บทที่ 3

ขั้นตอนและวิธีดำเนินงาน

การดำเนินงานวิจัยเพื่อให้ง่ายต่อการเพิ่มประสิทธิภาพของระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรด ผู้วิจัยได้มีแนวคิดในการเพิ่มประสิทธิภาพของระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรดโดยสามารถทำงานแทนผู้เชียวชาญคัดแยกความสุกของสับปะรดออกเป็น 5 ระดับได้โดยเป็นการแบ่งเบาภาระหน้าที่ของกระบวนการผลิค โดยเริ่มต้นจากศึกษาการออกแบบระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรด ศึกษาการใช้งานระบบ tensorflow การทำงานอัตโนมัติของระบบ การสร้างโมเดลโดยใช้อัลกอริทึม convolution neural networks รวมไปถึงการศึกษาการเขียนภาษา python ที่ต้องใช้ในการเขียนโปรแกรม โดยผู้วิจัยได้แบ่งขั้นตอนการดำเนินงานวิจัยดังนี้

3.1 แผนผังการดำเนินงาน

3.2 การเตรียมข้อมูลรูปภาพความสุกสับปะรดก่อนนำไปฝึกสอนโมเดล

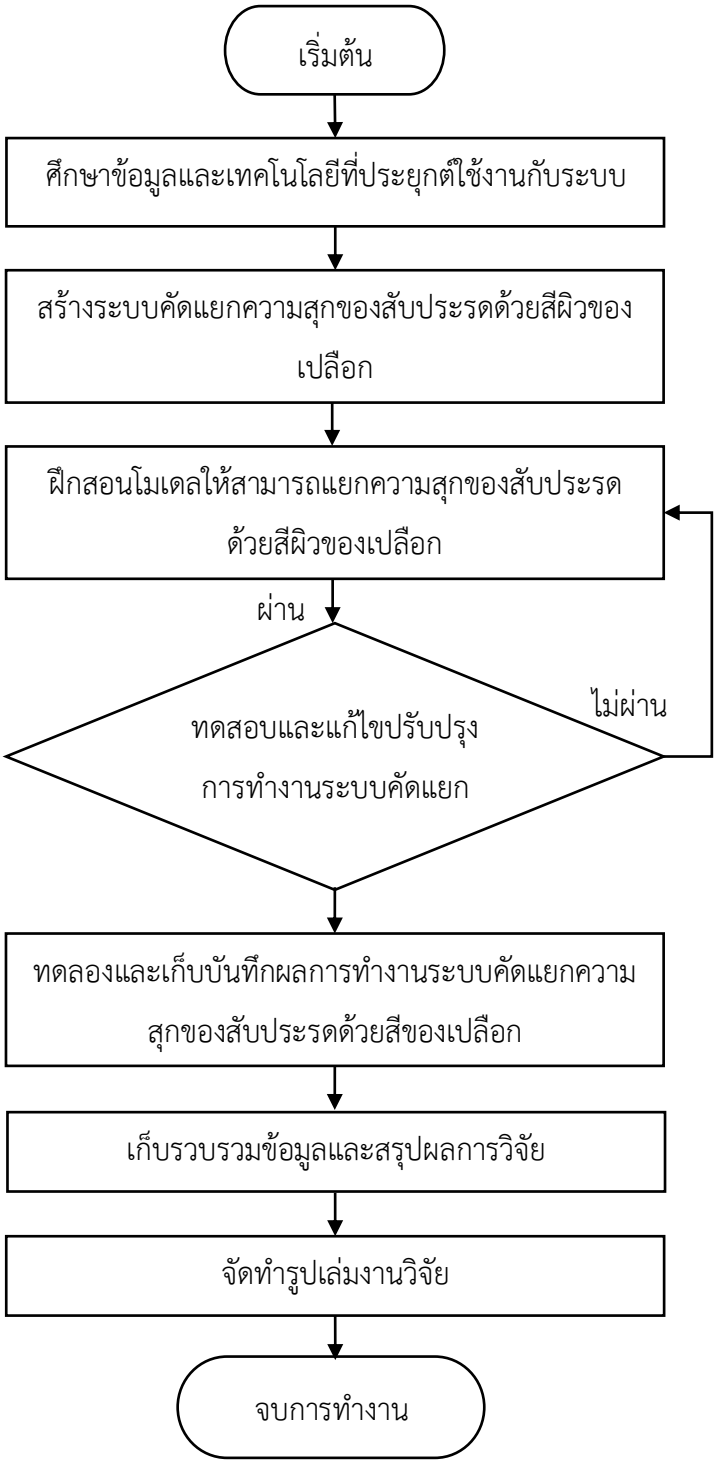
3.3 ขั้นตอนการเรียนรู้แบบ convolutional neural network

3.4 การนำโมเดลไปใช้งาน

3.5 การออกแบบและพัฒนาเครื่องระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรด

3.1 แผนผังการดำเนินงาน

ในการดำเนินการวิจัยผู้วิจัยได้ทำการศึกษาออกแบบระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรด เพื่อให้ได้ประสิทธิภาพสูงสุดในการดำเนินงานวิจัยนี้ ซึ่งผู้วิจัยได้แบ่งขั้นตอนต่างๆ ในการดำเนินงานเพื่อศึกษาหาวิธีการที่จะทำให้ระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรดสามารถแยกความสุกของสับปะรดออกเป็น 5 ระดับ โดยขั้นตอนต่างๆ ผู้วิจัยได้จัดทำเอาไว้แสดงในแผนผังการดำเนินงานวิจัยดังแสดงในภาพที่ 3.1



ภาพที่ 3.1 แผนภาพแสดงขั้นตอนการดำเนินงานวิจัยระบบคัดแยกความสุกของสับปะรดด้วยสีผิวของเปลือก

จากภาพที่ 3.1 แผนภาพแสดงขั้นตอนการดำเนินงานวิจัยระบบคัดแยกความสุกของสับปะรดด้วยสีผิวของเปลือก จากแผนผังการดำเนินงานวิจัยได้เริ่มศึกษาการออกแบบระบบส่วนประกอบโครงสร้างของระบบและวิธีการคัดแยกเพื่อให้สามารถคัดแยกความสุกสับปะรดออกเป็น 5 ระดับต่อไปจากนั้นเริ่มสร้างระบบคัดแยกความสุกของสับปะรดด้วยสีผิวของเปลือก โดยใช้ทฤษฎี Deep Learning อัลกอริทึม Convolutional Neural Networks ซึ่งมีทั้งวิธีการติดตั้งชุดซอฟต์แวร์และไลบรารี่ การใช้คำสั่งต่างๆ ที่เกี่ยวข้องทั้งหมดเมื่อเข้าใจระบบดีแล้วจึงได้วางแผนการพัฒนาโปรแกรมอย่างเป็นระบบ โดยอันดับแรกได้ทำการฝึกสอนโมเดลให้สามารถแยกความสุกของสับปะรดด้วยสีผิวของเปลือก มีอยู่ 2 ขั้นตอน ขั้นตอนแรกคือการสอนโมเดลเริ่มจากการเตรียมรูปภาพสับปะรดความสุกทั้ง 5 ระดับทำกระบวนการ image data generator และทำกระบวนการข้อมูลรูปภาพเป็นข้อมูลตัวเลขแบบ array โดยสร้างโมเดลการคัดแยกความสุกของสับปะรด 5 ระดับมารองรับจากรูปภาพที่เตรียมไว้ข้างต้นเมื่อกระบวนการสร้างโมเดลเสร็จสิ้นได้นำตัวโมเดลที่ผ่านการฝึกสอนมาแล้วนำไปเปลี่ยนแปลงให้ใช้กับบอร์ดประมวลผล raspberry pi จาก tensorflow เป็น tensorflow lite หลังการทำการเปลี่ยนเปลงแล้วความแม่นยำอาจคาดเคลื่อนในค่าที่ยอมรับได้ทดสอบและแก้ไขปรับปรุงการทำงานระบบคัดแยกโดยการทดสอบความแม่นยำของโมเดลทั้งด้านการฝึกสอนและนำไปใช้งานกับระบบจริง จากนั้นทดลองการดำเนินงานโดยการเก็บเกี่ยวผลสับปะรดภูแลจากไร่เกษตรกร ในตำบลบ้านดู่ โดยคัดเลือกเฉพาะผลสับปะรดที่สมบูรณ์และไม่มีตำหนินำมาทดลองกับงานของผู้วิจัย คือการคัดแยกผลโดยมีระดับตรงตามที่ผู้เชี่ยวชาญกำหนดไว้ 5 ระดับ นำมาถ่ายรูปภาพเพื่อเป็นข้อมูลสำหรับเทรนนิ่งโมเดล จากนั้นทำการฝึกสอนคอมพิวเตอร์ให้สามารถแยกความสุกของสับปะรดด้วยสีผิวของเปลือก ทดสอบการทำงานของระบบคัดแยก จนสามารถทำงานได้ และทดลองการทำงานระบบคัดแยกความสุกของสับปะรดด้วยสีของเปลือก เก็บรวบรวมข้อมูลและสรุปผล

3.2 การเตรียมข้อมูลรูปภาพความสุกสับปะรดก่อนนำไปฝึกสอนโมเดล

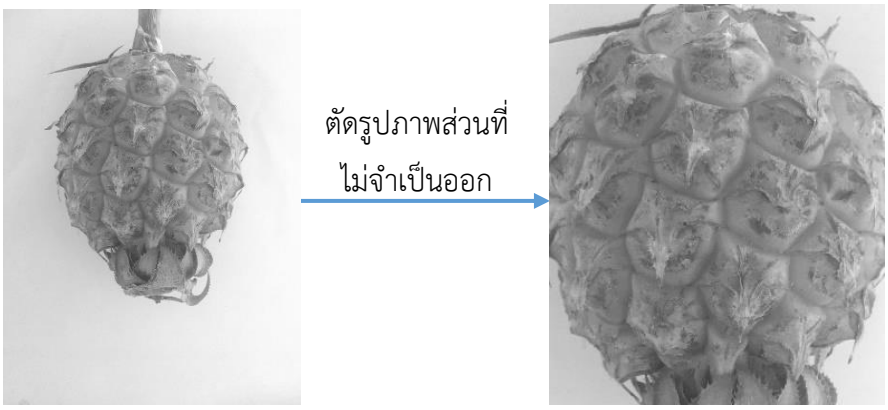
โดยกระบวนการเตรียมข้อมูลภาพเป็นข้อมูลที่ระบบต้องเรียนรู้จากรูปภาพความสุกสับปะรดทั้ง 5 ระดับ เริ่มจากถ่ายภาพสับปะรดจากโทรศัพท์ ซึ่งใช้ภาพทั้งหมด 500 ภาพ ข้อมูลไฟล์คือ png ประกอบไปด้วยระดับของสับปะรด 5 ระดับ เนื่องจากเป็นข้อมูลภาพถ่ายที่สามารถนำไปถ่ายได้ง่ายและมีความละเอียดที่ชัดเจน โดยภาพจากโทรศัพท์มีลักษณะดังภาพที่ 3.1

ภาพที่ 3.2 ลักษณะภาพถ่ายที่ได้จากกล้องโทรศัพท์

จากภาพที่ 3.2 เป็นภาพสับปะรดที่ได้จากการถ่ายภาพ โดยจัดเตรียมพื้นหลังเป็นสีขาวและทำการจัดแสงของภาพให้เห็นเนื้อผิวได้ชัดเจน

ภาพที่ 3.3 การแบ่งรูปภาพสับปะรดเป็น 5 ระดับจากผู้เชียวชาญ

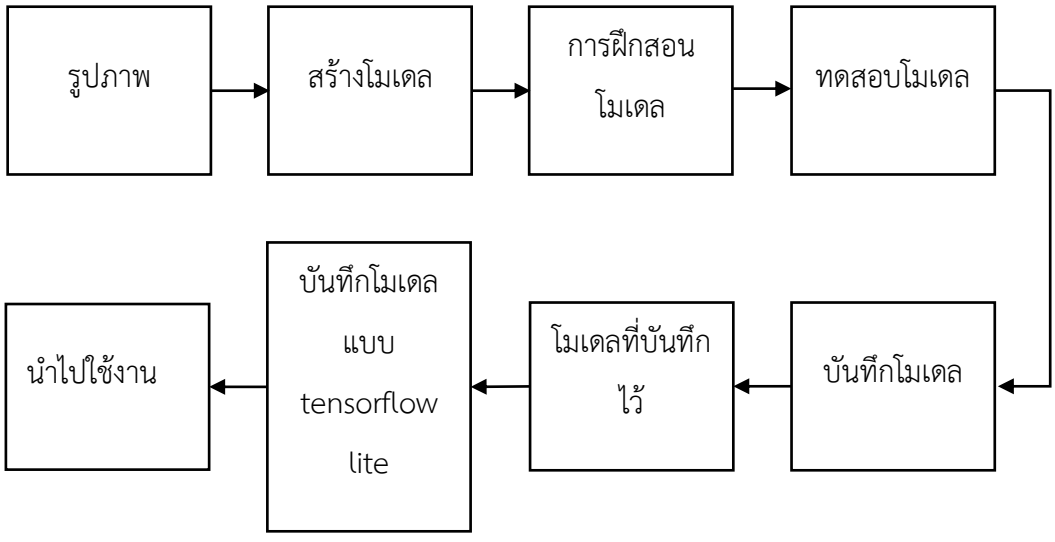
จากภาพ 3.3 การแบ่งรูปภาพสับปะรดเป็น 5 ระดับจากผู้เชียวชาญ ได้คัดแยกสับปะรดจากผู้เชี่ยวชาญแล้วนำมาเก็บแยกในโฟลเดอร์ทั้ง 5 ระดับเอาไว้ โดยระดับที่ 1 มีลักษณะเป็นผลที่ดิบอยู่ ตามีสีม่วงไม่เต่งตึงและลีบ ส่วนขอบตามีสีเขียวดำ ระดับนี้ไม่มีโอกาสสุกแล้ว สาเหตุที่ตัดมาเกิดจากความผิดพลาดของคนเกี่ยบเกี่ยว หรือต้องการล้างแปลงและปลูกสับปะรดรอบต่อไป ระดับ 2 มีลักษณะเป็นผลที่สุกแก่ มีสีเขียวอ่อน ตาเริ่มเหลือง 1-2 ตาและมีความเต่งตึง หากบ่มไว้ 1-3 วัน ก็จะสุกในระดับที่ 3 ระดับ 3 มีลักษณะเป็นผลที่สุกขึ้นตาหรือสุกครึ่งผลขึ้นไปเกือบทั้งผล แต่มีสีเขียวอ่อนอยู่ สุกประมาน 80% ระดับ 4 มีลักษณะเป็นผลที่สุกและเหลืองทั้งผล สุก 100% และระดับ 5 มีลักษณะเป็นผลที่สุกมาก มีสีเหลืองออกในโทนส้มเข้ม และตามีความเหี่ยวตามระยะเวลาที่เก็บไว้ ซึ่งความแตกต่างทั้ง 5 ระดับนี้นำไปเป็นจุดเด่นด้านการคัดแยกของระบบต่อไป



ภาพที่ 3.4 ตัดขอบรูปภาพในส่วนที่เกินขอบเขตออกไป

จากภาพที่ 3.4 ตัดขอบรูปภาพในส่วนที่เกินขอบเขตออกไป ในส่วนพื้นหลังที่เป็นสีขาวซึ่งใช้ไลบรารี่ Open CV และเขียนโค้ดโปรแกรมในการจัดการส่วนนี้วนลูปให้ตัดขอบจนติดขอบของผลสับปะรดอัตโนมัติจนครบทั้ง 500 ภาพ ภาพที่ได้จะเป็นการตัดส่วนที่ว่างออกไปจนเหลือแต่ผลสับปะรดที่ต้องการให้ระบบเรียนรู้ได้อย่างมีประสิทธิภาพ

3.3 ขั้นตอนเรียนรู้แบบ convolutional neural network



ภาพที่ 3.5 ขั้นตอนการวิจัยในกระบวนการเรียนรู้แบบ convolutional neural network

จากภาพที่ 3.5 ขั้นตอนการวิจัยในกระบวนการเรียนรู้แบบ convolutional neural network สร้างโมเดลขึ้นมาเพื่อนำรูปภาพที่เตรียมไว้ไปทำการฝึกสอนโมเดลให้สามารถคัดแยกความสุกของสับปะรดออกเป็น 5 ระดับเมื่อกระบวนการสอนเสร็จทำการทดสอบโมเดลให้ได้ประสิทธิภาพตามต้องการและทำการบันทึกโมเดลทางผู้วิจัยได้ต้องการนำไปใช้กับบอร์ด raspberry pi จึงจำเป็นต้องแปลงโมเดลให้อยู่ในรูปแบบจาก tensorflow เป็น tensorflow lite ก่อนนำไปใช่งานซึ่งค่าความแม่นยำยังคงอยู่ในระดับที่ยอมรับได้

```
import tensorflow as tf
import os

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import  Conv2D,MaxPooling2D,Dense,GlobalAveragePooling2D,Dropout,Flatten
from tensorflow.keras.optimizers import Adam

import matplotlib.pyplot as plt
%matplotlib inline


from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

ภาพที่ 3.6 อิมพอร์ตไลบรารี่ต่าง ๆ เข้ามาใช้ในงานวิจัย

จากภาพที่ 3.6 อิมพอร์ตไลบรารี่ต่างๆ เข้ามาใช้ในงานวิจัยหลังจากทำการดาวน์โหลดและติดตั้งเรียบร้อยในกระบวนการ convolutional neural network ประกอบด้วย tensorflow , keras , os , matplotlib

ภาพที่ 3.7 การสร้าง path ที่อยู่ของข้อมูลรูปภาพโดยใช้ไลบรารี่ os

จากภาพที่ 3.7 การสร้าง path ที่อยู่ของข้อมูลรูปภาพโดยใช้ไลบรารี่ os ทำการสร้าง path ที่อยู่ของข้อมูลรูปภาพสับปะรดซึ่งสามารถโหลดข้อมูลรูปภาพตั้งแต่ระดับที่ 1 ถึง 5 มาได้ทั้งหมดโดยไม่ต้องแยก path ที่อยู่ของแต่ละระดับใหม่

ภาพที่ 3.8 จัดการข้อมูลรูปภาพด้วยไลบรารี่ image data generator

จากภาพที่ 3.8 จัดการข้อมูลรูปภาพด้วยไลบรารี่ image data generator ของ keras ปรับรูปภาพให้มีความหลากหลายโดยเข้ารหัสผลเฉลยแบบ one-hot encoding หมุนภาพไม่เกิน 40 องศา แบบสุ่มทำการเลื่อนภาพขึ้นลงแบบสุ่มไม่เกิน 20 เปอร์เซ็นต์ ทำการเลื่อนภาพซ้ายขวาแบบสุ่มไม่เกิน 20 เปอร์เซ็นต์ บิดภาพแบบสุ่มไม่เกิน 20 องศาขยายภาพแบบสุ่มไม่เกิน 30 เปอร์เซ็นต์ พลิกภาพแนวตั้งแบบสุ่มพลิกภาพแนวนอนแบบสุ่มเติมสีข้างเคียงขณะที่ทำกระบวนการทั้งหมดและแบ่งข้อมูลรูปภาพสำหรับตรวจสอบ 20 เปอร์เซ็นต์ ตัวอย่างข้อมูลรูปภาพบางส่วนที่ได้ทำกระบวนการดังกล่าว ดังภาพที่ 3.6

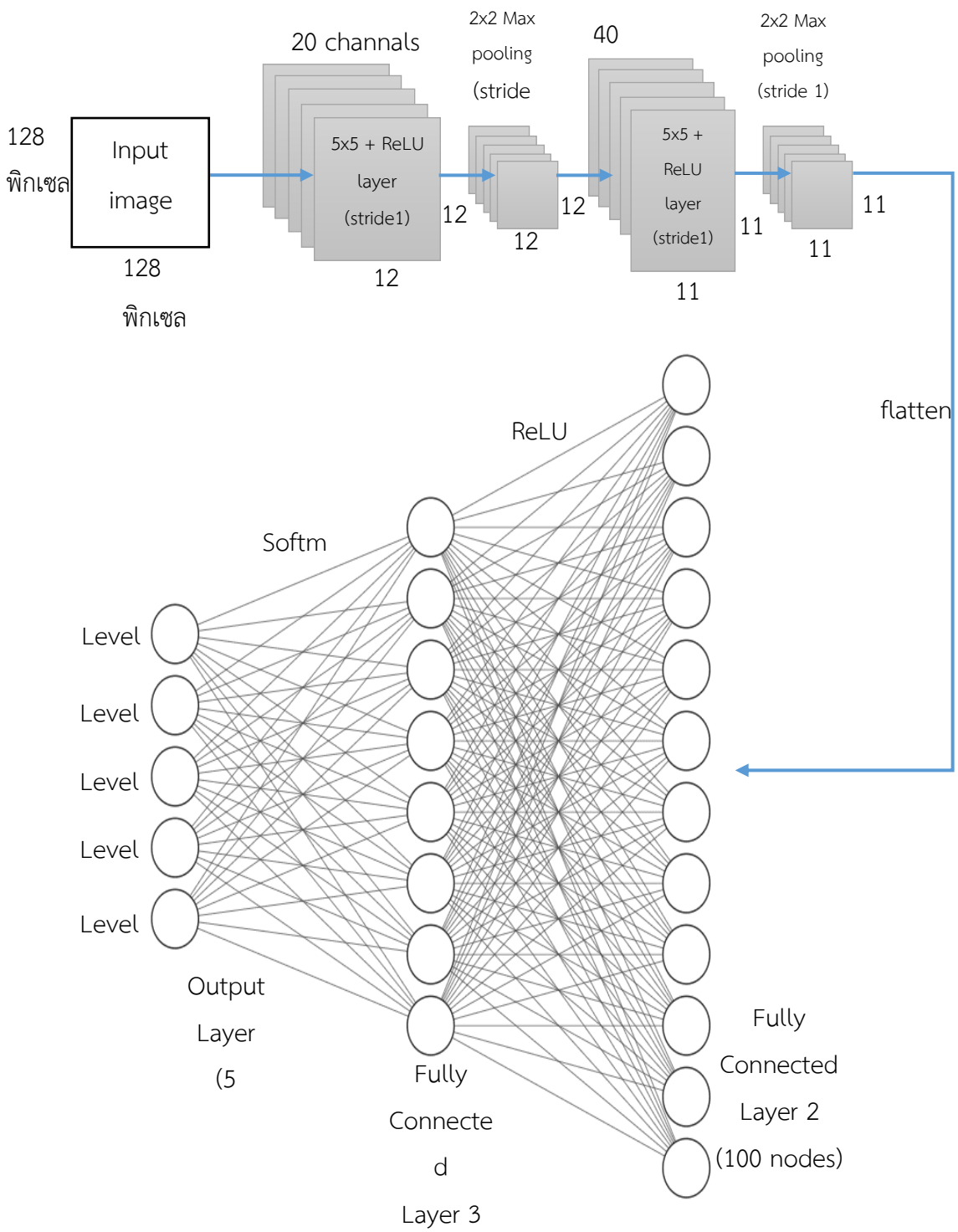
```
Found 516 images belonging to 5 classes.
Found 99 images belonging to 5 classes.
(array([[[[0.7411765 , 0.7490196 , 0.6666667 ],
          [0.74509805, 0.7411765 , 0.6627451 ],
          [0.7254902 , 0.7137255 , 0.6392157 ],
          ...,
          [0.77647066, 0.7686275 , 0.72156864],
          [0.7607844 , 0.75294125, 0.7019608 ],
          [0.7607844 , 0.7490196 , 0.6901961 ]],

         [[0.72156864, 0.7254902 , 0.6627451 ],
          [0.7568628 , 0.76470596, 0.68235296],
          [0.7490196 , 0.74509805, 0.6745098 ],
          ...,
          [0.7411765 , 0.7254902 , 0.6784314 ],
          [0.75294125, 0.74509805, 0.69411767],
          [0.75294125, 0.74509805, 0.6862745 ]],
```

ภาพที่ 3.9 แปลงข้อมูลรูปภาพให้อยู่ในรูปแบบ array

จากภาพที่ 3.9 แปลงข้อมูลรูปภาพให้อยู่ในรูปแบบ array กำหนดการตั้งค่ารูปภาพทำการปรับขนาดให้มีขนาดรูปภาพเป็น 16,384 พิกเซล เทรนครั้งละ 32 ภาพ โหมดสีของภาพคือ rgb และแบ่งออกเป็นสองชุดคือ training และ test ก่อนนำเข้าโมเดลเทรนนิ่ง

เริ่มทำการ deep learning แบบ convolutional neural network หลังจากเตรียมข้อมูลรูปภาพที่จะอินพุตเข้าไปไปเรียบร้อยแล้วขั้นตอนต่อไปคือการกำหนดชั้นของโมเดลซึ่งในขั้นตอนนี้เป็นขั้นตอนที่ละเอียดอ่อนเพราะเป็นส่วนสำคัญในการเรียนรู้ของโมเดลทางผู้วิจัยจึงทดลองหาค่าที่เหมาะสมในการตั่งค่าโมเดลให้สามารถเรียนรู้ระดับสับปะรด 5 ระดับได้อย่างมีประสิทธิภาพและสอดคล้องกับทรัพยากรของคอมพิวเตอร์ที่ทางผู้วิจัยมีไว้สำหรับสอนโมเดลดังกล่าว จึงทำการ import library ที่จำเป็นเริ่มจาก model ที่เป็นแบบ sequential และ layer ต่างๆ ได้แก่ Conv2D layer จะสร้าง feature map หรือ kernel ที่นำไปกับภาพที่เป็นอินพุตโดยจะสแกนไปให้ทั่วภาพ pooling layer ช่วยลดขนาดของเอาต์พุตที่ได้จาก layer ก่อนหน้าลง โดยคงไว้ซึ่งคุณสมบัติของข้อมูลให้มากที่สุด flatten layer ทำการแปลงข้อมูลเอาต์พุตที่มีหลายมิติ ให้เป็น 1 มิติ เพื่อเตรียมข้อมูลให้อยู่ใน format ที่พร้อมสำหรับเป็นอินพุตให้กับ Fully connected Layer ในส่วนสุดท้าย Dense layer หรือ Fully connected Layer ข้อมูลจากทุกๆ อินพุตจะเชื่อมต่อไปยังเอาต์พุตทุกๆ node โดยแต่ละการเชื่อมต่อจะคูณด้วย weight ที่ต่างกันและที่ทุกๆ node ของเอาต์พุตจะสามารถกำหนด activation ที่เหมาะสมได้

จากภาพที่ 3.10 architecture ของ model convolutional neural network ที่สร้างมี convolutional layer และ pooling layer 2 ชุด ตามด้วย flatten layer และ dense layer อีก 3 ชั้น โดยแต่ละ layer ที่มีคุณสมบัติดังนี้ กำหนดการตั้งค่าโมเดล ชั้นของ convolutional layer 1 รองรับอินพุตเป็นภาพสีขนาด 16,384 พิกเซล ให้เอาต์พุตออกมา 20 features โดยมี kernel ขนาด 5x5 stride 1 และใช้ activation function เป็น rectified linear unit ชั้นของ pooling layer 1 ใช้เป็น max pool ขนาด 2x2 stride 1 ชั้นของ convolutional layer 2 ให้เอาต์พุตออกมา 40 features โดยมี kernel ขนาด 5x5 stride 1 และใช้ activation function เป็น rectified linear unit ชั้นของ pooling layer 2 ใช้เป็น max pool ขนาด 2x2 stride 1 ชั้นของ flatten layer ทำการแปลงข้อมูล multi dimension ให้เป็น vector ชั้นของ dense layer 1 เป็น hidden layer กำหนดให้มีจำนวนเอาต์พุตเป็น 100 และใช้ activation function เป็น rectified linear unit ชั้น dense layer 2 เป็น hidden layer กำหนดให้มีจำนวนเอาต์พุตเป็น 50 และใช้ activation function เป็น rectified linear unit และชั้น dense layer 3 เป็น output layer ของ network กำหนดให้มีจำนวนเอาต์พุตเท่ากับจำนวนระดับสับปะรดกำหนดเป็น 5 และใช้ activation function เป็น soft max



ภาพที่ 3.10 Architecture ของ Model Convolutional Neural Network

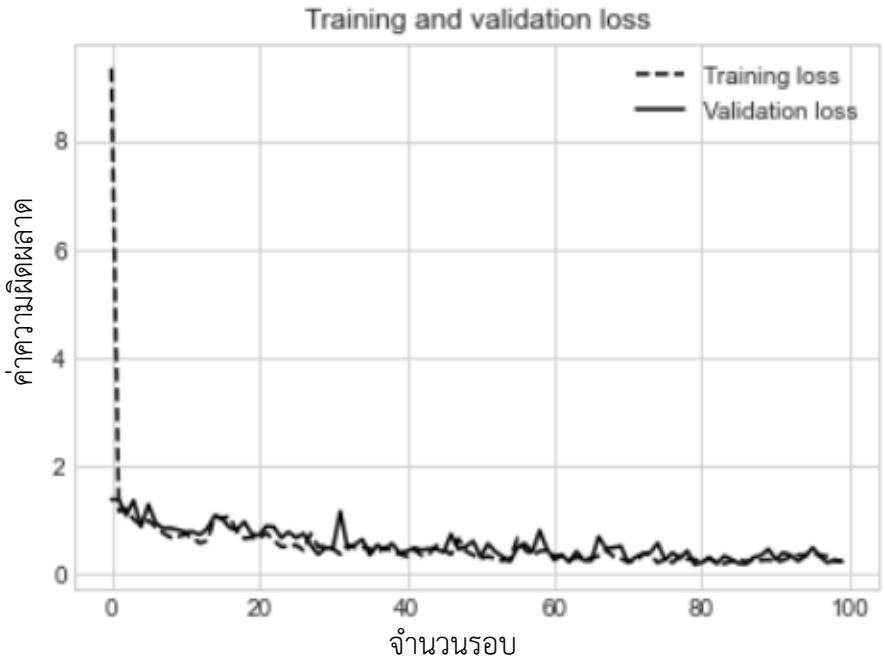
ก่อนนำข้อมูลมาสอนให้กับโมเดลให้ทำการรวบรวมข้อมูลของโมเดลอีกครั้งโดยใช้คำสั่ง model .compile() กำหนดให้โมเดลใช้ฟังก์ชั่น loss ชื่อ categorical crossentropy โดยโมเดลจะปรับค่าในขั้นตอนการ training เพื่อลดค่าของ loss ส่วนของ optimizer เป็นอีกตัวแปรที่เพิ่มประสิทธิภาพของโมเดลเลือกใช้อัลกอริทึม adam กำหนด metrics ให้โมเดลทำการคำนวณในระหว่าง training และ test ระบุเพิ่มคือ accuracy

เมื่อเตรียมข้อมูลรูปภาพสับปะรดและสร้างโมเดลเสร็จแล้วต่อไปคือการนำข้อมูลรูปภาพมาเข้าโมเดลในการ training โดยใช้คำสั่ง model.fit กำหนดการตั้งค่าคือข้อมูล training และข้อมูล test จำนวนในการ training คือ 100 รอบ

หลังจากที่ทำการ training ครบ 100 รอบแล้ว ทางผู้วิจัยได้นำไลบรารี่ matplotlib มาวาดกราฟประวัติการ training ทั้ง 100 รอบ ผลที่ได้ดังภาพที่ 3.11 และภาพที่ 3.12

ภาพที่ 3.11 กราฟแสดงค่าความแม่นยำการฝึกสอนโมเดล 100 รอบเสร็จสิ้น

จากภาพที่ 3.11 กราฟแสดงค่าความแม่นยำการฝึกสอนโมเดล 100 รอบเสร็จสิ้น เส้นปะคือค่า Training accuracy และเส้นธรรมดาคือค่า Validation accuracy ทั้งสองค่านี้คือค่าความแม่นยำซึ่งจะเพิ่มขึ้นเรื่อยๆ จนถึงร้อยละ90 ตั้งแต่รอบการฝึกสอนที่ 60 และไม่เพิ่มขึ้นอีกดังนั้นรอบการฝึกสอนที่ 100 จึงได้ค่าร้อยละ 90



ภาพที่ 3.12 กราฟแสดงค่าความผิดพลาดการฝึกสอนโมเดล 100 รอบเสร็จสิ้น

จากภาพที่ 3.12 กราฟแสดงค่าความผิดพลาดการเทรนโมเดล 100 รอบเสร็จสิ้น เส้นปะคือค่า Training loss และเส้นธรรมดาคือค่า Validation loss ทั้งสองค่านี้คือค่าความผิดพลาดซึ่งจะลดลงเรื่อยๆ จนถึงค่าที่ 0.182 ในรอบการฝึกสอนที่ 100

ก่อนนำโมเดลไปใช้งานให้ทำการจูนโมเดลอีกครั้งและคำนวณค่า accuracy โดยใช้คำสั่ง model.evaluate ทดสอบโมเดลพร้อมใช้งานหรือไม่ โดยใช้ข้อมูลทดสอบที่เตรียมไว้

```
_,acc=model.evaluate_generator(valid_generator,verbose=1)
print('>%.3f'%(acc*100.0))
```

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
4/4 [==============================] - 2s 415ms/step - loss: 0.1822 - accuracy: 0.9000
>90.000
```

ภาพที่ 3.13 ทดสอบความแม่นยำของโมเดลกับ Test set

จากภาพที่ 3.13 ทดสอบความแม่นยำของโมเดลกับ Test set ผลที่ได้คือความแม่นยำอยู่ที่ร้อยละ 90 ซึ่งสามารถนำไปใช้งานได้

3.4 การนำโมเดลไปใช้งาน

เนื่องจากการนำโมเดลของ tensorflow แบบธรรมดาไปใช้งานกับ raspberry pi 3.0 model b+ ไม่ได้ เพราะหน่วยความจำในการประมวลผลไม่เพียงพอ จึงได้นำ tensorflow lite มาปรับใช้งานโดย tensorflow lite คือเครื่องมือที่ช่วยให้สามารถรันโมเดล tensorflow ทำ inference บนมือถือ Android, iOS, อุปกรณ์ Edge, Raspberry Pi, Embedded และ Microcontroller ได้ โมเดลมีขนาดเล็กลงสามารถทำงานได้เร็วขึ้น โดยอาจจะลดความแม่นยำลงแต่ยังอยู่ในค่าที่ยอมรับได้ การใช้คำสั่ง convert() ให้เป็นโมเดล tensorflow lite เพื่อนำไปใช้งานกับ raspberry pi 3.0 model b+ ได้ดังภาพที่ 3.13

```
converter = tf.lite.TFLiteConverter.from_saved_model('model/pine5cDataGen.model')
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

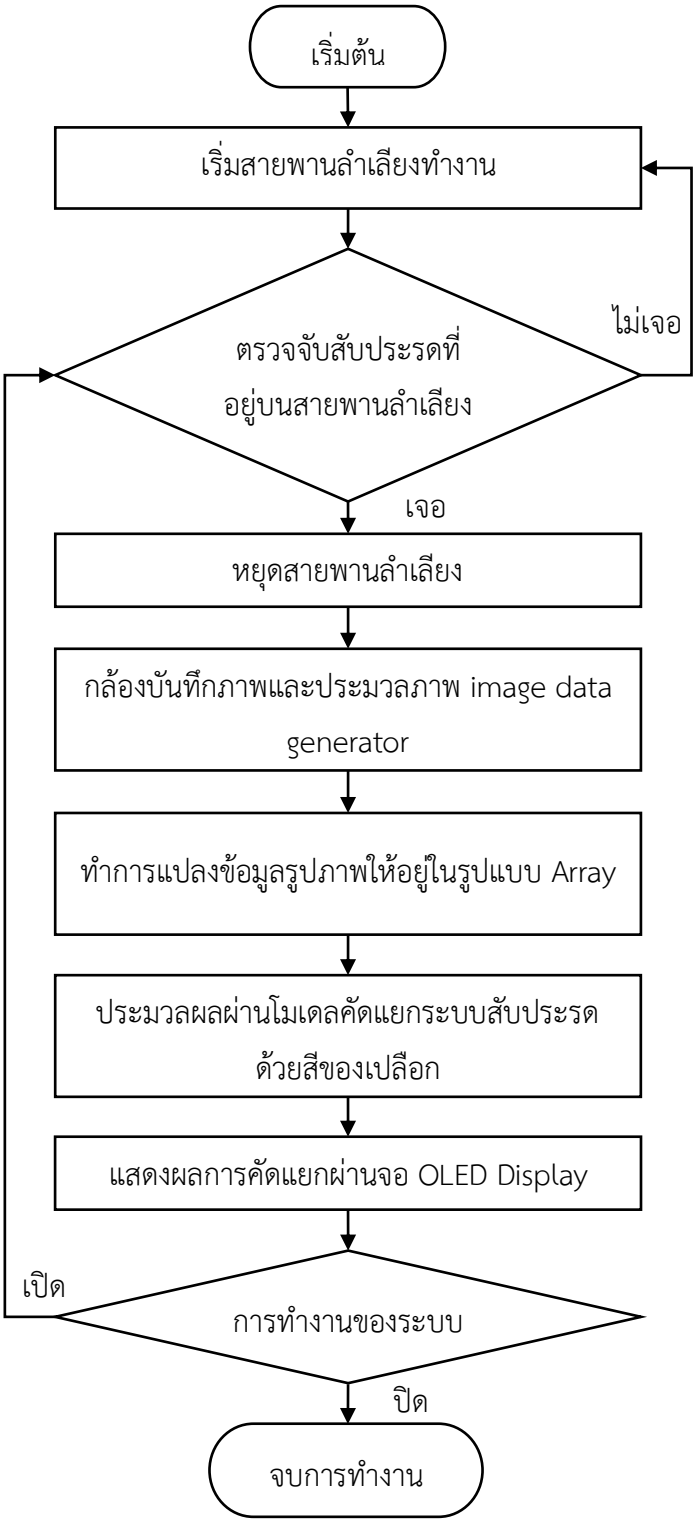
```
349159220
```

ภาพที่ 3.14 converter โมเดล tensorflow ให้เป็นโมเดล tensorflow lite

จากภาพที่ 3.14 converter โมเดล tensorflow ให้เป็นโมเดล tensorflow lite ทำการโหลดโมเดล tensorflow มาก่อนและแปลงเป็น tensorflow lite โดยตั้งชื่อคือ converted_model.tflite สังเกตได้ว่านามสกุลของไฟล์มีการเปลี่ยนจาก .model เป็น .tflite หลังจากที่ได้ทำกระบวนเสร็จสิ้นขนาดความจุของโมเดลจาก 800 MB เหลือเพียง 350 MB โดยประมาณ ซึ่งทางผู้วิจัยมีทรัพยากร raspberry pi 3 model b+ ขนาด ram เพียง 1 GB ไม่เพียงพอต่อการใช้งานสำหรับโมเดล tensorflow ดังนั้นทางผู้วิจัยจึงได้นำมา converter ก่อนนำไปใช้งานจริง

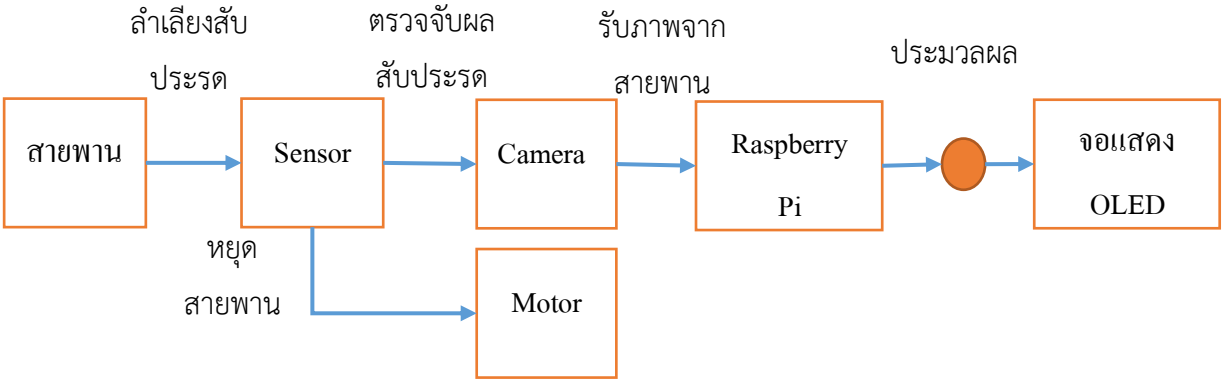
3.5 การออกแบบและพัฒนาเครื่องระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรด

การออกแบบและพัฒนาเครื่องระบบคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกสับปะรดนั้น ผู้วิจัยได้ทำการออกแบบระบบการทำงานเพื่อให้สามารถทำงานสอดคล้องกับตัวโมเดลได้อย่างมีประสิทธิภาพ ซึ่งผู้วิจัยได้แบ่งขั้นตอนการทำงานในส่วนต่างๆ ในการลำเลียงสับปะรดให้สอดคล้องกับระบบคัดแยกความสุกสับปะรดด้วยสีผิวของเปลือกแสดงดังภาพที่ 3.15



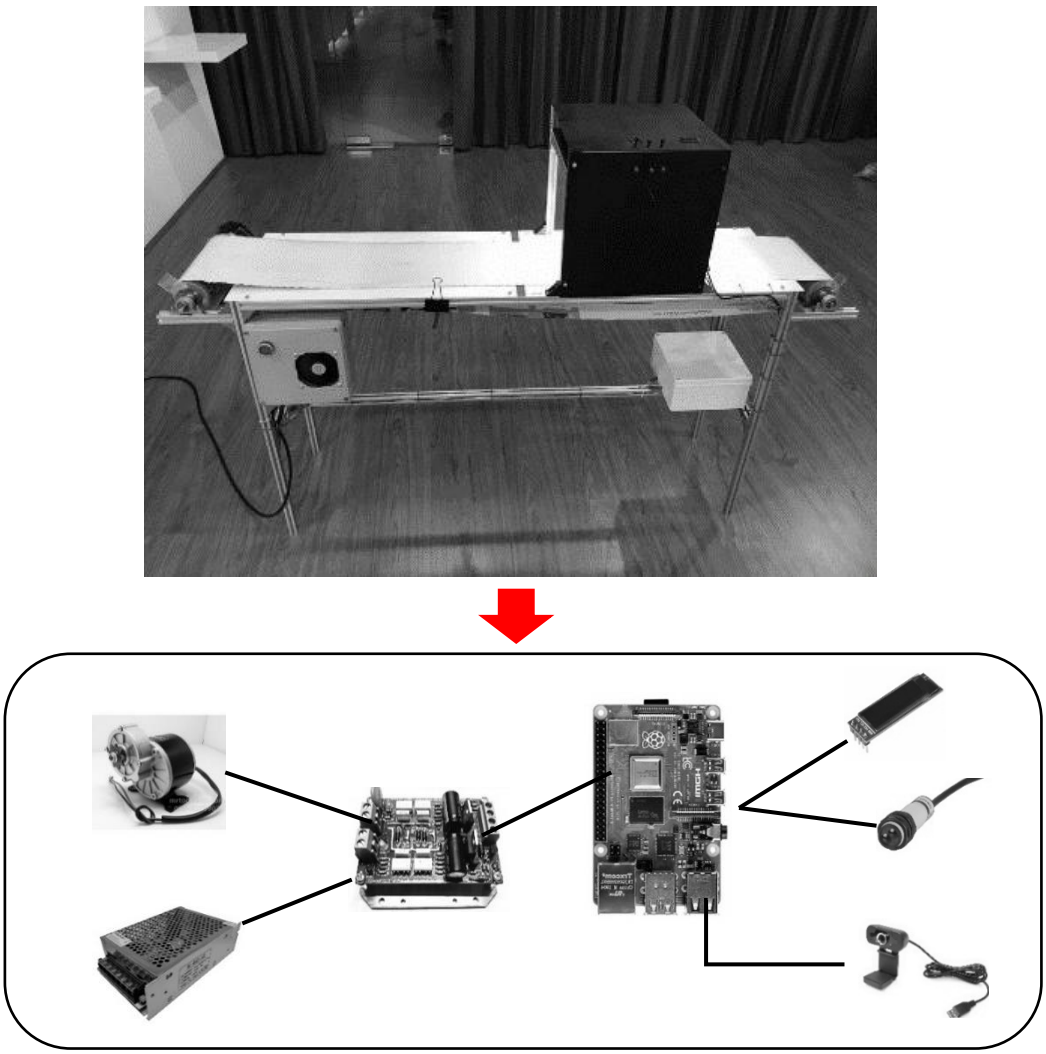
ภาพที่ 3.15 ขั้นตอนการทำงานของระบบคัดแยกความสุกของสับปะรดด้วยสีผิวของเปลือก

จากภาพที่ 3.15 ขั้นตอนการทำงานของระบบคัดแยกความสุกของสับปะรดด้วยสีผิวของเปลือกเมื่อระบบเริ่มการทำงานสายพานลำเลียงจะเริ่มทำงานเรื่อยๆ จนกระทั้งระบบคัดแยกตรวจเจอผลสับปะรดในกล่องบันทึกภาพ ระบบจะหยุดสายพานลำเลียงและทำการบันทึกภาพจากกล้องดิจิตอลในตัวกล่องโดยนำข้อมูลภาพจากการบันทึกเข้ามาประมวลผลภาพ image data generator หลังจากประมวลผลภาพเสร็จสิ้นทำการแปลงข้อมูลภาพให้อยู่ในรูปแบบของข้อมูลตัวเลข Array ก่อนนำเข้าไปประมวลผลผ่านตัวโมเดลระบบการคัดแยกระดับสับปะรดด้วยสีผิวของเปลือกเมื่อการประมวลผลเสร็จสิ้นระบบจะระบุผลการคัดแยกระดับสับปะรดเมื่อมีผลสับปะรดใหม่เข้ามายังกล่องถ่ายภาพและระบบตรวจเจอก็จะทำการประมวลผลต่อไป



ภาพที่ 3.16 การออกแบบระบบคัดแยกความสุกดิบของสับปะรดด้วยลักษณะสีของเปลือก

จากภาพที่ 3.16 การออกแบบระบบคัดแยกความสุกดิบของสับปะรดด้วยลักษณะสีของเปลือกมีส่วนประกอบดังนี้ สายพานทำหน้าที่ลำเลียงสับปะรดให้เคลื่อนไปยังจุดคัดแยก sensor ตรวจจับผลสับปะรดที่เข้ามายังจุดคัดแยก motor เป็นตัวควบคุมสายพานให้หยุดหรือเลื่อน camera ถ่ายบันทึกภาพสับปะรดจากจุดคัดแยกเพื่อนนำไปประมวลผล raspberry pi ทำหน้าที่เป็นสมองของระบบเป็นตัวควบคุมและประมวลผลการทำงานทั้งหมด จอแสดง OLED แสดงผลการคัดแยกสับปะรดจากทั้งหมด 5 ระดับด้วยกันและบอกสถานะอื่นๆ ในระบบ



ภาพที่ 3.17 แผนภาพแสดงฮาร์ดแวร์ของระบบคัดแยกความสุกดิบของสับปะรด
ด้วยลักษณะสีของเปลือก

จากภาพที่ 3.17 แผนภาพแสดงฮาร์ดแวร์ของระบบคัดแยกความสุกดิบของสับปะรดด้วยลักษณะสีของเปลือก จากภาพกล้อง webcam เชื่อมต่อกับราสเบอร์รี่พายผ่านยูเอสบีพอร์ต (USB Port) เซนเซอร์ตรวจจับวัตถุ จอแสดงผล OLED และไดร์มอเตอร์เชื่อมต่อกับราสเบอร์รี่พายผ่านจีพีไอโอพอร์ต (GPIO Port) มีการส่งสัญญาณ PWM ไปยังวงจรขับเคลื่อนมอเตอร์เพื่อควบคุมความเร็วรอบมอเตอร์และส่งสัญญาณส่งออกของดิจิตอลไปที่ช่อง IN1 ของตัวขับมอเตอร์เพื่อควบคุมการหมุนของมอเตอร์โดยมีอแดปเตอร์ 220v เป็น 12v เป็นตัวจ่ายไฟให้กับมอเตอร์