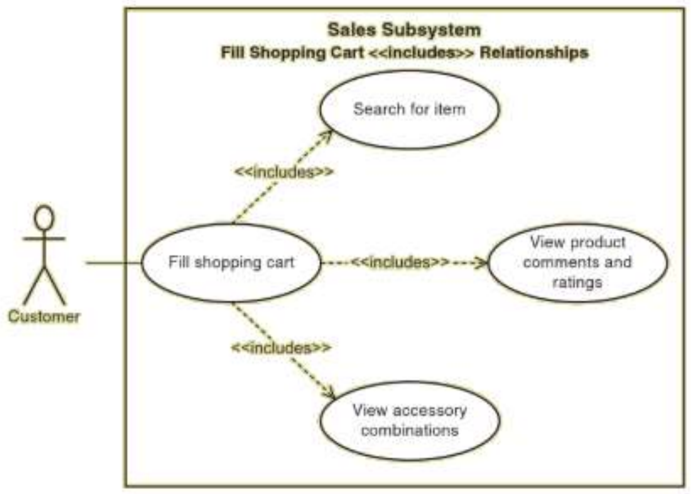

Sales Subsystem
Fill Shopping Cart <<includes>> Relationships
Search for item
<<includes>>
Customer
Fill shopping cart
<<includes>>
View product comments and ratings
<<includes>>
View accessory combinations

Figure 2.12 Use Case Example (Satzinger et al., 2014)

2.4.2. Activity Diagram

The purpose of activity diagram is to present graphical flow of interaction in a system. Activity diagram use rounded rectangles to define function in a system, arrow to present the flow in the system, diamond which imply decision node usually used to imply if-condition with each of the arrow appeared from diamond is labeled, and horizontal line to define parallel activities happening. Activity diagram notation is portrayed in Figure 2.13.

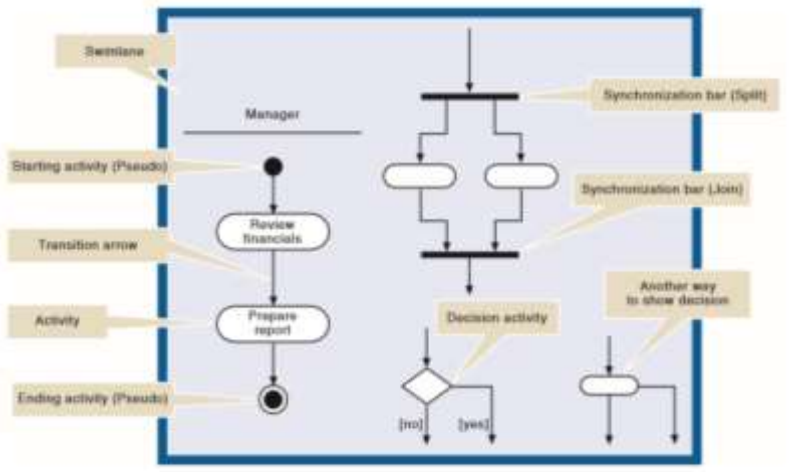


Figure 2.13 Activity Diagram Notation (Satzinger et al., 2014)

Example of swim lane activity diagram can be seen in Figure 2.14 below. This example is about returning book procedure in library. This activity diagram is involving two actor which are library employee and system.

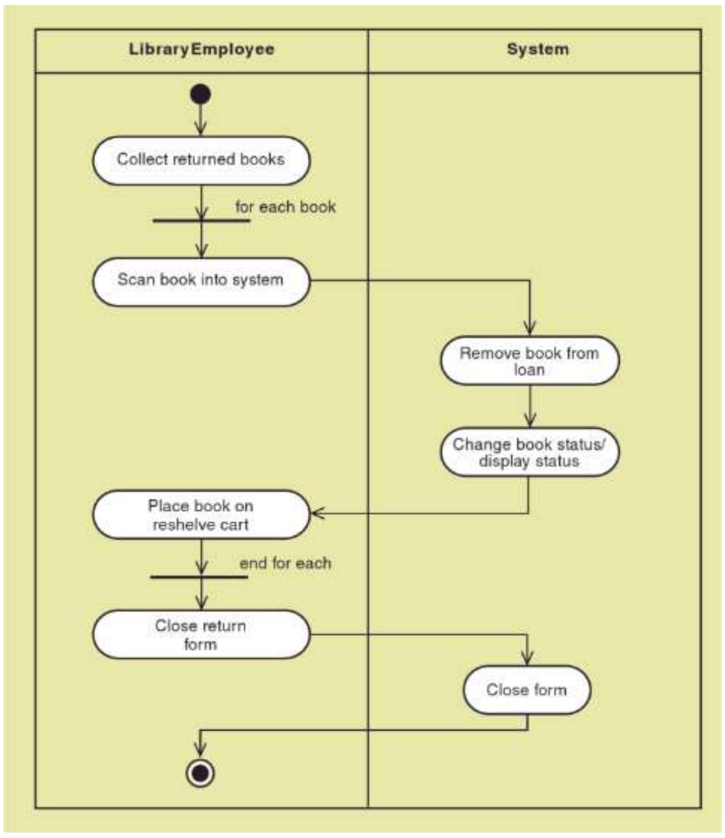


Figure 2.14 Example of Swim Lane Activity Diagram (Satzinger et al., 2014)

2.4.3. Sequence Diagram

Sequence diagram is a diagram that highlights the order of an action along with the object that does it. It shows the object that interacted with a certain activity along the x axis while the time will be shown in the y axis. The object on the leftmost is the initiator while the object in the right is the object that related to the action that currently runs. Figure 2.15 is an example of sequence notation.

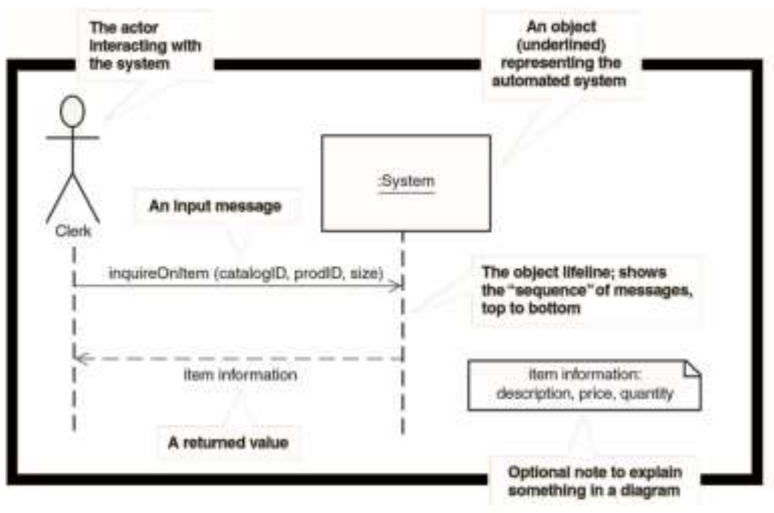

The actor interacting with the system
An object (underlined) representing the automated system
:System
Clerk
An input message
inquireOnItem (catalogID, prodID, size)
The object lifeline; shows the "sequence" of messages, top to bottom
item information
item information: description, price, quantity
A returned value
Optional note to explain something in a diagram

Figure 2.15 Sequence Diagram Notation (Satzinger et al., 2014)

2.4.4. Class Diagram

Class diagram is type of structure diagram for describe structure of the system by depict the classes, attributes, methods or function, and relationship among object. Purpose of class diagram is to show static structure of the system and provide basic notation. Class diagram notation can be seen in Figure 2.16

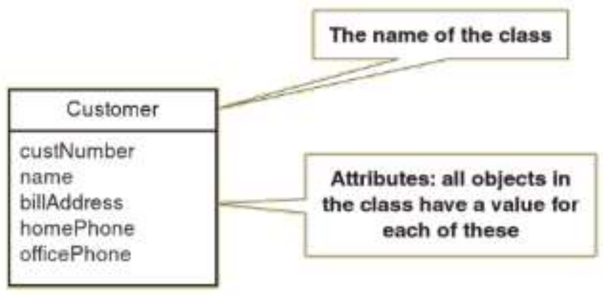


Figure 2.16 class diagram notation (Satzinger et al., 2014)

Example of aggregation in class diagram can be seen in Figure 2.17, where it has 6 class which is the computer as main class and another five classes which are processor, main memory, keyboard, disk storage, and monitor as part of the main class.

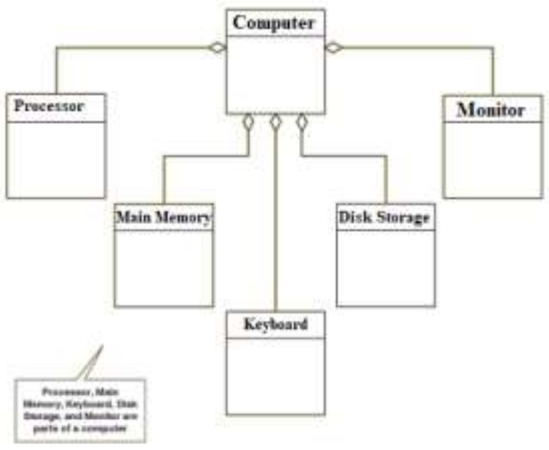

Computer
Processor
Monitor
Main Memory
Disk Storage
Keyboard
Processor, Main Memory, Keyboard, Disk Storage, and Monitor are parts of a computer

Figure 2.17 Class aggregate retrieved from Satzinger et al., (2014)

Example of class diagram can be seen in Figure 2.18, where the example is about relationship between customer class, account class, branch class and transaction class.

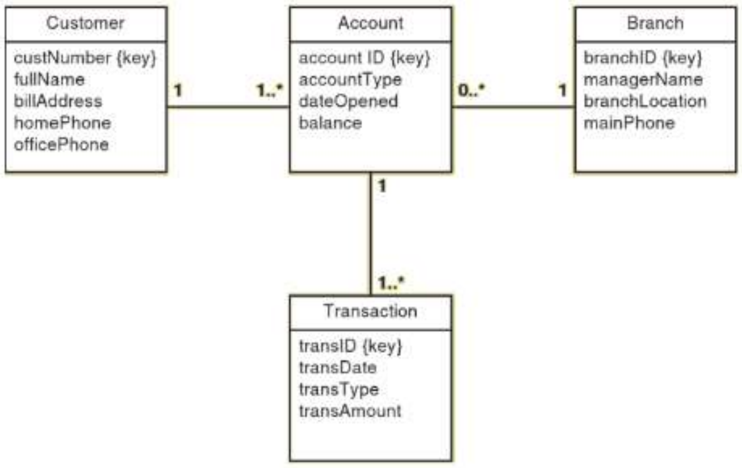


Figure 2.18 Example of Class diagram (Pressman & Maxim, 2015)

2.5 Python

Python is an open source programming language and widely used for machine learning, computer vision, image processing, shell script and even video game. There are two versions of Python, which is Python2 and Python3, where python 3 is now the main development version. In this thesis python is used to develop the proposed Deep Convolutional Neural Network model and User Interface application (Jackson, 2018). And following library is used to develop the application:

2.5.1. Tkinter

Tkinter is standard python library used to make a graphical user interface. Tkinter is commonly used in Python and mostly available in Unix, but also can be used on another platform as well such as Windows and Macintosh (Python org, 2019). Tkinter can be used by simply import the module and widget used will be explained as follow:

i. Button Widget

The button Widget can be used to implement many kinds of button such as plain buttons, check buttons, and radio buttons. Buttons can contain text or images and can be associated with function or method so the time the buttons is pressed, it will call that function or method (Moore, 2018).

ii. Canvas Widget

Canvas Widget is mainly used to display, edit graph or drawing, draw graph and plots, create graphic editor and create customize widget. Displaying image using canvas will override the previous image and show new image by default (Moore, 2018).

iii. Label Widget

Label widget can be used to display text or image. Display text using label do not have formatted text feature, which mean label can only display the text with the same font, but the text can have more than one line and one of the characters can be underlined. Label can update its value any time needed (Moore, 2018).

iv. List box Widget

List box widget is used to display a list of text that can be configured to choose one or more list. List box can only contain text with the same format (Moore, 2018).

v. Text Widget

Text widget have feature to store and display text with configurable format. Configure format allow to display text with customize style and attributes (Moore, 2018).

vi. Frame widget

Frame widget is used to padding, organize and put together other widget, and be a foundation to build a complex combination of widget (Moore, 2018).

vii. Menu widget

Menu widget contain toplevel menu, pulldown menu and pop up menu. Most common menu used is toplevel menu, where menu bar is displayed right below the title (Moore, 2018).

viii. Scrollbar widget

This scrollbar widget is used to implement scrollable listboxes, canvases, and textfields. This widget allows the widget to have slide controller that can create either vertical or horizontal scrollbars (Moore, 2018).

ix. Dialog Widget

Standard dialog can display message box to select files, colors and ask for user input. Dialog widget can also be used as file dialog where it can create file/directory selections windows (Moore, 2018).

2.5.2. Tensorflow

Tensorflow is an open source python library that used mainly for machine learning. Tensorflow provide all necessary means to develop model easily in any language or platform. For example, Tensorflow Extended for creating and manage production pipeline, Tensorflow Lite for mobile and edge devices and Tensorflow.js for Java Script environment (TensorFlow, n.d).

2.5.3. Keras

Keras is API for neural network development. Keras is written in python and run on top of TensorFlow, CNTK, or Theano. Keras focus on allowing easy and fast prototyping, support convolutional network and recurrent network or hybrid between those two, and able to run on CPU and GPU. Keras is compatible either using python 2 or python 3 (Keras, n.d.).

2.5.4. Pandas

Pandas is open source python library that famous for data analysis. Pandas is fast and efficient Data frame object for data manipulation, tools for reading and writing between in-memory data structure and different formats, flexible for reshaping and pivoting datasets and so on (Pandas, n.d).

2.5.5. NumPy

NumPy is library in python used for scientific computing. NumPy is known as array processing that able to process multi dimension array, useful linear algebra, Fourier transform, random number, and also very efficient for multi-dimensional container of generic data (NumPy, n.d.).

2.5.6. Pillow

Pillow is fork of PIL which is open source python library for image manipulating, reading, and writing. PIL itself is acronym for Python Image Library. Although, Pillow is fork of PIL, these two cannot coexist together in the same environments. Previously, Pillow support both Python 3 and Python 2 up to its version 6, but after the version 7, Pillow only supports python 3 (Clark, n.d.) .

2.6 Gantt Chart

Henry Gantt develop chart to help visualize and understand data better and this chart is more known as Gantt Chart. Gantt Chart original purpose is to show activities planning and timeline which tends to be retrospective and diagnostic. (Weaver, 2012)

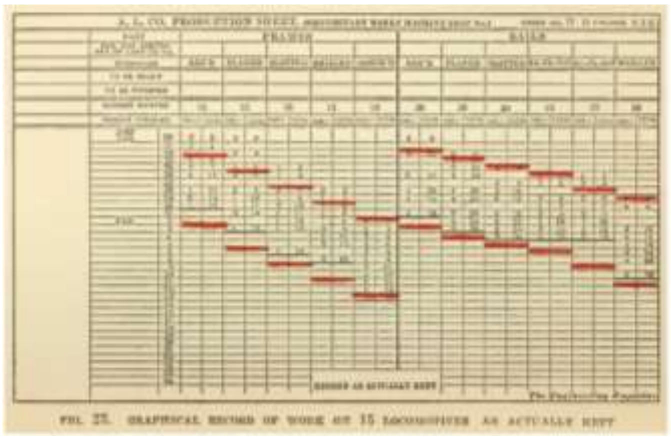


Figure 2.19 Gantt Chart Example

CHAPTER 3
METHODOLOGY

3.1. Theoretical Framework

By the time this thesis is written, there are many research, paper and journal which show that Convolutional Neural Network (CNN) is the best from all Neural Network in recognizing an object (Gu et al., 2017 and Sultana, Sufian, & Dutta, 2018). And stated before, this method focuses on answering the problem statement as stated in Figure 3.1. Firstly, CNN will be used to classify the Fruits and Vegetables. Secondly, the model will be trained to get a high accuracy. Then the CNN also will be improved to make it have a faster Computational Time. Lastly, the CNN will be implemented in the application to ease the usability to test the time of each model's prediction.

CNN is actually used much as a classification method. CNN accuracy can also be increased by making it deeper, but CNN itself have a limitation on how deep it could be. Thus ResNet and DenseNet are made in regards of solving that problem and make the limit deeper, which is either by concatenating or adding the input into the output layer of each hidden layer, which later on they are classified as Deep Convolutionl Neural Network (DCNN). DCNN is also going to be used as the method of Fruit and Vegetable Classification. The system itself will try to identify 11 fruits and vegetables which are: apples, avocados, bananas, carrots, garlic, grapes, kiwis, mangos, oranges, pineapples, tomatoes.

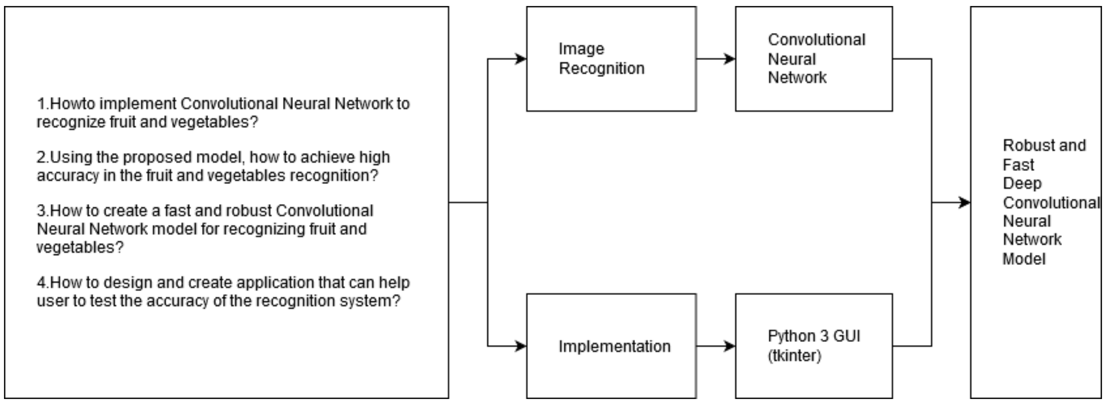

1.Howto implement Convolutional Neural Network to recognize fruit and vegetables?
2.Using the proposed model, how to achieve high accuracy in the fruit and vegetables recognition?
3.How to create a fast and robust Convolutional Neural Network model for recognizing fruit and vegetables?
4.How to design and create application that can help user to test the accuracy of the recognition system?
Image Recognition
Convolutional Neural Network
Implementation
Python 3 GUI (tkinter)
Robust and Fast Deep Convolutional Neural Network Model

Figure 3.1 Theoretical Framework Diagram

3.2. Research Methodology

This research will be executed by following these steps and procedures. Started by analysis, literature review, and planning, followed by data collection and backend building, then UI building, and finally testing and writing the report as stated in Figure 3.2.

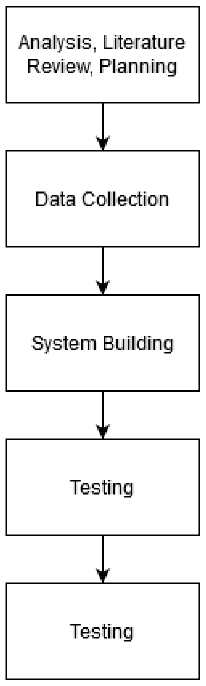


Figure 3.2 Research Method

3.2.1. Analysis, Literature Review and Planning

This step analyzes the needed requirement for the system, the Convolutional Neural Network (CNN) model, and the UI. For classifying the Vegetables and Fruits, Convolutional Neural Network itself has a limitation because of the accuracy loss which can be caused by over fitting or under fitting. But that problem can be solved by using Deep Convolutional Neural Network. Deep Convolutional Neural Network such as ResNet and DenseNet are solving the problem by either adding or concatenating the input to the output. Thus ResNet and DenseNet are taken as a comparison and goal for the accuracy.

Aside from analyzing the requirement, planning the software development along with the needed software development cycle to finish the needed software within 4 until 5 months of development and research which estimated based on the Literature Review which gives us the result of the Gantt chart as in Table 3.1.

Table 3.1 Workflow Cycle Gantt Chart

	September				October					November					December					January			
	1	2	3	4	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4
Analysis and Planning	🟥	🟥	🟥	🟥																			
Dataset		🟥	🟥	🟥	🟥	🟥	🟥																
Backend		🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥								
Image					🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧								
Training					🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨								
Testing												🟩	🟩	🟩	🟩	🟩	🟩						
UI																	🟩	🟩	🟩	🟩	🟩	🟩	
UI Design																	🟩	🟩	🟩				
UI Coding																				🟦	🟦	🟦	
Testing in UI																						🟦	🟦

3.2.2. Data Collection

After deciding the model and the requirement, the dataset will be made for training the CNN model. Thus in this step, some images will be collected by crawling on the internet to get images of objects to be recognized, which includes: apples, avocados, bananas, carrots, garlic, grapes, kiwis, mangos, oranges, pineapples, tomatoes. Firstly, a dataset which consist of any number of images from crawling will be made. After the accuracy result for each class has been gotten, and then the dataset will be normalized by equalizing the number of images in each class. After that the model will be trained using the normalized dataset, and by the result of each class, the dataset will be combined to make it better. These are the sample of the images after the collection.

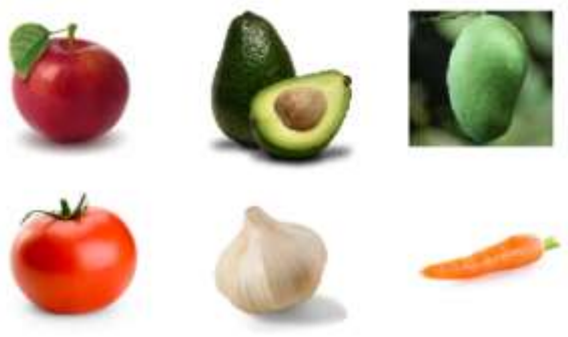


Figure 3.3 Training Set

Figure 3.4 Testing Set

To test the stability of the convolutional model, the model will be trained using the data with different number of images. And the model will be tested with 2 kind of validation test. The validation test itself consisted of 66 images and 88 images. The details of each dataset are written in Table 3.2.

Table 3.2 Dataset Differences

Class	Dataset 1		Dataset 2		Dataset 3	
	Train	Test	Train	Test	Train	Test
Apple	58	5	60	8	58	8
Avocado	50	8	60	8	50	8
Banana	36	6	60	8	36	9
Carrot	27	6	60	8	27	8
Garlic	36	8	60	8	36	8
Grape	41	8	60	8	60	8
Kiwi	60	5	60	8	60	8
Mango	45	6	60	8	45	10
Orange	35	8	60	8	60	9
Pineapple	34	8	60	8	60	8
Tomato	51	7	60	8	51	8
Total	**473**	**75**	**660**	**88**	**543**	**92**

3.2.3. Model Building

The Image Recognition will be implemented using Python 3 along with Tensorflow and Keras. Convolutional Neural Network (CNN) model will be used. So, this thesis will be talking specifically about creating the

CNN model. To produce a model that is most optimal and robust, the research will be done by following this flowchart:

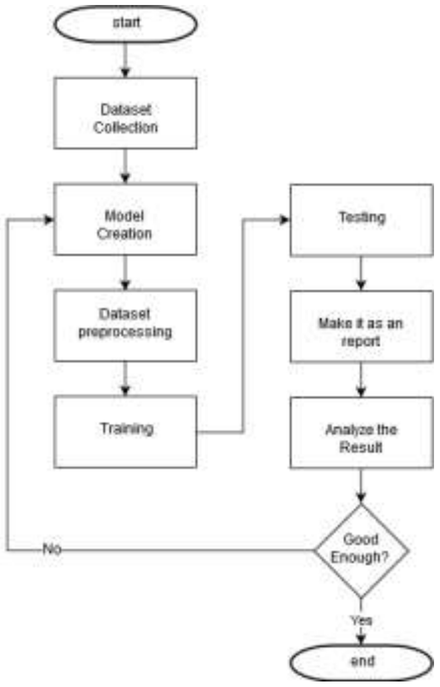


Figure 3.5 Research Structure

Convolutional Neural Network (CNN) model creation is the main purpose of this methodology. First, the standard CNN will be used. Afterwards, improving the accuracy of the standard CNN by adding and removing some layer, or changing the number of kernel used, or changing the kernel size or implementing the ideas from the state-of-art architectures as shown in Figure 3.6 to the CNN made, while constantly train and evaluate the CNN made. This thesis will implement the proposed model into the CNN bit by bit.

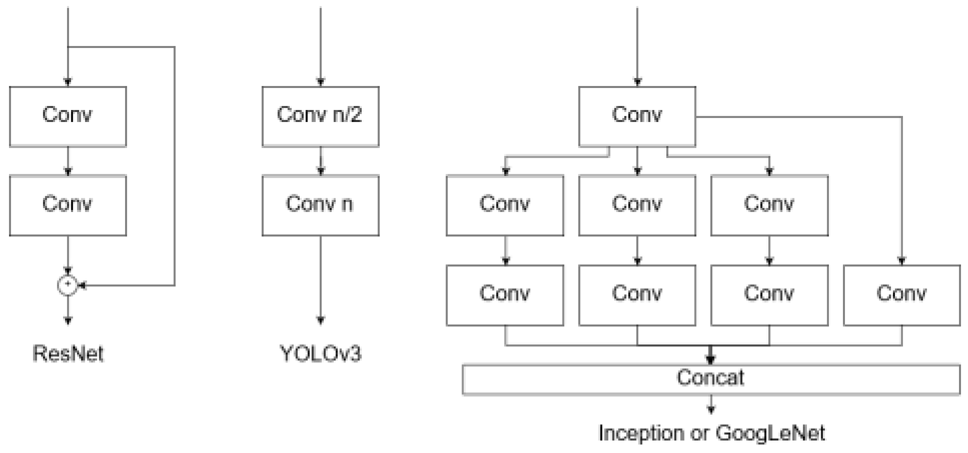


Figure 3.6 State-of-the-Art Architectures

And as the state-of-art architectures proves, mostly, CNN will have a better accuracy if they are made deep as He, Zhang, Ren, and Sun (2015) said. But Szegedy et al. (2015) have a different idea, where they made the Inception wider and deeper than other CNN architectures. And after studying the state-of-art and made bunch of models, the best architecture that made is found. The proposed model consisted of three main line (strand) which later on combines into one. The proposed model is wide and deep enough to predict the object in the image as in Figure 3.7. And on the Figure 3.7 there are some missing part where it could be filled by the layers that already explained briefly in the literature review such as Residual layer, Batch Normalization layer, Dropout layer, Inception layer, Stacked Bottleneck layer whereas few of them can be seen in Figure 3.6.

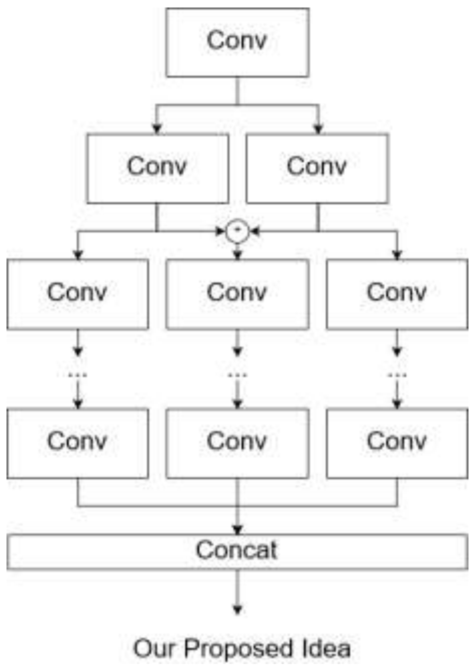
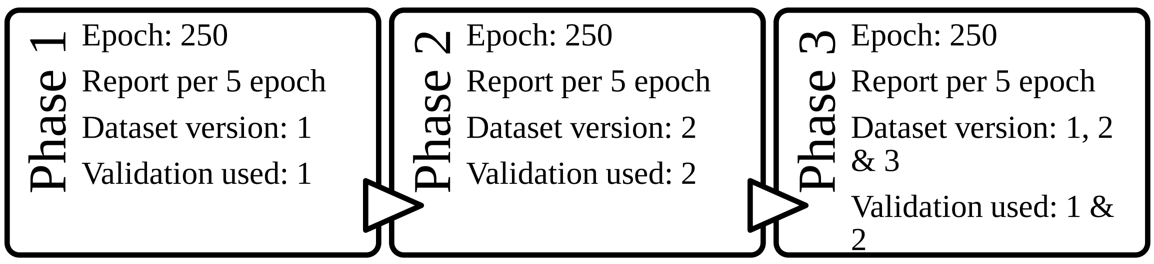


Figure 3.7 Proposed Architecture

After creating the models, the dataset will be preprocessed by resizing the images into 112, 112 then normalizing the images as in eq. 3.1 which means for every i and j from 0 until the respective weight and height, the member of matrix xi,j will be divided with 255. The result of eq. 3.1 will be normalized matrix where all of the member will be 0 < xi,j < 1.

$$\hat{x}_{i,j} = \frac{x_{i,j}}{255} \tag{3.1}$$

After the matrix or image is normalized, it will be used as the training input. Model training regime is shown in Figure 3.8. Other training menu to compare the top-7 model to the state-of-art architectures will be proposed.



Phase 1
Epoch: 250
Report per 5 epoch
Dataset version: 1
Validation used: 1

Phase 2
Epoch: 250
Report per 5 epoch
Dataset version: 2
Validation used: 2

Phase 3
Epoch: 250
Report per 5 epoch
Dataset version: 1, 2 & 3
Validation used: 1 & 2

Figure 3.8 Training Agenda

This process results in training one model by one model until a full list of 30 models have been gotten in the end which follows the Phase 1 training. Followed by training in Phase 2 resulted in the accuracy from validation set 2.

After the accuracies of validation set 1 and validation set 2 has been gotten, then continuing to calculate the average accuracy of each model as in attachment 1 until 3 in L-1 and L-2, which then gives us the top-7 models as can be seen in Table 3.3 along with each of the model differences.

Table 3.3 Top-7 Model

	Base Architecture	**Depth**	**Layer**	**Information**
CNN_2	Standard CNN	37	54	Residual: Conv, Conv
CNN_3	Standard CNN	40	57	Residual: Stacked Bottleneck, Conv
CNN_5	Standard CNN	40	61	Residual: Stacked Bottleneck, Bottleneck
CNN_11	Proposed Architecture	40	125	Factorization layer
CNN_14	Proposed Architecture	37	122	No Factorization layer
CNN_29	Proposed Architecture	44	151	3 Block each strand
CNN_30	Proposed Architecture	48	173	4 Block each strand

3.2.4. Application

The application will be made as shown in Figure 3.9 for single prediction and Figure 3.10 for batch prediction modules.

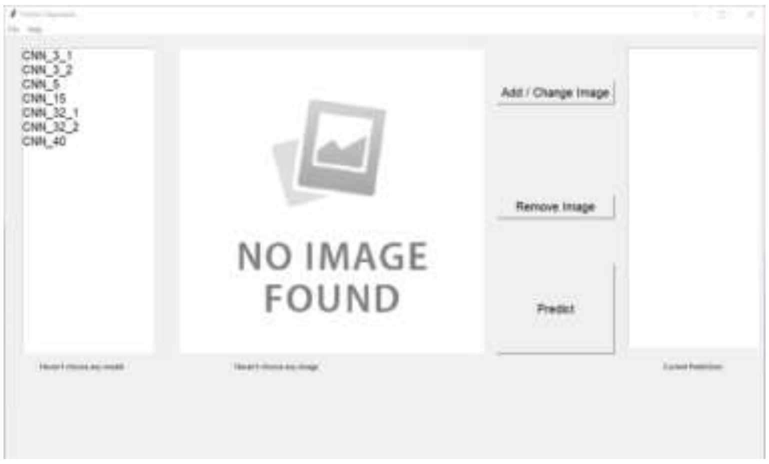


Figure 3.9 Single Prediction Module

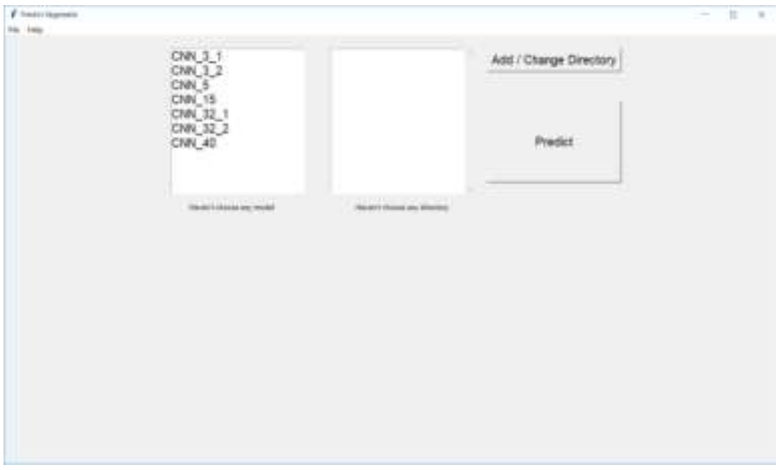


Figure 3.10 Batch Prediction Module

3.3. UML Diagram

For this thesis, the system is consisted of a Use Case Diagram as in Figure 3.11, a Class Diagram in Figure 3.12, and two Sequential Diagrams as shown in Figure 3.13 and 3.14 which already covers all of the software function and modules.

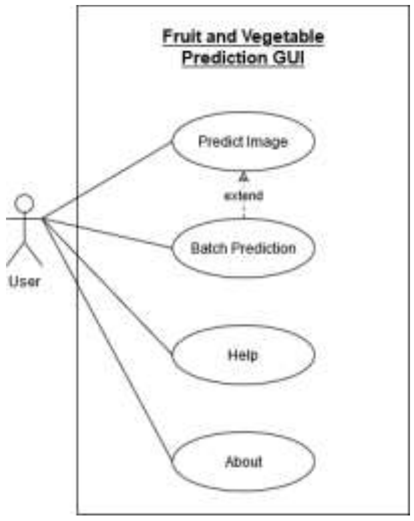


Figure 3.11 Use Case Diagram

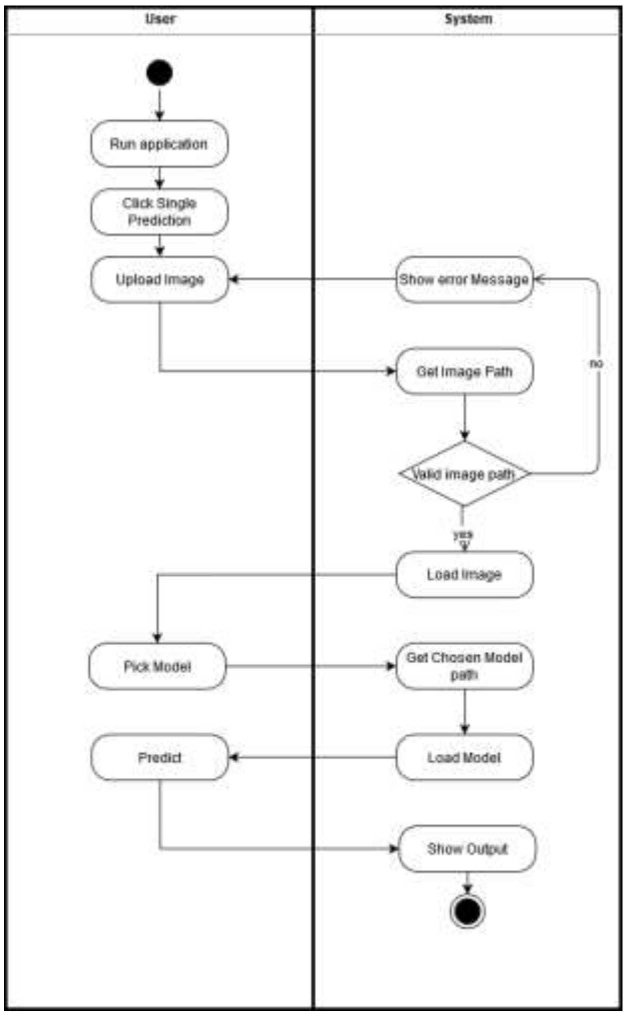


Figure 3.12 Activity Predict Image

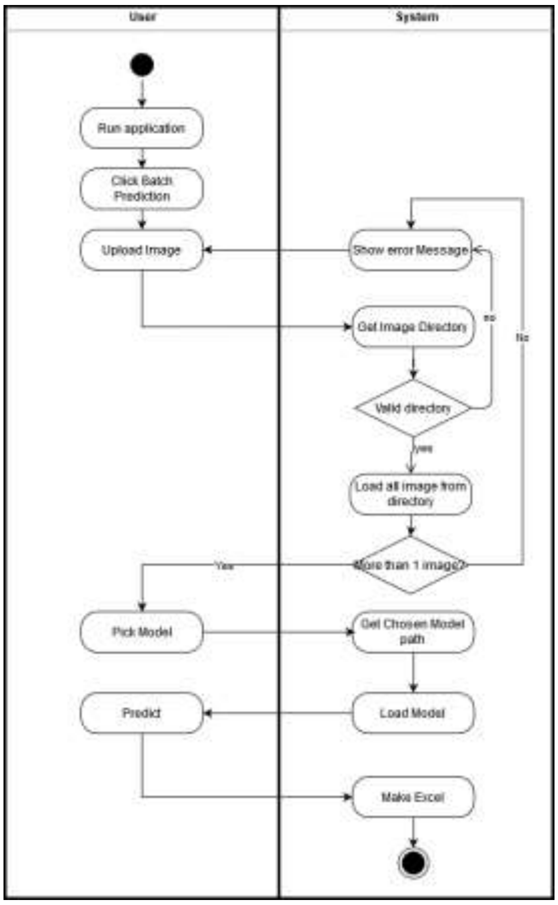


Figure 3.13 Activity Batch Prediction

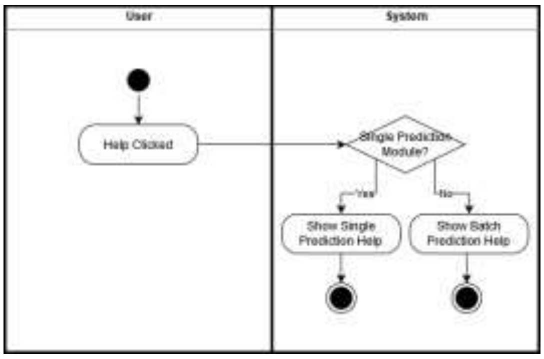

User
System
Help Clicked
Single Prediction Module?
Yes
No
Show Single Prediction Help
Show Batch Prediction Help

Figure 3.14 Activity Help

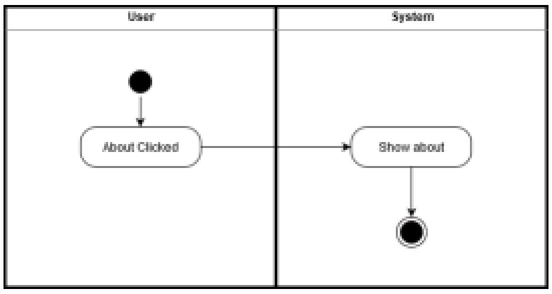


Figure 3.15 Activity About

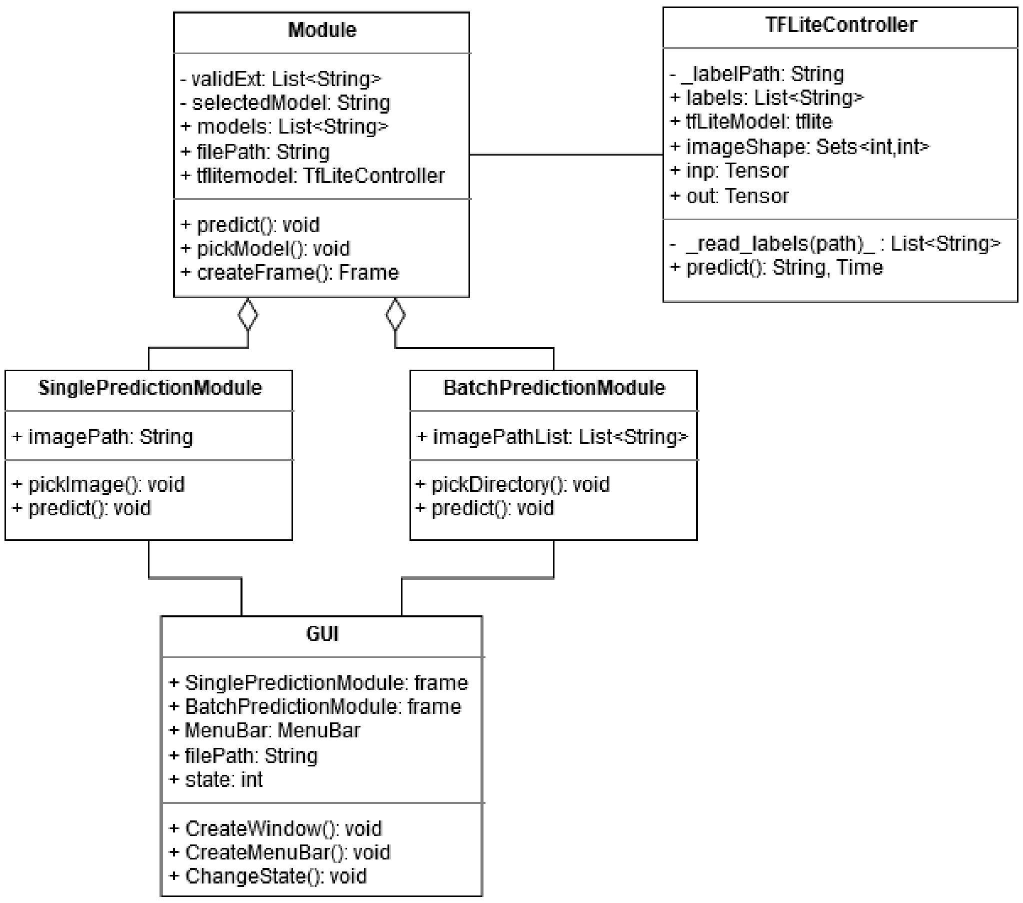


Figure 3.16 Class Diagram

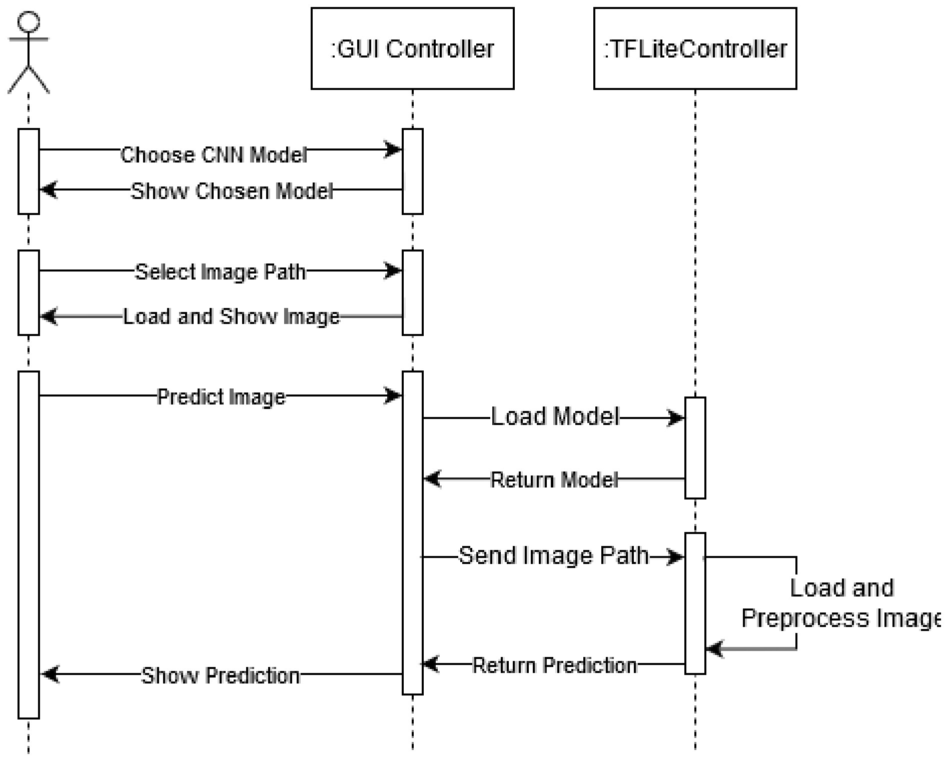


Figure 3.17 Sequence Diagram (Predict Image)

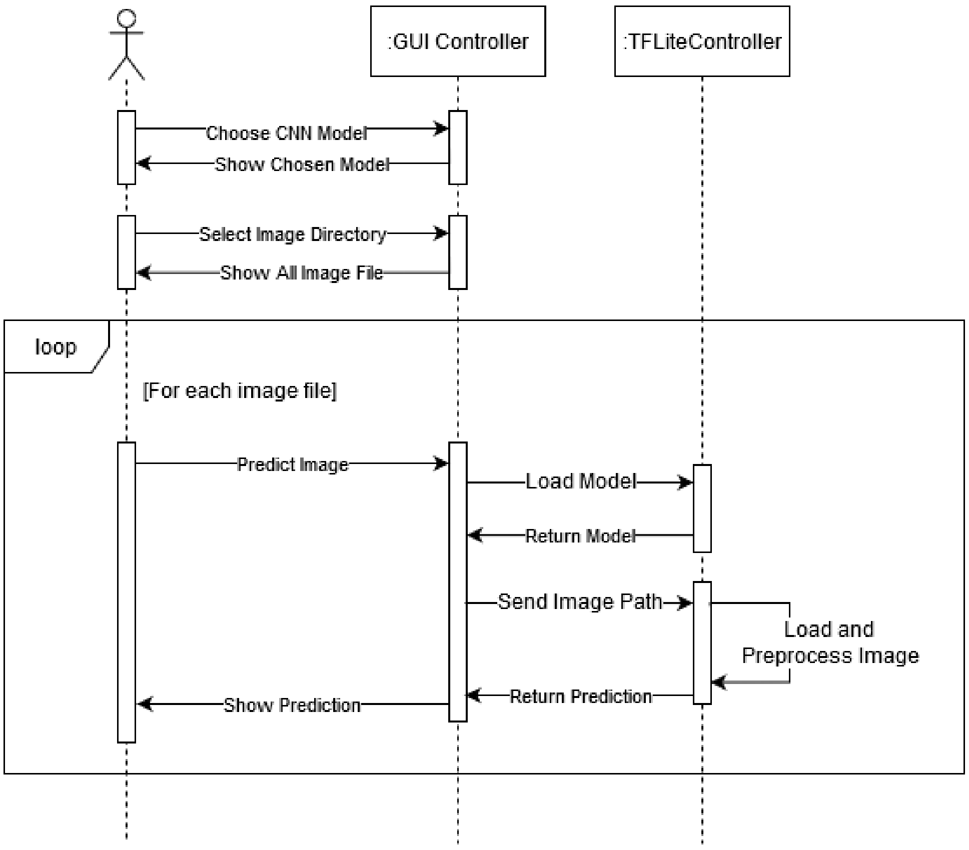


Figure 3.18 Sequence Diagram (Batch Prediction)

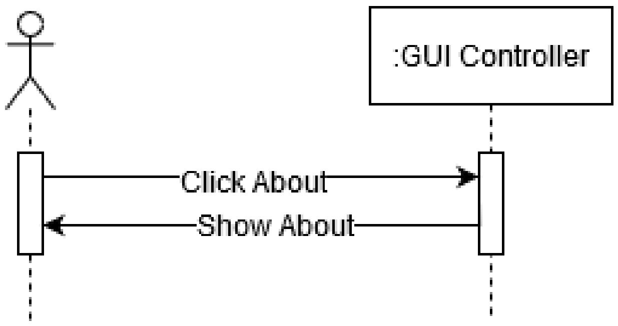


Figure 3.19 Sequence Diagram (About)

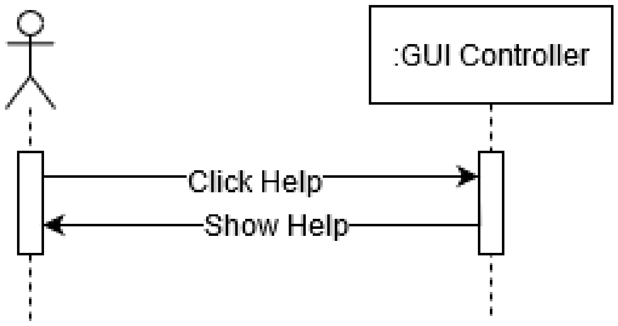


Figure 3.20 Sequence Diagram (Help)

CHAPTER 4
ANALYSIS AND RESULT

4.1. Experiment Result

The result from training the top-7 models that mentioned in the methodology using all of the dataset twice can be seen in Table 4.1 where the proposed architecture is implemented in model 11, 14, 29 and 30. Model 3 and 5 are quite good, where it was the combination of ResNet and YOLO, two of the state-of-the-art architectures. The result in Table 4.1 also proven that the proposed architecture works as well as the combination of state-of-art architectures.

Table 4.1 Accuracy Iteration Table

Iteration 1				
Model	Dataset			Average
	1	2	3	
2	65	79	65	69,67
3	64	82	84	76,67
5	78	84	75	79,00
11	79	76	71	75,33
14	64	76	76	72,00
29	75	83	75	77,67
30	77	80	78	78,33

Iteration 2				
Model	Dataset			Average
	1	2	3	
2	64	80	72	72,00
3	79	80	84	81,00
5	73	83	77	77,67
11	66	67	63	65,33
14	69	82	79	76,67
29	71	81	79	77,00
30	76	79	73	76,00

To compare the proposed model with the state-of-art architectures, training the top 7 models along with DenseNet121, ResNet50, and ResNet110 with the dataset will be needed. To compare the models, the evaluation of the current model every 5 epoch which shows the validation accuracy like previously done before is also needed, and training the models with data augmentation or image processing to improve the accuracy. The data augmentation setting is as follows:

1. Image are rescaled or normalized by dividing it by 255,
2. Image may be rotated randomly within -20 to 20 degrees,
3. Image may be flipped horizontally,
4. Image may be shifted from -10% to 10% from the original size horizontally or vertically.

4.2. Application System

The application system will be build using python and tkinter library. As for the application itself, it could be used to test the model along with certain images which the user uploads.

4.3.1. Application Functionality

This application has some functionality which includes the single prediction module and batch prediction module. The workflow of the application below will be explained.

I. Single Prediction Module

This function module is also set to be the home screen after opening the executable file. In this module there are some features such as Add/Change Image, Remove Image, Predict the image as shown in Figure 4.4. The application will automatically update the images in the image box every time a new image is uploaded. The user also can choose models that they wanted to use by clicking on the corresponding model list. After uploading the image and choosing the model they want to use, they could make the model do a prediction on current image by clicking on the Predict Button. The Prediction then will be shown in the rightmost part of the module. The user will see the result on the top as the history, and also on the bottom as the current prediction result and time taken as in Figure 4.5.

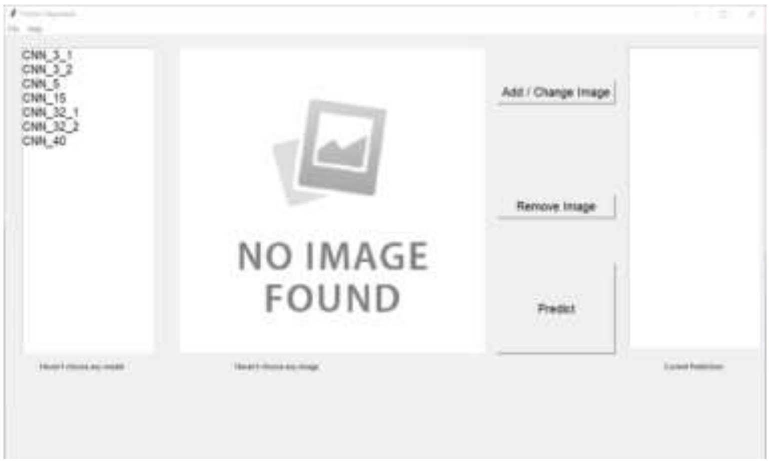


Figure 4.1 Single Prediction Module

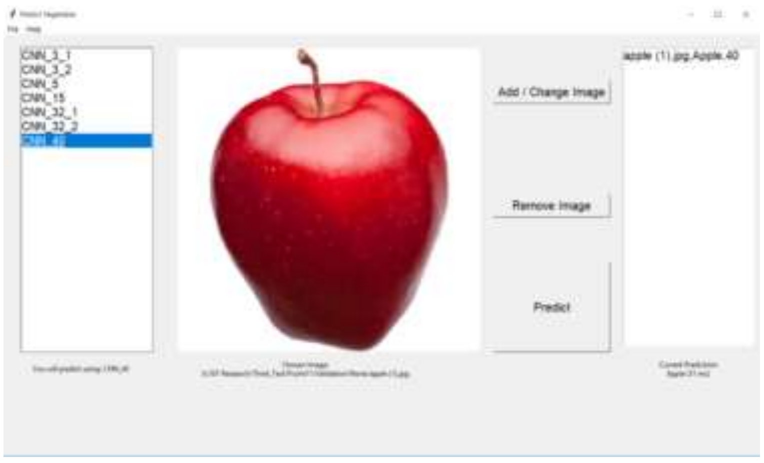


Figure 4.2 After Prediction

II. Batch Prediction Module

This function could be accessed from the File > Batch Prediction. In this module, the user would need to choose a directory containing images file as in Figure 4.6. The application then will take all images that it could process and show it on as in Figure 4.7 even if that folder has other files like in Figure 4.8. The predict button will predict all the images taken, and save it on a Comma-Separated Values (CSV) file and automatically opens up Microsoft Excel for the user.

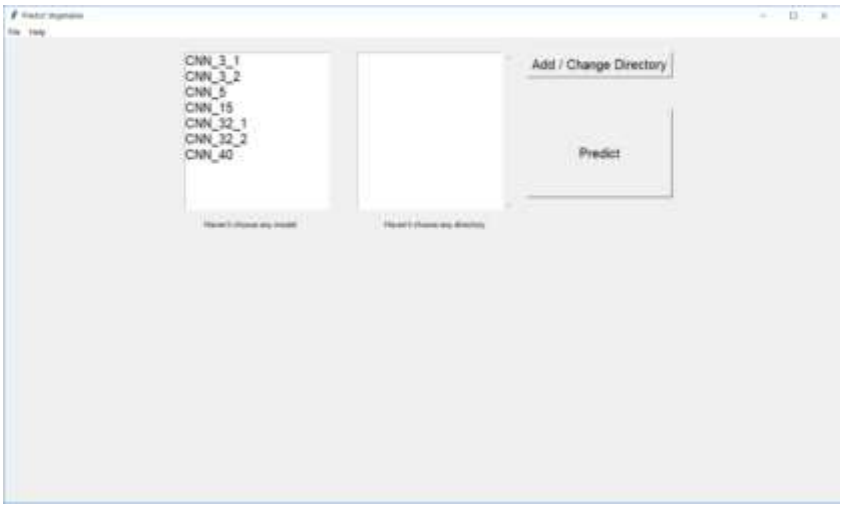


Figure 4.3 Batch Prediction Module

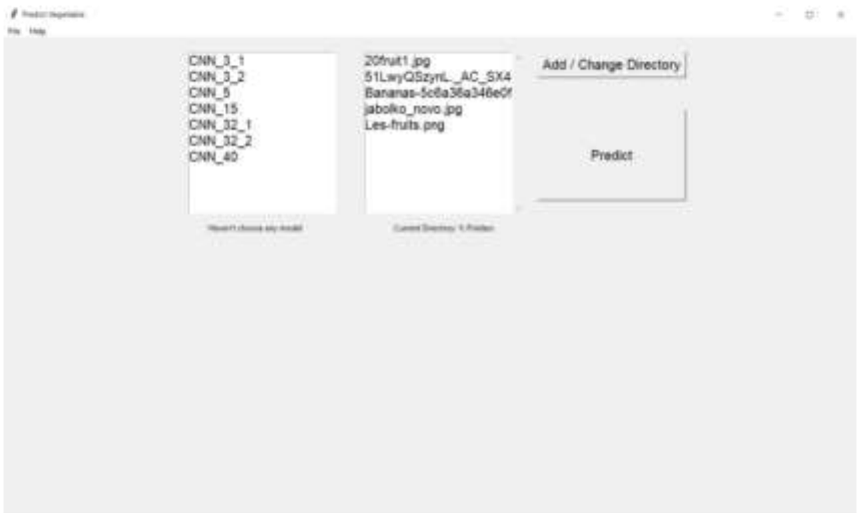


Figure 4.4 After Adding Directory

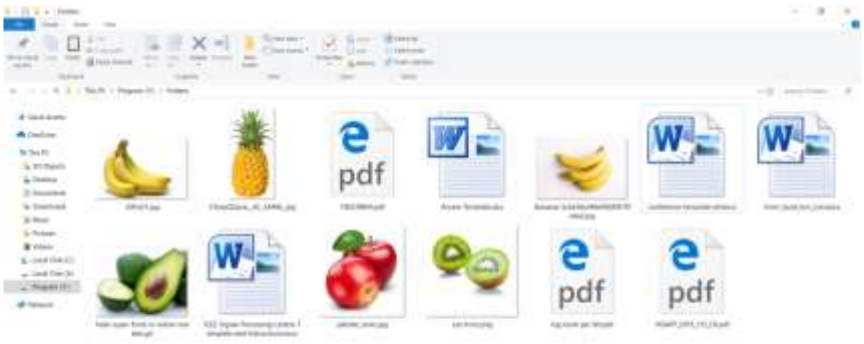


Figure 4.5 The Actual Folder Contents

4.3.2. Help Menu

As seen in Figure 4.9, there are help menu which consisted of the About sub menu and Manual sub menu. The About sub menu shows what is the application about and the Manual sub menu shows the User Manual or Guide to use the application. The User Manual will be divided into 2, depending on what module is active. If the user opens the manual while using the Single Prediction Module, it will open the manual like Figure 4.10. But if the user is using the Batch Prediction Module, it will open the manual as in Figure 4.11.

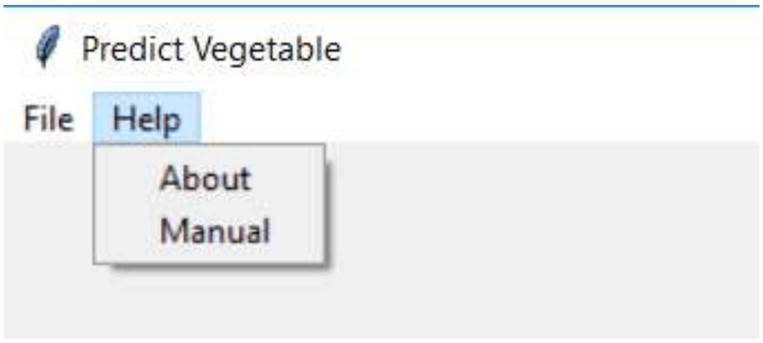
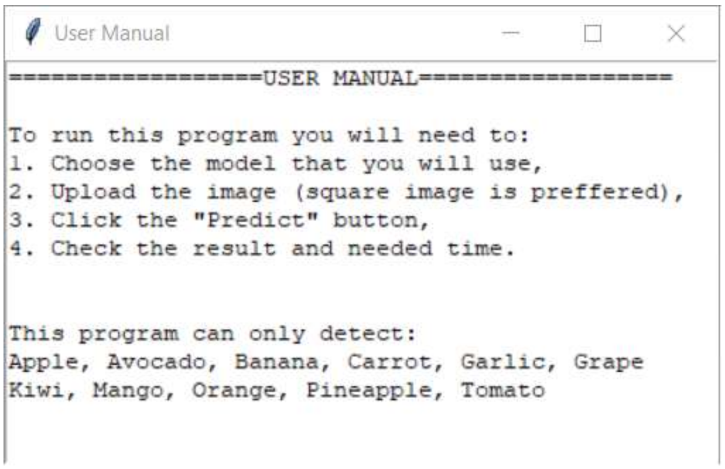


Figure 4.6 Help Sub Menus

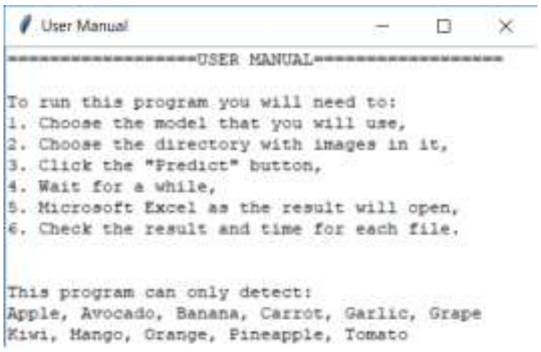


```
==================USER MANUAL==================

To run this program you will need to:
1. Choose the model that you will use,
2. Upload the image (square image is preffered),
3. Click the "Predict" button,
4. Check the result and needed time.


This program can only detect:
Apple, Avocado, Banana, Carrot, Garlic, Grape
Kiwi, Mango, Orange, Pineapple, Tomato
```

Figure 4.7 Manual for Single Prediction



```
==================USER MANUAL==================

To run this program you will need to:
1. Choose the model that you will use,
2. Choose the directory with images in it,
3. Click the "Predict" button,
4. Wait for a while,
5. Microsoft Excel as the result will open,
6. Check the result and time for each file.


This program can only detect:
Apple, Avocado, Banana, Carrot, Garlic, Grape
Kiwi, Mango, Orange, Pineapple, Tomato
```

Figure 4.8 Manual for Batch Prediction

4.3. Analysis and Evaluate CNN using the Application System

As seen, the CNN already implemented into the Application System, which means, the Application could be used to check the result of certain images. After trying to batch predict for the validation set, the data will be shown as follows.

Table 4.2 Result of Batch Prediction on Validation Data

	Top Accuracy	Average Time
CNN_2	82,01%	111
CNN_3	86,36%	148
CNN_5	84,66%	100
CNN_11	82,39%	37
CNN_14	79,36%	38
CNN_29	87,12%	50
CNN_30	90,34%	38
DenseNet121	89,77%	49
ResNet50	90,34%	62
ResNet110	90,34%	81

Thus the CNN_30 will be taken as the best model which supported by the high accuracy and fast prediction time.

4.4. Discussion and Result

Beside the time result from the application, there are some data that comes from each model. As seen in Figure 4.9, the proposed model can do as good as the other architecture such as ResNet50 and ResNet110. The CNN_30 model also runs faster than DenseNet121, ResNet50, and ResNet110. As seen in Table 4.7, the model that came from the proposed architecture (CNN_11, CNN_14, CNN_29, CNN_30) are not as complex as other model except the ResNet50 and ResNet110 but still can compete with the state-of-art architecture.

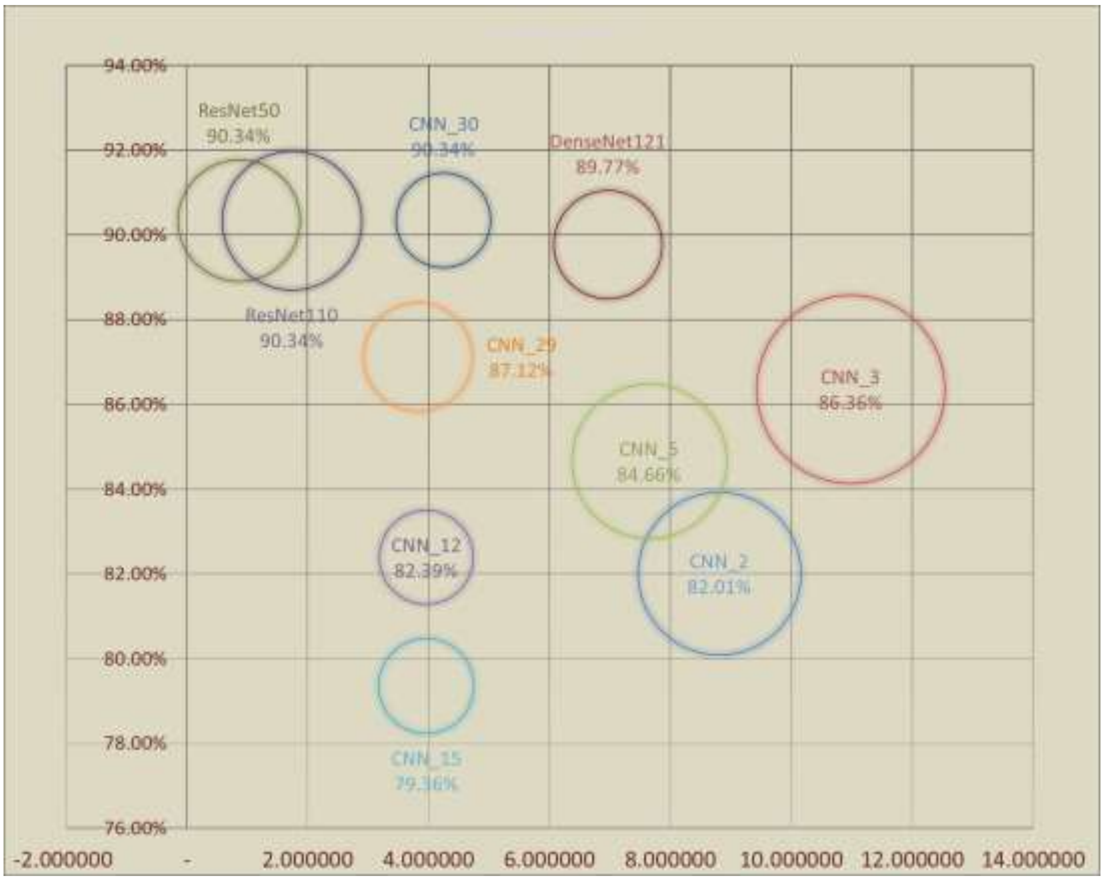


Figure 4.9 Accuracy-Complexity-Time

Table 4.3 Size of Each Model

Model	Complexity		Model Size (KB)
	Model	Computational	
CNN_2	8.811.435	17.610.000	34.429
CNN_3	10.980.427	21.940.000	42.895
CNN_5	7.649.211	15.270.000	29.884
CNN_11	3.962.707	7.910.000	15.510
CNN_14	3.965.163	7.910.000	15.518
CNN_29	3.825.187	7.630.000	14.982
CNN_30	4.248.431	8.470.000	16.626
DenseNet121	6.965.131	13.850.000	27.286
ResNet50	863.403	7.668.000	3.388
ResNet110	1.740.363	15.510.000	6.827

The result from Figure 4.9 and Table 4.3 shows that the CNN_30 is the best model from the proposed architecture, which also can be seen in Figure 4.10 and Figure 4.11.

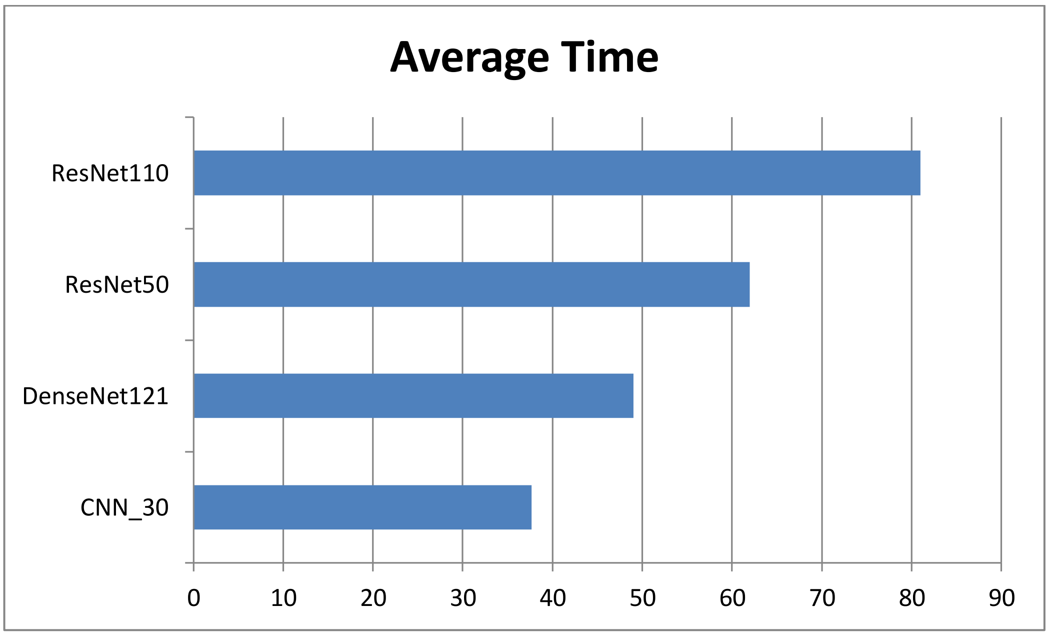


Figure 4.10 Time Comparison of Proposed Model and State-of-Art Architectures

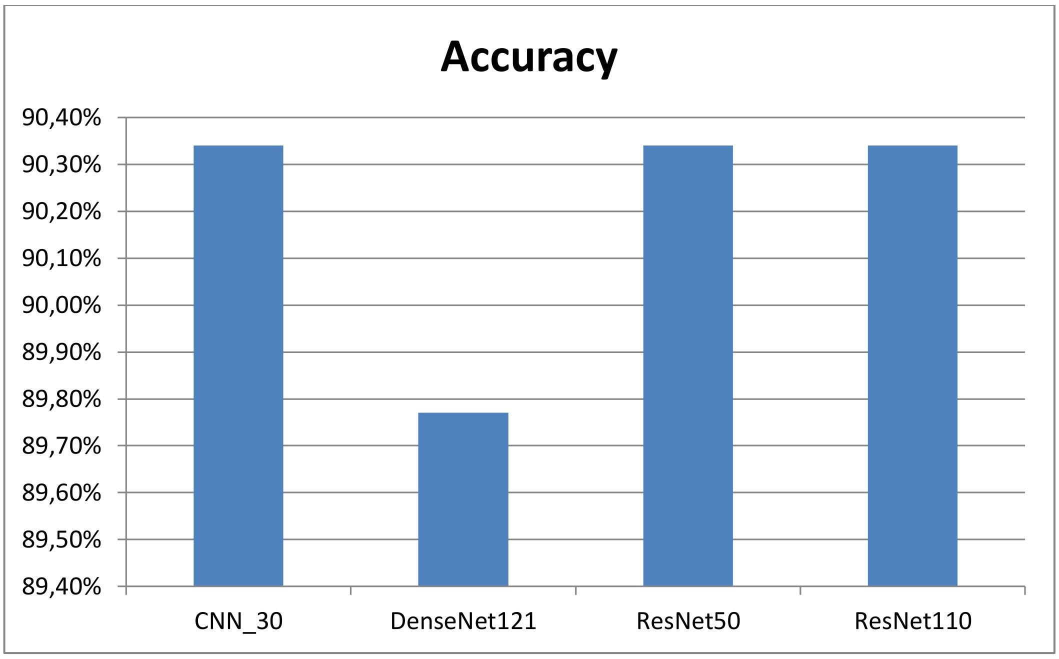


Figure 4.11 Accuracy Comparison of Proposed Model and State-of-Art Architectures

4.5. User Evaluation

Evaluation is done by interviewing some people after test the software. The number of respondent is 6 people. Based on the result of the interview, 6 out of 6 respondent says that the application is easy to understand and easy to use.

When testing the software, the respondent used 7 different images in average where the number of images used will be shown in Figure 4.12.

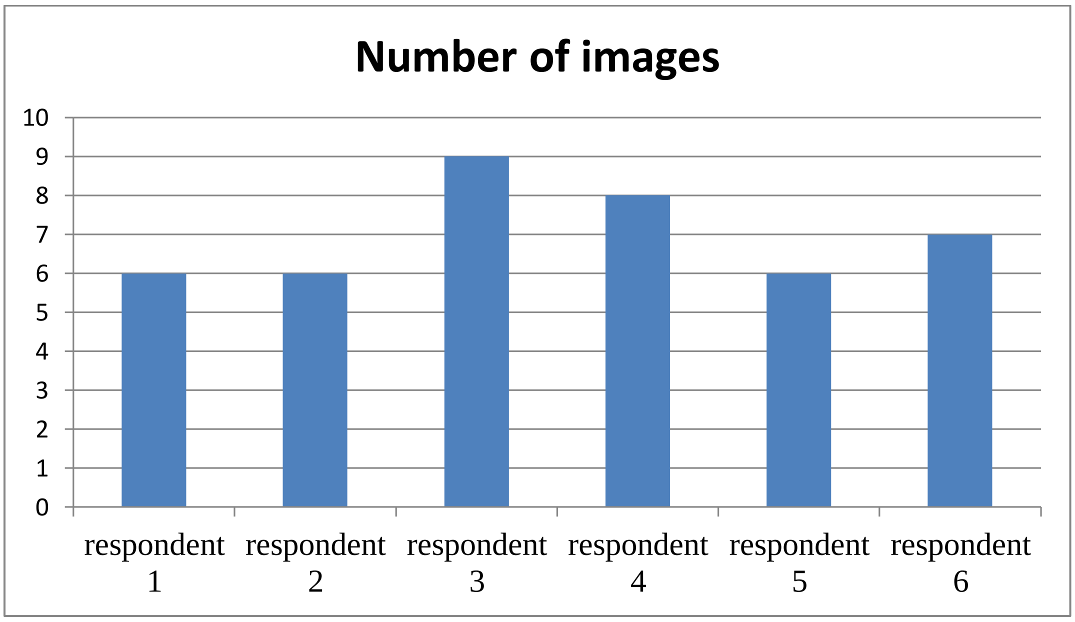


Figure 4.12 Number of Images per Respondent

Using those images, the respondent will perform test in each model selected. There are 4 models in total and the model used by respondent will be shown in Table 4.4.

Table 4.4 Model Used per Respondent

Respondent	**Model**			
	ResNet50	ResNet110	DenseNet	CNN_30
Respondent 1	X		X	X
Respondent 2		X		X
Respondent 3	X	X	X	X
Respondent 4	X	X		X
Respondent 5		X	X	X
Respondent 6	X	X	X	X

According to the tester, the proposed model successfully predicts more than 80% of the images tested. Average accuracy from testing is 84% where calculated based on correct prediction divided by number of images used. The result for each respondent can be seen in Figure 4.13, while the total result can be seen in Table 4.5.

Table 4.5 Accuracy Details per Respondent

Respondent	Correct Prediction	Total Image	Accuracy
Respondent 1	5	6	83%
Respondent 2	6	6	100%
Respondent 3	7	9	78%
Respondent 4	7	8	88%
Respondent 5	5	6	83%
Respondent 6	5	7	71%
Average	**6**	**7**	**84%**

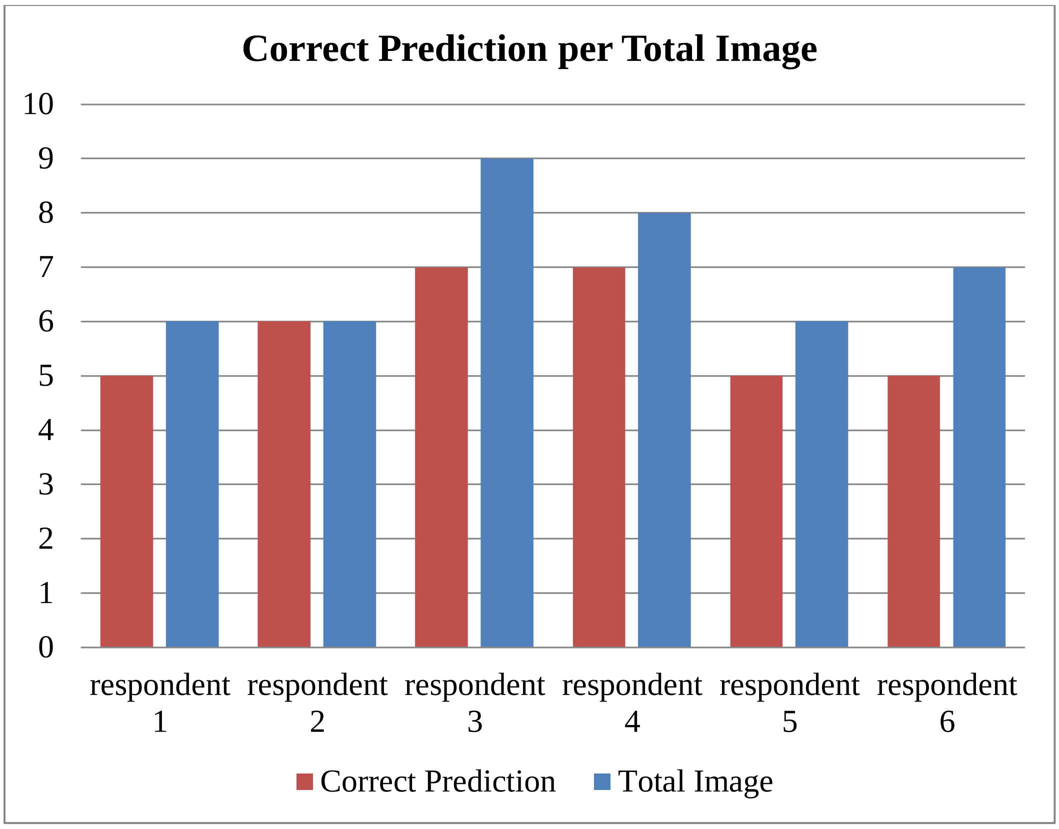


Figure 4.13 Correct Prediction per Total Image per Respondent

The respondent also gives us a positive feedback. And there are some inputs to the system where the user also asks us to improve the proposed model to detect multiple objects in an image, and implement it in real time. They also hope us to add more class so it can classify more categories of fruits and vegetables.

CHAPTER 5
CONCLUSION AND SUGGESTION

5.1. Conclusion

From the result of the experiment, there are some conclusion which answers the problem statement.

1. CNN_30 have been created and could classify fruit and vegetables even with only 60 images per class and 660 training images in total.
2. CNN_30 can even reach 90.34% accuracy, which is better than DenseNet, and on par with ResNet50 and ResNet110.
3. CNN_30 can predict each image within 38 ms averagely, which is faster than ResNet50 with 62 ms, ResNet110 with 81 ms, and DenseNet with 49 ms.
4. Based from the response from the users, the application is easy enough to be used and easy to understand. And from the tester, the accuracy is 84% from averagely 7 images. The testers also hoped to implement the application in real time.

Thus this thesis concluded that the proposed architecture works well on classifying the fruit and vegetables, furthermore the proposed architecture and CNN_30 could fulfill the hypothesis (H0) which is: getting accuracy greater than 80% with less than 42 ms per image as the H0.

5.2. Suggestion

After writing this thesis, there are some suggestion about some improvement on the fruits and vegetables detection as following:

1. Add some ROI Pooling layer to detect more than one object on one image,
2. Try to implement the fruits and vegetables detection to real-time image,
3. Refine the object by training to classify more dataset,
4. Implement the model to the Smart Fridge system.

REFERENCES

Bianco, S., Cadene, R., Celona, L., Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE Access,* 6, 64270-64277. doi: 10.1109/ACCESS.2018.2877890

Buzzelli, M., Belotti, F., & Schett, R. (2018). Recognition of Edible Vegetables and Fruits for Smart Home Appliances. *2018 IEEE 8th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).* doi: 10.1109/ICCE-Berlin.2018.8576236

Clarx, A. (n.d.). Pillow (PIL fork). Retrieved March 7, 2020, from https://pillow.readthedocs.io/en/stable/

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., & Chen, T. (2017). Recent advences in convolutional neural networks. *Pattern Recognition*, 77, 354-377. doi: 10.1016/j.patcog.2017.10.013

Hachani, A. Barouni, I., Said, Z. B., Amamou, L. (2016). RFID Based Smart Fridge. *2016 IFIP International Conference on new Technologies, Mobility and Security (NTMS)*. doi:10.1109/ntms.2016.7792472

Hagan, M. T., Demuth H. B.,& Beale M. H. (2014). *Neural network design* (2nd ed.). Retrived from https://hagan.okstate.edu/NNDesign.pdf

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *2016 IEEE Converence on Computer Vision and Patter Recognition (CVPR),* 1-6. doi: 10.1109/CVPR.2016.90

Hou, S., Feng, Y., Wang, Z. (2017). VegFru: A Domain-Specific Dataset for Fine-grained Visual Categorization. *2017 IEEE International Conference on Computer Vision (ICCV),* (541-549). doi:10.1109/ICCV.2017.66

Hou, L., Wu Q., Sun, Q., Yang, H., & Li, P. (2016). Fruit recognition based on convolutional neural network. *International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD),* 12, 18-22. doi: 10.1109/FSKD.2016.7603144

Huang, G., Liu, Z., Maaten, L., V., D., Weinberger, K., Q.(2017) Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2261 - 2269. doi: 10.1109/CVPR.2017.243

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv: 1502.03167v3

Jackson, C. A. (2018). Learn Programming in Python with Cody Jackson (1st ed.). Birmingham, United Kingdom: Packt Publishing Ltd.

Keras. (n.d.). Keras: The python deep leraning library. Retrieved March 7, 2020, from https://keras.io/

Li, B., & He, Y. (2018). An improved ResNet based on the adjustable shortcut connections. *IEEE Access*, 6, 18967 - 18974. doi: 10.1109/ACCESS.2018.2814605

Lundh F. (2005) An introdction to Tkinter. Retrieved march 6, 2020, from http://effbot.org/tkinterbook/

Luo, S., Jin, J. S., & Li, J. (2009). A Smart Fridge with an Ability to Enhance Health and Enable Better Nutrition. *International Journal of Multimedia and Ubiquitous Engineering, 4*(2), 69-80.

Luo, S., Xia, H., Gao, Y., Jin, J. S., Athauda, R. (2008). Smart Fridge with multimedia Capability for better Nutrition and Health. 2008 International Symposium on Ubiquitous Multimedia Computing. 39 - 44. doi:10.1109/umc.2008.17

Maas, A., Hannun, A., & Ng, A. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA. Retrieved from https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf

Miniaoui, S., Atalla, S., & Hashim, K. F. (2019). Introducing Innovative Item Management Process Towards Providing Smart Fridges. *2019 2nd International Conference on Communication Engineering and Technology* (pp. 62-67). Dubai: College of Engineering and IT University of Dubai.

Mohammed, M., Khan, M. B., & Bashier, E. B. (2017). *Machine Learning: Algorithms and Applications.* Florida: Taylor & Francis Group, LLC.

Moore, A. D. (2018). Python GUI Programming with Tkinter (1st ed.). Birmingham, United Kingdom: Packt Publishing Ltd.

Nixon, M. S., & Aguado, A. S.(2012). *Feature Extraction & Image Processing for Computer Vision* (3rd ed.). London, England: Academic Press.

NumPy. (n.d.). About NumPy. Retrieved March 7, 2020, from https://numpy.org/

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation Functions: Comparison of trends in Practice and Research for Deep Learning. arXiv:1811.03378

Pandas. (n.d.). About Pandas. Retrieved March 7, 2020, from https://pandas.pydata.org/about/index.html

Python Software Foundation. (2019). TkInter. Retrieved March 6, 2020, from https://wiki.python.org/moin/TkInter

Python Software Foundation. (n.d.). TkInter Dialogs. Retrieved March 7, 2020, from https://docs.python.org/3.9/library/dialog.html

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: unified, real-time object detection. *2016 IEEE Converence on Computer Vision and Pattern Recognition (CVPR)*, 779 - 788. doi: 10.1109/CVPR.2016.91

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6517 - 6525. doi: 10.1109/CVPR.2017.690

Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incemental Improvement. arXiv:1804.02767

Russell, S. J., & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

Russell, S.,& Norvig, P. (2010). Artificial intelligence: A modern approach(3rd ed.). Upper Saddle River, NJ: Pearson Education, Inc.

Saravanan, R., & Sujatha, P. (2018). A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification. *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. doi:10.1109/iccons.2018.8663155

Satzinger, J. W., Jackson, R. B., & Burd, S. D. (2014). Introduction to system analysis and design (6th ed.). London, United Kingdom: Cengage Learning EMEA.

Sharma, T., Singh, V., Sudhakaran, S., & Verma, N. K. (2019). Fuzzy based pooling in convolutional neural network for image classification. *2019 IEEE International Conference on Fuzzy System (FUZZ-IEEE)*, 1-6. doi: 10.1109/FUZZ-IEEE.2019.8859010

Smola, A., & Vishwanathan, S. (2008). *Introduction to Machine Learning.* United Kingdom: Cambridge University Pres.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929–1958.

Sultana, F., Sufian, A., & Dutta, P. (2018). Advancement in image classification using convolutional neural network. *2018 ICRCICN*, 4, 122 - 129. doi: 10.1109/ICRCICN.2018.8718718

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). Going Deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1-9. doi: 10.1109/CVPR.2015.7298594

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818 - 2826. doi: 10.1109/CVPR.2016.308

Szeliski, R. (2010). Computer Vision Algorithms and Application. London, England: Springer

TensorFlow. (n.d.). Why Tensorflow. Retrieved March 7, 2020, from https://www.tensorflow.org/about

Weaver, P. (2012). Henry L. Gantt, 1861-1919: A retrospective view if his work. PM *World Journal.* Retrieved from https://www.mosaicprojects.com.au/PDF_Papers/P158_Henry_L_Gantt.pdf

ATTACHMENT

Attachment 1 Result of Training using Validation 1

Name	Correct / 66	%
CNN_1	40	60,61
CNN_2	58	87,88
CNN_3	54	81,82
CNN_4	49	74,24
CNN_5	53	80,30
CNN_6	52	78,79
CNN_7	44	66,67
CNN_8	45	68,18
CNN_9	42	63,64
CNN_10	45	68,18
CNN_11	54	81,82
CNN_12	45	68,18
CNN_13	48	72,73
CNN_14	57	86,36
CNN_15	48	72,73
CNN_16	41	62,12
CNN_17	47	71,21
CNN_18	48	72,73
CNN_19	44	66,67
CNN_20	48	72,73
CNN_21	44	66,67
CNN_22	50	75,76
CNN_23	50	75,76
CNN_24	46	69,70
CNN_25	49	74,24
CNN_26	48	72,73
CNN_27	49	74,24
CNN_28	49	74,24
CNN_29	53	80,30
CNN_30	53	80,30

Attachment 2 Result of Training using Validation 2

Name	Correct / 88	%
CNN_1	50	56,82
CNN_2	64	72,73
CNN_3	69	78,41
CNN_4	64	72,73
CNN_5	67	76,14
CNN_6	65	73,86
CNN_7	57	64,77
CNN_8	61	69,32
CNN_9	48	54,55
CNN_10	62	70,45
CNN_11	64	72,73
CNN_12	66	75,00
CNN_13	61	69,32
CNN_14	62	70,45
CNN_15	54	61,36
CNN_16	65	73,86
CNN_17	65	73,86
CNN_18	65	73,86
CNN_19	61	69,32
CNN_20	53	60,23
CNN_21	8	9,09
CNN_22	68	77,27
CNN_23	58	65,91
CNN_24	55	62,50
CNN_25	62	70,45
CNN_26	66	75,00
CNN_27	65	73,86
CNN_28	66	75,00
CNN_29	68	77,27
CNN_30	67	76,14

Attachment 3 Average Accuracy of Every Model

Name	%_1	%_2	$\overline{x}$	Name	%_1	%_2	$\overline{x}$
CNN_1	60,61	56,82	58,71	**CNN_16**	62,12	73,86	67,99
CNN_2	87,88	72,73	80,30	**CNN_17**	71,21	73,86	72,54
CNN_3	81,82	78,41	80,11	**CNN_18**	72,73	73,86	73,30
CNN_4	74,24	72,73	73,48	**CNN_19**	66,67	69,32	67,99
CNN_5	80,30	76,14	78,22	**CNN_20**	72,73	60,23	66,48
CNN_6	78,79	73,86	76,33	**CNN_21**	66,67	9,09	37,88
CNN_7	66,67	64,77	65,72	**CNN_22**	75,76	77,27	76,52
CNN_8	68,18	69,32	68,75	**CNN_23**	75,76	65,91	70,83
CNN_9	63,64	54,55	59,09	**CNN_24**	69,70	62,50	66,10
CNN_10	68,18	70,45	69,32	**CNN_25**	74,24	70,45	72,35
CNN_11	81,82	72,73	77,27	**CNN_26**	72,73	75,00	73,86
CNN_12	68,18	75,00	71,59	**CNN_27**	74,24	73,86	74,05
CNN_13	72,73	69,32	71,02	**CNN_28**	74,24	75,00	74,62
CNN_14	86,36	70,45	78,41	**CNN_29**	80,30	77,27	78,79
CNN_15	72,73	61,36	67,05	**CNN_30**	80,30	76,14	78,22